Assignment 6

Program:

```
import matplotlib
matplotlib.use('TkAgg')
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, accuracy_score
```

1) Read in the CSV file using pandas. Pay attention to the file delimeter. Inspect the resulting dataframe with respect to the column names and the variable types.

```
df = pd.read_csv('bank.csv', delimiter=';')
print(df.head())
print(df.info())
```

2) Pick data from the following columns to a second dataframe 'df2': y, job, marital, default, housing, poutcome.

```
df2 = df[['y','job','marital','default','housing','poutcome']]
```

3) Convert categorical variables to dummy numerical values using the command df3 = pd.get_dummies(df2,columns=['job','marital','default','housing','poutcome'])

```
df3 = pd.get_dummies(df2,columns=['job','marital','default','housing','poutcome'])
```

4) Produce a heat map of correlation coefficients for all variables in df3. Describe the amount of correlation between the variables in your own words.

```
df3['y'] = df3['y'].apply(lambda x: 1 if x == 'yes' else 0)
plt.figure(figsize=(10,6))
sns.heatmap(df3.corr(),cmap='coolwarm',annot=True,fmt='.2f')
plt.title('Correlation Heatmap')
plt.show()
```

5) Select the column called 'y' of df3 as the target variable y, and all the remaining columns for the explanatory variables X.

```
X = df3.drop(columns=['y'])
y = df3['y']
```

6) Split the dataset into training and testing sets with 75/25 ratio.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
random_state=42)
```

7) Setup a logistic regression model, train it with training data and predict on testing data.

```
log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train, y_train)
```

8) Print the confusion matrix (or use heat map if you want) and accuracy score for the logistic regression model.

```
y_pred_log_reg = log_reg.predict(X_test)

conf_matrix_log_reg = confusion_matrix(y_test, y_pred_log_reg)
accuracy_log_reg = accuracy_score(y_test, y_pred_log_reg)
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix_log_reg, annot=True, fmt='d', cmap='Blues')
plt.title(f'Logistic Regression Confusion Matrix\nAccuracy: {accuracy_log_reg:.2f}')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

9) Repeat steps 7 and 8 for k-nearest neighbors model. Use k=3, for example, or experiment with different values.

```
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
y_pred_knn = knn.predict(X_test)
conf_matrix_knn = confusion_matrix(y_test, y_pred_knn)
accuracy_knn = accuracy_score(y_test, y_pred_knn)
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix_knn, annot=True, fmt='d', cmap='Greens')
plt.title(f'K-Nearest Neighbors (k=3) Confusion Matrix\nAccuracy: {accuracy_knn:.2f}')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

10) Compare the results between the two models.

Logistic Regression performed better with higher accuracy and fewer misclassifications. K-Nearest Neighbors (K=3) was less accurate and suffered from high-dimensional one-hot encoded data. Logistic regression is also computationally faster. Logistically, logistic regression is the model to use for this dataset.

Output:

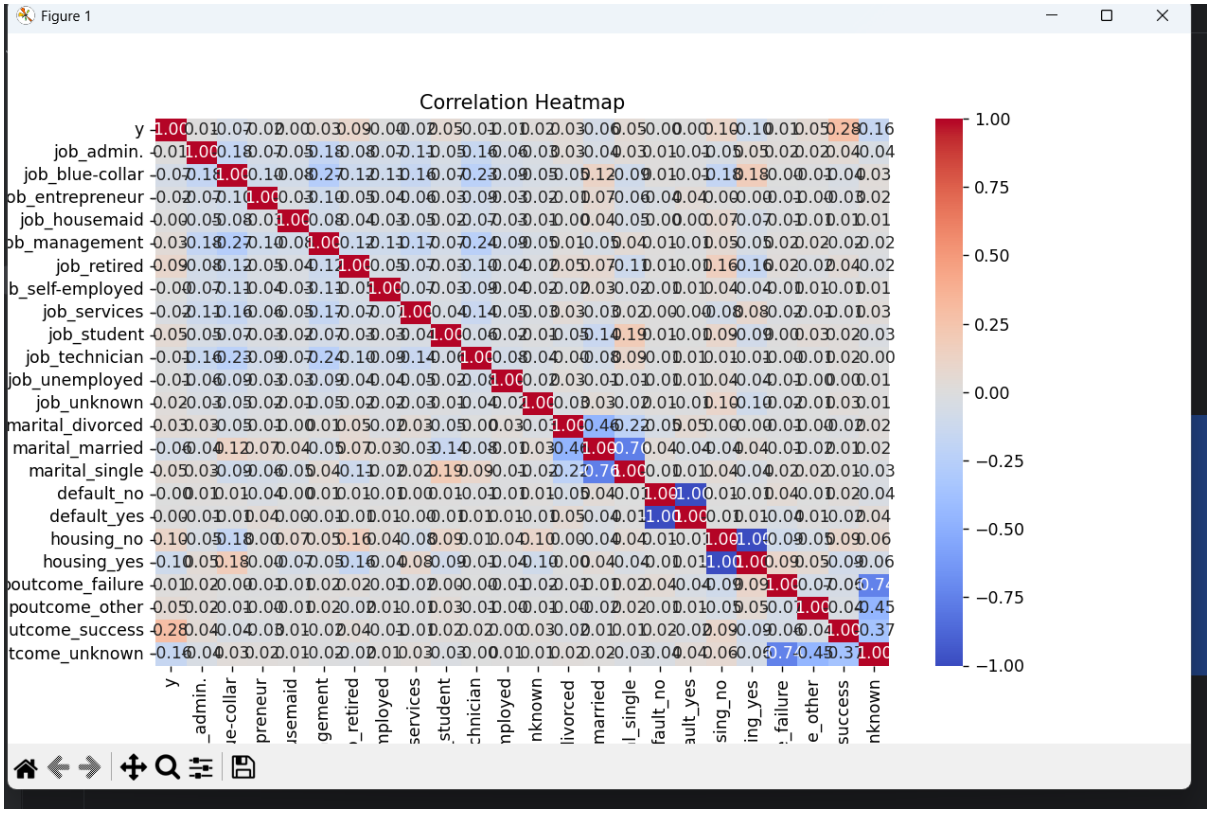


Figure 1
Correlation Heatmap
y
job_admin.
job_blue-collar
job_entrepreneur
job_housemaid
job_management
job_retired
job_self-employed
job_services
job_student
job_technician
job_unemployed
job_unknown
marital_divorced
marital_married
marital_single
default_no
default_yes
housing_no
housing_yes
poutcome_failure
poutcome_other
utcome_success
tcome_unknown
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

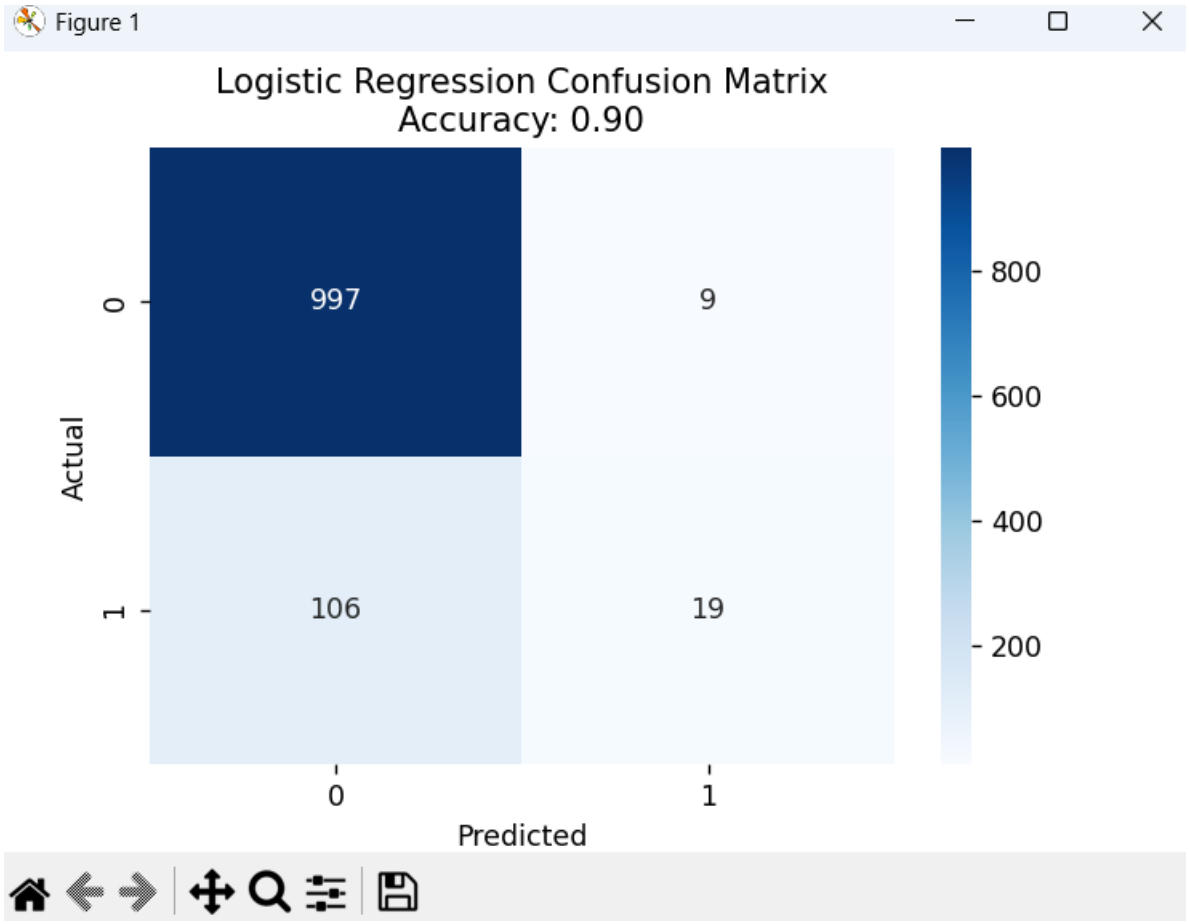


Figure 1
Logistic Regression Confusion Matrix
Accuracy: 0.90
997
9
106
19
Actual
Predicted
0
1
800
600
400
200

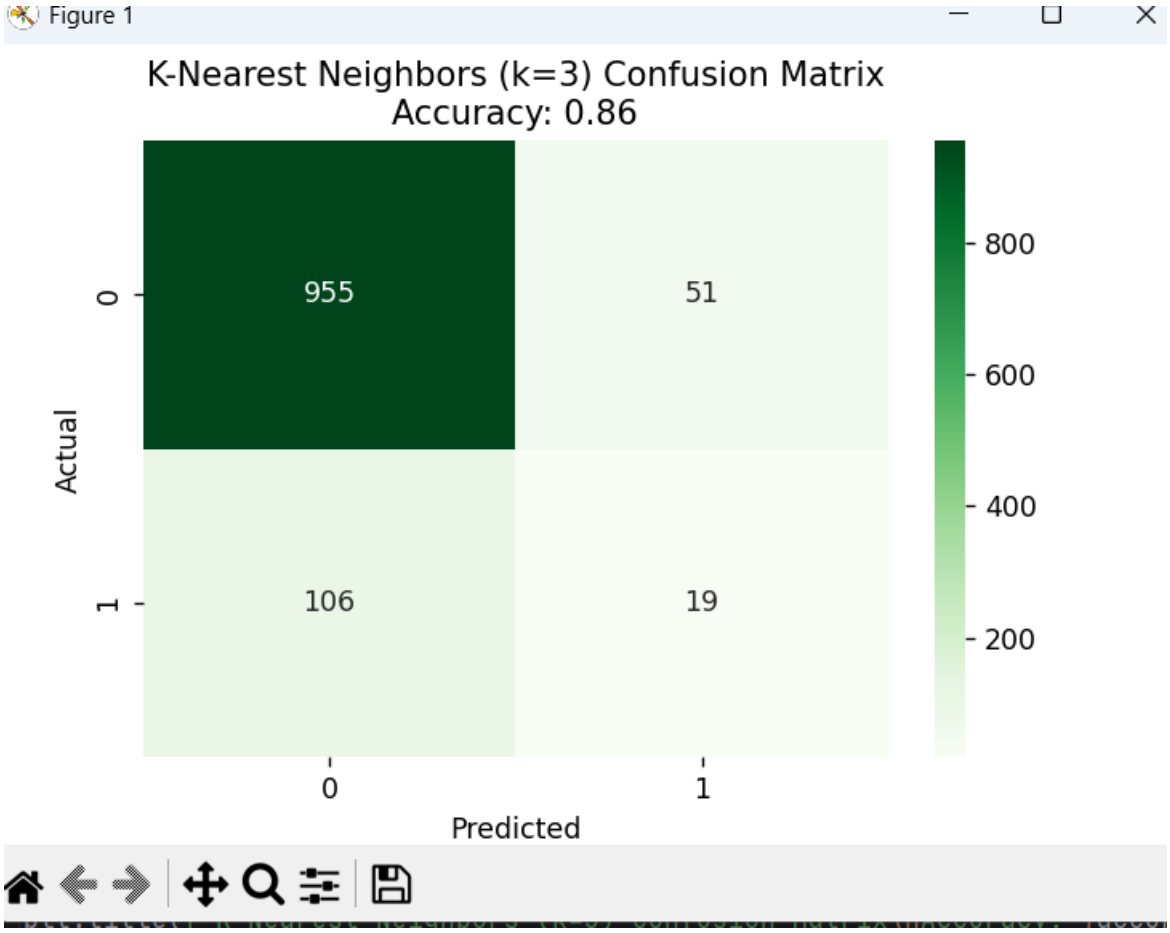
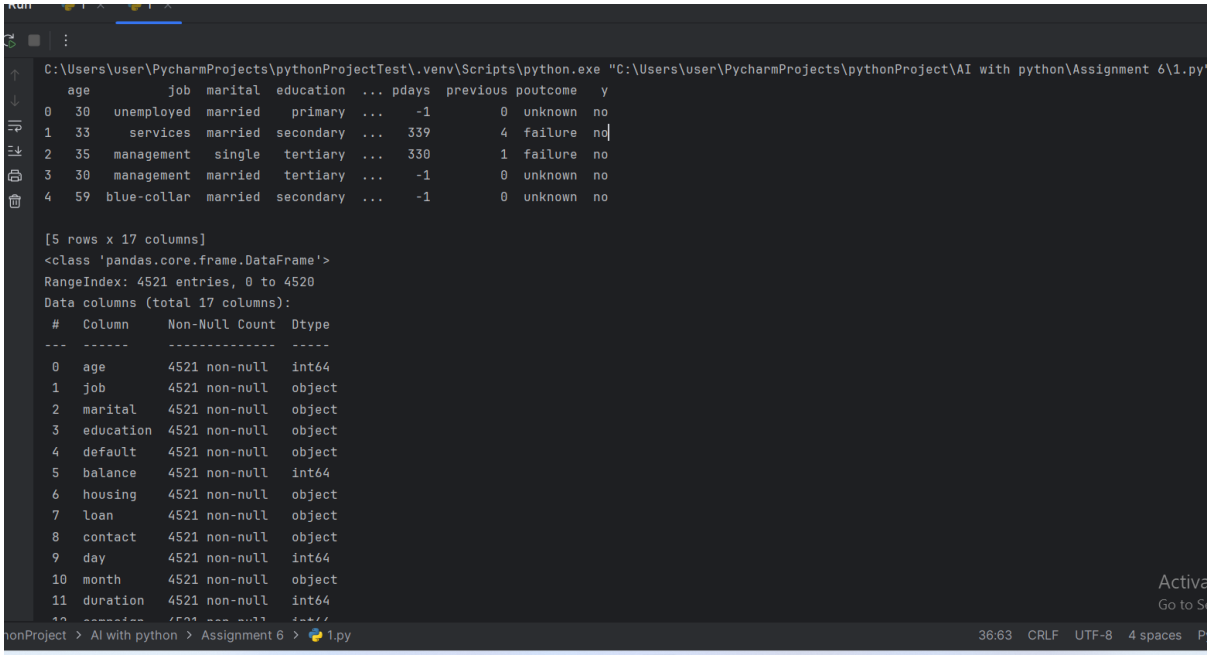


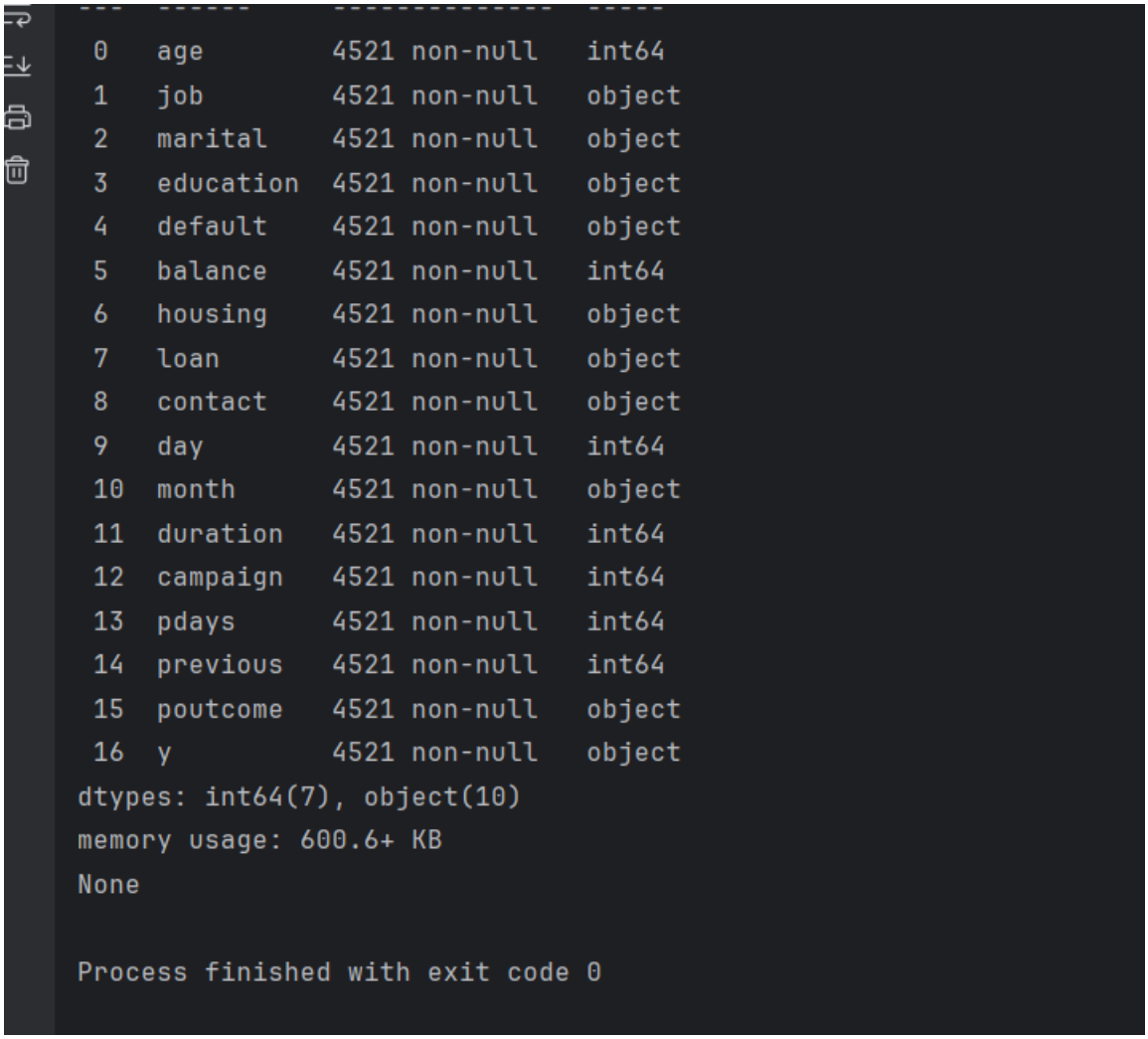
Figure 1
K-Nearest Neighbors (k=3) Confusion Matrix
Accuracy: 0.86
955
51
106
19
0
1
Actual
0
1
Predicted
800
600
400
200



```
C:\Users\user\PycharmProjects\pythonProjectTest\.venv\Scripts\python.exe "C:\Users\user\PycharmProjects\pythonProject\AI with python\Assignment 6\1.py
   age          job  marital  education  ... pdays  previous poutcome   y
0   30   unemployed  married    primary  ...    -1         0  unknown  no
1   33     services  married  secondary  ...   339         4  failure  no
2   35   management   single   tertiary  ...   330         1  failure  no
3   30   management  married   tertiary  ...    -1         0  unknown  no
4   59  blue-collar  married  secondary  ...    -1         0  unknown  no

[5 rows x 17 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4521 entries, 0 to 4520
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   age        4521 non-null   int64
 1   job        4521 non-null   object
 2   marital    4521 non-null   object
 3   education  4521 non-null   object
 4   default    4521 non-null   object
 5   balance    4521 non-null   int64
 6   housing    4521 non-null   object
 7   loan       4521 non-null   object
 8   contact    4521 non-null   object
 9   day        4521 non-null   int64
 10  month      4521 non-null   object
 11  duration   4521 non-null   int64
```

onProject > AI with python > Assignment 6 > 1.py 36:63 CRLF UTF-8 4 spaces



```
 0   age        4521 non-null   int64
 1   job        4521 non-null   object
 2   marital    4521 non-null   object
 3   education  4521 non-null   object
 4   default    4521 non-null   object
 5   balance    4521 non-null   int64
 6   housing    4521 non-null   object
 7   loan       4521 non-null   object
 8   contact    4521 non-null   object
 9   day        4521 non-null   int64
 10  month      4521 non-null   object
 11  duration   4521 non-null   int64
 12  campaign   4521 non-null   int64
 13  pdays      4521 non-null   int64
 14  previous   4521 non-null   int64
 15  poutcome   4521 non-null   object
 16  y          4521 non-null   object
dtypes: int64(7), object(10)
memory usage: 600.6+ KB
None

Process finished with exit code 0
```