Telecom Churn Case Study

Pratiksha Phapale

Pranit Roy

Pratap

Problem statement

- To reduce customer churn, telecom companies need to predict which customers are at high risk of churn.
- In this project, we will analyse customer-level data of a leading telecom firm, build predictive models to identify customers at high risk of churn and identify the main indicators of churn.
- Retaining high profitable customers is the main business goal here.
- Steps:- Reading, understanding and visualising the data Preparing the data for modelling Building the model Evaluate the model

Steps:-

1. Reading, understanding and visualising the data
2. Preparing the data for modelling
3. Building the model
4. Evaluate the model

Analysis Approach

- Telecommunications industry experiences an average of 15 - 25% annual churn rate. Given the fact that it costs 5 - 10 times more to acquire a new customer than to retain an existing one, customer retention has become even more important than customer acquisition.
- Here we are given with 4 months of data related to customer usage. In this case study, we analyse customer- level data of a leading telecom firm, build predictive models to identify customers at high risk of churn and identify the main indicators of churn.
- Churn is predicted using two approaches. Usage based churn and Revenue based churn
- This case study only considers usage-based churn. Usage based churn is Customers who have zero usage, either incoming or outgoing - in terms of calls, internet etc. over a period of time.
- In the Indian and the southeast Asian market, approximately 80% of revenue comes from the top 20% customers (called high-value customers). Thus, if we can reduce churn of the high-value customers, we will be able to reduce significant revenue leakage. Hence, this case study focuses on high value customers only.
- The dataset contains customer-level information for a span of four consecutive months - June, July, August and September. The months are encoded as 6, 7, 8 and 9, respectively.
- The business objective is to predict the churn in the last (i.e. the ninth) month using the data (features) from the first three months.
- This is a classification problem, where we need to predict whether the customers is about to churn or not. We have carried out Baseline Logistic Regression, then Logistic Regression with PCA, PCA + Random Forest.

Reading and understanding the data

Reading and Understanding the Data

In [3]:
```
telecom_data = pd.read_csv('telecom_churn_data.csv')
telecom_data.head()
```

Out[3]:

	mobile_number	circle_id	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou	last_date_of_month_6	last_date_of_month_7	last_date_of_month_8	last_date_of
0	7000842753	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	
1	7001865778	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	
2	7001625959	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	
3	7001204172	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	
4	7000142493	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	

In [4]:
```
telecom_data.shape
```

Out[4]: (99999, 226)

Handling the missing values

Handling missing values

Handling missing values in columns

In [7]:
```
telecom_data_missing_columns = telecom_data.isna().sum().sort_values(ascending=False).to_frame(name='Missing_count')
telecom_data_missing_columns['Missing_percentage'] = round((telecom_data_missing_columns['Missing_count'] / len(telecom_data.inde
telecom_data_missing_columns
```

Out[7]:

	Missing_count	Missing_percentage
arpu_3g_6	74846	74.85
night_pck_user_6	74846	74.85
total_rech_data_6	74846	74.85
arpu_2g_6	74846	74.85
max_rech_data_6	74846	74.85
...	...	...
max_rech_amt_7	0	0.00
max_rech_amt_6	0	0.00
total_rech_amt_9	0	0.00
total_rech_amt_8	0	0.00
sep_vbc_3g	0	0.00

226 rows × 2 columns

In [8]:
```
# List the columns having more than 30% missing values
col_list_missing_30 = list(telecom_data_missing_columns.index[telecom_data_missing_columns['Missing_percentage'] > 30])
```

In [9]:
```
# Delete the columns having more than 30% missing values
telecom_data = telecom_data.drop(col_list_missing_30, axis=1)
```

In [10]:
```
telecom_data.shape
```

Out[10]:
```
(99999, 186)
```

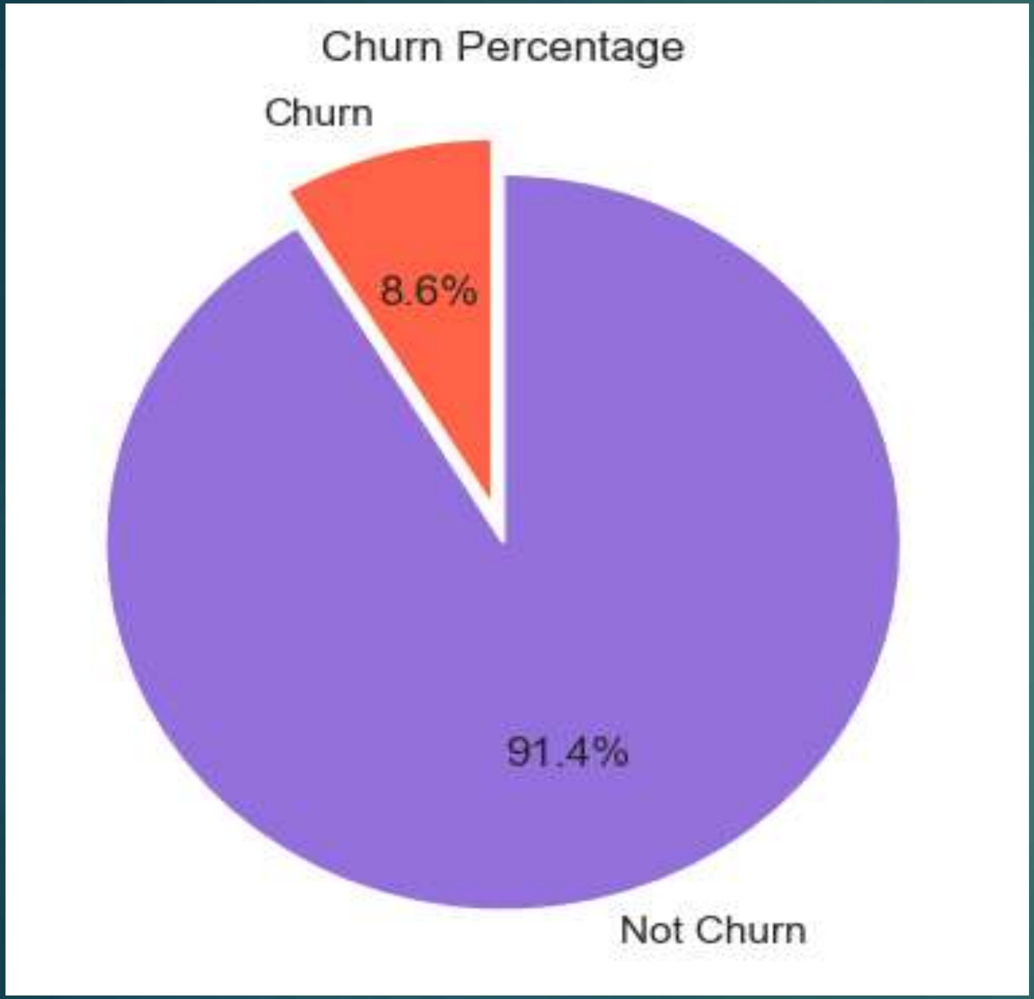
Data handling

- Handling the Missing Value missing - Dropping the columns with more 30% missing values
- Deleting the date columns as the date columns are not required in our analysis
- Filter high-value customers
- Deleting all the attributes corresponding to the churn phase
- Churn percentage came as 8.6%.
- There is very little percentage of churn rate. We will take care of the class imbalance later.

Outlier treatment

- In the filtered dataset except mobile_number and churn columns all the columns are numeric types. Hence, converting mobile_number and churn datatype to object.
- Drive new features
- Deriving new column decrease_arpu_action
- Deriving new column decrease_vbc_action

Churn Rate Calculation

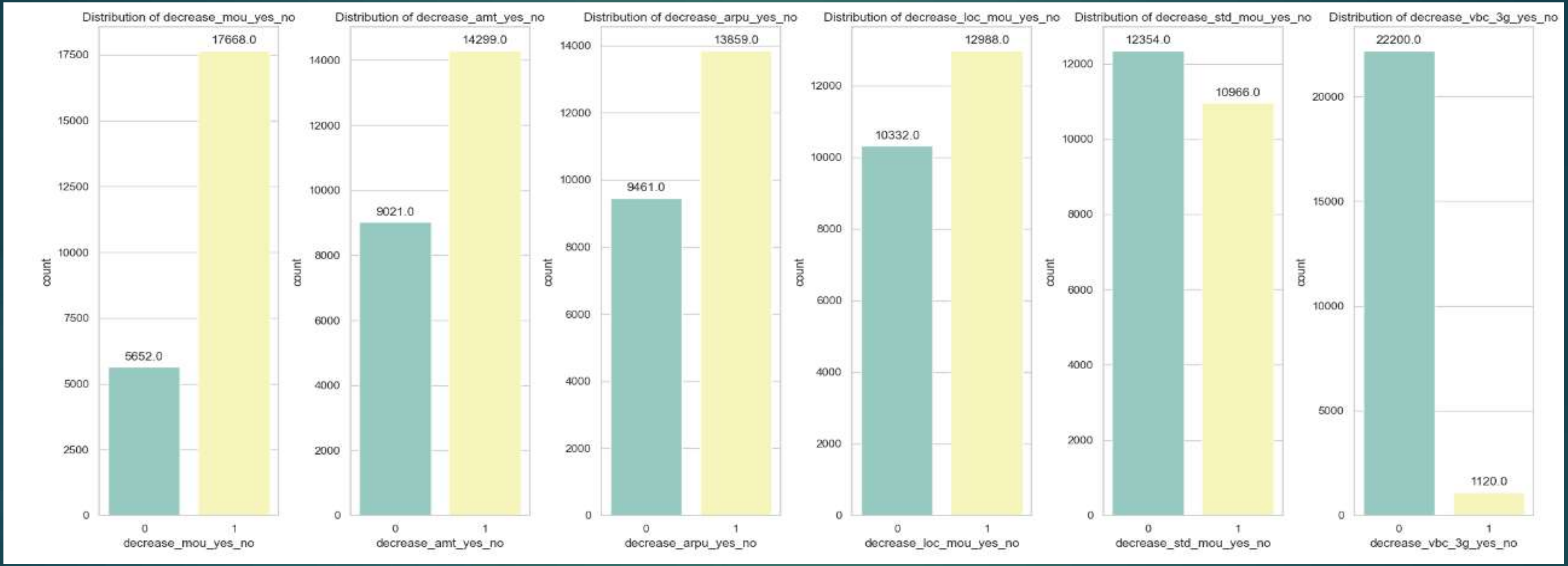


Analysis:

The Pie-chart illustrates a low churn rate of 8.6%, with a majority (91.4%) not churning.

EDA

Univariate analysis:



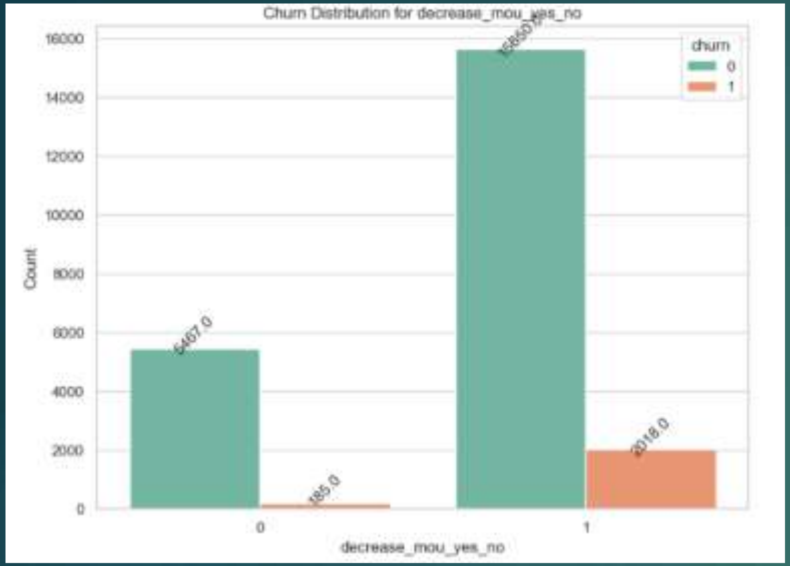
Analysis:

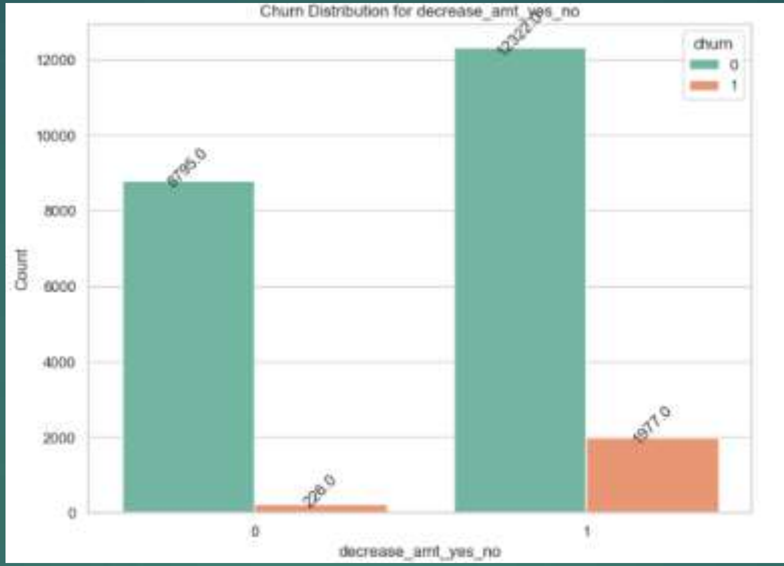
Customers with lower minutes of usage (MOU) during the action phase, reduced recharge frequency, or higher volume-based costs exhibit higher churn rates. Churned customers typically have an average revenue per user (ARPU) in the 0 to 900 range, while non-churned clients range from 0 to 1000 ARPU. Stronger MOU correlates with lower turnover, emphasizing the impact of usage patterns on customer retention.

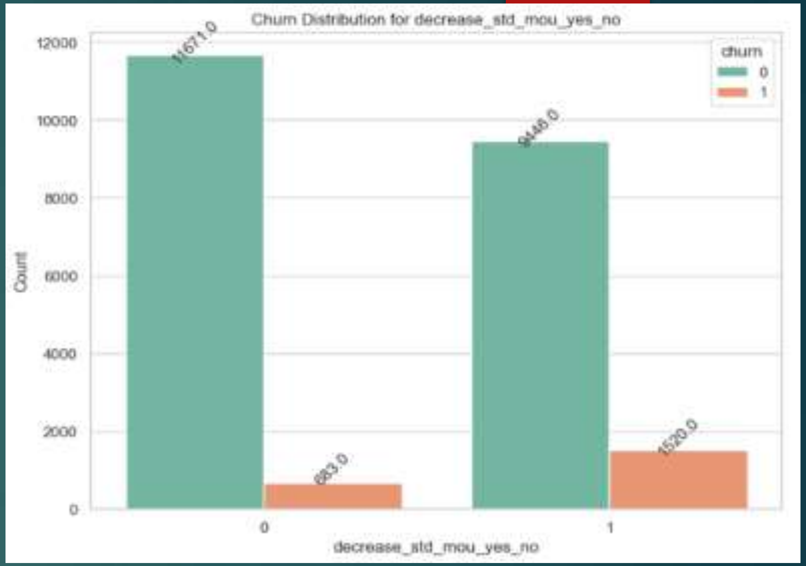
Bivariate analysis :

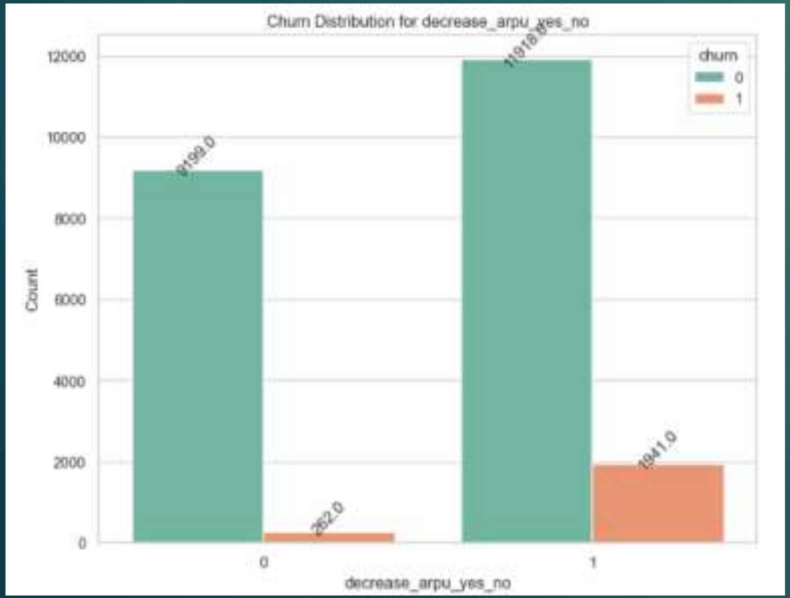
Analysis:

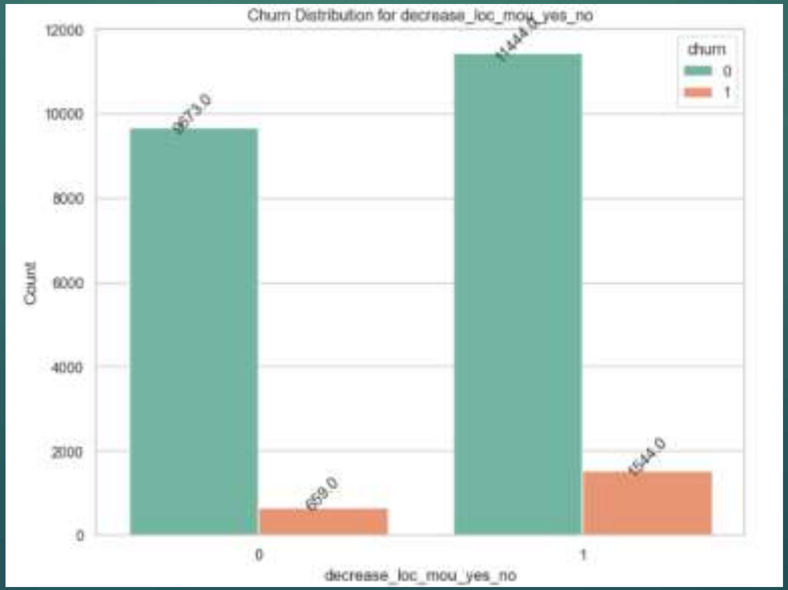
Decreasing recharge amounts and fewer recharges in the action phase correlate with higher churn rates. Higher volume-based costs exacerbate this trend. Proportional relationship observed between recharge quantity and number.

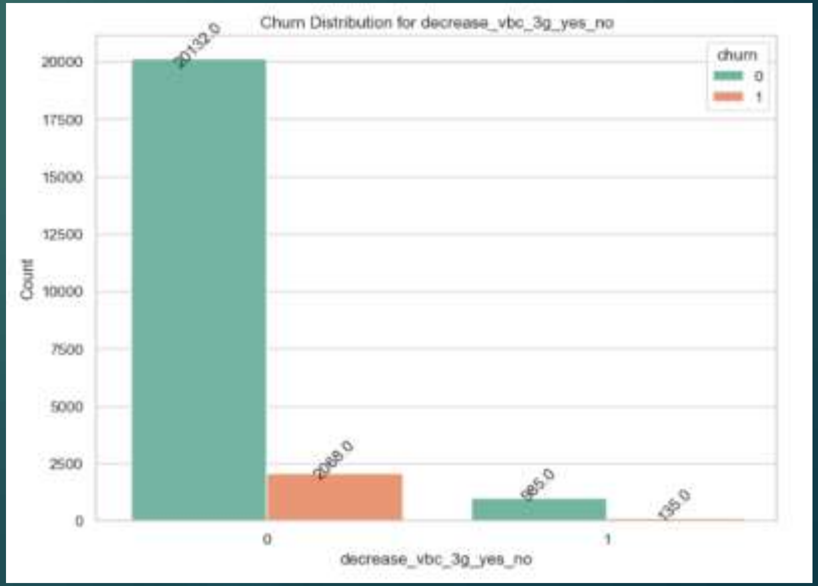








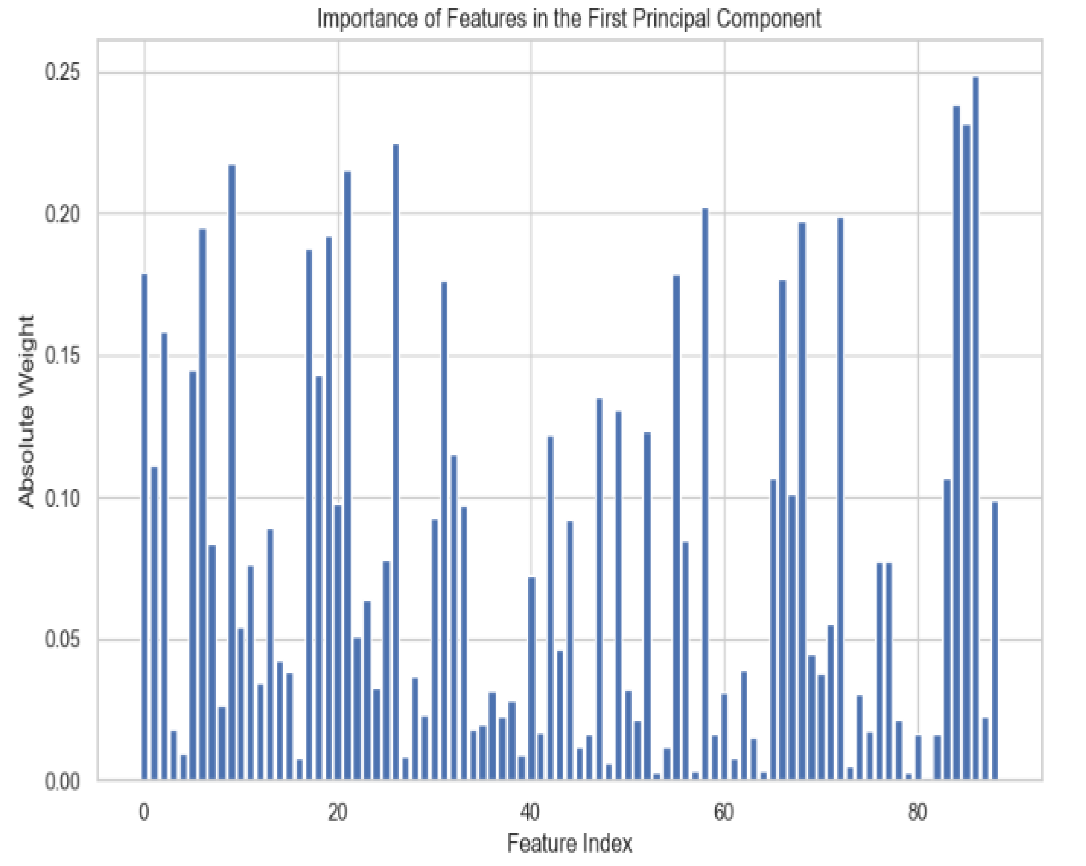


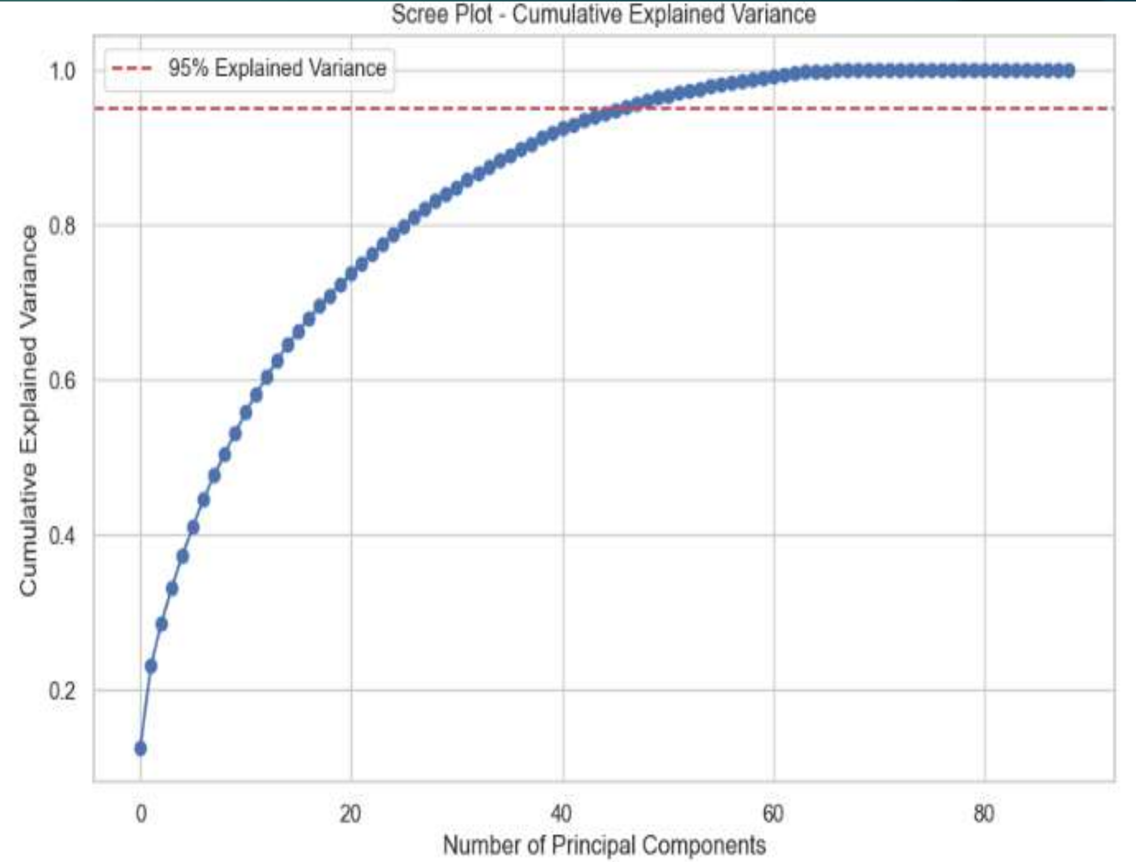


DATA Preparation

- Train –Test Split with 80 : 20
- Dealing with Data Imbalance with SMOTE
- Feature Scaling done with Standard Scaler Technique
- Model with Principal component analysis and Performing PCA with 46 components
- Logistic regression with PCA
- Decision Tree PCA
- Random forest
- Logistic regression with No PCA

Model with PCA

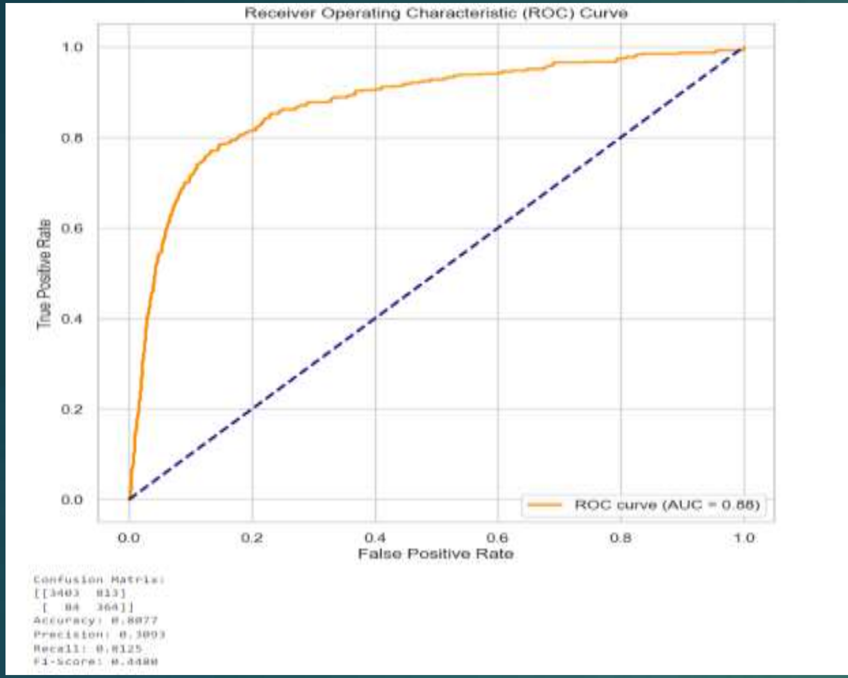


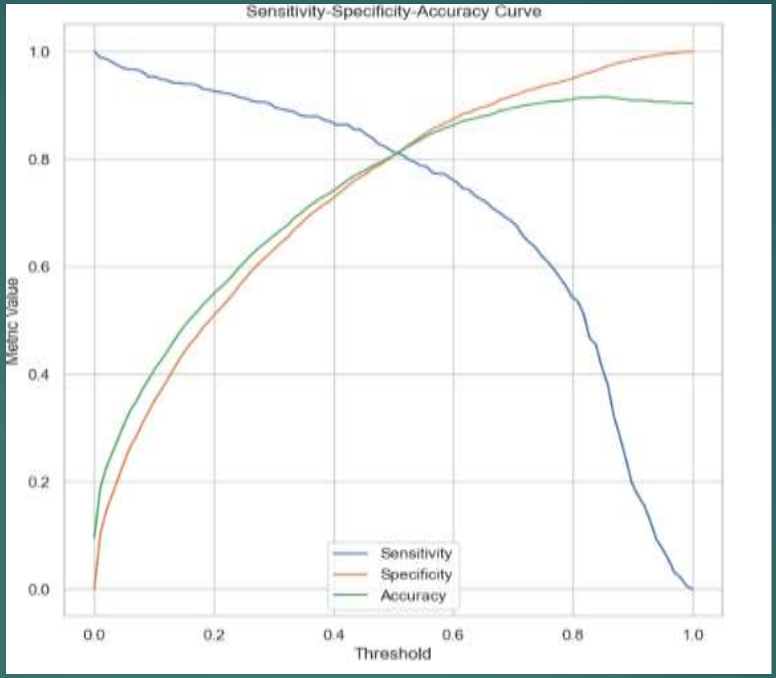


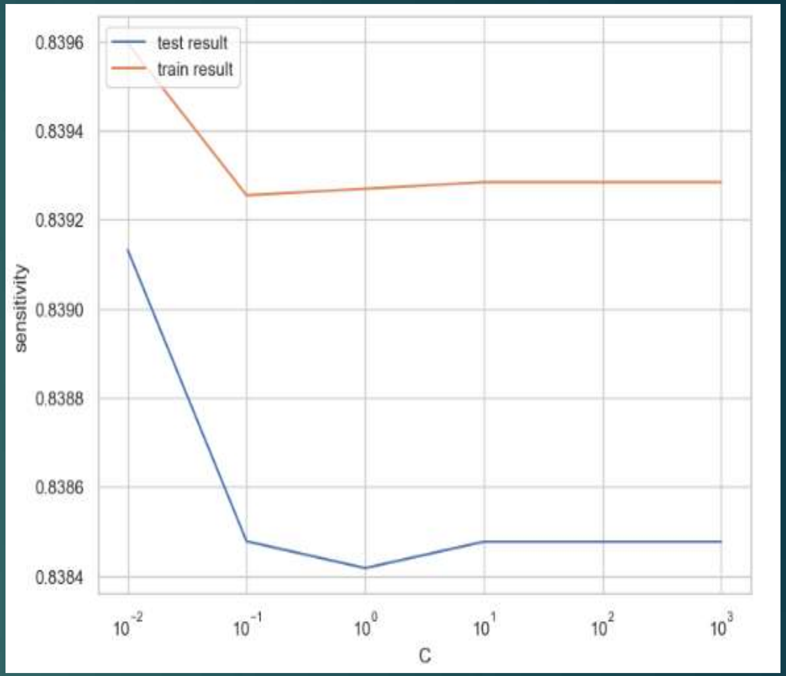
Analysis:

46 Principal components represents the point where you have 95% of the total variance explained.

Logistic regression with PCA







Analysis:

The model exhibits robust performance with an 82.55% mean training accuracy and 91.53% mean testing accuracy. However, low precision suggests challenges in correctly identifying positive cases.

Decision Tree PCA

Model summary

- Train set

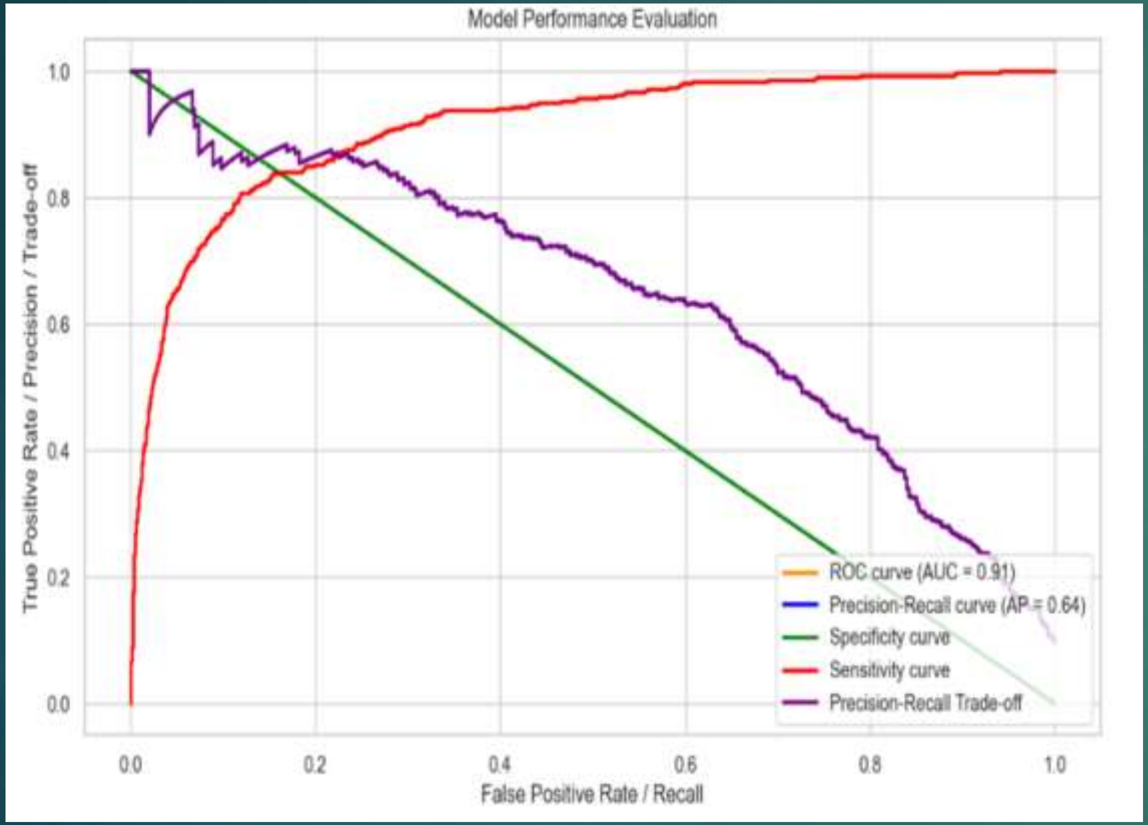
Accuracy = 0.90 Sensitivity = 0.91 Specificity = 0.88

- Test set

Accuracy = 0.86 Sensitivity = 0.70 Specificity = 0.87

```
Cross-Validation Scores (Train): [0.86362964 0.91022038 0.90310651 0.90695266 0.90695266]
Mean Cross-Validation Score (Train): 0.8981723713011187
Cross-Validation Scores (Test): [0.9249732  0.92175777 0.9249732  0.92926045 0.92596567]
Mean Cross-Validation Score (Test): 0.9253860590922265
```

Random forest



Train accuracy: 0.9488787645701438

Train precision: 0.9489833918380547

Test accuracy: 0.9093053173241853

Test precision: 0.9290389654563669

Confusion Matrix for Training Set:
[[15908 993]
[735 16166]]

Confusion Matrix for Test Set:
[[3903 313]
[110 338]]

	precision	recall	f1-score	support
0	0.97	0.93	0.95	4216
1	0.52	0.75	0.62	448
accuracy			0.91	4664
macro avg	0.75	0.84	0.78	4664
weighted avg	0.93	0.91	0.92	4664

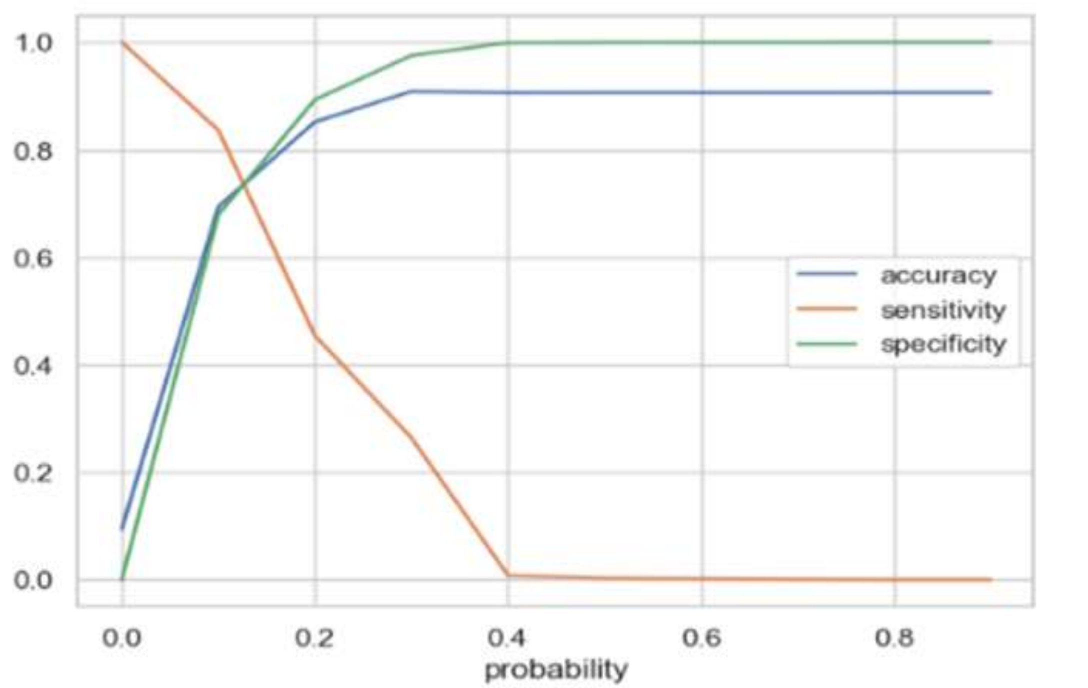
Cross-Validation Scores: [0.90119805 0.93285017 0.93180473 0.93032544 0.93461538]

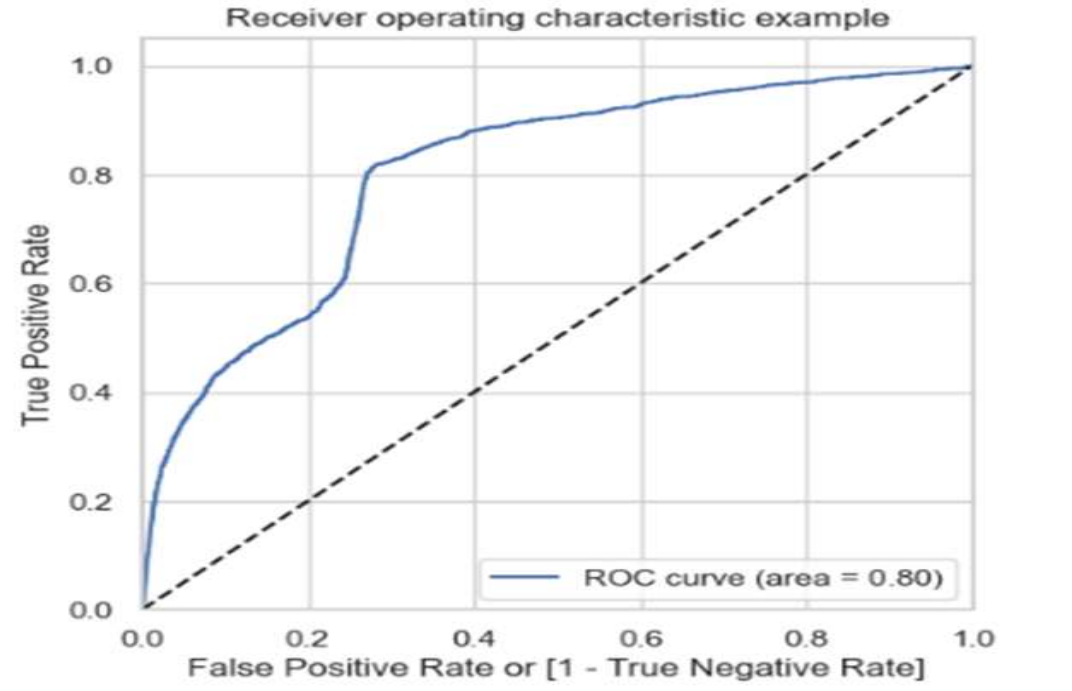
Mean Cross-Validation Score: 0.92615875596989

Cross-Validation Scores (Test): [0.92604502 0.93676313 0.9249732 0.93140407 0.92811159]

Mean Cross-Validation Score (Test): 0.9294594022696641

Logistic regression with No PCA





Model summary

Train set
Accuracy = 0.90 Sensitivity = 0.028 Specificity = 0.99

Test set
Accuracy = 0.90 Sensitivity = 0.022 Specificity = 0.99

Analysis on Model

```
Model - Logistic Regression with scikit learn (model_cv):
Accuracy: 0.6991852487135506
              precision    recall  f1-score   support

           0       0.90      0.75      0.82      4214
           1       0.10      0.25      0.14       450

    accuracy                           0.70      4664
   macro avg       0.50      0.50      0.48      4664
weighted avg       0.83      0.70      0.75      4664


Model - Logistic Regression with statsmodel (logreg_m4):
Accuracy: 0.9033018867924528
              precision    recall  f1-score   support

           0       0.90      1.00      0.95      4214
           1       0.33      0.00      0.00       450

    accuracy                           0.90      4664
   macro avg       0.62      0.50      0.48      4664
weighted avg       0.85      0.90      0.86      4664


Model - Decision Tree (dt_model):
Accuracy: 0.8880789022298456
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      4214
           1       0.38      0.26      0.31       450

    accuracy                           0.89      4664
   macro avg       0.65      0.61      0.62      4664
weighted avg       0.87      0.89      0.88      4664


Model - Random Forest (rf):
Accuracy: 0.9313893653516295
              precision    recall  f1-score   support

           0       0.95      0.98      0.96      4214
           1       0.72      0.47      0.57       450

    accuracy                           0.93      4664
   macro avg       0.83      0.72      0.77      4664
weighted avg       0.92      0.93      0.92      4664


Best Model:
Model Name: RandomForestClassifier(max_depth=10, max_features=5, n_estimators=15,
                       oob_score=True, random_state=25)
Accuracy: 0.9313893653516295
```

Analysis:

- **Logistic Regression with scikit-learn (model_cv): Accuracy: 0.6992 Key Insight: This model has a relatively lower accuracy compared to the other models. It struggles with both precision and recall for class 1.**
- **Logistic Regression with statsmodels (logreg_m4): Accuracy: 0.9033 Key Insight: This model shows high accuracy but has issues with recall for class 1. It seems to have a challenge correctly identifying instances of class 1.**
- **Decision Tree (dt_model): Accuracy: 0.8881 Key Insight: This model performs reasonably well but has lower precision, recall, and F1-score for class 1 compared to class 0. It might be sensitive to imbalances in the data.**
- **Random Forest (rf): Accuracy: 0.9314 Key Insight: The Random Forest model outperforms the other models in terms of accuracy. It provides a good balance between precision and recall for both classes. This model seems promising.**
- **Best Model (Random Forest): Model Name: RandomForestClassifier(max_depth=10, max_features=5, n_estimators=15, oob_score=True, random_state=25) Accuracy: 0.9314 Key Insight: This model is identified as the best-performing one. It achieves a high accuracy, and its precision, recall, and F1-scores for both classes are relatively well-balanced.**

TOP FEATURES OF RANDOM FOREST