ЛАБОРАТОРНА РОБОТА № 6

НАЇВНИЙ БАЙЄС В PYTHON

Мета роботи: набути навичок працювати з даними і опонувати роботу у Python з використанням теореми Байэса.

Сахно Андрій, 18 варіант

Завдання 1. Ретельно опрацювати теоретичні відомості.

Завдання 2. Ретельно розібрати приклад: прогнозування з використанням теореми Байєса.

Завдання 3. Використовуя данні з пункту 2 визначити відбудеться матч при наступних погодних умовах чи ні: Розрахунки провести з використанням Python.

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

Лістинг коду:

```
from collections import Counter

playData = [
    {"Day": "D1", "Outlook": "Sunny", "Humidity": "High", "Wind": "Weak",
"Play": "No"},
    {"Day": "D2", "Outlook": "Sunny", "Humidity": "High", "Wind": "Strong",
"Play": "No"},
    {"Day": "D3", "Outlook": "Overcast", "Humidity": "High", "Wind": "Weak",
"Play": "Yes"},
    {"Day": "D4", "Outlook": "Rain", "Humidity": "High", "Wind": "Weak",
"Play": "Yes"},
    {"Day": "D5", "Outlook": "Rain", "Humidity": "Normal", "Wind": "Weak",
"Play": "Yes"},
    {"Day": "D6", "Outlook": "Rain", "Humidity": "Normal", "Wind": "Strong",
"Play": "No"},
    {"Day": "D7", "Outlook": "Overcast", "Humidity": "Normal", "Wind":
"Strong", "Play": "Yes"},
    {"Day": "D8", "Outlook": "Sunny", "Humidity": "High", "Wind": "Weak",
"Play": "No"},
    {"Day": "D9", "Outlook": "Sunny", "Humidity": "Normal", "Wind": "Weak",
"Play": "Yes"},
```

```
    {"Day": "D10", "Outlook": "Rain", "Humidity": "Normal", "Wind": "Weak",
"Play": "Yes"},
    {"Day": "D11", "Outlook": "Sunny", "Humidity": "Normal", "Wind":
"Strong", "Play": "Yes"},
    {"Day": "D12", "Outlook": "Overcast", "Humidity": "High", "Wind":
"Strong", "Play": "Yes"},
    {"Day": "D13", "Outlook": "Overcast", "Humidity": "Normal", "Wind":
"Weak", "Play": "Yes"},
    {"Day": "D14", "Outlook": "Rain", "Humidity": "High", "Wind": "Strong",
"Play": "No"},
]



play_yes_count = Counter(row["Play"] for row in playData if row["Play"] ==
"Yes").total()
play_count = len(playData)
play_yes_prob = play_yes_count / play_count
print("Probability of the game being played: {0}/{1} = {2}"
      .format(play_yes_count, play_count, round(play_yes_prob, 3)))

sunny_yes_count = Counter(
    row["Outlook"] for row in playData if row["Play"] == "Yes" and
row["Outlook"] == "Sunny").total()
sunny_yes_prob = sunny_yes_count / play_yes_count
print("Probability of sunny for the game: {0}/{1} = {2}"
      .format(sunny_yes_count, play_yes_count, round(sunny_yes_prob, 3)))

humidity_high_yes_count = Counter(
    row["Humidity"] for row in playData if row["Play"] == "Yes" and
row["Humidity"] == "High").total()
humidity_high_yes_prob = humidity_high_yes_count / play_yes_count
print("Probability of high humidity for the game: {0}/{1} = {2}"
      .format(humidity_high_yes_count, play_yes_count,
round(humidity_high_yes_prob, 3)))

wind_weak_yes_count = Counter(
    row["Wind"] for row in playData if row["Play"] == "Yes" and row["Wind"]
== "Weak").total()
wind_weak_yes_prob = wind_weak_yes_count / play_yes_count
print("Probability of weak wind for the game: {0}/{1} = {2}"
      .format(wind_weak_yes_count, play_yes_count, round(wind_weak_yes_prob,
3)))



total_probability = play_yes_prob * sunny_yes_prob * humidity_high_yes_prob *
wind_weak_yes_prob
print("\nProbability of the game with outlook sunny, high humidity, weak
wind:"
      " \n{:.3f} * {:.3f} * {:.3f} * {:.3f} = {:.4f}"
      .format(play_yes_prob, sunny_yes_prob, humidity_high_yes_prob,
wind_weak_yes_prob, total_probability))
```

```
C:\Users\User\AppData\Local\Programs\Python\Python313\python.exe "C:\Users\User\Desktop\ua-un\7th semester\CWI\artificial-intelligence\lab6\LR_6_task_3.
Probability of the game being played: 9/14 = 0.643
Probability of sunny for the game: 2/9 = 0.222
Probability of high humidity for the game: 3/9 = 0.333
Probability of weak wind for the game: 6/9 = 0.667

Probability of the game with outlook sunny, high humidity, weak wind:
0.643 * 0.222 * 0.333 * 0.667 = 0.0317
```

Завдання 4. Застосуєте методи байєсівського аналізу до набору даних про ціни на квитки на іспанські високошвидкісні залізниці.

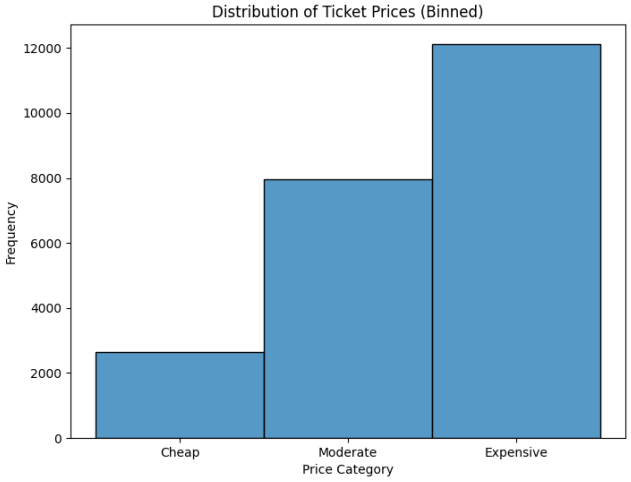
Лістинг коду:

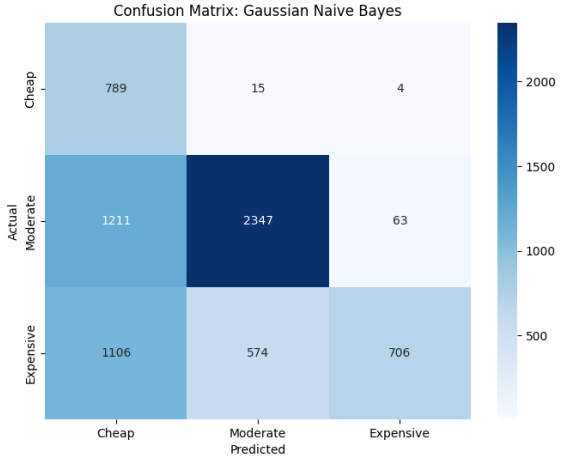
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB, MultinomialNB
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

# Load dataset from local file
df = pd.read_csv('ticket_data.csv')
df.dropna(inplace=True)

print(f"Dataset shape after removing NaN: {df.shape}")

# Preview data
print("Dataset preview:\n", df.head())

# Unique values for each column
print("\nUnique values per column:")
for col in df.columns:
    print(f"{col}: {df[col].nunique()} unique values")

# Focus on relevant columns (simplify for this example)
selected_cols = ["origin", "destination", "train_type", "train_class",
"fare", "price"]
df = df[selected_cols]

# Encode categorical features
df = pd.get_dummies(df, columns=["origin", "destination", "train_type",
"train_class", "fare"], drop_first=True)

# Splitting features and target
X = df.drop("price", axis=1)
y = df["price"]

# Binning the price into categories (cheap, moderate, expensive)
price_bins = [0, 30, 60, np.inf]
price_labels = ["Cheap", "Moderate", "Expensive"]
y_binned = pd.cut(y, bins=price_bins, labels=price_labels)

# Distribution of ticket prices
plt.figure(figsize=(8, 6))
sns.histplot(y_binned, kde=False)
plt.title("Distribution of Ticket Prices (Binned)")
plt.xlabel("Price Category")
plt.ylabel("Frequency")
plt.show()

# Splitting the data
X_train, X_test, y_train, y_test = train_test_split(X, y_binned,
test_size=0.3, random_state=42)

# Gaussian Naive Bayes
gnb = GaussianNB()
gnb.fit(X_train, y_train)
```

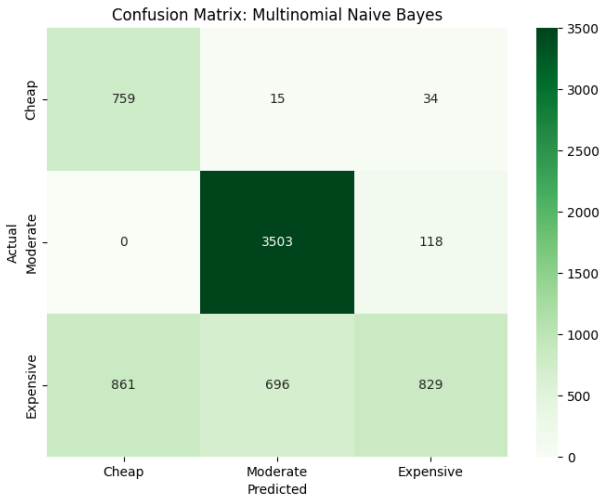
```python
y_pred = gnb.predict(X_test)

# Evaluate the Gaussian Naive Bayes model
print("\nGaussian Naive Bayes Classification Report:")
print(classification_report(y_test, y_pred))

# Confusion matrix for Gaussian Naive Bayes
cm_gnb = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm_gnb, annot=True, fmt="d", cmap="Blues",
xticklabels=price_labels, yticklabels=price_labels)
plt.title("Confusion Matrix: Gaussian Naive Bayes")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# Multinomial Naive Bayes (only for non-negative discrete features)
mnb = MultinomialNB()
X_train_discrete = (X_train * 100).astype(int)  # Scale to convert to
discrete
X_test_discrete = (X_test * 100).astype(int)
mnb.fit(X_train_discrete, y_train)
y_pred_mnb = mnb.predict(X_test_discrete)

# Evaluate the Multinomial Naive Bayes model
print("\nMultinomial Naive Bayes Classification Report:")
print(classification_report(y_test, y_pred_mnb))

# Confusion matrix for Multinomial Naive Bayes
cm_mnb = confusion_matrix(y_test, y_pred_mnb)
plt.figure(figsize=(8, 6))
sns.heatmap(cm_mnb, annot=True, fmt="d", cmap="Greens",
xticklabels=price_labels, yticklabels=price_labels)
plt.title("Confusion Matrix: Multinomial Naive Bayes")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# Accuracy comparison
print("\nAccuracy Comparison:")
print(f"Gaussian Naive Bayes Accuracy: {accuracy_score(y_test, y_pred):.2f}")
print(f"Multinomial Naive Bayes Accuracy: {accuracy_score(y_test,
y_pred_mnb):.2f}")
```

```
C:\Users\User\AppData\Local\Programs\Python\Python313\python.exe "C:\Users\User\Desktop\ua-un\7th semester\CWI\artificial-intelligence\lab6\LR_6_task_4.py"
Dataset shape after removing NaN: (22716, 9)
Dataset preview:
           insert_date    origin  ...        train_class     fare
1  2019-04-22 10:03:24    MADRID  ...            Turista    Promo
2  2019-04-25 19:19:46    MADRID  ...            Turista    Promo
3  2019-04-24 06:21:57   SEVILLA  ...  Turista con enlace  Promo +
4  2019-04-19 21:13:55  VALENCIA  ...            Turista    Promo
5  2019-04-13 14:05:18    MADRID  ...            Turista    Promo

[5 rows x 9 columns]

Unique values per column:
insert_date: 20270 unique values
origin: 5 unique values
destination: 5 unique values
start_date: 5163 unique values
end_date: 6396 unique values
train_type: 15 unique values
price: 182 unique values
train_class: 6 unique values
fare: 6 unique values
```

```
Gaussian Naive Bayes Classification Report:
              precision    recall  f1-score   support

       Cheap       0.25      0.98      0.40       808
   Expensive       0.80      0.65      0.72      3621
    Moderate       0.91      0.30      0.45      2386

    accuracy                           0.56      6815
   macro avg       0.66      0.64      0.52      6815
weighted avg       0.77      0.56      0.58      6815


Multinomial Naive Bayes Classification Report:
              precision    recall  f1-score   support

       Cheap       0.47      0.94      0.63       808
   Expensive       0.83      0.97      0.89      3621
    Moderate       0.85      0.35      0.49      2386

    accuracy                           0.75      6815
   macro avg       0.71      0.75      0.67      6815
weighted avg       0.79      0.75      0.72      6815


Accuracy Comparison:
Gaussian Naive Bayes Accuracy: 0.56
Multinomial Naive Bayes Accuracy: 0.75

Process finished with exit code 0
```







Результати аналізу показують, що класифікація цін на квитки на основі байєсівських методів дає різні результати залежно від обраного алгоритму. Gaussian Naive Bayes продемонстрував точність 56%, що вказує на труднощі з класифікацією через припущення про нормальний розподіл даних, яке не відповідає реальній природі набору даних. Зокрема, модель має високий рівень recall для класу "Дешеві" (98%), але низький precision (25%), що свідчить про велику кількість хибнопозитивних передбачень для цього класу. Клас "Помірні" виявився найважчим для класифікації через значне перекриття з іншими категоріями.

З іншого боку, Multinomial Naive Bayes досяг точності 75%, що демонструє кращу адаптованість до категоріальних та частотних даних. Модель значно краще передбачає класи "Дорогі" (precision 83%, recall 97%) та "Дешеві" (precision 47%, recall 94%). Однак класифікація "Помірних" залишається викликом: попри високий precision (85%), низький рівень recall (35%) свідчить про труднощі з розпізнаванням усіх екземплярів цього класу.

Загалом, результати вказують на доцільність використання Multinomial Naive Bayes для аналізу даного набору даних. Для подальшого покращення точності варто розглянути інші моделі, провести додаткову інженерію ознак

(наприклад, розрахунок тривалості подорожі) та глибше дослідити розподіл даних, особливо для "Помірних" цін.

Висновок: На лабораторній роботі я набув навичок працювати з даними і опонувати роботу у Python з використанням теореми Байэса.

https://github.com/andreylion06/artificial-intelligence