**ПРАКТИЧНА РОБОТА №2**

**ЛІНІЙНА РЕГРЕСІЯ**

Мета: Закріпити основні поняття використання моделі лінійної регресії в машинному навчанні.

Завдання:

* Прочитайте файл .csv у DataFrame pandas.
* Досліджуйте набір даних за допомогою бібліотек візуалізації Python.
* Експериментуйте з різними ознаками для створення моделі лінійної регресії.
* Налаштуйте гіперпараметри моделі.
* Порівняйте результати навчання, використовуючи середню квадратичну помилку та графіки втрат.
* Спробуйте побудувати модель та використати регресії для інших ознак та міток.

Хід роботи.

Для виконання завдання будемо використовувати файл з даними, який доступный за посиланням: [City of Chicago Taxi Trips dataset](https://www.google.com/url?q=https%3A%2F%2Fdata.cityofchicago.org%2FTransportation%2FTaxi-Trips%2Fwrvz-psew)

В ньому розташовані дані про таксі Чикаго. Перегляньте, яка інформація там є <https://data.cityofchicago.org/Transportation/Taxi-Trips-2024-/ajtu-isnz/about_data>

1 крок. Створіть новий jupyter notebook.

2. Імпортуйте необхідні бібліотеки



3. Завантажте Ваші данні:



4. Перегляньте перші декілька рядків завантажених даних



4. Оберіть лише ті колонки, які необхідні нам для навчання моделі та виведіть новий набір:



Опишіть у звіт, що означає кожен стовбчик.

5. Розрахуйте основні статистичні показники цього набору даних:



**6. Дайте відповіді на наступні запитання:**

**Яка максимальна плата?**

**Відповідь: $159.25**

**Яка середня відстань по всіх поїздках?**

**Відповідь: 8.2895 миль**

**Скільки таксомоторних компаній у наборі даних?**

**Відповідь: 31 Який найпоширеніший тип оплати?**

**Відповідь: Кредитна картка**

**Чи є які-небудь відсутні дані в характеристиках?**

Ні

Відповіді надані для того, щоб Ви могли перевірити себе.

Наприклад, для відповіді на перше запитання:



7. Побудова та аналіз матриці кореляцій

Важливою частиною машинного навчання є визначення, які ознаки корелюють із міткою.

 Якщо ви коли-небудь користувалися послугами таксі, ваш досвід, ймовірно, підказує вам, що вартість проїзду зазвичай асоціюється з відстанню та тривалістю поїздки. Але чи є спосіб дізнатися більше про те, наскільки добре ці ознаки корелюють із вартістю проїзду (міткою)?

 На цьому етапі ви використаєте \*матрицю кореляцій\*, щоб визначити ознаки, значення яких добре корелюють із міткою. Значення кореляції мають наступні значення:

\* 1.0`\*: ідеальна позитивна кореляція; тобто, коли одне значення зростає, інше значення також зростає.

\*`-1.0`\*: ідеальна негативна кореляція; тобто, коли одне значення зростає, інше значення зменшується.

\* `0.0`\*: відсутність кореляції; два стовпці [не мають лінійної залежності]

**Після побудови матриці ви повинні відповісти на запитання:**

 **Яка ознака має найсильнішу кореляцію з міткою FARE?**

**Яка ознака має найслабшу кореляцію з міткою FARE?**



8. Візуалізуємо наші дані у вигляді графіку



**Передивіться графіки та скажіть, чи підтверждують вони результати Вашого кореляційного аналізу**



9. Побудуємо модель лінійної регресії.

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_absolute\_error, r2\_score

# Визначення ознак та цільової змінної

X = dataset1[['TRIP\_MILES', 'TRIP\_SECONDS']]

y = dataset1['FARE']

# Розбиття на навчальний та тестовий набори

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model = LinearRegression()

# Навчання моделі

model.fit(X\_train, y\_train)

# Прогнозування на тестових даних

y\_pred = model.predict(X\_test)

# Оцінка моделі

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

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

print(f"Mean Absolute Error: {mae:.4f}")

print(f"R² Score: {r2:.4f}")

Отримаємо:



**Аналіз моделі**

**1. Оцінка якості прогнозу**

* **Mean Absolute Error (MAE) = 1.2064**
Це означає, що в середньому модель помиляється на $1.21 при прогнозуванні тарифу (FARE).
* **R² Score = 0.9561**
Це дуже високий показник, який означає, що 95.61% дисперсії значень FARE пояснюється змінними TRIP\_MILES і TRIP\_SECONDS.

**2. Можливі висновки**

* Високе значення R² свідчить про те, що модель добре підходить для прогнозування тарифу (FARE) на основі TRIP\_MILES та TRIP\_SECONDS.
* Низький MAE означає, що середня абсолютна помилка досить мала, що добре для точності передбачення.

**3. Можливі покращення**

1. **Додавання додаткових ознак**
	* PAYMENT\_TYPE (тип оплати: кредитна картка, готівка)
	* COMPANY (компанія, що надає таксі)
	* TIP\_RATE (чайові)
	* DAY\_OF\_WEEK (можливо, є сезонні коливання тарифів)
2. **Перевірка взаємозв'язку між змінними**
	* Можна візуалізувати FARE щодо TRIP\_MILES і TRIP\_SECONDS, щоб побачити, чи є нелінійні залежності.
	* Перевірити наявність мультиколінеарності між TRIP\_MILES і TRIP\_SECONDS (наприклад, якщо вони сильно корелюють, можна спростити модель).

10. Розробимо також модель нейронної мережі, яка дозволяє розв’язувати те ж завдання

import pandas as pd

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense

from tensorflow.keras.optimizers import Adam

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

from sklearn.preprocessing import StandardScaler

# Завантаження даних

training\_df = chicago\_taxi\_dataset[['TRIP\_MILES', 'TRIP\_SECONDS', 'FARE', 'COMPANY', 'PAYMENT\_TYPE', 'TIP\_RATE']]

# Попередня обробка даних

# Видаляємо не числові дані для простоти (у реальних сценаріях варто було б їх закодувати)

training\_df = training\_df.drop(columns=['COMPANY', 'PAYMENT\_TYPE'])

# Розділення даних на вхідні ознаки (X) та цільову змінну (y)

X = training\_df[['TRIP\_MILES', 'TRIP\_SECONDS']]

y = training\_df['FARE']

# Розділення даних на навчальний та тестовий набори

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Масштабування даних

scaler = StandardScaler()

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

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

# Створення моделі

model.add(Dense(1, input\_dim=2, activation='relu')) # Перший прихований шар із 1 нейрона

model.add(Dense(2, activation='relu')) # Другий прихований шар із 2 нейронами

model.add(Dense(1)) # Вихідний шар з 1 нейроном для регресії

# Компіляція моделі

model.compile(optimizer=Adam(learning\_rate=0.01), loss='mse') # Використовуємо середньоквадратичну помилку як функцію втрат

# Навчання моделі

model.fit(X\_train, y\_train, epochs=50, batch\_size=32, validation\_data=(X\_test, y\_test))

# Оцінка моделі

loss = model.evaluate(X\_test, y\_test)

print(f"Значення втрат на тестовому наборі: {loss}")

# Отримання ваг і зміщення

# Отримання всіх ваг і зміщень моделі

weights = model.get\_weights()

# Витягування ваг для першого шару

first\_layer\_weights = weights[0] # Матриця ваг розміром (2, 2)

first\_layer\_bias = weights[1] # Вектор зміщень розміром (2,)

# Вага для TRIP\_MILES (перший вхід) до першого нейрона

weight\_trip\_miles = first\_layer\_weights[0][0]

# Вага для TRIP\_SECONDS (другий вхід) до першого нейрона

weight\_trip\_seconds = first\_layer\_weights[1][0]

# Зміщення першого нейрона

bias\_value = first\_layer\_bias[0]

# Вивід значень

print(f"Вага для TRIP\_MILES: {weight\_trip\_miles}")

print(f"Вага для TRIP\_SECONDS: {weight\_trip\_seconds}")

print(f"Зміщення: {bias\_value}")

# Передбачення

predictions = model.predict(X\_test)

print(f"Передбачення: {predictions[:5].flatten()}")

Результат:



Напишіть, як можна використовувати цей результат.

10. Проведіть експерименти з гіперпараметрами моделі.

learning\_rate = 0.001: швидкість навчання, яка впливає на те, як швидко модель оновлює свої параметри під час навчання.

epochs = 20: кількість епох, або проходів через увесь навчальний набір даних.

batch\_size = 50: розмір пакета даних, який використовується для одного оновлення ваг моделі.

Створіть таблицю, в якій відобразіть, як впливає зміна одного або декілької параметрів одночасно на показник втрат.

11. Тепер використаємо нашу модель для прогнозування.



**Самостійно спробуйте побудувати модель лінійної регресії та нейронну мережу, яка дозволяє прогнозувати оплату в залежності тільки від часу поїздки, або рівень чайових від дальності.**