CUSTOMER CHURN PREDICTION

GROUP: 1

- KRITI BHARDWAJ
- PRAKSHITA AGRAWAL
- ANIL KUMAR BARIK
- ANJALI

TABLE OF CONTENTS

INTRODUCTION

Churn prediction is most common use case in machine learning domain. Churn means "leaving the company", It is very critical for business to have an idea about why and when customers are likely to churn. Having a robust and accurate churn prediction model helps businesses to take actions to prevent customers from leaving the company.

OUR MAIN OBJECTIVE

- Analyse the data and identify the main indicators of churn.
- Build a predictive model to predict customer churn.
- Find the best model based on the model performances.
- Model optimisation using GridSearchCV method.
- Predict “churn” for the given test dataset

DATA DESCRIPTION

- Each row and column of the Customer Churn Dataset represents a customer and customer’s attributes respectively.
- The "Customer churn" dataset contains 4250 rows (customers) and 20 columns (features). The "churn" column is the target to predict.

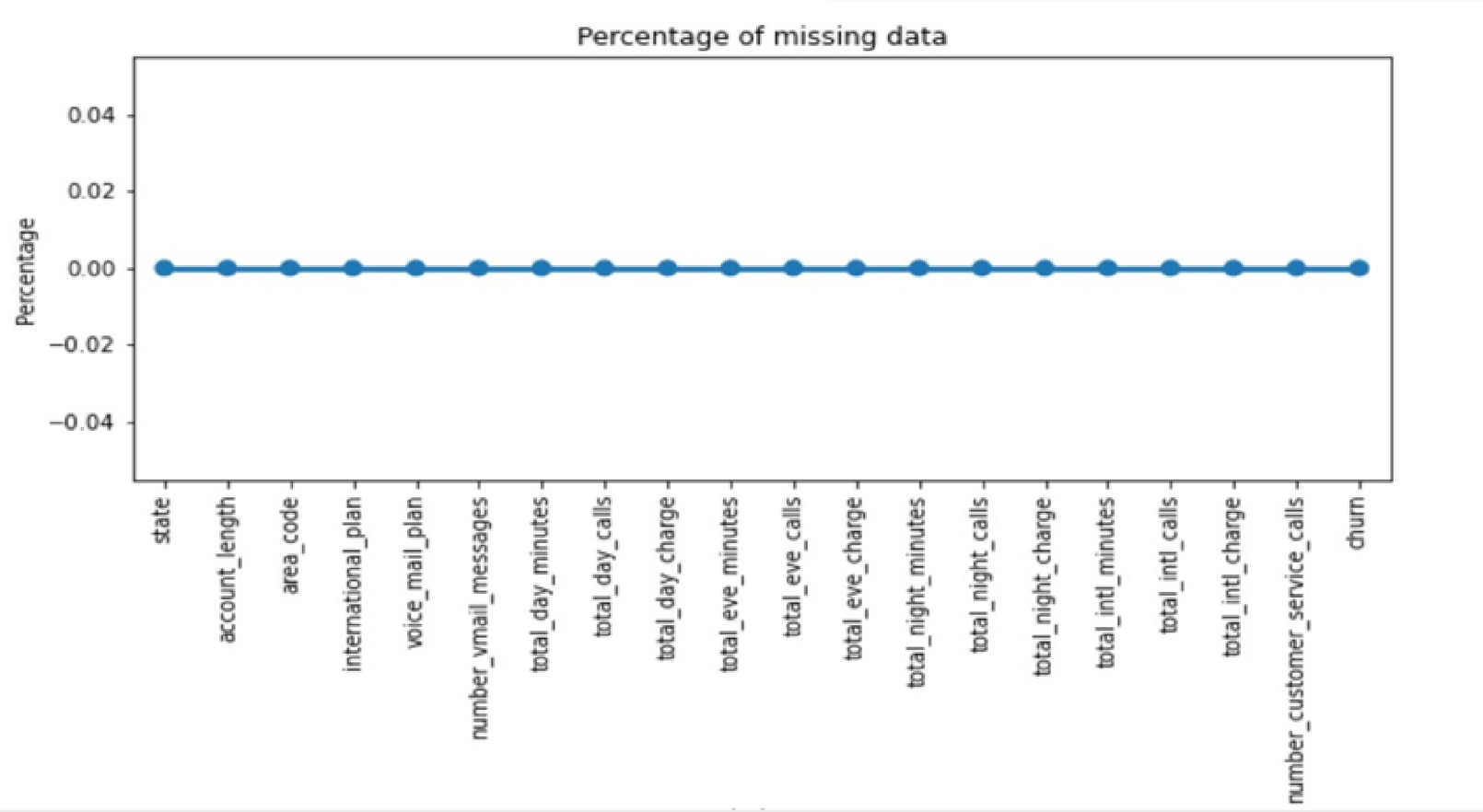
- state: string
- account_length: integer
- area_code: integer
- international_plan: string
- voice_mail_plan: string
- number_vmail_messages: integer
- total_day_minutes: float
- total_day_calls: integer
- total_day_charge: float
- total_eve_minutes: float
- total_eve_calls: integer
- total_eve_charge: float
- total_night_minutes: float
- total_night_calls: integer
- total_night_charge: float
- total_intl_minutes: float
- total_intl_calls: integer
- total_intl_charge: float
- customer_service_calls: integer
- churn: string

EXPLORATORY DATA ANALYSIS

- A detailed analysis and pre-processing are done in the dataset. It gave us a better idea of contribution of features towards the target variable.

1. Checking for NaN values:

No missing or NaN values in the dataset



2. Statistical Data:

```
df.describe()
```

	account_length	number_vmail_messages	total_day_minutes	total_day_calls	total_day_charge	total_eve_minutes	total_eve_calls	total_eve_charge
count	4250.000000	4250.000000	4250.000000	4250.000000	4250.000000	4250.000000	4250.000000	4250.000000
mean	100.236235	7.631765	180.259600	99.907294	30.644682	200.173906	100.176471	17.015012
std	39.698401	13.439882	54.012373	19.850817	9.182096	50.249518	19.908591	4.271212
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	73.000000	0.000000	143.325000	87.000000	24.365000	165.925000	87.000000	14.102500
50%	100.000000	0.000000	180.450000	100.000000	30.680000	200.700000	100.000000	17.060000
75%	127.000000	16.000000	216.200000	113.000000	36.750000	233.775000	114.000000	19.867500
max	243.000000	52.000000	351.500000	165.000000	59.760000	359.300000	170.000000	30.540000

3. Target variable:

```
df['churn'].value_counts()/len(df['churn'])*100
```

```
no     85.929412
yes    14.070588
Name: churn, dtype: float64
```

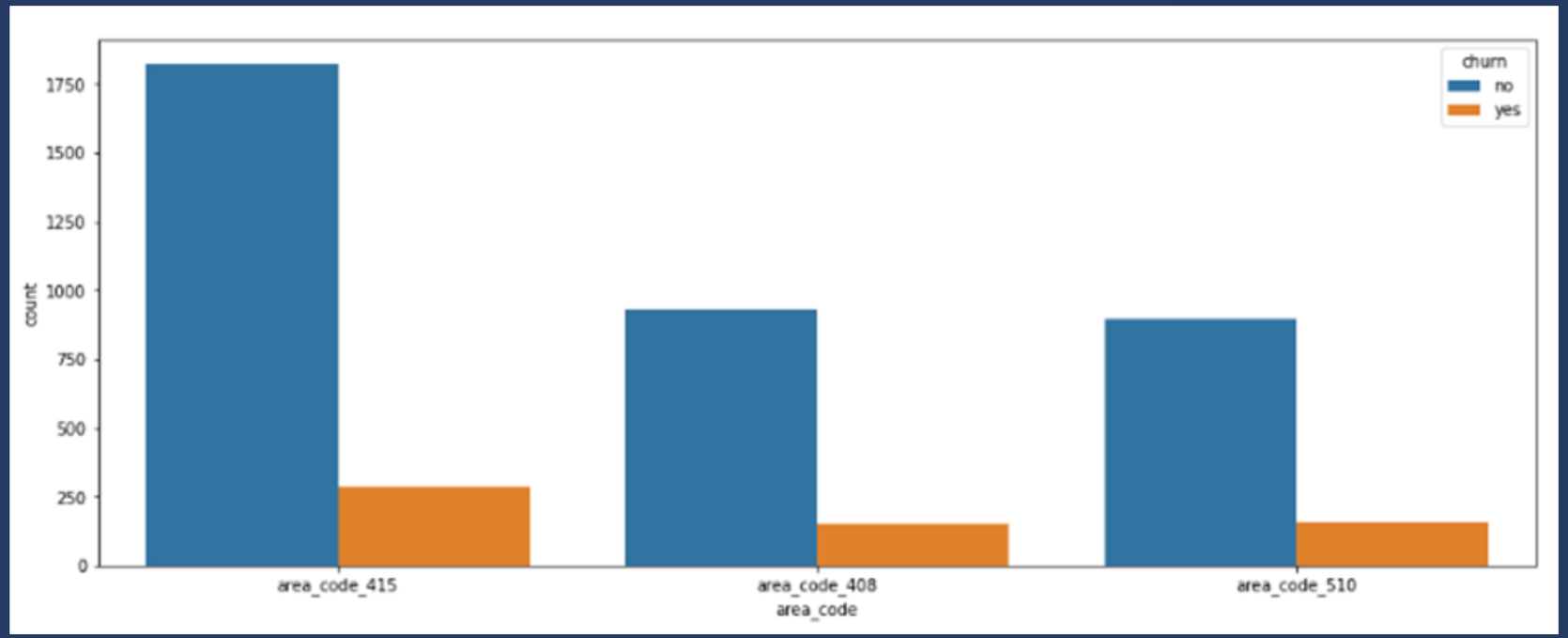
- Data is **imbalanced** since the ratio is 86:14

4. Univariate analysis:

4.1 Categorical columns: (area_code)

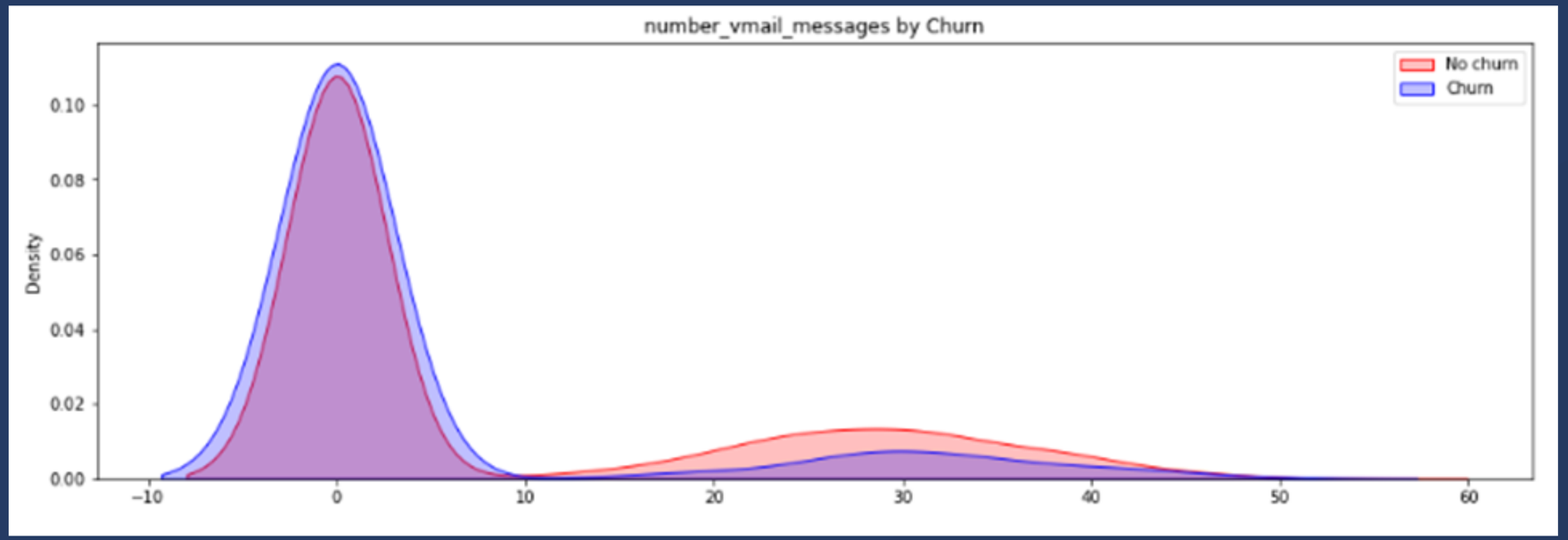
```
df[['area_code','churn']].groupby(['area_code']).mean()
```

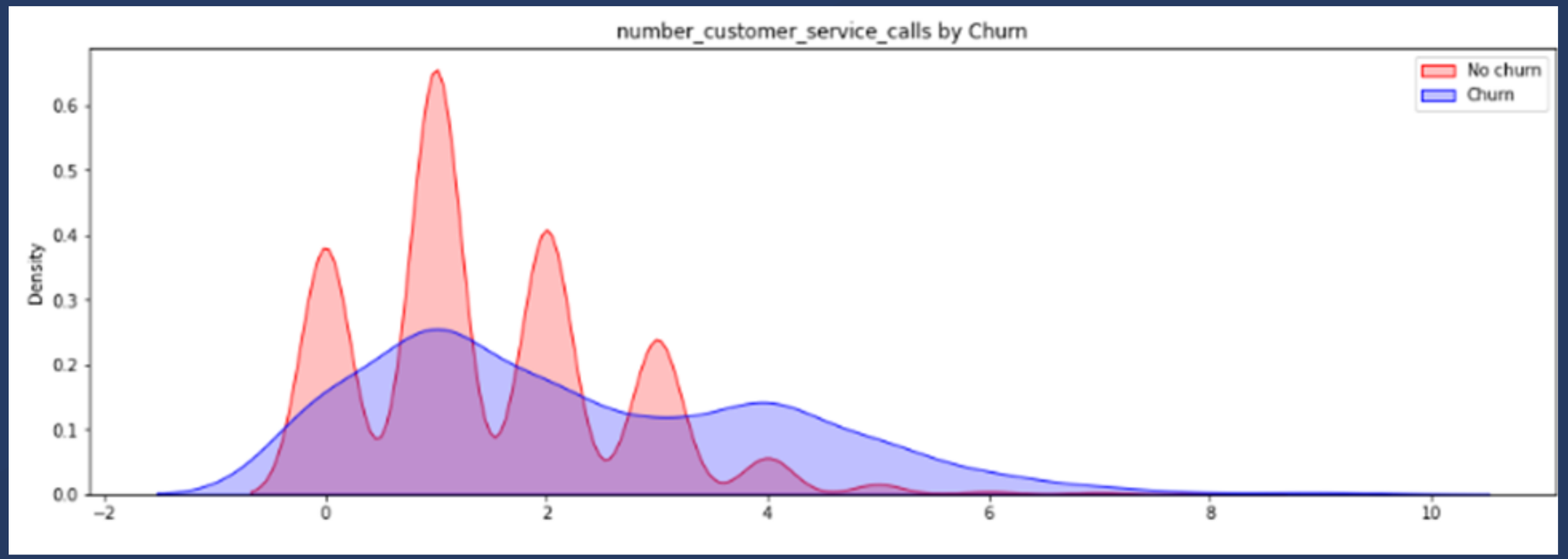
area_code	churn
area_code_408	0.139963
area_code_415	0.136148
area_code_510	0.150568



Similarly, other categorical columns are analysed using plot visualisation.

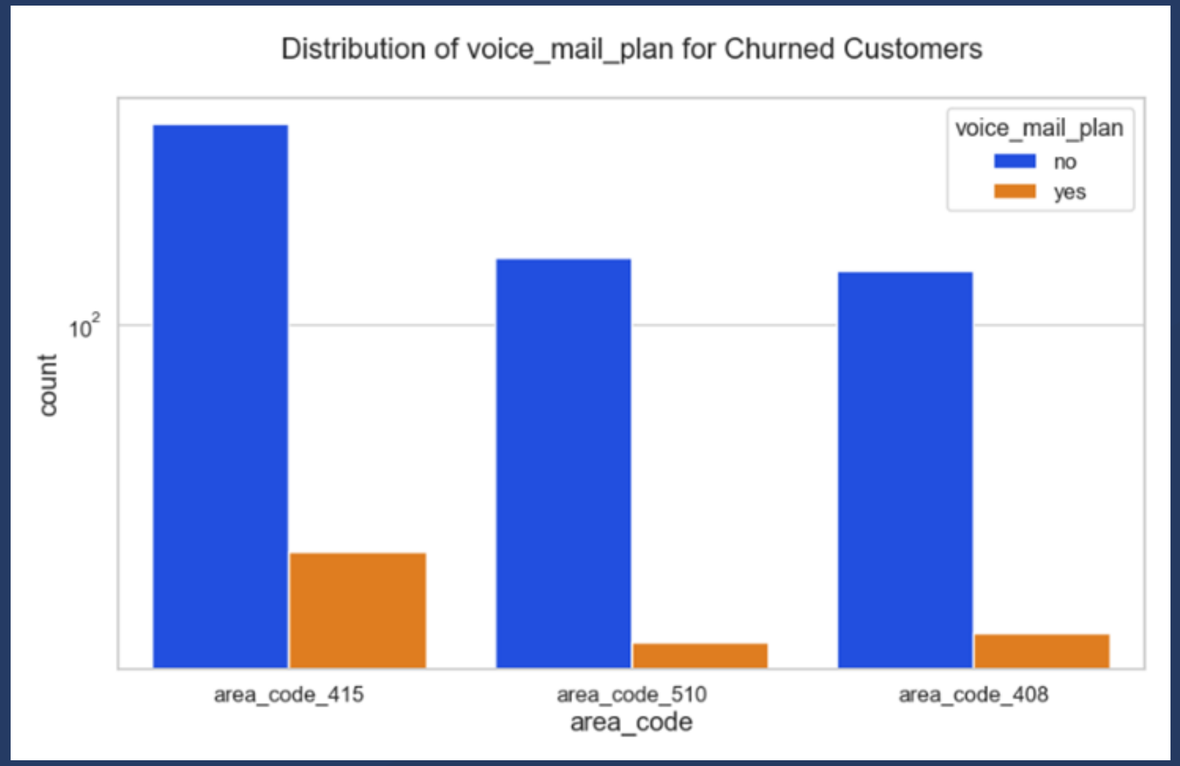
4.2 Numerical columns:

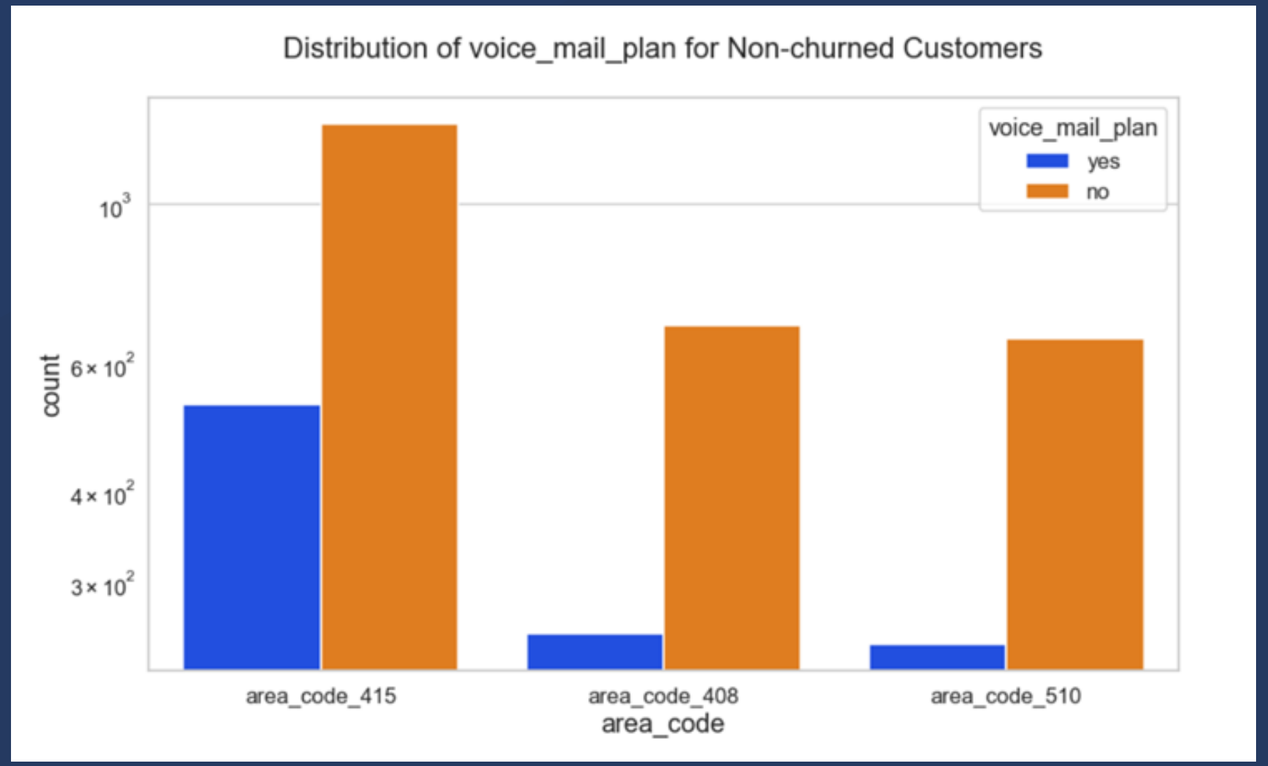




Similarly, other numerical columns are analysed using plot visualisation.

5. Bivariate analysis for categorical columns:



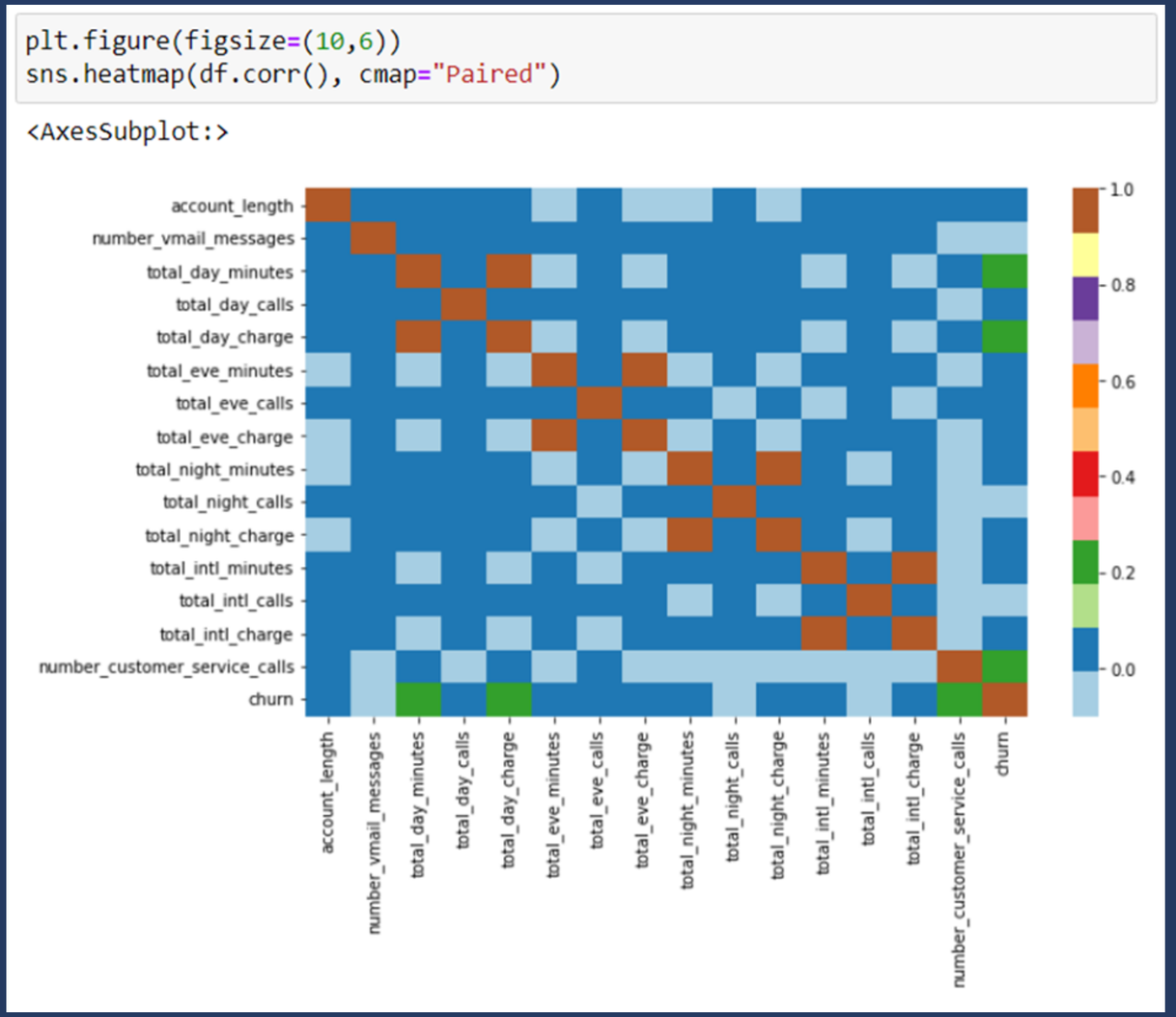


6. Correlation and Heatmap of all variables:

Heatmap gives a correlation matrix to quantify and summarize the relationships between the variables.

```
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(), cmap="Paired")
```

```
<AxesSubplot:>
```



DATA PRE-PROCESSING

Used **One Hot Encoding** to produce binary integers of 0 and 1 to encode our categorical features ‘churn’.

Performed similar data pre-processing in the test dataset too.

Dropped the “state” column as it is not contributing much to our target variable “churn”.

Created dummy variables for all categorical variables.

OBSERVATIONS AFTER EXPLORATORY DATA ANALYSIS

HIGH churning is observed with international_plan, number_customer_service_calls, total_day_minutes and total_day_charge .

LOW churning is observed with no international_plan, voice_mail_plan and number_vmail_messages

total_day_charge, total_day_minutes and number_customer_service_calls are contributing majorly to the target variable.

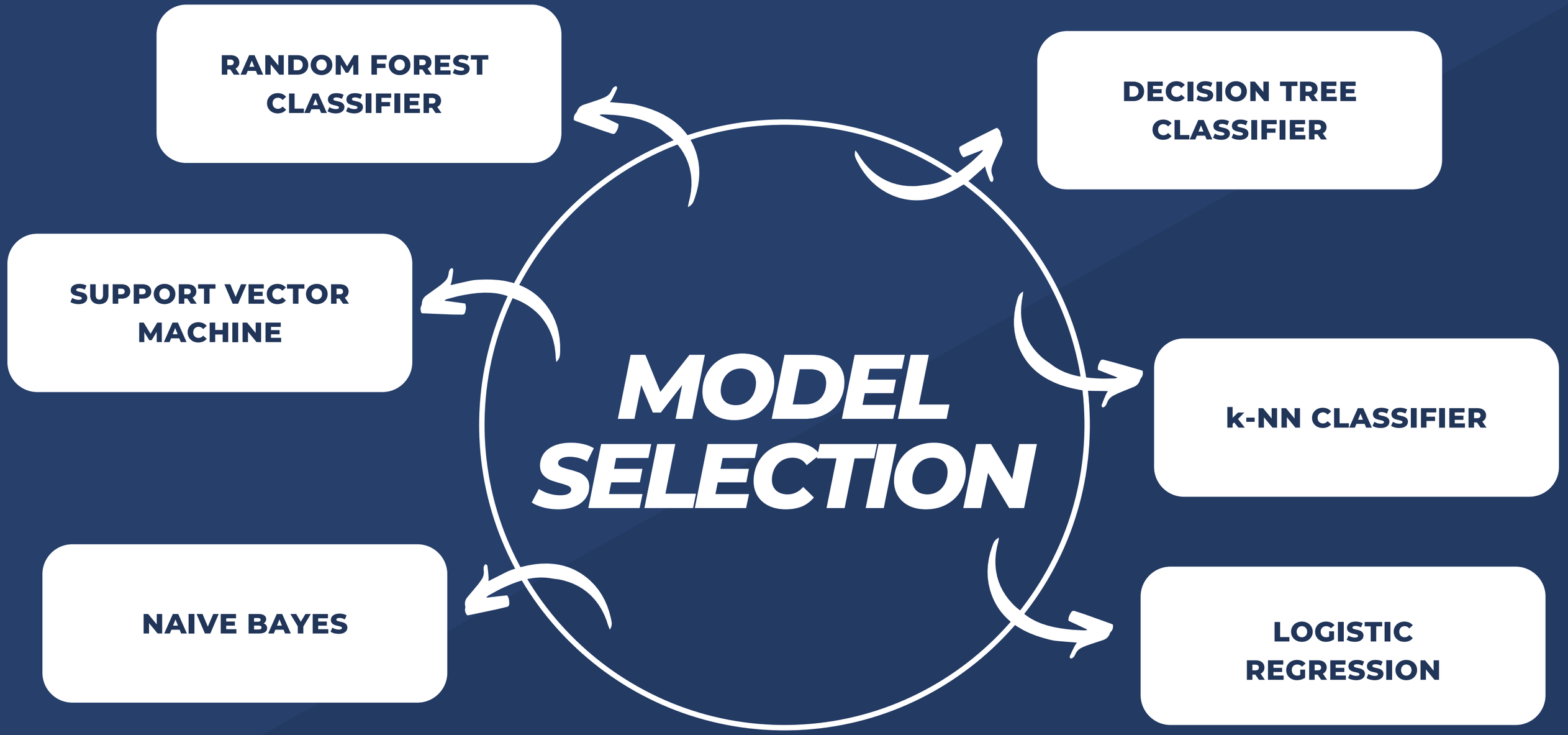
Features like area_code, total_day_calls, total_night_calls and total_eve_calls have almost NO impact on the target variable .

Test-train splitting for model training

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Performance metrics for classification models

```
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
```



After training and testing different classification models, the model with best performance will be selected as the final model.

MODEL TRAINING AND TESTING

```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
```

```
rf.score(X_train, y_train)*100 # training accuracy
```

```
99.97058823529412
```

```
rf.score(X_test, y_test)*100 # testing accuracy
```

```
96.47058823529412
```

```
print(classification_report(y_test, y_pred, labels=[0,1]))
```

	precision	recall	f1-score	support
0	0.96	1.00	0.98	721
1	0.98	0.78	0.87	129
accuracy			0.96	850
macro avg	0.97	0.89	0.93	850
weighted avg	0.97	0.96	0.96	850

(Random Forest Classifier)

- Similarly, other classification models are trained and tested.

MODEL PERFORMANCE SUMMARY

	Accuracy	Precision	Recall	f1_score
Random Forest Classifier	96.47%	0.98	0.79	0.87
Decision Tree Classifier	91.06%	0.70	0.73	0.71
Logistic Regression	85.89%	0.60	0.21	0.31
k-NN Classifier	87.53%	0.90	0.20	0.33
Naive Bayes	86.59%	0.56	0.53	0.54
Support Vector Machine	84.94%	1.00	0.01	0.02

FINAL MODEL AND ITS OPTIMIZATION

- After going through all the classification models, based on their performance measures, we conclude that **Random Forest Classifier** is the **best model** for our dataset.
- Although the high accuracy of the Random Forest Classifier can also be because of the model overfitting, to avoid this, it is optimized using the **GridSearchCV** model optimization method.

HYPERPARAMETERS:

```
n_estimators = [20,60,100,120]
max_features = [0.2,0.6,1.0]
max_depth = [2,8,None]
max_samples = [0.5,0.75,1.0]
```

```
param_grid = {'n_estimators': n_estimators,
              'max_features': max_features,
              'max_depth': max_depth,
              'max_samples': max_samples
             }
print(param_grid)
```

```
rf_grid.best_params_
```

```
{'max_depth': 8, 'max_features': 0.6, 'max_samples': 0.75, 'n_estimators': 100}
```

```
rf_grid.best_score_
```

```
0.9555882352941175
```

BEST PARAMETERS AND BEST SCORE OF OPTIMISED MODEL:

PERFORMANCE MEASURES OF OPTIMISED MODEL:

```
accuracy_score(y_pred, y_test)*100
```

```
95.6470588235294
```

```
precision_score(y_test, y_pred)
```

```
0.9181818181818182
```

```
recall_score(y_test, y_pred)
```

```
0.7829457364341085
```

```
f1_score(y_test, y_pred)
```

```
0.8451882845188284
```

FUTURE WORK

- Customer churn is a major problem and one of the most important concerns for large companies. Due to the direct effect on the revenues of the companies, especially in the telecom field, companies are seeking to develop means to predict potential customer to churn.

- Our model is trained and tested with all the features that can affect customers' churn. Also, it is well optimized to give accurate predictions.

- A model like this would be very valuable and helpful for a company to predict its customers' churn and based on the reasons of churning, it can then make an attempt so that its customer won't churn and continue with the company happily.

THANK YOU!