BankLeadInsight: Predictive Analytics for Term Deposit Subscriptions

PROJECT REPORT
Submitted by
Anila KR

INTRODUCTION

In response to the request from ABC Portugal Bank, this project aims to develop a classification algorithm to predict prospective leads likely to subscribe to a term deposit. Leveraging the dataset provided by the bank, our team at XYZ consultancy Ltd. undertakes a comprehensive approach encompassing data preprocessing, exploratory analysis, model development, validation, and optimization.

Motivated by the opportunity to enhance model accuracy during the lockdown period, we emphasize feature engineering as a critical aspect of our methodology. By transforming and selecting relevant features, we aim to improve the predictive power of our algorithm. Our objective is not only to build a reliable model but also to derive actionable insights for the bank to optimize its marketing funnel and increase term deposit subscriptions.

Technologically, we employ a range of Python libraries including scikit-learn, seaborn, Plotly, matplotlib, pandas, and NumPy. These tools facilitate efficient data manipulation, visualization, and model development, ensuring a robust analytical framework.

Throughout the project, our team collaborates closely to meet the objectives outlined, with Ben Roshan leading the effort. We acknowledge the Kaggle dataset as a valuable resource for this endeavor.

Ultimately, this project seeks to empower ABC Portugal Bank with a data-driven approach to target their marketing efforts, thereby enhancing customer acquisition and promoting financial inclusion.

GENERAL BACKGROUND

The banking industry is undergoing rapid transformation driven by technological advancements, changing consumer preferences, and evolving regulatory landscapes. In this dynamic environment, banks are increasingly relying on data analytics and machine learning techniques to gain competitive advantage and improve operational efficiency.

One area where data analytics plays a crucial role is in marketing campaigns. Banks often conduct targeted marketing campaigns to promote various financial products and services, including term deposits. Term deposits, also known as certificates of deposit (CDs), are fixed-term investments offered by banks where customers deposit funds for a specified period at a fixed interest rate.

Identifying and targeting the right customers for term deposit subscriptions is essential for banks to maximize their marketing ROI and increase their deposit base. Traditional approaches to customer segmentation and targeting are often limited in their effectiveness, leading banks to turn to advanced analytics and predictive modeling techniques for more precise and personalized targeting.

Predictive analytics enables banks to analyze large volumes of customer data to identify patterns, trends, and correlations that can help predict future behavior, such as the likelihood of subscribing to a term deposit. By leveraging machine learning algorithms, banks can develop predictive models that segment customers based on their likelihood to subscribe to a term deposit, allowing them to tailor their marketing efforts and messages accordingly.

Furthermore, data preprocessing, feature engineering, and exploratory data analysis are crucial steps in the predictive modeling process. Preprocessing involves cleaning and transforming raw data to make it suitable for analysis, while feature engineering involves selecting, creating, and transforming features (variables) that are most predictive of the target outcome. Exploratory data analysis helps uncover insights and patterns in the data that can inform modeling decisions and hypothesis generation.

Overall, the integration of data analytics and machine learning into marketing campaigns enables banks to better understand their customers, personalize their marketing efforts, and ultimately drive business growth. By leveraging data-driven insights, banks can improve customer targeting, optimize marketing ROI, and enhance customer satisfaction and loyalty.

SCOPE OF THE PROJECT

The scope of this project entails the creation of a classification algorithm aimed at predicting prospective leads likely to subscribe to a term deposit for ABC Portugal Bank. Commencing with the acquisition of the dataset from the bank, the project progresses through essential preprocessing steps, including data cleaning, handling missing values, and encoding categorical variables to ensure data suitability. Subsequently, feature engineering techniques are applied to enhance predictive power, alongside feature selection methods to identify the most influential variables. Through exploratory data analysis (EDA), insights into underlying patterns and trends are gleaned, utilizing visualizations and statistical analysis to uncover potential predictors of term deposit subscriptions. Machine learning algorithms, such as logistic regression, decision trees, and random forests, are then trained on the preprocessed data to develop a robust classification model. Following model development, validation procedures are employed to assess performance metrics and ensure reliability on unseen data. Optimization strategies, including hyperparameter tuning and addressing overfitting, are implemented to refine model performance. Finally, actionable recommendations derived from the analysis and modeling process are provided to ABC Portugal Bank to enhance marketing strategies and increase term deposit subscriptions. Throughout the project, comprehensive documentation and reporting are maintained to communicate methodology, findings, and recommendations effectively to stakeholders.

IMPLIMENTATIONS

About Dataset

It is a dataset that describing Portugal bank marketing campaigns results.Conducted campaigns were based mostly on direct phone calls, offering bank client to place a term deposit. If after all marking afforts client had agreed to place deposit - target variable marked 'yes', otherwise 'no'

Source of the data: https://archive.ics.uci.edu/ml/datasets/bank+marketing

Prepare Data for Consumption

Import Libraries

Let's import all necessary libraries for the analysis and along with it let's bring down our dataset

Meet and Greet data

Our first step is to create the get the csv and welcome it. Later we should dissect and perform descriptive analyis. Well that escalated quickly.

Dataset:

We have 4118 instances and 21 features. The information says there are no null values. Fishy right? anyway we will strictly scrutinize each feature and check for suspicious records and manipulate them

Attributes: Bank client data:

1. **Age** : Age of the lead (numeric)
2. **Job** : type of job (Categorical)
3. **Marital** : Marital status (Categorical)
4. **Education** : Educational Qualification of the lead (Categorical)
5. **Default:** Does the lead has any default(unpaid)credit (Categorical)
6. **Housing:** Does the lead has any housing loan? (Categorical)
7. **loan:** Does the lead has any personal loan? (Categorical)

Related with the last contact of the current campaign:

8. **Contact:** Contact communication type (Categorical)
9. **Month:** last contact month of year (Categorical) 10. **day_of_week:** last contact day of the week (categorical)
11. **duration:** last contact duration, in seconds (numeric).

Important note: Duration highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model.

Other attributes:

12. **campaign:** number of contacts performed during this campaign and for this client (numeric)
13. **pdays:** number of days that passed by after the client was last contacted from a previous campaign(numeric; 999 means client was not previously contacted))
14. **previous:** number of contacts performed before this campaign and for this client (numeric)
15. **poutcome:** outcome of the previous marketing campaign (categorical)

Social and economic context attributes

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 variable (desired target):

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

```
job              object marital           object
education          object default           object
housing            object loan              object
contact            object month             object
day_of_week        object duration
int64 campaign            int64 pdays
int64 previous            int64 poutcome
object emp.var.rate      float64
cons.price.idx    float64 cons.conf.idx
float64 euribor3m         float64 nr.employed
float64 y                 object
dtype: object
```

In [5]:

```
#Checking out the statistical parameters
bank_copy.describe()
```

Out[5]:

	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed
count	41188.00000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000	41188.000000
mean	40.02406	258.285010	2.567593	962.475454	0.172963	0.081886	93.575664	-40.502600	3.621291	5167.035911
std	10.42125	259.279249	2.770014	186.910907	0.494901	1.570960	0.578840	4.628198	1.734447	72.251528
min	17.00000	0.000000	1.000000	0.000000	0.000000	-3.400000	92.201000	-50.800000	0.634000	4963.600000
25%	32.00000	102.000000	1.000000	999.000000	0.000000	-1.800000	93.075000	-42.700000	1.344000	5099.100000
50%	38.00000	180.000000	2.000000	999.000000	0.000000	1.100000	93.749000	-41.800000	4.857000	5191.000000
75%	47.00000	319.000000	3.000000	999.000000	0.000000	1.400000	93.994000	-36.400000	4.961000	5228.100000
max	98.00000	4918.000000	56.000000	999.000000	7.000000	1.400000	94.767000	-26.900000	5.045000	5228.100000

Insights:

- We got unknown category in each feature, we should figure out how to deal with that

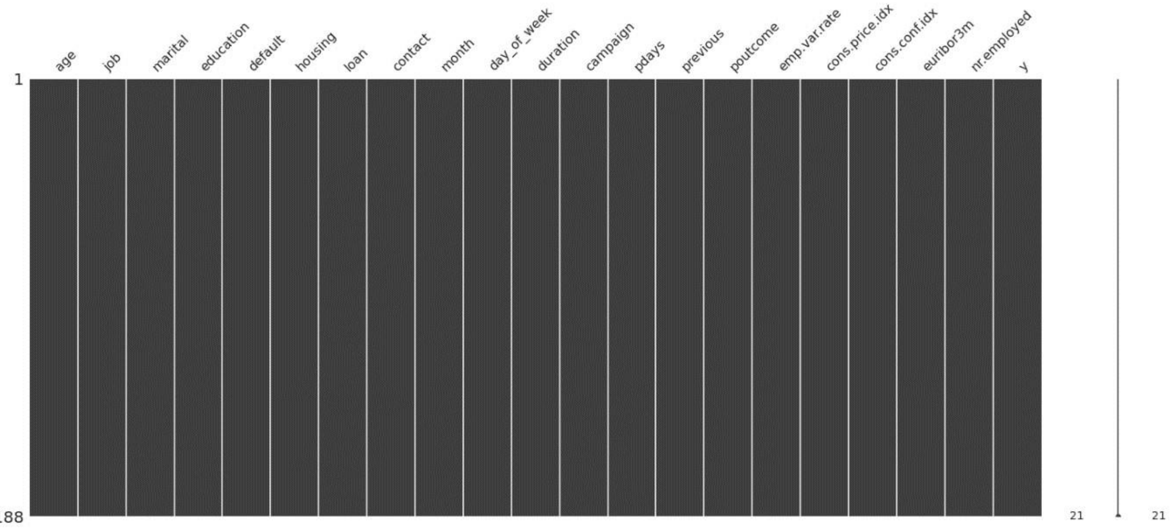
- This campaign only operated during weekdays
- I can't understand what is non-existent category in previous outcome aka poutcome, if you have figured out what is it let me know in the comments.

Data Cleaning

Checking for missing values
First lets check it visually

import missingno **as** msno msno**.**matrix(bank_copy)

<matplotlib.axes._subplots.AxesSubplot at 0x7fc08fb15e50>



Looks like we don't have any null values except one. But plots sometimes deceive us, numbers don't. Let's check with the numbers

print('Data columns with null values:',bank_copy**.**isnull()**.**sum(), sep = '\n')

```
Data columns with null values:
age              0 job              0
marital          0 education
0 default          0 housing
0 loan             0 contact
0 month            0
day_of_week      0 duration
0 campaign         0 pdays
0 previous         0 poutcome
0 emp.var.rate     0
cons.price.idx   0
cons.conf.idx    0 euribor3m
```

```
0 nr.employed       0 y
0 dtype: int64
```

We have the records of null values and looks like **we don't have any null values.**

DATA VISUALIZATION

Since we have much numerical data, let's keep our plots much targetted towards our machine learning models. Also let's figure out which feature importances and prune away least important ones

Duration of calls vs Job roles

```
import plotly.express as px

fig = px.box(bank_copy, x="job", y="duration", color="y")
fig.update_traces(quartilemethod="exclusive") # or "inclusive", or "linear" by default
fig.show()
```

Insights:

- The leads who have not made a deposit have lesser duration on calls
- Comparing the average, the blue collar, entrepreneur have high duration in calls and student, retired have less duration in average
- Large distribution of leads were from self employed clients and management people.

Campaign vs Duration calls

```
fig = px.scatter(bank_copy, x="campaign", y="duration", color="y") fig.show()
```

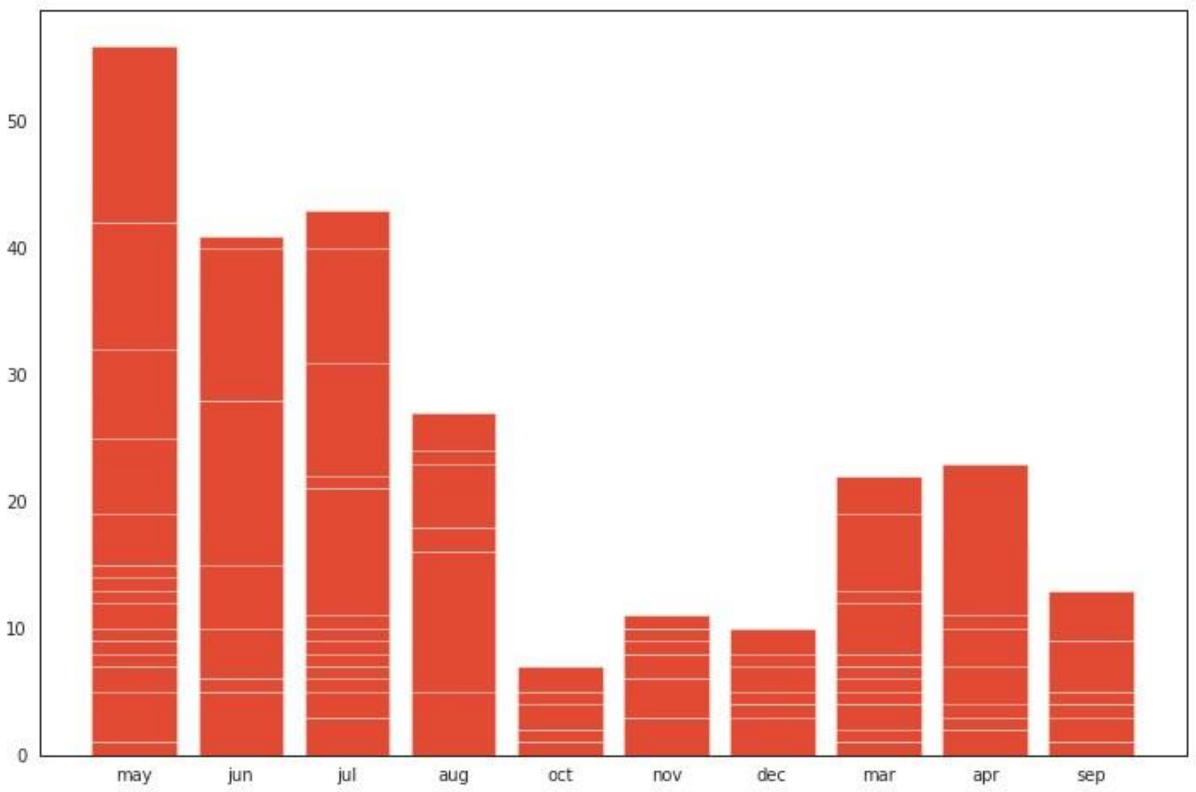
Insights:

- The more the duration the calls were, they had higher probability in making a deposit
- Duration of calls faded as the time period of campaign extended further
- There were many positive leads in the initial days of campaign

Campaign vs Month

```
plt.bar(bank_copy['month'], bank_copy['campaign'])
```

```
<BarContainer object of 41188 artists>
```



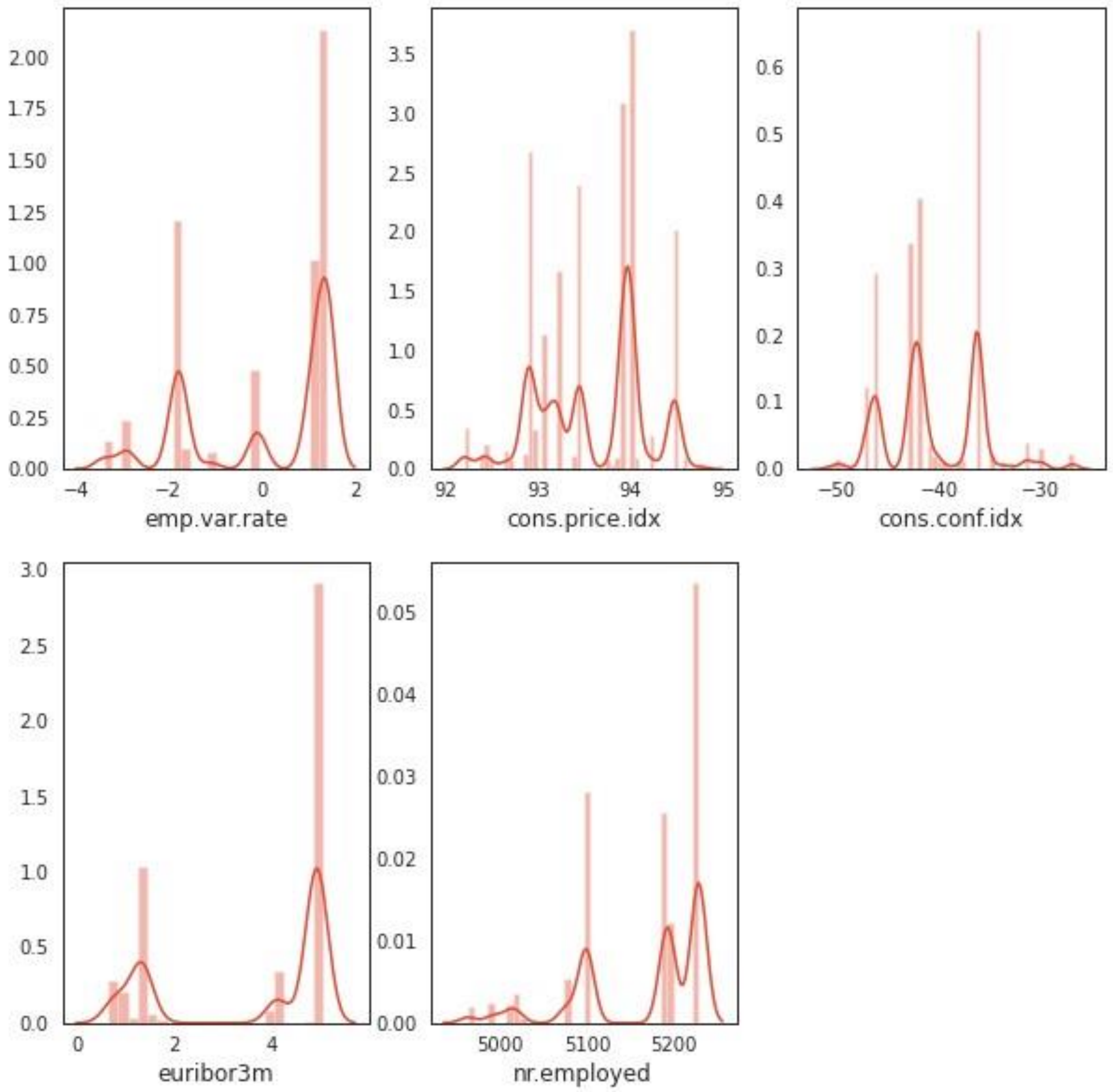
Insights:

- We can see the campaign were mostly concentrated in the starting of the bank period (May, June and July)
- Usually education period starts during that time so there is a possibility that parents make deposits in the name of their children
- They also have made their campaign in the end of the bank period.

Distribution of Quarterly Indicators

```
plt.subplot(231) sns.distplot(bank_copy['emp.var.rate'])
fig = plt.gcf()
fig.set_size_inches(10,10)
 plt.subplot(232)
sns.distplot(bank_copy['cons.price.idx']) fig =
plt.gcf()
fig.set_size_inches(10,10)
 plt.subplot(233)
sns.distplot(bank_copy['cons.conf.idx']) fig =
plt.gcf()
fig.set_size_inches(10,10)
```

```
 plt.subplot(234)
sns.distplot(bank_copy['euribor3m']) fig =
plt.gcf()
fig.set_size_inches(10,10)
 plt.subplot(235)
sns.distplot(bank_copy['nr.employed']) fig =
plt.gcf() fig.set_size_inches(10,10)
```



Insights:

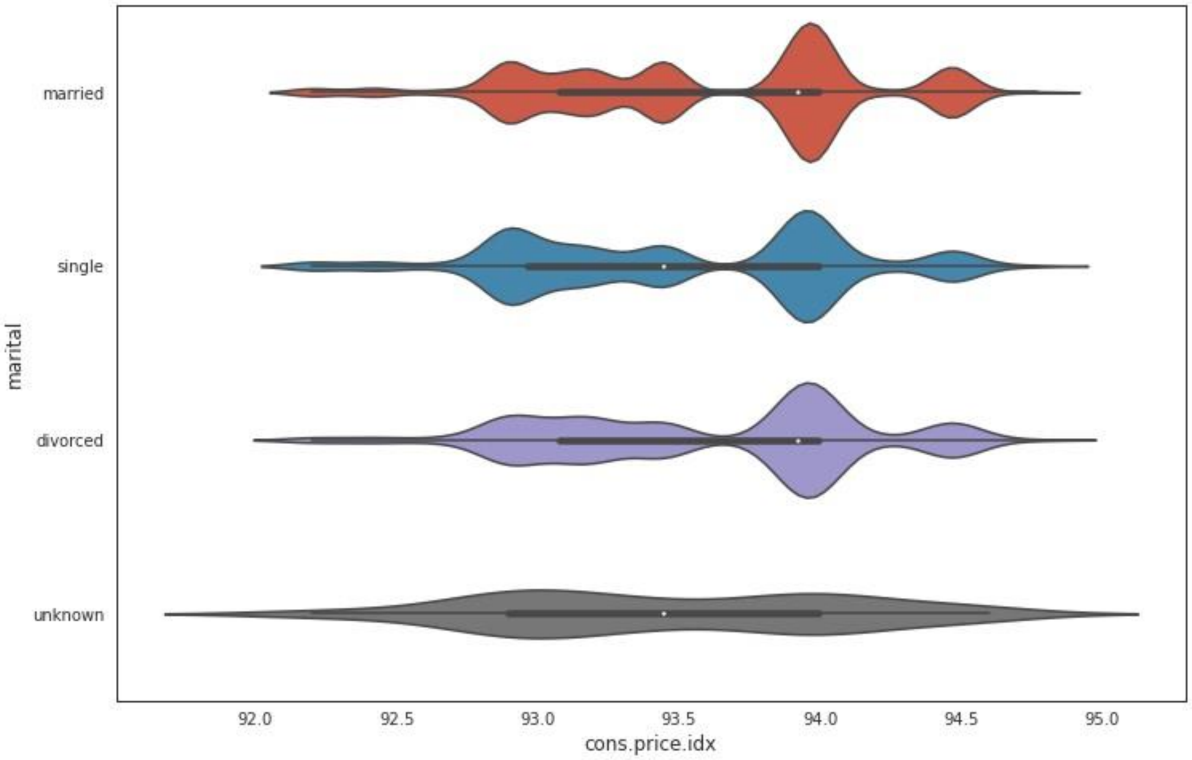
- We can see there is a high employee variation rate which signifies that they have made the campaign when there were high shifts in job due to conditions of economy
- The Consumer price index is also good which shows the leads where having good price to pay for goods and services may be that could be the reason to stimulate these leads into making a deposit and plant the idea of savings

- Consumer confidence index is pretty low as they don't have much confidence on the fluctuating economy
- The 3 month Euribor interest rate is the interest rate at which a selection of European banks lend one another funds denominated in euros whereby the loans have a maturity of 3 months. In our case the interest rates are high for lending their loans
- The number of employees were also at peak which can increase their income index that could be the reason the campaign targetted the leads who were employeed to make a deposit

Marital Status vs Price index

sns.violinplot(y=bank_copy["marital"], x=bank_copy["cons.price.idx"])

<matplotlib.axes._subplots.AxesSubplot at 0x7fc06cc4add0>



Insights:

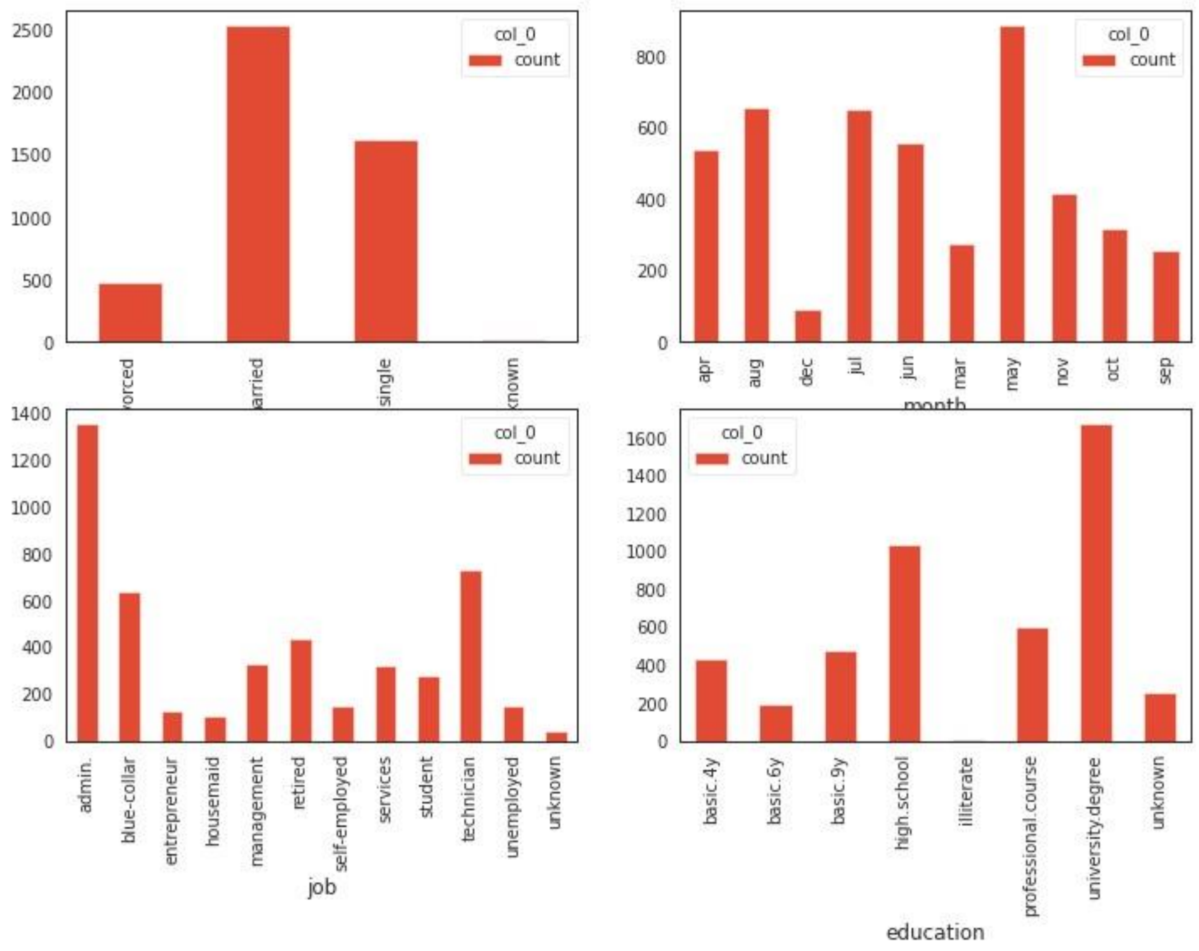
- There are very minute differences among the price index
- Married leads have considerably have an upper hand as they have index contributing as couple

Positive deposits vs attributes

bank_yes = bank_copy[bank_copy['y']=='yes'] df1 = pd.crosstab(index = bank_yes["marital"],columns="count") df2 = pd.crosstab(index = bank_yes["month"],columns="count") df3= pd.crosstab(index =

```
bank_yes["job"],columns="count")  df4=pd.crosstab(index =
bank_yes["education"],columns="count")
 fig, axes = plt.subplots(nrows=2, ncols=2)
df1.plot.bar(ax=axes[0,0]) df2.plot.bar(ax=axes[0,1])
df3.plot.bar(ax=axes[1,0]) df4.plot.bar(ax=axes[1,1])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc06a9b9f90>
```

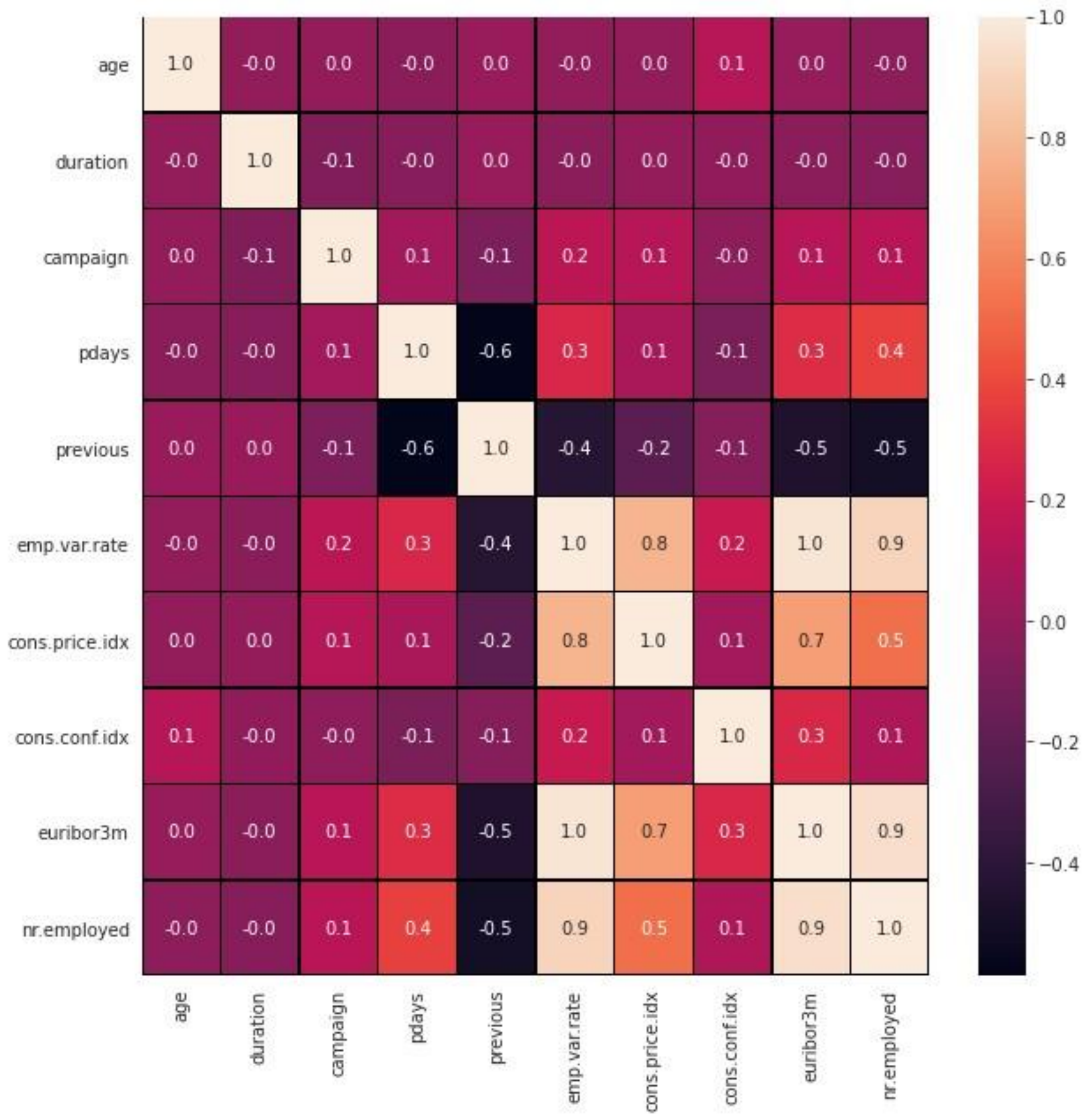


Insights:

- Married leads have made high deposits followed by single
- There were much deposist made during may month as it is the start of bank period
- Leads who work in administrative position made deposits followed by technicians and blue collar employees
- Leads who had atleast university degree had made te deposits followed by highschool

Correlation plot of attributes

```
f,ax=plt.subplots(figsize=(10,10))
sns.heatmap(bank_copy.corr(),annot=True,linewidths=0.5,linecolor="black",fm t=".1f",ax=ax)
plt.show()
```



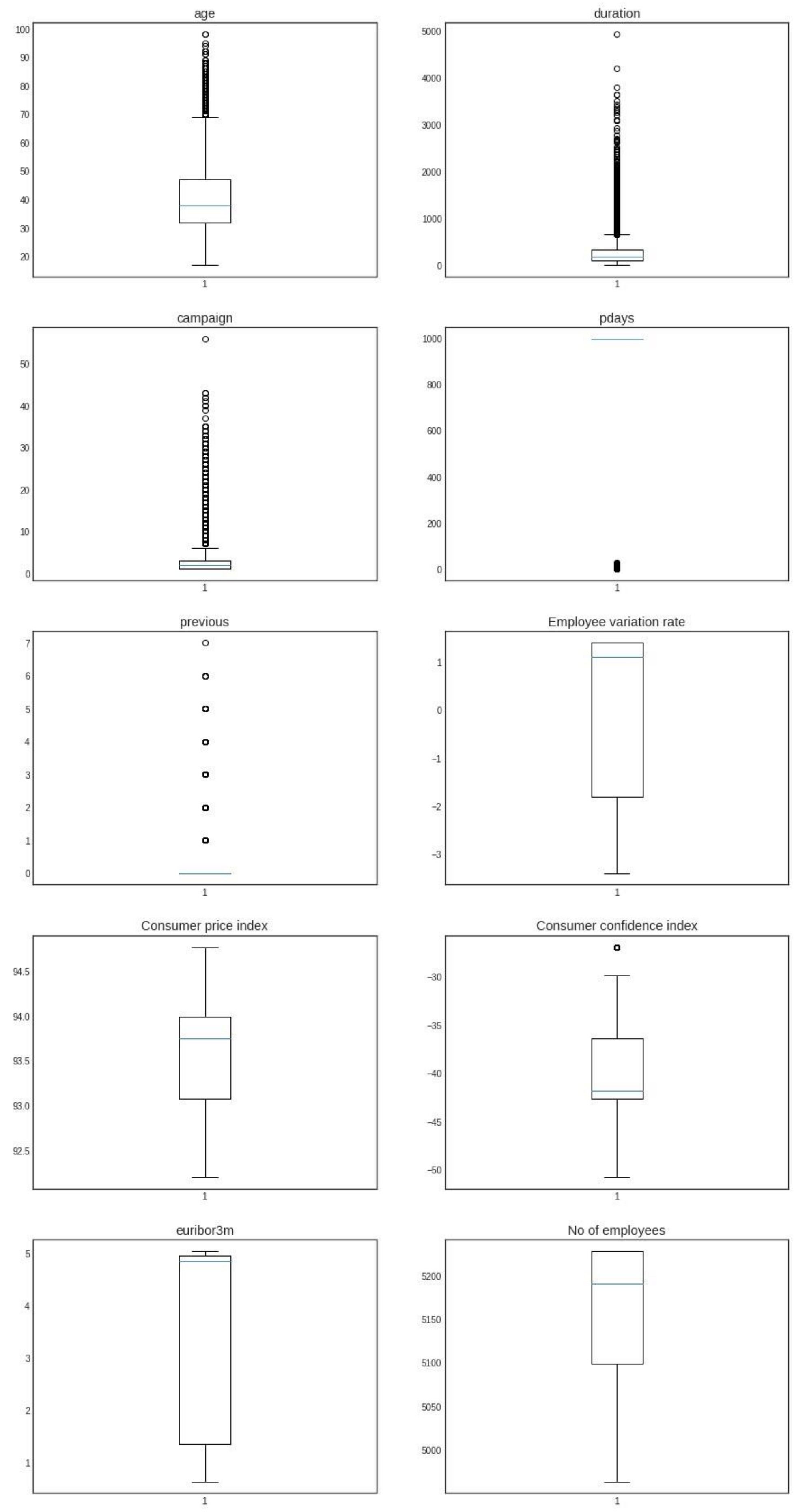
Insights:

- The indicators have correlation among themselves
- Number of employees rate is highly correlated with employee variation rate
- Consumer price index is highly correlated with bank interest rate(higher the price index, higher the interest rate)
- Employee variation rate also correlates with the bank interest rates

FEATURE ENGINEERING

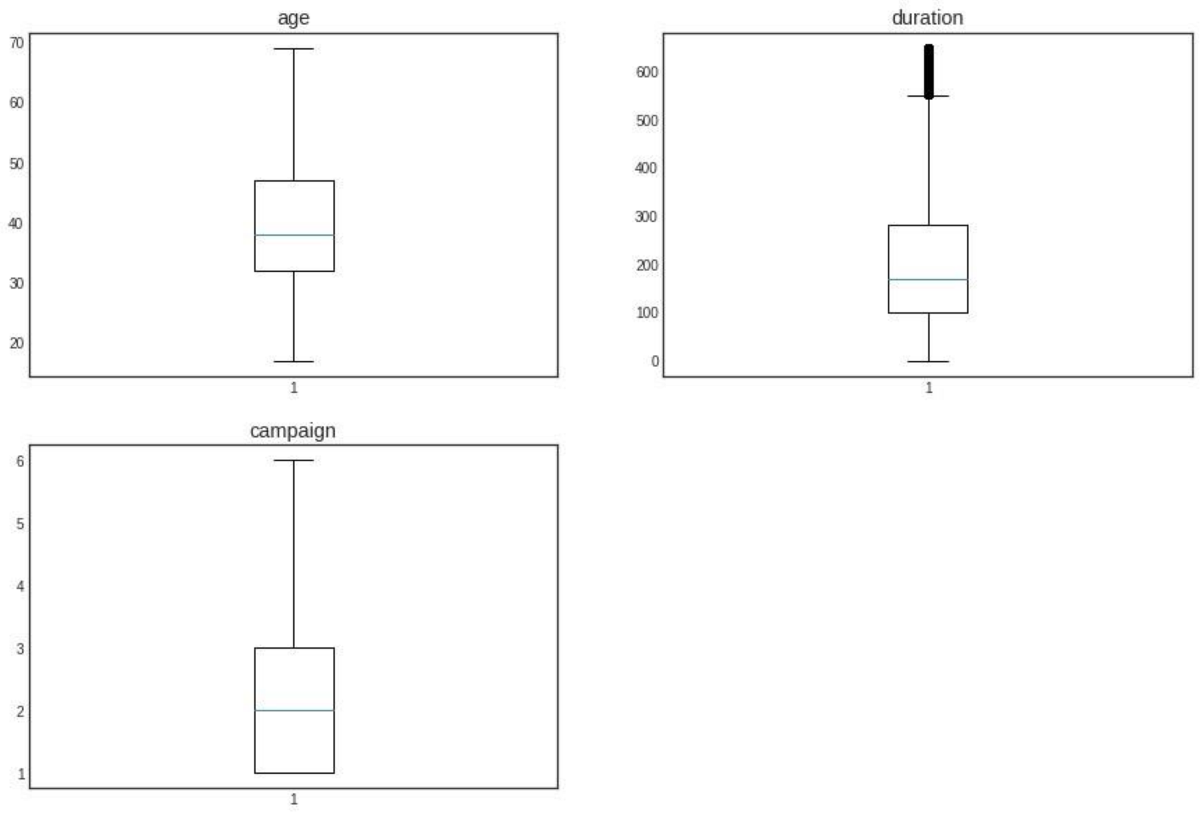
Handling outliers

Let's check out our numerical feature outliers through boxplot


age
duration
campaign
pdays
previous
Employee variation rate
Consumer price index
Consumer confidence index
euribor3m
No of employees

We see that many features doesn't have much outliers except for age,duration and campaign. So, let's fix only those features using IQR method.

```
Text(0.5, 1.0, 'campaign')
```



Now that we have removed outliers, we can proceed for more feature engineering techniques.

Education- category clubbing

Here we are clubbing category in education such as 'basic.9y','basic.6y','basic.4y' to 'middle school'

In [19]:

```
bank_features=bank_copy.copy() lst=['basic.9y','basic.6y','basic.4y'] for i in lst:
bank_features.loc[bank_features['education'] == i, 'education'] =
"middle.school"

bank_features['education'].value_counts()
```

Out[19]:

```
middle.school         10688 university.degree
10559 high.school            8287
professional.course     4554 unknown
1459 illiterate             14
Name: education, dtype: int64
```

Great, we have clubbed all the categories in education into one

Encoding - Month and Day of week

Encoding the categories in month and day of week to the respective numbers.

In [20]:

```
month_dict={'may':5,'jul':7,'aug':8,'jun':6,'nov':11,'apr':4,'oct':10,'sep'
:9,'mar':3,'dec':12} bank_features['month']=
bank_features['month'].map(month_dict)
 day_dict={'thu':5,'mon':2,'wed':4,'tue':3,'fri':6}
bank_features['day_of_week']= bank_features['day_of_week'].map(day_dict)
```

In [21]:

```
bank_features.loc[:, ['month', 'day_of_week']].head()
```

Out[21]:

	month	day_of_week
0	5	2
1	5	2
2	5	2
3	5	2
4	5	2

We have hard encoded the month and day of week features

Encoding 999 in pdays as 0

Encoding 999 in pdays feature(i.e clients who haven't been contacted for the previous campaign) into 0

In [22]:

```
bank_features.loc[bank_features['pdays'] == 999, 'pdays'] = 0
```

In [23]:

```
bank_features['pdays'].value_counts()
```

Out[23]:

```
0    34305
3      367
6      343
```

```
4       105
9        54
2        51
12       50
7        48
10       44
5        38
13       28
1        23
11       22
15       20
14       15
8        14
16             10
17             8
18             6
22       3
21       2
25       1
19             1
20             1
26             1
27             1
Name: pdays, dtype: int64
```

We have converted 999 to 0 in pdays

Ordinal Number Encoding

Here we are gonna encode the features which has yes,no and unknown. We'll assign yes:1,no:0 and unknown:-1

In [24]:

```
dictionary={'yes':1,'no':0,'unknown':-1}
bank_features['housing']=bank_features['housing'].map(dictionary)
bank_features['default']=bank_features['default'].map(dictionary)
bank_features['loan']=bank_features['loan'].map(dictionary)
```

In [25]:

```
dictionary1={'no':0,'yes':1}
bank_features['y']=bank_features['y'].map(dictionary1)
```

In [26]:

```
bank_features.loc[:,['housing','default','loan','y']].head()
```

```
Out[26]: housing   default        loan     y

0          0   0        0        0

1          0   -1       0        0

2          1   0        0        0

3          0   0        0        0

4          0   0        1        0
```

We have encoded the yes/no features with hard encoding

Ordinal Encoding

In [27]:

```
dummy_contact=pd.get_dummies(bank_features['contact'],
prefix='dummy',drop_first=True)
dummy_outcome=pd.get_dummies(bank_features['poutcome'],
prefix='dummy',drop_first=True) bank_features =
pd.concat([bank_features,dummy_contact,dummy_outcome],axis=1)
bank_features.drop(['contact','poutcome'],axis=1, inplace=True)
```

In [28]:

```
bank_features.loc[:,['dummy_telephone','dummy_nonexistent','dummy_success'] ].head()
```

Out[28]:

```
     dummy_telephone    dummy_nonexistent    dummy_success

0                    1      1        0

1                    1      1        0

2                    1      1        0

3                    1      1        0
```

```
4                    1      1      0
```

We have performed one-hot encoding for the above features and dropped the original features

Frequency encoding

Let's use frequency encoding with job and education features in our dataset

In [29]:

```
bank_job=bank_features['job'].value_counts().to_dict()
bank_ed=bank_features['education'].value_counts().to_dict()
```

Converted the frequency into key value pairs. Let's map them

In [30]:

```
bank_features['job']=bank_features['job'].map(bank_job)
bank_features['education']=bank_features['education'].map(bank_ed)
```

In [31]:

```
bank_features.loc[:,['job','education']].head()
```

```
Out[31]: job    education

0      899      10688

       job   education

  1   3456        8287

  2   3456        8287

  3   9110       10688

  4   3456        8287
```

We have encoded the job and education feature based on its frequency

Target Guided Ordinal Encoding

Lets encode marital feature based on the target 'y' . First let's find the mean of target with respect to marital feature

In [32]:

```
bank_features.groupby(['marital'])['y'].mean()
```

Out[32]:

```
marital divorced    0.063988
married      0.069050 single
0.113226             unknown
0.129032
Name: y, dtype: float64
```

In [33]:

```
ordinal_labels=bank_features.groupby(['marital'])['y'].mean().sort_values()
.index ordinal_labels
```

Out[33]:

```
Index(['divorced', 'married', 'single', 'unknown'], dtype='object', name='m arital')
```

We have sorted the categories based on the mean with respect to our outcome

In [34]:

```
ordinal_labels2={k:i for i,k in enumerate(ordinal_labels,0)} ordinal_labels2
```

Out[34]:

```
{'divorced': 0, 'married': 1, 'single': 2, 'unknown': 3}
```

Changed into key:value pairs, let's map them

In [35]:

```
bank_features['marital_ordinal']=bank_features['marital'].map(ordinal_label s2)
bank_features.drop(['marital'], axis=1,inplace=True)
```

In [36]:

```
bank_features.marital_ordinal.value_counts()
```

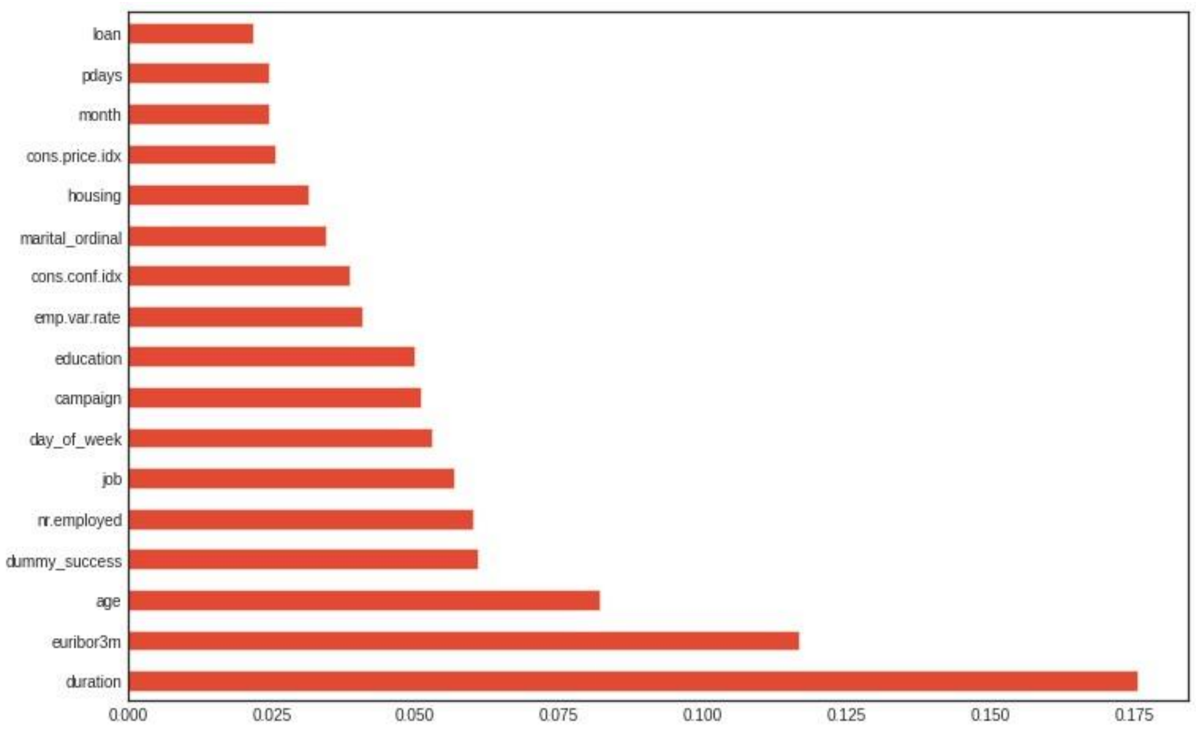
Out[36]:

```
1     21506
2     10086
0    3907
3      62
Name: marital_ordinal, dtype: int64
```

We have encoded the marital feature

Feature Selection

Let's check the feature importances and prune our features to make our model perform well.



From the bar plot we can see the importances of features based on it's impact towards output. Let's take up the top 15 features

Modelling our Data

Let's enter into the crucial phase of building THE machine learning model. Before checking "what could be the best algorithm for prediction" we have to decide on the "why". It is highly important.

Why?

Our main aim is to predict whether there is a deposit made made owing to those values from the features. The output is either going to be 0 or 1. So we can decide that we can use classification models for our problem

What ?

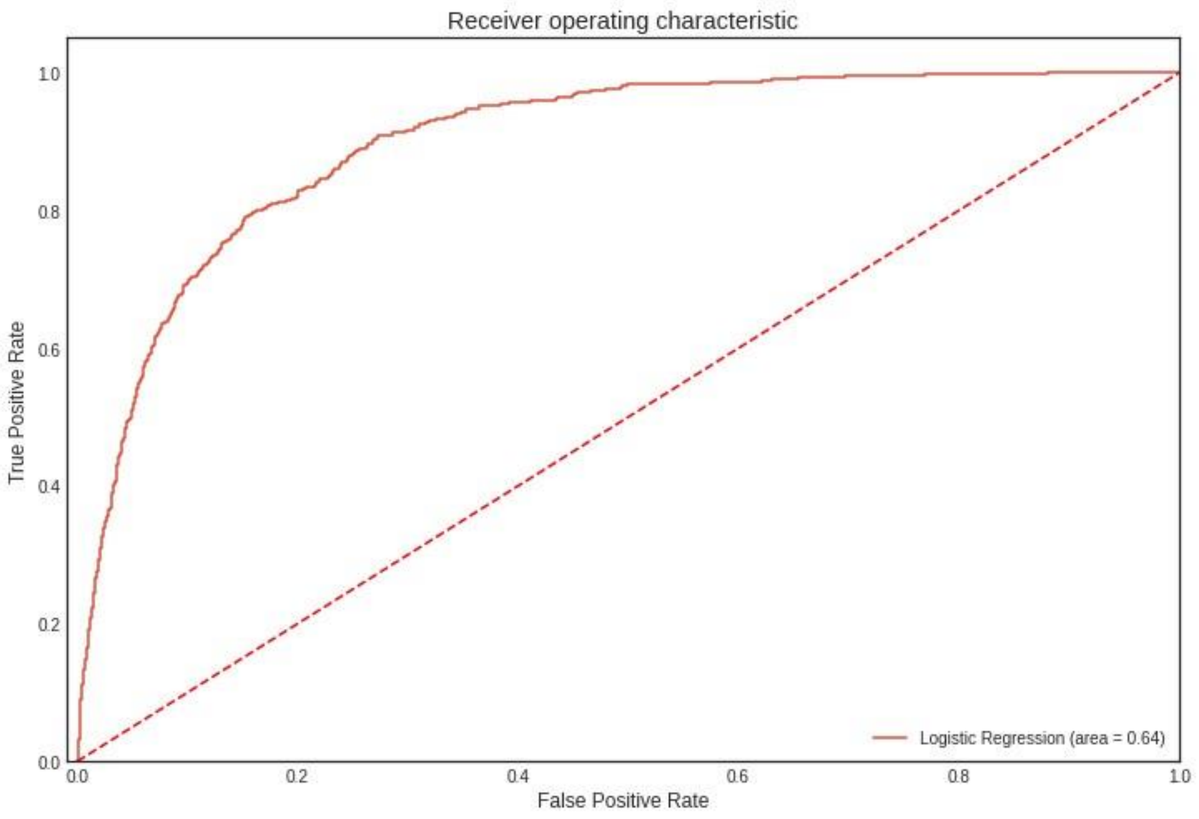
To decide on what can be the best possible classification models let's not waste time running models. Instead we do quality code by creating cross validation and check all the model accuracy at once. After that we will select one model based on it's accuracy.

Insights:

- The Confusion matrix result is telling us that we have **6399+178** correct predictions and **397+139** incorrect predictions.
- The Classification report reveals that we have **94%** precision which means the accuracy that the model classifier not to label an instance positive that is actually negative which is important as we shouldn't label a lead as positive in making a term deposit when he/she isn't interested in making a deposit

ROC Curve

Let's check out the performance of our model through ROC curve



From the ROC curve we can infer that our logistic model has classified the prospective leads who made deposit correctly rather than predicting false positive. The more the ROC curve(red) lies towards the top left side the better our model is. We can choose any value between **0.8 to 0.9** for the threshold value which can reap us true positive results

Support vector classifier

Let's fit our best model from model selection and predict outcome.

CONCLUSION

From the EDA and model selection part we can clearly identify duration playing an important attribute in defining the outcome of our dataset. It is absolute that the more the leads are interested in starting a deposit will have higher number of calls and the call duration will be higher than the average. We have also figured out that job and education also acts as a crucial deciding factor and influences the outcome alot.

Here are the few recommendations for the bank than can help improve the deposit rate

- Classify job roles based on corporate tiers and approach all tier 1 employees within few days after the campaign commences
- Listen to the leads and extract more information to deliver the best deposit plan, which can increase the duration of calls and that can lead to a deposit
- Approaching the leads during the start of new bank period(May-July) will be a good choice as many have shown positive results from data history
- Tune the campaign according to the national econometrics, don't chanelize the expenses on campaign when the national economy is performing poor