BLACKSPI /
Phase3-Project

Code Issues 1 Pull requests Actions Projects Wiki Security Insights Se

This project provides EDA on SyriaTel company and also has predictive models that predict customer churn.

0 stars 0 forks 1 watching Branches Activity
Tags

Public repository

1 Branch 0 Tags Go to file t Go to file + Add file Code ···

BLACKSPI Final Update cd53049 · now

.ipynb_checkpoints	Updated	yesterday
Images	Final Update	2 minutes ago
Presentation.pptx	Final Update	2 minutes ago
README.md	Update README.md	1 hour ago
bigml_59c28831336c6604c80...	First commit	yesterday
index.ipynb	Final Update	2 minutes ago
index.pdf	Final Update	2 minutes ago

README

Predicting Customer Churn: Unlocking Insights for Retention at SyriaTel

Business Understanding

Introduction

Customer churn is a key problem for telecommunications firms since losing customers has a direct influence on revenues and growth. This project seeks to create a machine learning classifier that can predict if a client will churn (leave SyriaTel). By studying customer usage habits, plan subscriptions, and contacts with the organization, we identify significant churn contributors and give practical ideas for reducing it.

Data Understanding

Dataset used: https://www.kaggle.com/datasets/becksddf/churn-in-telecoms-dataset. The collection has 3,333 entries, each representing a consumer. The idea is to evaluate these records for trends that suggest client turnover. Here's a full analysis of the data:
Target Variable

Churn: A binary variable that indicates if a client has churned. True: Customer churn occurred. False: The customer was kept. Features: The number of features is 21 columns, including the target variable (churn).

Data Types:

1. `Categorical:` state, phone numbe, international plan, voice mail plan, churn.
2. `Numerical:` Integer: account length, area code, amount of vmail messages, total day calls, total evening calls, total night calls, total international calls, and customer service calls.
3. `Float:` total day minutes, total day charge, total evening minutes, total evening charge, total night minutes, total night charge, total international minutes, total international charge.

Project Plan

1. Data Preparation
2. Model Creation
3. Model Selection
4. Recommendations

1. Data Preparation

Lets get into it! Okay, we're attempting to figure out why consumers are leaving SyriaTel and who will churn in the near future. The dataset appears clean at first inspection, which is a good start; there are no missing values or duplicates.

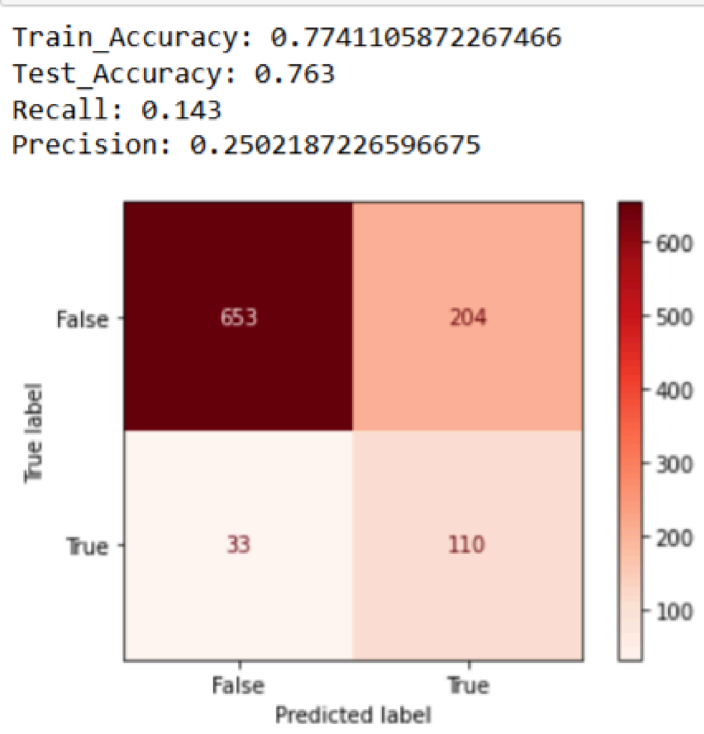
Let's take a look at the goal variable, churn. I can already tell that the data is skewed. Only 14.5% ((483/3333)*100) of consumers have churned. That is not unusual in churn prediction, but it implies we cannot depend just on accuracy as a performance indicator. To ensure that the model correctly detects churners without biasing toward the majority class, we will need to consider accuracy, recall, and AUC-ROC values.

Another item on my radar is the state column. It contains too many distinct categories, thus it may not be very useful until we can aggregate it. And let's remove the phone number column it's merely an identifier with no predictive ability.

Once we've investigated and cleaned up the data, we'll need to address the class imbalance. Perhaps oversampling churners using SMOTE or modifying class weights in our model would do the job. Following that, we may experiment with several models—Logistic Regression for interpretability and something like Decision Tree.

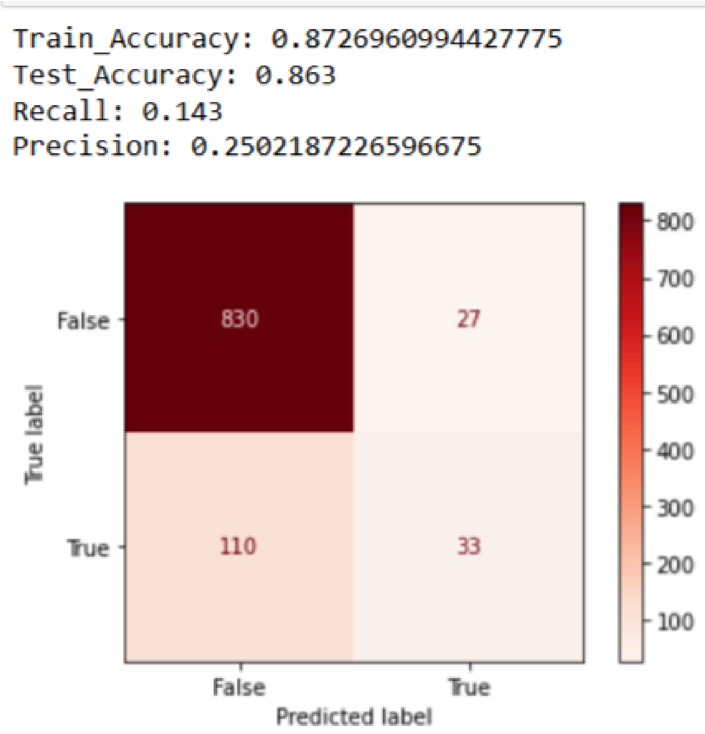
2. Model Creation

In this project I used 4 models:
I. Logistic Regression Model



```
Train_Accuracy: 0.7741105872267466
Test_Accuracy: 0.763
Recall: 0.143
Precision: 0.2502187226596675
```

II. Logistic Regression Model with SMOTE

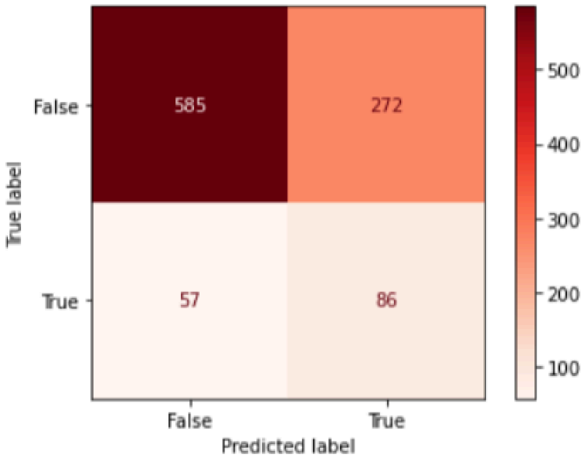


```
Train_Accuracy: 0.8726960994427775
Test_Accuracy: 0.863
Recall: 0.143
Precision: 0.2502187226596675
```

III. KNN Model

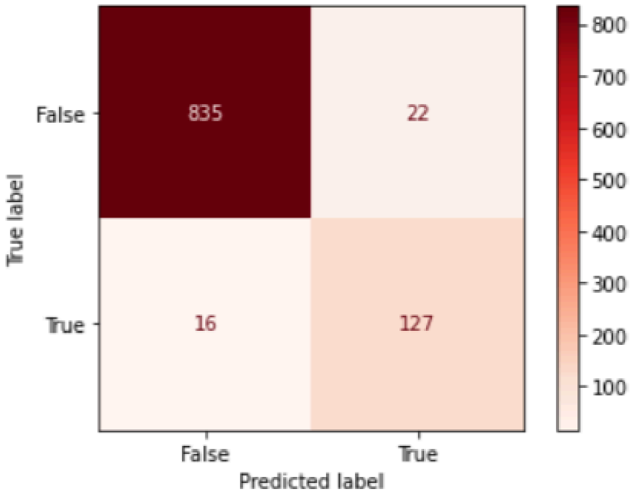
```
Train_Accuracy: 0.8406924234821876
Test_Accuracy: 0.671
Recall: 0.6013986013986014
Precision: 0.24022346368715083
F1_Score: 0.343313373253493
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1ca02ccea30>
```

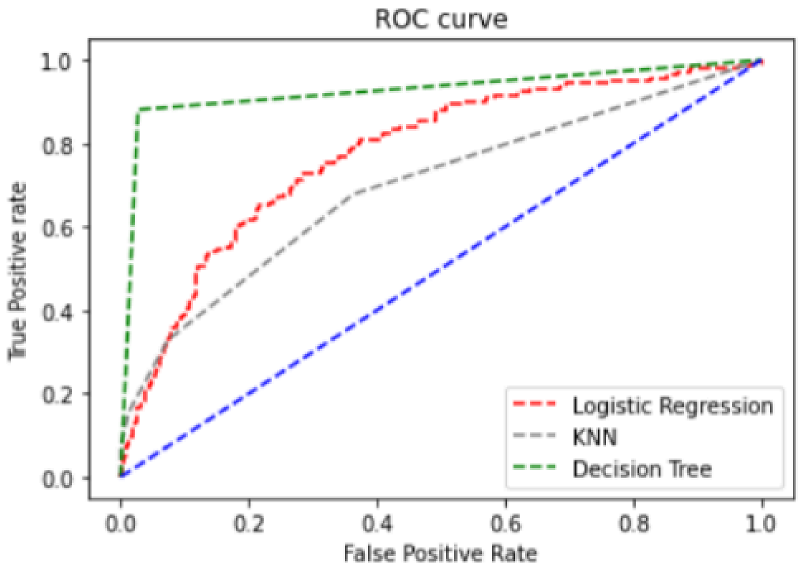


IV. Decision Tree

```
Train_Accuracy: 1.0
Test_Accuracy: 0.962
Recall: 0.143
Precision: 0.2502187226596675
```



3. Model Selection



The ROC curve clearly shows that the Decision Tree model outperforms the others. Its green curve is the steepest and spans the biggest area under the curve (AUC), indicating that it performs the best at differentiating between positive and negative classes. Essentially, the Decision Tree model finds a good mix between properly recognizing true positives and reducing false positives, making it the best option for this dataset.

4. Recommendations

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

● **Jupyter Notebook** 100.0%