**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

Інститут **КНІТ**

Кафедра **ПЗ**

### ЗВІТ

До лабораторної роботи № 2

**З дисципліни:** *“Технології ШІ в інженерії даних”*

**На тему:** *“Побудова рішення розпізнавання тексту з застосуванням ШІ”*

**Лектор:**

асист. каф. ПЗ

Капковський Ю.І.

**Виконав:**

ст. гр. ПЗ-45

Хруставчук М.Л.

**Прийняв:**

асист. каф. ПЗ

БаускО.Є.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2025 р.

∑= \_\_\_\_\_ .

Львів – 2025

**Тема роботи:** Побудова рішення розпізнавання тексту з застосуванням ШІ.

**Мета роботи:** Отримати практичні навички роботи з глибинним навчанням і розпізнавання різного роду текстових даних використовуючи нейронні мережі.

**TЕОРЕТИЧНІ ВІДОМОСТІ**

Глибинне навчання (Deep Learning) – це напрям штучного інтелекту, який базується на використанні багатошарових штучних нейронних мереж для автоматичного виділення ознак і побудови складних моделей прогнозування або розпізнавання. Глибинні моделі здатні самостійно навчатися з великих обсягів даних, знаходячи нелінійні залежності між вхідними ознаками та результатом. Основними архітектурами таких мереж є рекурентні (RNN) та згорткові (CNN) нейронні мережі.

Рекурентні нейронні мережі (RNN) використовуються для роботи з послідовними даними – текстом, аудіо або часовими рядами. Вони мають внутрішній стан (пам’ять), який дозволяє враховувати контекст попередніх елементів послідовності. Модифікації RNN, такі як LSTM (Long Short-Term Memory) і GRU (Gated Recurrent Unit), усувають проблему затухання або вибуху градієнтів, тому краще запам’ятовують довгострокові залежності у даних. Ці мережі широко застосовуються у розпізнаванні намірів користувачів, автоматичному перекладі, генерації тексту та голосових асистентах.

Згорткові нейронні мережі (CNN) призначені для обробки зображень та просторових структур даних. Вони використовують згорткові фільтри для автоматичного виділення важливих ознак, таких як контури, текстури чи форми. CNN стали основою сучасних систем комп’ютерного зору – від розпізнавання облич до автоматичного аналізу медичних знімків.

Таким чином, глибинне навчання забезпечує високу точність і гнучкість у задачах класифікації, прогнозування та розпізнавання. Завдяки комбінації великих датасетів, обчислювальних ресурсів і сучасних архітектур нейронних мереж, воно стало основним інструментом побудови інтелектуальних систем у сучасній інженерії даних.

**ЗАВДАННЯ**

Підзадача №1

1. Розробити ПЗ яке використовує вхідний датасет, вчить на ньому алгоритм поліноміальної регресії, потім демонструє результати порівняння роботи навченої моделі з test датасет.
2. Зробити аналогічне до пункту 1, але на базі алгоритму Elastic Net.
3. Зробити звіти порівняння отриманих результатів двома регресіями і сформулювати головні три висновки.

Підзадача №2

1. Розробити ПЗ яке використовує вхідний датасет, вчить на ньому алгоритм класифікації (логістична регресія), потім демонструє результати порівняння роботи навченої моделі з test датасет.

**ХІД ВИКОНАННЯ**

**1. Бізнес-задача та обґрунтування вибору датасету***.*

Для реалізації задач машинного навчання у сфері оцінки нерухомості було обрано датасет Home Value Insights, який містить 1000 записів та 7 характеристик житла: площу (Square\_Footage), кількість спалень (Num\_Bedrooms), кількість ванних кімнат (Num\_Bathrooms), рік побудови (Year\_Built), площу ділянки (Lot\_Size), розмір гаража (Garage\_Size) та якість району (Neighborhood\_Quality). Цільова змінна – House\_Price, що відображає реальну ринкову вартість житла у доларах.

Регресійна підзадача:

* Мета – передбачити ціну житла на основі його характеристик.
* Для цього застосовуються два алгоритми: поліноміальна регресія (для врахування нелінійних залежностей між площею, роком побудови та якістю району) і Elastic Net (для відбору найважливіших ознак та запобігання перенавчанню).
* Очікуваний результат – модель, здатна точно прогнозувати ринкову вартість житла, що може використовуватись для оцінки майна, планування інвестицій та аналізу впливу окремих факторів на ціну.

Класифікаційна підзадача:

* Мета – класифікувати житло за рівнем вартості: до 250 000 – низька категорія; 250 000 – 600 000 – середня; понад 600 000 млн – висока.
* Метод – логістична регресія.
* Очікуваний результат – автоматичне віднесення житла до відповідної цінової категорії, що спрощує аналіз ринку та підтримує ухвалення управлінських рішень.

Датасет має оптимальний розмір і чітко визначені змінні з реальним економічним змістом. Він є зручним для лабораторних досліджень, дозволяє тренувати та тестувати моделі регресії й класифікації, а також демонструє практичне застосування алгоритмів штучного інтелекту для вирішення реальних бізнес-задач у сфері інженерії даних.

**2. Порівняння результатів поліноміальної та Elastic Net регресій**

*Поліноміальна модель* показала високу точність прогнозування для всіх ступенів полінома, що свідчить про наявність виражених нелінійних залежностей між характеристиками житла (площа, рік побудови, якість району) та його ринковою ціною.

* При degree = 2: модель досягає коефіцієнта детермінації R² = 0.9932, що означає, що вона пояснює понад 99% варіації цін.  
  Середня абсолютна помилка (MAE) становить ≈ $15,879, що є прийнятним рівнем для задач оцінки вартості нерухомості.
* При degree = 3: точність ще зростає (R² = 0.9960), а помилки зменшуються (MAE = $12,730, RMSE = $16,138). Це свідчить, що третій ступінь краще моделює складні взаємозв’язки між ознаками.
* При degree = 4: незначне зниження R² (0.9955) і зростання RMSE ($17,103) свідчать про початок перенавчання – модель надто чутлива до коливань у даних.

Отже, оптимальним є degree **=** 3, оскільки він забезпечує найкращий баланс між точністю та узагальненням, демонструючи найнижчі похибки при стабільно високому R².

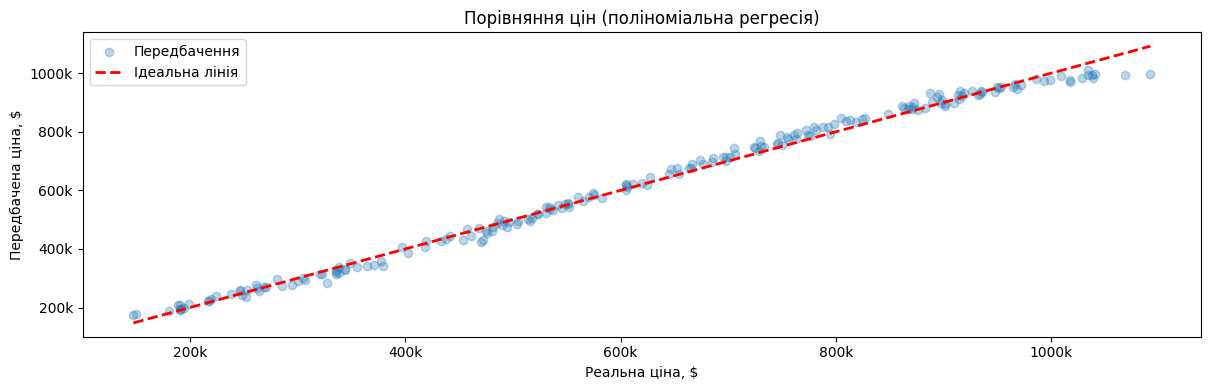


Рис. 1. Результат роботи поліноміальної регресії з degree = 2

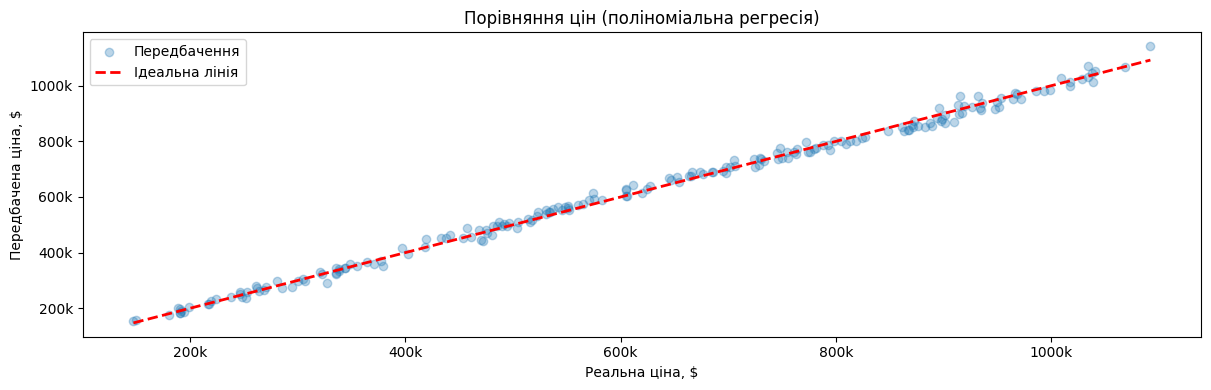


Рис. 2. Результат роботи поліноміальної регресії з degree = 3

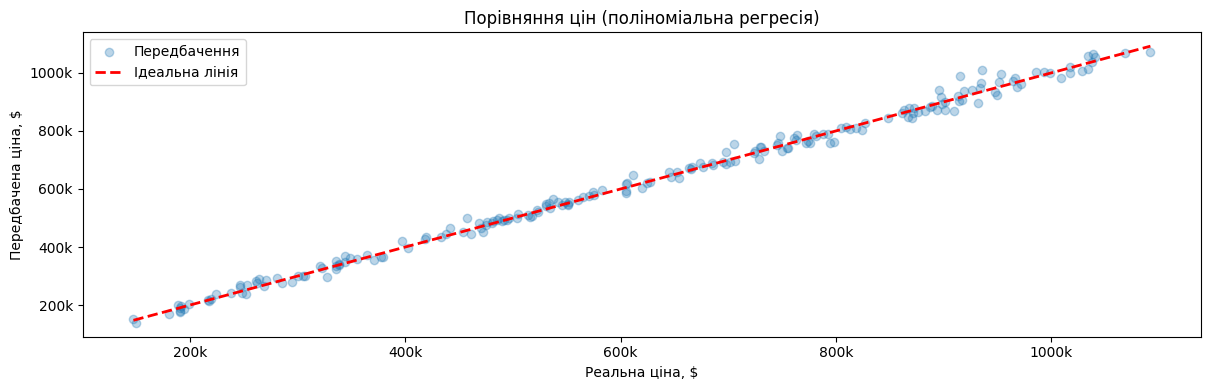


Рис. 3. Результат роботи поліноміальної регресії з degree = 4

*Elastic Net.* Для оцінки впливу сили регуляризації було проведено три експерименти з різними значеннями гіперпараметрів α та l1. Результати подано нижче:

* Експеримент №1: α = 0.1, l1\_ratio = 0.3 R² = 0.9666, MAE = $39,067.43, RMSE = $47,701.51 Модель показала високу точність і добре узагальнила дані, проте при надто малій регуляризації з’являються ризики часткового перенавчання.
* Експеримент №2: α = 0.5, l1\_ratio = 0.5 R² = 0.5628, MAE = $151,568.86, RMSE = $172,621.07 При середній силі регуляризації точність знижується, модель спрощується, але стає стійкішою до мультиколінеарності та шуму. Це компроміс між стабільністю та гнучкістю.
* Експеримент №3: α = 1.0, l1\_ratio = 0.8 R² = –0.1285, MAE = $238,258.30, RMSE = $277,336.31 Надмірна регуляризація призводить до значного зниження якості — модель стає надто простою і втрачає здатність відтворювати реальні залежності.

Отже, модель Elastic Net чутлива до вибору гіперпараметрів. Найкращий баланс між стабільністю та точністю досягається при α = 0.1, l1 = 0.3, коли R² перевищує 0.96. При надмірній регуляризації (α ≥ 0.5) модель занадто спрощується й втрачає прогностичну силу. Для задач із вираженими нелінійними залежностями, як у цьому датасеті, поліноміальна регресія залишається ефективнішою.

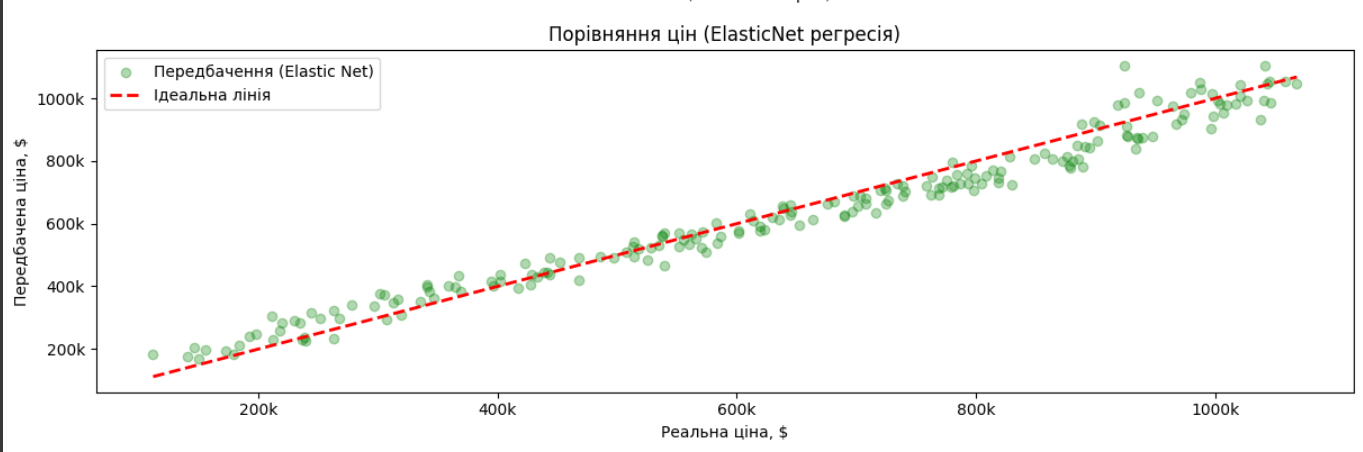


Рис. 4. Результат роботи Elastic Net при α = 0.1, l1 = 0.3

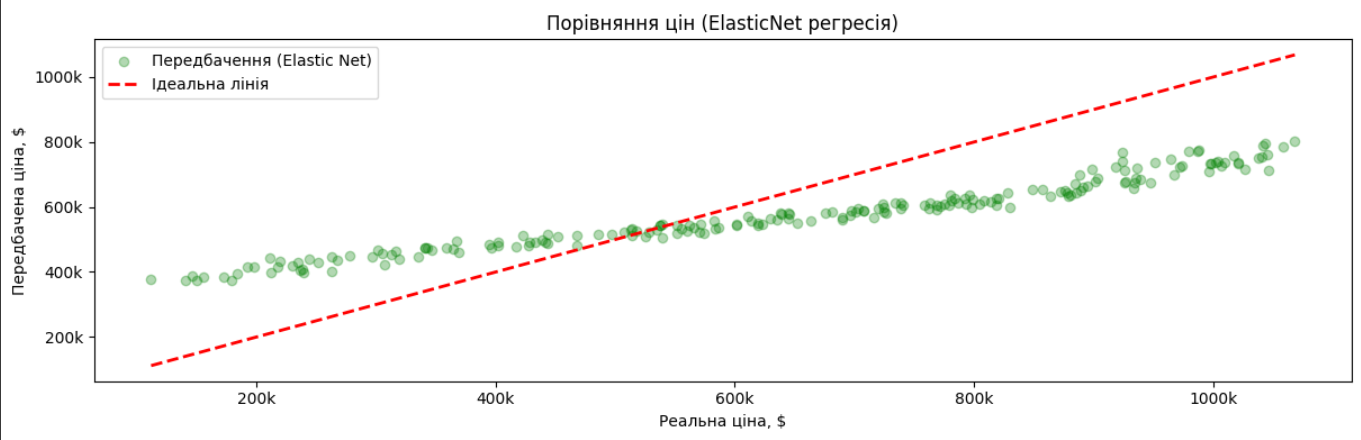


Рис. 5. Результат роботи Elastic Net при α = 0.5, l1 = 0.5

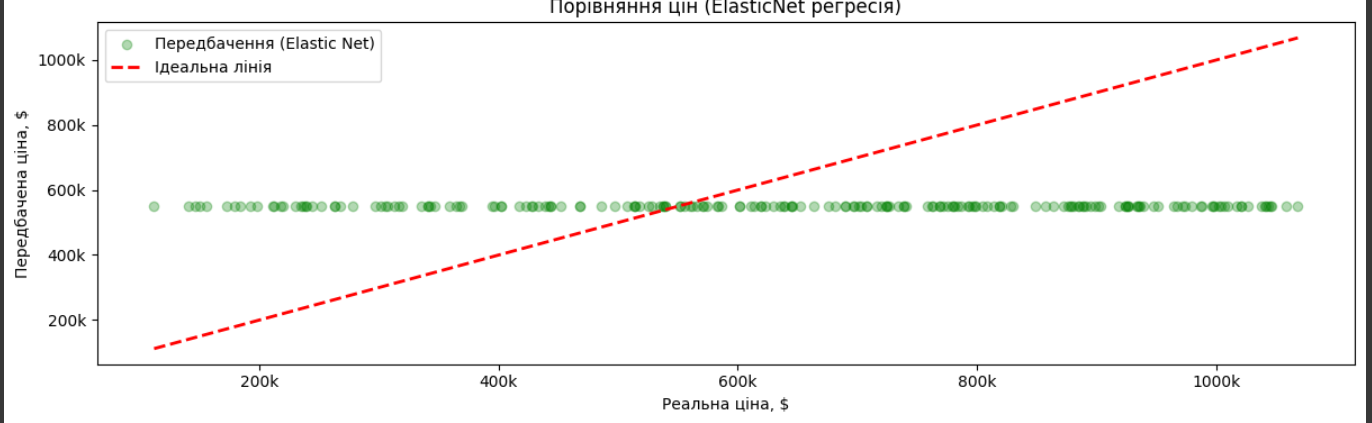


Рис. 6. Результат роботи Elastic Net при α = 1.0, l1 = 0.8

Ступінь поліноміальних ознак: 4. Для обох моделей (Polynomial Regression та Elastic Net) було використано поліноміальне розширення ознак четвертого ступеня, що дозволило врахувати нелінійні залежності між параметрами (наприклад, площа будинку, наявність гаража) та ціною.

Модель логістичної регресії тестувалась з трьома значеннями гіперпараметра C, який визначає силу регуляризації.

* При C = 0.1 модель має сильну регуляризацію, тому поводиться обережно й частково недонавчається. Є кілька помилкових класифікацій між класами High і Medium, але загалом результат стабільний.
* При C = 1 спостерігається найкращий баланс між узагальненням і точністю. Модель чітко розділяє всі три класи, показує найвищу точність і найменшу кількість помилок.
* При C = 10 регуляризація слабшає, модель починає підлаштовуватись під тренувальні дані. Результати залишаються добрими, але з’являються ознаки перенавчання.

Отже, логістична регресія найкраще працює коли C = 1.

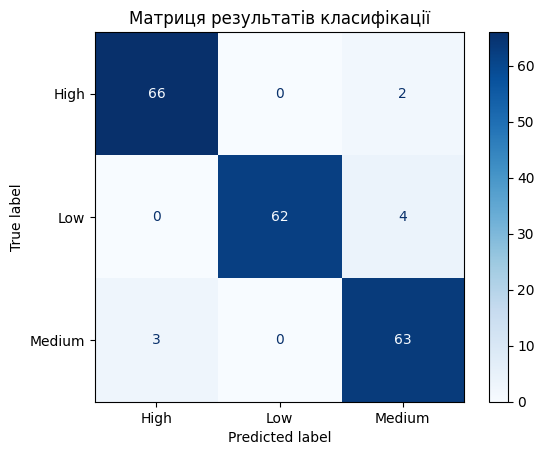


Рис. 7. Результат роботи логістичної регресії з C = 0.1

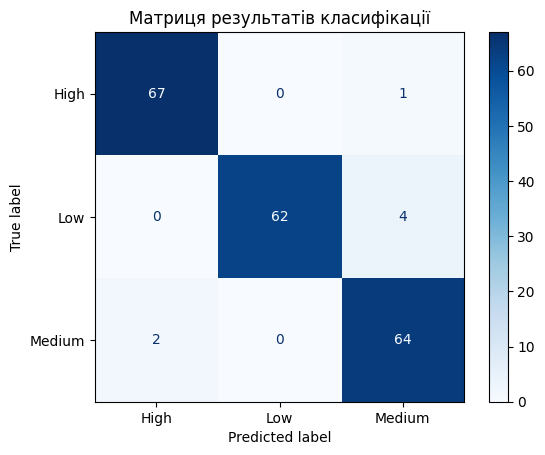


Рис. 8. Результат роботи логістичної регресії з C = 1

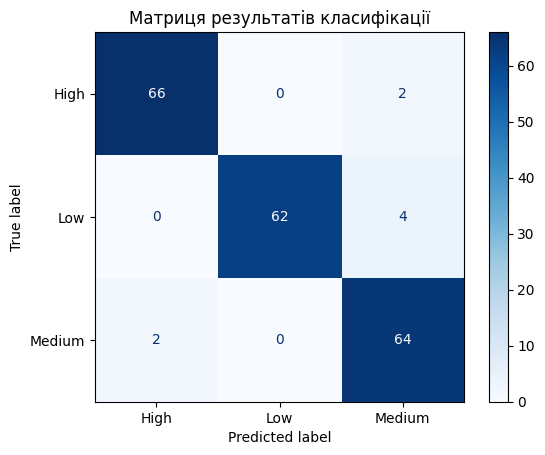


Рис. 9. Результат роботи логістичної регресії з C = 10

Файл Lab\_01.ipynb

# --- Підключення бібліотек ---

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from matplotlib.ticker import FuncFormatter

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import PolynomialFeatures, StandardScaler

from sklearn.linear\_model import LinearRegression, ElasticNet, LogisticRegression

from sklearn.metrics import r2\_score, mean\_absolute\_error, mean\_squared\_error, classification\_report, confusion\_matrix, ConfusionMatrixDisplay

from google.colab import drive

# --- Завантаження даних ---

drive.mount('/content/drive')

file\_path = "/content/drive/MyDrive/Colab Notebooks/Colab\_Data/Dataset2.csv"

df = pd.read\_csv(file\_path)

X = df.select\_dtypes(include=[np.number]).drop(columns=['House\_Price'])

y = df['House\_Price']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.2, random\_state=52

)

# Перевірка кореляції ознак

numeric\_features\_df = df.select\_dtypes(include=[np.number])

corr\_matrix = numeric\_features\_df.corr()

print("Кореляція всіх числових ознак з 'House\_Price':")

print(corr\_matrix['House\_Price'].sort\_values(ascending=False))

# Форматування

def fmt\_money\_k(x, pos):

return f'{int(x/1000)}k'

formatter = FuncFormatter(fmt\_money\_k)

# --- Створення та масштабування ознак ---

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

y\_train\_log = np.log1p(y\_train)

poly = PolynomialFeatures(degree=4)

X\_poly\_train = poly.fit\_transform(X\_train\_scaled)

X\_poly\_test = poly.transform(X\_test\_scaled)

scaler\_poly = StandardScaler()

X\_poly\_train\_scaled = scaler\_poly.fit\_transform(X\_poly\_train)

X\_poly\_test\_scaled = scaler\_poly.transform(X\_poly\_test)

print(f"Кількість вихідних ознак: {X\_train.shape[1]}")

print(f"Кількість поліноміальних ознак: {X\_poly\_train\_scaled.shape[1]}")

# --- Поліноміальна регресія ---

model\_poly\_log = LinearRegression()

model\_poly\_log.fit(X\_poly\_train\_scaled, y\_train\_log)

y\_pred\_poly\_log = np.expm1(model\_poly\_log.predict(X\_poly\_test\_scaled))

r2 = r2\_score(y\_test, y\_pred\_poly\_log)

mae = mean\_absolute\_error(y\_test, y\_pred\_poly\_log)

rmse = np.sqrt(mean\_squared\_error(y\_test, y\_pred\_poly\_log))

print("=== Поліноміальна регресія ===")

print(f"R²: {r2:.4f}")

print(f"MAE: ${mae:,.2f}")

print(f"RMSE: ${rmse:,.2f}")

# --- Візуалізація поліноміальної регресії ---

# Лінійний графік

plt.figure(figsize=(12,4))

plt.plot(range(len(y\_test)), y\_test.values, label="Реальна ціна", color='red', linewidth=2)

plt.plot(range(len(y\_test)), y\_pred\_poly\_log, label="Передбачена ціна", color='blue', linestyle='--')

plt.xlabel("Об'єкти (тестова вибірка)")

plt.ylabel("Ціна, $")

plt.title("Порівняння цін (поліноміальна регресія)")

plt.legend()

plt.tight\_layout()

ax = plt.gca()

ax.yaxis.set\_major\_formatter(formatter)

plt.show()

# Scatter plot ---

plt.figure(figsize=(12, 4))

plt.scatter(y\_test, y\_pred\_poly\_log, alpha=0.3, label='Передбачення')

plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'r--', linewidth=2, label='Ідеальна лінія')

plt.xlabel("Реальна ціна, $")

plt.ylabel("Передбачена ціна, $")

plt.title("Порівняння цін (поліноміальна регресія)")

plt.legend()

plt.tight\_layout()

ax = plt.gca()

ax.xaxis.set\_major\_formatter(formatter)

ax.yaxis.set\_major\_formatter(formatter)

plt.show()

# --- ElasticNet регресія ---

model\_en\_log = ElasticNet(alpha=0.5, l1\_ratio=0.5, random\_state=42, max\_iter=2000)

model\_en\_log.fit(X\_poly\_train\_scaled, y\_train\_log)

y\_pred\_en\_log = np.expm1(model\_en\_log.predict(X\_poly\_test\_scaled))

r2\_en = r2\_score(y\_test, y\_pred\_en\_log)

mae\_en = mean\_absolute\_error(y\_test, y\_pred\_en\_log)

rmse\_en = np.sqrt(mean\_squared\_error(y\_test, y\_pred\_en\_log))

print("=== Elastic Net ===")

print(f"R² (Elastic Net): {r2\_en:.4f}")

print(f"MAE (Elastic Net): ${mae\_en:,.2f}")

print(f"RMSE (Elastic Net): ${rmse\_en:,.2f}")

# --- Візуалізація ElasticNet регресії ---

# Лінійний графік

plt.figure(figsize=(12,4))

plt.plot(range(len(y\_test)), y\_test.values, label="Реальна ціна", color='red', linewidth=2)

plt.plot(range(len(y\_test)), y\_pred\_en\_log, label="Передбачена ціна", color='green', linestyle='--')

plt.xlabel("Об'єкти (тестова вибірка)")

plt.ylabel("Ціна, $")

plt.title("Порівняння цін (ElasticNet регресія)")

plt.legend()

plt.tight\_layout()

ax = plt.gca()

ax.yaxis.set\_major\_formatter(formatter)

plt.show()

# Scatter plot

plt.figure(figsize=(12, 4))

plt.scatter(y\_test, y\_pred\_en\_log, alpha=0.3, label='Передбачення (Elastic Net)', color='green')

plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'r--', linewidth=2, label='Ідеальна лінія')

plt.xlabel("Реальна ціна, $")

plt.ylabel("Передбачена ціна, $")

plt.title("Порівняння цін (ElasticNet регресія)")

plt.legend()

plt.tight\_layout()

ax = plt.gca()

ax.xaxis.set\_major\_formatter(formatter)

ax.yaxis.set\_major\_formatter(formatter)

plt.show()

# --- Порівняння результатів ---

results = pd.DataFrame({

'Model': ['Polynomial', 'Elastic Net'],

'R²': [r2, r2\_en],

'MAE': [mae, mae\_en],

'RMSE': [rmse, rmse\_en]

})

print(results)

# --- Підготовка даних для класифікації ---

df\_class = df.copy()

quantiles = df\_class['House\_Price'].quantile([0.33, 0.66])

low, high = quantiles[0.33], quantiles[0.66]

def categorize\_price(p):

if p <= low:

return 'Low'

elif p <= high:

return 'Medium'

else:

return 'High'

df\_class['Price\_Category'] = df\_class['House\_Price'].apply(categorize\_price)

X\_class = df\_class.select\_dtypes(include=[np.number]).drop(columns=['House\_Price'])

y\_class = df\_class['Price\_Category']

X\_train\_c, X\_test\_c, y\_train\_c, y\_test\_c = train\_test\_split(

X\_class, y\_class, test\_size=0.2, random\_state=42, stratify=y\_class

)

# --- Логістична регресія ---

scaler\_c = StandardScaler()

X\_train\_c\_scaled = scaler\_c.fit\_transform(X\_train\_c)

X\_test\_c\_scaled = scaler\_c.transform(X\_test\_c)

model\_log = LogisticRegression(C=10,

penalty='l2',

solver='lbfgs',

max\_iter=500,

random\_state=42)

model\_log.fit(X\_train\_c\_scaled, y\_train\_c)

y\_pred\_c = model\_log.predict(X\_test\_c\_scaled)

# Матриця результатів класифікації

print(classification\_report(y\_test\_c, y\_pred\_c))

cm = confusion\_matrix(y\_test\_c, y\_pred\_c, labels=model\_log.classes\_)

disp = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=model\_log.classes\_)

disp.plot(cmap='Blues', values\_format='d')

plt.title("Матриця результатів класифікації")

plt.show()

# Гістограма важливості ознак

importance = pd.Series(model\_log.coef\_[0], index=X\_class.columns)

importance\_abs = importance.abs().sort\_values(ascending=False)

plt.figure(figsize=(8, 5))

sns.barplot(x=importance\_abs.values, y=importance\_abs.index, hue=importance\_abs.index, legend=False, palette="Blues\_r")

plt.title("Гістограма важливості ознак")

plt.xlabel("Абсолютне значення коефіцієнта")

plt.ylabel("Ознака")

plt.tight\_layout()

plt.show()

**ВИСНОВКИ**

У ході лабораторної роботи було досліджено три алгоритми машинного навчання: поліноміальну регресію, Elastic Net та логістичну регресію. Поліноміальна регресія показала найкращі результати прогнозування цін житла, Elastic Net продемонструвала стабільність і здатність відсікати малозначущі ознаки, а логістична регресія ефективно класифікувала житло за ціновими категоріями. Отримані результати підтверджують, що вибір моделі залежить від характеру даних: для нелінійних залежностей доцільно використовувати поліноміальну регресію, для відбору ознак – Elastic Net, а для задач класифікації – логістичну регресію.