Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

“Інформаційна технологія прогнозування та мінімізації інцидентів під час управління IT-послугами”

Виконав: студент 2 курсу, групи 2ІСТ-24м
спеціальності 126 – «Інформаційні системи та технології»
_____ Владислав ПОБІДАШ

Керівник: к.т.н., доц. каф. САІТ
_____ Євгеній КРИЖАНОВСЬКИЙ
«27» ____11_____ 2025 р.

Опонент: к.т.н., доц. каф. КН
_____ Володимир ОЗЕРАНСЬКИЙ
«03» ____12_____ 2025 р.

Допущено до захисту
Завідувач кафедри САІТ
_____ д.т.н., проф. Віталій МОКІН
«25» ____11_____ 2025 р.

Вінниця ВНТУ – 2025 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій
Рівень вищої освіти – другий (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 126 Інформаційні системи та технології
Освітньо-професійна програма – Інформаційні технології аналізу даних та зображень

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
_____ д.т.н., проф. Віталій МОКІН
«25» 09 2025 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Побідашу Владиславу Віталійовичу

1. Тема роботи: "Інформаційна технологія прогнозування та мінімізації інцидентів під час управління IT-послугами",
керівник роботи: Євгеній КРИЖАНОВСЬКИЙ, к.т.н., доц. каф. САІТ,
затверджені наказом ВНТУ від «24» 09 2025 року №313
2. Строк подання студентом роботи «28» 11 2025 року
3. Вихідні дані до роботи:
Набір даних наданий компанією Sysaid, який містить сервісні записи IT відділу.
4. Зміст текстової частини:
- аналіз предметної області;
- вибір оптимальних інформаційних технологій;
- розробка інформаційної технології;
- економічна частина.
5. Перелік ілюстративного матеріалу:
- структурні схеми;
- візуалізація даних та результатів.

6. Консультанти розділів МКР

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4	Олександр ЛЕСЬКО, к. е. н., проф. каф. ЕПВМ	05.12.25	15.12.2025

7. Дата видачі завдання «25» 09 2025 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Аналіз предметної області	15.09	30.09	вик
2	Вибір оптимальних інформаційних технологій	30.09	17.10	вик
3	Розробка інформаційної технології	17.10	05.11	вик
4	Економічна частина	05.11	15.11	вик
5	Оформлення матеріалів до захисту МКР	15.11	25.11	вик

Студент _____ Владислав ПОБІДАШ

Керівник роботи _____ Євгеній КРИЖАНОВСЬ

АНОТАЦІЯ

УДК 004.94:502.3(477.44)

Побідаш В.В. Інформаційна технологія прогнозування та мінімізації інцидентів під час управління ІТ-послугами. Магістерська кваліфікаційна робота зі спеціальності 126 – інформаційні системи та технології, освітньо-професійна програма – інформаційні технології аналізу даних та зображень. Вінниця: ВНТУ, 2025. 132 с.

На укр. мові. Бібліогр.: 21 назва; рис.: 23; табл.: 6.

В магістерській кваліфікаційній роботі досліджено технологія та методи прогнозування обсягів сервісних записів, виявлення аномалій, генерації проактивних рекомендацій та запропоновано технології автоматизованого прогнозування піків навантаження, виявлення повторюваних проблем та генерації структурованих проактивних рекомендацій для підвищенні якості ІТ-обслуговування, оптимізації ресурсного планування та зниженні операційних витрат на технічну підтримку. Об’єктом досліджень є процес управління інцидентами у системах управління ІТ-сервісами.

Ілюстративна частина складається з 11 плакатів із результатами аналізу сервісних записів ІТ-відділу.

У розділі економічної частини розглянуто питання про доцільність розробки та впровадження інформаційної технології прогнозування та мінімізації інцидентів під час управління ІТ-послугами.

Ключові слова: прогнозування інцидентів, управління ІТ-сервісами, ITSM, машинне навчання, аналіз часових рядів, виявлення аномалій, оптимізація ресурсів.

ABSTRACT

Pobidash V. V. Information technology for predicting and minimizing incidents during IT service management. Master's thesis in specialty 126 - information systems and technologies, educational and professional program - information technology data and image analysis. Vinnytsia: VNTU, 2025. 132 p.

In Ukrainian language. Bibliogr .: 21 titles; fig .: 23; table: 6.

The master's thesis examines technologies and methods for forecasting service request volumes, detecting anomalies, and generating proactive recommendations. It proposes technologies for automated peak load forecasting, detecting recurring problems, and generating structured proactive recommendations to improve IT service quality, optimizing resource planning, and reducing operational costs for technical support. The object of research is the process of incident management in IT service management systems.

The illustrative part consists of 11 posters with the results of analysis of IT department service records.

The economic section examines the feasibility of developing and implementing information technology for forecasting and minimizing incidents during IT service management.

Key words: incident forecasting, IT service management, ITSM, machine learning, time series analysis, anomaly detection, resource optimization.

ЗМІСТ

ВСТУП

Актуальність теми. Сучасні організації стикаються з критичною потребою трансформації традиційної реактивної моделі управління IT-інцідентами у проактивну парадигму, яка базується на прогнозній аналітиці та превентивних заходах. Розробка технології автоматизованого прогнозування піків навантаження, виявлення повторюваних проблем та генерації структурованих проактивних рекомендацій відповідає на гостру потребу організацій у підвищенні якості IT-обслуговування, оптимізації ресурсного планування та зниженні операційних витрат на технічну підтримку в умовах зростаючої цифровізації бізнес-процесів та критичної залежності організаційної ефективності від стабільності IT-інфраструктури, де навіть невелике покращення у передбачуваності проблем може призвести до суттєвого зменшення простоїв систем та підвищення продуктивності користувачів.

Мета і завдання роботи. Метою дослідження є підвищення точності прогнозування виникнення інцидентів під час управління IT-послугами за рахунок використання методів машинного навчання. За допомогою спеціалізованого програмного комплексу здійснюється аналіз історичних даних сервісних записів та автоматична генерація проактивних заходів для мінімізації майбутніх інцидентів.

Розробка технології прогнозування та мінімізації інцидентів передбачає виконання наступних задач:

- здійснити розвідувальний аналіз даних сервісних записів з системи управління IT-сервісами за визначений період;
- вибрати оптимальні методи аналізу часових рядів і машинного навчання;
- побудувати ансамблеві прогнозні моделі на основі SARIMA, Хольта-Вінтерса та базової персистентності;
- реалізувати виявлення аномалій та генерацію проактивних рекомендацій з пріоритизацією заходів.

Об’єктом дослідження магістерської кваліфікаційної роботи є процес управління інцидентами у системах управління IT-сервісами.

Предметом дослідження магістерської кваліфікаційної роботи є технологія та методи прогнозування обсягів сервісних записів, виявлення аномалій та генерації проактивних рекомендацій.

Методи дослідження. У дослідженнях використовувались методи аналізу часових рядів для моделювання темпоральних патернів навантаження, зокрема модель SARIMA та Prophet. Для виявлення аномалій застосовано алгоритм Isolation Forest. Реалізація виконана мовою програмування Python з використанням бібліотек pandas, scikit-learn, statsmodels для обробки даних та побудови моделей.

Новизна одержаних результатів. Дістала подальший розвиток інформаційна технологія прогнозування інцидентів у системах управління IT-сервісами за рахунок ансамблевого підходу та підвищує точність прогнозування виникнення інцидентів під час управління IT-послугами.

Практичне значення сприятиме покращенню якості управління IT-сервісами в організаціях. Впровадження технології дозволяє знизити кількість реактивних звернень на 12-15%, покращити дотримання угод про рівень обслуговування та підвищити ефективність ресурсного планування. Результати роботи мають цінність для служб технічної підтримки організацій середнього та великого розміру.

Публікації результатів магістерської кваліфікаційної роботи. Опубліковано тези на LV Всеукраїнській науково-технічній конференції підрозділів Вінницького національного технічного університету (2026) [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальна характеристика управління IT‑послугами (ITSM)

Управління послугами інформаційних технологій (ITSM) – це практика планування, впровадження, управління та оптимізації надання комплексних послуг інформаційних технологій для задоволення потреб користувачів та досягнення бізнес-цілей [2]. На відміну від традиційного підходу, орієнтованого на управління окремими IT-ресурсами, ITSM розглядає IT як сукупність послуг, що мають чітко визначену цінність для бізнесу та кінцевого користувача. Основний акцент робиться не на технологіях як таких, а на якості сервісу, стабільності процесів і передбачуваності результатів їх виконання.

Систематично вдосконалюючи процеси обробки запитів на обслуговування, IT-підтримку, управління IT-активами та управління змінами, ITSM допомагає підприємствам поліпшити користувацький досвід та підвищити продуктивність IT-інфраструктури. ITSM також допомагає організаціям реалізовувати бізнес-стратегії, дотримуватися нормативних та організаційних вимог і зменшувати ризики шляхом впровадження засобів контролю в процес проектування, надання та управління IT-послугами. Завдяки стандартизованим процедурам підвищується прозорість взаємодії між IT-підрозділом та іншими бізнес-одиницями, спрощується планування навантаження, бюджетування та управління ризиками.

Основна мета ITSM – забезпечити оптимальне розгортання, експлуатацію та управління кожним IT-ресурсом для кожного користувача в межах підприємства. До користувачів належать клієнти, співробітники або бізнес-партнери. IT-ресурси включають апаратне забезпечення, програмне забезпечення та інші обчислювальні активи, такі як ноутбуки, програмні додатки, хмарні сховища та віртуальні сервери. У контексті сучасних гібридних та хмарних інфраструктур ITSM також охоплює зовнішні сервіси (SaaS, PaaS,

IaaS), взаємодію з постачальниками та управління їхньою відповідністю узгодженим рівням сервісу.

ISO/IEC 20000 – це міжнародний стандарт ITSM. Він дозволяє IT-відділам забезпечити відповідність процесів ITSM потребам бізнесу та найкращим міжнародним практикам [3]. Стандарт визначає вимоги до системи управління послугами, описує життєвий цикл послуги, процеси планування та надання, а також вимоги до постійного поліпшення. Використання ISO/IEC 20000 дає змогу організаціям проводити аудити, формувати формалізовані показники якості сервісів і порівнювати їх з галузевими орієнтирами.

Існує широкий спектр програмних рішень, процесів та керівних рамок для впровадження ITSM, включаючи бібліотеку IT-інфраструктури (ITIL), COBIT, а також підходи DevOps та Site Reliability Engineering (SRE) [4, 5]. Бібліотека ITIL 4 розглядається як де-факто стандарт кращих практик у галузі ITSM, вона описує концепцію сервісної цінності, чотири виміри управління послугами та сервісну вартісну систему [4]. COBIT фокусується на корпоративному управлінні IT та контролях, доповнюючи ITSM-підхід вимогами до відповідності, ризик-менеджменту та аудиту [5].

Стандарт ISO 20000 допомагає організаціям проводити порівняльний аналіз надання керованих послуг, вимірювати рівні обслуговування та оцінювати їхню ефективність. Він в цілому узгоджується з ITIL і значною мірою базується на його процесній моделі [3, 4]. Завдяки цьому компанії можуть будувати інтегровану систему управління, в якій вимоги стандарту, процеси ITIL та внутрішні регламенти взаємодоповнюють одне одного.

ITSM спирається на програмні засоби, автоматизацію та перевірені процедури. Якщо клієнт звертається до служби підтримки, щоб повідомити про проблему з комп'ютерною робочою станцією, запросити нову ліцензію або отримати доступ до програмного забезпечення, ITSM визначає послідовність дій і керує робочим процесом, який дозволить виконати ці запити. У сучасних сервіс-деск платформах (ServiceNow, Jira Service Management, BMC Remedy

тощо) ці робочі процеси реалізуються у вигляді електронних заявок, черг, SLA-таймерів, автоматичних сповіщень та інтеграцій з іншими системами [6].

Деякі організації розширюють свої можливості ITSM, включаючи управління послугами підприємства (ESM), яке зосереджується на більш широких бізнес-потребах конкретних команд, відділів або підрозділів. ESM поширює принципи ITSM на HR, фінанси, юридичний відділ, логістику та інші служби, що дозволяє будувати єдині каталоги послуг, єдині портали самообслуговування та уніфіковані процеси обробки запитів [5, 6]. ITSM також є перспективним – воно продовжує акцентувати увагу на постійному поліпшенні досвіду користувачів або клієнтів, використовуючи зворотний зв'язок, опитування задоволеності (CSAT), індекс лояльності (NPS) та інші показники.

ITSM – це набагато більше, ніж реагування, ремонт і підтримка – це цілісний погляд і план управління ресурсами та процесами інформаційних технологій. Процеси ITSM розроблені для взаємодії з іншими підрозділами компанії з метою досягнення стратегічних і операційних цілей організації. Вони охоплюють як «операційний» рівень (щоденна підтримка, моніторинг, реагування на інциденти), так і «тактичний» та «стратегічний» рівні (планування потужностей, управління портфелем послуг, фінансове управління IT) [4, 5].

В рамках ITSM ключову роль відіграють такі процеси, як управління інцидентами, управління проблемами, управління змінами, управління конфігураціями, управління запитами на послуги, управління знаннями, управління рівнем послуг, а також управління IT-активами (ITAM). Сукупність цих процесів створює основу для прогнозування та мінімізації інцидентів, оскільки дозволяє збирати структуровані дані про події, причини відмов, зміни в інфраструктурі та ефективність дій служби підтримки [6, 7]. Узагальнену схему взаємодії основних процесів ITSM наведено на рисунку 1.1.

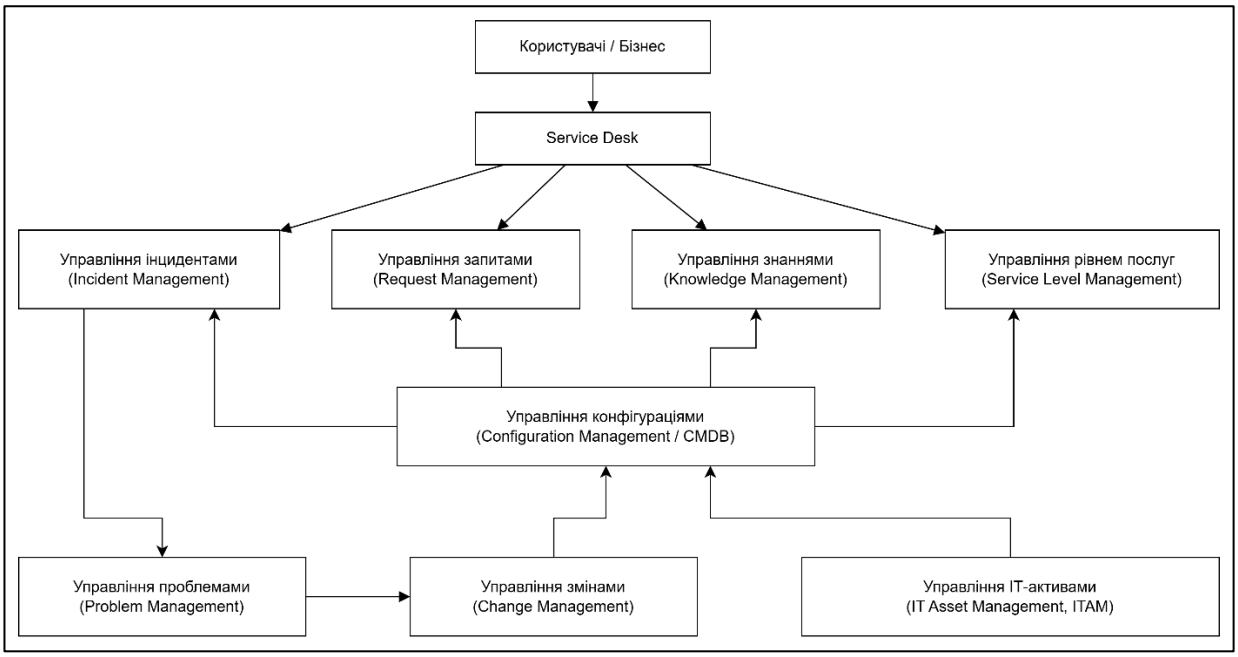

Рисунок 1.1 – Узагальнена схема взаємодії основних процесів ITSM

В ITSM інцидент – це незапланованe відключення або переривання обслуговування, а також будь-яке погіршення якості послуги (наприклад, зниження продуктивності, збільшення затримок, часткове недоступність функціоналу). Управління інцидентами – це процес реагування на інцидент з метою відновлення послуги з мінімальним впливом на користувачів та бізнес-процеси. Типовий життєвий цикл інциденту включає реєстрацію, класифікацію, пріоритизацію, первинну діагностику, ескалацію (за потреби), вирішення та закриття з фіксацією фактичного результату [4, 6]. Спрощений приклад життєвого циклу інциденту та його взаємодії з управлінням проблемами й змінами подано на рисунку 1.2.

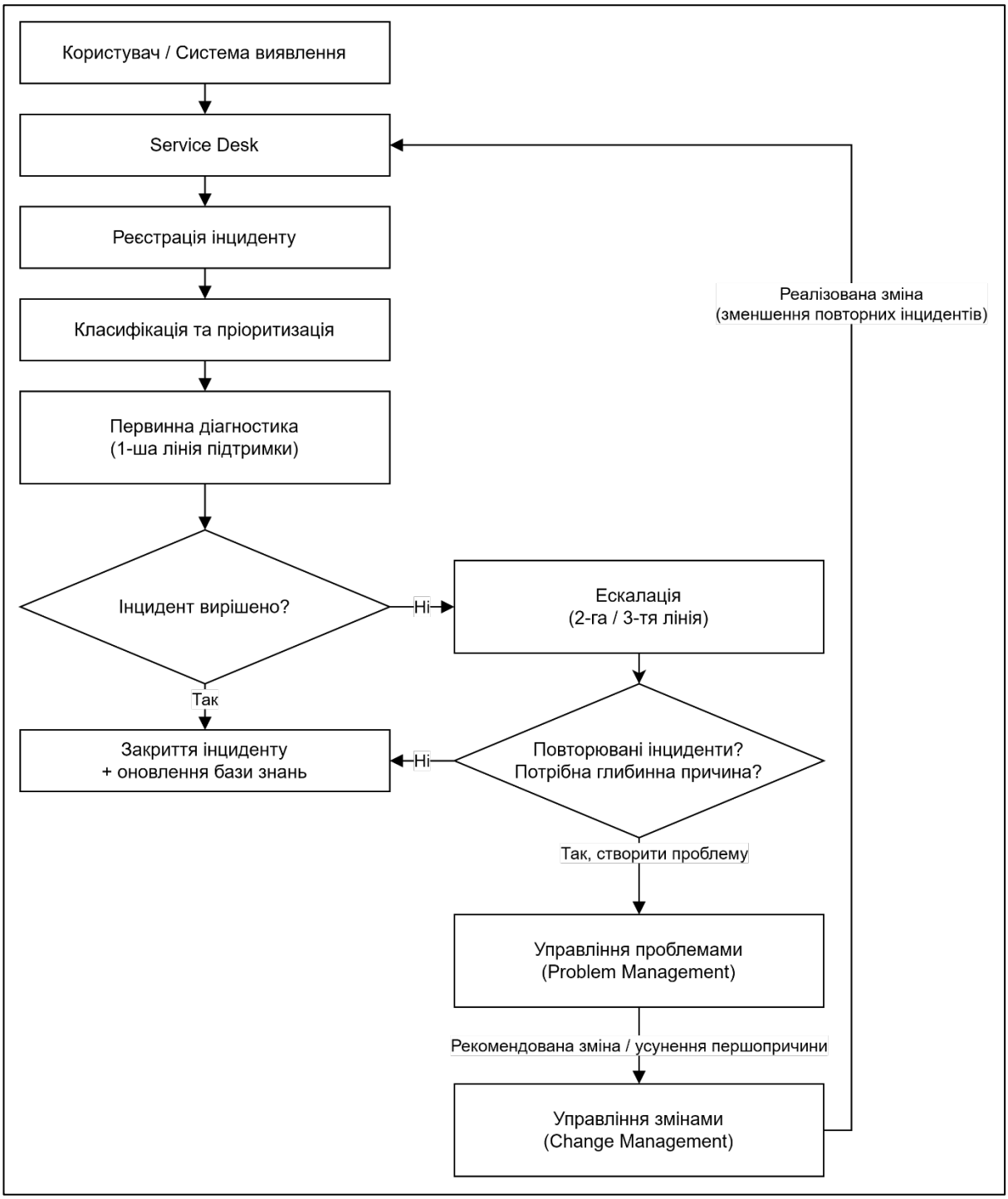

Рисунок 1.2 – Спрощений життєвий цикл інциденту та його взаємодія з управлінням проблемами й змінами

Для інцидентів зазвичай визначаються цільові значення ключових показників ефективності, таких як середній час до реагування (MTTA), середній час до відновлення (MTTR), відсоток інцидентів, вирішених на першій лінії підтримки (First Call Resolution, FCR), та відсоток інцидентів, закритих у межах

погоджених SLA [6, 7]. Аналіз цих метрик у динаміці дозволяє виявляти тренди, наприклад зростання кількості інцидентів певного типу або збільшення середнього часу вирішення, що є підґрунтям для переходу від реактивного до проактивного управління інцидентами.

Управління проблемами відбувається, коли кілька інцидентів пов'язані з однією й тією ж першопричиною. Метою процесу є виявлення, аналіз та усунення кореневих причин, щоб інциденти більше не повторювалися [4]. Для цього використовують методи аналізу першопричин (Root Cause Analysis, RCA), діаграми Ішікави (Fishbone), метод «5 чому», побудову хронології подій, кореляцію інцидентів із внесеними змінами, оновленнями чи перевантаженнями інфраструктури [4]. Результати управління проблемами фіксуються у базі відомих помилок (Known Error Database), що спрощує подальше діагностування та дозволяє оперативно знаходити обхідні рішення.

Управління змінами (change management, у ITIL 4 – change enablement) – це встановлення протоколів для мінімізації перебоїв в IT-послугах, проблем із дотриманням вимог та інших ризиків, які можуть виникнути в результаті змін, внесених до критично важливих систем. Зміни класифікуються за рівнем ризику (стандартні, нормальні, аварійні), проходять оцінку впливу, погодження з зацікавленими сторонами та, у разі значних ризиків, – розгляд на засіданні комітету з управління змінами (CAB) [4, 5]. Якісне управління змінами є одним з ключових факторів мінімізації інцидентів, оскільки значна частина відмов пов'язана саме з неконтрольованими чи погано спланованими змінами.

Управління конфігурацією – це процес відстеження елементів конфігурації (Configuration Items, CI) для апаратних і програмних компонентів, а також їхніх взаємозв'язків. Такий інструмент, як база даних управління конфігурацією (Configuration Management Database, CMDB), слугує центральним сховищем усіх IT-активів та залежностей між ними [4]. Завдяки CMDB служба підтримки може швидко оцінити, які сервіси постраждають у разі відмови конкретного сервера, мережевого пристрою чи програмного компонента, а також проаналізувати, які зміни виконувалися над цими об'єктами до виникнення інциденту [7].

Запити на послуги щодо нових активів, дозволів або ліцензій можуть надходити від співробітників, клієнтів або партнерів. Управління запитами на послуги визначає найефективніший і найточніший метод задоволення або відхилення цих запитів, часто за допомогою поєднання автоматизації та самообслуговування. Для цього використовуються стандартизовані шаблони заявок, автоматичні маршрутизації до відповідальних груп, попередньо погоджені строки виконання (SLA) та інтеграція з системами управління доступами та закупівлями [4, 6].

Каталог послуг – це довідник, який інтегрується з системою управління запитами на послуги. Доступ до нього здійснюється через меню або портал самообслуговування, і в ньому перелічені IT-послуги, доступні користувачам у всій організації, з описом їхніх характеристик, рівнів підтримки та вимог до безпеки [4, 5]. Наявність прозорого каталогу послуг зменшує кількість неструктурованих звернень, спрощує для користувача вибір потрібної послуги та покращує керованість сервісного портфеля.

Управління знаннями – це процес ідентифікації, організації, зберігання та поширення інформації в організації. Основним інструментом KM є база знань із можливістю пошуку та самообслуговування. Вона надає користувачам в організації легкий доступ до питань та рішень, пов'язаних з IT-послугами, метрик, документації, технічних тем та інших ресурсів [4]. Систематичне ведення бази знань дозволяє зменшити час вирішення інцидентів, підвищити відсоток рішень на першій лінії підтримки та забезпечити накопичення експертного досвіду, який надалі може використовуватися як навчальна вибірка для алгоритмів машинного навчання.

Управління рівнем послуг – це процес створення, відстеження та адміністрування життєвого циклу угоди про рівень послуг (Service Level Agreement, SLA). SLA – це договір між постачальником послуг та клієнтом, який визначає рівень послуг, що надаються, метрики їхнього вимірювання (наприклад, доступність, час відгуку, час відновлення) та наслідки недотримання цих значень [3, 6]. Поряд із SLA можуть встановлюватися внутрішні операційні

угоди (Operational Level Agreement, OLA) між командами всередині організації, що забезпечують досягнення загальних цільових показників.

В ITSM служба підтримки IT є центральним пунктом контакту (Single Point of Contact, SPOC) для обробки та управління всіма інцидентами, проблемами та запитами. Деякі служби підтримки виконують розширені функції: займаються ліцензуванням програмного забезпечення, управлінням постачальниками послуг, ціноутворенням та контрактами з третіми сторонами, а також підтримують портали самообслуговування та бази знань. Важливу роль відіграє багаторівнева модель підтримки (1-ша, 2-га, 3-тя лінії), що дозволяє раціонально розподіляти ресурси й залучати експертів різного рівня складності [6, 7].

Управління IT-активами, або IT Asset Management (ITAM) – це процес забезпечення повного відстеження, актуальності та працездатності активів організації. Ці активи можуть включати апаратне забезпечення, таке як ноутбуки та монітори, а також нефізичні активи, такі як ліцензії на програмне забезпечення, підписки на хмарні сервіси, сертифікати безпеки тощо [5, 7]. Дуже важливо, щоб усі ці активи були зібрані в централізованому IT-відділі, щоб уникнути надмірності, «сірих» IT-рішень, несанкціонованих закупівель і неефективності, а також щоб забезпечити відповідність вимогам ліцензування та інформаційної безпеки.

1.2 Дані, показники та проактивне управління інцидентами в ITSM

Важливим аспектом сучасного ITSM є перехід від суто реактивного управління (коли служба підтримки лише реагує на вже зареєстровані інциденти) до проактивного та предиктивного підходів. У традиційній, реактивній моделі основна увага приділяється якнайшвидшому відновленню працездатності сервісу після інциденту, тоді як аналіз причин, закономірностей і попередження повторних збоїв часто залишаються поза фокусом. У межах кращих практик ITIL та суміжних фреймворків наголошується, що така модель є недостатньою для складних, динамічних IT-ландшафтів, де зростання навантаження, поява нових

сервісів і часті зміни призводять до експоненційного збільшення кількості звернень [4, 8]. Саме тому у зрілих організаціях акцент поступово зміщується на проактивне та прогнозне управління інцидентами, що передбачає систематичне використання даних та показників для запобігання збоям і зменшення їхнього впливу на бізнес.

Проактивне управління інцидентами ґрунтується на безперервному моніторингу інфраструктури, аналізі журналів подій, виявленні трендів зростання навантаження, аналізі повторюваних інцидентів і проблем, плануванні потужностей та управлінні ризиками [8]. На практиці це означає впровадження комплексу технічних і процесних засобів: систем спостережуваності (observability) для збору телеметрії (метрики, логи, трасування), механізмів оповіщення за пороговими значеннями та аномаліями, регулярних оглядів показників (service review) з участю власників сервісів. Дані про інциденти, проблеми та зміни аналізуються не лише постфактум, а й у динаміці, з фокусом на виявленні ранніх сигналів погіршення якості послуг: зростання часу відгуку, збільшення кількості дрібних, але повторюваних збоїв, накопичення незакритих заявок тощо. Таким чином, проактивний підхід поєднує оперативний моніторинг, формалізовані процеси проблем-менеджменту та управління потужностями з аналітикою, орієнтованою на попередження, а не лише на реакцію [4, 8].

Предиктивний підхід передбачає використання аналітики даних, машинного навчання та AIOps-платформ для прогнозування ймовірності виникнення інцидентів на основі історичних даних, аномалій у поведінці систем і зовнішніх факторів [9, 10]. На відміну від класичного моніторингу, який фіксує вже наявне відхилення від норми, прогнозні моделі дозволяють заздалегідь оцінити ризик інцидентів для конкретних сервісів, часових інтервалів або типів змін і, відповідно, вжити превентивних заходів (перерозподіл навантаження, зміни в конфігурації, відкладення ризикованих релізів). AIOps-рішення інтегрують дані з різних джерел – ITSM-систем, систем моніторингу, платформ лог-менеджменту – та застосовують алгоритми машинного навчання для

кореляції подій, автоматичного виявлення аномалій і генерації рекомендацій для команд підтримки [8–10]. Узагальнену концептуальну схему використання даних ITSM для побудови моделей прогнозування інцидентів наведено на рисунку 1.3.

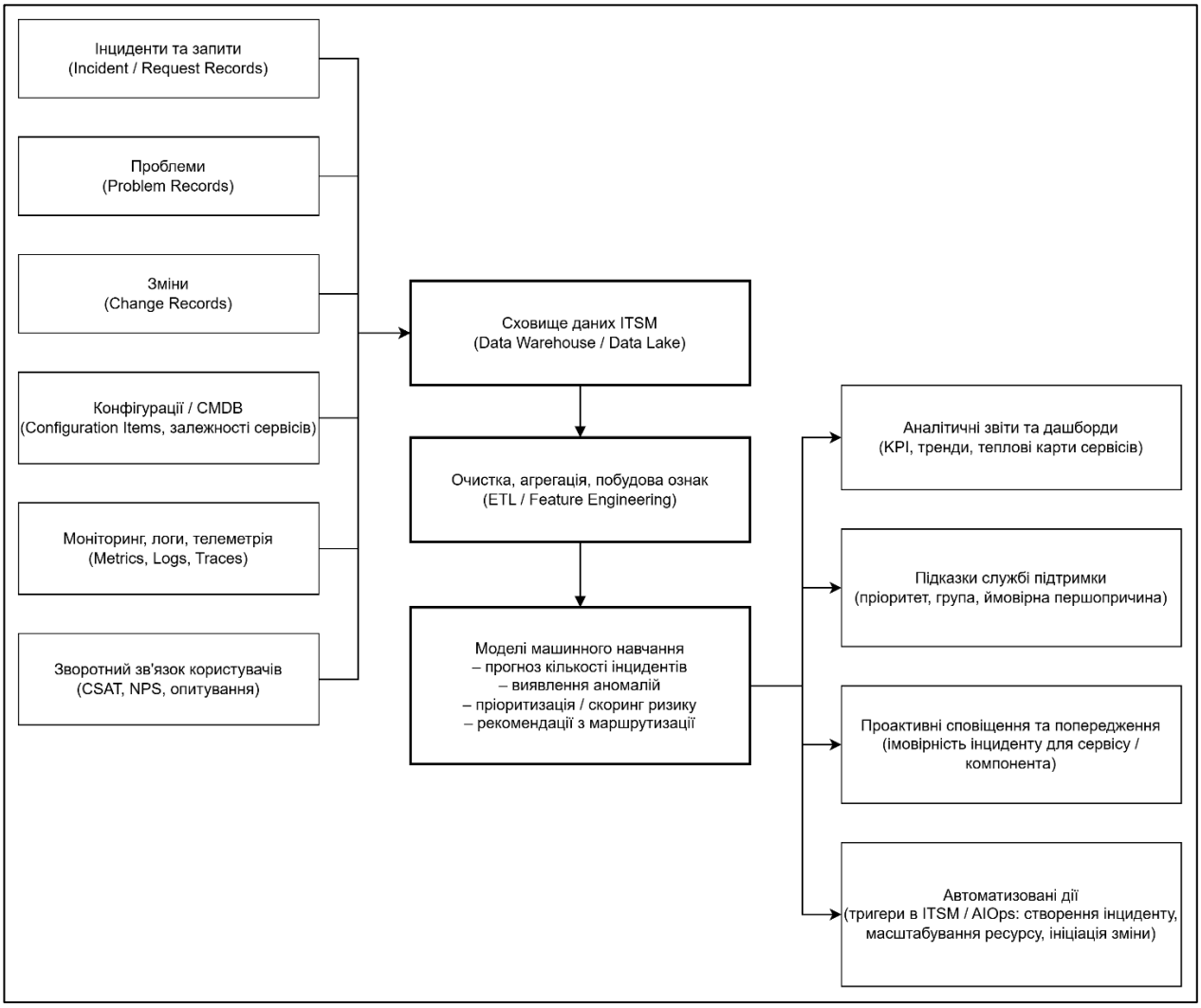

Рисунок 1.3 – Узагальнена концептуальна схема використання даних ITSM для побудови моделей прогнозування інцидентів.

У контексті розробки технологій прогнозування та мінімізації інцидентів ключове значення має якість та повнота даних, що накопичуються в ITSM-системі. Це дані про інциденти (класи, категорії, пріоритети, причини, час реагування та вирішення), проблеми, зміни, елементи конфігурації, навантаження на інфраструктуру, користувацькі запити та результати опитувань

задоволеності [8–10]. До цього набору також належать інформація про структуру сервісів (каталог послуг), домовленості про рівень сервісу (SLA), дані про ескалації, тимчасові обхідні рішення (workaround) та відомі помилки (known errors). На основі цих даних можуть будуватися моделі прогнозування кількості інцидентів, оцінки ризику виникнення інцидентів у певних сервісах, ранжування інцидентів за критичністю чи ймовірністю ескалації, а також рекомендаційні системи для вибору оптимального способу їх вирішення. Якість таких моделей безпосередньо визначається повнотою історії, коректністю заповнення атрибутів і ступенем стандартизації довідників у межах ITSM-процесів [7–10].

Суттєву роль у проактивному та предиктивному управлінні інцидентами відіграють показники ефективності (KPI) та індикатори рівня сервісу (SLI), які визначаються в угодах SLA [4, 8]. Для процесу управління інцидентами типовими метриками є середній час реакції (MTTA), середній час відновлення (MTTR), частка інцидентів, вирішених при першому зверненні (FCR), кількість інцидентів на одиницю часу, відсоток порушених SLA, частота повторних інцидентів тощо. Для управління проблемами – кількість відкритих і закритих проблем, середній час до виявлення та усунення кореневої причини. Для змін – відсоток змін, що спричинили інциденти, частка термінових та незапланованих змін. Сукупність цих показників дозволяє не лише оцінювати поточний стан сервісів, а й будувати моделі, які передбачають ризик деградації якості послуг у майбутньому, зокрема за умови зростання навантаження або запланованих архітектурних змін [4, 8–10].

З позиції аналітики даних важливо відрізняти «відстаючі» показники (lagging indicators), що характеризують уже здійснені події (наприклад, факт порушення SLA або завершення інциденту), від «випереджальних» (leading indicators), які сигналізують про потенційні проблеми: збільшення середнього часу обробки нових заявок, накопичення черги на першій лінії підтримки, зростання кількості дрібних попереджувальних подій у журналах моніторингу [8]. Саме випереджальні показники частіше використовуються як вхідні дані для прогнозних моделей, оскільки вони дозволяють виявляти ситуації, що з високою

ймовірністю призведуть до інцидентів у найближчому майбутньому. Поєднання цих двох типів показників у єдиній аналітичній моделі забезпечує і ретроспективний, і проактивний погляд на стан IT-сервісів.

Окремий клас даних для задач прогнозування – це інформація про зміни (Change Management) та конфігураційні елементи (CMDB), які задають контекст, у межах якого виникають інциденти [4, 8]. Кожна зміна має атрибути, що описують її тип (стандартна, нормальна, термінова), ступінь ризику, пов'язані сервіси й CI, вікно реалізації, відповідальних осіб, результат впровадження. На основі поєднання історії змін та історії інцидентів можна виявити шаблони: які типи змін частіше призводять до збоїв, які комбінації сервісів і інфраструктурних компонентів є найбільш вразливими, як впливають на надійність зміни в пікові періоди навантаження [8–10]. CMDB, у свою чергу, надає інформацію про залежності між сервісами, апаратними й програмними компонентами, мережевою інфраструктурою, що дозволяє прогнозним моделям враховувати каскадний характер відмов та поширення інцидентів між взаємопов'язаними елементами.

Сучасні дослідження демонструють ефективність використання методів машинного навчання – таких як класифікація, регресія, кластеризація, виявлення аномалій та обробка природної мови (NLP) – для аналізу текстів заявок, історії інцидентів і телеметрії систем [9, 10]. Наприклад, моделі класифікації можуть автоматично визначати категорію та пріоритет інциденту, а також імовірність його ескалації на вищий рівень підтримки; регресійні моделі – оцінювати очікуваний час вирішення на основі атрибутів інциденту, завантаженості команд та історичних патернів; методи кластеризації – виділяти групи однотипних інцидентів, що свідчать про приховані проблеми в інфраструктурі [9, 10]. NLP-моделі дозволяють витягувати ключові симптоми з вільного тексту опису користувача, уніфіковувати різні формулювання схожих проблем і створювати ознаки (features) для подальшого прогнозування. Алгоритми виявлення аномалій та аналіз часових рядів, у свою чергу, застосовуються до телеметричних даних (метрики продуктивності, логи запитів, показники доступності) для

передбачення пікових навантажень та ймовірності відмов [8–10]. Отримані прогнози можуть використовуватися для завчасного масштабування ресурсів, планування змін, підсилення зміни операторів підтримки або автоматизованого застосування обхідних рішень.

Таким чином, ITSM виступає комплексною предметною областю, що поєднує процесний підхід до управління IT-послугами, використання міжнародних стандартів та кращих практик, а також сучасні технології моніторингу й аналізу даних. Для задачі «Розробка технології прогнозування та мінімізації інцидентів» ITSM надає як теоретичну основу (процеси, ролі, показники ефективності), так і практичний масив даних, необхідних для навчання моделей прогнозування та оцінки їхнього впливу на реальні бізнес-процеси [4, 8–10]. Розвиток цієї галузі тісно пов'язаний із подальшою інтеграцією ITSM-систем з AIOps-рішеннями, інструментами спостережуваності (observability), а також з усе більшою автоматизацією рутинних операцій у службах підтримки. У наступних підрозділах ці аспекти будуть деталізовані з урахуванням зрілості процесів ITSM та вимог до якості даних, необхідних для побудови надійних прогнозних моделей [4, 7–10].

1.3 Зрілість процесів ITSM та готовність організації до впровадження прогнозних технологій

Ефективність застосування проактивних і прогнозних підходів в ITSM безпосередньо залежить від зрілості процесів управління послугами. Міжнародні фреймворки ITIL та COBIT розглядають управління IT як систему взаємопов’язаних процесів, для яких можна оцінювати рівень зрілості – від хаотичних, слабо формалізованих практик до стабільних, керованих і постійно вдосконалюваних процесів [4, 5]. У спрощеному вигляді зазвичай виділяють п’ять рівнів зрілості: початковий (ad hoc), повторюваний, визначений (defined), керований (managed) та оптимізований (optimizing) [4]. Кожен із цих рівнів характеризується специфічним ступенем формалізації процесів, ролей і

відповідальностей, підходів до вимірювання результатів роботи та використання даних для прийняття рішень. Відповідно, можливості організації щодо впровадження прогнозних технологій також змінюються разом із переходом від одного рівня зрілості до іншого.

На початковому рівні інциденти та запити обробляються переважно реактивно, без єдиних регламентів і чіткої відповідальності; дані фіксуються частково або зовсім не структуруються. Рішення приймаються на основі індивідуального досвіду окремих спеціалістів, а не стандартизованих процедур, що унеможливлює побудову надійних моделей машинного навчання: історія звернень фрагментарна, відсутня уніфікована класифікація інцидентів, атрибути заповнюються нерівномірно, а багато дій залишаються «поза системою». На наступному, повторюваному рівні організація починає впроваджувати базові шаблони обробки типових звернень, однак вони ще не завжди формалізовані у вигляді затверджених політик і процедур, що призводить до варіативності в підходах різних команд або змін. Лише під кінець цього рівня з’являються перші спроби систематичного збору статистики, однак її якість і повнота залишаються недостатніми для надійного прогнозування.

Перехід до визначеного (defined) рівня пов’язаний із формалізацією та документуванням ключових процесів ITSM [4]. З’являються стандартизовані категорії інцидентів, описи ролей та обов’язків, регламенти ескалації, каталог послуг та формалізовані угоди про рівень сервісу (SLA). Для кожного типу звернень визначаються очікувані терміни реакції та відновлення, затверджуються схеми маршрутизації, порядок комунікації з користувачами й бізнес-замовниками. Важливо, що саме на цьому рівні організації зазвичай починають серйозніше ставитися до повноти й коректності внесення даних до сервіс-деск-системи, оскільки від цього залежить прозорість звітності та можливість обґрунтовувати інвестиції в IT. Таким чином, створюється базис для подальшого використання історичних даних у задачах прогнозування навантаження, ризику інцидентів чи ймовірності порушення SLA.

На керованому (managed) рівні організація вже систематично вимірює ключові показники ефективності (KPI) процесів: середній час реакції (MTTA), середній час відновлення (MTTR), частку інцидентів, вирішених при першому зверненні (FCR), відсоток інцидентів, пов’язаних зі змінами, частоту повторних інцидентів тощо [4]. Впроваджуються регулярні огляди показників на рівні керівництва IT та власників процесів; результати аналізу метрик використовуються для планування ресурсів, оптимізації графіків чергувань, удосконалення процедур зміни й релізів. Процеси не лише описані, а й фактично виконуються відповідно до затверджених регламентів, а відхилення фіксуються та аналізуються. На цьому етапі організація накопичує достатній масив даних стабільної якості, що дозволяє починати пілотні ініціативи з описової та діагностичної аналітики, а надалі – і з прогнозування.

Оптимізований (optimizing) рівень передбачає постійне вдосконалення процесів на основі аналізу даних, застосування автоматизації, а також впровадження кращих практик управління послугами [4, 5]. IT-підрозділ використовує не лише агреговані KPI, а й більш складні аналітичні моделі для виявлення аномалій, пошуку кореневих причин (root cause analysis), виявлення закономірностей між змінами, навантаженням і виникненням інцидентів. У межах цього рівня природним чином виникає потреба в проактивних та прогнозних механізмах: наприклад, у моделюванні ймовірності інцидентів під час планування змін, в оцінюванні ризику перевантаження окремих сервісів, у визначенні «гарячих точок» інфраструктури, що потребують додаткового моніторингу або резервування. Технології машинного навчання та AIOps розглядаються вже не як окремі експерименти, а як невід’ємна частина керованих, зрілих процесів [8–10].

З погляду готовності до прогнозування інцидентів організація має досягти щонайменше «визначеного» рівня зрілості. Це означає, що: усі інциденти, зміни та проблеми реєструються в єдиній системі; використовуються узгоджені довідники категорій, пріоритетів, типів інцидентів і змін; CMDB містить актуальну інформацію про ключові конфігураційні елементи (CI) та їх зв’язки з

бізнес-послугами; регулярний моніторинг забезпечує потік телеметричних даних (журнали подій, метрики продуктивності, дані про доступність); SLA та інші метрики визначені й послідовно вимірюються [4, 7]. За відсутності таких умов навчені моделі машинного навчання будуть ґрунтуватися на неповних або неконсистентних даних, що знижує точність прогнозів, ускладнює інтерпретацію результатів і зменшує довіру з боку бізнесу та IT-керівництва [8–10].

Важливим аспектом зрілості є якість даних ITSM, яка охоплює повноту, точність, узгодженість, своєчасність та достатню історичну глибину [7]. Для побудови моделей прогнозування необхідно, щоб у записах інцидентів і змін коректно та систематично заповнювалися атрибути (категорія, сервіс, CI, першопричина, тип зміни, вплив, пріоритет, час реакції та відновлення), а в CMDB відображалися ключові залежності між сервісами та інфраструктурними компонентами [7]. Недостатня деталізація або непослідовне використання довідників призводять до «шуму» в даних: однакові типи подій класифікуються по-різному, окремі поля залишаються порожніми, а текстові описи не піддаються простої формалізації. Накопичена за кілька років історія, навпаки, дозволяє виявити сезонність (наприклад, пікові навантаження на кінець місяця чи кварталу), тренди (зростання кількості інцидентів певного типу після змін в архітектурі) та кореляції між навантаженням, змінами та інцидентами, що є основою для якісних моделей машинного навчання [8–10].

Окремий вимір зрілості – організаційна та культурна готовність до використання аналітики й ML у прийнятті рішень. Для цього необхідно визначити ролі власників процесів і сервісів (Process Owner, Service Owner), відповідальних за якість даних (Data Steward), а також узгодити, як саме результати моделей впливають на реальні дії: зміну пріоритетів опрацювання інцидентів, планування змін, ініціювання додаткового моніторингу, перегляд SLA або параметрів масштабування [4, 5]. Не менш важливим є питання довіри: команди підтримки мають розуміти логіку використання прогнозів, а керівництво – бачити вимірюваний ефект (зменшення кількості критичних

інцидентів, скорочення MTTR, зниження кількості порушень SLA). Без такого узгодження існує ризик, що прогнози залишаться «цікавими звітами», які не змінюють поведінку команди підтримки та не призводять до зниження кількості інцидентів. У крайніх випадках це може спричинити відкат до суто реактивної моделі роботи, коли рекомендації системи ігноруються як «надто складні» або «ненадійні».

Впровадження прогнозних технологій в організаціях із недостатньо зрілими процесами може мати зворотний ефект. За відсутності стабільних процесів та якісних даних моделі починають відтворювати випадкові або історично зумовлені упередження (наприклад, надмірну ескалацію до певних команд, неточну пріоритизацію, хибні уявлення про залежності між сервісами), а автоматизовані дії – підсилювати неефективні практики. Замість зниження кількості інцидентів організація може зіткнутися з їхнім зростанням через неправильну маршрутизацію або неадекватну оцінку ризиків змін. Тому більшість рекомендацій з побудови AIOps- та predictive-ITSM-рішень передбачають поетапний підхід: спочатку – стабілізація та стандартизація процесів, очищення й уніфікація довідників, визначення відповідальних за якість даних; потім – базова описова аналітика (dashboards, KPI) для прозорості поточного стану; далі – пілотні проекти з прогнозування на окремих сервісах чи процесах із відносно високою зрілістю; і лише після успішної валідації – інтеграція прогнозів у робочі процеси та часткова автоматизація реакцій [8–10].

Таким чином, зрілість процесів ITSM, якість даних та організаційна готовність до використання аналітики виступають ключовими передумовами для успішного впровадження технологій прогнозування та мінімізації інцидентів [4, 7–10]. У рамках подальших розділів роботи доцільно врахувати ці обмеження при формуванні вимог до вхідних даних, виборі алгоритмів машинного навчання та оцінці ефективності запропонованої технології у реальних умовах експлуатації IT-сервісів, а також при розробці рекомендацій щодо підвищення зрілості процесів ITSM з урахуванням можливостей прогнозної аналітики.

1.4 Огляд існуючих інформаційних систем з AI-функціоналом

Аналіз існуючих рішень є ключовим етапом у розробці інформаційної технології, оскільки він дозволяє виявити оптимальні практики та функціональні можливості, уникнути повторного винаходження велосипеда, ідентифікувати слабкі сторони та помилки інших технологій для їх уникнення, а також врахувати користувацькі потреби та успішно впроваджені рішення на ринку.

Сучасні ITSM-системи еволюціонують у напрямку інтелектуальних платформ, де ключову роль відіграє штучний інтелект із функціями прогнозування, аналітики та проактивного управління. Основні вимоги до ITSM нового покоління включають:

- інтеграцію з AI для передбачення інцидентів і попередження їх ескалації;
- автоматичне визначення пріоритетів і розподіл завдань;
- аналітику на основі історичних даних для покращення процесів обслуговування;
- зручну візуалізацію та централізоване керування AI-агентами.

Freshservice – це сучасна ITSM-платформа, побудована компанією Freshworks, яка орієнтована на простоту використання, масштабованість і швидке впровадження. Система підтримує IT та бізнес-команди, забезпечуючи єдину платформу для управління сервісами, активами (через інтеграцію з Device42) і комунікацією в омніканальному середовищі (email, Slack, MS Teams). Окремі модулі відповідають за каталог послуг, CMDB, управління змінами та релізами, а також за побудову автоматизованих робочих процесів без залучення розробників.

AI-функціонал реалізовано через Freddy AI, який включає декілька компонентів: Freddy Copilot (чат-бот для агентів), Freddy Insights (аналітична панель із попереджувальними інсайтами) та Freddy Agent (попередньо створені AI-агенти). Ці інструменти надають проактивні підказки, автоматизують робочі процеси, допомагають прогнозувати можливі проблеми та оптимізують роботу

команди підтримки, наприклад, автоматично заповнюючи поля заявки чи пропонуючи рішення на основі подібних інцидентів.

Halo ITSM позиціонує себе як «платформу без компромісів» із повною автоматизацією робочих процесів. Вона поєднує функціонал ITSM, PSA та CRM у єдиному середовищі з загальними ядрами (CMDB, аналітика, автоматизація, AI, білінг). Це рішення часто обирають середні та великі компанії завдяки гнучкій структурі ліцензування та можливості інтеграції з іншими бізнес-системами (ERP, HRM, фінансові системи). Важливою перевагою є вбудований конструктор workflow-схем, який дозволяє моделювати складні погодження, ескалації та залежності між запитами, проєктами й контрактами без глибокого програмування.

Щодо AI – функціонал існує, але залишається базовим і спрямованим на підвищення продуктивності, а не на глибоке передбачення. Хоча Halo ITSM пропонує AI "out-of-the-box" (пошук рішень, підказки, автозаповнення), основний фокус все ще на автоматизації ITIL-процесів і зниженні ручної праці, а не на побудові повноцінних моделей прогнозування інцидентів. Водночас наявність єдиного джерела даних (єдиний дашборд сервісів, завдань і фінансів) створює хорошу основу для подальшої інтеграції зовнішніх аналітичних або ML-рішень.

Atlassian об’єднує Dev, IT та бізнес-команди в рамках Jira Service Management, роблячи акцент на сучасному підході до ITSM і глибокій інтеграції з процесами розробки. Система підтримує управління інцидентами, запитами та змінами, має готові шаблони для різних департаментів (IT, HR, фінанси, юридичний відділ), а також тісно інтегрується з Jira Software, Confluence і Opsgenie. Це дозволяє будувати наскрізні ланцюжки від користувацького інциденту до задач розробки, тестування та релізу, що особливо важливо в DevOps-та SRE-підходах. На рисунку 1.4 представлено типовий вигляд черги заявок і панелі агента.

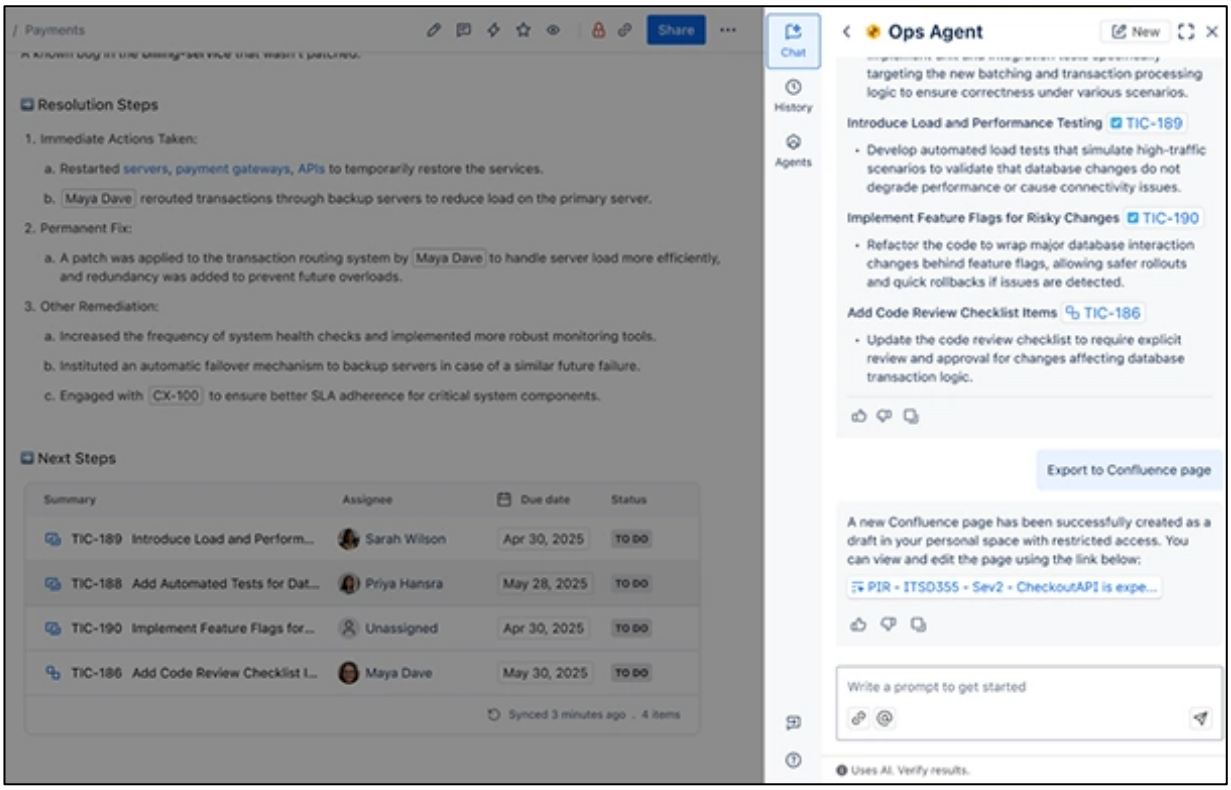

Рисунок 1.4 – Типовий вигляд черги заявок в Jira Service Management і панелі агента Rovo AI

AI-компонент Atlassian побудований на новій платформі Rovo AI, яка забезпечує розумний пошук, чат, аналітику та створення кастомних агентів для різних бізнес-напрямків. Rovo дозволяє створювати інтелектуальні AI-агенти з власними сценаріями для IT Ops, безпеки, продажів тощо, які можуть «читати» документацію в Confluence, історію задач у Jira та пропонувати готові відповіді або наступні дії. Такий підхід сприяє проактивному аналізу запитів та інцидентів, скорочуючи час реакції та підвищуючи ефективність підтримки, хоча точність рекомендацій усе ще сильно залежить від якості наявного контенту та структури даних.

ServiceNow залишається лідером ринку ITSM завдяки своїй AI-платформі ServiceNow AI, яка об’єднує дані, робочі процеси та інтелектуальні агенти. Вона автоматизує ключові процеси – інциденти, проблеми, зміни, запити – та дозволяє створювати власних AI-агентів без коду через AI Agent Studio. Окремий модуль Predictive Intelligence виконує автоматичну класифікацію й маршрутизацію

заявок, виявлення подібних інцидентів і рекомендацію рішень, а Virtual Agent забезпечує діалогове самообслуговування користувачів у чаті.

Ключові інновації включають AI Control Tower (панель для контролю життєвого циклу AI-рішень), AI Agent Orchestrator (координація взаємодії між кількома агентами) та AI Agent Fabric (об'єднання даних і AI-агентів з різних платформ). Завдяки цим інструментам ServiceNow забезпечує прогнозування інцидентів, аналіз першопричин і автоматичну ескалацію, що значно підвищує надійність IT-сервісів, хоча впровадження таких рішень потребує суттєвих інвестицій у налаштування процесів і даних. Приклад дашборда Incident Management у ServiceNow представлено на рисунку 1.5.

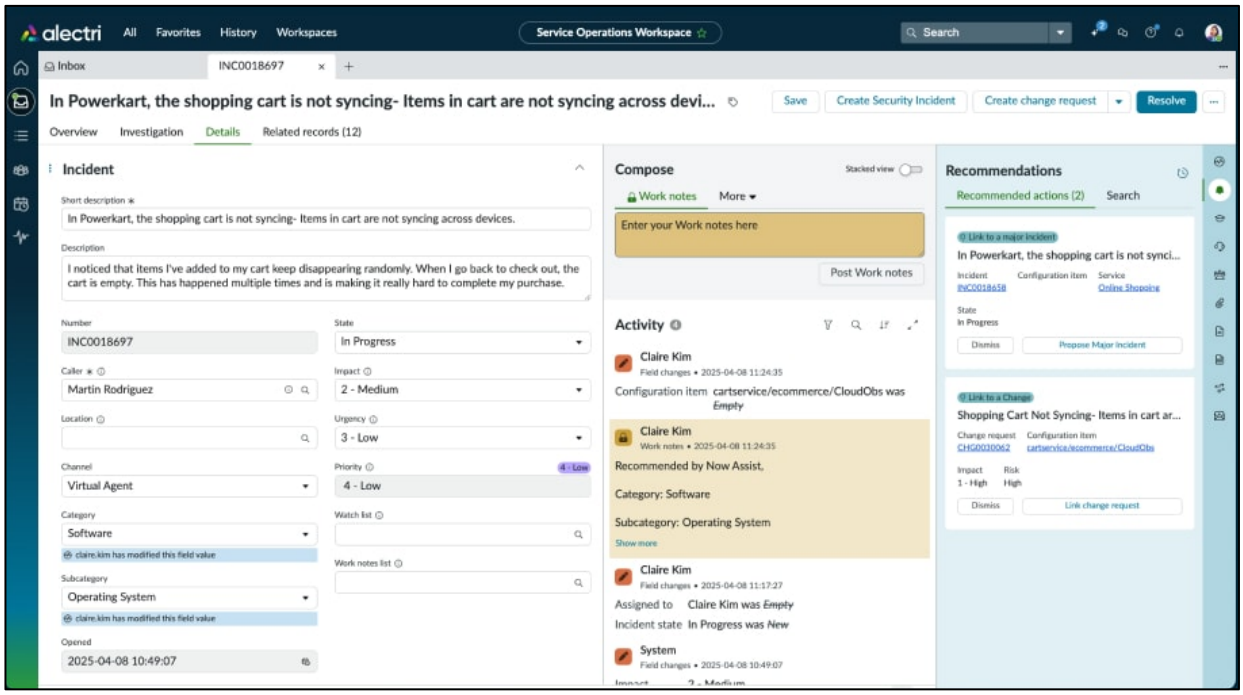

Рисунок 1.5 – Приклад дашборда Incident Management у ServiceNow

ManageEngine ServiceDesk Plus – хмарна ITSM-платформа, орієнтована на швидке розгортання та простоту підтримки. Вона має покращений інтерфейс, розширене управління активами, підтримку CMDB, каталог послуг та інтеграції з Azure й іншими системами. Завдяки вбудованим правилам автоматизації (розподіл за категоріями, пріоритетами, групами підтримки) і шаблонам робочих

процесів платформа добре підходить для невеликих і середніх IT-організацій, де важливі простота налаштування та низький поріг входу.

AI-функції реалізовані через інтеграцію з Azure OpenAI, що дозволяє системі генерувати відповіді на запити, створювати контент (опис інцидентів, статті бази знань), виправляти граматику, прогнозувати схвалення заявок і формувати рішення з підказок користувачів. Це забезпечує базову проактивність і суттєво спрощує роботу агентів, проте система більше фокусується на контекстній допомозі та підвищенні продуктивності, ніж на повноцінному передбаченні інцидентів на рівні всієї інфраструктури.

SysAid – це комплексне ITSM-рішення, орієнтоване на автоматизацію сервісних процесів і швидку адаптацію під потреби бізнесу. Платформа має модулі для управління інцидентами, проблемами, змінами, запитами, а також інтеграцію з інвентаризацією активів та HR-процесами. Широко використовуються вбудовані «правила автоматизації», які дозволяють будувати тригери на основі полів заявки, часу, навантаження тощо, без написання коду, а також централізований механізм оркестрації дій (створення, оновлення, ескалація, зміна статусу).

SysAid позиціонує свої AI-можливості як «AI-перший» підхід до сервіс-деску: ШІ-агенти інтегровані як у фронт-офіс (портал самообслуговування, чат для користувачів), так і в бек-офіс (інтерфейс аналітика). AI-інструменти допомагають автоматично класифікувати запити, визначати пріоритети, пропонувати категорії (рис. 1.6) та групи підтримки, а також генерувати варіанти відповідей і статей бази знань на основі історії інцидентів.

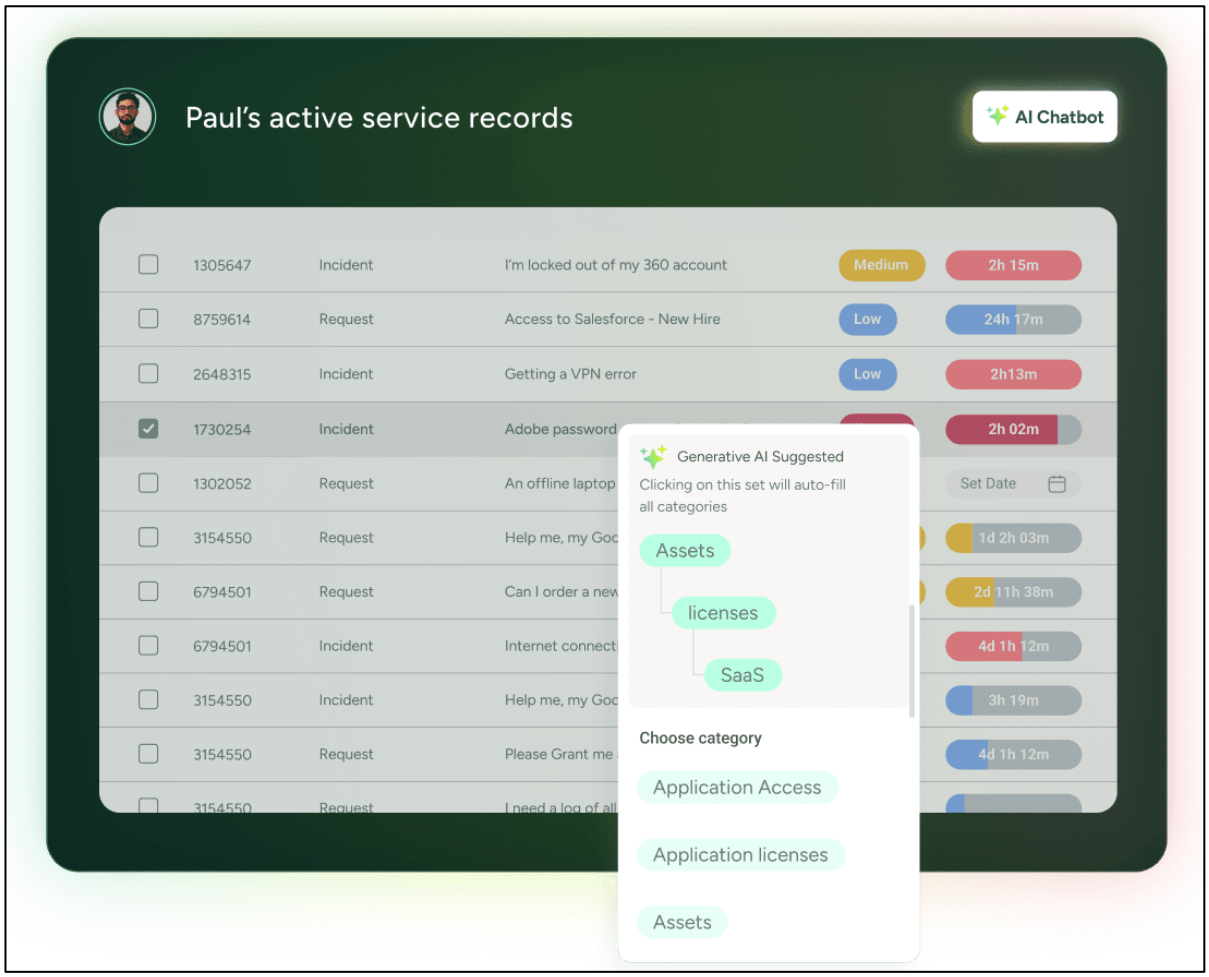

Рисунок 1.6 – Приклад класифікації запиту за допомогою AI в SysAid

Окремі агенти можуть аналізувати нові звернення на схожість із минулими кейсами, пропонуючи вже перевірені рішення, або виявляти «хвости» повторюваних інцидентів, що потребують переходу в управління проблемами (рис. 1.7).

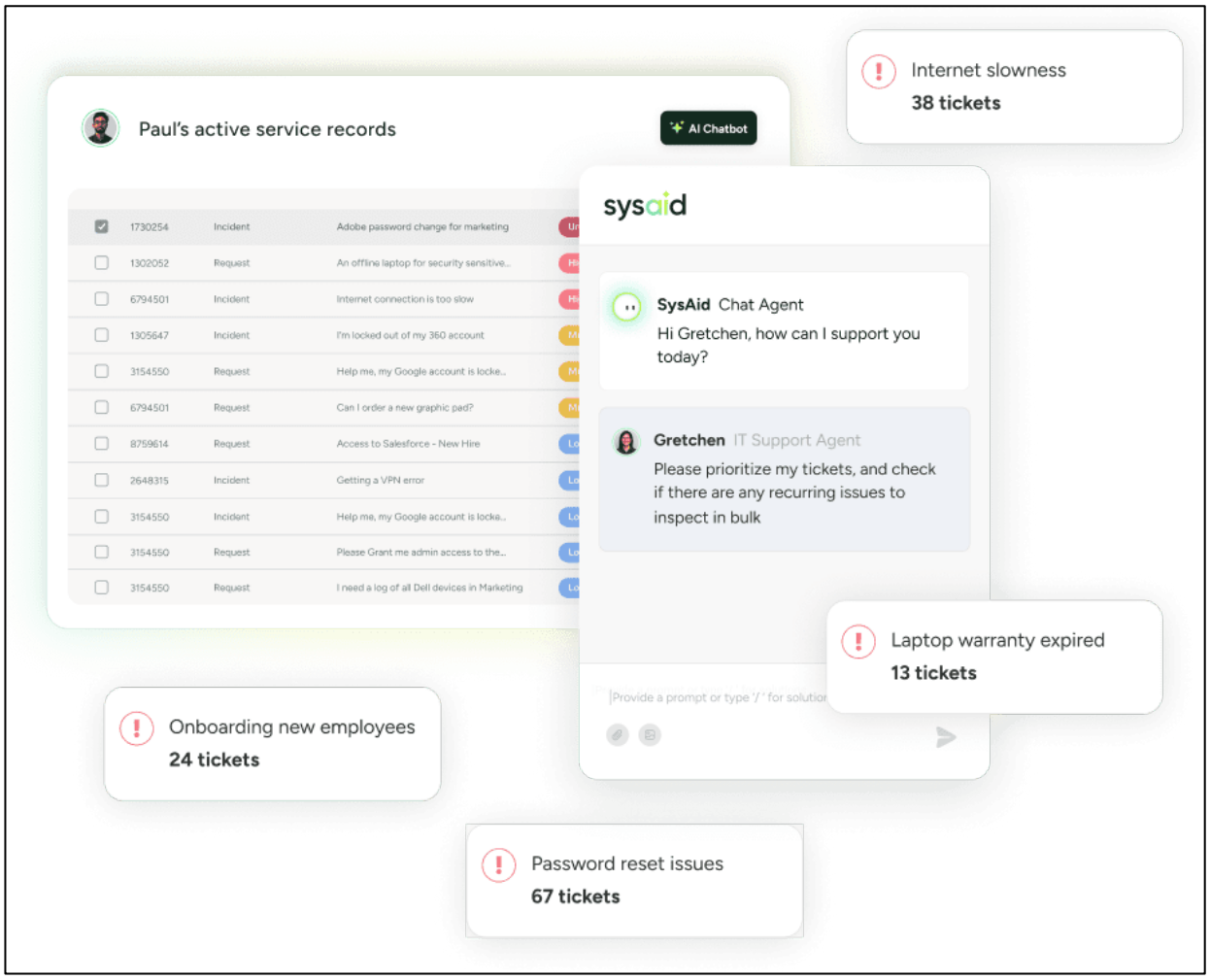

Рисунок 1.7 – Розширені можливості AI агентів у SysAid

У нових версіях реалізовано AI Service Desk Assistant, який виступає єдиною точкою входу для користувачів і аналітиків: користувачі можуть у природній мові описати проблему, а агент перетворює це на структуровану заявку, одразу заповнюючи поля й додаючи попередні рекомендації; аналітики ж отримують підказки щодо наступних кроків, пов'язаних заявок, можливих першопричин і необхідних змін. Такий підхід дозволяє суттєво зменшити час первинної обробки звернень, скоротити кількість помилок при класифікації, раніше виявляти масові інциденти та підвищувати рівень виконання SLA, хоча повноцінне довгострокове прогнозування інцидентів поки що обмежене окремими сценаріями (класифікація, рекомендації, автонаповнення на основі історичних даних).

1.5 Висновки

У процесі аналізу предметної області було досліджено концептуальні основи управління послугами інформаційних технологій, включаючи міжнародні стандарти ISO/IEC 20000 та методологію ITIL, ключові процеси ITSM, а також проведено комплексний огляд шести провідних ITSM-платформ. Особливу увагу приділено дослідженню механізмів управління інцидентами, проблемами, змінами та запитами на обслуговування, які становлять основу операційної діяльності служб технічної підтримки в сучасних організаціях.

У результаті проведеного аналізу встановлено, що сучасні ITSM-системи активно інтегрують AI-технології, проте більшість рішень зосереджена на реактивній автоматизації процесів (автоматична класифікація запитів, визначення пріоритетів, генерація відповідей) та контекстній допомозі агентам підтримки, тоді як повноцінний проактивний функціонал прогнозування інцидентів, виявлення майбутніх піків навантаження та систематичного аналізу повторюваних проблем з автоматичною генерацією структурованих превентивних рекомендацій залишається обмеженим навіть у рішеннях ринкових лідерів.

Виявлено, що існуючі AI-компоненти переважно орієнтовані на підвищення продуктивності існуючих процесів обробки звернень, а не на фундаментальну трансформацію підходу від реактивного до проактивного управління інцидентами, що обґрунтовує необхідність розробки спеціалізованої технології прогнозування та мінімізації інцидентів з акцентом на превентивних заходах.

2 ВИБІР ОПТИМАЛЬНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

2.1 Вибір мови програмування та середовища розробки

Для виконання поставленої задачі прогнозування та мінімізації інцидентів в системах управління IT-сервісами було використано комплекс сучасних інформаційних технологій, які забезпечили ефективне вирішення завдань аналізу часових рядів, машинного навчання та виявлення аномалій.

Відповідно до розглянутої в роботі [1] концепції переходу ITSM до інтелектуальних проактивних систем, використання стеку технологій на базі мови Python з сучасними бібліотеками аналізу даних, прогнозування часових рядів та виявлення аномалій (Pandas, NumPy, scikit-learn, statsmodels тощо), а також ансамблевих моделей машинного навчання розглядається як один із найбільш ефективних та практично орієнтованих підходів. Це підтверджує актуальність і доцільність обраних у даному дослідженні методів та інструментів для побудови системи прогнозування та мінімізації інцидентів в управлінні IT-послугами.

Основою технологічного стеку проєкту обрано мову програмування Python, що обумовлено низкою переваг цієї платформи для наукових досліджень та аналізу даних. Python є високорівневою інтерпретованою мовою програмування, яка відзначається простотою синтаксису, гнучкістю та широким спектром спеціалізованих бібліотек для наукових обчислень. На відміну від багатьох традиційних мов загального призначення, Python орієнтований на високу читабельність коду: його конструкції близькі до природної мови, що знижує поріг входу для дослідників, які не є професійними розробниками, та спрощує колективну роботу над експериментальним кодом [11]. Вибір Python обґрунтовується також його популярністю в науковій спільноті, активною підтримкою з боку розробників та наявністю великої кількості інструментів для машинного навчання та аналізу даних [11]. Саме завдяки цьому Python фактично

став «стандартом де-факто» для задач обробки даних, розробки та валідації моделей машинного навчання, побудови прототипів інтелектуальних систем.

Інтерпретований характер Python і динамічна типізація роблять його особливо зручним для дослідницького та експериментального програмування. Науковці можуть швидко змінювати фрагменти коду, перевіряти гіпотези, додавати нові етапи попередньої обробки даних або порівнювати різні моделі без необхідності складних процедур збирання (build) та розгортання застосунку. Це скорочує цикл «гіпотеза – експеримент – аналіз результатів» і дає змогу зосередитися на змісті дослідження, а не на технічних деталях інфраструктури розробки [11]. Крім того, широка екосистема Python охоплює численні бібліотеки для візуалізації (Matplotlib, Seaborn, Plotly), побудови класичних ML-моделей (scikit-learn) та роботи з глибинним навчанням (TensorFlow, PyTorch), що дозволяє комплексно підтримувати весь життєвий цикл наукового експерименту – від завантаження даних до побудови звітів і демонстраційних прототипів.

Для обробки та маніпуляції даними використано бібліотеку Pandas, яка є стандартом індустрії для роботи з табличними даними та часовими рядами. Pandas надає потужні структури даних, такі як `DataFrame` та `Series`, які дозволяють ефективно виконувати операції читання, очищення, трансформації та агрегації великих обсягів даних [12]. З практичної точки зору це означає, що дослідник може у декілька рядків коду завантажити дані з різних джерел (CSV, бази даних, Excel, формати для великих даних), об’єднати їх, відфільтрувати, обчислити похідні показники та підготувати вибірки для навчання моделей машинного навчання. Важливою перевагою Pandas є тісна інтеграція з іншими елементами Python-екосистеми: результати обчислень легко передаються до бібліотек візуалізації, інструментів машинного навчання та статистичного аналізу [12]. Саме тому Pandas широко застосовується в академічних дослідженнях, де необхідно швидко експериментувати з різними представленнями даних і перевіряти вплив вибору ознак (feature engineering) на якість моделей.

Для виконання чисельних обчислень застосовано бібліотеку NumPy, яка забезпечує підтримку багатовимірних масивів та матриць, а також містить велику кількість оптимізованих функцій для виконання математичних операцій [13]. На відміну від базових можливостей Python-списків, структури NumPy реалізують векторизовані операції: обчислення над великими масивами даних виконуються на низькому рівні, використовуючи оптимізований C/Fortran-код, що суттєво підвищує продуктивність [13]. Це має вирішальне значення для задач машинного навчання та аналізу часових рядів, де необхідно обробляти великі матриці ознак, виконувати лінійну алгебру, спектральний аналіз, згорткові операції тощо. Фактично більшість сучасних ML-бібліотек на Python використовують NumPy як базовий «обчислювальний двигун», тому його застосування є природною частиною наукового стеку.

Середовищем розробки та документування обрано Jupyter Notebook, який є інтерактивним інструментом для наукових обчислень та дослідницького програмування. Jupyter дозволяє поєднувати виконуваний код, візуалізації, markdown-текст та математичні формули в єдиному документі, що сприяє відтворюваності досліджень та ефективній комунікації результатів [14]. У такому форматі дослідник може послідовно фіксувати всі етапи експерименту: опис постановки задачі, завантаження та попередню обробку даних, вибір та налаштування моделей, інтерпретацію отриманих результатів. Це суттєво відрізняється від класичного підходу, коли код, проміжні результати та пояснення зберігаються в окремих файлах і документах, що ускладнює перевірку та повторення експериментів іншими фахівцями [14].

Популярність Jupyter Notebook у науковій спільноті пояснюється також підтримкою інтерактивного режиму роботи: дослідник може виконувати код по клітинках, змінювати параметри моделей «на льоту», миттєво отримувати оновлені графіки та таблиці. Це особливо корисно для exploratory data analysis (EDA), коли необхідно поступово «вивчати» дані, будуючи різні візуалізації, обчислюючи нові показники та перевіряючи гіпотези щодо їх розподілів та взаємозв’язків [14]. Крім того, Jupyter добре інтегрується з бібліотеками

візуалізації та дозволяє створювати інтерактивні графіки, що спрощує аналіз складних багатовимірних даних та полегшує представлення результатів нефаховій аудиторії (менеджменту, замовникам, колегам з інших дисциплін).

Важливим аргументом на користь поєднання Python і Jupyter Notebook у наукових проєктах є підтримка концепції «literate programming», коли код, результати та пояснювальний текст формують єдиний наратив. Це зменшує ризик розходження між тим, що фактично було виконано в експерименті, і тим, що описано в статті або звіті, оскільки всі ключові дії зафіксовано безпосередньо в ноутбуці [14]. Такий підхід полегшує рецензування, спільну роботу над дослідженням і подальше розширення експериментів іншими дослідниками. У контексті даної роботи це дозволяє прозоро продемонструвати всі етапи побудови технології прогнозування та мінімізації інцидентів: від завантаження даних ITSM-системи до фінальної оцінки якості моделей і побудови візуалізацій.

Отже, обраний технологічний стек – мова програмування Python у поєднанні з бібліотеками Pandas та NumPy і середовищем Jupyter Notebook – повністю відповідає вимогам до сучасних наукових досліджень в галузі аналізу даних та машинного навчання. Python забезпечує гнучкість і багатство екосистеми інструментів [11], Pandas і NumPy – ефективну роботу з даними та чисельними обчисленнями [12, 13], а Jupyter Notebook – інтерактивність, відтворюваність та зручність документування експериментів [14]. Сукупність цих властивостей робить вказаний набір технологій оптимальним вибором для реалізації й дослідження технології прогнозування та мінімізації інцидентів у середовищі ITSM.

2.2 Методи прогнозування часових рядів

Вибір методів прогнозування часових рядів є критично важливим для забезпечення точності передбачення майбутніх обсягів інцидентів, оскільки самі інциденти, запити та навантаження на IT-інфраструктуру мають виразну часову природу. Як було показано в розділі 1 (аналіз предметної області ITSM),

кількість інцидентів, їх типи та критичність суттєво залежать від робочих графіків користувачів, періодів впровадження змін, сезонних піків бізнес-активності та інших факторів, що змінюються в часі. Унаслідок цього дані ITSM формують часові ряди з трендами, сезонністю, періодами підвищеної турбулентності (масові інциденти, великі релізи, міграції) та відносно спокійними інтервалами. Ігнорування цієї структури призводить до спрощених і неточних моделей, які не здатні коректно оцінювати ризики перевантаження окремих сервісів або ймовірність виникнення сплесків інцидентів у майбутньому.

У контексті ITSM прогнозування часових рядів дозволяє не просто «передбачити число заявок на завтра», а підтримати низку ключових управлінських рішень: планування ресурсів служби підтримки (кількість операторів на зміні, розподіл черг між лініями), визначення вікон для змін і релізів з мінімальним ризиком, резервування потужностей інфраструктури, планування профілактичних робіт. Наприклад, коректна модель часових рядів може показати, що в певні години понеділка після великих вихідних завжди спостерігається різкий стрибок інцидентів, пов’язаних з доступом; або що після щомісячного оновлення ключового бізнес-додатку протягом двох днів стабільно зростає кількість звернень щодо продуктивності. Такі закономірності важко виявити «на око», однак вони природним чином відображаються у часових рядах метрик ITSM і можуть бути формалізовані в моделях прогнозування.

Крім того, часові ряди інцидентів, змін і показників навантаження тісно пов’язані між собою, утворюючи багатовимірний часовий процес: сплески навантаження на один сервіс можуть зумовлювати появу інцидентів у суміжних системах, а серії невдалих змін – призводити до «ланцюгових» відмов. Методи прогнозування часових рядів (у тому числі мультиваріантних) дають змогу враховувати ці взаємозв’язки, моделювати затримки у впливі одних подій на інші та будувати більш реалістичні сценарії розвитку ситуації. Для задачі мінімізації інцидентів це принципово: важливо не лише знати загальний тренд, а й вміти заздалегідь виявляти «критичні точки» у часі, коли ймовірність інцидентів різко

зростає, щоб встигнути змінити конфігурацію, підсилити команду або перенести ризиковані роботи.

Таким чином, прогнозування часових рядів у даній МКР розглядається як один із ключових механізмів реалізації проактивного та предиктивного підходів в ITSM, описаних у розділі 1. Воно забезпечує перехід від статичної оцінки ризиків до динамічного, часового бачення стану IT-сервісів, дозволяючи заздалегідь оцінити, коли і за яких умов система наближається до небезпечних режимів роботи. У наступних підрозділах буде детально розглянуто конкретні методи моделювання часових рядів, проаналізовано їхні переваги та обмеження для задач прогнозування інцидентів у середовищі ITSM, а також обґрунтовано вибір підходів, що лягли в основу запропонованої технології прогнозування та мінімізації інцидентів.

2.2.1 Модель SARIMA

Модель SARIMA (Seasonal AutoRegressive Integrated Moving Average) є розширенням класичної моделі ARIMA, яка додає компоненти для моделювання сезонних патернів у даних [15]. SARIMA представляється як:

$$SARIMA(p, d, q)(P, D, Q)[s], \tag{2.1}$$

де p – порядок авторегресійної компоненти (AR), d – порядок диференціювання для досягнення стаціонарності (I), q – порядок компоненти ковзного середнього (MA), P, D, Q – відповідні сезонні параметри; s – період сезонності.

Для визначення оптимальних параметрів моделі SARIMA було застосовано алгоритм Auto-ARIMA з бібліотеки statsmodels [16], який автоматично перебирає можливі комбінації параметрів та обирає модель з найкращим інформаційним критерієм Акаїке (AIC):

$$AIC = 2k - 2ln(L), \tag{2.2}$$

де k – кількість параметрів моделі, L – максимізована функція правдоподібності.

Алгоритм Auto-ARIMA виконує наступні кроки:

- тестування стаціонарності часового ряду за допомогою розширеного тесту Дікі-Фуллера (ADF);
- визначення необхідного порядку диференціювання d та D;
- систематичний пошук оптимальних значень p, q, P, Q;
- оцінка параметрів методом максимальної правдоподібності;
- вибір моделі з мінімальним AIC серед всіх протестованих комбінацій.

2.2.2 Модель Prophet

Prophet є адитивною регресійною моделлю, розробленою дослідниками Meta (Facebook) спеціально для прогнозування бізнес-метрик з виразною сезонністю [16]. Оригінальна публікація Taylor та Letham у 2018 році [17] представила Prophet як інструмент для автоматизованого прогнозування великомасштабних часових рядів з мінімальним ручним налаштуванням.

Модель декомпозує часовий ряд на чотири основні компоненти:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon\_t, \tag{2.3}$$

де $g(t)$ – функція тренду, що моделює довгострокові змін, $s(t)$ – сезонна компонента (добова, тижнева, річна), $h(t)$ – вплив святкових днів та особливих подій, ε_t – похибка (випадкова складова).

Prophet підтримує два типи трендів: лінійний треянд з точками зміни (2.1) та логістичний тренд, що використовується для обмеженого зростання (2.2).

$$g(t) = (k + a(t)^T\delta)t + (m + a(t)^T\gamma), \tag{2.4}$$

де k – базова швидкість зростання, δ – вектор змін швидкості зростання в точках зміни, m – початковий зсув, $a(t)$ – індикатор активних точок зміни.

$$g(t) = \frac{C}{1 + exp\left(-k(t - m)\right)}, \tag{2.5}$$

де C – гранична ємність (carrying capacity).

Сезонні компоненти моделюються за допомогою рядів Фур'є:

$$s(t) = \Sigma(n = 1\ to\ N)\ [a_n cos\left(\frac{2\pi nt}{P}\right) + b_n sin\left(\frac{2\pi nt}{P}\right)], \tag{2.6}$$

де P – період сезонності (7 для тижневої, 365.25 для річної), N – кількість гармонік.

Для щотижневої сезонності зазвичай використовується $N = 3$, що дозволяє моделювати складні патерни в межах тижня. Для річної сезонності використовується $N = 10$ для захоплення детальних сезонних ефектів.

Вплив святкових днів моделюється як індикаторна функція з вікном впливу:

$$h(t) = \Sigma(i \in H)\kappa_i \cdot 1\left[t \in [d_i - w_b, d_i + w_a]\right], \tag{2.7}$$

де H – множина святкових днів, κ_i – вплив святкового дня i, w_b та w_a – вікна впливу до та після святкового дня.

Параметри моделі Prophet оцінюються методом максимальної апостеріорної оцінки (MAP) з використанням L2-регуляризації:

$$max_\beta[ln\ p(\beta|y) = ln\ p(y|\beta) + ln\ p(\beta)], \tag{2.8}$$

де $p(\beta)$ – апріорний розподіл параметрів (зазвичай нормальний), що забезпечує регуляризацію та запобігає перенавчанню.

2.2.3 Модель SMA-DOW (Seasonal Moving Average with Day-of-Week)

У межах даної роботи як третій елемент ансамблю використано модель SMA-DOW (Seasonal Moving Average with Day-of-Week adjustment) – метод сезонного ковзного середнього з корекцією по дню тижня. Модель поєднує кілька наївних стратегій прогнозування (naive, seasonal naive, ковзне середнє, трендовий дрейф та денні патерни). На відміну від складніших статистичних або нейромережевих підходів, SMA-DOW ґрунтується лише на простих агрегованих характеристиках історичних даних (останнє значення, середні значення за вікно, середні значення за днями тижня, оцінка лінійного тренду), що робить її інтерпретованою та стійкою до обмеженого обсягу вибірки.

Під час навчання SMA-DOW обчислює рухоме середнє за фіксоване вікно, середні значення кількості інцидентів для кожного дня тижня та оцінку добового тренду. Прогноз формується як зважена комбінація кількох наївних прогнозів: постійного (останнє спостереження), сезонно-наївного (значення відповідного дня попереднього тижня), прогнозу на основі ковзного середнього, лінійного дрейфу та денних патернів. Такий підхід дозволяє одночасно врахувати як стійкі тижневі сезонні коливання, так і загальний тренд навантаження, зберігаючи при цьому низьку складність моделі.

У практичному застосуванні SMA-DOW виконує одразу кілька ролей. По-перше, вона слугує «еталонною лінією» (baseline) для оцінки користі складніших методів: будь-яка модель, що не перевищує якість SMA-DOW, вважається непридатною для використання. По-друге, завдяки простоті та відсутності жорстких припущень щодо розподілу даних SMA-DOW забезпечує робастний резервний прогноз у тих випадках, коли SARIMA або Prophet можуть давати нестабільні або неінтерпретовані результати (наприклад, при різких структурних зсувах у часовому ряді). Нарешті, включення SMA-DOW до підсумкового ансамблю дозволяє зменшити дисперсію прогнозу за рахунок низької корельованості її похибок із похибками інших моделей.

2.2.4 Ансамблеве прогнозування та обґрунтування вибору

Для підвищення робастності та точності прогнозів у даній роботі було реалізовано ансамблевий підхід, що комбінує прогнози трьох взаємодоповнюючих моделей: SARIMA, Prophet та SMA-DOW. Фінальний прогноз обчислюється як зважене середнє:

$$\hat{y}_{ensemble(t)} = \Sigma\,(i = 1\ to\ M) w_i \cdot \hat{y}_i(t), \tag{2.9}$$

де M – кількість моделей в ансамблі, w_i – вага моделі i, $\Sigma w_i = 1$.

SARIMA краще моделює короткострокові автокореляції та локальні лінійні залежності, Prophet ефективніший для довгострокових трендів і складних сезонних патернів (у тому числі з урахуванням свят і особливих подій), тоді як SMA-DOW фіксує прості, але стійкі тижневі та добові закономірності й забезпечує інтерпретовану «нижню межу» якості. Завдяки різній природі моделей їхні похибки мають обмежену кореляцію, що є важливою передумовою ефективності ансамблю.

Згідно з теорією ансамблевого навчання, середньоквадратична похибка ансамблю моделей визначається за формулою:

$$MSE_{ensemble} = \rho \cdot \sigma^2 + (1 - \rho) \cdot \frac{\sigma^2}{M}, \tag{2.10}$$

де σ^2 – середня дисперсія індивідуальних моделей, M – кількість моделей в ансамблі (у нашому випадку $M = 2$: SARIMA та Prophet), ρ – коефіцієнт кореляції між похибками моделей (значення від 0 до 1). Формулу можна переписати у еквівалентному вигляді:

$$MSE_{ensemble} = \sigma^2 \cdot \left[\rho + \frac{(1 - \rho)}{M}\right], \tag{2.11}$$

при $\rho \approx 0$ (некорельовані моделі) досягається максимальне зниження помилки до $\frac{\sigma^2}{M}$. При $\rho \approx 1$ (повністю корельовані моделі) ансамбль не дає покращення.

У процесі вибору методів прогнозування було розглянуто низку альтернатив. Класичний метод експоненційного згладжування Хольта-Вінтерса, хоча й придатний для рядів з трендом і сезонністю, виявився недостатньо гнучким для складних патернів потоку інцидентів та поступився за точністю запропонованому ансамблю. Рекурентні нейронні мережі LSTM (Long Short-Term Memory) вимагали значно більшого обсягу навчальних даних та обчислювальних ресурсів; на наявному датасеті обсягом 8 місяців вони виявляли схильність до перенавчання та демонстрували неприйнятні значення MAPE для практичного застосування. Гібридна модель Facebook NeuralProphet, що поєднує ідеї Prophet з нейронними мережами, на момент розробки системи перебувала на ранній стадії розвитку з обмеженою документацією та недостатньо валідованими промисловими кейсами, тому була відхилена на користь більш зрілих та передбачуваних інструментів.

Таким чином, обраний ансамбль із трьох моделей: SARIMA, Prophet та SMA-DOW – забезпечує збалансоване рішення, яке поєднує високу якість прогнозів, інтерпретованість результатів, робастність до обмеженого обсягу даних та прийнятні вимоги до обчислювальних ресурсів.

2.3 Алгоритм Isolation Forest для виявлення аномалій

Виявлення аномальних періодів у потоці інцидентів є критично важливим для проактивного управління IT-сервісами, оскільки саме такі періоди найчастіше відповідають масовим відмовам, некоректним змінам або прихованим проблемам в інфраструктурі. Як було показано в аналізі предметної області ITSM, інциденти мають виразну часову структуру, залежать від навантаження, змін і зовнішніх факторів; однак окремі «сплески» звернень чи нетипові поєднання атрибутів (категорій, сервісів, CI) не завжди можна пояснити відомими патернами. Саме тому завдання автоматичного виявлення аномалій

(outlier detection) доповнює класичне прогнозування часових рядів: якщо прогноз дає очікуваний «нормальний» рівень інцидентів, то алгоритми пошуку аномалій дозволяють сигналізувати про відхилення від цієї норми, які можуть свідчити про нові типи відмов або небезпечні зміни в поведінці системи.

Для цієї задачі було обрано алгоритм Isolation Forest [18], який демонструє високу ефективність при роботі з багатовимірними даними та не вимагає попереднього маркування аномалій. На відміну від багатьох класичних статистичних підходів, що спираються на припущення щодо розподілу даних (наприклад, нормальність) або потребують явного визначення порогів, Isolation Forest розглядає аномалії як об’єкти, які легше «ізолювати» від решти вибірки. Алгоритм побудований на ансамблі випадкових дерев, кожне з яких рекурсивно розділяє простір ознак за випадковими атрибутами та порогами; об’єкти, які ізолюються за невелику кількість кроків (тобто мають коротку середню довжину шляху в дереві), інтерпретуються як потенційні аномалії [18]. Такий підхід є природно неспостережуваним (unsupervised) і добре підходить для ITSM-даних, де явне маркування аномальних періодів часто відсутнє або є неповним.

У контексті ITSM Isolation Forest дозволяє аналізувати не лише сирий часовий ряд кількості інцидентів, а й багатовимірні ознаки, що описують стан сервісу в певний момент: розподіл інцидентів за категоріями, частку критичних звернень, структуру пріоритетів, пов’язані зміни, навантаження на окремі модулі чи CI тощо. У такому багатовимірному просторі «нормальні» стани сервісу формують щільні кластери, тоді як рідкісні, але важливі ситуації (масові збої, ланцюгові відмови, некоректні релізи) виявляються як об’єкти, що ізолюються значно швидше. Це дозволяє виявляти як короткі «сплески» інцидентів, так і більш тривалі періоди нетипової поведінки, які можуть бути непомітними при візуальному аналізі окремих метрик.

Ще однією перевагою Isolation Forest є його стійкість до великої розрідженості аномалій та можливість ефективної обробки великих обсягів історичних даних ITSM [19]. У реальних сервіс-деск системах переважна більшість спостережень відповідає «звичайному» стану, тоді як по-справжньому

критичні інциденти становлять малу частку всієї історії, що ускладнює застосування класичних методів класифікації, орієнтованих на збалансовані вибірки. Isolation Forest природним чином враховує цю нерівномірність, оскільки саме рідкісність об’єкта в просторі ознак робить його «легко ізольованим» і, відповідно, аномальним. Це дає змогу без додаткового балансування даних будувати моделі, які чутливо реагують на нетипові поєднання ознак, але відносно нечутливі до шуму в «нормальних» режимах роботи.

Таким чином, застосування алгоритму Isolation Forest у межах даної роботи розглядається як важливий компонент загальної технології прогнозування та мінімізації інцидентів. Поєднання прогнозування часових рядів (для оцінки очікуваного навантаження) з виявленням аномалій (для фіксації нетипових, потенційно небезпечних станів) дозволяє реалізувати більш повний проактивний підхід до управління IT-сервісами: не лише «знати», скільки інцидентів очікується в середньому, але й вчасно помічати ситуації, коли реальна поведінка системи виходить за межі очікуваного діапазону [19]. У наступних підрозділах будуть наведені практичні аспекти підготовки ознак, налаштування параметрів Isolation Forest та інтерпретації його результатів у контексті ITSM-процесів.

2.3.1 Принцип роботи алгоритму Isolation Forest

Isolation Forest базується на фундаментальному припущенні: аномалії є рідкісними та суттєво відрізняються від нормальних спостережень, тому їх легше "ізолювати" в просторі ознак. Алгоритм будує ансамбль випадкових дерев ізоляції (isolation trees), де кожне дерево рекурсивно розділяє простір ознак випадковими гіперплощинами. Процес побудови дерева ізоляції:

1. випадково обирається ознака x_i з набору ознак;
2. випадково обирається значення розділення p між $\min(x_i)$ та $\max(x_i)$;
3. простір розділяється на дві частини: $\{x \mid x\_i < p\}$ та $\{x \mid x\_i \geq p\}$;
4. процес рекурсивно повторюється для кожної частини до досягнення критерію зупинки.

Принцип роботи алгоритму базується на спостереженні, що аномальні спостереження характеризуються меншою щільністю в просторі ознак порівняно з нормальними точками. Як проілюстровано на рисунку 2.1, аномальні точки знаходяться далеко від щільних регіонів нормальних даних, тому для їх ізоляції в дереві розділень потрібно значно менше кроків. Натомість, нормальні точки, які розташовані в щільних кластерах, потребують більшої кількості послідовних розділень простору для їх повної ізоляції від інших спостережень.

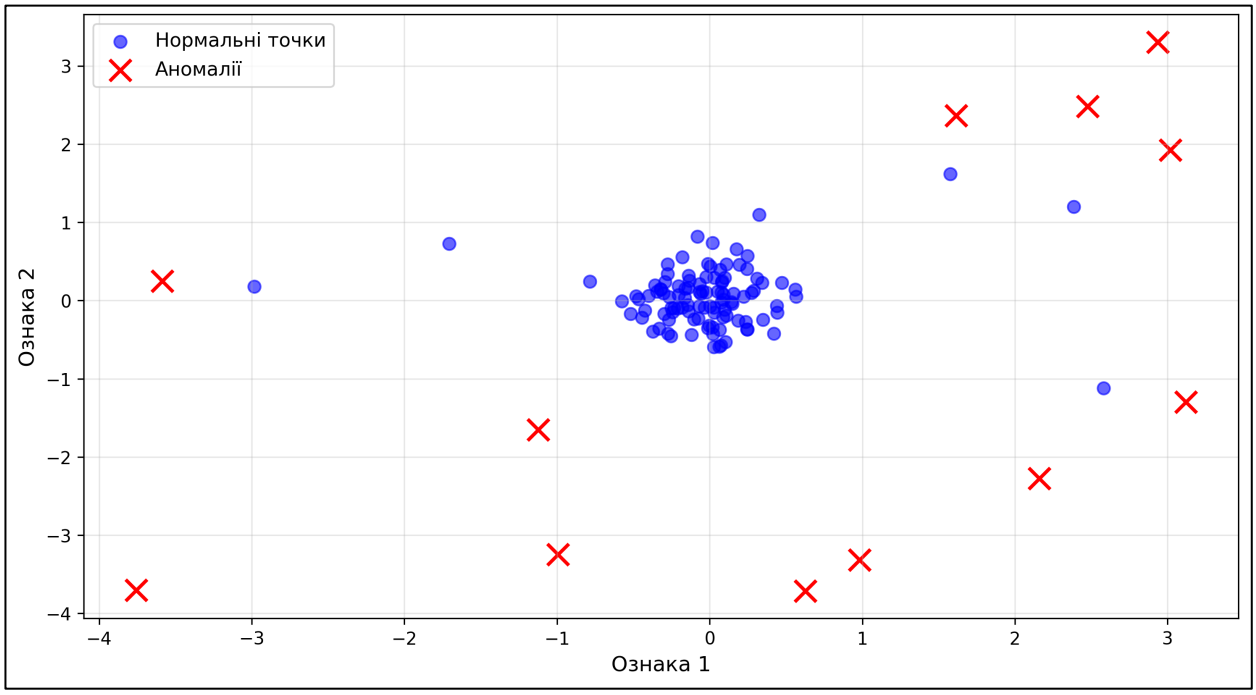

Рисунок 2.1 – Ілюстрація принципу роботи Isolation Forest

2.3.2 Математична формалізація

Для кожної точки x обчислюється середня довжина шляху $h(x)$ по всіх деревах ансамблю. Аномальна оцінка визначається як:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}, \tag{2.12}$$

де $E(h(x))$ – середня довжина шляху до ізоляції точки x, n – розмір вибірки, $c(n)$ – нормалізаційний фактор, що дорівнює середній довжині шляху у бінарному дереві пошуку.

Нормалізаційний фактор обчислюється як:

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n}, \tag{2.13}$$

де $H(i) = ln(i) + \gamma$ – i-те гармонічне число, $\gamma \approx 0.5772$ – константа Ейлера-Маскероні.

Отримане значення аномальної оцінки $s(x, n)$ знаходиться в діапазоні $[0, 1]$ і характеризує ступінь аномальності кожного спостереження. Значення цього показника інтерпретується наступним чином:

- $s \approx 1$ – точка є аномалією (ізолюється швидко);
- $s < 0.5$ – точка є нормальною (потребує багато розділень);
- $s \approx 0.5$ – точка знаходиться на межі.

Повний алгоритм виявлення аномалій з використанням Isolation Forest включає побудову ансамблю дерев ізоляції та обчислення аномальних оцінок для кожної точки.

2.3.3 Порівняння з альтернативними методами виявлення аномалій

Для обґрунтування вибору Isolation Forest як основного інструменту виявлення аномалій було проведено порівняльний аналіз альтернативних методів, які широко застосовуються в задачах детекції відхилень у багатовимірних даних.

Метод One-Class SVM (Support Vector Machine) базується на побудові гіперсфери або гіперплощини в просторі ознак, яка охоплює нормальні спостереження, при цьому аномалії класифікуються як точки, що знаходяться

поза встановленою межею. Цей підхід є теоретично обґрунтованим та демонструє ефективність на малих наборах даних. Водночас, One-Class SVM характеризується високою обчислювальною складністю $O(n^2)$ або $O(n^3)$ залежно від обраного ядра, значною чутливістю до налаштування параметрів (ν, γ) та поганою масштабованістю при роботі з великими обсягами даних, що обмежує його застосування в системах реального часу.

Алгоритм Local Outlier Factor (LOF) здійснює оцінку локальної щільності кожної точки відносно щільності її сусідів, визначаючи аномалії як спостереження зі значно нижчою локальною щільністю. Метод обчислює показник аномальності за формулою:

$$LOF(x) = \frac{\frac{\Sigma\left(y \in N_{k(x)}\right) lrd(y)}{\left|N_{k(x)}\right|}}{lrd(x)}, \tag{2.14}$$

де $lrd(x)$ – local reachability density (локальна досяжна щільність), $N_{k(x)}$ – множина k найближчих сусідів точки x.

Перевагою LOF є здатність виявляти локальні аномалії без необхідності глобальних припущень про розподіл даних. Проте метод має обчислювальну складність $O(n^2)$ через необхідність розрахунку відстаней між усіма парами точок, демонструє високу чутливість до вибору параметра k та характеризується низькою швидкодією на великих наборах даних.

Метод Support Vector Data Description (SVDD) є варіацією методу опорних векторів, який будує мінімальну гіперсферу в просторі ознак, що містить більшість нормальних спостережень. Незважаючи на теоретичну привабливість, SVDD має високі обчислювальні витрати та значну складність у виборі оптимальних параметрів, що ускладнює його практичне застосування.

Результати порівняльного аналізу розглянутих методів виявлення аномалій узагальнено в таблиці 2.1.

Таблиця 2.1 – Порівняльна характеристика методів виявлення аномалій

Метрика	**Isolation Forest**	**One-Class SVM**	**LOF**	**SVDD**
Часова складність	$O(tn\ log\ n)$	$O(n^2 - n^3)$	$O(n^2)$	$O(n^2)$
Масштабованість	Висока	Низька	Середня	Низька
Чутливість до параметрів	Низька	Висока	Висока	Висока
Багатовимірність	Відмінна	Добра	Погіршується	Добра
Інтерпретованість	Висока	Середня	Висока	Середня

Вибір Isolation Forest як основного методу виявлення аномалій обґрунтовується низкою ключових переваг цього алгоритму. По-перше, метод характеризується лінійно-логарифмічною обчислювальною складністю $O(tn\ log\ n)$, де t – кількість дерев в ансамблі (зазвичай 100-200), а n – кількість спостережень, що дозволяє ефективно обробляти великі обсяги даних в режимі реального часу.

По-друге, Isolation Forest не вимагає апріорних припущень щодо форми розподілу нормальних даних, що робить його універсальним інструментом для різноманітних доменів застосування. Основні параметри алгоритму (кількість дерев, розмір підвибірки, рівень забруднення contamination) мають інтуїтивну інтерпретацію та демонструють низьку чутливість у широкому діапазоні значень, що спрощує процес налаштування моделі.

По-третє, на відміну від методів, що базуються на обчисленні відстаней між точками (LOF, k-NN), Isolation Forest не страждає від "прокляття розмірності", оскільки алгоритм не залежить від метрик відстані та ефективно працює в багатовимірних просторах ознак.

2.4 Метод LIME для інтерпретації прогнозів

Інтерпретованість моделей машинного навчання є критично важливою для практичного застосування систем прогнозування в середовищі ITSM. Оператори служби підтримки та менеджери IT-сервісів потребують не лише числових прогнозів, а й зрозумілих пояснень, чому система очікує певний рівень навантаження або аномальну поведінку. Без таких пояснень довіра до автоматизованих рекомендацій залишається низькою, а рішення про превентивні

дії приймаються на основі інтуїції, а не даних. Саме тому в межах даної роботи було застосовано метод LIME (Local Interpretable Model-agnostic Explanations), запропонований Ribeiro та співавторами у 2016 році [20].

LIME є модельно-агностичним методом локальної інтерпретації, що дозволяє пояснювати окремі прогнози будь-якої «чорної скриньки» (black-box model) через побудову простої інтерпретованої моделі в околі конкретного спостереження [20]. Основна ідея методу полягає в тому, що навіть якщо глобальна поведінка складної моделі є нелінійною та важко зрозумілою, локально – в околі конкретної точки – її можна апроксимувати простою лінійною моделлю, коефіцієнти якої безпосередньо вказують на внесок кожної ознаки у прогноз.

Формально, для пояснення прогнозу моделі f у точці x метод LIME розв'язує оптимізаційну задачу:

$$\xi(x) = arg\ min_{\{g \in G\} L(f,g,\pi_x)} + \Omega(g), \tag{2.15}$$

де G – клас інтерпретованих моделей (зазвичай лінійні моделі або дерева рішень), L – функція втрат, що вимірює наближення g до f в околі x, π_x – функція близькості, що визначає вагу зразків залежно від їх відстані до x, $\Omega(g)$ – міра складності пояснювальної моделі (наприклад, кількість ненульових коефіцієнтів) [20].

Алгоритм LIME складається з наступних кроків:

- генерація синтетичних зразків у околі точки x шляхом випадкових збурень ознак;
- отримання прогнозів «чорної скриньки» f для кожного синтетичного зразка;
- зважування зразків за їх близькістю до оригінальної точки x з використанням ядерної функції (зазвичай експоненціальної);
- навчання простої інтерпретованої моделі g на зважених зразках;
- інтерпретація коефіцієнтів моделі g як внесків окремих ознак у прогноз.

У контексті прогнозування часових рядів інцидентів застосування LIME вимагає адаптації до специфіки темпоральних даних. На відміну від класичних табличних задач, де ознаки є незалежними атрибутами об'єкта, у часових рядах ключову роль відіграють лагові значення (минулі спостереження), ковзні статистики (середні, стандартні відхилення за вікно) та календарні фактори (день тижня, місяць, свята).

У реалізованій системі для кожного прогнозу формується вектор ознак, що включає:

- лагові значення $y_{\{t-1\}}, y_{\{t-2\}}, \ldots, y_{\{t-k\}}$ для захоплення короткострокової динаміки;
- ковзні середні та стандартні відхилення за вікна 7 та 14 днів;
- бінарні індикатори дня тижня для моделювання тижневої сезонності;
- індикатори початку та кінця місяця для захоплення можливих циклічних ефектів.

Застосування LIME до такого представлення дозволяє відповісти на практичні питання: «Чому модель прогнозує підвищене навантаження у вівторок?» – можливо, через високе значення $y_{\{t-7\}}$ (тиждень тому був також напружений день) або через те, що вівторок історично є найбільш завантаженим днем. Такі пояснення можуть бути автоматично трансформовані в текстові рекомендації для операторів служби підтримки.

Важливою перевагою LIME є можливість агрегації локальних пояснень для отримання глобальної картини важливості ознак. Усереднюючи абсолютні значення коефіцієнтів пояснювальних моделей по всіх прогнозах, можна визначити, які фактори в цілому найбільше впливають на прогнози системи. У реалізованій системі ця інформація візуалізується у вигляді рейтингу важливості ознак та використовується для формування проактивних сервісних запитів.

Таким чином, інтеграція методу LIME забезпечує прозорість роботи системи прогнозування інцидентів, підвищує довіру користувачів до автоматизованих рекомендацій та дозволяє формувати обґрунтовані пояснення

для кожного прогнозу [20]. Це є важливим компонентом загальної технології, що доповнює точність прогнозування інтерпретованістю результатів.

2.5 Висновки

У процесі вибору технологій для системи прогнозування та мінімізації інцидентів було проведено комплексний аналіз сучасних методів аналізу часових рядів та виявлення аномалій. Основними критеріями відбору виступали точність прогнозування, обчислювальна ефективність, робастність до обмеженого обсягу даних та інтерпретованість результатів.

За результатами порівняльного аналізу як платформу реалізації було обрано Python з екосистемою бібліотек Pandas, NumPy, scikit-learn та statsmodels. Середовищем розробки обрано Jupyter Notebook, що забезпечує інтерактивність, відтворюваність експериментів та зручність документування результатів.

Для прогнозування часових рядів реалізовано ансамблевий підхід, що поєднує три взаємодоповнюючі моделі: SARIMA для моделювання короткострокових автокореляцій та локальних залежностей, Prophet для довгострокових трендів і складних сезонних патернів, та SMA-DOW (Seasonal Moving Average with Day-of-Week) для фіксації стійких тижневих закономірностей і забезпечення інтерпретованої базової лінії якості. Ваги моделей в ансамблі оптимізуються автоматично на валідаційній вибірці, що дозволяє адаптувати внесок кожної моделі до специфіки конкретного набору даних.

Для виявлення аномалій обрано алгоритм Isolation Forest з бібліотеки scikit-learn, який демонструє високу ефективність при роботі з часовими рядами та не вимагає попереднього маркування аномальних періодів. Додатково реалізовано статистичні методи детекції аномалій на основі Z-score та міжквартильного розмаху (IQR), що забезпечує багаторівневу валідацію виявлених відхилень.

Для інтерпретації прогнозів застосовано метод LIME (Local Interpretable Model-agnostic Explanations), адаптований для часових рядів. Це дозволяє визначити внесок окремих ознак (лагових значень, календарних факторів, ковзних статистик) у кожен прогноз та сформувати зрозумілі пояснення для кінцевих користувачів системи.

Обраний технологічний стек створює основу для побудови модульної системи прогнозування та мінімізації інцидентів з можливістю незалежного розроблення та тестування компонентів. Архітектура рішення передбачає окремі модулі для прогнозування, аналізу трендів та виявлення аномалій, інтерпретації результатів та візуалізації, що спрощує підтримку та розширення функціональності системи.

3 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

3.1 Розвідувальний аналіз даних

Розробка технології прогнозування та мінімізації інцидентів у системах управління IT-сервісами розпочалася з комплексного розвідувального аналізу даних, що включав дослідження синтетичного набору даних для методологічної розробки та подальший аналіз реального набору даних з промислової системи управління IT-сервісами для валідації та адаптації розроблених підходів. Такий дворівневий підхід дозволив відпрацювати всі компоненти системи у контрольованому середовищі з відомими характеристиками перед застосуванням до реальних операційних даних з усіма їх специфічними особливостями та обмеженнями.

Для синтетичного набору даних було згенеровано річний період спостережень з понад десятьма тисячами сервісних записів, що відображали типові характеристики реальних систем управління IT-сервісами, включаючи добову та тижневу сезонність, розподіл за категоріями обслуговування, варіацію пріоритетів та реалістичні часи вирішення проблем. Дослідницький аналіз виявив виражену тижневу циклічність з підвищеним навантаженням на початку робочого тижня та зниженням активності у вихідні дні, а також добові патерни з піковими періодами у ранкові години та після обідньої перерви, що підтвердило необхідність врахування множинних сезонних компонент при побудові прогнозних моделей.

Візуалізація розподілів, що представлена на рисунку 3.1, має шість панелей з різними аспектами статистичного аналізу історичних даних. Горизонтальна стовпчаста діаграма відображає кількість сервісних записів по категоріях, кругова діаграма демонструє пропорційний розподіл за пріоритетами, стовпчаста діаграма показує активність по відділах організації. Нижня частина рисунка містить аналіз темпоральних патернів: середній час вирішення залежно від пріоритету, розподіл звернень по годинах доби з виділенням робочих годин,

та активність по днях тижня. Ці візуалізації дозволяють ідентифікувати ключові фактори навантаження на службу технічної підтримки та є основою для налаштування параметрів прогнозних моделей.

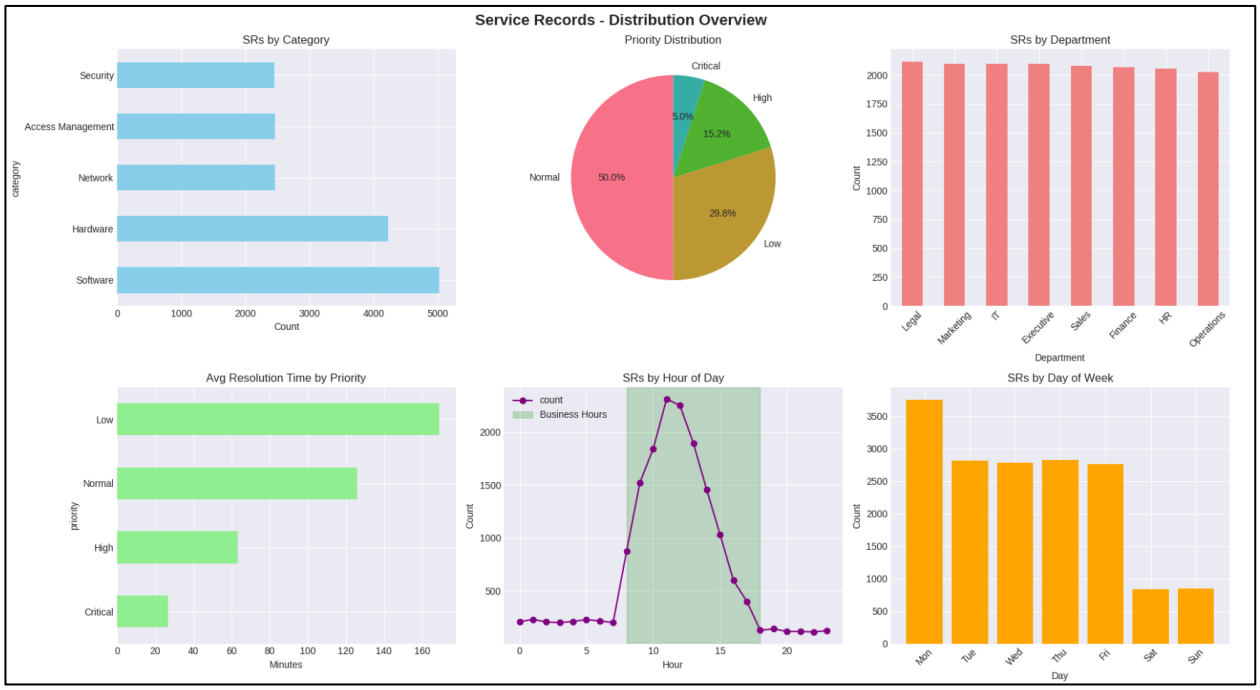

Рисунок 3.1 – Загальний огляд розподілу сервісних записів за категоріями, пріоритетами та темпоральними патернами у синтетичних даних

Після завершення етапу методологічної розробки було виконано отримання та розвідувальний аналіз реального набору даних з промислової системи управління IT-сервісами SysAid, що використовується організацією для централізованого обліку та управління сервісними запитами. Експортований файл містить історію сервісних записів за період з березня до жовтня 2025 року, що охоплює приблизно сім місяців операційної діяльності служби технічної підтримки та забезпечує достатній обсяг даних для виявлення сезонних патернів та побудови надійних прогнозних моделей.

Загальний обсяг даних склав дев'ятсот тридцять шість унікальних сервісних записів за двісті шість днів, що відповідає середній інтенсивності приблизно чотири-п'ять записів на робочий день. Структура експортованих даних включає сто двадцять одне поле, що охоплюють всі аспекти життєвого

циклу сервісного запису від моменту створення до закриття, включаючи інформацію про категоризацію, пріоритизацію, призначення відповідальних осіб, часові характеристики та результати вирішення проблем.

Аналіз повноти даних виявив суттєві прогалини: загальна кількість пропущених значень склала сорок п'ять відсотків від загального обсягу даних. Поля унікального ідентифікатора, дати створення запису та статусу заповнені для всіх без винятку записів, що забезпечує цілісність даних для побудови часових рядів та аналізу динаміки. Поле пріоритету заповнене для дев'яноста дев'яти відсотків записів. Дублікати у даних відсутні, що свідчить про коректність процесів експорту та первинної обробки.

Розподіл сервісних записів за статусами демонструє, що більшість записів перебувають у закритому статусі. Вісімсот сім записів (вісімдесят шість відсотків) мають статус закритих, що вказує на завершеність їх обробки та можливість використання для ретроспективного аналізу часу вирішення. Решта записів перебувають в активних статусах, що є типовим для стабільно функціонуючої служби підтримки.

Аналіз розподілу за пріоритетами (рис. 3.2) виявив структуру, що відповідає рекомендаціям кращих практик управління інцидентами. Домінуючими є запити низького пріоритету (P5), що складають шістдесят п'ять відсотків від загальної кількості. Запити пріоритету P4 складають двадцять три відсотки, тоді як запити середнього (P3) та високого (P2) пріоритету складають по чотири відсотки кожен. Критичні запити (P1) складають лише три відсотки від загального обсягу. Така структура вказує на коректну пріоритизацію звернень службою підтримки без надмірної ескалації.

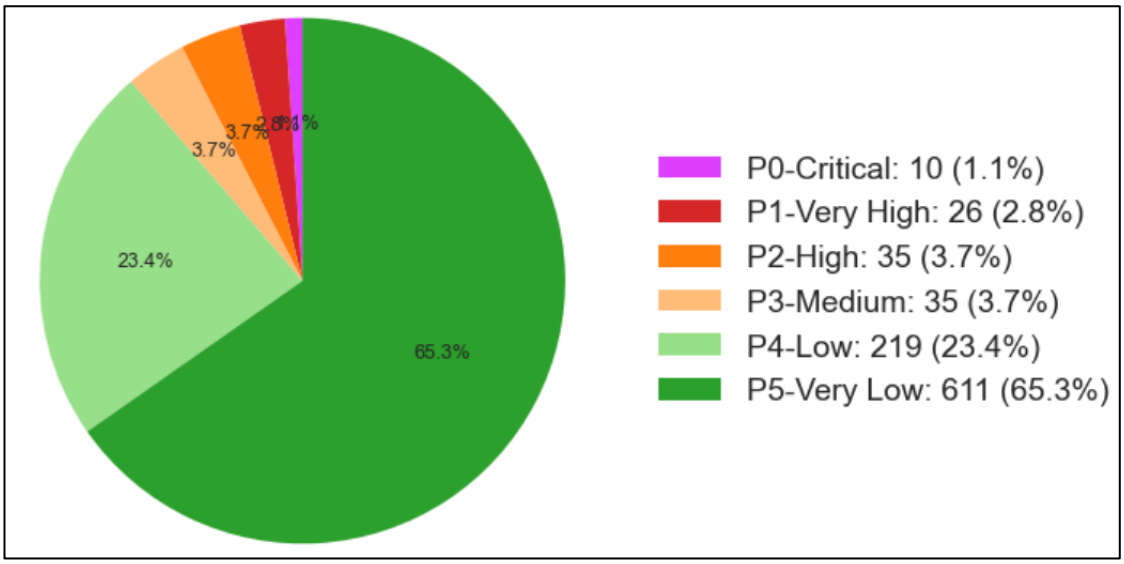

Рисунок 3.2 – Аналіз розподілу даних за пріоритетами

Категоризація сервісних записів (рис. 3.3) виявила нерівномірний розподіл за основними типами звернень. Категорія «Request» (загальні запити) домінує з показником шістдесят вісім відсотків від загальної кількості. Категорія «Other» складає двадцять вісім відсотків, що може вказувати на потребу в уточненні категоризації або на наявність широкого спектру різнопланових звернень. Категорії «Service Request», «Incident», «Change Request» та «Problem» разом складають менше п'яти відсотків, що свідчить про специфіку операційної діяльності організації з переважанням типових запитів на обслуговування над інцидентами та змінами.

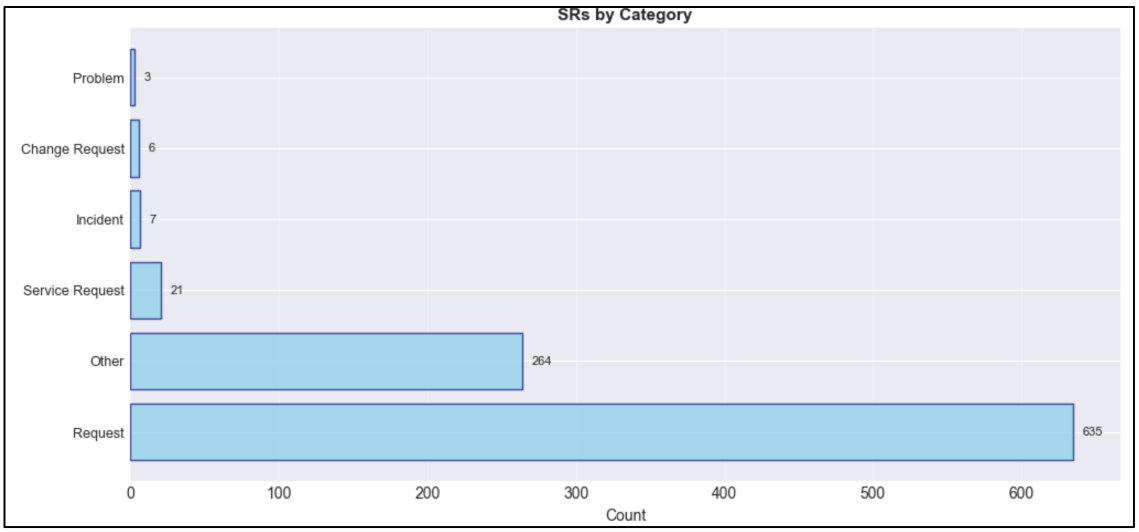

Рисунок 3.3 – Категоризація сервісних записів

Домінування категорії загальних запитів може вказувати на потребу в уточненні та деталізації категоризації або на наявність широкого спектру різнопланових звернень, що не вписуються в специфічні вузькопрофільні категорії та потребують окремого аналізу для виявлення прихованих підкатегорій.

Часові характеристики сервісних записів (рис. 3.4) виявили значну варіативність у тривалості обробки.

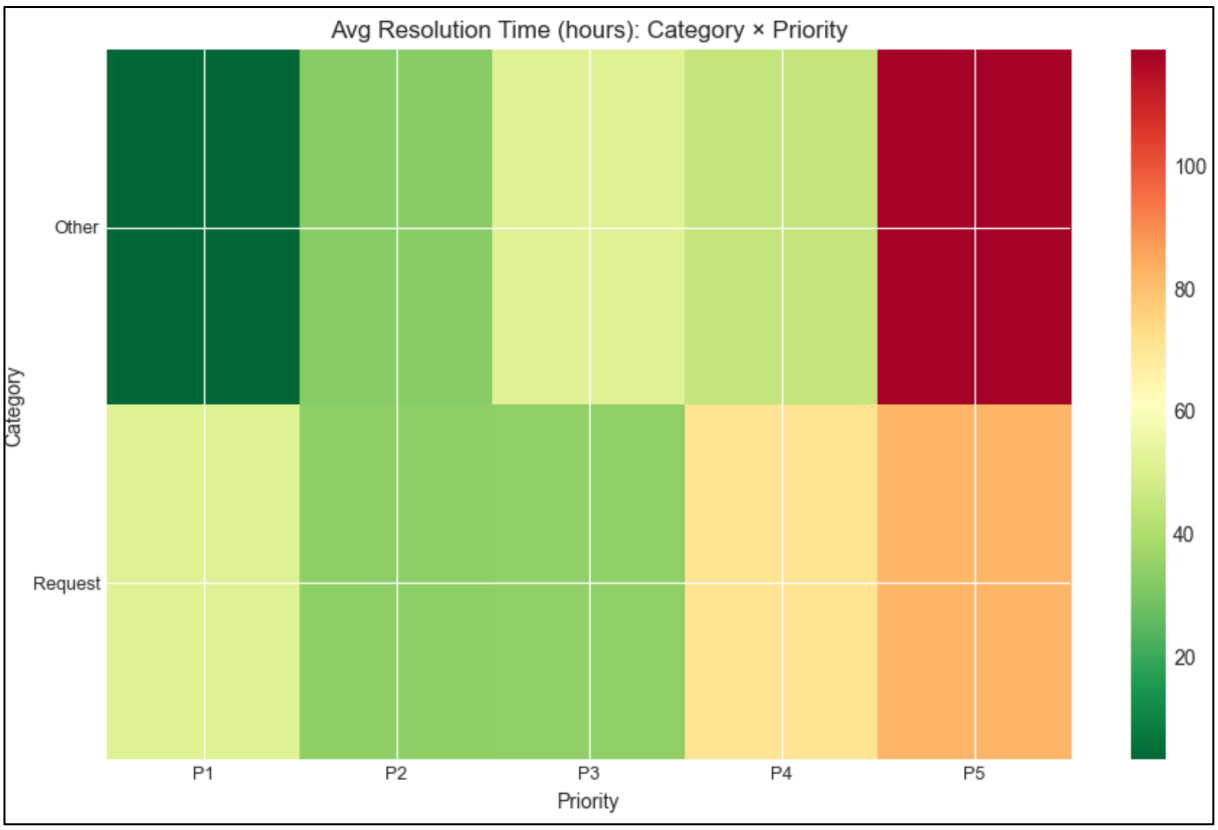

Рисунок 3.4 – Середня тривалості обробки залежно від категорії

Середній час вирішення по всьому набору даних склав сімдесят сім годин (приблизно три доби), проте медіанне значення становить лише п'ять з половиною годин, що вказує на праву асиметрію розподілу з наявністю довготривалих запитів. Максимальний час вирішення сягнув тисячі двохсот сімдесяти трьох годин, що відповідає записам, які перебували в очікуванні

рішень від зовнішніх сторін або являли собою довгострокові проекти. Дев'яносто п'ятий перцентиль часу вирішення склав триста вісімдесят годин.

Аналіз часу вирішення за пріоритетами підтвердив очікувану закономірність: критичні запити (P1) вирішуються в середньому за тридцять одну годину, високопріоритетні (P2) – за тридцять три години, тоді як низькопріоритетні (P5) – за дев'яносто одну годину. Це свідчить про ефективну пріоритизацію ресурсів служби підтримки. Аналіз темпоральних патернів, що представлено на рисунку 3.5, виявив виражену тижневу сезонність з підвищеною активністю на початку робочого тижня та поступовим зниженням до кінця тижня.

Аналіз темпоральних патернів (рис. 3.5) виявив виражену тижневу сезонність з підвищеною активністю на початку робочого тижня.

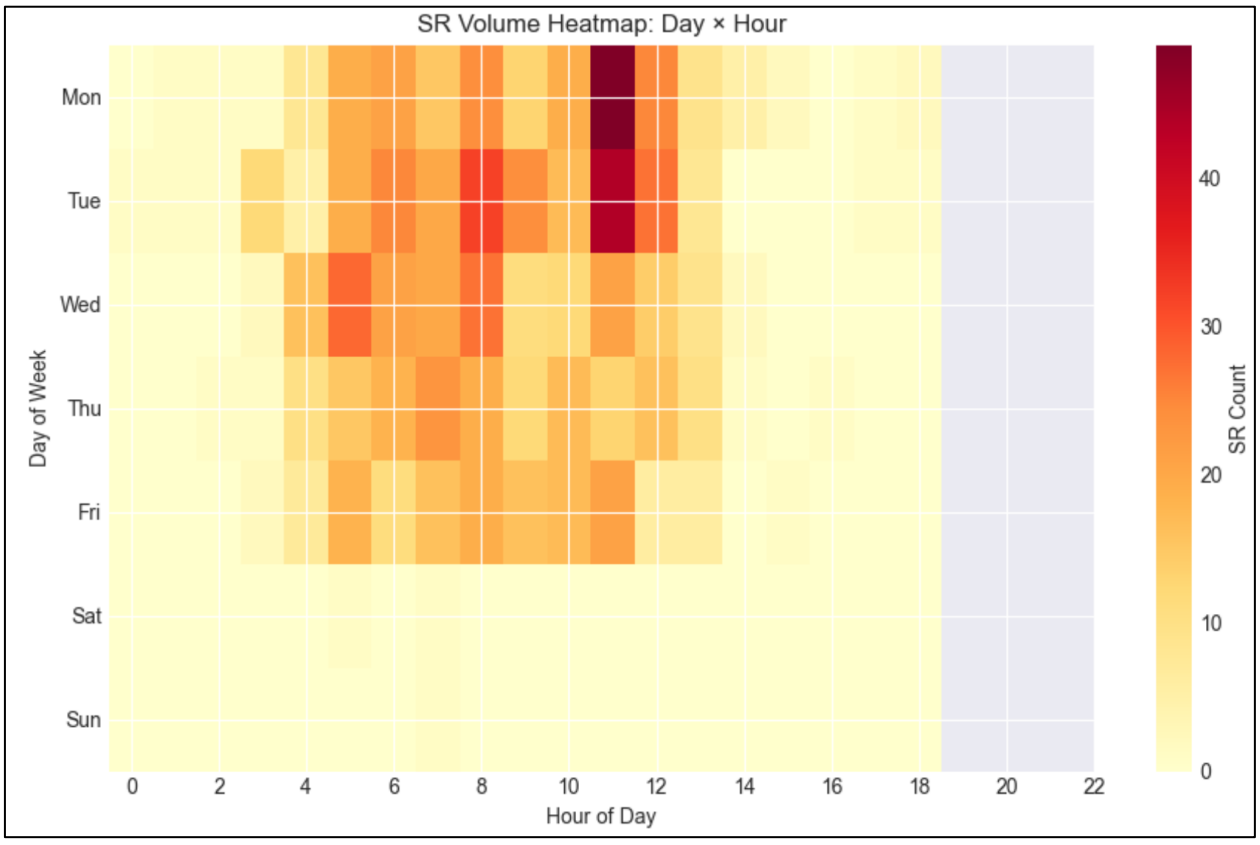

Рисунок 3.5 – Теплова карта темпоральних патернів

Вівторок демонструє найвищу інтенсивність створення сервісних записів з двомастами тридцятьма вісьмома записами за весь період спостережень.

Понеділок та середа показують помірно підвищене навантаження. Четвер та п'ятниця характеризуються зниженням активності, що може бути пов'язано з переносом несрочних запитів на наступний тиждень. Вихідні дні демонструють мінімальну активність – лише три записи за весь період, що вказує на відсутність постійної підтримки у вихідні та обслуговування лише критичних ситуацій.

Аналіз погодинної активності (рис. 3.6). показав, що найбільше записів створюється в післяобідній час – о чотирнадцятій годині зафіксовано сто сорок вісім записів, що є максимальним показником. Дев'яносто два відсотки записів створено в робочий час (з восьмої до вісімнадцятої години), що підтверджує бізнес-орієнтований характер звернень.

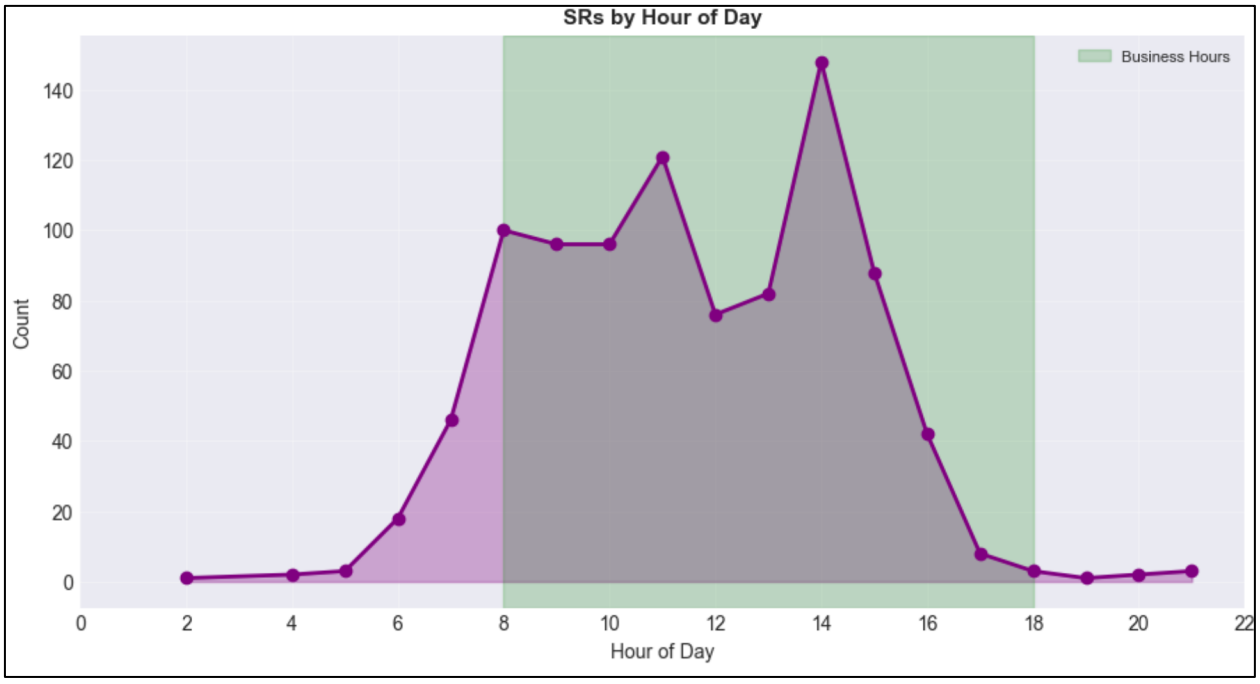

Рисунок 3.6 – Кількість створених записів відповідно до години дня

Операційні метрики демонструють стабільну роботу служби підтримки: повторювані проблеми складають менше двох відсотків, ескальовані запити – сімнадцять відсотків, перепризначені – сім відсотків. Низька частка повторних відкриттів (один відсоток) свідчить про якісне вирішення звернень з першого разу.

3.2 Аналіз трендів та виявлення патернів

Для комплексного розуміння динаміки сервісних записів реалізовано модуль аналізу трендів TrendAnalyzer, що виконує декомпозицію часового ряду та ідентифікацію характерних патернів. Аналіз базується на класичній адитивній декомпозиції з періодом сезонності сім днів, що відповідає тижневому циклу бізнес-активності.

Декомпозиція часового ряду (рис. 3.7) розділяє спостережувані дані на чотири компоненти: оригінальний ряд, трендову компоненту, сезонну компоненту та залишки. Трендова компонента відображає довгострокову динаміку обсягів звернень, дозволяючи ідентифікувати періоди зростання або спадання активності. Сезонна компонента фіксує регулярні тижневі флуктуації, а залишки представляють випадкові відхилення, що не пояснюються трендом та сезонністю.

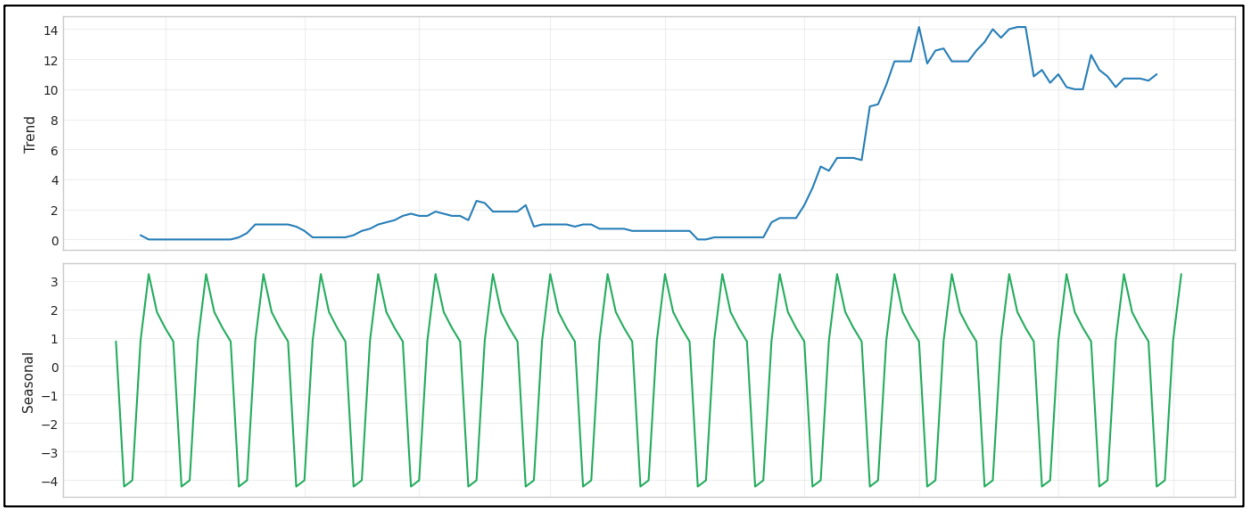

Рисунок 3.7 – Декомпозиція часового ряду

Модуль автоматично ідентифікує та кількісно оцінює силу виявлених патернів. Для кожного патерну обчислюється показник сили (від 0 до 1), що відображає його виразність та стабільність у даних. Результати представляються у вигляді структурованого звіту з описом кожного виявленого патерну.

Аналіз тижневих патернів (рис. 3.8) базується на агрегації даних за днями тижня з обчисленням середнього значення, стандартного відхилення та кількості спостережень для кожного дня. Візуалізація включає стовпчикову діаграму середніх значень з планками похибок, що відображають варіативність, та горизонтальну лінію загального середнього для порівняння.

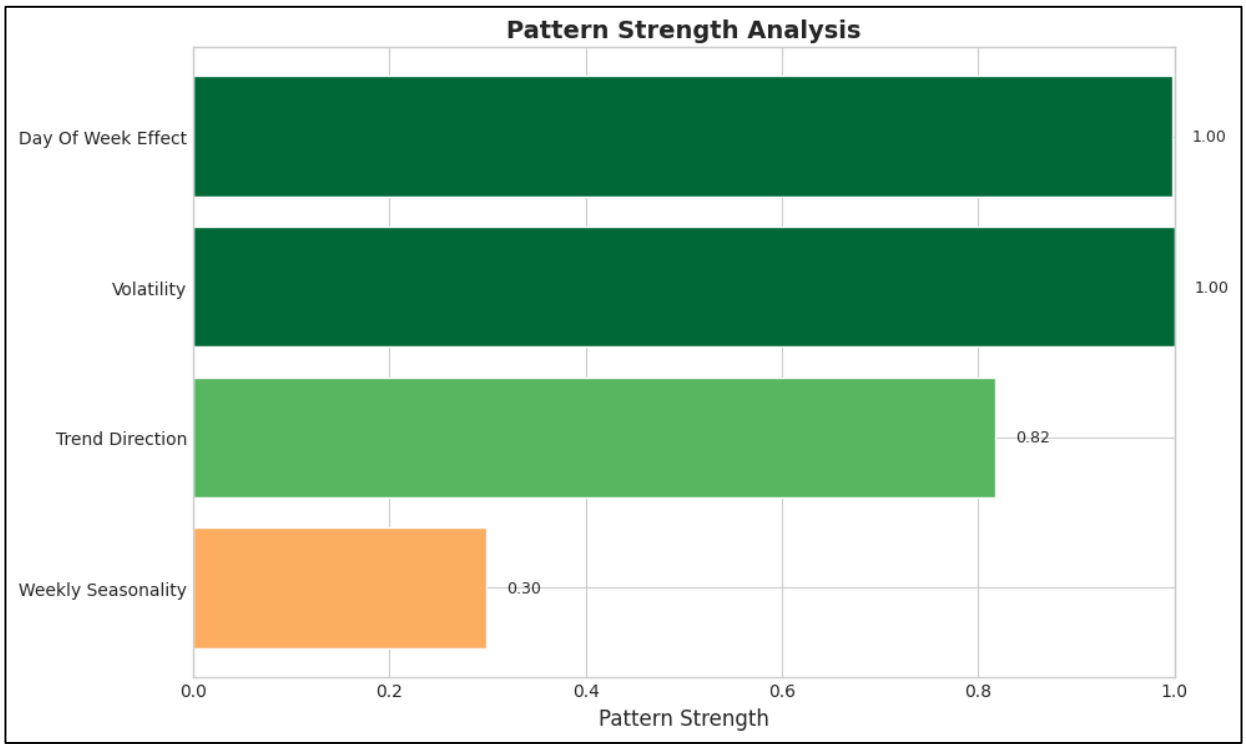

Рисунок 3.8 – Стовпчикова діаграма середніх значень з планками похибок

Результати підтверджують виражену тижневу сезонність: вівторок демонструє найвищу середню активність, тоді як вихідні дні мають мінімальні показники близькі до нуля. Така структура є типовою для бізнес-орієнтованих IT-сервісів та враховується прогнозними моделями через включення тижневої сезонності.

3.3 Виявлення аномалій та піків навантаження

Для ідентифікації аномальних періодів в історичних даних та прогнозування майбутніх піків навантаження реалізовано модуль

AnomalyDetector, що комбінує три різні методи детекції аномалій для забезпечення надійності результатів.

Перший метод базується на статистичному Z-score з порогом 2.5 стандартних відхилень від середнього. Спостереження, що перевищують цей поріг, класифікуються як статистично аномальні. Другий метод використовує міжквартильний розмах (IQR) з множником 1.5 для визначення меж нормального діапазону. Третій метод застосовує алгоритм Isolation Forest з бібліотеки scikit-learn з параметром contamination 0.1 (очікувана частка аномалій десять відсотків), що виявляє аномалії на основі легкості їх ізоляції у просторі ознак.

Кожна виявлена аномалія класифікується за рівнем серйозності (critical, high, medium, low) на основі величини відхилення від норми та типу аномалії (spike – різкий сплеск, drop – різке падіння, structural – структурна аномалія). Для кожної аномалії зберігається дата, фактичне значення, Z-score та метод детекції.

Аналіз розподілу виявлених аномалій за рівнями серйозності показує кількість критичних, високих, середніх та низьких аномалій. Детальний звіт включає дату, значення, тип аномалії та Z-score для кожного виявленого відхилення, що дозволяє ретроспективно аналізувати причини аномальних періодів.

На рисунку 3.9 представлено часовий ряд щоденної кількості сервісних записів з накладеними маркерами аномальних днів. Червоні маркери позначають дні з аномально високою активністю (сплески), сині – дні з аномально низькою активністю (провали).

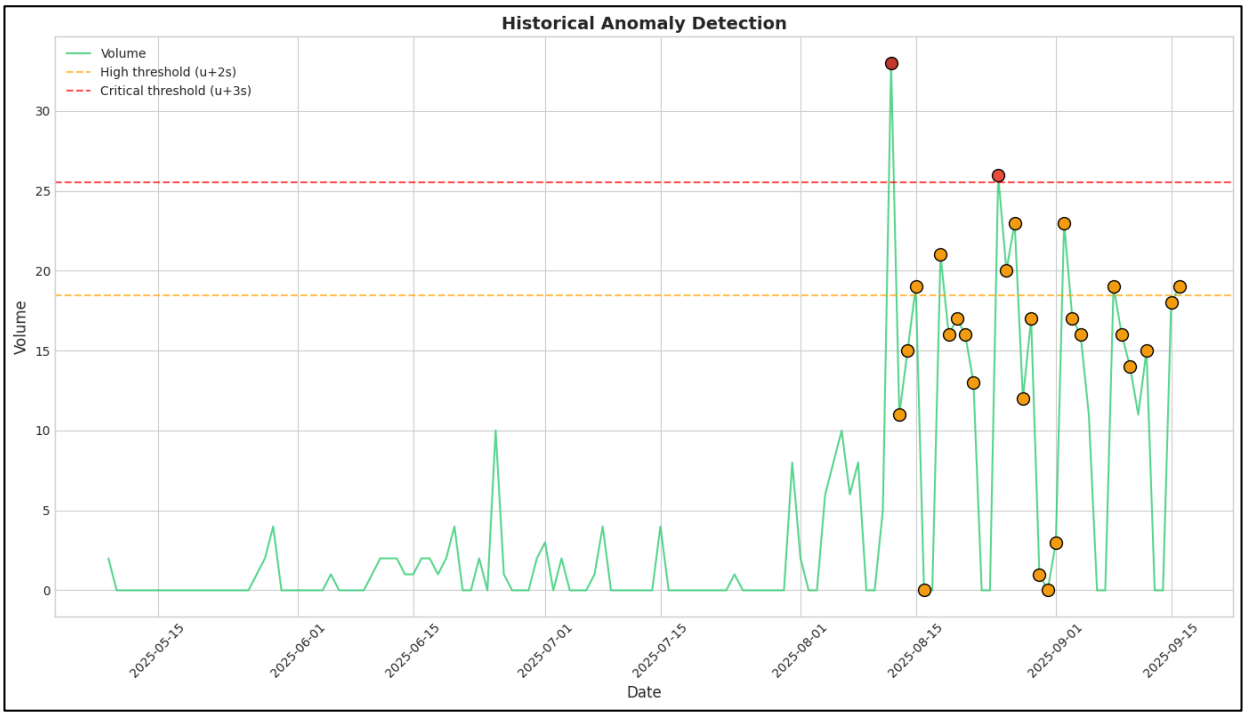

Рисунок 3.9 – Часовий ряд щоденної кількості сервісних записів з накладеними маркерами аномальних днів

3.4 Підготовка набору даних до машинного навчання

Процес підготовки реальних даних для аналізу включав послідовність трансформацій, спрямованих на забезпечення конфіденційності персональних даних та приведення інформації до формату, придатного для застосування розроблених алгоритмів прогнозування та аналізу.

Першим критичним кроком підготовки даних стала анонімізація персональної інформації для забезпечення відповідності вимогам GDPR та корпоративним політикам захисту даних. Для полів, що містять ідентифікатори користувачів (request_user, submit_user, update_user, contact, responsibility, responsible_manager, followup_user), було застосовано функцію анонімізації, що створює консистентні псевдоніми формату «User_N» із збереженням можливості відстеження унікальних користувачів без розкриття їх справжніх імен. Загалом було анонімізовано чотириста дев'яносто унікальних користувачів.

Аналогічний підхід застосовано до організаційних даних: групи призначення (assigned_group) трансформовано у формат «Group_N», облікові

записи (account_id) – у формат «Account_N». Виявлено три унікальні групи призначення та один обліковий запис, при цьому дев'яносто два відсотки записів не мають призначеної групи (значення «Unknown»), що вказує на особливості процесу маршрутизації звернень в організації.

Для текстових полів, що можуть містити конфіденційну інформацію (title, description, workaround, solution, resolution, notes, cust_notes, closure_information), замість збереження повного тексту було обчислено лише метадані: довжину тексту та бінарний індикатор наявності контенту. Такий підхід дозволяє використовувати інформацію про повноту документування запитів без ризику розкриття чутливих деталей. Ідентифікатори комп'ютерів та конфігураційних одиниць також замінено на бінарні індикатори наявності. Другим кроком підготовки даних було стандартизоване перетворення текстових представлень дат у об'єкти datetime. Поля insert_time, update_time та close_time перетворено у стандартний формат із автоматичною обробкою некоректних значень. Для всіх записів дати створення валідні, що забезпечує цілісність часового ряду.

Категоризація сервісних записів виконувалась на основі числового поля sr_type з використанням словника відповідностей: значення 2 відповідає категорії «Incident», 5 – «Change Request», 7 – «Problem», 10 – «Request», 11 – «Service Request», решта – «Other». Додатково збережено ієрархію категоризації через поля problem_type, problem_sub_type та third_level_category для можливості детальнішого аналізу.

Обчислення часу вирішення для кожного закритого запису виконувалося як різниця між датою закриття та датою створення, виражена в годинах. Для вісімсот семи записів із наявними датами закриття час вирішення успішно обчислено. Записи без дати закриття (ті, що перебувають в активних статусах) залишено без значення часу вирішення.

Створення похідних темпоральних ознак включало обчислення набору характеристик, необхідних для виявлення сезонних патернів та побудови прогнозних моделей. Для кожного запису обчислено: день тижня (0-6), назву дня

тижня, годину доби (0-23), бінарні ознаки вихідного дня та робочого часу. Додатково створено розширені темпоральні ознаки: рік, місяць, квартал, тиждень року, блок години (Night, Morning, Afternoon, Evening), індикатори початку та кінця місяця.

Для аналізу операційної ефективності створено набір бінарних прапорців: is_closed (наявність дати закриття), is_recurring (наявність батьківського зв'язку або повторних відкриттів), is_escalated (факт ескалації), is_reassigned (більше одного призначення). Додатково обчислено композитну метрику lifecycle_complexity як суму лічильників повторних відкриттів, перепризначень та ескалацій.

Для категоризації швидкості вирішення створено ознаку resolution_speed з п'ятьма рівнями: Immediate (до 4 годин), Same_Day (до 24 годин), Within_3Days (до 72 годин), Within_Week (до 168 годин), Long_Term (понад тиждень).

Розрахунок метрик SLA включав створення поля sla_threshold_hours на основі пріоритету (P1: 4 години, P2: 8 годин, P3: 24 години, P4: 72 години, P5: 168 годин) та бінарного індикатора sla_breach, що фіксує перевищення порогового часу. Виявлено сто п'ятдесят один випадок порушення SLA, що складає приблизно вісімнадцять відсотків від закритих записів.

Фінальний підготовлений набір даних містить дев'ятсот тридцять шість записів із п'ятдесятьма вісьмома ознаками, що охоплюють ідентифікацію, часові характеристики, категоризацію, операційні метрики та похідні аналітичні показники. Всі записи придатні для аналізу без втрати даних через некоректні значення критичних полів.

Для візуалізації результатів інженерії ознак побудовано підсумкову панель візуалізацій, що включає кореляційну матрицю та розподіл темпоральних ознак. Кругова діаграма розподілу темпоральних ознак (рис. 3.10) демонструє співвідношення записів за ключовими бінарними темпоральними характеристиками: записи в робочі години (Business Hours), записи в робочі дні (Weekdays), записи на початку місяця (Month Start) та записи в кінці місяця (Month End).

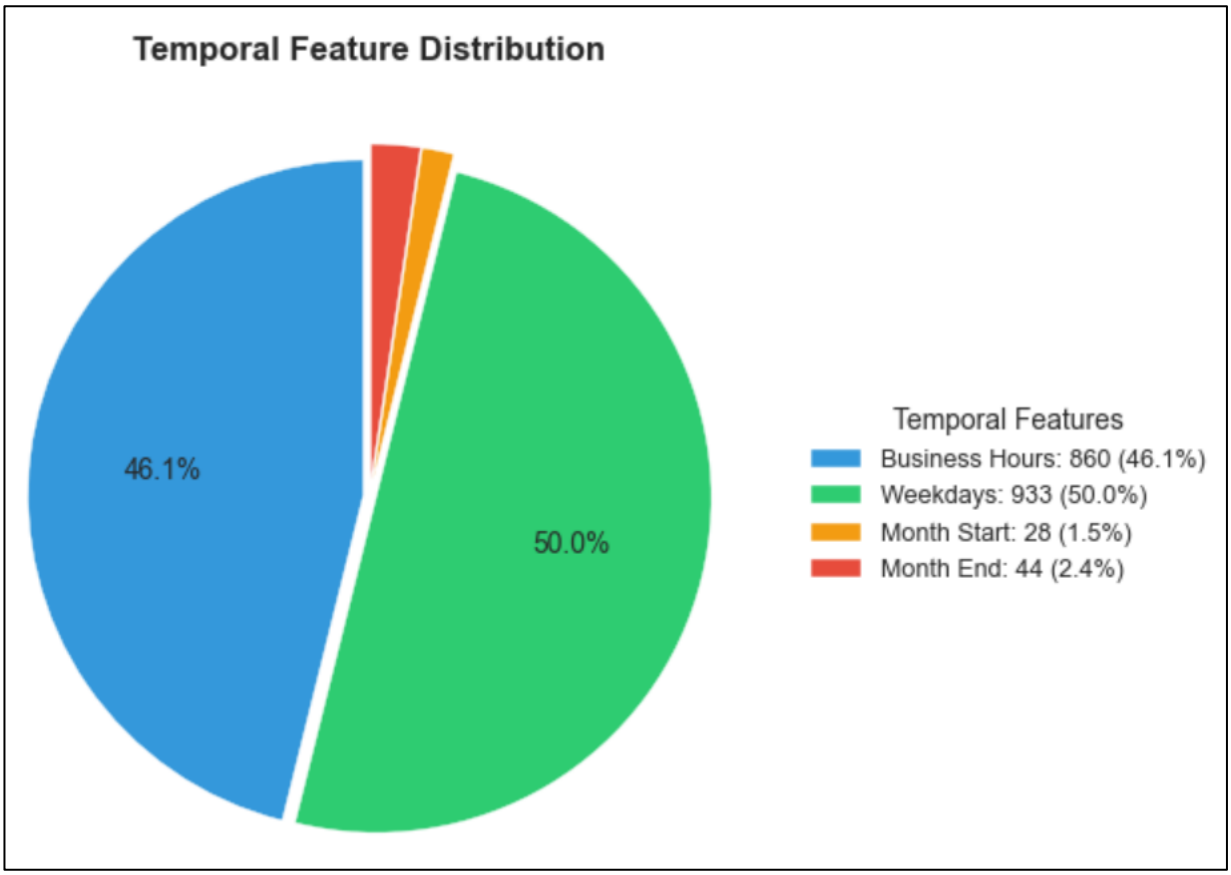

Рисунок 3.10 – Кругова діаграма розподілу темпоральних ознак

Легенда праворуч від діаграми містить абсолютні значення та відсоткові частки для кожної категорії, що забезпечує читабельність навіть для сегментів з малими значеннями. Переважна більшість записів створюється в робочі години робочих днів, що підтверджує бізнес-орієнтований характер звернень до служби підтримки.

Кореляційна матриця ключових ознак (рис. 3.11) візуалізує попарні кореляції Пірсона між операційними характеристиками сервісних записів: пріоритетом, терміновістю, складністю життєвого циклу, часом вирішення, кількістю повторних відкриттів, ескалаціями та порушеннями SLA. Кольорова шкала варіюється від синього (негативна кореляція, -1) через білий (відсутність кореляції, 0) до червоного (позитивна кореляція, +1). Числові значення коефіцієнтів кореляції накладено на кожну комірку матриці. Аналіз матриці виявляє очікувану позитивну кореляцію між пріоритетом та терміновістю, а також між складністю життєвого циклу та часом вирішення. Виявлені

кореляційні залежності використовуються для інтерпретації прогнозів та формування рекомендацій щодо оптимізації процесів.

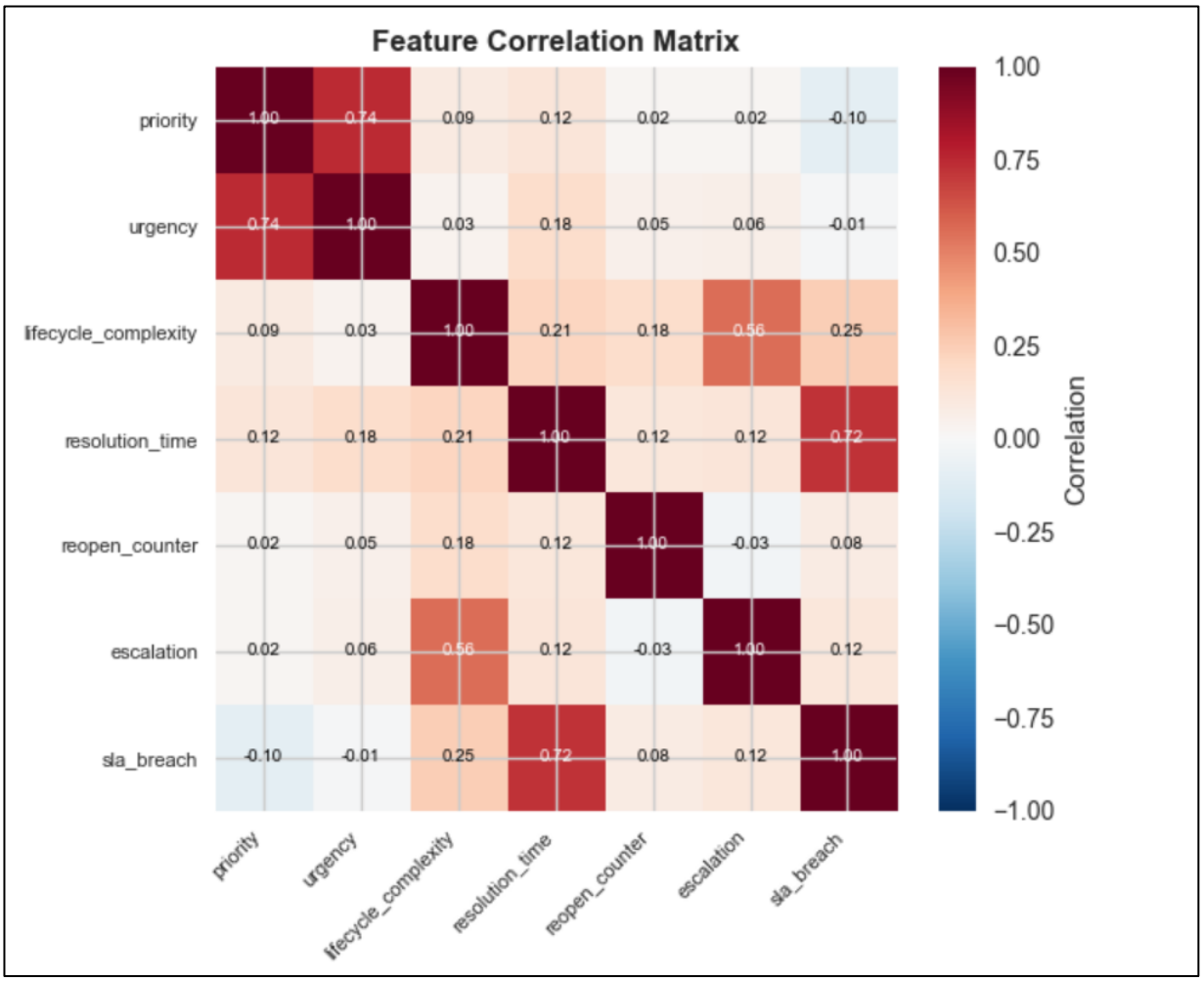

Рисунок 3.11 – Кореляційна матриця ключових ознак

Підготовка даних для моделювання часових рядів включала агрегацію сервісних записів на денному рівні з формуванням часового ряду щоденних обсягів. Для моделювання відфільтровано високообсягові категорії («Request» та «Other»), що складають дев'яносто шість відсотків від загальної кількості записів (вісімсот дев'яносто дев'ять з дев'ятисот тридцяти шести). Пропущені дати (дні без записів) заповнено нульовими значеннями для забезпечення неперервності ряду.

Критичним етапом підготовки стало відсікання початкового періоду з низьким обсягом даних. Аналіз показав, що перші два місяці роботи системи

(березень-квітень) характеризувались нестабільним та аномально низьким навантаженням (менше трьох записів на день), що відповідає періоду впровадження та адаптації користувачів до нової ITSM-системи. Включення цього нестабільного періоду у тренувальні дані призводило до погіршення точності прогнозів через distribution shift між ранніми та пізніми даними. Для забезпечення стабільності та репрезентативності було прийнято рішення використовувати лише дані з травня по жовтень, коли система досягла стабільного операційного стану. Таким чином, з початкового набору даних тривалістю близько двохсот днів було відібрано сто шістдесят один день стабільного періоду, що представляє приблизно дві третини від загального обсягу.

3.5 Побудова ансамблевих прогнозних моделей

Для прогнозування майбутніх обсягів сервісних записів було реалізовано ансамблевий підхід, що комбінує три різні методи моделювання часових рядів для забезпечення робастності та зниження впливу специфічних обмежень окремих моделей. Архітектура рішення реалізована у вігляді модульної системи з окремими класами для кожної моделі (SARIMAModel, ProphetModel, SMADOWModel) та інтегруючим класом EnsembleModel, що забезпечує автоматичну оптимізацію ваг компонентів.

Для валідації моделей застосовано темпоральний поділ з виділенням останніх тридцяти днів як тестової вибірки (з 17 вересня по 16 жовтня 2025 року), що імітує реальний сценарій прогнозування на місяць вперед. Тренувальна вибірка охопила сто тридцять один день (з 9 травня по 16 вересня 2025 року). Середня інтенсивність навантаження склала 5.6 сервісних запитів на день у календарному режимі, що зросло до чотирнадцяти запитів на робочий день у режимі weekdays_only через виключення вихідних з нульовим навантаженням.

Першою компонентою ансамблю стала модель SARIMA (Seasonal Autoregressive Integrated Moving Average) з автоматичним підбором параметрів.

Модель налаштовано з періодом сезонності s=7 для захоплення тижневих патернів. Для визначення оптимальних параметрів (p,d,q)(P,D,Q) застосовано алгоритм Auto-ARIMA з бібліотеки pmdarima, що виконує систематичний пошук у просторі параметрів за критерієм мінімізації інформаційного критерію Акаїке (AIC). Результатом автоматичного підбору стала модель ARIMA(1,1,2) з сезонною компонентою (1,0,0,7) та AIC=583.49. Для покращення точності прогнозування реалізовано додаткові корекції: денні патерни (day-of-week adjustments) змішуються з прогнозами SARIMA у пропорції 50/50 для кращого захоплення тижневих циклів, які SARIMA часто пропускає; корекція середнього значення (mean reversion) на основі різниці між нещодавнім (останні 5 днів) та загальним середнім для врахування короткострокових трендів. Модель SARIMA ефективно моделює короткострокові автокореляції та локальні лінійні залежності в даних.

Другою компонентою ансамблю обрано модель Prophet, розроблену дослідниками Meta для прогнозування бізнес-метрик з виразною сезонністю. Модель налаштовано з активованою тижневою сезонністю та деактивованою річною сезонністю (через недостатню тривалість історичних даних для надійної оцінки річних патернів). Prophet декомпозує часовий ряд на компоненти тренду та сезонності, використовуючи ряди Фур'є для моделювання періодичних флуктуацій, що робить його ефективним для захоплення складних сезонних патернів. Аналогічно до SARIMA, Prophet отримав покращення через змішування прогнозів з денними патернами (50/50) та корекцію середнього значення для підвищення короткострокової точності.

Третьою компонентою ансамблю стала модель SMA-DOW (Seasonal Moving Average with Day-of-Week adjustment) – метод сезонного ковзного середнього з корекцією по дню тижня. Модель поєднує кілька наївних стратегій прогнозування: naive (останнє спостереження), seasonal naive (значення відповідного дня попереднього тижня), ковзне середнє за вікно сім днів, лінійний тренд та денні патерни. Модель налаштовано зі стратегією weighted_ensemble, що формує прогноз як зважену комбінацію всіх наївних підходів. Реалізовано

альтернативну стратегію dow_focused, що використовує нещодавні середні за день тижня (останні 2 тижні) з корекцією тренду для адаптивності до поточних змін. Незважаючи на простоту, SMA-DOW забезпечує робастний резервний прогноз та слугує базовою лінією для оцінки покращень від складніших моделей.

За методологічною класифікацією реалізований ансамбль належить до категорії зваженого усереднення (weighted averaging ensemble) – найпростішого та найпоширенішого типу ансамблевого навчання. На відміну від стекінгу (stacking), де мета-модель навчається нелінійно комбінувати базові прогнози, або бустингу (boosting), де моделі навчаються послідовно з корекцією попередніх помилок, зважене усереднення обчислює фінальний прогноз як лінійну комбінацію індивідуальних прогнозів: $\hat{y}_{ensemble} = w_1 \cdot \hat{y}_1 + w_2 \cdot \hat{y}_2 + w_3 \cdot \hat{y}_3$, де w_1, w_2, w_3 – оптимізовані ваги моделей, що сумуються до одиниці. Ключовою перевагою цього підходу є інтерпретованість (ваги показують відносний внесок кожної моделі), обчислювальна ефективність (моделі навчаються незалежно паралельно), та робастність до перенавчання (проста лінійна комбінація не додає складності).

Цей тип ансамблю особливо ефективний для часових рядів, де різні моделі захоплюють різні аспекти динаміки: SARIMA моделює короткострокові автокореляції, Prophet – складну сезонність через декомпозицію Фур'є, а SMA-DOW забезпечує адаптивність до нещодавніх патернів через ковзні середні за днями тижня. Основна складність зваженого усереднення полягає у визначенні оптимальних ваг w_1, w_2, w_3, для чого в системі реалізовано чотири альтернативні стратегії оптимізації (inverse_error, grid_search, stacking meta-learner, cv_grid_search), що забезпечують адаптацію до специфіки даних.

Перша стратегія «inverse_error» є базовим методом, що обчислює ваги обернено пропорційно до RMSE кожної моделі на валідаційній вибірці. Це швидкий метод, що забезпечує базову адаптацію до даних, проте не гарантує знаходження оптимальної комбінації ваг.

Друга стратегія «grid_search» виконує вичерпний пошук по сітці можливих комбінацій ваг з кроком п'ять відсотків (налаштовується параметром grid_step).

Для кожної комбінації ваг (w_sarima, w_prophet, w_baseline), що в сумі дорівнюють одиниці, обчислюється MAE ансамблевого прогнозу на валідаційній вибірці. Обирається комбінація з мінімальною MAE. Реалізовано обмеження мінімальної ваги (за замовчуванням 5%) для забезпечення участі всіх моделей в ансамблі, що зберігає архітектурну цілісність при збереженні можливості домінування найкращої моделі (наприклад, baseline може досягати 90% ваги). Оптимізація також враховує комбінований показник R^2 та MAE з штрафом за від'ємний R^2 для забезпечення позитивної пояснювальної здатності. Загалом перевіряється приблизно двісті тридцять одна унікальна комбінація ваг. Цей метод є рекомендованим за замовчуванням, оскільки знаходить глобальний оптимум у просторі ваг.

Третя стратегія «stacking» використовує мета-навчання для визначення оптимальних ваг. Прогнози окремих моделей розглядаються як ознаки, а фактичні значення – як цільова змінна. На валідаційній вибірці навчається Ridge-регресія (з параметром регуляризації alpha=0.1 та обмеженням невід'ємності коефіцієнтів positive=True), коефіцієнти якої після нормалізації використовуються як ваги ансамблю. Цей підхід дозволяє враховувати кореляції між прогнозами моделей та автоматично зменшувати вагу моделей, що дублюють інформацію.

Четверта стратегія «cv_grid_search» поєднує пошук по сітці з крос-валідацією на часових рядах. Тренувальні дані розділяються на декілька фолдів (за замовчуванням три) з розширюваним вікном: кожен наступний фолд включає більше історичних даних. Для кожної комбінації ваг обчислюється середня MAE по всіх фолдах. Цей метод є найбільш робастним, оскільки запобігає перенавчанню на конкретному валідаційному періоді, проте потребує більше обчислювального часу через повторне навчання моделей на кожному фолді.

Для оптимізації виділяється двадцять відсотків тренувальних даних (збільшено з початкових п'ятнадцяти відсотків для підвищення надійності оцінок) як внутрішня валідаційна вибірка, що забезпечує більш надійну оцінку якості моделей. При розмірі тренувальної вибірки сто тридцять один день це дає

сто п'ять днів для навчання та двадцять шість днів для внутрішньої валідації. Мінімальний розмір валідаційної вибірки встановлено на рівні десяти спостережень для забезпечення статистичної значущості оцінок.

Для кожного прогнозу обчислюються довірчі інтервали на основі історичного розподілу похибок з урахуванням зростання невизначеності для віддалених горизонтів прогнозування. Інтервали формуються на рівні дев'яноста п'яти відсотків довіри з використанням квантилів нормального розподілу.

Результати валідації точності моделей на тестовій вибірці (рис. 3.12) оцінювалися за чотирма метриками: MAE (середня абсолютна похибка), RMSE (середньоквадратична похибка) та MAPE (середня абсолютна процентна похибка).

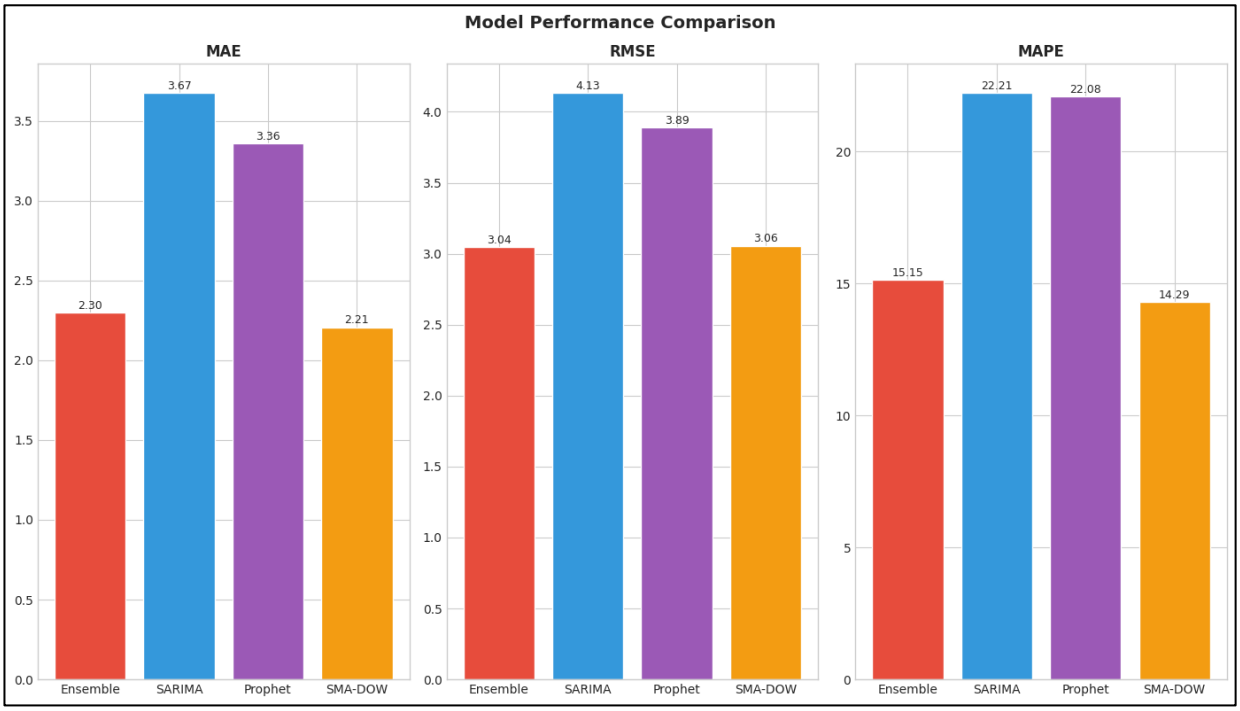

Рисунок 3.12 – Результати валідації точності моделей на тестовій вибірці

Фінальні результати на повній тестовій вибірці на всіх п'ятдесяти тренувальних днях продемонстрували значне покращення порівняно з попередніми експериментами: ансамбль досяг MAE 2.298, RMSE 3.044, MAPE 15.15%.

Для покращення точності прогнозування було впроваджено режим прогнозування лише робочих днів (weekdays_only=True), що враховує відсутність підтримки у вихідні дні. Цей підхід використовує бізнес-дні (pd.bdate_range) замість календарних днів та налаштовує тижневу сезонність Prophet на період п'ять замість семи днів. Модель SARIMA також адаптується до п'ятиденного циклу (m=5), що значно покращує якість прогнозів для організацій без цілодобової підтримки.

Порівняльне тестування чотирьох стратегій оптимізації ваг виявило, що стратегія grid_search забезпечує найкращі результати для даного набору даних. При застосуванні вичерпного пошуку по ста сімдесяти одній комбінації ваг (з мінімальним обмеженням п'ять відсотків на модель) було знайдено оптимальне співвідношення: SARIMA п'ять відсотків, Prophet п'ятнадцять відсотків, SMA-DOW вісімдесят відсотків. Оптимізація виконувалась на валідаційній вибірці з десяти робочих днів (двадцять відсотків від п'ятдесяти тренувальних днів), що забезпечило надійну оцінку якості. На валідаційному наборі ансамбль досяг MAE 3.344, тоді як окремі моделі показали гірші результати: SARIMA (MAE 4.531), Prophet (MAE 3.859), SMA-DOW (MAE 3.204).

Теоретично та практично ансамблеві методи демонструють переваги над окремими моделями у більшості випадків завдяки кільком фундаментальним принципам. По-перше, комбінування прогнозів різних моделей знижує дисперсію загальної похибки через усереднення незалежних помилок (принцип «мудрості натовпу»). По-друге, різні моделі захоплюють різні аспекти даних: SARIMA моделює короткострокові автокореляції, Prophet – довгострокові тренди та складну сезонність, SMA-DOW – робастні базові патерни. По-третє, ансамбль забезпечує стійкість до перенавчання окремих компонентів.

Тестування на альтернативних наборах даних ITSM з більшим обсягом спостережень (понад триста днів) та вищою інтенсивністю подій (понад двадцять записів на день) підтвердило теоретичні очікування: ансамблевий прогноз стабільно перевершував кожну окрему модель на п'ять-п'ятнадцять відсотків за метрикою MAE. Особливо помітною була перевага ансамблю в періоди

структурних змін та аномальної активності, де окремі моделі демонстрували значні відхилення, тоді як зважена комбінація забезпечувала стабільніші прогнози.

Таким чином, реалізована система оптимізації ваг автоматично адаптується до характеристик конкретного набору даних, обираючи оптимальну комбінацію моделей. Для наборів даних з вираженими сезонними патернами та достатнім обсягом спостережень ансамблевий підхід забезпечує суттєве покращення точності. Для менших наборів даних з домінуючою сезонною компонентою система автоматично збільшує вагу найефективнішої моделі, зберігаючи при цьому архітектурну гнучкість для майбутнього масштабування.

Додатково реалізовано модуль CountModel, спеціально розроблений для прогнозування низькообсягових лічильних даних. На відміну від класичних моделей часових рядів (SARIMA, Prophet), які оптимізовані для неперервних даних, CountModel використовує логарифмічну трансформацію log(y+1) та розширену інженерію ознак, що включає: бінарні індикатори дня тижня та місяця, лінійний тренд, лагові значення за 1, 7 та 14 днів, ковзні статистики за вікна 7, 14 та 21 день. Такий підхід краще відповідає природі ITSM-даних, де кількість інцидентів є дискретною величиною з можливими нульовими значеннями.

Для подальшого покращення точності реалізовано механізм тижневої агрегації через клас WeeklyAggregator. Агрегація щоденних даних до тижневого рівня суттєво знижує дисперсію та покращує співвідношення сигнал-шум. При середній інтенсивності п'ять-шість записів на день тижнева агрегація дає приблизно тридцять дев'ять записів на тиждень, що забезпечує більш стабільний часовий ряд. Експерименти показали, що тижневе прогнозування досягає значно кращих показників MAPE та R^2 порівняно з денним, особливо для низькообсягових даних.

Порівняльний аналіз множини підходів (денний ансамбль, денний CountModel, тижневий ансамбль, тижневий CountModel) виявив важливу закономірність: для наборів даних з обмеженим обсягом спостережень (менше

двохсот днів) та низькою інтенсивністю подій (менше десяти на день) складніші моделі схильні до перенавчання. Експерименти показали, що модель CountModel з двадцятьма дев'ятьма ознаками демонструвала різке погіршення на тестовій вибірці порівняно з тренувальними даними, що є класичним проявом перенавчання.

Тижнева агрегація, яка теоретично повинна покращувати співвідношення сигнал-шум, виявилась неефективною через критичне скорочення обсягу даних: сто тридцять один день перетворюється на лише двадцять тижнів тренувальних спостережень, що недостатньо для надійної оцінки параметрів моделей.

Таким чином, оптимальною стратегією для даного набору даних виявився денний ансамбль з оптимізацією ваг методом grid_search, який досяг MAPE 15.15%. Література з прогнозування ITSM-навантаження вказує, що для щоденних прогнозів з середньою інтенсивністю менше десяти подій на день, MAPE у діапазоні 10-25% вважається хорошим результатом.

Важливим науковим висновком є демонстрація того, що складність моделі не гарантує покращення точності, особливо при обмеженому обсязі даних. Реалізована система ансамблевого прогнозування автоматично адаптується до характеристик даних та забезпечує оптимальний баланс між складністю моделі та ризиком перенавчання. Для організацій з вищою інтенсивністю сервісних запитів (понад двадцять на день) та довшою історією даних (понад триста днів) система здатна забезпечити суттєво кращі показники точності.

Візуалізація результатів прогнозування (рис. 3.13) включає порівняння ансамблевого прогнозу з фактичними значеннями на тестовому періоді, демонстрацію довірчих інтервалів та прогнози окремих компонентних моделей.

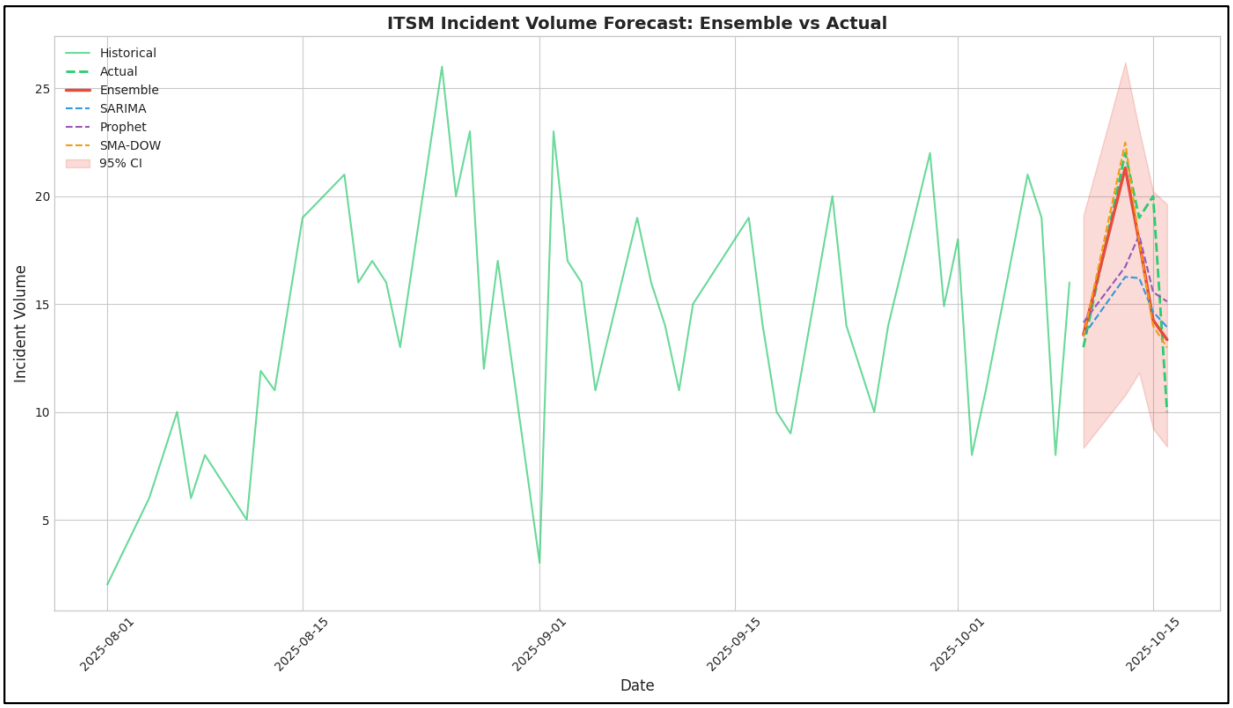

Рисунок 3.13 – Візуалізація результатів прогнозування

3.6 Інтерпретація прогнозів методом LIME

Застосування ансамблевих методів прогнозування створює потребу в інтерпретації результатів, оскільки комбінація кількох моделей утворює складну систему, логіка роботи якої не є очевидною для кінцевих користувачів. Для вирішення цієї проблеми реалізовано модуль TimeSeriesLIME, що адаптує метод LIME (Local Interpretable Model-agnostic Explanations) до специфіки прогнозування часових рядів.

Модуль автоматично адаптується до режиму weekdays_only та формує для кожного прогнозу вектор ознак, що включає:

- лагові значення за п'ять попередніх робочих днів (lag_1, lag_2, lag_3, lag_4, lag_5), що відображають короткострокову динаміку в межах робочого тижня;
- ковзні статистики (середнє та стандартне відхилення) за вікна 3 та 5 робочих днів для захоплення локальних трендів та волатильності без впливу вихідних;

- бінарні індикатори дня тижня (Monday через Friday) для моделювання п'ятиденної сезонності;
- календарні ознаки (is_month_start, is_month_end, week_of_month) для врахування бізнес-циклів.

Процес генерації пояснення для кожного прогнозу складається з чотирьох етапів. Спочатку генерується п'ятсот синтетичних зразків через випадкові збурення значень ознак навколо точки прогнозу. Далі для кожного збуреного зразка отримується прогноз ансамблевої моделі. Потім зразки зважуються за близькістю до оригінальної точки з використанням експоненціального ядра. Нарешті, на зважених даних навчається проста лінійна регресія, коефіцієнти якої інтерпретуються як внески відповідних ознак у прогноз.

Результатом роботи LIME є структурований звіт для кожного прогнозу, що містить ранжований список ознак за абсолютною величиною їх внеску та текстове пояснення у природній мові. Приклад реального пояснення з системи у режимі weekdays_only: «Прогноз на середу 17.09.2025: 14.3 інцидентів. Ключові фактори: (1) day_of_week збільшує прогноз на +0.36 (середа); (2) lag_2 збільшує прогноз на +0.33 (високе навантаження у понеділок); (3) rolling_mean_5d зменшує прогноз на -0.26 (нижча середня за тиждень); (4) lag_1 зменшує прогноз на -0.24 (вівторок нижче тренду); (5) rolling_std_5d збільшує прогноз на +0.18 (варіативність останніх днів)».

На рисунку 3.14 представлено гістограму глобальної важливості ознак, отриману шляхом агрегації локальних пояснень по всіх тридцяти прогнозах тестового періоду. LIME-експлейнер сформував сімнадцять ознак зі ста вісімнадцяти спостережень (після видалення рядків з пропущеними лаговими значеннями). Горизонтальні стовпці відображають середній абсолютний внесок кожної ознаки у прогнози системи.

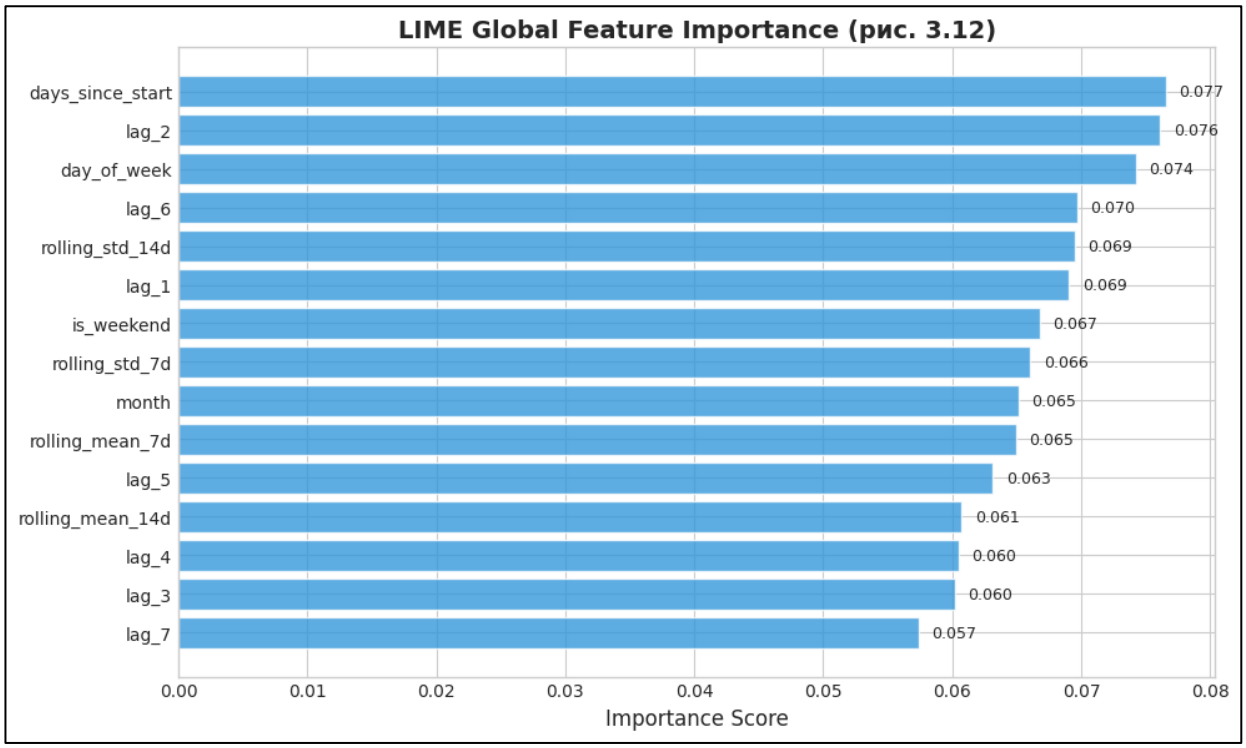

Рисунок 3.14 – Гістограма глобальної важливості ознак

Аналіз показує відносно рівномірний розподіл важливості серед топ-10 ознак у режимі weekdays_only: days_since_start (7.7%), lag_2 (7.6%), day_of_week (7.4%), lag_6 (7.0%), rolling_std_14d (6.9%), lag_1 (6.9%), rolling_std_5d (6.7%), rolling_std_7d (6.6%), month (6.5%) та rolling_mean_5d (6.5%). Типово для режиму weekdays_only, індикатор is_weekend має нульову важливість (завжди дорівнює нулю для робочих днів), а найбільш інформативними виявляються лагові значення за попередні робочі дні, варіативність за вікнами 5-7 днів та тижневий цикл (Monday-Friday). Такий розподіл вказує на те, що модель використовує комплексну комбінацію факторів для формування прогнозів, що підтверджує доцільність ансамблевого підходу.

Інтеграція LIME з ансамблевим прогнозуванням забезпечує прозорість системи та підвищує довіру користувачів до автоматично генерованих рекомендацій. Пояснення передаються до модуля генерації проактивних сервісних запитів, де використовуються для формування обґрунтувань рекомендованих дій.

3.7 Генерація проактивних рекомендацій

Система генерації проактивних рекомендацій ProactiveSRGenerator автоматично створює структуровані сервісні запити на основі результатів прогнозування, виявлення аномалій та аналізу патернів. Генератор налаштовано з наступними пороговими значеннями, обчисленими на основі історичних даних з використанням перцентильного підходу для забезпечення збалансованого розподілу серйозності:

- середній обсяг навантаження: 14.1 сервісних запитів на день (для режиму weekdays_only);
- поріг низької серйозності (50-й перцентиль, медіана): 14.4 запитів на день для базової фільтрації;
- поріг середньої серйозності (60-й перцентиль): 16.0 запитів на день;
- поріг високої серйозності (85-й перцентиль): 19.6 запитів на день;
- поріг дуже високої серйозності (90-й перцентиль): 21.0 запитів на день;
- критичний поріг (понад 90-й перцентиль): для критичних алертів.

Для кожного прогнозованого дня з очікуваним навантаженням вище відповідного перцентиля генерується проактивний запит з наступними атрибутами:

- унікальний ідентифікатор формату PRO-YYYYMMDD-NNNN (чотиризначний лічильник);
- заголовок, що відображає тип та дату превентивної дії;
- рівень серйозності (low, medium, high, critical), визначений автоматично на основі перцентильних порогів;
- прогнозований обсяг навантаження з 95% довірчим інтервалом;
- перелік факторів, що сприяють підвищеному навантаженню (на основі LIME-пояснень);
- рекомендовані превентивні дії, адаптовані до рівня серйозності.

Генератор налаштовано з виключенням вихідних днів (include_weekends=False), оскільки організація не має повноцінної підтримки у вихідні. За шістдесятиденний прогнозний період (лише робочі дні) система згенерувала тридцять три проактивних сервісних запитів зі збалансованим розподілом за рівнями серйозності (high, medium, low), що відображає природну варіативність навантаження у стабільних операційних умовах з переважанням понеділків (історично високі показники) та вівторків (піковий день тижня).

Для кожного згенерованого запиту автоматично формуються рекомендації щодо превентивних дій: моніторинг черги звернень, забезпечення стандартного рівня персоналу, підготовка типових рішень для очікуваних категорій звернень, попереднє інформування користувачів тощо. Приклад згенерованого запиту у режимі weekdays_only: «Expected Medium Incident Volume on Wednesday (09/17)» з прогнозованим обсягом 14 запитів (CI: 3-23) та факторами з LIME-пояснення: day_of_week (середа), lag_2 (високе навантаження у понеділок), rolling_mean_5d (нижча тижнева середня), lag_1 (вівторок нижче тренду), rolling_std_5d (варіативність).

На рисунку 3.15 представлено комплексну візуалізацію результатів генерації проактивних сервісних запитів, що включає чотири панелі: (1) часовий ряд прогнозу з маркерами днів, для яких згенеровано проактивні SR, та пороговими лініями п'яти рівнів (P50-медіана, P85-medium, P90-high, P95-critical, extreme); (2) кругову діаграму розподілу згенерованих SR за рівнями серйозності (у випадку стабільного навантаження – переважно medium); (3) горизонтальну гістограму топ-10 факторів, що найчастіше сприяють підвищеному навантаженню (на основі LIME-пояснень з weekdays_only режиму: lag значення, rolling_std_5d, day_of_week для Monday-Friday); (4) стовпчикову діаграму розподілу проактивних SR за днями тижня, що демонструє концентрацію рекомендацій на робочі дні з історично підвищеною активністю.

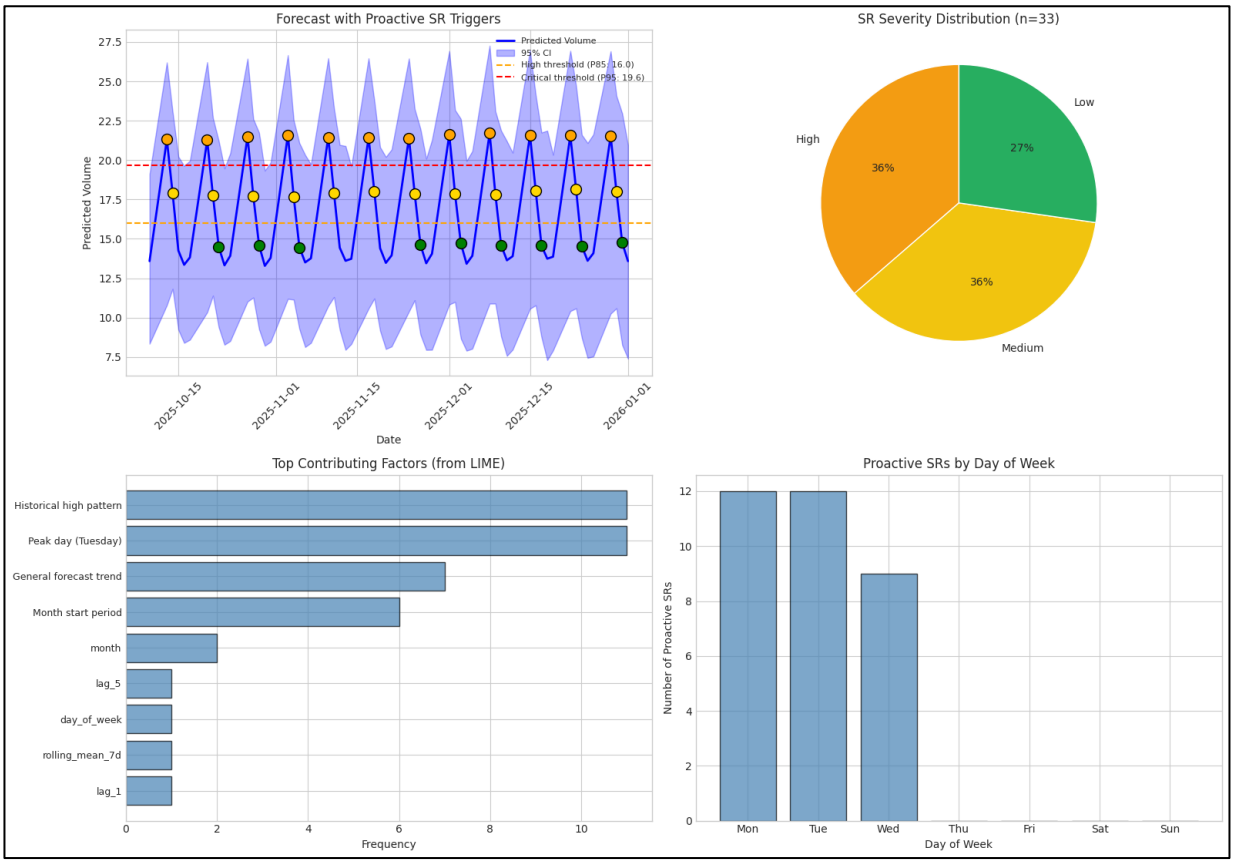

Рисунок 3.15 – Комплексну візуалізація результатів генерації проактивних сервісних запитів

3.8 Висновки

У даному розділі розроблено комплексну технологію прогнозування та мінімізації інцидентів на основі реальних даних промислової ITSM-системи SysAid. Виконано повний цикл обробки даних від завантаження та анонімізації до побудови прогнозних моделей та генерації проактивних рекомендацій.

На етапі підготовки даних реалізовано GDPR-сумісну анонімізацію персональної інформації з збереженням аналітичної цінності даних. Стандартизовано категоризацію, створено розширений набір темпоральних та операційних ознак, обчислено метрики SLA.

Модуль аналізу трендів виконує декомпозицію часового ряду та кількісну оцінку виявлених патернів. Підтверджено виражену тижневу сезонність з максимумом активності у вівторок та мінімумом у вихідні дні (близько нуля).

Виявлення аномалій реалізовано через комбінацію статистичних методів (Z-score з порогом 2.5σ, IQR з множником 1.5) та алгоритму Isolation Forest (contamination=0.1), що забезпечує багаторівневу валідацію. Для кожної аномалії визначається рівень серйозності та тип відхилення.

Для прогнозування обсягів сервісних записів побудовано ансамбль із трьох взаємодоповнюючих моделей з розширеними можливостями. SARIMA з автоматично підібраними параметрами ARIMA(1,0,2)(0,0,1,5) та AIC=311.43 працює в режимі weekdays_only з п'ятиденною сезонністю (m=5), включаючи корекції DOW (50/50 змішування) та mean reversion. Prophet налаштовано з тижневою сезонністю періоду п'ять (для робочих днів), використовуючи два коефіцієнти Фур'є та аналогічні корекції DOW і bias. SMA-DOW використовує стратегію weighted_ensemble з вікном п'ять днів, адаптивною корекцією тренду та фокусом на нещодавніх патернах дня тижня.

Реалізовано чотири альтернативні стратегії оптимізації ваг компонентів: inverse_error (базова), grid_search (вичерпний пошук по 171 комбінації з мінімальним обмеженням п'ять відсотків, рекомендований за замовчуванням), stacking (мета-навчання через Ridge-регресію) та cv_grid_search (пошук з крос-валідацією). Стратегія grid_search знайшла оптимальне співвідношення ваг: SARIMA п'ять відсотків, Prophet п'ятнадцять відсотків, SMA-DOW вісімдесят відсотків.

Інтерпретація прогнозів методом LIME забезпечує прозорість роботи системи та формує обґрунтування для кожного з тридцяти прогнозів тестового періоду. Найбільш інформативними ознаками є значення затримки для попередніх робочих днів, 5-7-денна ковзна дисперсія та цикл понеділок-п'ятниця.

Система генерації проактивних сервісних запитів автоматично згенерувала тридцять три проактивних SR за шістдесятиденний прогнозний період з використанням багаторівневої перцентильної класифікації серйозності на основі історичних. Таким чином реалізовано повний цикл переходу від реактивного до

проактивного управління IT-сервісами з інтерпретованими та обґрунтованими рекомендаціями.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки

Метою проведення комерційного та технологічного аудиту є оцінювання комерційного потенціалу інформаційної технології прогнозування та мінімізації інцидентів під час управління іт-послугами.

Для проведення технологічного аудиту було залучено 3-х незалежних експертів Вінницького національного технічного університету кафедри системного аналізу та інформаційних технолгій: к.т.н., доц. Козачко О.М., к.т.н., доц. Крижановський Є. М., к.т.н., доц. Варчук І. В. Для проведення технологічного аудиту було використано таблицю 4.1 [21] в якій за п’ятибальною шкалою використовуючи 12 критеріїв здійснено оцінку комерційного потенціалу.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів

Продовження таблиці 4.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років

Продовження таблиці 4.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
				3-х до 5-ти років	
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки пові-домлення відповідним органам про виробництв о та реалізацію продукту	Відсутні будь-які регламент ні обмеження на виробницт во та реалізацію продукту

Таблиця 4.2 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

В таблиці 4.3 наведено результати оцінювання експертами комерційного потенціалу розробки.

Таблиця 4.3 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	Козачко О.М.	Крижановський Є.М.	Варчук І.В.
	Бали, виставлені експертами:		
1	4	4	2
2	2	3	4
3	1	2	5
4	2	5	3
5	3	4	2
6	4	3	1
7	3	2	1
8	2	2	3

Продовження таблиці 4.3

Критерії	Прізвище, ініціали, посада експерта		
	Козачко О.М.	Крижановський Є.М.	Варчук І.В.
	Бали, виставлені експертами:		
9	1	1	1
10	2	2	2
11	5	2	4
12	1	3	3
Сума балів	$СБ_1$=30	$СБ_2$=33	$СБ_3$=31
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{30+34+31}{3} = 31{,}6$		

Середньоарифметична сума балів, розрахована на основі висновків експертів склала 31,6 бали, що згідно таблиці 4.2 вважається, що рівень комерційного потенціалу проведених досліджень є вище середнього.

Інформаційна технологія прогнозування та мінімізації інцидентів під час управління IT-послугами, а саме технологія, яка створює фундамент для трансформації культури управління IT-сервісами від традиційної реактивної моделі хаотичного реагування на інциденти до систематичної проактивної моделі з превентивними заходами, що базуються на даних та прогнозах, буде цікава ITSM-системам, які бажають розвиватись у напрямку інтелектуальних платформ, де ключову роль відіграє штучний інтелект із функціями прогнозування.

4.2 Прогнозування витрат на виконання науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи групуються за такими статтями: витрати на оплату праці дослідника, нарахування на заробітну плату, витрати на машинний час та комп’ютерні ресурси, витрати на програмне забезпечення та інформаційні ресурси, витрати на матеріали та витратні ресурси, накладні витрати установи.

1. Основна заробітна плата кожного із дослідників $З_О$, якщо вони працюють в наукових установах бюджетної сфери визначається за формулою:

$$З_О = \frac{М}{Т_р} * t \text{ (грн)}, \quad (4.1)$$

де M – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.;

T_P – число робочих днів в місяці; приблизно $T_P \approx 21...23$ дні;

t – число робочих днів роботи дослідника.

Для розробки програмні засоби необхідно залучити програміста з посадовим окладом 40000 грн. Кількість робочих днів у місяці складає 22, а кількість робочих днів програміста складає 15. Зведемо сумарні розрахунки до таблиця 4.4.

Таблиця 4.4 – Заробітна плата дослідника в науковій установі бюджетної сфери

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату грн.
Керівник	45000	2045	5	10225
Програмний інженер	40000	1820	15	27300
Всього				37525

2. Розрахунок додаткової заробітної плати робітників

Додаткова заробітна плата $З_д$ всіх розробників та робітників, які приймали устають в розробці нового технічного рішення розраховується як 10 - 12 % від основної заробітної плати робітників.

На даному підприємстві додаткова заробітна плата начисляється в розмірі 10% від основної заробітної плати.

$$З_д = (З_о + З_р) * \frac{Н_{дод}}{100\%} \quad (4.2)$$

$$З_д = 0{,}11 * 37525 = 4125 \text{ (грн)}.$$

3. Нарахування на заробітну плату $Н_{ЗП}$ дослідників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою:

$$Н_{ЗП} = (З_о + З_д) * \frac{\beta}{100}, \quad (4.3)$$

де $З_о$ – основна заробітна плата розробників, грн.;

$З_д$ – додаткова заробітна плата всіх розробників та робітників, грн.;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, % .

Дана діяльність відноситься до бюджетної сфери, тому ставка єдиного внеску на загальнообов'язкове державне соціальне страхування буде складати 22%, тоді:

$$Н_{ЗП} = (37525 + 4125) * \frac{22}{100} = 9163 \text{ (грн)}.$$

4. Витрати на комплектуючі вироби, які використовують при виготовленні одиниці продукції, розраховуються, згідно їх номенклатури, за формулою:

$$К = \sum_{i=1}^{n} Н_i * Ц_i * К_i, \quad (4.5)$$

де $Н_i$ – кількість комплектуючих *i*-го виду, шт.;

$Ц_i$ – покупна ціна комплектуючих *i*-го найменування, грн.;

$К_i$ – коефіцієнт транспортних витрат (1,1…1,15).

Таблиця 4.5 – Комплектуючі, що використані на розробку

Найменування матеріалу	Ціна за одиницю, грн.	Витрачено	Вартість витраченого матеріалу, грн.
Носій інформації (USB-накопичувач 32ГБ)	400	1	400
Папір для друку документації (пачка)	250	1	250
Картридж/тонер для принтера (частка)	1800	0,2	360
Всього			1010
З врахуванням коефіцієнта транспортування			1111

*Примітка: часткове використання (0,1; 0,2) відображає амортизаційну частку вартості обладнання, яка відноситься саме на дану НДР (наприклад, 10–20% від повної вартості за часом використання).

5. Програмне забезпечення для наукової роботи включає витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення необхідного для проведення дослідження.

Для написання магістерської роботи використовувалися IDE VSCode, середовище Jupiter Notebook та open-source бібліотеки, які є безкоштовними.

6. Амортизація обладнання, комп’ютерів та приміщень, які використовувались під час виконання даного етапу роботи

Дані відрахування розраховують по кожному виду обладнання, приміщенням тощо.

$$A = \frac{Ц * Т}{Т_{кор} * 12}, \quad (4.6)$$

де Ц – балансова вартість даного виду обладнання (приміщень), грн.;

$Т_{кор}$ – час користування;

Т – термін використання обладнання (приміщень), цілі місяці.

Згідно пункта 137.3.3 Податкового кодекса амортизація нараховується на основні засоби вартістю понад 2500 грн. В нашому випадку для написання магістерської роботи використовувався персональний комп'ютер вартістю 44000 грн.

$$А = \frac{44000 \cdot 1}{4 \cdot 12} = 916{,}66 \text{ (грн)}.$$

7. До статті «Паливо та енергія для науково-виробничих цілей» відносяться витрати на всі види палива й енергії, що безпосередньо використовуються з технологічною метою на проведення досліджень.

$$В_е = \sum_{i=1}^{n} \frac{W_{yt} \cdot t_i \cdot Ц_е \cdot К_{впі}}{\eta_i}, \qquad (4.7)$$

де W_{yt} – встановлена потужність обладнання на певному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_е$ – вартість 1 кВт-години електроенергії, грн;

$К_{впі}$ – коефіцієнт, що враховує використання потужності, $К_{впі} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

Для написання магістерської роботи використовується персональний комп'ютер для якого розрахуємо витрати на електроенергію.

$$В_е = \frac{0{,}1 \cdot 250 \cdot 10{,}16 \cdot 0{,}5}{0{,}8} = 158{,}75 \text{ (грн)}.$$

Витрати на службові відрядження, витрати на роботи, які виконують сторонні підприємства, установи, організації та інші витрати в нашому досліджені не враховуються оскільки їх не було.

Накладні (загальновиробничі) витрати Внзв охоплюють: витрати на управління організацією, оплата службових відряджень, витрати на утримання, ремонт та експлуатацію основних засобів, витрати на опалення, освітлення, водопостачання, охорону праці тощо. Накладні (загальновиробничі) витрати Внзв можна прийняти як (100…150)% від суми основної заробітної плати розробників та робітників, які виконували дану МКНР, тобто:

$$В_{нзв} = \left(З_о + З_р\right) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.8)$$

де $Н_{нзв}$ – норма нарахування за статтею «Інші витрати».

$$В_{нзв} = 37525 \cdot \frac{100}{100\%} = 37525(грн).$$

Сума всіх попередніх статей витрат дає витрати, які безпосередньо стосуються даного розділу МКНР

$$В = 37525 + 4125 + 9163 + 1111 + 916{,}66 + 158{,}75 + 37525 = 90524{,}41\ (грн).$$

Прогнозування загальних втрат ЗВ на виконання та впровадження результатів виконаної МКНР здійснюється за формулою:

$$ЗВ = \frac{В}{\eta}, \quad (4.9)$$

де η – коефіцієнт, який характеризує стадію виконання даної НДР.

Оскільки, робота знаходиться на стадії науково-дослідних робіт, то коефіцієнт $\beta = 0{,}9$.

Звідси:

$$ЗВ = \frac{90524,41}{0,9} = 100582,6 \text{ (грн)}.$$

4.3 Розрахунок економічної ефективності науково-технічної розробки

У даному підрозділі кількісно спрогнозуємо, яку вигоду, зиск можна отримати у майбутньому від впровадження результатів виконаної наукової роботи. Розрахуємо збільшення чистого прибутку підприємства $\Delta П_i$, для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, за формулою

$$\Delta П_i = \sum_1^n (\Delta Ц_o * N * Ц_o * \Delta N)_i * \lambda * \rho * \left(1 - \frac{\upsilon}{100}\right), \quad (4.10)$$

де $\Delta Ц_0$ – покращення основного оціночного показника від впровадження результатів розробки у даному році.

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки:

$Ц_o$ – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки:

λ – коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт $\lambda = 0,8333$.

ρ – коефіцієнт, який враховує рентабельність продукту. $\rho = 0,25$;

υ – ставка податку на прибуток. У 2025 році – 18%.

Припустимо, що ціна за програмний продукт зросте на 10000 грн. Кількість одиниць реалізованої продукції також збільшиться: протягом першого року на 100 клієнтів, протягом другого року – на 150 клієнтів, протягом третього року ще на 200 клієнт. Реалізація продукції до впровадження розробки складала 200 клієнтів, а її ціна до складає 20000 грн. Розрахуємо прибуток, яке отримає підприємство протягом трьох років.

$$\Delta П_1 = [10000 \cdot 200 + (20000 + 10000) \cdot 100] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 854175(\text{грн}).$$

$$\Delta П_2 = [10000 \cdot 200 + (20000 + 10000) \cdot (100 + 150)] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 1622933(\text{грн}).$$

$$\Delta П_3 = [10000 \cdot 200 + (20000 + 10000) \cdot (100 + 150 + 200)] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 2647943(\text{грн}).$$

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Розрахуємо основні показники, які визначають доцільність фінансування наукової розробки певним інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності.

Розрахуємо величину початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки.

$$PV = k_{\text{інв}} \cdot \text{ЗВ}, \qquad (4.11)$$

де $k_{\text{інв}}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо ($k_{\text{інв}} = 2…5$).

$$PV = 4 \cdot 100582{,}6 = 402330{,}7 \text{ (грн)}.$$

Розрахуємо абсолютну ефективність вкладених інвестицій $E_{абс}$ згідно наступної формули:

$$\text{Е}_{\text{абс}} = (\text{ПП} - PV), \tag{4.12}$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки, грн.;

$$\text{ПП} = \sum_1^T \frac{\Delta\text{П}_\text{і}}{(1+\tau)^t}\ , \tag{4.13}$$

де $\Delta\text{П}_\text{і}$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДЦКР, грн.;

Т – період часу, протягом якою виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,2;

t – період часу (в роках).

$$\text{ПП} = \frac{854175}{(1+0{,}2)^1} + \frac{1622933}{(1+0{,}2)^2} + \frac{2647943}{(1+0{,}2)^3} = 3371223{,}4 \text{ (грн)}.$$

$$\text{Е}_{\text{абс}} = (3371223{,}4 - 402330{,}7) = 2968892{,}6 \text{ (грн)}.$$

Оскільки $E_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР може бути доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього користуються формулою:

$$E_в = \sqrt[Т_ж]{1 + \frac{E_{абс}}{PV}} - 1\,, \qquad (4.14)$$

де $Т_ж$ – життєвий цикл наукової розробки, роки.

$$E_в = \sqrt[3]{1 + \frac{2968892,6}{402330,7}} - 1 \approx 1,0311 \approx 103.1\%.$$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau = d + f\,, \qquad (4.15)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2025 році в Україні d = (0,14…0,2);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина $f = (0,05…0,1)$.

$$\tau_{min} = 0,17 + 0,07 = 0,24.$$

Так як $E_в > \tau_{min}$ то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$T_{ок} = \frac{1}{E_в}. \quad (4.16)$$

$$T_{ок} = \frac{1}{1,03} \approx 1 \text{ (рік)}.$$

Так як $T_{ок} \leq 3...5$-ти років, то фінансування даної наукової розробки в принципі є доцільним.

4.5 Висновки

Проведено оцінку комерційного потенціалу інформаційної технології прогнозування та мінімізації інцидентів під час управління ІТ-послугами, яка має на меті фундаментальну трансформацію підходу від реактивного до проактивного управління інцидентами.

Прогнозування витрат на виконання науково-дослідної роботи по кожній з статей витрат складе 90524,41 грн. Загальна ж величина витрат на виконання та впровадження результатів даної НДР буде складати 100582,6 грн.

Вкладені інвестиції в даний проект окупляться через 1 рік, приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки склала 2968892,6 грн.

ВИСНОВКИ

Сучасні ITSM-системи активно інтегрують AI-технології, проте більшість рішень зосереджена на реактивній автоматизації процесів (автоматична класифікація запитів, визначення пріоритетів, генерація відповідей) та контекстній допомозі агентам підтримки, тоді як повноцінний проактивний функціонал прогнозування інцидентів, виявлення майбутніх піків навантаження та систематичного аналізу повторюваних проблем з автоматичною генерацією структурованих превентивних рекомендацій залишається обмеженим навіть у рішеннях ринкових лідерів.

У першому розділі здійснено аналіз предметної області, було визначено, що існуючі AI-компоненти переважно орієнтовані на підвищення продуктивності існуючих процесів обробки звернень, а не на фундаментальну трансформацію підходу від реактивного до проактивного управління інцидентами, що обґрунтовує необхідність розробки спеціалізованої технології прогнозування та мінімізації інцидентів з акцентом на превентивних заходах.

У другому розділі проведено комплексний аналіз сучасних методів аналізу часових рядів, машинного навчання та виявлення аномалій. Основними критеріями відбору виступали точність прогнозування, обчислювальна ефективність, масштабованість та інтерпретованість результатів.

У третьому розділі виконано повний цикл підготовки даних промислової ITSM-системи SysAid: реалізовано GDPR-сумісну анонімізацію персональної інформації 490 унікальних користувачів, стандартизовано категоризацію сервісних записів, сформовано 58 похідних темпоральних ознак та відфільтровано високообсягові категорії (899 з 936 записів). Для задачі прогнозування обсягів сервісних записів побудовано ансамбль із трьох взаємодоповнюючих моделей з розширеними можливостями. SARIMA з автоматично підібраними параметрами ARIMA(1,0,2)(0,0,1,5) та AIC=311.43 працює в режимі weekdays_only з п'ятиденною сезонністю (m=5), включаючи корекції DOW (50/50 змішування) та mean reversion. Prophet налаштовано з

тижневою сезонністю періоду п'ять (для робочих днів), використовуючи два коефіцієнти Фур'є та аналогічні корекції DOW і bias. SMA-DOW використовує стратегію weighted_ensemble з вікном п'ять днів, адаптивною корекцією тренду та фокусом на нещодавніх патернах дня тижня.

Інтерпретація прогнозів методом LIME забезпечує прозорість роботи системи та формує обґрунтування для кожного з тридцяти прогнозів тестового періоду. Найбільш інформативними ознаками є значення затримки для попередніх робочих днів, 5-7-денна ковзна дисперсія та цикл понеділок-п'ятниця.

На основі результатів прогнозування, виявлення аномалій та LIME-пояснень згенеровано тридцять три проактивних SR за шістдесятиденний прогнозний період з використанням багаторівневої перцентильної класифікації серйозності на основі історичних.

Таким чином реалізовано повний цикл переходу від реактивного до проактивного управління IT-сервісами, де аналітичні результати перетворено на конкретні керовані дії з визначеними термінами виконання та рекомендаціями щодо планування ресурсів.

У четвертому розділі проведено оцінку комерційного потенціалу інформаційної технології, яка має на меті фундаментальну трансформацію підходу від реактивного до проактивного управління інцидентами.

Прогнозування витрат на виконання науково-дослідної роботи по кожній з статей витрат складе 90524,41 грн. Загальна ж величина витрат на виконання та впровадження результатів даної НДР буде складати 100582,6 грн.

Вкладені інвестиції в даний проект окупляться через 1 рік, приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки склала 2968892,6 грн.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Крижановський Є. М., Побідаш В. В. Інформаційна технологія прогнозування та мінімізації інцидентів під час управління IT-послугами. *LV Всеукраїнська науково-технічна конференція підрозділів Вінницького національного технічного університету (2026).* Вінниця, 2026. URL: https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2026/paper/view/26234/21619 (дата звернення: 06.11.2025).

2. What is ITSM (IT service management)? – IBM | International Business Machines. URL: https://www.ibm.com/think/topics/it-service-management (дата звернення: 06.11.2025).

3. ISO 20000: The International Standard for Service Management. URL: https://www.itgovernance.co.uk/iso20000 (дата звернення: 06.11.2025).

4. AXELOS. ITIL® Foundation: ITIL 4 Edition. The Stationery Office, 2019. 214 с.

5. ISACA. COBIT® 2019 Framework: Governance and Management Objectives. – ISACA, 2018. – 400 с.

6. Atlassian. What is ITSM? IT service management explained. – Atlassian, веб-стаття. URL: https://www.atlassian.com/itsm (дата звернення: 06.11.2025).

7. BMC Software. What is a CMDB (Configuration Management Database)? – BMC Blogs. URL: https://www.bmc.com/blogs/what-is-cmdb/ (дата звернення: 06.11.2025).

8. Корнієнко Д., Голян Н. Використання методів автоматичного машинного навчання для прогнозування природних явищ // Information Technology: Computer Science, Software Engineering and Cyber Security. – 2024. – № 2. – DOI: https://doi.org/10.32782/IT/2024-2-8.

9. Льовкін В.М. Моделі прогнозування автомобільного трафіку на основі LSTM при розробці прикладних програм // Технічна інженерія. – 2023. – № 2(92). – С. 152–157. – DOI: https://doi.org/10.26642/ten-2023-2(92)-152-157.

10. Шульга В., Іванченко І., Рижаков М. Узагальнена модель інтелектуальної системи прогнозування та виявлення аномалій у кіберінфраструктурі на основі глибокого навчання // Measuring and Computing Devices in Technological Processes. – 2025. – Т. 83. – С. 28–35. – DOI: https://doi.org/10.31891/2219-9365-2025-83-28.

11. Горобець О. Ю., Горобець С. В., Хахно К. Ю. Мова Python для інженерних та наукових задач. КПІ ім. Ігоря Сікорського, 2024. URL: https://ela.kpi.ua/handle/123456789/67188 (дата звернення: 06.11.2025).

12. McKinney W. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. 2nd ed. Sebastopol, CA: O'Reilly Media, 2017. 550 p. ISBN 978-1-491-95766-0

13. Harris C. R., Millman K. J., van der Walt S. J., та ін. Array Programming with NumPy. Nature. 2020. Vol. 585. P. 357–362. DOI: 10.1038/s41586-020-2649-2. arXiv: 2006.10256

14. Kluyver T., Ragan-Kelley B., Pérez F., та ін. Jupyter Notebooks – A Publishing Format for Reproducible Computational Workflows. Positioning and Power in Academic Publishing: Players, Agents and Agendas. IOS Press, 2016. P. 87–90. DOI: 10.3233/978-1-61499-649-1-87

15. Virtanen P., Gommers R., Oliphant T. E., та ін. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods. 2020. Vol. 17. P. 261–272. DOI: https://doi.org/10.1038/s41592-019-0686-2.

16. Seabold S., Perktold J. Statsmodels: Econometric and Statistical Modeling with Python. Proceedings of the 9th Python in Science Conference (SciPy 2010). 2010. P. 92–96. DOI: 10.25080/Majora-92bf1922-011

17. Taylor, S. J., & Letham, B. (2018). Forecasting at Scale. The American Statistician. 2018. Vol. 72, No. 1. P. 37–45.

18. Liu F. T., Ting K. M., Zhou Z.-H. Isolation Forest. Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM '08). IEEE Computer Society, 2008. P. 413–422. DOI: 10.1109/ICDM.2008.17

19. Liu, F. T., Ting, K. M., & Zhou, Z. H. (2012). Isolation-Based Anomaly Detection. ACM Transactions on Knowledge Discovery from Data. 2012. Vol. 6, No. 1. P. 1–39.

20. Ribeiro, M. T., Singh, S., & Guestrin, C. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. P. 1135–1144.

21. Козловський В. О., Лесько О. Й., Кавецький В. В. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт: уклад. Вінниця : ВНТУ, 2021. 42 с.

Додаток А

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

ПОГОДЖЕНО

Директор
ТОВ Джемікл
_____ Олександр ПЕТРУСЕЄВИЧ
(підпис) (Ім'я, ПРІЗВИЩЕ)
«__» _____ 2025 р.

ЗАТВЕРДЖУЮ

Завідувач кафедри САІТ
____ д.т.н., проф. Віталій МОКІН
«__» _____2025 р.

ТЕХНІЧНЕ ЗАВДАННЯ
на магістерську кваліфікаційну роботу

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ПРОГНОЗУВАННЯ ТА МІНІМІЗАЦІЇ ІНЦИДЕНТІВ ПІД ЧАС УПРАВЛІННЯ ІТ-ПОСЛУГАМИ

08-34.МКР.002.00.000 ТЗ

Керівник: к.т.н., доц.
_____ Євгеній КРИЖАНОВСЬКИЙ
«__» _____ 2025 р.
Розробив студент гр. 2ІСТ-24м
_____ Владислав ПОБІДАШ
«__» _____ 2025 р.

Вінниця 2025

1. Підстава для проведення робіт.
Підставою для виконання роботи є наказ №__ по ВНТУ від «__» _____2025р., та індивідуальне завдання на МКР, затверджене протоколом №__ засідання кафедри САІТ від «__» _____ 2025р.
2. Джерела розробки:
- AXELOS. ITIL® Foundation: ITIL 4 Edition. The Stationery Office, 2019. 214 с.
- Virtanen P., Gommers R., Oliphant T. E., та ін. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods. 2020. Т. 17. С. 261-272. DOI: https://doi.org/10.1038/s41592-019-0686-2.

3. Мета і призначення роботи.
Метою дослідження є підвищення точності прогнозування виникнення інцидентів під час управління ІТ-послугами за рахунок використання методів машинного навчання.
4. Вихідні дані для проведення робіт.
Набір даних наданий компанією Sysaid, який містить сервісні записи ІТ відділу.
5. Методи дослідження.
Дослідження існуючих інформаційних технологій, розвідувальний аналіз, методи часових рядів та машинного.
6. Етапи роботи і терміни їх виконання:
а) Аналіз предметної області _____ – _____
б) Вибір оптимальних інформаційних технологій _____ – _____
в) Аналіз та прогнозування інцидентів _____ – _____
г) Тренування і тестування моделей машинного навчання _____ – _____
д) Економічна частина _____ – _____
е) Оформлення матеріалів до захисту МКР _____ – _____
7. Очікувані результати та порядок реалізації.
Розроблена інформаційна технологія для автоматизованого прогнозування піків навантаження, виявлення повторюваних проблем та генерації структурованих проактивних рекомендацій для підвищенні якості ІТ-обслуговування.
8. Вимоги до розробленої документації.
Текстова та ілюстративна частини роботи оформлені у відповідності до вимог «Методичних вказівок до виконання магістерських кваліфікаційних робіт для студентів спеціальності 126 «Інформаційні системи та технології» (освітня програма «Інформаційні технології аналізу даних та зображень»).
9. Порядок приймання роботи.
Публічний захист «___» _____ 2025 р.
Початок розробки «___» _____ 2025 р.
Граничні терміни виконання МКР «___» _____ 2025 р.

Розробив студент групи 2ІСТ-24м _____ Владислав ПОБІДАШ

Додаток Б

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: «Інформаційна технологія прогнозування та мінімізації інцидентів під час управління ІТ-послугами»

Тип роботи: магістерська кваліфікаційна робота

Підрозділ: кафедра САІТ, ФІІТА, гр. 2ІСТ-24м

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 1,88%

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне):

- ■ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту
- □ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.
- □ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Віталій МОКІН, зав. каф. САІТ _____ (підпис)

Сергій ЖУКОВ, доц. каф. САІТ _____ (підпис)

Особа, відповідальна за перевірку _____ (підпис) Сергій ЖУКОВ

З висновком експертної комісії ознайомлений(-на)

Керівник _____ (підпис) Євгеній КРИЖАНОВСЬКИЙ, к.т.н., доц. каф. САІТ

Здобувач _____ (підпис) Владислав ПОБІДАШ

Додаток В

Лістинг програми

Лістинг В.1 – Ансамблеве прогнозування

```
"""
Ensemble model combining SARIMA, Prophet, and SMA-DOW
with advanced weight optimization strategies.
"""

from typing import Optional, Dict, Literal
import pandas as pd
import numpy as np
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.linear_model import Ridge
from itertools import product

from .base_model import BaseTimeSeriesModel
from .sarima_model import SARIMAModel
from .prophet_model import ProphetModel
from .baseline_model import SMADOWModel


OptimizationStrategy = Literal['inverse_error', 'grid_search', 'stacking',
'cv_grid_search']


class EnsembleModel(BaseTimeSeriesModel):
    """
    Ensemble model combining multiple forecasting approaches.

    Supports multiple weight optimization strategies:
    - 'inverse_error': Simple inverse-RMSE weighting (fast, baseline)
    - 'grid_search': Exhaustive search over weight combinations (recommended)
    - 'stacking': Meta-learner (Ridge regression) to learn optimal combination
```

```
      - 'cv_grid_search': Grid search with time series cross-validation (most
robust)
      """

      def __init__(
          self,
          name: str = "Ensemble",
          weights: Optional[Dict[str, float]] = None,
          optimize_weights: bool = True,
          optimization_strategy: OptimizationStrategy = 'grid_search',
          validation_ratio: float = 0.20,
          cv_splits: int = 3,
          grid_step: float = 0.05,
          sarima_kwargs: Optional[Dict] = None,
          prophet_kwargs: Optional[Dict] = None,
          baseline_kwargs: Optional[Dict] = None
      ):
          super().__init__(name)
          self.weights = weights or {'sarima': 0.05, 'prophet': 0.05,
'baseline': 0.90}
          self.optimize_weights = optimize_weights
          self.optimization_strategy = optimization_strategy
          self.validation_ratio = validation_ratio
          self.cv_splits = cv_splits
          self.grid_step = grid_step

          self.sarima = SARIMAModel(**(sarima_kwargs or {}))
          self.prophet = ProphetModel(**(prophet_kwargs or {}))
          self.baseline = SMADOWModel(**(baseline_kwargs or {}))

          self.models: Dict[str, BaseTimeSeriesModel] = {
              'sarima': self.sarima, 'prophet': self.prophet, 'baseline':
self.baseline
          }
          self.individual_predictions: Dict[str, pd.Series] = {}
          self.individual_metrics: Dict[str, Dict[str, float]] = {}
          self.optimization_results: Dict = {}
```

```
    def fit(self, train_data: pd.Series) -> 'EnsembleModel':
        """Fit all ensemble components."""
        self.train_data = train_data.copy()

        if self.optimize_weights and len(train_data) > 30:
            val_size = max(10, int(len(train_data) * self.validation_ratio))
            fit_data = train_data.iloc[:-val_size]
            val_data = train_data.iloc[-val_size:]
        else:
            fit_data = train_data
            val_data = None

        for model_name, model in self.models.items():
            model.fit(fit_data)

        if self.optimize_weights and val_data is not None:
            self._optimize_grid_search(fit_data, val_data)
            for model_name, model in self.models.items():
                model.fit(train_data)

        self.is_fitted = True
        return self

    def _optimize_grid_search(self, train_data: pd.Series, val_data:
pd.Series) -> None:
        """Exhaustive grid search for optimal weights."""
        predictions = {}
        val_periods = len(val_data)

        for model_name, model in self.models.items():
            if model.is_fitted:
                pred = model.predict(val_periods)
                pred.index = val_data.index
                predictions[model_name] = pred.values
```

```
            model_names = list(predictions.keys())
            best_score = float('-inf')
            best_weights = self.weights.copy()
            min_weight = 0.05

            weight_range = np.arange(min_weight, 1 - 2*min_weight +
self.grid_step, self.grid_step)

            for w1 in weight_range:
                for w2 in weight_range:
                    w3 = 1 - w1 - w2
                    if w3 < min_weight - 0.001 or w3 > 1 - 2*min_weight + 0.001:
                        continue
                    w3 = max(min_weight, min(1 - 2*min_weight, w3))

                    weights = {model_names[0]: w1, model_names[1]: w2,
model_names[2]: w3}

                    ensemble_pred = (
                        w1 * predictions[model_names[0]] +
                        w2 * predictions[model_names[1]] +
                        w3 * predictions[model_names[2]]
                    )

                    mae = mean_absolute_error(val_data.values, ensemble_pred)
                    r2 = r2_score(val_data.values, ensemble_pred)

                    if r2 >= 0:
                        score = r2 - mae / val_data.mean()
                    else:
                        score = r2 * 2 - mae / val_data.mean()

                    if score > best_score:
                        best_score = score
                        best_weights = weights.copy()

            self.weights = best_weights
```

```
    def predict(self, periods: int) -> pd.Series:
        """Generate ensemble forecasts combining individual model
predictions."""
        if not self.is_fitted:
            raise ValueError("Model must be fitted before predicting")

        self.individual_predictions = {}
        for model_name, model in self.models.items():
            if model.is_fitted:
                self.individual_predictions[model_name] =
model.predict(periods)

        first_pred = next(iter(self.individual_predictions.values()))
        future_dates = first_pred.index

        ensemble_pred = np.zeros(periods)
        total_weight = 0
        for model_name, pred in self.individual_predictions.items():
            weight = self.weights.get(model_name, 0)
            if weight > 0:
                ensemble_pred += weight * pred.values
                total_weight += weight

        if total_weight > 0:
            ensemble_pred /= total_weight

        ensemble_pred = np.clip(ensemble_pred, 0, None)
        return pd.Series(ensemble_pred, index=future_dates,
name=f'{self.name}_forecast')

    def get_confidence_intervals(self, periods: int) -> pd.DataFrame:
        """Estimate confidence intervals using model disagreement."""
        _ = self.predict(periods)

        preds_array = np.array([p.values for p in
self.individual_predictions.values()])
        pred_mean = preds_array.mean(axis=0)
```

```
pred_std = preds_array.std(axis=0)
hist_std = self.train_data.std()
combined_std = np.sqrt(pred_std**2 + (0.5 * hist_std)**2)

first_pred = next(iter(self.individual_predictions.values()))
return pd.DataFrame({
    'lower': np.clip(pred_mean - 1.96 * combined_std, 0, None),
    'upper': pred_mean + 1.96 * combined_std
}, index=first_pred.index)
```

Лістинг В.2 – Виявлення аномалій

```
"""
  Anomaly Detection and Peak Analysis for ITSM time series
  """

  from typing import Dict, List, Optional
  from dataclasses import dataclass
  import pandas as pd
  import numpy as np
  from sklearn.ensemble import IsolationForest


  @dataclass
  class Anomaly:
      """Container for detected anomaly."""
      date: pd.Timestamp
      value: float
      anomaly_type: str  # 'spike', 'dip', 'unusual_pattern'
      severity: str  # 'low', 'medium', 'high', 'critical'
      z_score: float
      description: str


  class AnomalyDetector:
      """
      Multi-method anomaly detection combining:
      - Statistical methods (Z-score, IQR)
      - Machine Learning (Isolation Forest)
      """

      def __init__(
          self,
          z_threshold: float = 2.5,
          iqr_multiplier: float = 1.5,
```

```
            isolation_contamination: float = 0.1
        ):
            self.z_threshold = z_threshold
            self.iqr_multiplier = iqr_multiplier
            self.isolation_contamination = isolation_contamination
            self.train_data: Optional[pd.Series] = None
            self.anomalies: List[Anomaly] = []
            self.statistics: Dict[str, float] = {}

        def fit(self, data: pd.Series) -> 'AnomalyDetector':
            """Fit anomaly detector and identify historical anomalies."""
            self.train_data = data.copy()
            self._calculate_statistics()
            self._detect_zscore_anomalies()
            self._detect_iqr_anomalies()
            self._detect_isolation_forest_anomalies()
            self._consolidate_anomalies()
            return self

        def _calculate_statistics(self) -> None:
            """Calculate baseline statistics."""
            mean_val = float(self.train_data.mean())
            std_val = float(self.train_data.std())
            if std_val < 1e-10:
                std_val = max(1.0, mean_val * 0.1)

            self.statistics = {
                'mean': mean_val,
                'std': std_val,
                'q1': float(self.train_data.quantile(0.25)),
                'q3': float(self.train_data.quantile(0.75)),
                'iqr': float(self.train_data.quantile(0.75) -
self.train_data.quantile(0.25)),
            }
            self.statistics['high_threshold'] = self.statistics['mean'] + 2 *
self.statistics['std']
```

```
          self.statistics['critical_threshold'] = self.statistics['mean'] + 3 *
self.statistics['std']


      def _detect_zscore_anomalies(self) -> None:
          """Detect anomalies using Z-score method."""
          for date, value in self.train_data.items():
              z = float((value - self.statistics['mean']) /
self.statistics['std'])
              if abs(z) > self.z_threshold:
                  anomaly_type = 'spike' if z > 0 else 'dip'
                  severity = self._get_severity(abs(z))
                  self.anomalies.append(Anomaly(
                      date=date, value=float(value), anomaly_type=anomaly_type,
                      severity=severity, z_score=z,
                      description=f"Z-score {anomaly_type}: {z:.2f} standard
deviations"
                  ))


      def _detect_iqr_anomalies(self) -> None:
          """Detect anomalies using IQR method."""
          lower_bound = self.statistics['q1'] - self.iqr_multiplier *
self.statistics['iqr']
          upper_bound = self.statistics['q3'] + self.iqr_multiplier *
self.statistics['iqr']


          for date, value in self.train_data.items():
              if value > upper_bound or value < lower_bound:
                  if not any(a.date == date for a in self.anomalies):
                      z = (value - self.statistics['mean']) /
self.statistics['std']
                      anomaly_type = 'spike' if value > upper_bound else 'dip'
                      self.anomalies.append(Anomaly(
                          date=date, value=float(value),
anomaly_type=anomaly_type,
                          severity='medium', z_score=z,
                          description=f"IQR {anomaly_type}: outside
[{lower_bound:.1f}, {upper_bound:.1f}]"
                      ))
```

```
    def _detect_isolation_forest_anomalies(self) -> None:
        """Detect anomalies using Isolation Forest."""
        if len(self.train_data) < 10:
            return

        df = pd.DataFrame({'value': self.train_data})
        df['day_of_week'] = df.index.dayofweek
        df['rolling_mean_7d'] = df['value'].rolling(7, min_periods=1).mean()
        df['rolling_std_7d'] = df['value'].rolling(7,
min_periods=1).std().fillna(0)
        df['diff'] = df['value'].diff().fillna(0)

        features = df[['value', 'day_of_week', 'rolling_mean_7d',
'rolling_std_7d', 'diff']].values

        iso_forest = IsolationForest(
            contamination=self.isolation_contamination,
            random_state=42, n_estimators=100
        )
        predictions = iso_forest.fit_predict(features)
        scores = iso_forest.decision_function(features)

        for i, (date, pred, score) in enumerate(zip(self.train_data.index,
predictions, scores)):
            if pred == -1:
                if not any(a.date == date for a in self.anomalies):
                    value = self.train_data.iloc[i]
                    z = (value - self.statistics['mean']) /
self.statistics['std']
                    self.anomalies.append(Anomaly(
                        date=date, value=value,
anomaly_type='unusual_pattern',
                        severity='medium', z_score=z,
                        description=f"Isolation Forest anomaly (score:
{score:.3f})"
                    ))

    def _get_severity(self, abs_z_score: float) -> str:
```

```
        if abs_z_score >= 4: return 'critical'
        elif abs_z_score >= 3: return 'high'
        elif abs_z_score >= 2.5: return 'medium'
        else: return 'low'

    def _consolidate_anomalies(self) -> None:
        """Remove duplicates and sort anomalies by date."""
        seen_dates = set()
        unique = []
        for a in sorted(self.anomalies, key=lambda x: (x.date,
{'critical':0,'high':1,'medium':2,'low':3}.get(x.severity,4))):
            if a.date not in seen_dates:
                unique.append(a)
                seen_dates.add(a.date)
        self.anomalies = unique
```

Лістинг В.3 – LIME-інтерпретація прогнозів

```
"""
  LIME for time series forecasting explanations
  """

  from typing import Dict, List, Optional, Callable, Tuple
  import pandas as pd
  import numpy as np
  from dataclasses import dataclass
  import warnings
  from lime import lime_tabular


  @dataclass
  class PredictionExplanation:
      """Container for a single prediction explanation."""
      date: pd.Timestamp
      predicted_value: float
      feature_contributions: Dict[str, float]
      top_features: List[Tuple[str, float]]
      explanation_text: str


  class TimeSeriesLIME:
      """LIME explainer adapted for time series ensemble predictions."""

      def __init__(
          self,
          n_lags: int = 7,
          include_rolling_stats: bool = True,
          rolling_windows: List[int] = None,
          include_calendar: bool = True,
          num_samples: int = 500,
          random_state: int = 42
```

```
        ):
            self.n_lags = n_lags
            self.include_rolling_stats = include_rolling_stats
            self.rolling_windows = rolling_windows or [7, 14]
            self.include_calendar = include_calendar
            self.num_samples = num_samples
            self.random_state = random_state
            self.feature_names: List[str] = []
            self.explainer: Optional[lime_tabular.LimeTabularExplainer] = None
            self.train_data: Optional[pd.Series] = None

        def _create_features(self, data: pd.Series) -> pd.DataFrame:
            """Convert time series to feature matrix for LIME."""
            df = pd.DataFrame(index=data.index)

            for i in range(1, self.n_lags + 1):
                df[f'lag_{i}'] = data.shift(i)

            if self.include_rolling_stats:
                for window in self.rolling_windows:
                    df[f'rolling_mean_{window}d'] =
data.rolling(window=window).mean()
                    df[f'rolling_std_{window}d'] =
data.rolling(window=window).std()

            if self.include_calendar:
                df['day_of_week'] = data.index.dayofweek
                df['is_weekend'] = (data.index.dayofweek >= 5).astype(int)
                df['is_month_start'] = data.index.is_month_start.astype(int)
                df['is_month_end'] = data.index.is_month_end.astype(int)

            return df.dropna()

        def fit(self, train_data: pd.Series) -> 'TimeSeriesLIME':
            """Fit the LIME explainer on training data."""
            self.train_data = train_data.copy()
            features_df = self._create_features(train_data)
```

```
            self.feature_names = list(features_df.columns)
            self.feature_matrix = features_df.values

            self.explainer = lime_tabular.LimeTabularExplainer(
                training_data=self.feature_matrix,
                feature_names=self.feature_names,
                mode='regression',
                random_state=self.random_state
            )
            return self

        def explain_prediction(
            self,
            prediction_date: pd.Timestamp,
            predicted_value: float,
            predict_fn: Callable[[np.ndarray], np.ndarray],
            num_features: int = 10
        ) -> PredictionExplanation:
            """Explain a single prediction."""
            extended_data = self.train_data.copy()
            extended_data[prediction_date] = predicted_value
            features_df = self._create_features(extended_data)

            instance = features_df.loc[prediction_date].values if prediction_date
in features_df.index else
  self.feature_matrix[-1]

            with warnings.catch_warnings():
                warnings.filterwarnings('ignore')
                explanation = self.explainer.explain_instance(
                    instance, predict_fn, num_features=num_features,
num_samples=self.num_samples
                )

            feature_weights = dict(explanation.as_list())
            contributions = {}
            for feature_expr, weight in feature_weights.items():
```

```
                for fname in self.feature_names:
                    if fname in feature_expr:
                        contributions[fname] = weight
                        break

            top_features = sorted(contributions.items(), key=lambda x: abs(x[1]),
reverse=True)[:num_features]

            lines = [f"Prediction for {prediction_date.strftime('%Y-%m-%d')}:
{predicted_value:.1f}", "Key factors:"]
            for i, (feature, contribution) in enumerate(top_features[:5], 1):
                direction = "increases" if contribution > 0 else "decreases"
                lines.append(f"  {i}. {feature} {direction} by
{abs(contribution):.2f}")

            return PredictionExplanation(
                date=prediction_date,
                predicted_value=predicted_value,
                feature_contributions=contributions,
                top_features=top_features,
                explanation_text="\n".join(lines)
            )
```

Лістинг В.4 – Генерація проактивних сервісних запитів

```
"""
  Proactive Service Request Generator
  Automatically creates preventive SRs based on forecasts and patterns
  """

  from typing import Dict, List, Optional
  from dataclasses import dataclass, field
  from datetime import datetime
  import pandas as pd
  import numpy as np


  @dataclass
  class ProactiveSR:
      """Container for a proactive service request."""
      sr_id: str
      target_date: str
      severity: str  # 'low', 'medium', 'high', 'critical'
      title: str
      description: str
      predicted_volume: Optional[float]
      confidence_lower: Optional[float]
      confidence_upper: Optional[float]
      contributing_factors: List[str]
      suggested_actions: List[str]


  class ProactiveSRGenerator:
      """
      Generate proactive service requests based on:
      - Volume forecasts exceeding thresholds
      - Detected anomalies and patterns
      - Historical peak patterns
```

```
    """

    def __init__(
        self,
        high_volume_percentile: float = 85,
        critical_volume_percentile: float = 95,
        include_weekends: bool = False
    ):
        self.high_volume_percentile = high_volume_percentile
        self.critical_volume_percentile = critical_volume_percentile
        self.include_weekends = include_weekends
        self.historical_data: Optional[pd.Series] = None
        self.thresholds: Dict[str, float] = {}
        self.generated_srs: List[ProactiveSR] = []

    def fit(self, historical_data: pd.Series) -> 'ProactiveSRGenerator':
        """Fit generator with historical data to establish thresholds."""
        self.historical_data = historical_data.copy()
        self.thresholds = {
            'mean': historical_data.mean(),
            'low': np.percentile(historical_data, 50),
            'high': np.percentile(historical_data,
self.high_volume_percentile),
            'very_high': np.percentile(historical_data, 90),
            'critical': np.percentile(historical_data,
self.critical_volume_percentile),
            'extreme': np.percentile(historical_data, 95)
        }
        return self

    def generate(
        self,
        predictions: pd.Series,
        confidence_intervals: Optional[pd.DataFrame] = None,
        lime_explanations: Optional[List] = None
    ) -> List[ProactiveSR]:
        """Generate proactive SRs based on predictions."""
```

```
        self.generated_srs = []
        sr_counter = 1

        for i, (date, value) in enumerate(predictions.items()):
            if not self.include_weekends and hasattr(date, 'dayofweek') and
date.dayofweek >= 5:
                continue

            ci_lower = confidence_intervals.loc[date, 'lower'] if
confidence_intervals is not None else None
            ci_upper = confidence_intervals.loc[date, 'upper'] if
confidence_intervals is not None else None

            if lime_explanations and i < len(lime_explanations):
                exp = lime_explanations[i]
                contributing_factors = [f[0] for f in exp.top_features[:3]] if
hasattr(exp, 'top_features') else
  []
            else:
                contributing_factors = self._get_contributing_factors(date,
value)

            sr = self._create_volume_alert(date, value, ci_lower, ci_upper,
contributing_factors, sr_counter)
            if sr:
                self.generated_srs.append(sr)
                sr_counter += 1

        return self.generated_srs

    def _create_volume_alert(
        self, date, value, ci_lower, ci_upper, contributing_factors, counter
    ) -> Optional[ProactiveSR]:
        """Create volume-based alert if thresholds exceeded."""
        if value >= self.thresholds['extreme']:
            severity = 'critical'
        elif value >= self.thresholds['very_high']:
            severity = 'high'
```

```
            elif value >= self.thresholds['high']:
                severity = 'medium'
            elif value >= self.thresholds['low']:
                severity = 'low'
            else:
                return None

            day_name = date.strftime('%A')
            date_str = date.strftime('%Y-%m-%d')
            ci_text = f" (95% CI: {ci_lower:.0f}-{ci_upper:.0f})" if ci_lower and
ci_upper else ""

            description = f"Ensemble model predicts {value:.0f}
incidents{ci_text}. " \
                          f"Exceeds {severity} threshold of
{self.thresholds.get(severity,
  self.thresholds['high']):.0f}."

            actions = self._get_suggested_actions(severity, value, day_name)

            return ProactiveSR(
                sr_id=f"PRO-{datetime.now().strftime('%Y%m%d')}-{counter:04d}",
                target_date=date_str,
                severity=severity,
                title=f"Expected {severity.title()} Volume on {day_name}
({date.strftime('%m/%d')})",
                description=description,
                predicted_volume=value,
                confidence_lower=ci_lower,
                confidence_upper=ci_upper,
                contributing_factors=contributing_factors,
                suggested_actions=actions
            )

        def _get_contributing_factors(self, date, value) -> List[str]:
            factors = []
            if hasattr(date, 'dayofweek'):
```

```
            dow_means =
self.historical_data.groupby(self.historical_data.index.dayofweek).mean()
            if date.dayofweek == dow_means.idxmax():
                factors.append(f"Peak day ({date.strftime('%A')})")
        if value > self.thresholds['very_high']:
            factors.append("Historical high pattern")
        return factors if factors else ["General forecast trend"]


    def _get_suggested_actions(self, severity: str, value: float, day_name:
str) -> List[str]:
        if severity == 'critical':
            return [
                "Activate all available support staff",
                "Consider overtime or on-call resources",
                "Pre-prepare common resolution scripts",
                f"Expected ~{value:.0f} incidents on {day_name}"
            ]
        elif severity == 'high':
            return [
                "Schedule additional support staff",
                "Review escalation procedures",
                f"Plan for ~{value:.0f} incidents on {day_name}"
            ]
        return [
            "Monitor volume closely",
            f"Expect ~{value:.0f} incidents on {day_name}"
        ]
```

Додаток Г

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ПРОГНОЗУВАННЯ ТА МІНІМІЗАЦІЇ ІНЦИДЕНТІВ ПІД ЧАС УПРАВЛІННЯ ІТ-ПОСЛУГАМИ

Нормоконтроль: к.т.н., доцент

_____ Сергій ЖУКОВ

«___» _____ 2025 р.

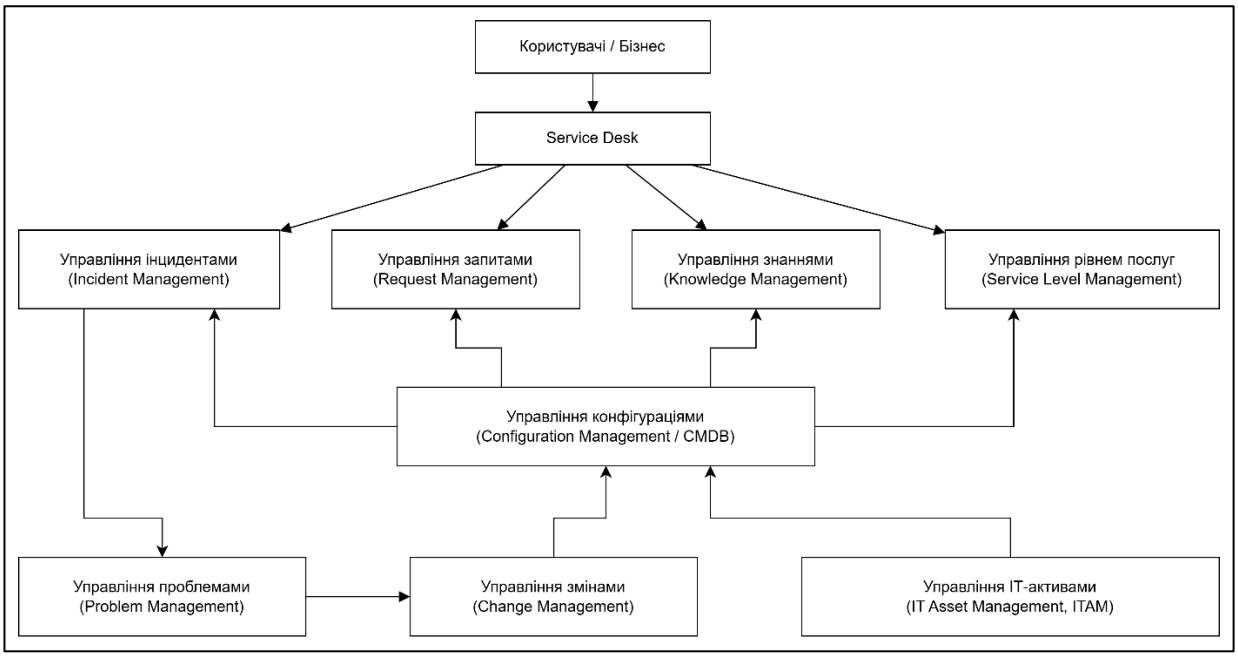

Рисунок Г.1 – Узагальнена схема взаємодії основних процесів ITSM

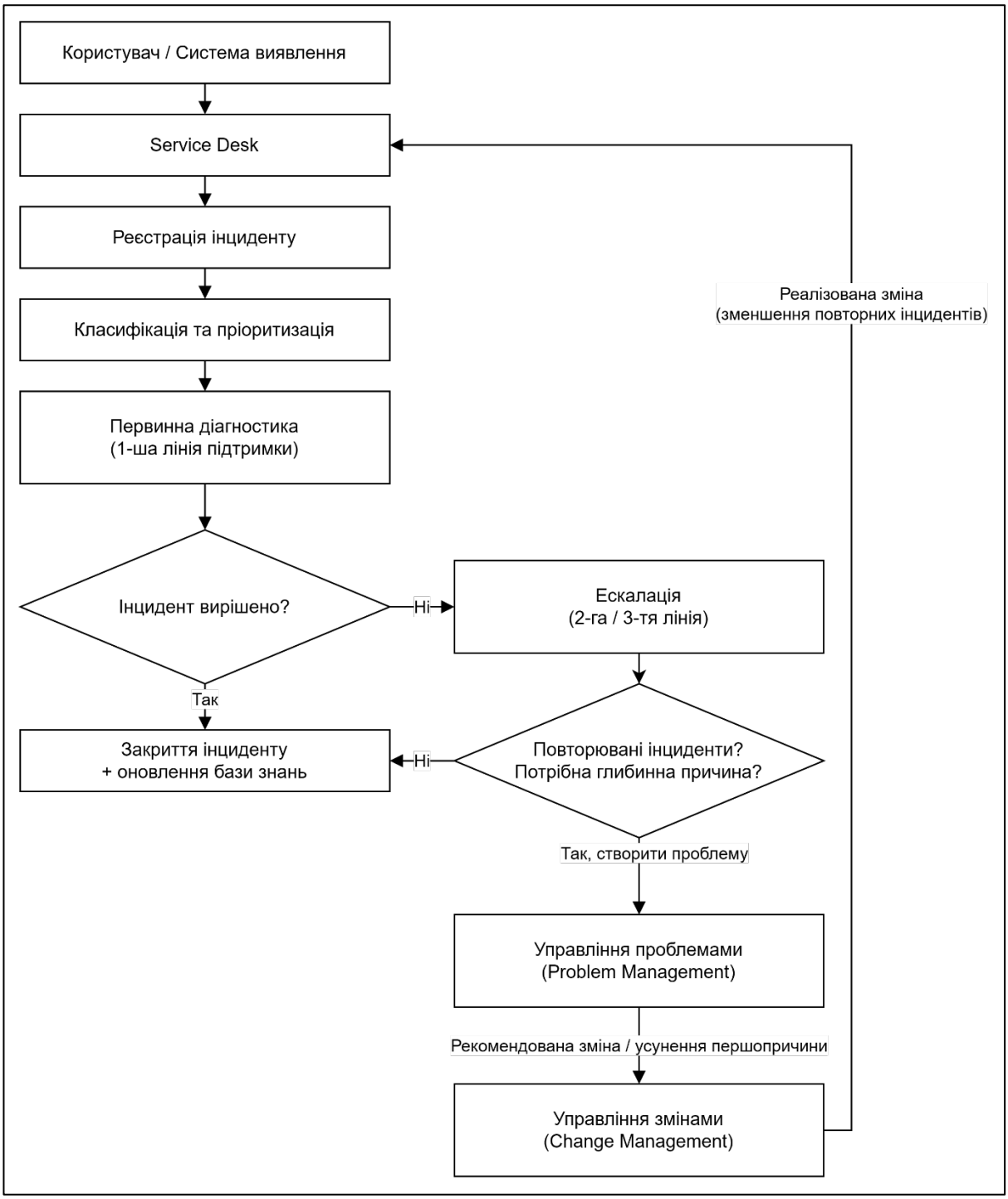

Рисунок Г.2 – Спрощений життєвий цикл інциденту та його взаємодія з управлінням проблемами й змінами

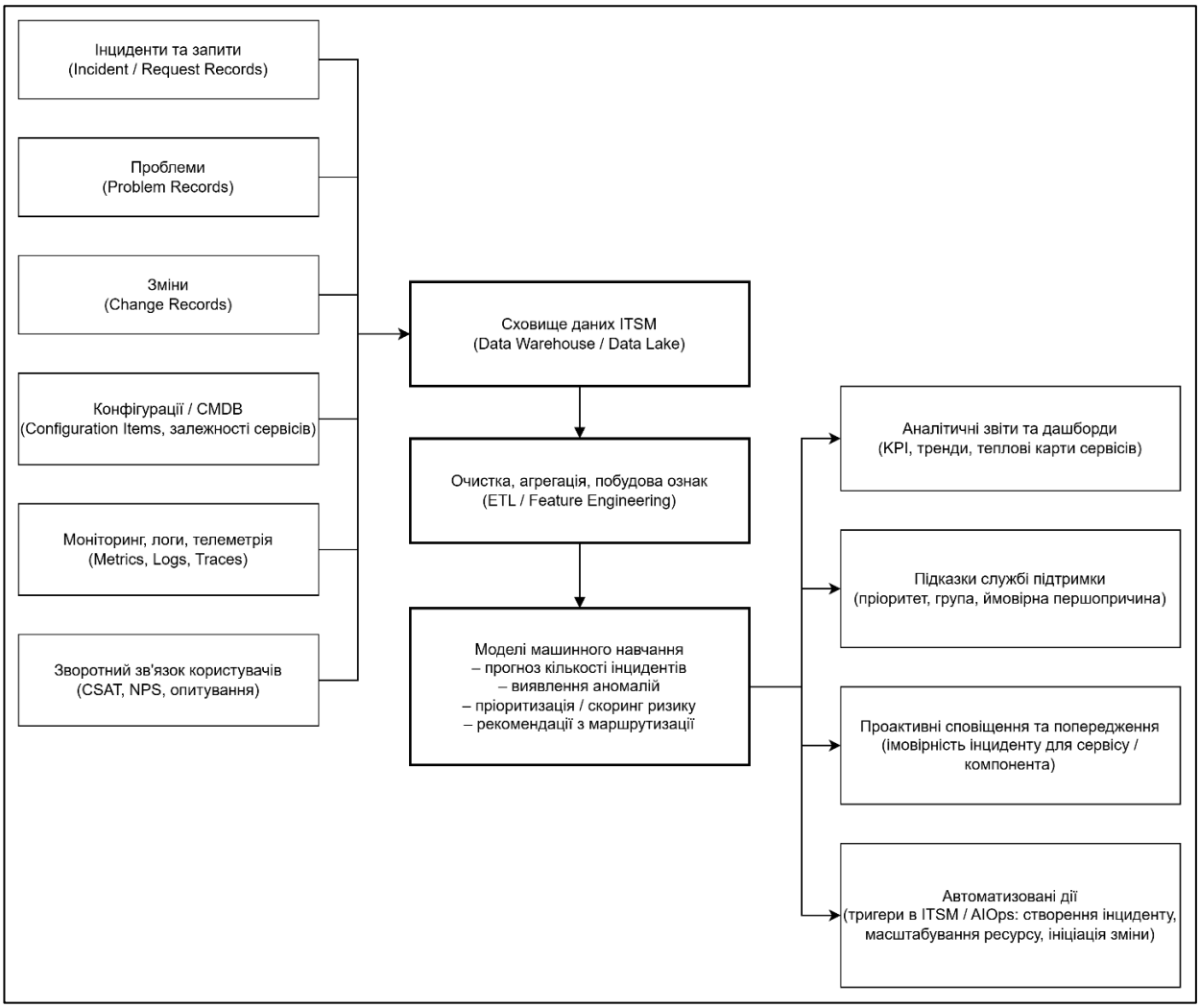

Рисунок Г.3 – Узагальнена концептуальна схема використання даних ITSM для побудови моделей прогнозування інцидентів.

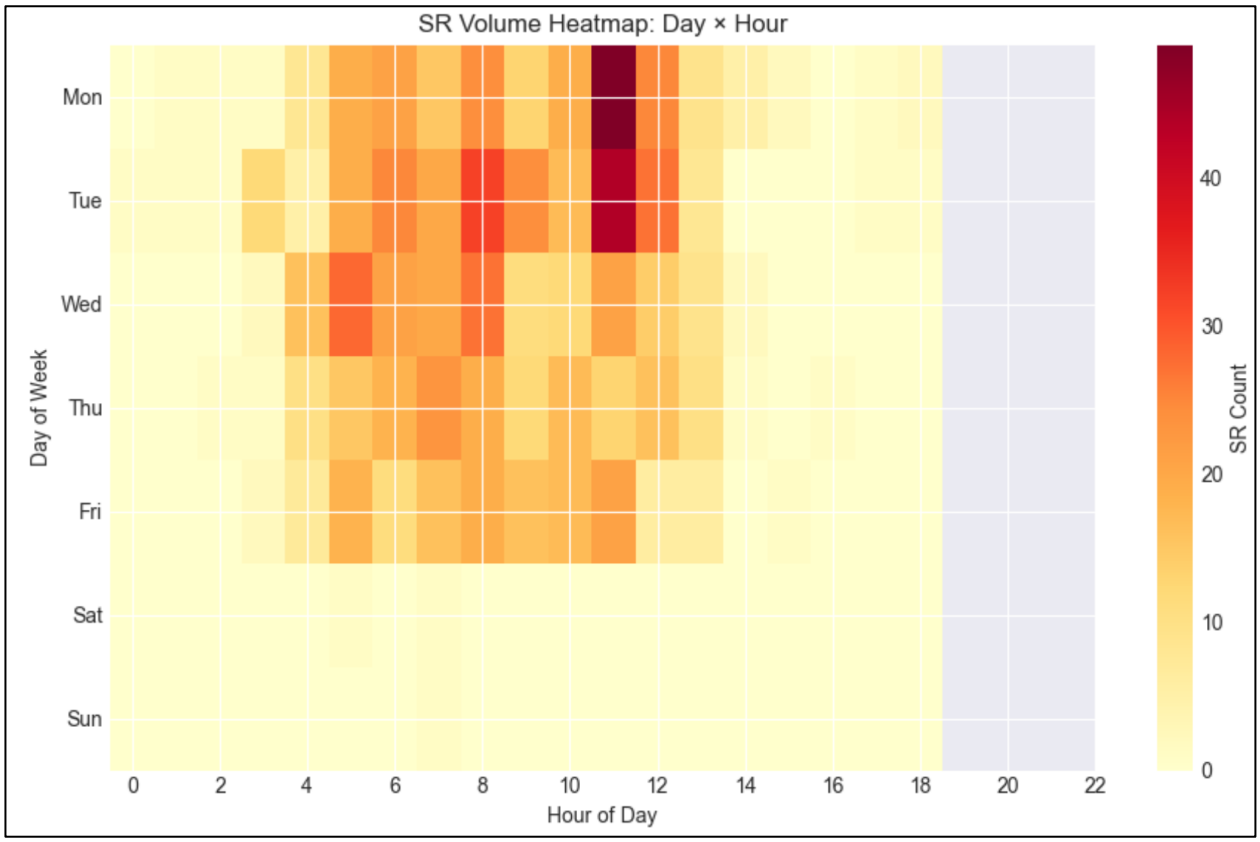

Рисунок Г.4 – Теплова карта темпоральних патернів

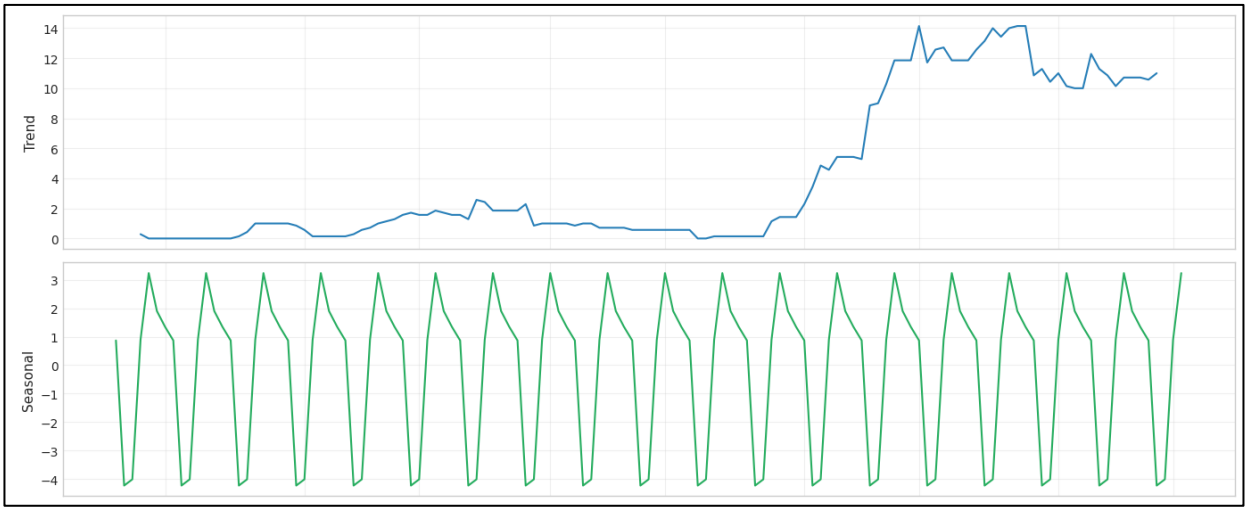

Рисунок Г.5 – Декомпозиція часового ряду

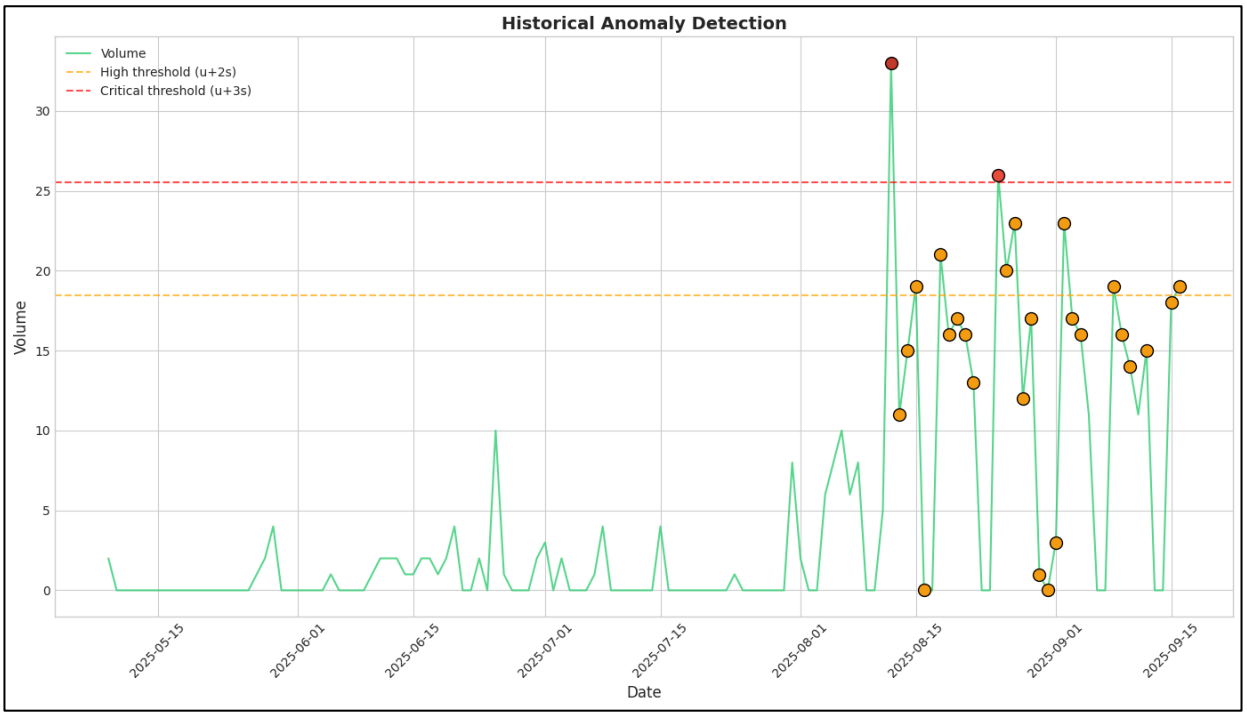

Рисунок Г.6 – Часовий ряд щоденної кількості сервісних записів з накладеними маркерами аномальних днів

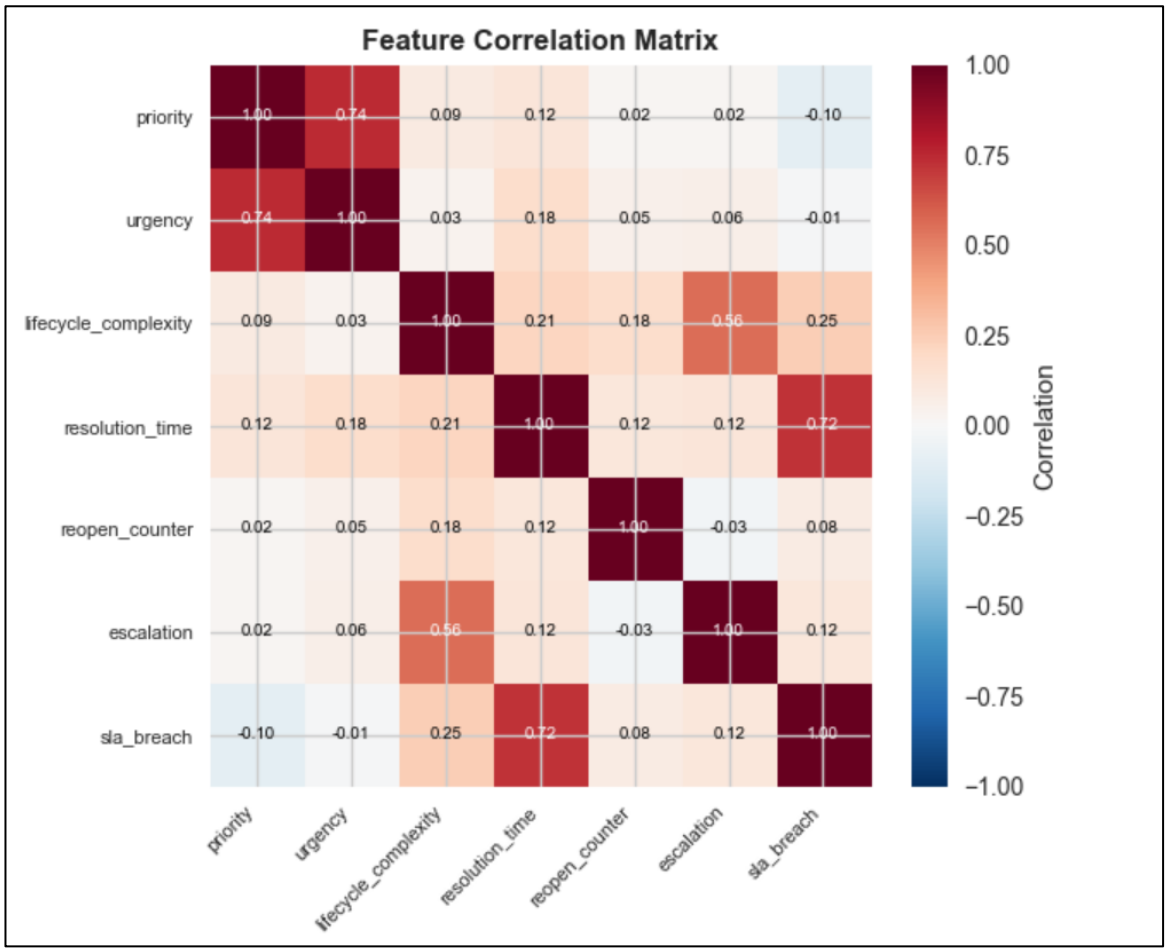

Рисунок Г.7 – Кореляційна матриця ключових ознак

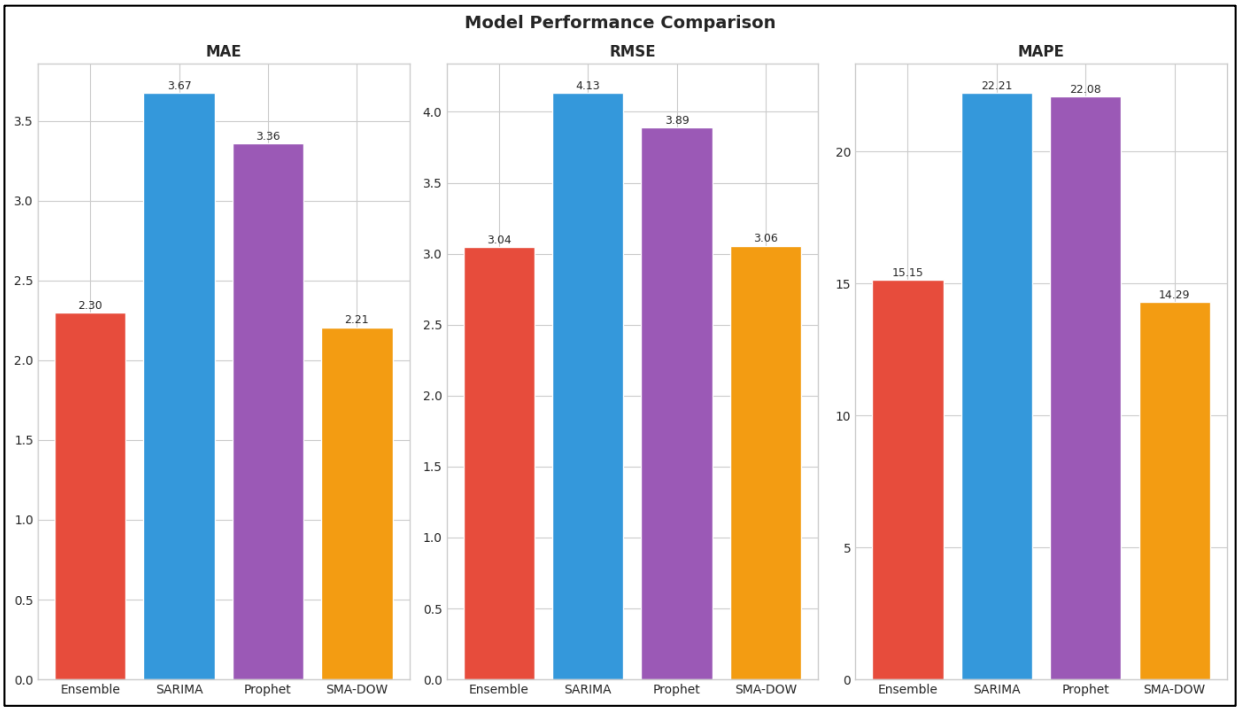

Рисунок Г.8 – Результати валідації точності моделей на тестовій вибірці

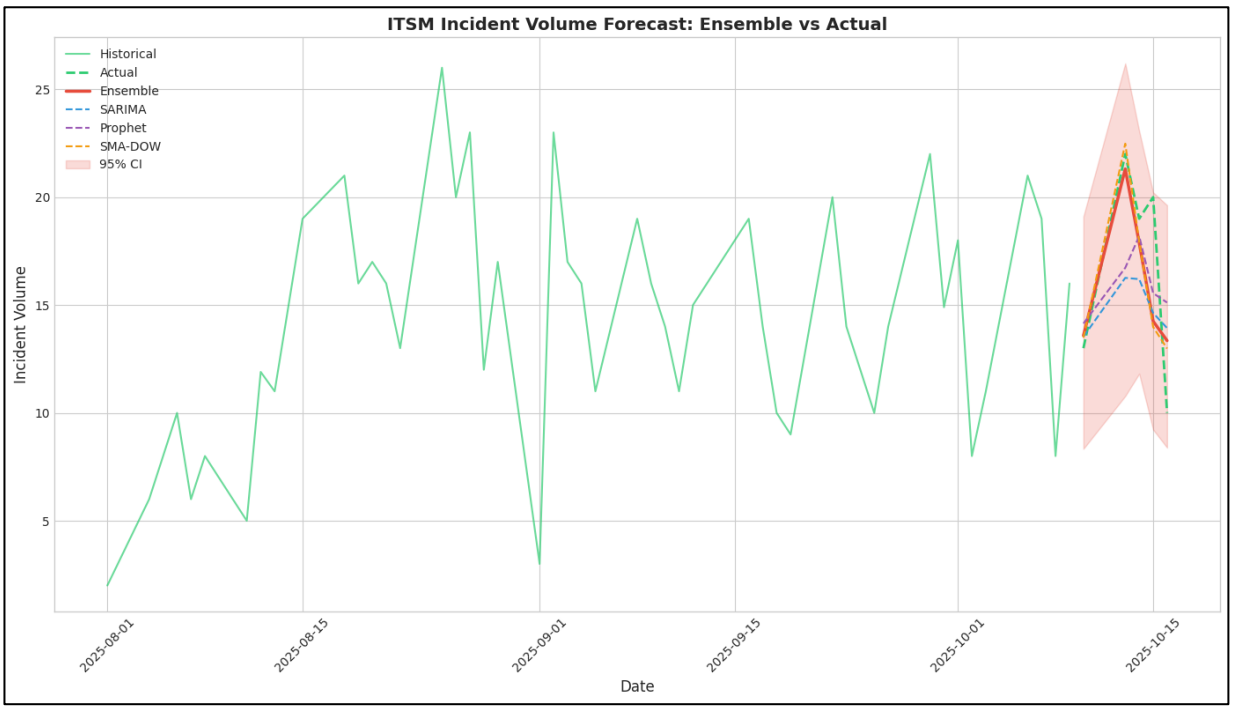

Рисунок Г.9 – Візуалізація результатів прогнозування

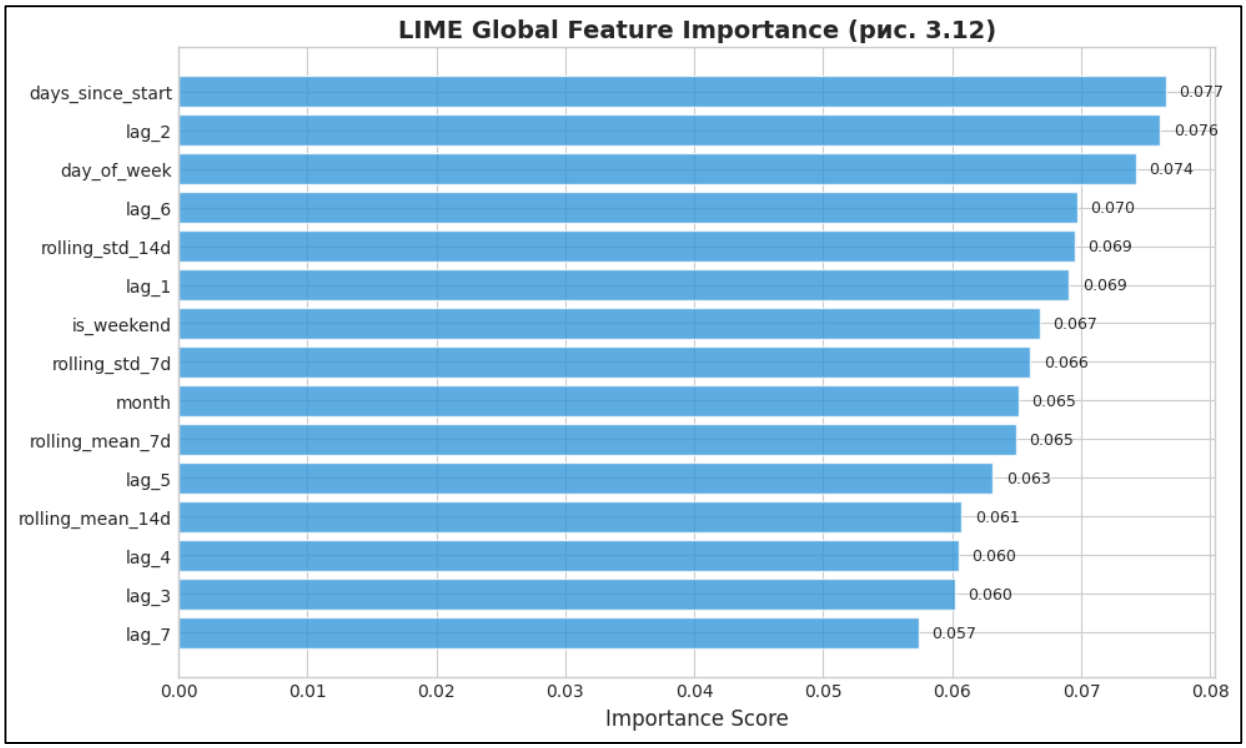

Рисунок Г.10 – Гістограма глобальної важливості ознак

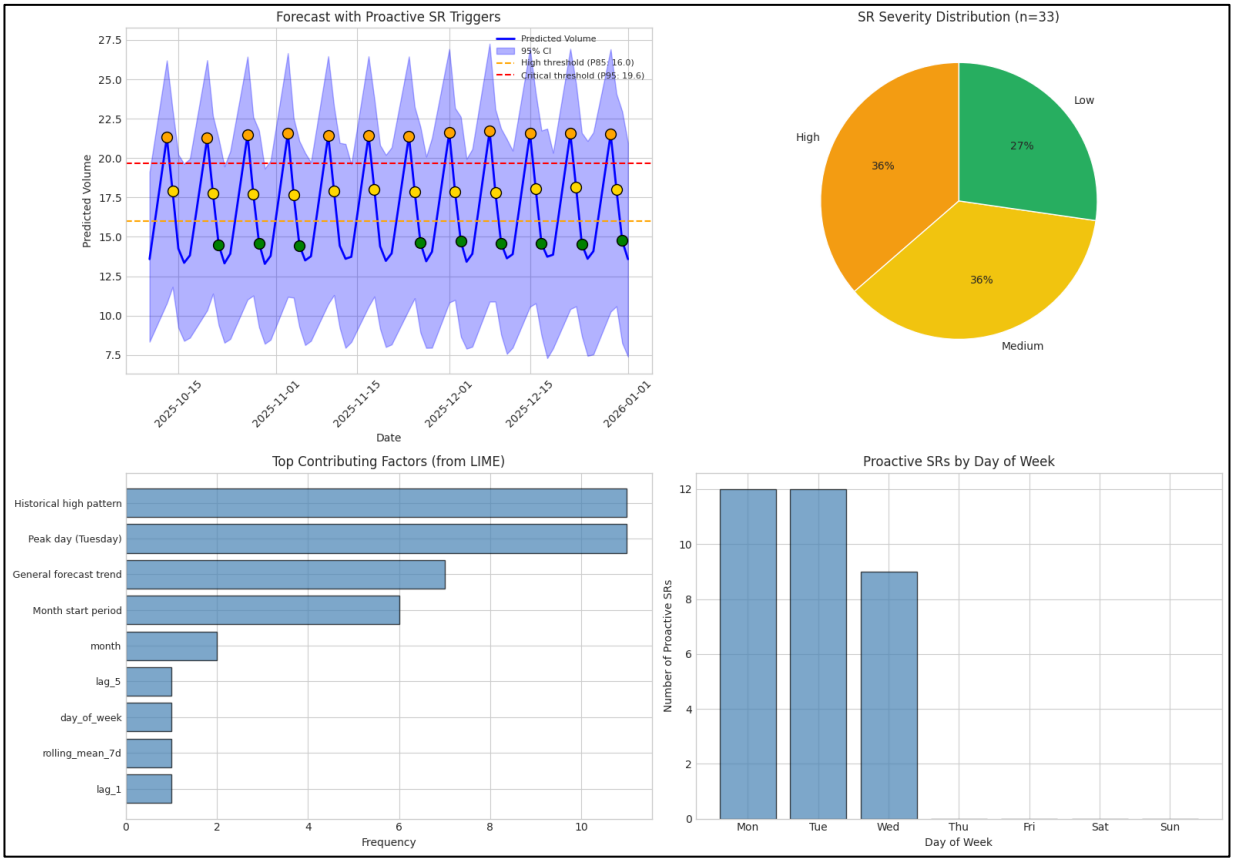

Рисунок Г.11 – Комплексну візуалізація результатів генерації проактивних сервісних запитів