ITP and NPV Report

Prepared by group 4

MEMBERS:

Rashmi

Madhurya P Barman

Sundar SP

Sree K

CONTENT

INTRODUCTION TO PROBLEM STATEMENT

TECHNOLOGY USED

SKILLS DEVELOPED

PROBLEM SOLVING STEPS

QUESTIONS and SOLUTIONS

TAKEAWAYS and FUTURE STEPS

PROBLEM STATEMENT

The problem is that the Bank Marketing campaigns of a Portuguese banking institution need to identify the factors that cause the customers to tend to take the subscription, as well as Bank Marketing campaigns of a Portuguese banking institution need to identify the reasons behind the customer which make them not take the subscription.

TECHNOLOGY USED

The technology that we have used in this project is Python and to effetively use this tecnhology we have used jypiter notebook.

The Jupyter Notebook is an open-source web application that allows us to create and share documents that contain live code, equations, visualizations, and narrative text.
Uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

SKILLS DEVELOPED

Data Analysis: After working on this project we get to know how to analyse the data from datasets and explore them in various plot.

Data Exploration: We learned how to explore and understand the dataset, including its structure. Also we were able to learn and implement various data visualization techniques.

Data Cleaning: we came to know how to clean and preprocess the dataset and handling missing values.

Exploratory Data Analysis (EDA):

we learned and implemented EDA to uncover insights into the dataset and understand the relationships between different features and the target variable.

Problem solving steps

There are some steps mentioned below to solve a specific problem in python:

- Identify the problem – In the first step we went through the problem deeply according to the described issue and understood its scope and consequences.
- Create solutions – In the next step we thought of various approaches to the issue and discussed potential solutions. This involved working together to develop fresh ideas and viewpoints as well as taking into consideration various problem-solving strategies.
- Evaluated solutions – After possible solutions was generated the next step was to evaluate them and to determine the best course of action. This involved analyzing the potential outcomes of each solution.

QUESTIONS AND SOLUTIONS

1. Import data sets and Load data set

Hint: Make use of **with** statement and write a function to load the data if you are not able to load through the panda's method.

1. We imported/read data using

```
df=pd.read_csv('bank.csv')
df.head(10)
```

2. Checked total number of null values in each column

```
df.isnull().sum()
```

3. Checked information like column name, Non-Null Count and Data type of each column

```
df.info()
```

2. Make the data proper to make use of data for analysis

A. Identify the Features data types before entering into the analysis

B. .Convert the datatypes which are wrongly identified according to the business(domain). Kindly use the User Defined function and loop to convert the data types once.

C. Find and Remove missing if any. Use visualization to find the missing values or Use general method to find the missing values.

D. Find duplicates (if necessary)

a. data.dtypes

b. def convert(dataframe, conversion_dict):

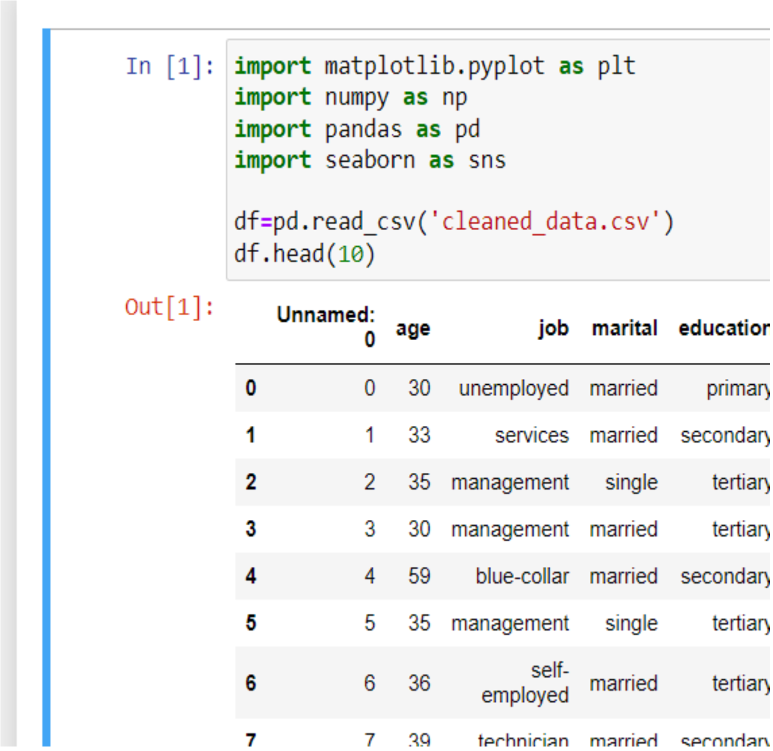
```
    for column, dtype in conversion_dict.items()
        dataframe[column] = dataframe[column].astype(dtype)


conversion_dict = {
    'job': 'category',
    'marital':'category',
    'education':'category',
    'contact':'category',
    'month':'category',
    'poutcome':'category',
}

     convert(data, conversion_dict)
```

c. data.isnull().sum()

d. df=df.drop_duplicates(keep='first')

DATA LOADING:

1. Imports the matplotlib.pyplot library, which is used for plotting graphs.
   import matplotlib.pyplot as plt
2. Imports the numpy library, which is used for scientific computing
   import numpy as np
3. Imports the pandas library, which is used for data manipulation and analysis.
   import pandas as pd
4. Imports the seaborn library, which is used for statistical data visualization.
   import seaborn as sns
5. Reads the cleaned_data.csv file into a Pandas DataFrame called df.
   df=pd.read_csv('cleaned_data.csv')
6. Prints the first 10 rows of the DataFrame.
   df.head(10)



```
In [1]: import matplotlib.pyplot as plt
        import numpy as np
        import pandas as pd
        import seaborn as sns

        df=pd.read_csv('cleaned_data.csv')
        df.head(10)
```

Out[1]:

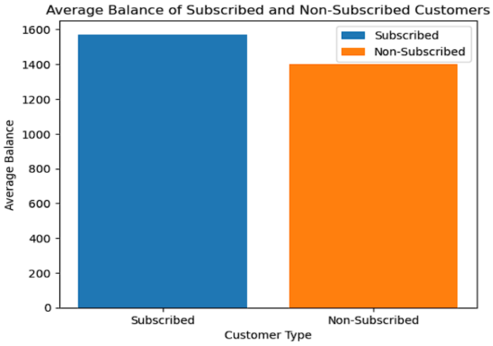
	Unnamed: 0	age	job	marital	education
0	0	30	unemployed	married	primary
1	1	33	services	married	secondary
2	2	35	management	single	tertiary
3	3	30	management	married	tertiary
4	4	59	blue-collar	married	secondary
5	5	35	management	single	tertiary
6	6	36	self-employed	married	tertiary
7	7	39	technician	married	secondary

3. Find the average balance of the customer who belongs to the subscribed customer and non-subscribed customer and also use a related plot to show them in visualization.

- This code creates two new dataframes for subscribed and non-subscribed customers. Next, it will calculate the average balance for subscribed and non-subscribed customers. Finally, it will create a bar plot of the average balance for subscribed and non-subscribed customers.
- The output of the code is a bar plot that shows that the average balance of subscribed customers is higher than the average balance of non-subscribed customers. This suggests that there is a positive correlation between balance and subscription, meaning that customers with higher balances are more likely to subscribe to the term deposit.

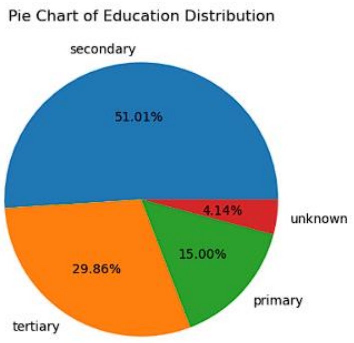
```
 import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
```

```
subscribed_df = df[df['y']=='yes']
non_subscribed_df = df[df['y']=='no']

subscribed_mean_balance = subscribed_df['balance'].mean()
non_subscribed_mean_balance = non_subscribed_df['balance'].mean()

plt.bar('Subscribed',subscribed_mean_balance,label='Subscribed')
plt.bar('Non-Subscribed',non_subscribed_mean_balance,label='Non-Subscribed')

plt.xlabel('Customer Type')
plt.ylabel('Average Balance')
plt.title('Average Balance of Subscribed and non-Subscribed Customers')
plt.show()
```



4. Use a pie plot to find the distribution(frequency) of the education. Make sure to add labels and show the percentage of each education distribution

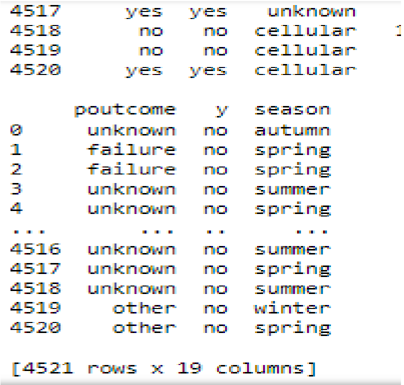
```
distribution_education = df['education'].value_counts()
distribution_education.to_frame()

plt.pie(distribution_education,labels=['secondary','tertiary','primary','unknown'],autopct ='%1.2f%%')
plt.title('Pie Chart of Education Distribution')
plt.show()
```



5. Create a function that should be able to create a new feature(Variable) called season using the month column.

```
def create_season_feature(df):
    # Get the unique months in the dataset
    months = df['month'].unique()

    # Create a dictionary to map months to seasons
    season_map = {
        'jan': 'winter',
        'feb': 'winter',
        'mar': 'spring',
        'apr': 'spring',
        'may': 'spring',
        'jun': 'summer',
        'jul': 'summer',
        'aug': 'summer',
        'sep': 'autumn',
        'oct': 'autumn',
        'nov': 'autumn',
        'dec': 'winter',
    }

    # Create the 'season' column
    df['season'] = df['month'].map(season_map)

    return df
```

```
df = create_season_feature(df)
print(df)
```



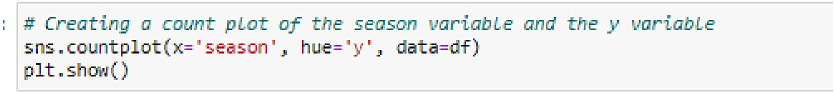
```
4517      yes   yes    unknown
4518       no    no   cellular   1
4519       no    no   cellular
4520      yes   yes   cellular

      poutcome    y   season
0      unknown   no   autumn
1      failure   no   spring
2      failure   no   spring
3      unknown   no   summer
4      unknown   no   spring
...        ...   ..      ...
4516   unknown   no   summer
4517   unknown   no   spring
4518   unknown   no   summer
4519     other   no   winter
4520     other   no   spring

[4521 rows x 19 columns]
```

6. Use the count plot with a variable that you created in the above question and also the Y variable to find the class distribution

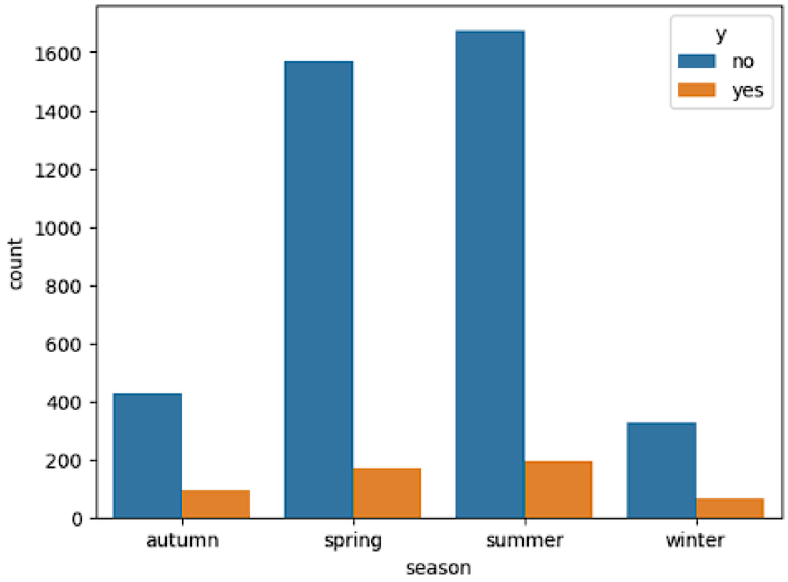
```
sns.countplot(x='season', hue='y', data=df)
plt.show()
```

7. Use the Pdays feature and find does it cause any effect on the subscription of the term using the bar plot.

```
sns.barplot(y='pdays', x='y', data=df)
plt.show()
```



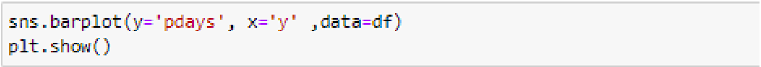
```
# Creating a count plot of the season variable and the y variable
sns.countplot(x='season', hue='y', data=df)
plt.show()
```



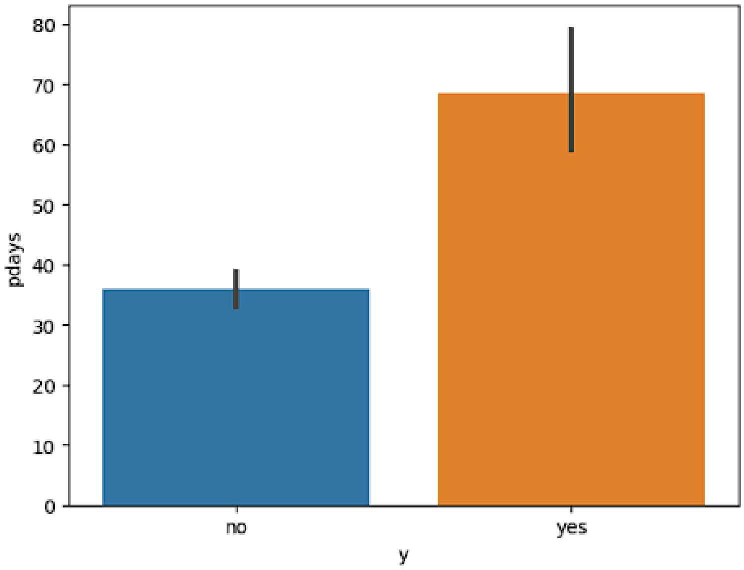
8. Replace the -1 as nan values for the P-days store.

df['pdays'].replace(-1, np.NAN, inplace=True)

print(df['pdays'].head())

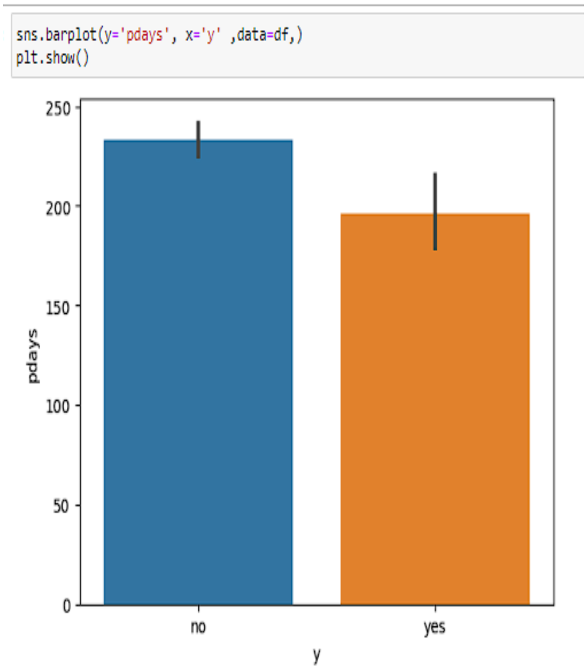


```
sns.barplot(y='pdays', x='y' ,data=df)
plt.show()
```

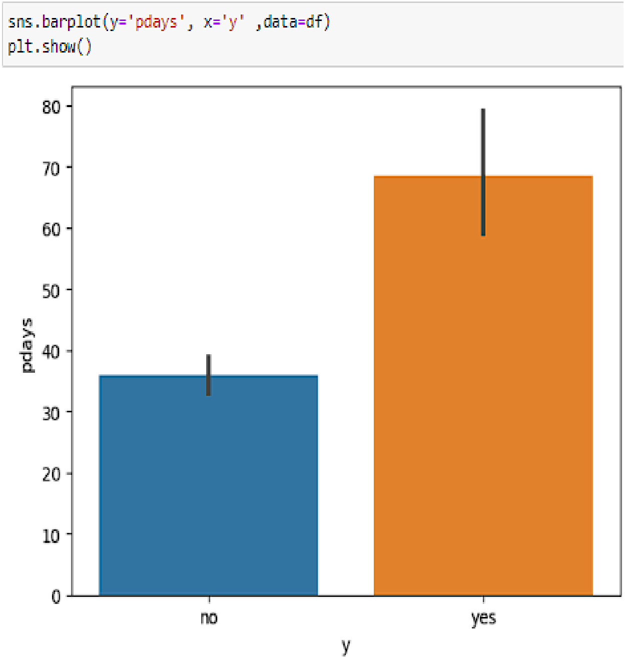


9. Once you are done with question number 8, do the same analysis as question number 7. And observe the difference between question number 7 and question number 9.

sns.barplot(y='pdays', x='y', data=df)
plt.show()



```
sns.barplot(y='pdays', x='y' ,data=df,)
plt.show()
```

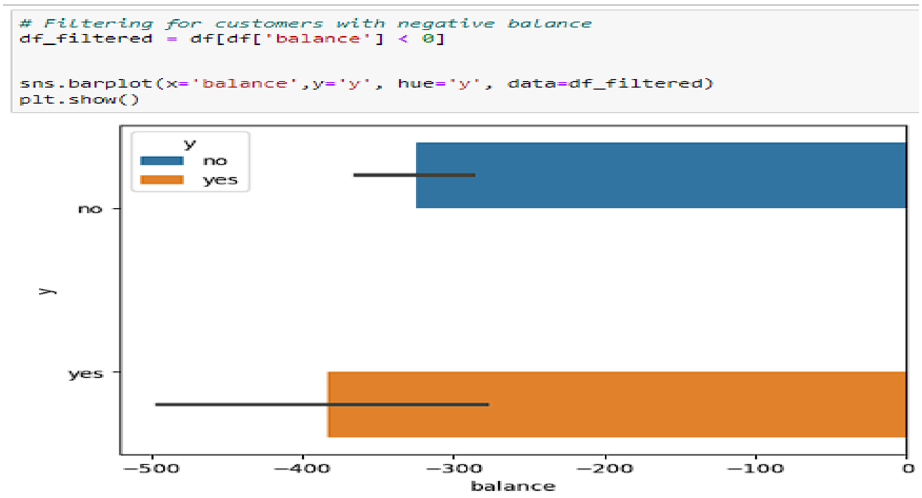


```
sns.barplot(y='pdays', x='y' ,data=df)
plt.show()
```

10. Does the customer take the term subscription who has less than 0 balance? Hint: Use any kind of plot which would you the related information to this question.

df_filtered = df[df['balance']<0]

sns.barplot(x='balance',y='y', hue='y', data= df_filtered)
plt.show()

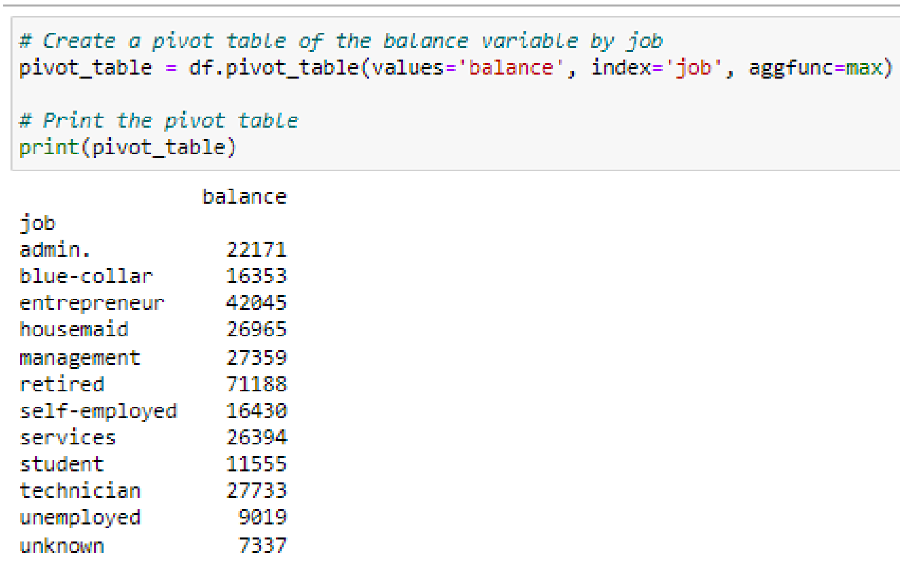


```
# Filtering for customers with negative balance
df_filtered = df[df['balance'] < 0]

sns.barplot(x='balance',y='y', hue='y', data=df_filtered)
plt.show()
```

11. Use Pivot table to find the maximum balance for each type of job.

pivot_table = df.pivot_table(values='balance', index = 'job', aggfunc=max)

#printing the pivot table
print(pivot_table)

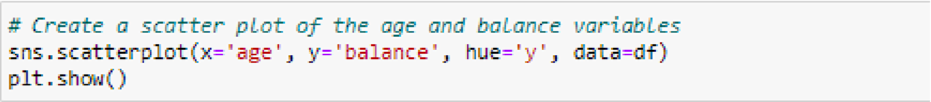


```
# Create a pivot table of the balance variable by job
pivot_table = df.pivot_table(values='balance', index='job', aggfunc=max)

# Print the pivot table
print(pivot_table)
```

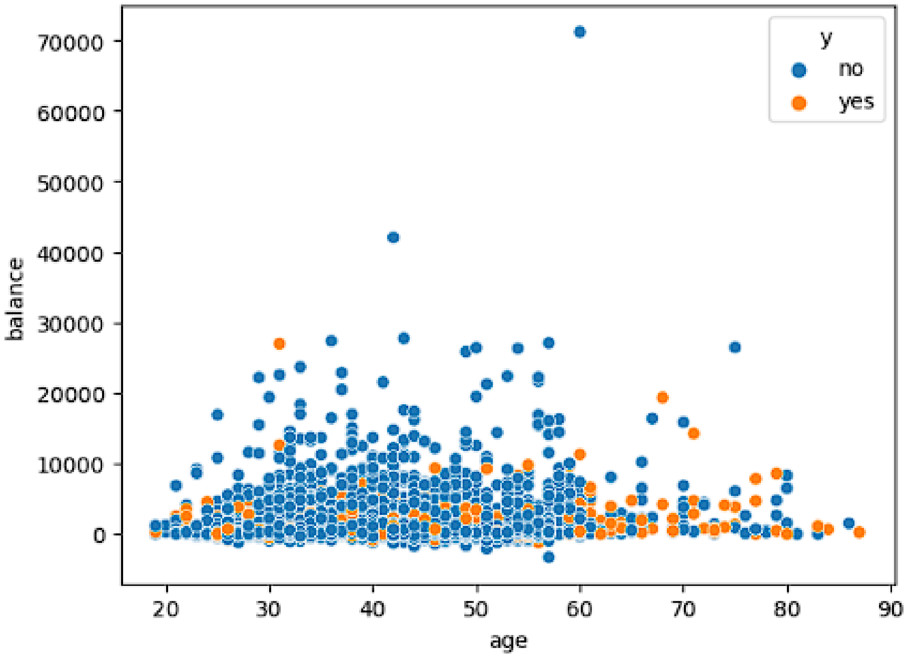
```
               balance
job
admin.           22171
blue-collar      16353
entrepreneur     42045
housemaid        26965
management       27359
retired          71188
self-employed    16430
services         26394
student          11555
technician       27733
unemployed        9019
unknown           7337
```

12. Use the Age, balance, and Y column to plot the scatter plot and find what kind of relationship Age and balance had, and See the points which belong 0 and 1 class and how they are distributed.

sns.scatterplot(x='age', y='balance', hue='y', data=df)
plt.show()

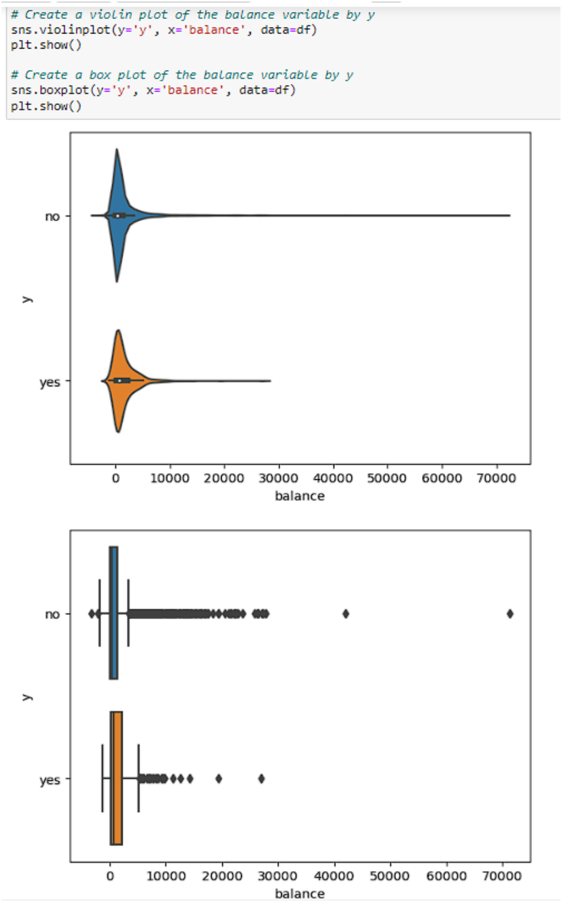


```
# Create a scatter plot of the age and balance variables
sns.scatterplot(x='age', y='balance', hue='y', data=df)
plt.show()
```



13. Use the violin plot and also the box plot to find the distribution of the balance for each class of the Y column. And try to tell why we have a Violin plot and Box plot both rather than one.

```
sns.violinplot(y='y', x='balance', data=df)
plt.show()

#Creating a box plot of the balance variable by y
sns.boxplot(y='y',x='balance',data=df)
plt.show()
```

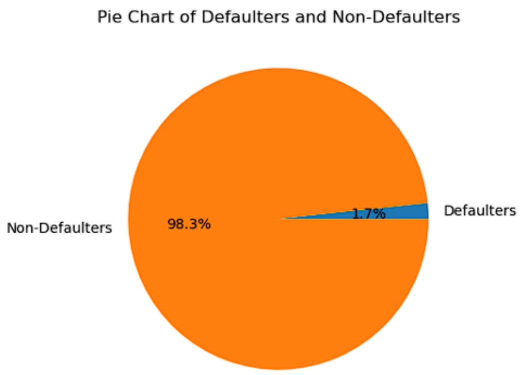


```
# Create a violin plot of the balance variable by y
sns.violinplot(y='y', x='balance', data=df)
plt.show()

# Create a box plot of the balance variable by y
sns.boxplot(y='y', x='balance', data=df)
plt.show()
```

14. Use a pie plot to know the Proportion(distribution) of the defaulters and non-defaulters.
Note: Try to explore more parameters that are there in the pie-plot method.

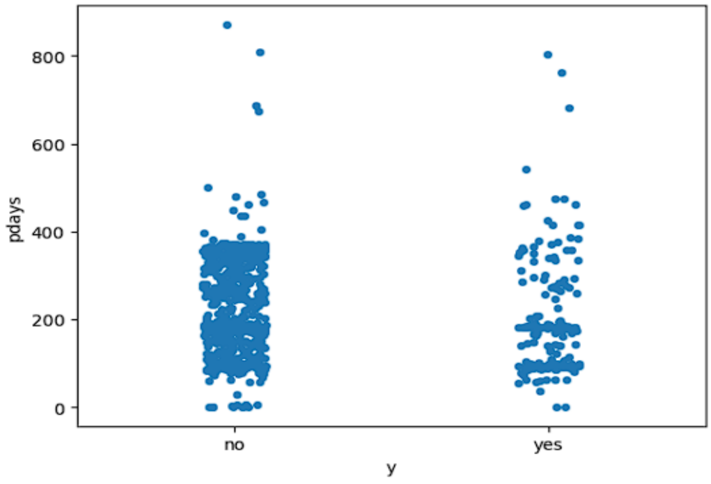
```
#Get the defaulters and non-defaulters
defaulters = df[df['default'] == 'yes']
non_defaulters = df[df['default'] == 'no']

#Get the number of defaulters and non-defaulters
number_of_defaulters = len(defaulters)
number_of_non_defaulters = len(non_defaulters)

#Get pie chart slice lables
pie_chart_slice_labels = ['Defaulters', 'Non-Defaulters']

#Get pie chart slice values
pie_chart_slice_values = [number_of_defaulters,
number_of_non_defaulters]
```

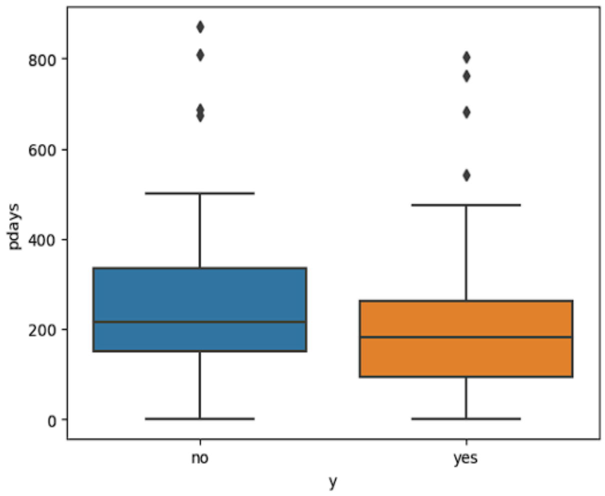
```
#Create a pie chart of the defaulters and non-defaulters
plt.pie(pie_chart_slice_values, labels=pie_chart_slice_labels,
autopct="%1.1f%%")
plt.title("Pie Chart of Defaulters and Non-Defaulters")
plt.show()
```



15. Use Box plot and strip plot to know the distribution of the Pdays with respect to Y classes and differentiate both plots.

```
sns.boxplot(x='y',y='pdays',data=df)
plt.show()

#Create a strip plot of the pdays variable by y
sns.stripplot(x='y', y='pdays', data=df)
plt.show()
```





TAKEAWAYS AND FUTURE STEPS

Based on the analysis of the Bank Marketing Project, we can conclude that the following factors are most likely to influence whether a customer subscribes to a term deposit:

- Age: Older customers are more likely to subscribe to a term deposit than younger customers.
- Balance: Customers with higher balances are more likely to subscribe to a term deposit than customers with lower balances.
- Job: Customers with higher-paying jobs are more likely to subscribe to a term deposit than customers with lower-paying jobs.
- Marital status: Married customers are more likely to subscribe to a term deposit than single customers.
- Education: Customers with higher levels of education are more likely to subscribe to a term deposit than customers with lower levels of education.

The analysis of the Bank Marketing Project suggests that there are a number of factors that can influence whether a customer subscribes to a term deposit. By understanding these factors, banks can improve their marketing campaigns and increase their chances of success.

Future steps: The dataset could be used to build predictive models that can predict whether a customer is likely to subscribe to a term deposit. This would help the bank to target its marketing efforts more effectively.

THANK
YOU!