13

# Marketing for Financial Services

Exploratory Data Analysis and Predictive Modeling for Marketing Term-Deposit Scheme in the Financial Services Industry

# AGENDA

***Introduction***

- ❑ *Our Team*
- ❑ *Overview of our Topic & Technologies Used*

***Check Point 1***

- ❑ *Data Manipulation and Visualization using Python*
- ❑ *Exploratory Data Analysis & Statistical Analysis*

***Check Point 2***

- ❑ *Visualization using Power-BI*
- ❑ *Model building using ML algorithms*

***Check Point 3***

- ❑ *Summary*
- ❑ *Question and Answers*

***SNEHA SARKAR***
***2041019160***

***SMRITIK SWAIN***
***2041019342***

***RAMAKANTA PARIDA***
***2041016102***

***ROHIT KUMAR SHARMA***
***2041019212***

***RANBIR SINGH***
***2041019280***

***SUBHRAJEET PRADHAN***
***2041016232***

# Problem-Statement

DB Bank is a large public sector bank that wants to increase the number of term deposits it sells. The bank has conducted a marketing campaign to promote its term-deposit scheme, but it wants to improve the success rate of the campaign.
The bank has collected data on the customers who were contacted during the campaign, as well as the outcome of the campaign (whether or not the customer agreed to place a term deposit). The bank wants to use this data to perform exploratory data analysis and predictive modeling.
The goal of the analysis is to identify the factors that are most predictive of whether or not a customer will place a term deposit. The bank can then use this information to improve the targeting of its marketing campaigns and increase the success rate.

***The specific objectives of the analysis are to:***

- Identify the most important factors that are predictive of whether or not a customer will place a term deposit.
- Develop a predictive model that can be used to predict whether or not a customer will place a term deposit.
- Evaluate the performance of the predictive model.
- Make recommendations for how the bank can improve the targeting of its marketing campaigns.

## Predictive modeling & Evaluation

*Build model to predict customer behavior & accessing model performance using metrics*

## Descriptive analytics

*Summarize & visualize key features*

## Feature engineering

*Prepare data for modeling using Python*

## Data exploration

*Find patterns & trends in data*

## Insights & recommendation

*Provide insights for Model improvement*

# Methodology

# Data Collection & exploration

*After importing the data sets, we concatenated them to create a single data set. This allowed us to analyze the data more easily and efficiently.*

bank-campaign.ipynb

```python
import pandas as pd
import numpy as np
df1 = pd.read_csv('Data/Customer_and_bank details_p1.csv')
df2 = pd.read_csv('Data/Customer_campaign_details_p1.csv')
df3 = pd.read_csv('Data/Customer_social_economic_data_p1.csv')
df4 = pd.read_csv('Data/Customer_Response_data_p1.csv')
```

#Data Collection

bank-campaign.ipynb

```python
df2.drop(['Customer_id'], axis=1, inplace=True)
df3.drop(['Customer_id'], axis=1, inplace=True)
df4.drop(['Customer_id'], axis=1, inplace=True)
df = pd.concat([df1,df2,df3,df4], axis=1)
df.shape
```

#Data Exploration

bank-campaign.ipynb

```python
print('df1 shape : ', df1.shape)
print('df2 shape : ', df2.shape)
print('df3 shape : ', df3.shape)
print('df4 shape : ', df4.shape)
```

In [8]:
```python
df.head()
```

Out[8]:

| | Customer_id | age | job | marital | education | default | housing | loan | Region_Coc |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 56 | services | married | high.school | no | no | yes | |
| 1 | 2 | 45 | services | married | basic.9y | unknown | no | no | |
| 2 | 3 | 59 | admin. | married | professional.course | no | no | no | |
| 3 | 4 | 41 | blue-collar | married | unknown | unknown | no | no | |
| 4 | 5 | 24 | technician | single | professional.course | no | yes | no | |

5 rows × 25 columns

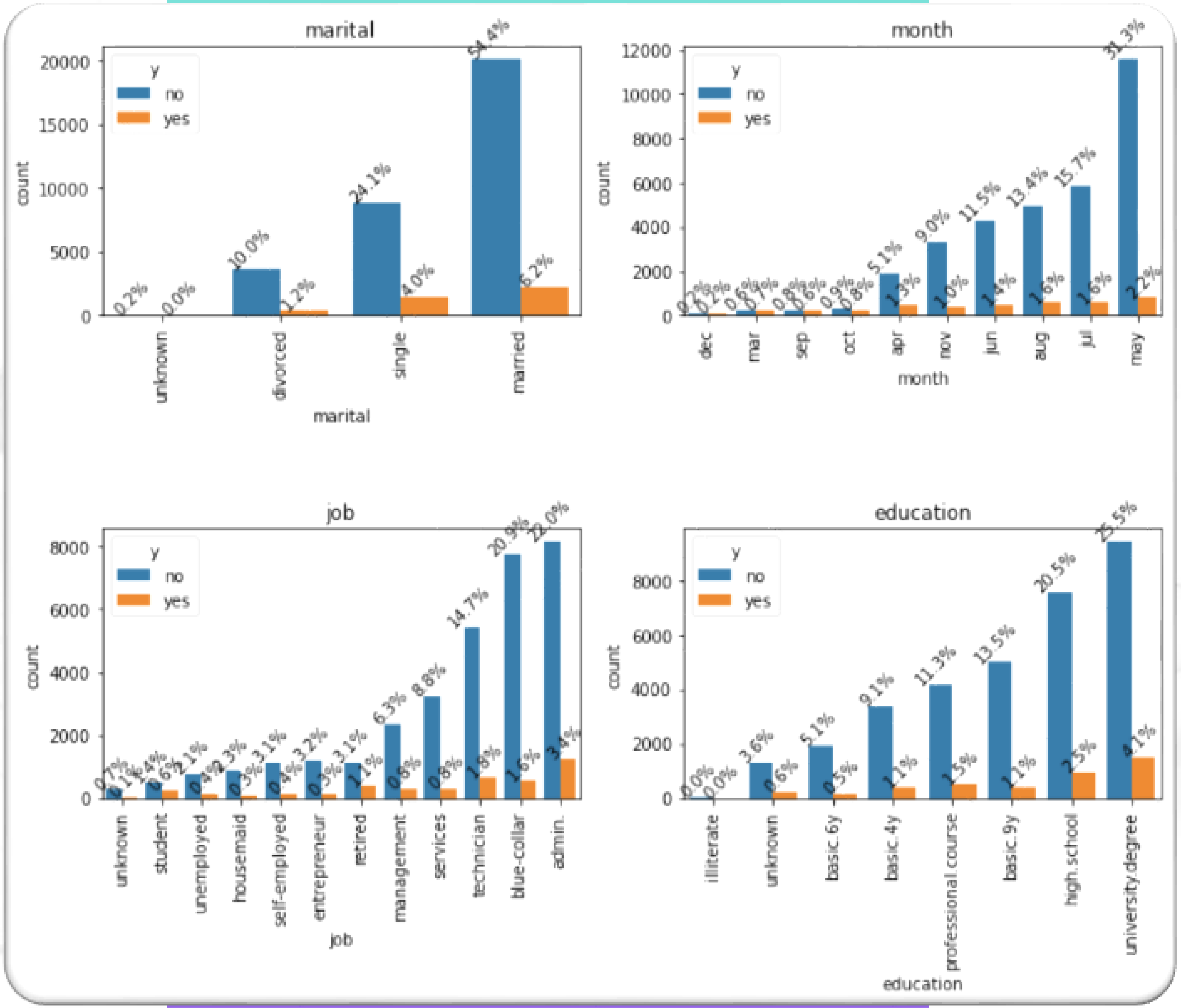
In [10]:
```python
df.describe()
```

Out[10]:

| | Customer_id | age | duration | campaign | pdays | previous |
|---|---|---|---|---|---|---|
| count | 37084.000000 | 37084.000000 | 37084.000000 | 37084.000000 | 37084.000000 | 37084.000000 |
| mean | 18542.500000 | 40.042714 | 258.237946 | 2.569545 | 962.530849 | 0.172986 |
| std | 10705.373028 | 10.432965 | 258.730909 | 2.770611 | 186.773063 | 0.495681 |
| min | 1.000000 | 17.000000 | 0.000000 | 1.000000 | 0.000000 | 0.000000 |
| 25% | 9271.750000 | 32.000000 | 102.000000 | 1.000000 | 999.000000 | 0.000000 |
| 50% | 18542.500000 | 38.000000 | 180.000000 | 2.000000 | 999.000000 | 0.000000 |
| 75% | 27813.250000 | 47.000000 | 319.250000 | 3.000000 | 999.000000 | 0.000000 |
| max | 37084.000000 | 98.000000 | 4918.000000 | 56.000000 | 999.000000 | 7.000000 |

In [9]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 37084 entries, 0 to 37083
Data columns (total 25 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Customer_id     37084 non-null  int64
 1   age             37084 non-null  int64
 2   job             37084 non-null  object
 3   marital         37084 non-null  object
 4   education       37084 non-null  object
 5   default         37084 non-null  object
 6   housing         37084 non-null  object
 7   loan            37084 non-null  object
 8   Region_Code     37084 non-null  object
 9   State_Code      37084 non-null  object
 10  City_Code       37084 non-null  object
 11  contact         37084 non-null  object
 12  month           37084 non-null  object
 13  day_of_week     37084 non-null  object
 14  duration        37084 non-null  int64
 15  campaign        37084 non-null  int64
 16  pdays           37084 non-null  int64
 17  previous        37084 non-null  int64
 18  poutcome        37084 non-null  object
 19  emp.var.rate    37084 non-null  float64
 20  cons.price.idx  37084 non-null  float64
 21  cons.conf.idx   37084 non-null  float64
 22  euribor3m       37084 non-null  float64
 23  nr.employed     37084 non-null  float64
 24  y               37084 non-null  object
dtypes: float64(5), int64(6), object(14)
memory usage: 7.1+ MB
```




# Data Visualization

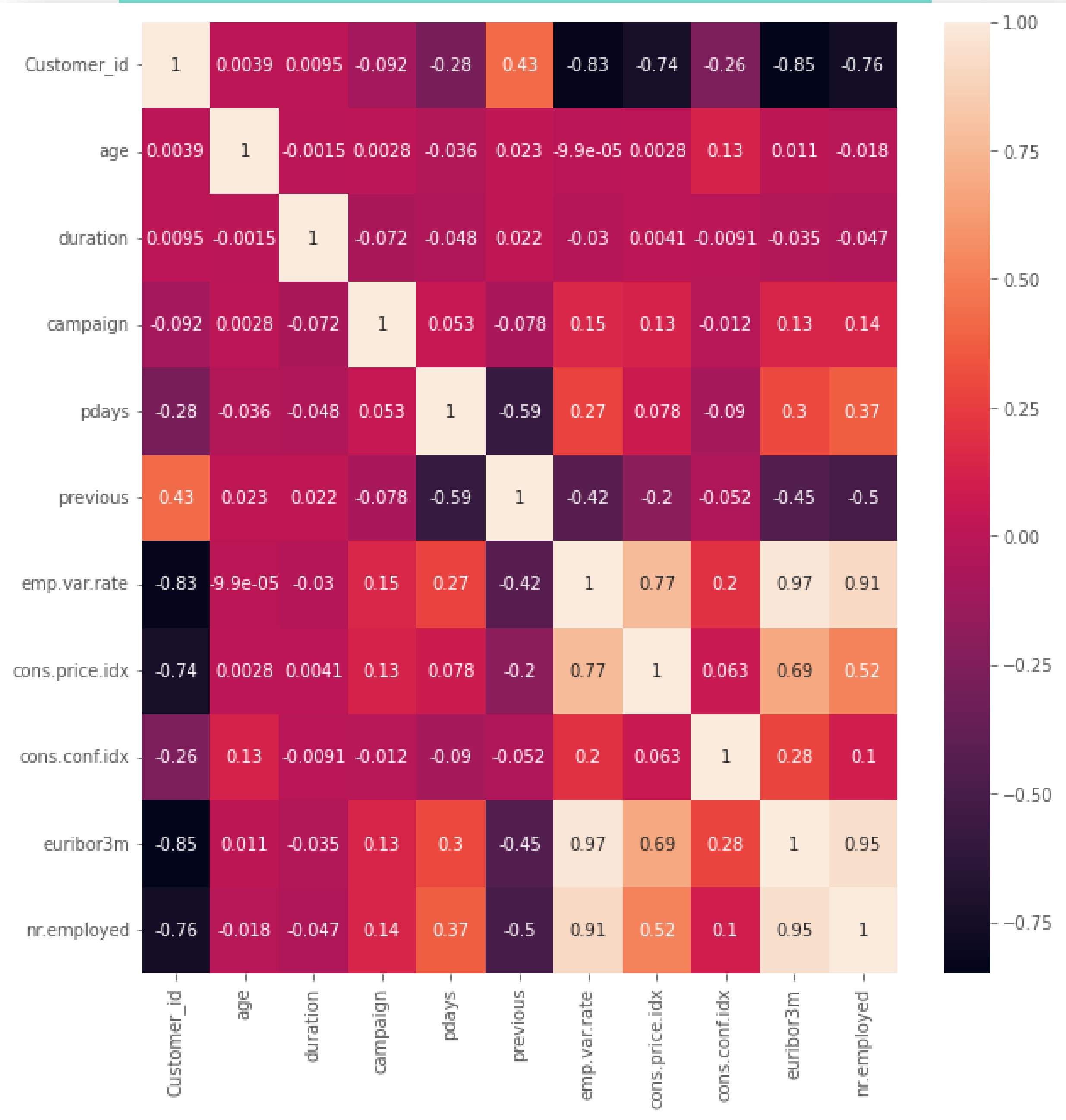
*Conclusion-*

- *Married customers have done most deposit i.e. 54%*
- *31% of customers have done most deposit whom campaign was done in month of May*
- *25% of customers deposits are done by university-degree students*
- *Among job personal admin and blue_collar category have done approx 20% deposits*

# Correlation Matrix

bank-campaign.ipynb

```
_,ax=plt.subplots(figsize=(10,10))
sn.heatmap(df.corr(),annot=True)
plt.show()
```



# Correlation Values

bank-campaign.ipynb

```python
target_column = 'campaign'

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

# Find the correlation with the target column
correlation_with_target_column =
correlation_matrix[target_column].drop(target_column)

# Find the top correlated columns
top_correlated_columns =
correlation_with_target_column.abs().nlargest(7)

print("Top correlated columns with", target_column, ":")
print(top_correlated_columns)
```

*OUTCOMES*

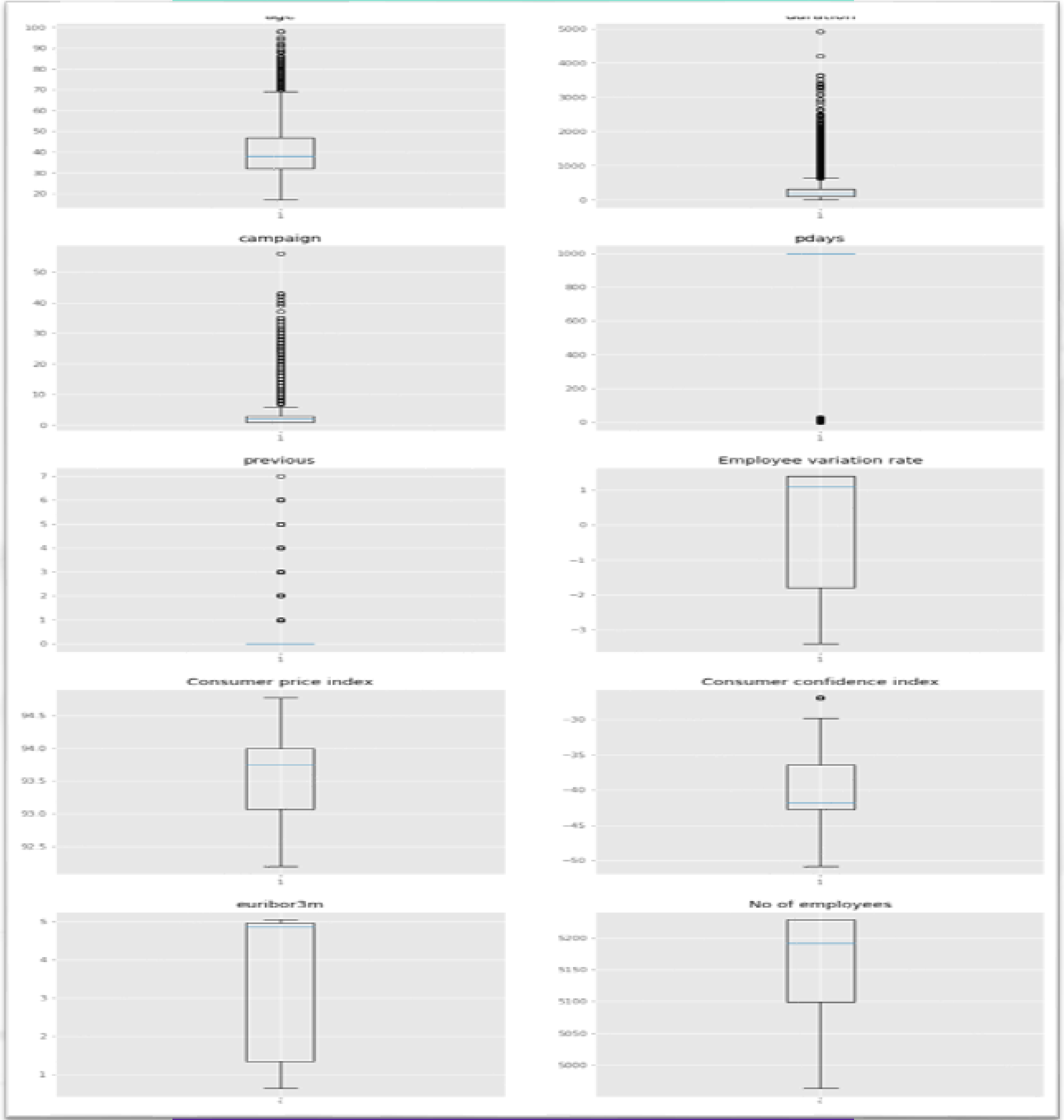
- *The number of employees is highly correlated with the employee variation rate. This suggests that when the number of employees in a company changes significantly, it tends to be correlated with changes in the overall number of employees. For example, if a company experiences a large increase in the number of employees, it is likely that the employee variation rate will also be high.*
- *The consumer price index is highly correlated with bank interest rates. Specifically, when the consumer price index is higher (indicating higher inflation), the bank interest rates tend to be higher as well. This relationship is often observed because central banks might raise interest rates to control inflation.*
- *The employee variation rate also correlates with bank interest rates. This means that there is some relationship between changes in the number of employees and changes in bank interest rates. This correlation suggests that economic factors influencing employment levels may also impact interest rates.*

# Feature Engineering

bank-campaign.ipynb

```
plt.figure(figsize = (15, 30))

ax=plt.subplot(521)
plt.boxplot(df6['age'])
ax.set_title('age')
ax=plt.subplot(522)
plt.boxplot(df6['duration'])
ax.set_title('duration')
ax=plt.subplot(523)
plt.boxplot(df6['campaign'])
ax.set_title('campaign')
```



# Classification Report

bank-campaign.ipynb

```
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.model_selection import cross_val_score

print("Classification Report:\n",classification_report(y_test,
y_pred))
```

#Classification Report

```
Classification Report:
               precision    recall  f1-
score   support

           0       0.95      0.98      0.96      5883
           1       0.62      0.37      0.46       513

    accuracy                           0.93      6396
   macro avg       0.78      0.67      0.71      6396
weighted avg       0.92      0.93      0.92      6396
```

# Logistic-Regression

bank-campaign.ipynb

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter=1000)
model.fit(X_train,y_train)
model.score(X_test, y_test)
```

# OUTPUT

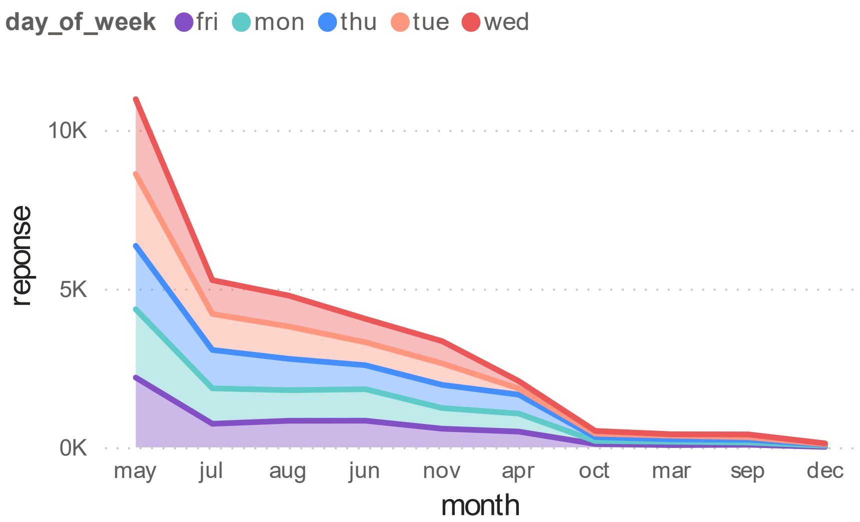
0.9312070043777361

# Bank Campaign Report

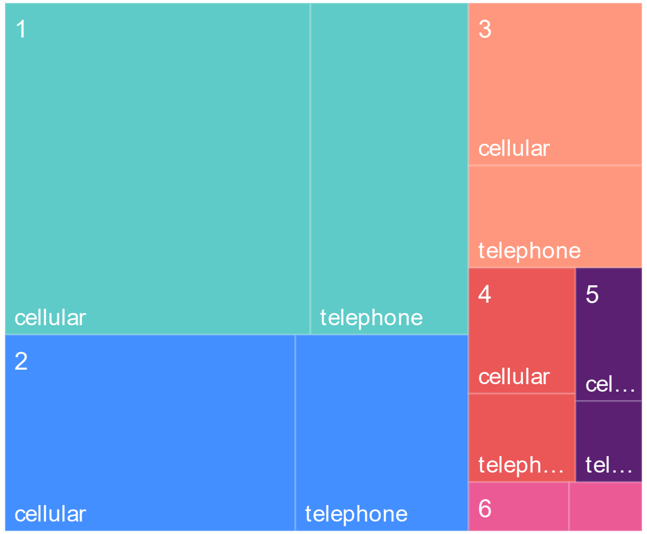
Region_Name

| | | |
|---|---|---|
| (Blank) | East | South |
| Central | North | West |

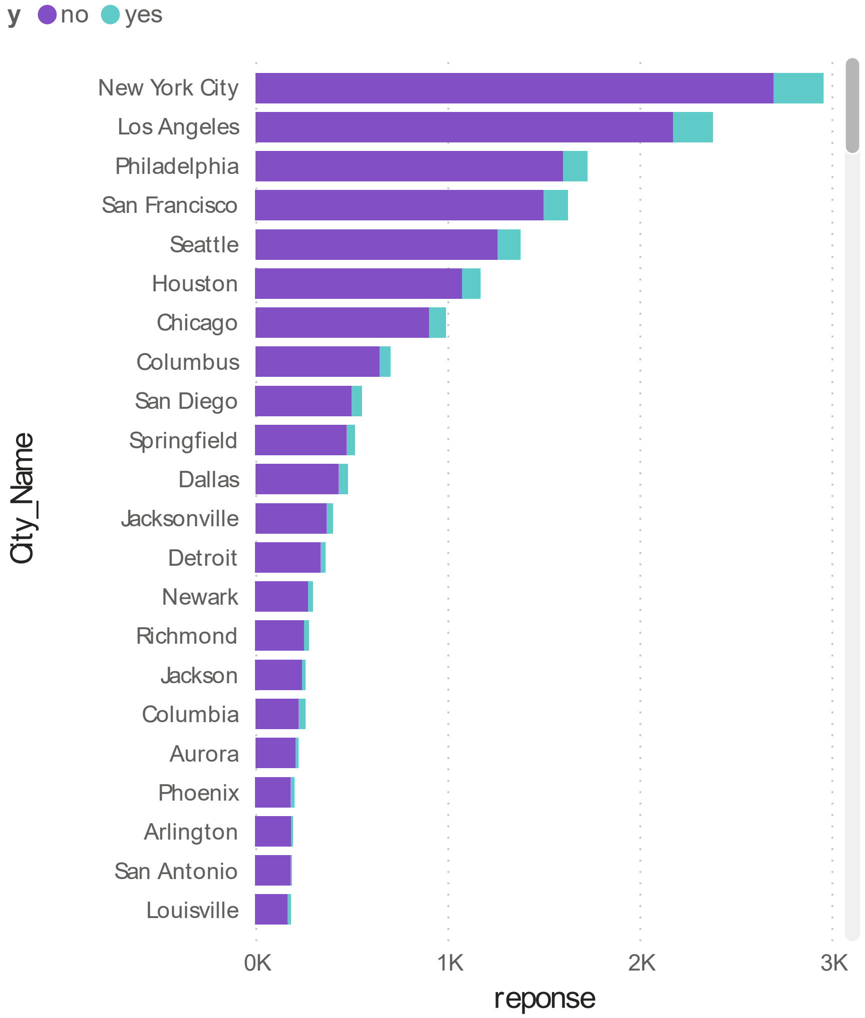
## reponse by month and day_of_week



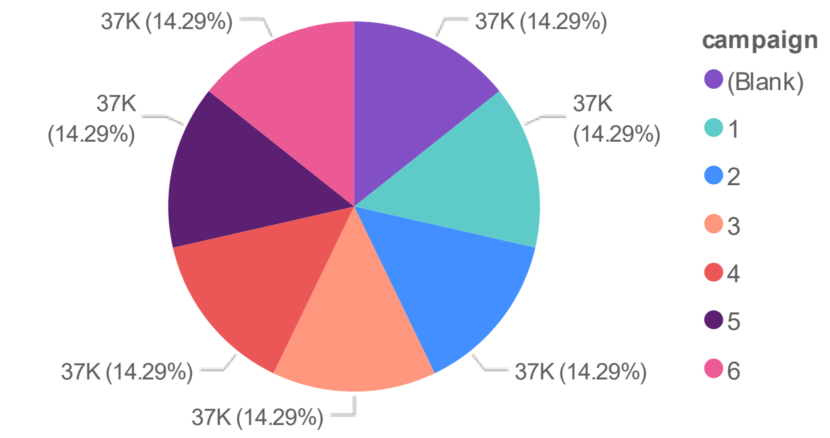

## reponse by campaign and contact



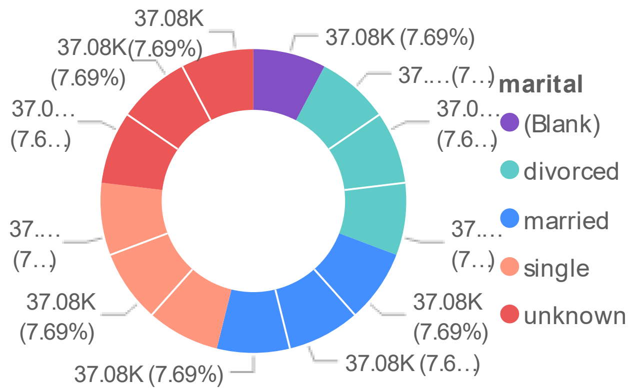

## reponse by City_Name and y




## response by campaign




## Count of y by marital and loan



# Thank You