FINLATICS DATA SCIENCE

CASE PROJECT

- Name: Ullas G

TOPIC: Banking

Problem Statement:

The goal of this data science project is to **analyze and predict client behavior** in a banking dataset related to direct marketing campaigns. Specifically**, the objective is to explore correlations and patterns among various client attributes** and determine the likelihood of clients subscribing to a term deposit.

Libraries Used:

1) Matplotlib – Version: 3.9.0

2) Seaborn – Version: 0.13.2

3) Pandas - Version: 2.2.2

4) Numpy – Version: 1.26.4

Questions:

1) What is the distribution of age among the clients?

   Solution:

q1.py > ...

```python
#Q1--   What is the distribution of age among the clients?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df = pd.read_csv('banking_data.csv')
fig,ax = plt.subplots(ncols=2,figsize=(12,6))
#Histogram of the given age distribution
sns.histplot(df['age'],bins = 25,kde = True,ax = ax[0])
ax[0].set_title('Age Distribution of Clients')
ax[0].set_xlabel('Age')
ax[0].set_ylabel('Frequency')

#Box plot for the age of clients to learn about the measures of central tendancy
min_age = df['age'].min()
q1_age = df['age'].quantile(0.25)
median_age = df['age'].median()
q3_age = df['age'].quantile(0.75)
max_age = df['age'].max()

sns.boxplot(x=df['age'],ax=ax[1])
ax[1].annotate(f'Min: {min_age}', xy=(min_age, 0.5
                arrowprops=dict(facecolor='black', arrowstyle="->"))
ax[1].annotate(f'Q1: {q1_age}', xy=(q1_age, 0.5), xytext=(q1_age - 10, 0.6),
                arrowprops=dict(facecolor='black', arrowstyle="->"))
ax[1].annotate(f'Median: {median_age}', xy=(median_age, 0.5), xytext=(median_age - 10, 0.4),
                arrowprops=dict(facecolor='black', arrowstyle="->"))
ax[1].annotate(f'Q3: {q3_age}', xy=(q3_age, 0.5), xytext=(q3_age - 10, 0.6),
                arrowprops=dict(facecolor='black', arrowstyle="->"))
ax[1].annotate(f'Max: {max_age}', xy=(max_age, 0.5), xytext=(max_age - 10, 0.5),
                arrowprops=dict(facecolor='black', arrowstyle="->"))
ax[1].set_title('Box plot for age of client')
ax[1].set_xlabel('Age')
plt.tight_layout()
plt.show()
```

(function) arrowstyle: str

2) How does the job type vary among the clients?

Solution:

q2.py > ...

```python
#Q2-How does the job type vary among the clients?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df = pd.read_csv('banking_data.csv')
fig,ax = plt.subplots(ncols=2,figsize = (12,6))

#print(df.shape[0])
#print(df[df['job'].notnull()].shape[0])  , no missing values

#Count plot for distribution of clients among different jobs
sns.countplot(y='job', data=df, order=df['job'].value_counts().index, ax=ax[0])
ax[0].set_title('Distribution of Job Types Among Clients')
ax[0].set_xlabel('Count')
ax[0].set_ylabel('Job Type')

# Pie chart of job type distribution
df['job'].value_counts().plot.pie(autopct='%1.1f%%', startangle=140, ax=ax[1])
ax[1].set_title('Proportion of Job Types Among Clients')
ax[1].set_ylabel('')
plt.show()
```

3) What is the marital status distribution of the clients?

Solution:

q3.py > ...

```python
#Q3- What is the marital status distribution of the clients?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df = pd.read_csv('banking_data.csv')

#print(df.shape[0])
#print(df[df['marital'].notnull()].shape[0])
#only 3 missing values out of 45216 hence no need to worry about null values

fig,ax = plt.subplots(ncols = 2,figsize=(12,6))
#Count plot for marital status distribution
sns.countplot(y='marital',data=df,order=df['marital'].value_counts().index,ax = ax[0])
ax[0].set_title('Distribution of Marital Status Among Clients')
ax[0].set_xlabel('Count')
ax[0].set_ylabel('Marital Status')

#Pie chart of marital status distribution
df['marital'].value_counts().plot.pie(autopct='%1.1f%%', startangle=140, ax=ax[1])
ax[1].set_title('Proportion of Marital Status Among Clients')
ax[1].set_ylabel('')
plt.show()
```

4) What is the level of education among the clients?

Solution:

q4.py > ...

```python
#Q4 - - What is the level of education among the clients?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df=pd.read_csv('banking_data.csv')
#print(df.shape[0])
#print(df[df['education'].notnull()].shape[0])
#Only 3 missing values, hence no need to worry about the null values
fig, ax = plt.subplots(ncols=2, figsize=(12,6))

# Count plot of education level distribution
#The plot also depicts the number of clients who have credit in each educational category
sns.countplot(x='education', data=df, order=df['education'].value_counts().index, ax=ax[0],hue=df['default'])
ax[0].set_title('Distribution of Education Levels Among Clients')
ax[0].set_xlabel('Education Level')
ax[0].set_ylabel('Count')
ax[0].tick_params(axis='x', rotation=45)
handles, labels = ax[0].get_legend_handles_labels()
ax[0].legend(handles, labels, title='Credit')#Changing the title of the legend

# Pie chart of education level distribution
df['education'].value_counts().plot.pie(autopct='%1.1f%%', startangle=140, ax=ax[1])
ax[1].set_title('Proportion of Education Levels Among Clients')
ax[1].set_ylabel('')
plt.show()
```

5) What proportion of clients have credit in default?

Solution:

q5.py > ...

```python
#Q5-What proportion of clients have credit in default?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df=pd.read_csv('banking_data.csv')

#calculating the proportion of clients who have credit
proportion = df['default'].value_counts(normalize=True) * 100
print(proportion)

#Pie chart depicting proportion of clients who have credit
fig,ax = plt.subplots(ncols=2,figsize=(12,6))
proportion.plot(kind='pie', autopct='%1.1f%%', startangle=140,  ax=ax[0])
ax[0].set_title('Proportion of Clients with Credit in Default')
ax[0].set_ylabel('')


#View of Age group vs count of clients who have credit
bins = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

# Grouping by age and default to get counts
age_default_counts = df.groupby(pd.cut(df['age'], bins=bins))['default'].value_counts().unstack(fill_value=0)

age_default_counts.plot(kind='bar', ax=ax[1])
ax[1].set_title('Distribution of Credit  Status by Age Groups')
ax[1].set_xlabel('Age Groups')
ax[1].set_ylabel('Count')
ax[1].legend(title='Credit', labels=['No', 'Yes'])
ax[1].tick_params(axis='x', rotation=45)
plt.show()
```

6) What is the distribution of average yearly balance among the clients?

Solution:

q6.py > ...

```python
#Q6-What is the distribution of average yearly balance among the clients?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df=pd.read_csv('banking_data.csv')
#print(df.shape[0])
#print(df[df['balance'].notnull()].shape[0])  ,-no missing values

# Plotting the distribution of average yearly balance
plt.figure(1,figsize=(10, 6))
sns.histplot(df['balance'], bins=40,kde=True)
plt.title('Distribution of Average Yearly Balance Among Clients')
plt.xlabel('Average Yearly Balance (euros)')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```

7) How many clients have housing loans?

Solution:

q7.py > ...

```python
#Q7-How many clients have housing loans?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df=pd.read_csv('banking_data.csv')
#print(df.shape[0])
#print(df[df['housing'].notnull()].shape[0])  -no missing values

#printing the number of clients who have housing loans
housing_loans = df[df['housing'] == 'yes'].shape[0]
housing_counts = df['housing'].value_counts()
print(f'Number of clients with housing loans:{housing_loans}')

fig,ax = plt.subplots(ncols = 2,figsize=(12,6))
#pie chart representing clients with an without housing loans
ax[0].pie(housing_counts, labels=housing_counts.index, autopct='%1.1f%%',colors=['skyblue', 'lightcoral'])
ax[0].set_title('Proportion of Clients with and without Housing Loans')
ax[0].set_ylabel('')


ton_project_learnings\q6.py • Untracked  g clients with and without housing loans
housing_counts.plot(kind='bar')
ax[1].set_title('Number of Clients with and without Housing Loans')
ax[1].set_xlabel('Housing Loan')
ax[1].set_ylabel('Count')
plt.show()
```

8) How many clients have personal loans?

Solution:

q8.py > ...

```python
#Q8-How many clients have personal loans?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df=pd.read_csv('banking_data.csv')

#printing the number of clients who have personal loans
personal_loans = df[df['loan'] == 'yes'].shape[0]
personal_counts = df['loan'].value_counts()
print(f'Number of clients with personal loans:{personal_loans}')

fig,ax = plt.subplots(ncols = 2,figsize=(12,6))
#pie chart representing clients with an without personal loans
ax[0].pie(personal_counts, labels=personal_counts.index, autopct='%1.1f%%',colors=['skyblue', 'lightcoral'])
ax[0].set_title('Proportion of Clients with and without Personal Loans')
ax[0].set_ylabel('')


#Histogram representing clients with and without personal loans
personal_counts.plot(kind='bar')
ax[1].set_title('Number of Clients with and without Personal Loans')
ax[1].set_xlabel('Personal Loan')
```

ton_project_learnings\q7.py • Untracked

9) What are the communication types used for contacting clients during the campaign?

Solution:

```python
#Q9-What are the communication types used for contacting clients during the campaign

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df=pd.read_csv('banking_data.csv')
#print(df.shape[0])
#print(df[df['contact'].notnull()].shape[0])   , - no missing values

types_of_contact = df['contact'].value_counts()
print('The different Types of Communication used to contact the Client are:')
for i in types_of_contact.index:
    if(i=='unknown'):
        continue
    print(i)
print('The communication type for some of the clients are unknown')
#Printing the types of communication used


#Bar plot representing the types of communication used by the Bank to contact the Clients
fig = plt.figure(figsize=(12,6))
types_of_contact.plot(kind='bar',color ='skyblue',title='Types Of Communication Used By the Bank')
for i, count in enumerate(types_of_contact):
    plt.text(i, count + 100, str(count), ha='center', va='bottom', fontsize=10)
plt.xlabel('Communication Type')
plt.ylabel('Count')
arnings\q8.py • Untracked
ut()
plt.show()
```

10) What is the distribution of the last contact day of the month?

Solution:

```python
#Q10-What is the distribution of the last contact day of the month?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df=pd.read_csv('banking_data.csv')

# Plotting the distribution of last contact day of the month
plt.figure(figsize=(10, 6))
sns.histplot(df['day'], bins=31, kde=False)
plt.title('Distribution of Last Contact Day of the Month')
plt.xlabel('Day of the Month')
plt.ylabel('Frequency')


df['month'] = pd.Categorical(df['month'], categories=['jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'sep', 'oct', 'nov', 'dec

#HEAT MAP
#To represent the distribution of last contact day of the month for each month
pivot = df.pivot_table(index='day', columns='month', values='day_month', aggfunc='count')
plt.figure(figsize=(10,6))
sns.heatmap(pivot, cmap='YlGnBu', annot=True, fmt='g', linewidths=.5)
plt.title('Distribution of Last Contact Day of the Month')
plt.xlabel('Month')
plt.ylabel('Last Contact Day of the Month')
plt.tight_layout()
plt.show()
```

11) How does the last contact month vary among the clients?

Solution:

q11.py > ...

```python
#Q11-How does the last contact month vary among the clients?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df=pd.read_csv('banking_data.csv')
last_contact_month = df['month'].value_counts()
#PIE CHART
plt.pie(last_contact_month.values,labels=last_contact_month.index,autopct='%1.1f%%')
plt.title('Proportion of last contact month of Clients')
plt.ylabel('')

#COUNT PLOT WITH PREVIOUS OUTCOME
g = sns.catplot(x='month', col='poutcome', data=df, kind='count', palette='viridis')
g.set_xlabels('Month')
g.set_ylabels('Number of Clients')
g.set_titles('previous outcome: {col_name}')
g.fig.suptitle('Distribution Of Last Contact Month Of Clients with Previous Outcome', y=1)

for ax in g.axes.flat:
    for p in ax.patches:
        height = p.get_height()
        ax.text(p.get_x() + p.get_width() / 2., height + 50, f'{int(height)}', ha='center', va='bottom', fontsize=10)
#COUNT PLOT WITH JOB TYPE
l = sns.catplot(x='month', col='job', data=df, kind='count', palette='viridis', col_wrap=3,sharex=True)
l.set_xlabels('Month')
l.set_ylabels('Number of Clients')
l.set_titles('Job: {col_name}',y=0.90)
l.fig.suptitle('Distribution Of Last Contact Month Of Clients with Job Type', y=1)

for ax in l.axes.flat:
    for p in ax.patches:
        height = p.get_height()
        ax.text(p.get_x() + p.get_width() / 2., height + 50, f'{int(height)}', ha='center', va='bottom', fontsize=10)
plt.tight_layout()
plt.show()
```

12) What is the distribution of the duration of the last contact?

Solution:

q12.py > ...

```python
#Q12-What is the distribution of the duration of the last contact?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

df=pd.read_csv('banking_data.csv')
sns.histplot(df['duration'], bins=30, kde=True, color='skyblue')
plt.title('Distribution of Duration of Last Contact')
plt.xlabel('Duration (seconds)')
plt.ylabel('Frequency')
plt.xlim(right=2000)
plt.grid()
plt.show()
```

13) How many contacts were performed during the campaign for each client?

Solution:

q13.py > ...

```python
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd


df=pd.read_csv('banking_data.csv')
print('Summary Statistics for Camapign:')
print(df['campaign'].describe())

#Handling the missing values by filling them up with median
df['campaign'].fillna(df['campaign'].median(), inplace=True)

#Hist plot to display the the distribution of number of contacts made in the campaign
plt.figure(figsize=(10,6))
sns.histplot(df['campaign'], bins=30, kde=False, color='skyblue')
plt.title('Distribution of Number of Contacts Performed During Campaign')
plt.xlabel('Number of Contacts')
plt.ylabel('Frequency')
plt.grid(True)
plt.xlim(right=20)


#Box plot to Observe the statistics of the number of contacts made
plt.figure(figsize=(10,6))
sns.boxplot(y='campaign',data=df,palette='viridis')
plt.title('Number of Contacts During Campaign')
plt.ylabel('Number of Contacts')
plt.ylim()

median = df['campaign'].median()
min_val=df['campaign'].min()
max_val=df['campaign'].max()
plt.text(0, median, f'Median: {median}', ha='center', va='bottom', color='black', fontsize=10)
plt.text(0, min_val, f'Min: {min_val}', ha='center', va='bottom', color='black', fontsize=10)
plt.text(0, max_val, f'Max: {max_val}', ha='center', va='bottom', color='black', fontsize=10)
plt.show()

```

14) What is the distribution of the number of days passed since the client was last contacted from a previous campaign?

Solution:

q14.py > ...

```python
#Q14-What is the distribution of the number of days passed since the client was last contacted from a previous campaign?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd


df=pd.read_csv('banking_data.csv')
df['pdays'] = df['pdays'].replace(-1, pd.NA)
plt.figure(figsize=(10,6))
sns.histplot(df['pdays'],bins=10,color='skyblue')
plt.title('Distribution of Number of Days Since Last Contact from Previous Campaign')
plt.xlabel('Number of Days')
plt.ylabel('Frequency')
plt.grid()


#Distribution Of number of days since client was previously contacted vs previous outcome with client
plt.figure(figsize=(10, 6))
sns.barplot(x='poutcome', y='pdays', data=df, estimator='mean', palette='viridis')
plt.title('Mean Number of Days Since Last Contact for Each Previous Outcome')
plt.xlabel('Previous Outcome')
plt.ylabel('Mean Number of Days')
plt.show()
```

15) How many contacts were performed before the current campaign for each client?

Solution:

q15.py > ...

```python
#Q15-How many contacts were performed before the current campaign for each client?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd


df=pd.read_csv('banking_data.csv')
df.fillna({'previous':0})
num_of_contacts = df['previous'].sum()
print(f'The total number of contacts made to each client before the current campaign is:{num_of_contacts}')
```

16) What were the outcomes of the previous marketing campaigns?

Solution:

q16.py > ...

```python
#Q16-What were the outcomes of the previous marketing campaigns?

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd


df=pd.read_csv('banking_data.csv')
df.fillna({'poutcome':'unknown'})
prev_out = df['poutcome'].value_counts()

print('Summary of Outcomes of previous marketing campaigns')
print(f'Total number of Reported Success:{df[df['poutcome']=='success'].shape[0]}')
print(f'Total number of Reported Failure:{df[df['poutcome']=='failure'].shape[0]}')
print(f'Other Results:{df[df['poutcome']=='other'].shape[0]}')
print(f'Unknown:{df[df['poutcome']=='unknown'].shape[0]}')


#Proportion of Each Previous Outcome
plt.figure(figsize=(10,6))
plt.pie(prev_out.values,labels=prev_out.index,autopct='%1.1f%%',colors=['lightgreen', 'lightcoral', 'lightskyblue', 'lightgrey'])
plt.title('Distribution of Outcomes from Previous Marketing Campaigns')
plt.ylabel('')
plt.show()
```

17) What is the distribution of clients who subscribed to a term deposit vs. those who did not?

Solution:

q17.py > ...

```python
#Q17-What is the distribution of clients who subscribed to a term deposit vs. those who did not?

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv('banking_data.csv')
subscription_counts = df['y'].value_counts(normalize=True) * 100
print(f"Percentage subscribed to term deposit: {subscription_counts['yes']:.1f}%")
print(f"Percentage not subscribed to term deposit: {subscription_counts['no']:.1f}%")

#Distribution of clients subscribing to term deposit
plt.figure(figsize=(10, 6))
sns.countplot(x='y', data=df, palette=['lightcoral', 'lightskyblue'])
plt.title('Distribution of Clients Subscribing to Term Deposit')
plt.xlabel('Term Deposit Subscription')
plt.ylabel('Number of Clients')


for p in plt.gca().patches:
    plt.gca().annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                       ha='center', va='center', xytext=(0, 9), textcoords='offset points', fontsize=12, color='black')
plt.legend(labels=['Not Subscribed', 'Subscribed'], loc='upper right')
plt.grid()
plt.show()
```

18) Are there any correlations between different attributes and the likelihood of subscribing to a term deposit?

Solution:

q18.py > ...

```python

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_csv('banking_data.csv')
numerical_columns = ['age', 'balance', 'duration', 'campaign', 'pdays', 'previous']
categorical_columns = ['job', 'education', 'marital','poutcome']

#Converting some columns to numerical type for correlation
df['y_numerical'] = df['y'].apply(lambda x: 1 if x == 'yes' else 0)
df['Housing_Loan'] = df['housing'].apply(lambda x: 1 if x == 'yes' else 0)
df['Personal_Loan'] = df['loan'].apply(lambda x: 1 if x == 'yes' else 0)
numerical_data = df[numerical_columns +['Housing_Loan']+['Personal_Loan']+['y_numerical']]

# Correlation Matrix for Numerical Variables
corr_matrix = numerical_data.corr()
plt.figure(figsize=(12, 6))
sns.heatmap(corr_matrix, annot=True, cmap='viridis', fmt=".2f", annot_kws={"size": 10})
plt.title('Correlation Matrix of Numerical Variables')

#Contingency Tables for Categorical Variables
for col in categorical_columns:
    contingency_table = pd.crosstab(df[col], df['y'], margins=True, margins_name='Total')
    print(f"Contingency Table for {col} vs y:")
    print(contingency_table)
    print("\n")

print('The above Heat map shows how the numerical variables are related with themselves and other variables')
print("Summary Of the logically drawn results of correlation Statistics of Numerical variables with the target variable 'y'")
print('Variable    Correlation_Coefficient')
variables_of_interest = ['Housing_Loan', 'Personal_Loan', 'age', 'balance']
for variable in variables_of_interest:
    correlation = corr_matrix.loc[variable, 'y_numerical']
    print(f"{variable}    {correlation:.2f}")
print("\n\nThe contigency table above shows how the value of 'y' depends on some of the categorical variables")
plt.tight_layout()
plt.show()
```