Лабораторна робота №6

На тему: "Наївний Байєс в Python"

Мета роботи: набути навичок працювати з даними і опанувати роботу у Python з використанням теореми Байєса.

Хід роботи:

Завдання 1. Теоретичні відомості

Опрацьовано теоретичні відомості щодо теореми Байєса та наївного байєсівського класифікатора.

Теорема Байєса описує ймовірність події, ґрунтуючись на попередньому знанні умов.

Типи класифікаторів: Гаусівський (Gaussian): для безперервних даних (нормальний розподіл).

Мультиноміальний (Multinomial): для дискретних даних (частота слів).

Бернуллі (Bernoulli): для бінарних ознак.

Застосування: фільтрація спаму, класифікація текстів, медична діагностика, прогнозування погоди.

Завдання 2. Аналіз прикладу прогнозування

Я проаналізував приклад, наведений у методичних вказівках, де на основі погодних умов (Outlook, Humidity, Wind) прогнозується можливість проведення гри (Play).

Суть методу: Будуються частотні таблиці для кожного атрибута.

Створюються таблиці правдоподібності (Likelihood tables), де розраховуються ймовірності (наприклад, P(Overcast|Yes).

За допомогою теореми Байєса обчислюються апостеріорні ймовірності для класів "Yes" і "No". Клас із найвищою ймовірністю стає результатом прогнозу.

Завдання 3. Прогнозування гри (Python)

Використовуючи дані з пункту 2, визначити програмно, чи відбудеться матч для мого варіанту.

Мій варіант (№8 згідно таблиці):

3, 8, 13	Outlook = Sunny Humidity = High Wind = Weak	Перспектива = Сонячно Вологість = Висока Вітер = Слабкий

Код програми:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.naive_bayes import CategoricalNB

data = {
    'Outlook': ['Sunny', 'Sunny', 'Overcast', 'Rain', 'Rain', 'Rain', 'Overcast',
'Sunny', 'Sunny', 'Rain', 'Sunny', 'Overcast', 'Overcast', 'Rain'],
    'Humidity': ['High', 'High', 'High', 'High', 'Normal', 'Normal', 'Normal',
'High', 'Normal', 'Normal', 'Normal', 'High', 'Normal', 'High'],
    'Wind': ['Weak', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong', 'Weak',
'Weak', 'Weak', 'Strong', 'Strong', 'Weak', 'Strong'],
    'Play': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes',
'Yes', 'Yes', 'Yes', 'No']
}
df = pd.DataFrame(data)

le = LabelEncoder()
for col in df.columns:
    df[col] = le.fit_transform(df[col])

X = df.drop('Play', axis=1)
y = df['Play']

model = CategoricalNB()
model.fit(X, y)

test_sample = [[2, 0, 1]]

prob = model.predict_proba(test_sample)
print(f"Ймовірність No: {prob[0][0]:.4f}")
print(f"Ймовірність Yes: {prob[0][1]:.4f}")
print(f"Прогноз: {'Yes' if prob[0][1] > 0.5 else 'No'}")
```

Результати роботи програми:

```
Ймовірність No: 0.5951
Ймовірність Yes: 0.4049
Прогноз: No
```

Рис. 1. Результат прогнозування.

Програма визначила, що при сонячній погоді, високій вологості та слабкому вітрі гра, найімовірніше, не відбудеться (No).

Завдання 4. Аналіз цін на квитки (Renfe)

Застосувати методи байєсівського аналізу до набору даних renfe_small.csv для класифікації цін на квитки.

Код програми:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report, accuracy_score
from sklearn.preprocessing import LabelEncoder

url = "https://raw.githubusercontent.com/susanli2016/Machine-Learning-with-
Python/master/data/renfe_small.csv"
df = pd.read_csv(url)

# Перетворюємо рядки дат у формат datetime
df['start_date'] = pd.to_datetime(df['start_date'])
df['end_date'] = pd.to_datetime(df['end_date'])

# Рахуємо різницю в часі та переводимо в хвилини
df['duration'] = (df['end_date'] - df['start_date']).dt.total_seconds() / 60

# Очищення даних
df['price'] = df['price'].fillna(df['price'].mean())
df = df.dropna()

# Створення цільової змінної (Класифікація: Дешевий/Дорогий)
median_price = df['price'].median()
df['price_category'] = (df['price'] > median_price).astype(int)

# Кодування категоріальних ознак
le = LabelEncoder()
for col in ['train_type', 'train_class', 'fare', 'origin', 'destination']:
    df[col] = le.fit_transform(df[col])

# Вибір ознак (тепер 'duration' існує)
features = ['train_type', 'train_class', 'fare', 'duration']
X = df[features]
y = df['price_category']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

gnb = GaussianNB()
gnb.fit(X_train, y_train)
y_pred = gnb.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

plt.figure(figsize=(10, 6))
```

```
sns.histplot(df['price'], kde=True, color='blue', bins=30)
plt.axvline(median_price, color='red', linestyle='--', label=f'Median Price
({median_price:.2f})')
plt.title('Розподіл цін на квитки')
plt.legend()
plt.show()
```

Результати:

```
Accuracy: 0.6520723876240514

Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.44      0.59      2945
           1       0.55      0.94      0.70      2194

    accuracy                           0.65      5139
   macro avg       0.73      0.69      0.64      5139
weighted avg       0.76      0.65      0.64      5139
```

Рис. 2. Показники ефективності класифікатора цін

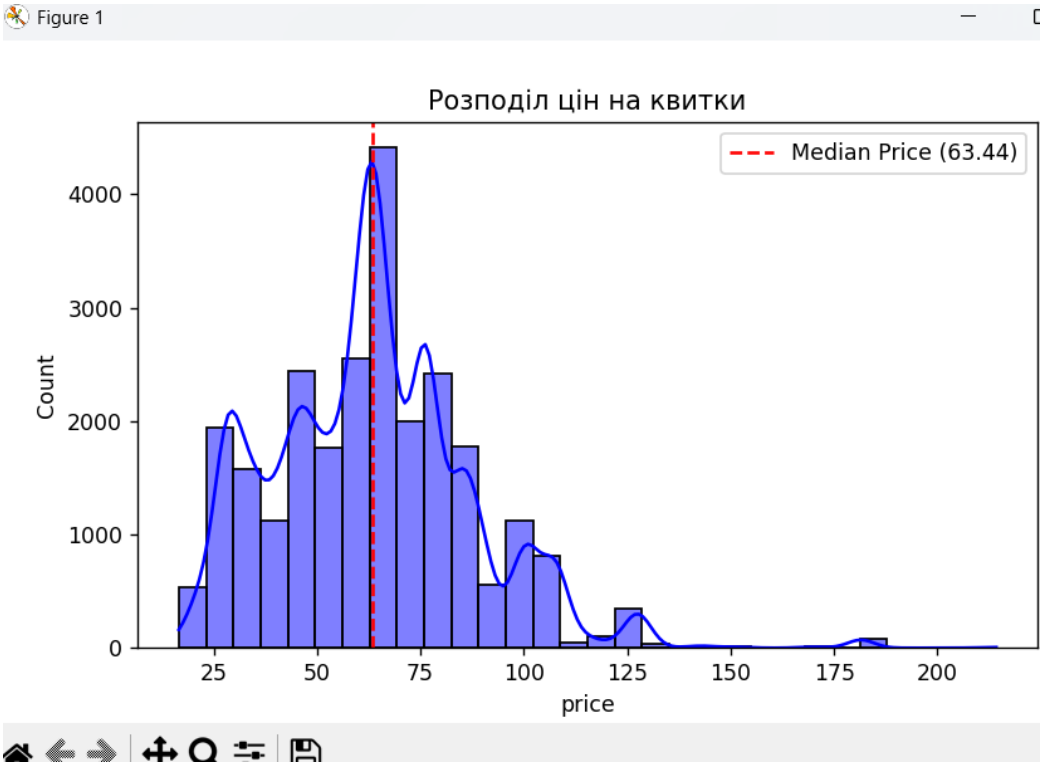

Рис. 3. Гістограма розподілу цін

Висновок: у ході лабораторної роботи я ознайомився з теоретичними основами теореми Байєса та типами наївних байєсівських класифікаторів.