МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Тернопільський національний технічний університет
імені Івана Пулюя

Кафедра програмної інженерії


# Методичні вказівки до практичних робіт

з дисципліни

## «Розподілені та паралельні системи обчислень»

CEH ID: **6843**

для здобувачів третього (освітньо-наукового) рівня вищої освіти ОНП
«Інженерія програмного забезпечення»

Тернопіль — 2025

УДК 004.4(075.8)
ББК 32.973.202я73
Б83

**Укладач:**
В.М. Бревус, PhD

**Рецензент:**
В.В. Яцишин, к.т.н., доцент
завідувач кафедри систем штучного інтелекту та аналізу даних ТНТУ ім. І. Пулюя

Розглянуто й затверджено на засіданні кафедри програмної інженерії.
Протокол № 16 від 29.05.2025.

Розглянуто й затверджено на засіданні методичної комісії факультету комп'ютерно-інформаційних систем та програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя.
Протокол № 5 від 05.06.2025.

Б83 Методичні вказівки до практичних робіт з дисципліни «Розподілені та паралельні системи обчислень» для здобувачів третього (освітньо-наукового) рівня вищої освіти ОНП «Інженерія програмного забезпечення» / Укладач: Бревус В.М. – Тернопіль: ТНТУ імені Івана Пулюя, 2025 – 22 с.

**Відповідальний за випуск:** М.Р. Петрик, докт. фіз.-мат. наук, професор

# АНОТАЦІЯ

У збірнику практичних робіт представлено комплексний наскрізний проєкт «Розробка та розгортання системи прогнозування попиту в електронній комерції з використанням PySpark». Проєкт присвячений вирішенню реальних бізнес-завдань з оптимізації ланцюгів постачання та управління запасами.

Практичні роботи охоплюють весь цикл розробки системи машинного навчання в розподіленому середовищі: від обробки великих даних за допомогою Apache Spark до побудови, оцінки та інтерпретації регресійних моделей. Особливу увагу приділено інженерії ознак, використанню бібліотеки Spark MLlib та валідації моделей на реальних даних.

Збірник практичних робіт призначений для здобувачів третього (освітньо-наукового) рівня вищої освіти, які навчаються за освітньо-науковою програмою «Інженерія програмного забезпечення». Він також буде корисним для аспірантів, науковців та інженерів, що спеціалізуються в галузі аналізу великих даних та машинного навчання.

**Ключові слова:** розподілені системи, Apache Spark, PySpark, машинне навчання, прогнозування попиту, великі дані, інженерія ознак.

# ABSTRACT

The collection of practical works presents a comprehensive end-to-end project “Development and Deployment of a Demand Forecasting System in E-commerce using PySpark”. The project is dedicated to solving real-world business problems of supply chain optimization and inventory management.

The practical works cover the complete machine learning system development cycle in a distributed environment: from big data processing with Apache Spark to building, evaluating, and interpreting regression models. Special attention is given to feature engineering, the use of the Spark MLlib library, and model validation on real-world data.

The collection of practical works is intended for students of the third (educational and scientific) level of higher education enrolled in the “Software Engineering” educational and scientific program. It will also be useful for postgraduate students, researchers, and engineers specializing in big data analytics and machine learning.

**Keywords:** distributed systems, Apache Spark, PySpark, machine learning, demand forecasting, big data, feature engineering.

# Зміст

# Вступ

Деталі щодо оформлення програмного коду та змісту звіту практичних робіт узгоджуються з викладачем у системі електронного навчання відповідного курсу. Приклад оформлення звіту з титульним аркушем, структурою та рекомендаціями наведено в додатку А.

Усі практичні роботи повинні бути виконані з дотриманням принципів академічної доброчесності, відповідно до Положення про академічну доброчесність учасників освітнього процесу Тернопільського національного технічного університету імені Івана Пулюя.

## Наскрізний проєкт

**«Розробка та розгортання системи прогнозування попиту в електронній комерції з використанням PySpark»**

Протягом серії практичних робіт здобувачі розроблятимуть комплексну систему для прогнозування попиту на товари в онлайн-ритейлі. Проєкт спрямований на вирішення реальних бізнес-завдань, пов'язаних з оптимізацією ланцюгів постачання, управлінням запасами та плануванням маркетингових активностей.

### Бізнес-обґрунтування

Електронна комерція стикається з постійною невизначеністю, що ускладнює планування операцій. Неточні прогнози призводять до дефіциту товарів (втрачена вигода) або надлишкових запасів (збільшені витрати на зберігання). Точне прогнозування попиту дозволяє компаніям забезпечити наявність потрібних товарів у потрібний час, підвищуючи задоволеність клієнтів та операційну ефективність. Цей проєкт імітує роботу в команді планування продажів та операцій (S&OP) великої e-commerce компанії.

**Ключові переваги підходу:**

- **Оптимізація запасів:** Зниження ризиків дефіциту та надлишків.
- **Підвищення ефективності:** Покращення планування логістики та постачання.
- **Стратегічне планування:** Обґрунтовані рішення щодо промо-акцій та ціноутворення.
- **Масштабованість:** Використання розподілених обчислень для обробки великих обсягів даних.

### Технічний підхід

Проєкт базується на використанні Apache Spark через PySpark для побудови моделі машинного навчання, що прогнозує кількість проданих одиниць товару.

1. **Розподілена обробка даних:** Використання Spark DataFrame API для ефективної маніпуляції великими наборами даних.

2. **Інженерія ознак:** Створення нових прогностичних ознак з наявних даних (наприклад, часових характеристик з дати транзакції).

3. **Машинне навчання в розподіленому середовищі:** Побудова та налаштування регресійних моделей за допомогою бібліотеки Spark MLlib.

4. **Валідація моделі:** Оцінка точності прогнозування за допомогою метрик, таких як Mean Absolute Error (MAE), на тестових даних.

**Структура практичних робіт**

**ПР№1: Основи роботи з PySpark та аналіз даних**

- Налаштування середовища Spark та завантаження датасету «Online Retail».
- Агрегація даних на щоденному рівні для підготовки до моделювання.
- Проведення експлоративного аналізу для виявлення ключових закономірностей у даних.
- Розбиття датасету на тренувальний та тестовий набори за часовою ознакою.

**ПР№2: Інженерія ознак та побудова конвеєра обробки**

- Створення та трансформація ознак (features) з використанням PySpark.
- Індексація категоріальних змінних (наприклад, ‘Country‘, ‘StockCode‘).
- Об’єднання ознак у вектор за допомогою ‘VectorAssembler‘.
- Побудова конвеєра (pipeline) обробки даних для автоматизації процесу.

**ПР№3: Побудова та оцінка моделі машинного навчання**

- Побудова регресійної моделі (наприклад, ‘RandomForestRegressor‘) для прогнозування ‘Quantity‘.
- Навчання моделі на тренувальному наборі даних.
- Виконання прогнозів на тестовому наборі.
- Оцінка якості моделі за допомогою ‘RegressionEvaluator‘ та розрахунок MAE.

**ПР№4: Оптимізація та інтерпретація результатів**

- Аналіз результатів прогнозування для вирішення бізнес-задач.
- Розрахунок прогнозованої кількості товарів на конкретний період (наприклад, тиждень).
- Дослідження методів оптимізації Spark-застосунків (за бажанням).
- Підготовка фінального звіту з отриманими результатами та висновками.

**Очікувані результати**

По завершенні проєкту здобувачі матимуть:

1. **Функціональну модель** для прогнозування попиту.
2. **Глибоке розуміння** принципів розподіленої обробки даних з PySpark.
3. **Практичні навички** побудови конвеєрів машинного навчання в Spark.
4. **Досвід вирішення** реалістичної бізнес-задачі з аналізу даних.
5. **Портфоліо проєкту** для демонстрації своїх навичок.

**Зв'язок з теоретичним курсом**

Практичні роботи тісно інтегровані з лекційним матеріалом:

- **Лекції 2-3:** Вступ до Spark, робота з DataFrame — ПР№1.
- **Лекція 4:** Інженерія ознак та очищення даних — ПР№2.
- **Лекція 5:** Машинне навчання з Spark MLlib — ПР№3.
- **Лекція 6:** Оптимізація Spark-застосунків — ПР№4.

**Інструментарій та ресурси**

**Програмне забезпечення:** Python 3.8+, Apache Spark, Jupyter Notebook.

**Дані:** Датасет 'Online Retail.csv', що містить транзакційні дані міжнародної компанії.

**Обчислювальні ресурси:** Стандартний ноутбук або комп'ютер.

# Практична робота №1

## Основи роботи з PySpark та аналіз даних

### Мета роботи

Ознайомитися з основними принципами роботи в середовищі Apache Spark. Освоїти методи завантаження, попередньої обробки та агрегації великих даних за допомогою PySpark. Підготувати набір даних для подальшого побудови моделі прогнозування.

### Наскрізний проєкт

**«Розробка та розгортання системи прогнозування попиту в електронній комерції з використанням PySpark»**

Протягом серії практичних робіт ми будемо розробляти комплексну систему для прогнозування попиту на товари в онлайн-ритейлі. Проєкт спрямований на вирішення реальних бізнес-завдань, пов'язаних з оптимізацією ланцюгів постачання та управлінням запасами.

### Теоретичні відомості

**Прогнозування попиту в e-commerce** — це процес передбачення майбутніх продажів, що є критично важливим для ефективного управління запасами. Неточність прогнозів може призвести до двох основних проблем:

- **Дефіцит (Stockout):** Втрачені продажі та незадоволені клієнти через відсутність товару.
- **Надлишок (Overstock):** Заморожені кошти та збільшені витрати на зберігання товарів, що не продаються.

**Apache Spark** — це відкритий розподілений обчислювальний фреймворк, що дозволяє ефективно обробляти великі обсяги даних. Ключові компоненти, що використовуються в проєкті:

- **Spark Core:** Надає базову функціональність, включно з концепцією RDD (Resilient Distributed Dataset).
- **Spark SQL:** Дозволяє працювати зі структурованими даними через DataFrame API та SQL-запити.
- **PySpark:** Python API для роботи зі Spark, що поєднує простоту Python з потужністю розподілених обчислень.

У цій роботі ми будемо використовувати дані з файлу `Online Retail.csv`, що містить транзакції міжнародної e-commerce компанії.

**Завдання**

**Завдання 1.1. Налаштування середовища та завантаження даних**

1. Ініціалізувати `SparkSession` — точку входу для роботи з функціональністю Spark.
2. Завантажити датасет `Online Retail.csv` у Spark DataFrame. Вказати, що файл містить заголовок (`header=True`) та автоматично визначити схему даних (`inferSchema=True`).
3. Провести попередню обробку:
   - Перетворити колонку `InvoiceDate` з рядкового типу в тип дати.
   - Відфільтрувати записи, де `Quantity` є негативним (повернення товарів) та де відсутній `CustomerID`.

**Завдання 1.2. Агрегація даних**

1. Агрегувати дані для отримання сумарної кількості (`Quantity`) проданих товарів за день для кожної країни та кожного товару.
2. Створити новий DataFrame, що містить колонки: `InvoiceDate`, `Country`, `StockCode`, та сумарну `Quantity`.

**Завдання 1.3. Розбиття набору даних**

1. Розділити агреговані дані на два набори: тренувальний та тестовий.
2. **Тренувальний набір** повинен містити всі дані до **25 вересня 2011 року** (включно).
3. **Тестовий набір** повинен містити всі дані після 25 вересня 2011 року.
4. Зберегти тренувальний набір у змінну `pd_daily_train_data`.

**Приклад коду для ініціалізації Spark:**

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, to_date, to_timestamp

# Initialize Spark session
my_spark = SparkSession.builder.appName("SalesForecast").getOrCreate()

# Importing sales data
sales_data = my_spark.read.csv(
    "Online Retail.csv", header=True, inferSchema=True, sep=",")

# Convert InvoiceDate to datetime
sales_data = sales_data.withColumn("InvoiceDate", to_date(
```

```
    to_timestamp(col("InvoiceDate"), "d/M/yyyy H:mm")))

# Display schema and first few rows
sales_data.printSchema()
sales_data.show(5)
```

**Інструментарій**

**Програмне забезпечення:**

- **Python 3.8+**
- **Apache Spark**
- **Jupyter Notebook** або будь-яке інше середовище для роботи з Python.

**Приклад оформлення коду проєкту** доступний у файлі: `Workshop/src/notebook.ipynb` репозиторію курсу: https://github.com/TNTU-F2-SE-PhD/SE-PhD-Distributed-and-parallel-computing-systems

**Очікувані результати**

1. Налаштоване середовище для роботи з PySpark.
2. Spark DataFrame із завантаженими та очищеними даними.
3. Два DataFrame: один для тренування, інший для тестування моделі, розділені за вказаною датою.
4. Звіт з описом виконаних кроків та структури отриманих даних.

**Контрольні питання**

1. Що таке SparkSession і яка її роль у роботі з Apache Spark?
2. Поясніть різницю між трансформаціями та діями (actions) у Spark. Наведіть приклади.
3. Чому важливо фільтрувати дані перед їх аналізом та моделюванням?
4. Які переваги надає використання DataFrame API у PySpark порівняно з традиційними бібліотеками, як-от Pandas, при роботі з великими даними?
5. Чому для задач прогнозування часових рядів дані розділяють на тренувальний та тестовий набори за часовою ознакою, а не випадковим чином?

# Практична робота №2

## Інженерія ознак та побудова конвеєра обробки

### Мета роботи

Освоїти техніки інженерії ознак (feature engineering) у середовищі PySpark. Навчитися створювати та трансформувати прогностичні змінні. Побудувати автоматизований конвеєр (pipeline) обробки даних за допомогою бібліотеки Spark MLlib для підготовки даних до моделювання.

### Продовження наскрізного проєкту

У цій роботі ми продовжуємо розробку системи прогнозування попиту. Використовуючи агреговані дані, отримані в ПР№1, ми створимо нові ознаки, що допоможуть моделі краще зрозуміти закономірності в даних, та об'єднаємо всі кроки трансформації в єдиний конвеєр.

### Теоретичні відомості

**Інженерія ознак** — це процес створення нових, більш інформативних ознак (features) з наявних сирих даних. Це один з найважливіших етапів у побудові моделей машинного навчання, оскільки якість ознак безпосередньо впливає на точність прогнозу.

**Spark MLlib** — бібліотека машинного навчання в Apache Spark, що надає інструменти для трансформації даних, побудови моделей та їх оцінки в розподіленому середовищі. Ключові компоненти, що використовуються в цій роботі:

- **Transformer:** Алгоритм, що перетворює один DataFrame на інший. Приклади: `StringIndexer`, `VectorAssembler`.
- **Estimator:** Алгоритм, який навчається на DataFrame для створення Transformer. Приклад: `RandomForestRegressor`.
- **Pipeline:** Дозволяє об'єднувати декілька етапів (Transformers та Estimators) у єдиний робочий процес, що спрощує та автоматизує обробку даних.

Основні трансформатори, що будуть використані:

- `StringIndexer`: Перетворює категоріальні ознаки (рядки) в числові індекси. Більшість алгоритмів машинного навчання не можуть працювати з текстовими даними напряму.
- `VectorAssembler`: Об'єднує декілька числових колонок в одну векторну колонку, яка є стандартним вхідним форматом для моделей в Spark MLlib.

### Завдання

#### Завдання 2.1. Інженерія часових ознак

1. З колонки `InvoiceDate` створити нові ознаки, що можуть фіксувати сезонність та тренди:
    - `Year` (рік)
    - `Month` (місяць)
    - `WeekOfYear` (тиждень року)
    - `DayOfMonth` (день місяця)
    - `DayOfWeek` (день тижня)
2. Використовувати відповідні функції з `pyspark.sql.functions`.

**Завдання 2.2. Індексація категоріальних ознак**

1. Створити два екземпляри `StringIndexer` для перетворення колонок `Country` та `StockCode` в числові індекси.
2. Вхідними колонками будуть `Country` та `StockCode`, а вихідними — `CountryIndex` та `StockCodeIndex`.

**Завдання 2.3. Створення вектора ознак**

1. Створити екземпляр `VectorAssembler`.
2. Вказати вхідні колонки: всі нові часові ознаки та індексовані категоріальні ознаки.
3. Вказати вихідну колонку — `features`.

**Завдання 2.4. Побудова та застосування конвеєра (Pipeline)**

1. Створити об'єкт `Pipeline`, що послідовно виконуватиме всі створені етапи: два `StringIndexer` та один `VectorAssembler`.
2. Навчити конвеєр на тренувальному наборі даних, отриманому в ПР№1.
3. Трансформувати тренувальний набір даних за допомогою навченого конвеєра.

**Приклад коду для створення конвеєра:**

```
from pyspark.ml import Pipeline
from pyspark.ml.feature import StringIndexer, VectorAssembler
from pyspark.sql.functions import year, month, weekofyear, dayofmonth, dayofweek

# 1. Feature Engineering
data_with_features = daily_aggregated_data.withColumn("Year", year(col("InvoiceDate")))
# ... add other time features

# 2. String Indexing
country_indexer = StringIndexer(inputCol="Country", outputCol="CountryIndex")
```

```
stock_indexer = StringIndexer(inputCol="StockCode", outputCol="StockCodeIndex")

# 3. Vector Assembler
assembler = VectorAssembler(
    inputCols=["Year", "Month", "WeekOfYear", "DayOfMonth", "DayOfWeek",
               "CountryIndex", "StockCodeIndex"],
    outputCol="features")

# 4. Pipeline
pipeline = Pipeline(stages=[country_indexer, stock_indexer, assembler])

# Fit and transform the data
model = pipeline.fit(train_data)
transformed_train_data = model.transform(train_data)
```

**Інструментарій**

**Програмне забезпечення:**

- **Python 3.8+**
- **Apache Spark**
- **Jupyter Notebook**

**Очікувані результати**

1. DataFrame з доданими часовими ознаками.
2. Навчений об'єкт `PipelineModel`, що містить всі етапи трансформації.
3. Трансформований тренувальний DataFrame, що містить векторну колонку `features` і готовий для передачі в модель машинного навчання.
4. Звіт з описом створених ознак та структури конвеєра.

**Контрольні питання**

1. Що таке інженерія ознак і чому вона є критично важливою для успіху проєкту машинного навчання?
2. Поясніть призначення `StringIndexer`. Чому не можна просто передати текстові дані в регресійну модель?
3. Яку роль виконує `VectorAssembler` і чому він є обов'язковим кроком для більшості моделей в Spark MLlib?

4. Які переваги надає використання `Pipeline` для організації робочого процесу машинного навчання?

5. Як `Pipeline` допомагає уникнути витоку даних (data leakage) з тестового набору в тренувальний?

# Практична робота №3

## Побудова та оцінка моделі машинного навчання

### Мета роботи

Навчитися будувати та навчати регресійні моделі за допомогою бібліотеки Spark MLlib. Інтегрувати модель машинного навчання в існуючий конвеєр обробки даних. Оцінити якість прогнозування за допомогою релевантних метрик.

### Розвиток наскрізного проєкту

На цьому етапі ми використаємо підготовлені дані та конвеєр трансформації з попередніх робіт для побудови фінальної моделі прогнозування. Ми навчимо модель на тренувальних даних, зробимо прогнози на тестовому наборі та оцінимо її точність.

### Теоретичні відомості

**Регресійні моделі** в машинному навчанні використовуються для прогнозування неперервних значень. У нашому випадку ми прогнозуємо кількість проданих товарів (`Quantity`).

**Random Forest Regressor** (Регресор на основі випадкового лісу) — це ансамблевий алгоритм, який будує велику кількість дерев рішень під час тренування та видає середнє значення їхніх прогнозів. Його переваги:

- Висока точність та стійкість до перенавчання.
- Здатність працювати з великою кількістю ознак.
- Невимогливість до масштабування даних.

**Оцінка моделі.** Для оцінки якості регресійної моделі використовуються спеціальні метрики. У Spark MLlib для цього існує `RegressionEvaluator`.

- **Mean Absolute Error (MAE)** — середня абсолютна помилка. Показує, на скільки в середньому прогноз моделі відхиляється від реального значення. Це інтуїтивно зрозуміла метрика, що вимірюється в тих же одиницях, що й цільова змінна (у нашому випадку — в одиницях товару).

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

де $y_i$ — реальне значення, а $\hat{y}_i$ — прогнозоване.

### Завдання

#### Завдання 3.1. Створення регресійної моделі

1. Створити екземпляр `RandomForestRegressor`.

2. Вказати вхідну колонку з ознаками (`featuresCol='features'`) та колонку з цільовою змінною (`labelCol='Quantity'`).

**Завдання 3.2. Додавання моделі до конвеєра**

1. Розширити конвеєр, створений у ПР№2, додавши до нього регресор як останній етап.
2. Новий конвеєр буде складатися з трьох етапів: два `StringIndexer`, один `VectorAssembler` та один `RandomForestRegressor`.

**Завдання 3.3. Навчання моделі**

1. Навчити повний конвеєр на тренувальному наборі даних за допомогою методу `fit()`.
2. Результатом буде навчена модель (`PipelineModel`), яка містить усі налаштовані трансформатори та навчений регресор.

**Завдання 3.4. Оцінка якості моделі**

1. Застосувати навчену модель до тестового набору даних за допомогою методу `transform()`, щоб отримати прогнози.
2. Створити екземпляр `RegressionEvaluator`.
3. Вказати необхідні параметри: `labelCol='Quantity'`, `predictionCol='prediction'`, `metricName='mae'`.
4. Обчислити MAE на прогнозах, отриманих на тестовому наборі. Зберегти результат у змінну `mae`.

**Приклад коду для навчання та оцінки:**

```
from pyspark.ml.regression import RandomForestRegressor
from pyspark.ml.evaluation import RegressionEvaluator

# 1. Create the model
rf = RandomForestRegressor(featuresCol='features', labelCol='Quantity')

# 2. Add the model to the pipeline
# The pipeline from PR-2 had 3 stages
pipeline = Pipeline(stages=[country_indexer, stock_indexer, assembler, rf])

# 3. Train the model
model = pipeline.fit(train_data)

# 4. Make predictions and evaluate
predictions = model.transform(test_data)
```

```
evaluator = RegressionEvaluator(
    labelCol="Quantity", predictionCol="prediction", metricName="mae")
mae = evaluator.evaluate(predictions)

print(f"Mean Absolute Error (MAE) on test data: {mae}")
```

**Інструментарій**

**Програмне забезпечення:**

- **Python 3.8+**
- **Apache Spark**
- **Jupyter Notebook**

**Очікувані результати**

1. Повністю навчена модель `PipelineModel`, готова до використання.
2. DataFrame з прогнозами для тестового набору даних.
3. Розраховане значення метрики MAE, що характеризує точність моделі.
4. Звіт з аналізом отриманої точності та висновками щодо ефективності моделі.

**Контрольні питання**

1. Поясніть різницю між регресією та класифікацією в машинному навчанні.
2. Як працює алгоритм випадкового лісу? Які його переваги та недоліки?
3. Що означає отримане значення MAE в контексті бізнес-задачі? (Наприклад, якщо MAE = 10, що це означає для компанії?)
4. Чому важливо оцінювати модель на тестових даних, які не використовувалися під час навчання?
5. Які ще метрики, крім MAE, можна використовувати для оцінки регресійних моделей і в яких випадках вони можуть бути кращими?

# Практична робота №4

## Оптимізація та інтерпретація результатів

### Мета роботи

Навчитися використовувати навчену модель машинного навчання для отримання конкретних бізнес-інсайтів. Освоїти методи інтерпретації результатів моделі та розглянути підходи до її оптимізації для покращення точності прогнозів.

### Завершення наскрізного проєкту

Це фінальна частина нашого наскрізного проєкту. Ми використаємо побудовану та навчену модель для відповіді на конкретне бізнес-питання, проаналізуємо, які фактори найбільше впливають на прогнози, та обговоримо шляхи подальшого вдосконалення системи прогнозування.

### Теоретичні відомості

**Інтерпретація моделі** — це процес пояснення, чому модель приймає ті чи інші рішення. Для бізнесу недостатньо просто отримати прогноз; важливо розуміти, що стоїть за цим прогнозом. У моделях, побудованих на основі дерев рішень, як-от `RandomForestRegressor`, можна отримати **важливість ознак (feature importances)**. Цей показник демонструє, який внесок кожна ознака (наприклад, країна, день тижня, товар) робить у точність моделі.

**Оптимізація моделі** — це процес налаштування її параметрів для досягнення кращої продуктивності. Один з найпоширеніших методів — це **пошук по сітці (Grid Search) з перехресною валідацією (Cross-Validation)**. У PySpark для цього використовуються інструменти `ParamGridBuilder` та `CrossValidator`. Вони дозволяють автоматично перебрати різні комбінації гіперпараметрів моделі (наприклад, кількість дерев у лісі) та вибрати найкращу на основі метрики якості.

**Бізнес-застосування прогнозу.** Кінцева мета моделювання — допомогти бізнесу приймати кращі рішення. Наприклад, прогноз продажів на конкретний тиждень дозволяє оптимізувати логістику, спланувати маркетингові акції та уникнути дефіциту товарів на складі.

### Завдання

#### Завдання 4.1. Прогнозування на конкретний період

1. Використовуючи навчений `Pipeline` з попередньої роботи, зробіть прогнози для тестового набору даних.
2. Відфільтруйте результати прогнозування, щоб отримати дані лише за **39-й тиждень 2011 року**.
3. Просумуйте значення з колонки `prediction` для відфільтрованих даних, щоб отримати загальну прогнозовану кількість товарів.

4. Збережіть результат в цілочисельну змінну `quantity_sold_w39`.

**Приклад коду для прогнозування та фільтрації:**

```
from pyspark.sql.functions import weekofyear, col, sum

# Make predictions on the test data
predictions = pipeline_model.transform(test_data)

# Filter for week 39 of 2011
predictions_w39 = predictions.filter(
    (weekofyear(col("InvoiceDate")) == 39) & (year(col("InvoiceDate")) == 2011)
)

# Calculate the total predicted quantity
total_quantity_w39 = predictions_w39.agg(
    sum("prediction").alias("total_quantity")
).collect()[0]["total_quantity"]

# Store as an integer
quantity_sold_w39 = int(total_quantity_w39)

print(f"Predicted quantity sold in week 39 of 2011: {quantity_sold_w39}")
```

**Завдання 4.2. Інтерпретація моделі**

1. Отримайте доступ до навченої моделі `RandomForestRegressor` з об'єкта `PipelineModel`.
2. Витягніть важливість ознак за допомогою атрибута `featureImportances`.
3. Створіть візуалізацію (наприклад, стовпчасту діаграму), яка показує, які ознаки є найвпливовішими для моделі. Для цього може знадобитися отримати імена ознак з `VectorAssembler`.

**Завдання 4.3. Аналіз результатів**

1. Проаналізуйте отримані результати важливості ознак. Які фактори (країна, товар, час) найбільше впливають на обсяги продажів?
2. Як отриманий прогноз на 39-й тиждень може бути використаний S&OP командою для планування? Опишіть можливі бізнес-рішення.

**Завдання 4.4 (Додатково). Оптимізація моделі**

1. Створіть сітку параметрів (`ParamGridBuilder`) для `RandomForestRegressor`, вказавши декілька значень для гіперпараметрів, наприклад, `numTrees` (кількість дерев) та `maxDepth` (максимальна глибина дерева).

2. Налаштуйте `CrossValidator` для пошуку найкращої моделі з використанням створеної сітки параметрів та `RegressionEvaluator`.

3. Навчіть `CrossValidator` на тренувальних даних та порівняйте метрику MAE нової, оптимізованої моделі з результатом попередньої роботи.

**Очікувані результати**

1. Прогноз кількості товарів, що будуть продані на 39-му тижні 2011 року.

2. Аналіз та візуалізація важливості ознак, що впливають на прогноз.

3. Письмові висновки щодо бізнес-цінності отриманих результатів.

4. (Додатково) Оптимізована модель з потенційно кращою метрикою якості.

5. Фінальний звіт, що описує весь процес від аналізу даних до інтерпретації результатів та відповіді на бізнес-питання.

**Рекомендована література**

Для глибшого вивчення теми рекомендується звернутися до наступних джерел: [1], [2], [3], [4], [5].

**Контрольні питання**

1. Що таке важливість ознак і чому вона є корисною для бізнесу?

2. Як інтерпретація моделі допомагає підвищити довіру до систем штучного інтелекту?

3. Поясніть принцип роботи перехресної валідації. Чому вона є важливою при оптимізації гіперпараметрів?

4. Яка різниця між прогнозом моделі (наприклад, MAE = 15) та бізнес-інсайтом? Наведіть приклад.

5. Які ще бізнес-питання можна було б вирішити за допомогою побудованої моделі прогнозування попиту?

# Список джерел


[1] B. Chambers та M. Zaharia, *Spark: The Definitive Guide. Big Data Processing Made Simple*. O'Reilly Media, 2018.

[2] M. Kleppmann, *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems*. O'Reilly Media, 2017.

[3] H. Karau, A. Konwinski, P. Wendell та M. Zaharia, *Learning Spark*. O'Reilly Media, 2015.

[4] Apache Software Foundation, *Apache Spark Documentation. Best Practices and Optimization Techniques*, https://spark.apache.org/docs/latest/tuning.html, Accessed: 2025-08-17, 2025.

[5] Databricks, *Databricks Engineering Blog. Spark Performance and Optimization*, https://databricks.com/blog/category/engineering, Accessed: 2025-08-17, 2025.

# Додаток А. Приклад оформлення звіту практичної роботи

## Титульний аркуш звіту

Звіт про виконання практичної роботи повинен містити титульний аркуш наступного формату:

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Тернопільський національний технічний університет
імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем
та програмної інженерії

Кафедра програмної інженерії

**ЗВІТ**

про виконання практичної роботи № __

з дисципліни

**«Розподілені та паралельні системи обчислень»**

**Тема:** _____

**Виконав(ла):** _____
_____
____

**Перевірив(ла):** _____
____

Тернопіль — 2025

## Рекомендована структура звіту

Звіт повинен містити такі основні розділи:

### 1. Вступ

- Мета виконання практичної роботи
- Актуальність проблеми, що розглядається
- Задачі роботи (перелік із нумерацією)

### 2. Теоретичні основи

- Короткий виклад теоретичних концепцій, необхідних для розв'язання задачі
- Основні визначення та формули
- Огляд використаних методів

### 3. Практична реалізація

- Описання алгоритму розв'язання
- Вихідні код (або посилання на Jupyter notebook в репозиторії)
- Технічні особливості реалізації
- Використані бібліотеки та інструменти

### 4. Результати та аналіз

- Результати експериментів
- Візуалізація (графіки, таблиці, діаграми)
- Інтерпретація результатів
- Порівняння з очікуваними результатами

### 5. Висновки

- Підсумування основних результатів
- Відповідь на поставлені в роботі питання
- Рекомендації для подальших досліджень
- Аналіз труднощів, з якими довелося мати справу

**6. Список посилань**

- Список всіх використаних джерел
- Рекомендується використовувати стиль цитування IEEE
- Включити посилання на використані бібліотеки та датасети

## Контрольний список для звіту

Перед здачею переконайтеся, що звіт містить:

- Титулка з усіма необхідними реквізитами
- Вступ з ясною постановкою задачі
- Теоретичні основи методів, що використовуються
- Опис алгоритму та реалізації
- Результати експериментів з візуалізацією
- Аналіз та інтерпретація результатів
- Висновки, що узагальнюють проведену роботу
- Список посилань на використані джерела
- Послідовність сторінок відповідно до змісту
- Правильне оформлення таблиць, рисунків та рівнянь
- Послідовність нумерації та посилань
- Відсутність граматичних та синтаксичних помилок
- Посилання на вихідний код в репозиторії