EXPLORATORY DATA ANALYSIS FOR BANK LOAN WORKSHOP

Agenda

1. **Data Reading and Cleaning**
2. **Data Analysis according to Deposit Status according Camping Results**
3. **Reporting and Recommendations**

Present by GROUP – 2

1) **Khin Yandar Hlaing**
2) **Thiri Whar Whar Lwin**

EXPLORATORY DATA ANALYSIS FOR BANK LOAN WORKSHOP

Data Analysis (EDA) for Banking Loane Workshop

The Bank Lone Project'purpose is applying whatever lerned Python for Data Secience

Step – 1) Install Matplotlib

[1]:
```
!pip install matplotlib
```

```
Requirement already satisfied: matplotlib in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (3.7.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from matplotlib) (1.0.5)
Requirement already satisfied: cycler>=0.10 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from matplotlib) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from matplotlib) (1.4.4)
Requirement already satisfied: numpy>=1.20 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from matplotlib) (1.24.3)
Requirement already satisfied: packaging>=20.0 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from matplotlib) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\thiri whar whar lwin\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

Step 2) Import pandas,numpy& matplotib.pyplot

[2]:
```
#Importing the libraries necessary for this project.
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

Dataset exploration

Step 3) Dataset Exploration

[3]:
```
import pandas as pd

# Read the CSV file with the correct delimiter
df = pd.read_csv('./bank-additional.csv', sep=';')

# Display the DataFrame
df
```

[3]:

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	campaign	pdays	previous	poutcome	emp.var.rat
0	30	blue-collar	married	basic.9y	no	yes	no	cellular	may	fri	...	2	999	0	nonexistent	-1
1	39	services	single	high.school	no	no	no	telephone	may	fri	...	4	999	0	nonexistent	1
2	25	services	married	high.school	no	yes	no	telephone	jun	wed	...	1	999	0	nonexistent	1
3	38	services	married	basic.9y	no	unknown	unknown	telephone	jun	fri	...	3	999	0	nonexistent	1
4	47	admin.	married	university.degree	no	yes	no	cellular	nov	mon	...	1	999	0	nonexistent	-0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4114	30	admin.	married	basic.6y	no	yes	yes	cellular	jul	thu	...	1	999	0	nonexistent	1
4115	39	admin.	married	high.school	no	yes	no	telephone	jul	fri	...	1	999	0	nonexistent	1
4116	27	student	single	high.school	no	no	no	cellular	may	mon	...	2	999	1	failure	-1
4117	58	admin.	married	high.school	no	no	no	cellular	aug	fri	...	1	999	0	nonexistent	1
4118	34	management	single	high.school	no	yes	no	cellular	nov	wed	...	1	999	0	nonexistent	-0

4119 rows × 21 columns

INPUT FEATURE (Each COLUMN Nmae & Describe):

1. `age` - client's age in years (numeric)
2. `job` - type of job (categorical: `admin.`, `blue-collar`, `entrepreneur`, `housemaid`, `management`, `retired`, `self-employed`, `services`, `student`, `technician`, `unemployed`, `unknown`)
3. `marital` - marital status (categorical: `divorced`, `married`, `single`, `unknown`)
4. `education` - client's education (categorical: `basic.4y`, `basic.6y`, `basic.9y`, `high.school`, `illiterate`, `professional.course`, `university.degree`, `unknown`)
5. `default` - has credit in default? (categorical: `no`, `yes`, `unknown`)
6. `housing` - has housing loan? (categorical: `no`, `yes`, `unknown`)
7. `loan` - has personal loan? (categorical: `no`, `yes`, `unknown`)
8. `contact` - contact communication type (categorical: `cellular`, `telephone`)
9. `month` - last contact month of the year (categorical: `jan`, `feb`, `mar`, ..., `nov`, `dec`)
10. `day_of_week` - last contact day of the week (categorical: `mon`, `tue`, `wed`, `thu`, `fri`)
11. `duration` - last contact duration, in seconds (numeric).
12. `campaign` - number of contacts performed and for this client during this campaign (numeric, includes the last contact)
13. `pdays` - number of days that have passed after the client was last contacted from the previous campaign (numeric; 999 means the client has not been previously contacted)
14. `previous` - number of contacts performed for this client before this campaign (numeric)
15. `poutcome` - outcome of the previous marketing campaign (categorical: `failure`, `nonexistent`, `success`)
16. `emp.var.rate` - employment variation rate, quarterly indicator (numeric)
17. `cons.price.idx` - consumer price index, monthly indicator (numeric)
18. `cons.conf.idx` - consumer confidence index, monthly indicator (numeric)
19. `euribor3m` - euribor 3 month rate, daily indicator (numeric)
20. `nr.employed` - number of employees, quarterly indicator (numeric)

OUTPUT FEATURE(Desired TARGET):

21. `y` - has the client subscribed a term deposit? (binary: `yes`, `no`)

1) Deposite Status by Age after Campaing(BOXPLOT)

```
plt.figure(figsize=(6, 6))
sns.boxplot(x='y', y='age', data=df,  width = 0.3, palette={'no': 'red', 'yes': 'green'})

plt.title('Distribution of Loan approval by Age using a box plot')
plt.xlabel('Deposite Status')
plt.ylabel('Age Group')

plt.show()
```

INSIGHT

Age between 30 to near 60, they might to plan campaigns to deposit.

Entire Client Outreach for Campaign Invitation

```
df.value_counts().sum()
```

```
4119
```

Total Deposited Clients

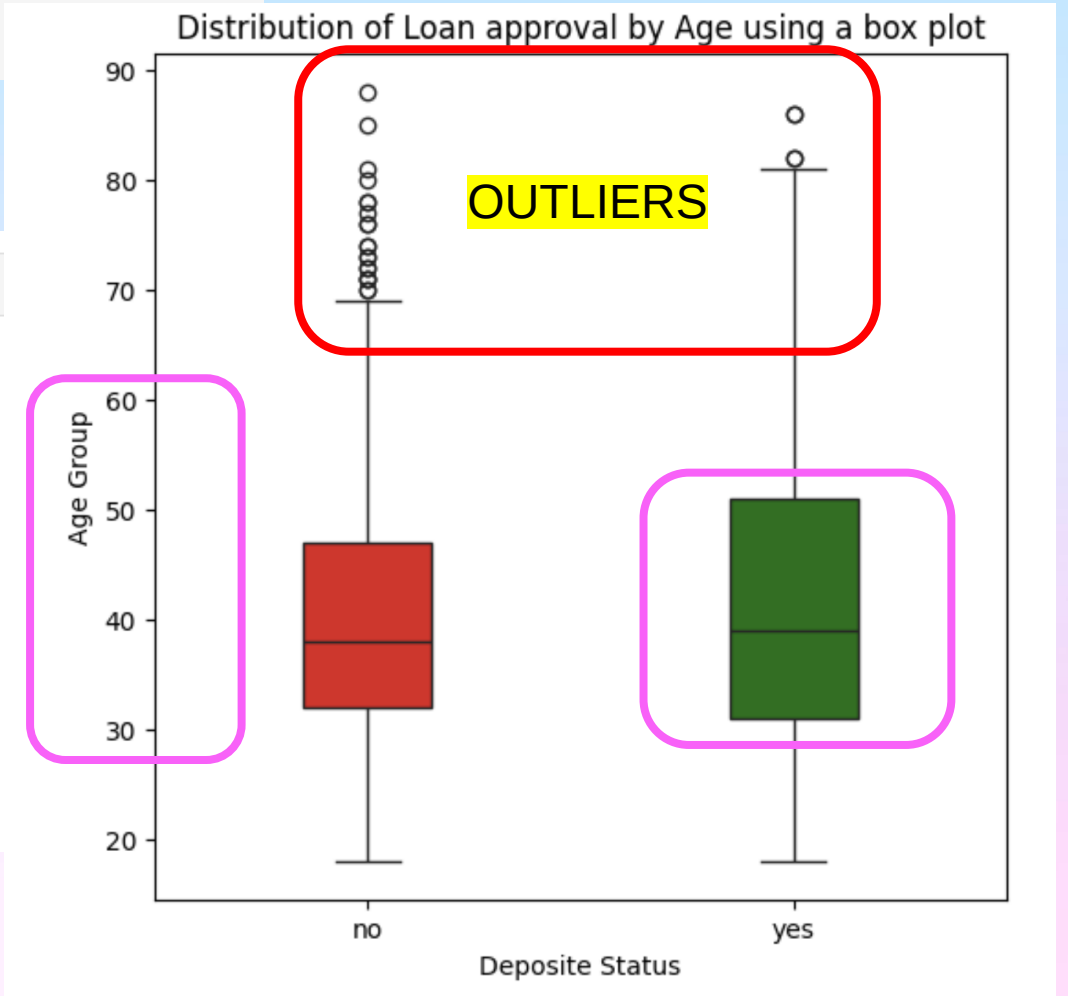
```
df['y'].value_counts()
```

```
y
no     3668
yes     451
Name: count, dtype: int64
```

```
df['age'].value_counts(normalize=True)*100
```

```
age
32    5.243991
31    4.637048
30    4.297160
34    4.224326
35    4.175771
        ...
69    0.048555
70    0.048555
85    0.024278
88    0.024278
19    0.024278
Name: proportion, Length: 67, dtype: float64
```

Deposited Client Age Group



2) Deposite Status by Job after Campaing(BOXPLOT)

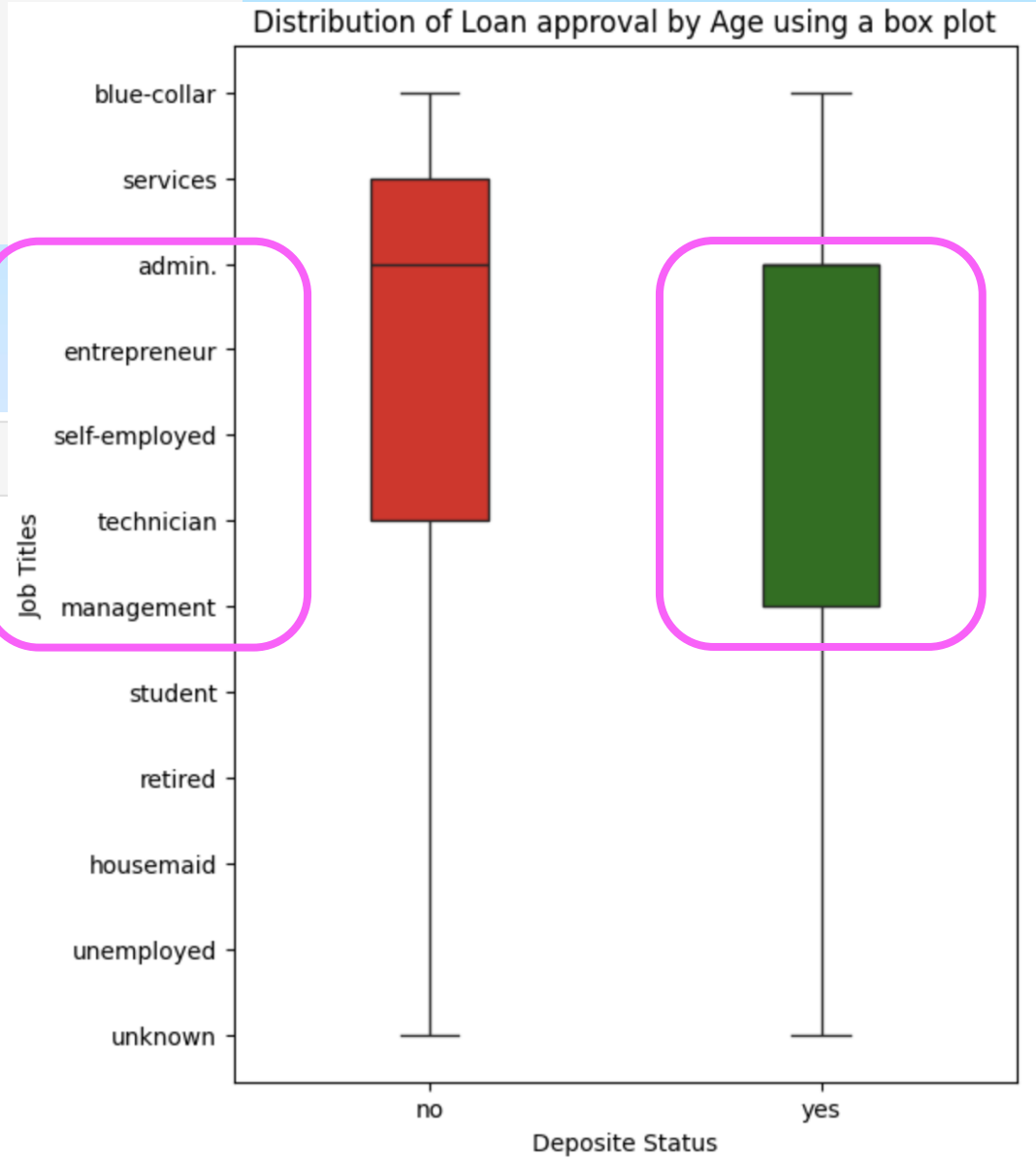
```
plt.figure(figsize=(6, 8))
sns.boxplot(x='y', y='job', data=df, width = 0.3, palette={'no': 'red', 'yes': 'green'})

plt.title('Distribution of Loan approval by Age using a box plot')
plt.xlabel('Deposite Status')
plt.ylabel('Job Titles')

plt.show()
```

INSIGHT

Admin, blue-collar, Technical, Services, Management job types mostly deposited that time.



Entire Client Outreach for Campaign Invitation

```
df.value_counts().sum()
```

```
4119
```

```
df['job'].value_counts(normalize=True)*100
```

```
job
admin.           24.569070
blue-collar      21.461520
technician       16.775916
services          9.541151
management        7.865987
retired           4.030104
self-employed     3.860160
entrepreneur      3.593105
unemployed        2.694829
housemaid         2.670551
student           1.990774
unknown           0.946832
Name: proportion, dtype: float64
```

3) Deposite Status by whoever housing and personal loan & Credit Card Payment on going (PIECHART)

```
# Aggregate the data for the pie chart
housing_counts = df['housing'].value_counts()
loan_counts = df['loan'].value_counts()
y_counts=df['y'].value_counts()
default_counts=df['default'].value_counts()

# Plot the pie chart
plt.figure(figsize=(12, 3))

# First pie chart: Housing
plt.subplot(1, 4, 1)
plt.pie(housing_counts, labels=housing_counts.index, autopct='%1.1f%%')
plt.title('Housing')

# Second pie chart: Loan
plt.subplot(1, 4, 2)
plt.pie(loan_counts, labels=loan_counts.index, autopct='%1.1f%%')
plt.title('Loan')

plt.subplot(1,4, 3)
plt.pie(y_counts, labels=y_counts.index, autopct='%1.1f%%')
plt.title('Y')

plt.subplot(1,4, 4)
plt.pie(default_counts, labels=default_counts.index, autopct='%1.1f%%')
plt.title('Default')
plt.suptitle('Distribution of Housing, Loan, Y and Default', fontsize=20)

plt.tight_layout()
plt.show()
```

```
df['housing'].value_counts()
```

```
housing
yes        2175
no         1839
unknown     105
Name: count, dtype: int64
```

```
df.value_counts().sum()
```

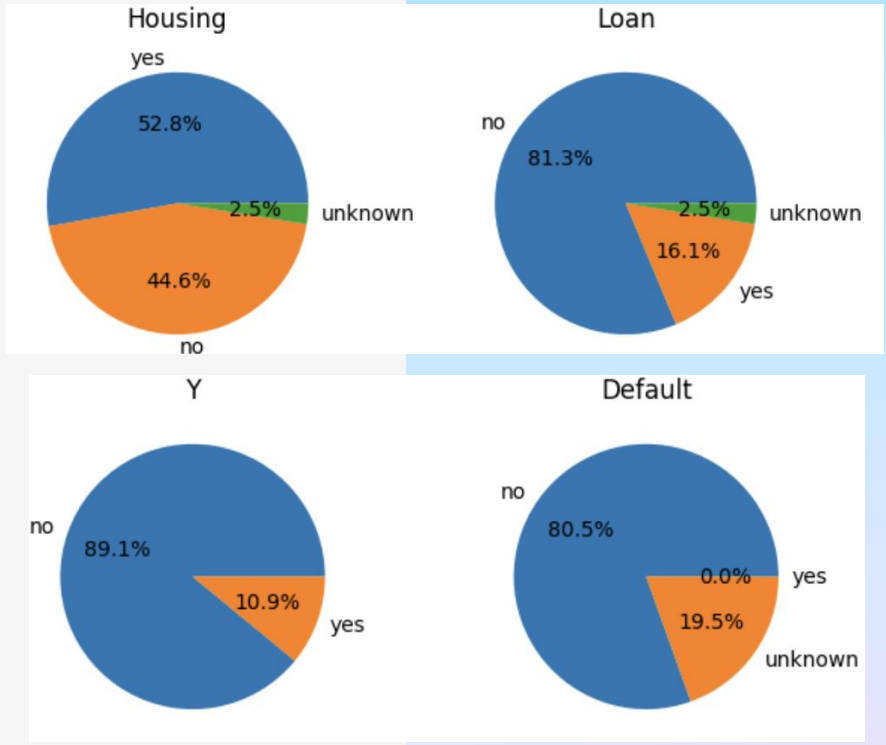
```
4119
```

```
df['loan'].value_counts()
```

```
loan
no         3349
yes         665
unknown     105
Name: count, dtype: int64
```

```
df['default'].value_counts()
```

```
default
no         3315
unknown     803
yes           1
Name: count, dtype: int64
```



INSIGHT

1) HOUSING _ Campaing Clients mostly had house loan.
2) PERSONAL LOAN _ Most of the Clients were belonging Personal Loan.
3) Y _ A term deposit status (potential client or not)
4) DEFAULT _ Most of the clients are using Credit card and on going payments.

4) Deposite Status by Previous and Recently Campaings (SCATTERPLOT)

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

# Use a color palette for bright colors
palette = sns.color_palette("bright")

plt.figure(figsize=(5, 8))
sns.scatterplot(data=df, x='previous', y='poutcome', hue='campaign', alpha=0.5, palette=palette)
plt.title('Deposit Status by Both Campaign Result')
plt.xlabel('Previous Campaing contact Times')
plt.ylabel('Deposite Status')
plt.show()
```

```python
df['poutcome'].value_counts(normalize=True)*100
```

```
poutcome
nonexistent    85.530469
failure        11.022093
success         3.447439
Name: proportion, dtype: float64
```

```python
df['previous'].value_counts(normalize=True)*100
```

```
previous
0    85.530469
1    11.531925
2     1.893664
3     0.606943
4     0.339888
5     0.048555
6     0.048555
Name: proportion, dtype: float64
```

```python
df['campaign'].value_counts(normalize=True)*100
```

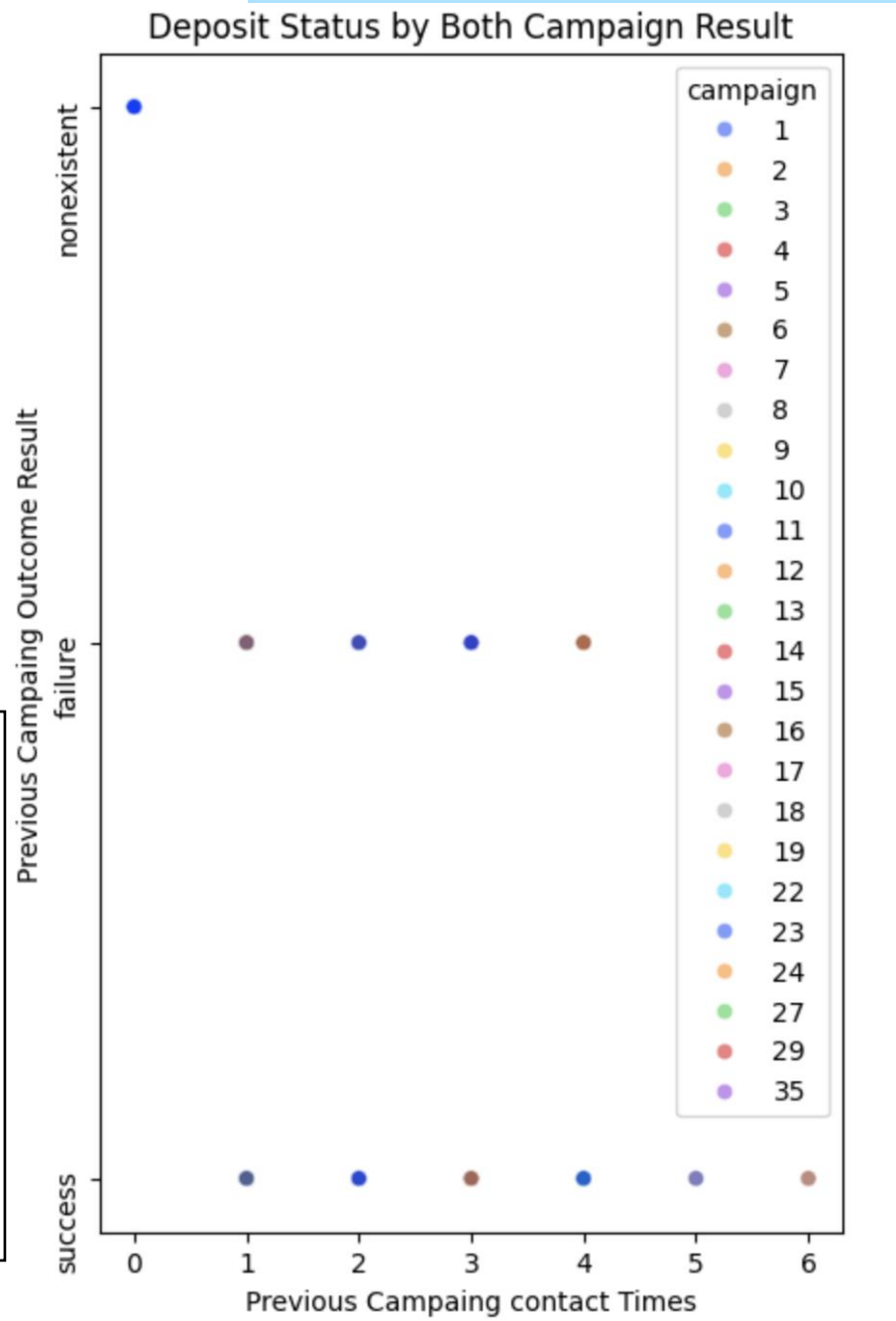
```
campaign
1     42.825929
2     25.224569
3     13.328478
4      7.064822
5      3.447439
6      2.403496
7      1.456664
8      0.873999
9      0.776888
10     0.485555
11     0.461277
12     0.388444
17     0.339888
13     0.267055
16     0.169944
14     0.145666
15     0.048555
22     0.048555
19     0.048555
23     0.048555
29     0.048555
27     0.024278
18     0.024278
24     0.024278
35     0.024278
Name: proportion, dtype: float64
```

INSIGHT

1)Clients with no previous campaign outcome are rare or have little data.

2)There is a noticeable clustering of failure outcomes with a small number of previous contacts, suggesting that a small number of interactions are not always sufficient for a successful outcome.

3)Success is possible across a wider range of previous contact times, implying that consistent engagement might eventually lead to a positive outcome, even if not immediately.



5) Deposite Status by Education Level and Marital Status (BARCHART))

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

# Create the bar plot
sns.barplot(x='marital', y='education', hue='y', data=df, width=0.5,palette={'no': 'red', 'yes': 'green'})

# Set title and labels
plt.title('Deposit Status by Education Level and Marital Status')
plt.xlabel('Marital Status')
plt.ylabel('Education Level')
plt.yticks(rotation=45)
# Display legend
plt.legend(title='Deposit Status')

plt.show()
```

```
df.value_counts().sum()
```

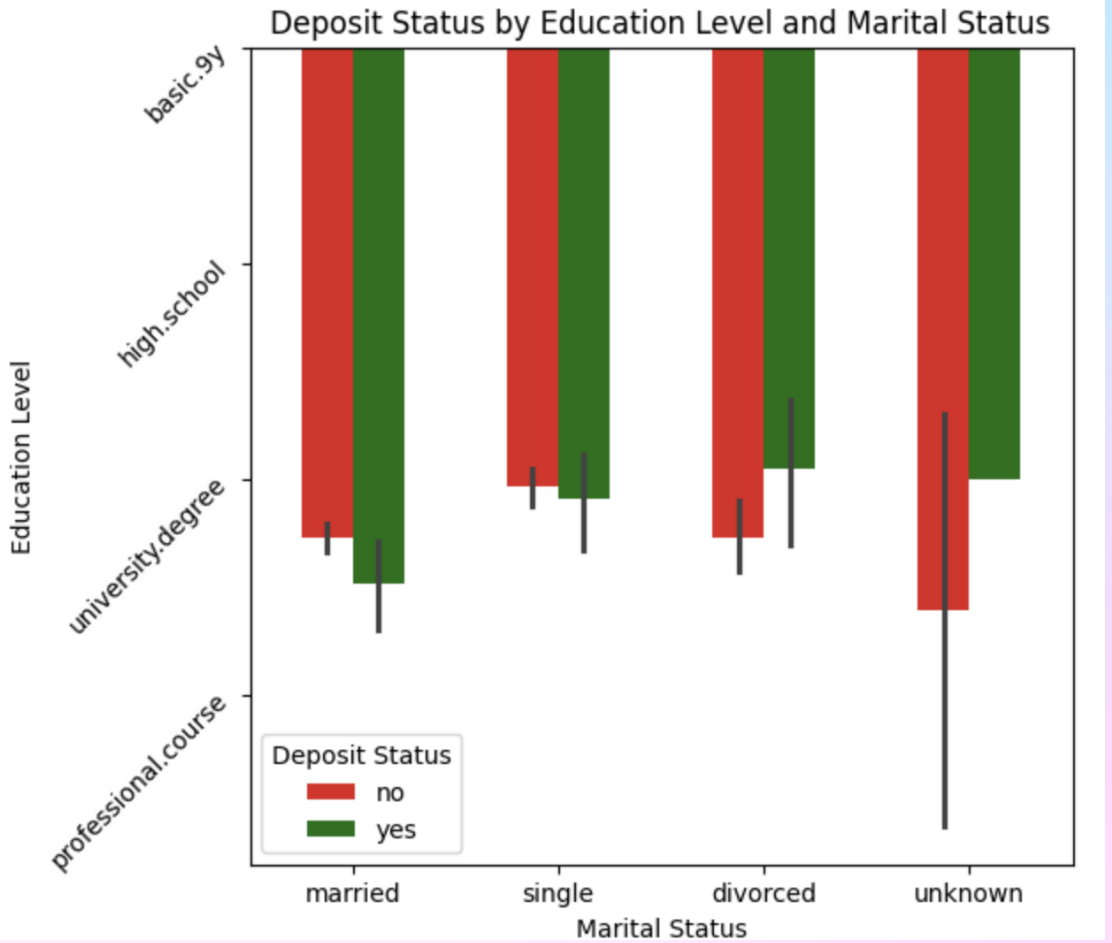
```
4119
```

```
df['education'].value_counts(normalize=True)*100
```

```
education
university.degree      30.687060
high.school            22.359796
basic.9y               13.935421
professional.course    12.988589
basic.4y               10.415149
basic.6y                5.535324
unknown                 4.054382
illiterate              0.024278
Name: proportion, dtype: float64
```

```
df['marital'].value_counts(normalize=True) *100
```

```
marital
married     60.912843
single      27.992231
divorced    10.827871
unknown      0.267055
Name: proportion, dtype: float64
```



INSIGHT

Impact of Marital Status:

Married individuals generally show a higher tendency not to make a deposit. Single individuals are more likely to make deposits if they have a basic.9y or high school education.

Impact of Education Level:

University degree holders and professional course attendees, irrespective of their marital status, tend to show higher non-deposit rates. Those with basic.9y or high school education have more varied outcomes based on their marital status.

Unknown Marital Status:

The mixed pattern for individuals with unknown marital status suggests that other factors might be influencing their deposit decisions.

6) Deposite Status by job and Credit Payments on going (SCATTERPLOT))

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(17 ,3))
sns.scatterplot(data=df, x='job', y='default', hue='y', alpha=0.5,palette={'no': 'red', 'yes': 'green'})
plt.title('Deposit Status by Age and Credit Payments')
plt.xlabel('Job Position')
plt.ylabel('Credit Payments')
```

```
df['default'].value_counts()
```

```
default
no         3315
unknown     803
yes           1
Name: count, dtype: int64
```

```
df['job'].value_counts(normalize=True)*100
```

```
job
admin.           24.569070
blue-collar      21.461520
technician       16.775916
services          9.541151
management        7.865987
retired           4.030104
self-employed     3.860160
entrepreneur      3.593105
unemployed        2.694829
housemaid         2.670551
student           1.990774
unknown           0.946832
Name: proportion, dtype: float64
```

INSIGHT

Impact of Credit Payments:

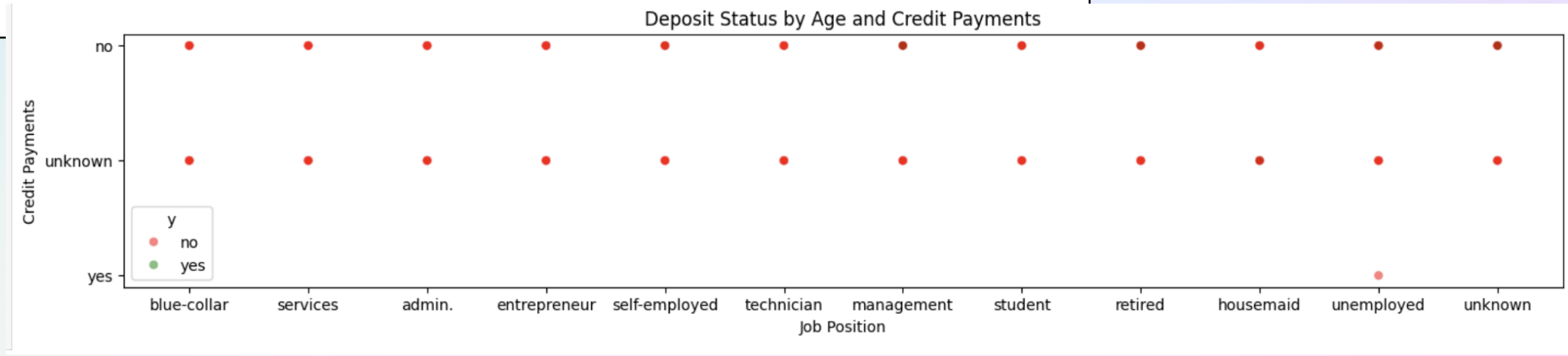
The absence of green data points in both the "no" and "yes" credit payment categories suggests that individuals with credit payments, regardless of the status, are not making deposits.

Job Position Influence:

Job position does not seem to have a significant influence on deposit status when considering credit payment status, as all job positions show similar patterns.

Focus on Unknown Credit Payment Status:

The "unknown" credit payment category should be investigated further to understand why these individuals are not making deposits.



7) Deposite Status by Contacting Month of week (LINEPLOT))

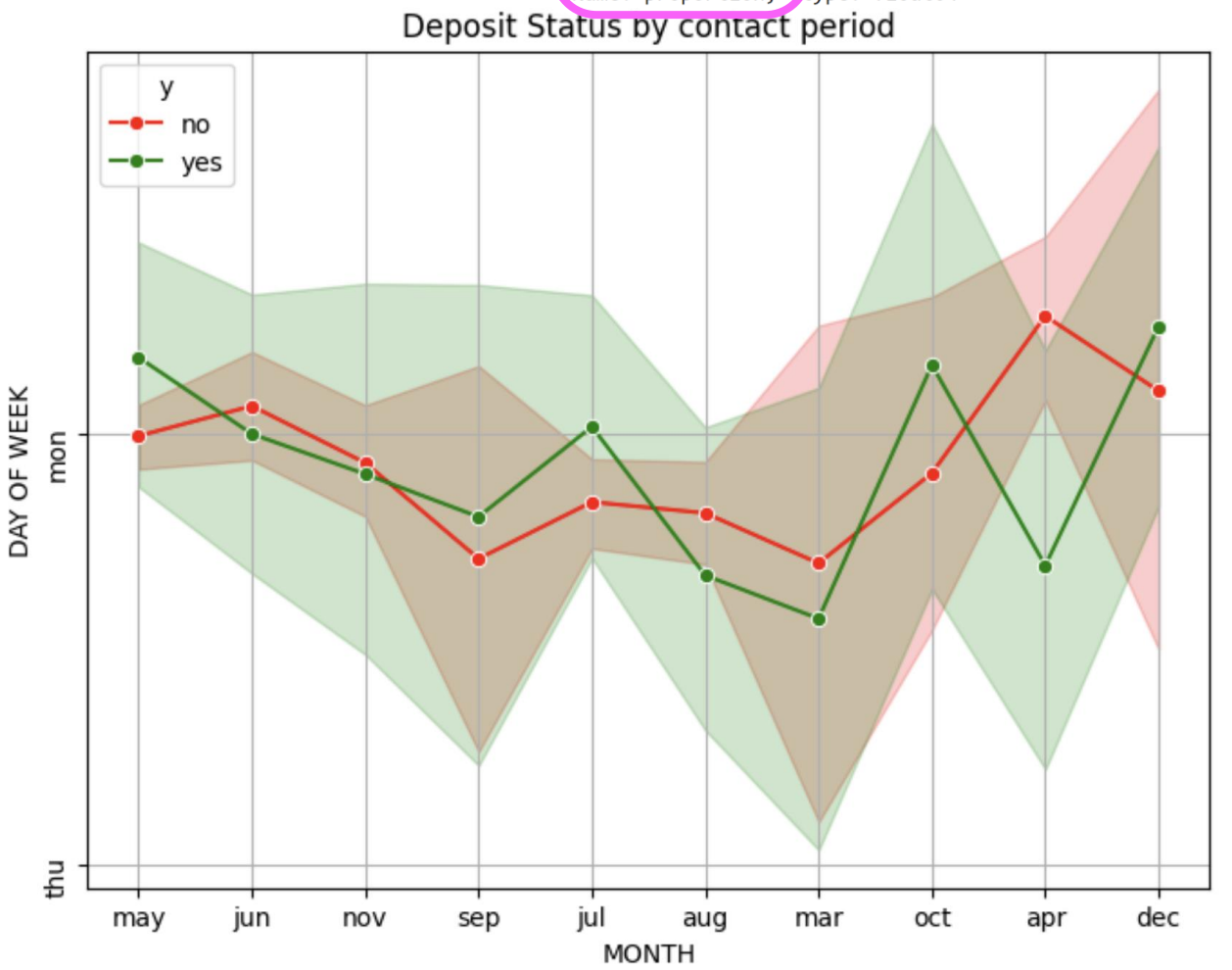
```
# Plot the line chart
plt.figure(figsize=(8, 6))
sns.lineplot(data=df, x='month', y='day_of_week',  hue='y', marker='o', palette={'no': 'red', 'yes': 'green'})
plt.xlabel('MONTH')
plt.ylabel('DAY OF WEEK')
plt.title('Deposit Status by contact period')
plt.yticks(rotation=90)
plt.grid(True)
plt.show()
```

```
df['day_of_week'].value_counts(normalize=True)*100
```

```
day_of_week
thu    20.878854
mon    20.757465
tue    20.417577
wed    19.300801
fri    18.645302
Name: proportion, dtype: float64
```

```
df['month'].value_counts(normalize=True)*100
```

```
month
may    33.454722
jul    17.261471
aug    15.440641
jun    12.867201
nov    10.827871
apr     5.219714
oct     1.675164
sep     1.553775
mar     1.165331
dec     0.534110
Name: proportion, dtype: float64
```



INSIGHT

Campaign Timing:

Focus on the months of May and October for campaigns, as these months show higher deposit rates.

Consider investigating why there is a decline in deposits from June to September and address potential issues.

Day of the Week Strategy:

Since the day of the week does not show a significant difference, campaigns can be uniformly distributed across the week.

Seasonal Analysis:

Further analysis could be done to understand the reasons behind the seasonal trends and optimize campaign strategies accordingly.

8) Deposite Status by Contacting Period (LINEPLOT))

```
# Plot the line chart
plt.figure(figsize=(8, 4))
sns.lineplot(data=df, x='contact', y='duration',  hue='y', marker='o', palette={'no': 'red', 'yes': 'green'})
plt.xlabel('CONTACT TYPE')
plt.ylabel('Contact duration show Second')
plt.title('Deposit Status by contact period')
plt.yticks(rotation=90)
plt.grid(True)
plt.show()
```

```
df['duration'].value_counts(normalize=True)*100
```

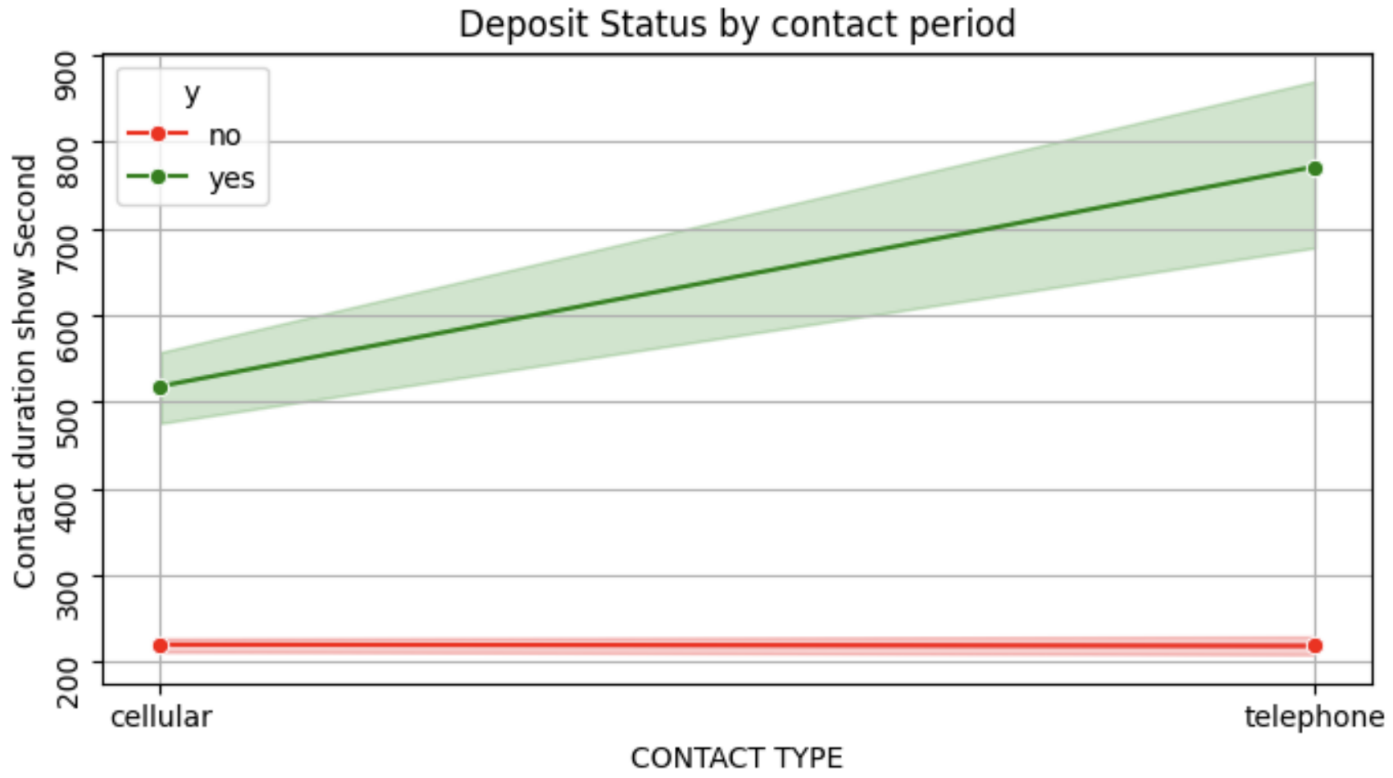
```
duration
77      0.582666
112     0.558388
73      0.534110
81      0.509832
122     0.485555
          ...
568     0.024278
776     0.024278
433     0.024278
440     0.024278
1386    0.024278
Name: proportion, Length: 828, dtype: float64
```

INSIGHT

The graph hints that longer telephone calls may lead to more successful deposits. However, further analysis would require additional context or data.

```
df['contact'].value_counts(normalize=True)*100
```

```
contact
cellular     64.384559
telephone    35.615441
Name: proportion, dtype: float64
```



9) Deposite Status by emp.var.rate, cons.price.idx, cons.conf.idx (HEATMAP))

```python
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Convert 'loan' column from 'yes'/'no' to 1/0
df['y'] = df['y'].apply(lambda x: 1 if x == 'yes' else 0)

# Selecting relevant columns for correlation analysis
columns_to_analyze = ["emp.var.rate", "cons.price.idx", "cons.conf.idx", "y"]
correlation_matrix = df[columns_to_analyze].corr()

# Plotting the heatmap
plt.figure(figsize=[8,6])
sns.heatmap(correlation_matrix, cmap="Reds", annot=True)

# Setting the title
plt.title("Deposit Status by emp.var.rate, cons.price.idx, cons.conf.idx, and loan")
plt.show()
```

```python
df['emp.var.rate'].value_counts(normalize=True)*100
```

```
emp.var.rate
 1.4    39.475601
-1.8    21.437242
 1.1    18.402525
-0.1     9.516873
-2.9     3.981549
-3.4     2.524885
-1.7     2.112163
-1.1     2.015052
-3.0     0.509832
-0.2     0.024278
Name: proportion, dtype: float64
```

```python
df['cons.conf.idx'].value_counts(normalize=True)*100
```

```
cons.conf.idx
-36.4    18.402525
-42.7    16.193251
-46.2    14.493809
-36.1    12.818645
-41.8    10.463705
-42.0     9.371207
-47.1     4.879825
-40.8     1.820830
-31.4     1.820830
-26.9     1.043943
-37.5     0.946832
-30.1     0.873999
-38.3     0.801165
-40.3     0.728332
-29.8     0.606943
-50.0     0.606943
-39.8     0.582666
-50.8     0.582666
-34.8     0.558388
-40.0     0.558388
-33.0     0.509832
-49.5     0.485555
-34.6     0.339888
-33.6     0.339888
-40.4     0.145666
-45.9     0.024278
Name: proportion, dtype: float64
```

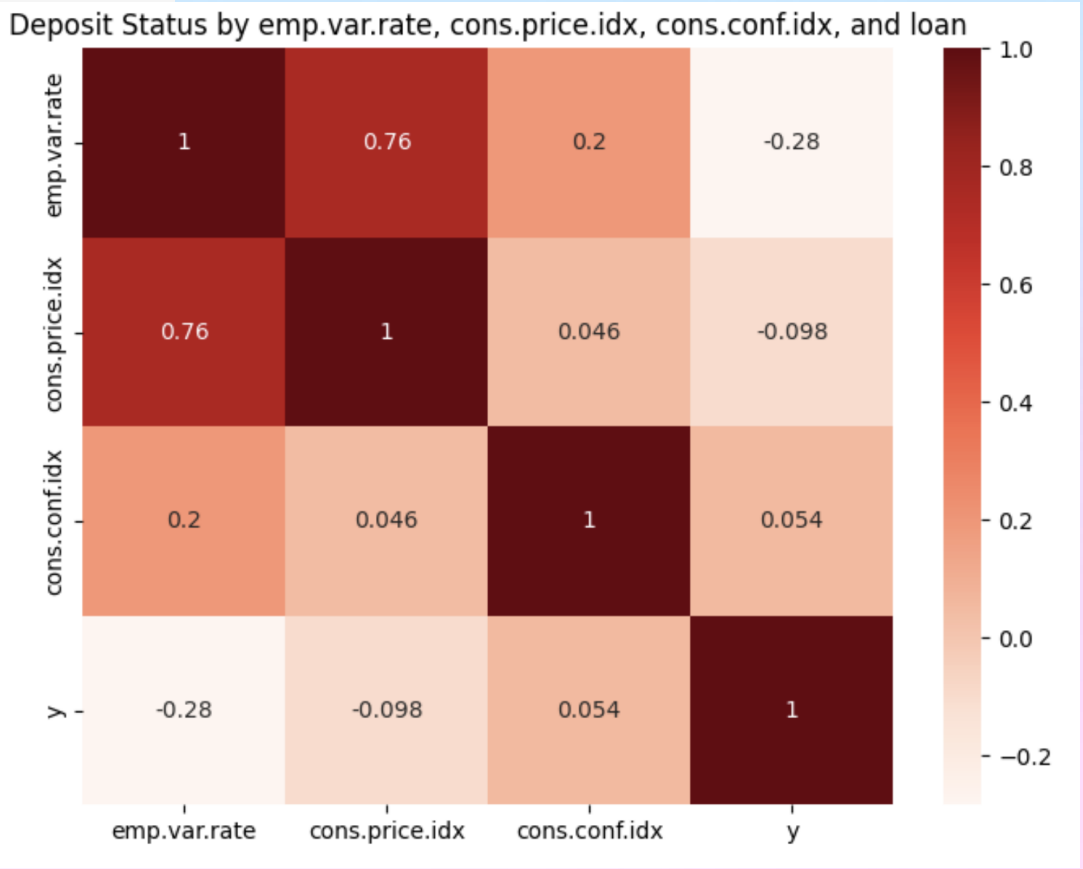
```python
df['cons.price.idx'].value_counts(normalize=True)*100
```

```
cons.price.idx
93.994    18.402525
93.918    16.193251
92.893    14.493809
93.444    12.818645
94.465    10.463705
93.200     9.371207
93.075     4.879825
92.963     1.820830
92.201     1.820830
92.431     1.043943
94.199     0.946832
92.649     0.873999
94.027     0.801165
94.215     0.728332
92.379     0.606943
92.843     0.606943
94.055     0.582666
94.767     0.582666
93.369     0.558388
93.876     0.558388
92.713     0.509832
94.601     0.485555
93.749     0.339888
92.469     0.339888
93.798     0.145666
92.756     0.024278
Name: proportion, dtype: float64
```

INSIGHT

The highest correlation observed is between `emp.var.rate` and `cons.price.idx` (0.76), indicating a significant relationship between these economic indicators.

Other variables (`emp.var.rate` and `y`) show very weak or negligible correlations with each other and with `emp.var.rate` and `cons.price.idx`, suggesting that these factors are not strongly interdependent in the given dataset.



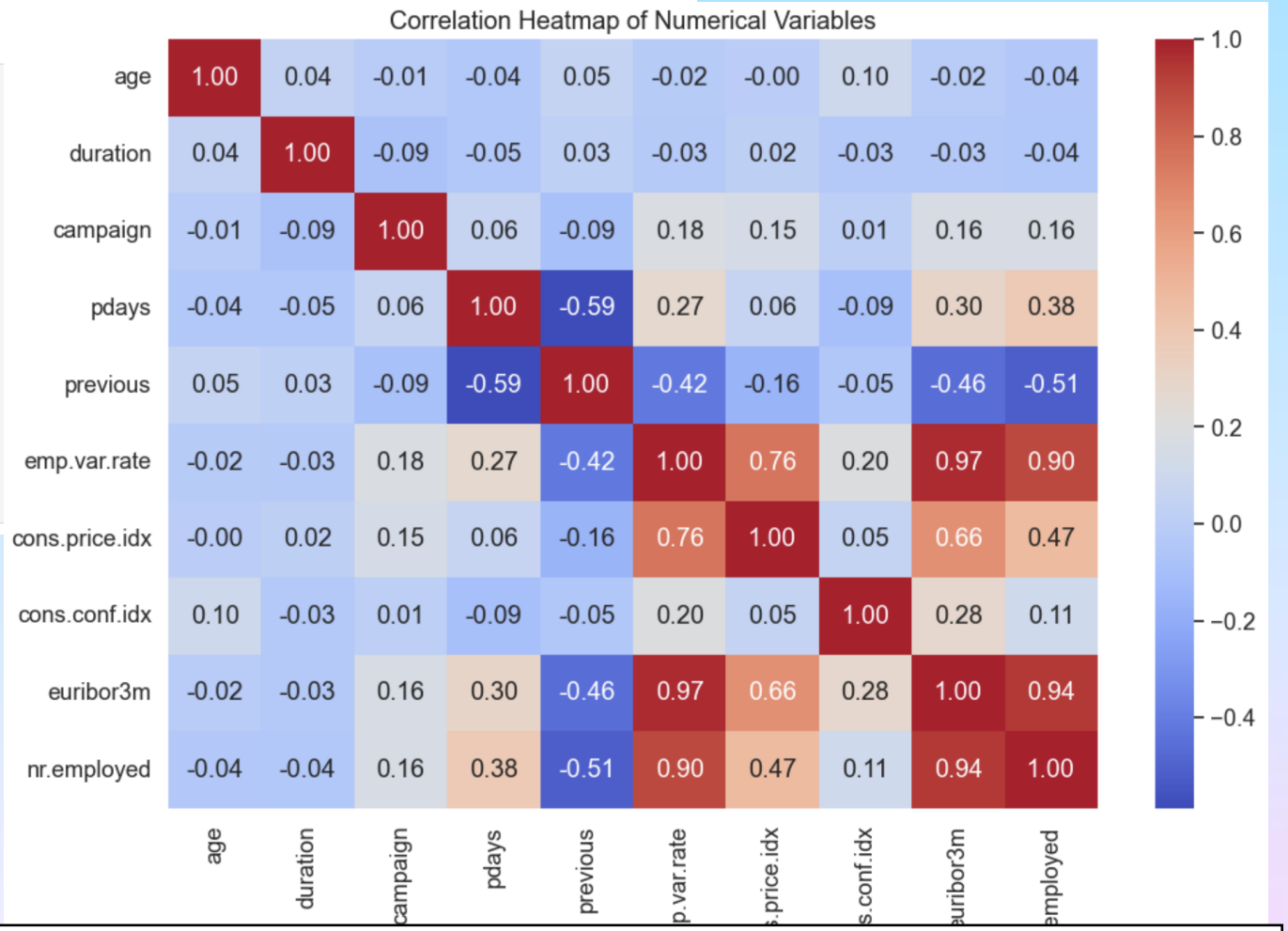
10) Bank Marketing Campaing _Correlation Heatmap of Numerical Variables (HEATMAP))

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

# Exclude non-numeric columns
numeric_df = df.select_dtypes(include='number')

# Calculate the correlation matrix
correlation_matrix = numeric_df.corr()

# Plot the heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap of Numerical Variables')
plt.show()
```



This was provided a correlation heatmap of numerical variables

1. **Positive Correlations:**

- •Variables like ‘emp.var.rate’ and ‘euribor3m’ exhibit a strong positive correlation (indicated by the dark red square). This suggests that these two variables tend to move together in the same direction.
- •Similarly, ‘duration’ and ‘previous’ have a positive correlation.

2.Negative Correlations:

- •‘emp.var.rate’ and ‘cons.price.idx’ show a strong negative correlation (dark blue color).
- •‘pdays’ and ‘previous’ also have a negative correlation.

3.No Correlation:

- •Variables along the diagonal (e.g., ‘age,’ ‘duration’) have no correlation with themselves (white squares).

This heatmap helps identify relationships between variables.

MEET OUR TEAM

KHIN YANDAR HLAING
STUDENT

THIRI WHAR WHAR LWIN
STUDENT

THANK YOU