ANALYZING CUSTOMER BEHAVIOR: PREDICTING TELECOM COMPANY CHURN USING MACHINE LEARNING TECHNIQUES

Venkata Ravi Kiran Kolla
Sr. Research Scientist and Software Engineer Department of Information Technology
venkat0634@gmail.com

ABSTRACT

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. Therefore, finding factors that increase customer churn is important to take necessary actions to reduce this churn. The main contribution of our work is to develop a churn prediction model which assists telecom operators to predict customers who are most likely subject to churn. The model developed in this work uses machine learning techniques on big data platform and builds a new way of features' engineering and selection.

INTRODUCTION

Customer churn means shifting from one service provider to its competitor in the market. Customer churn is one of the biggest fears of any industry, particularly for the telecom industry. With an increase in the number of telecom service providers in South Asia, the level of competition is quite high. Although there are many reasons for customer churn, some of the major reasons are service dissatisfaction, costly subscription, and better alternatives. The telecom service providers strive very hard to sustain in this competition. So, to sustain this competition they often try to retain their customers rather than acquiring new ones as it proved to be much costlier. Hence predicting churn in the telecom industry is very important. To reduce customer churn, telecom companies need to predict which customers are at high risk of churn.

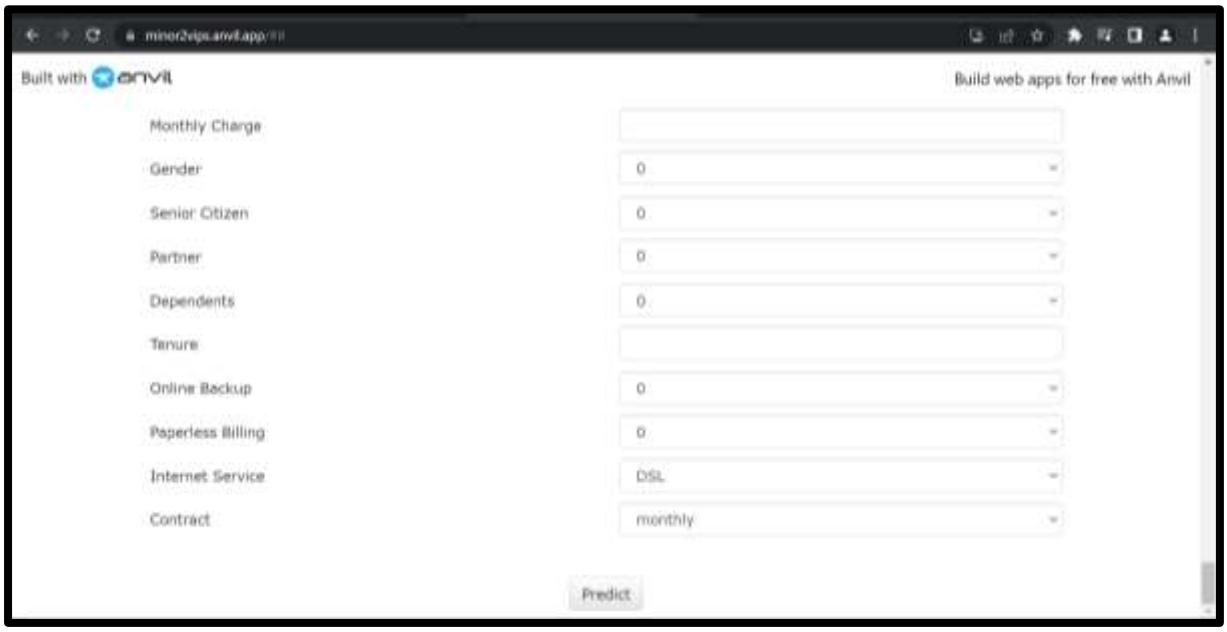
METHODOLOGY USED FOR ANALYSIS, DESIGN & DEVELOPMENT:

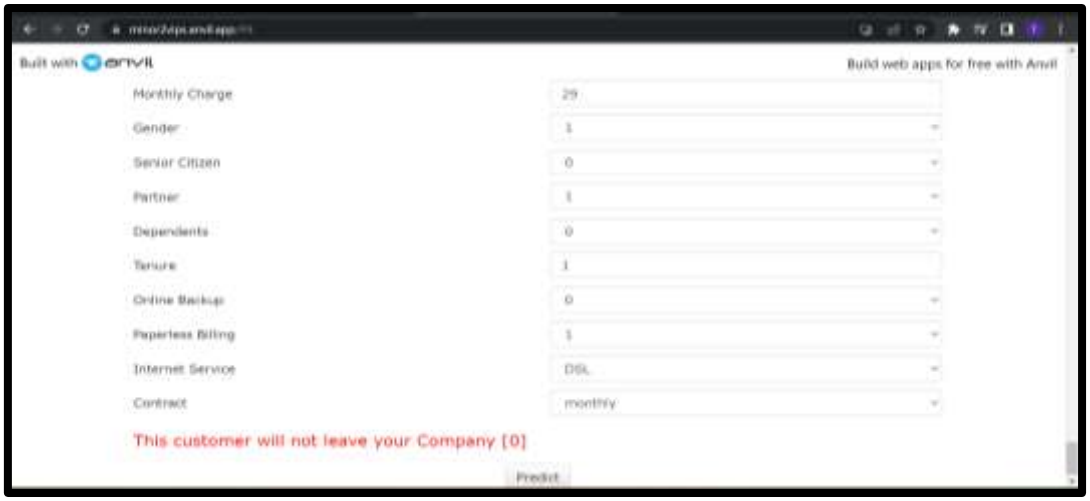
The tools used for the project were in two parts. In the first part the machine learning coding is done in Google colab and the web deployment is done in Anvil AI.

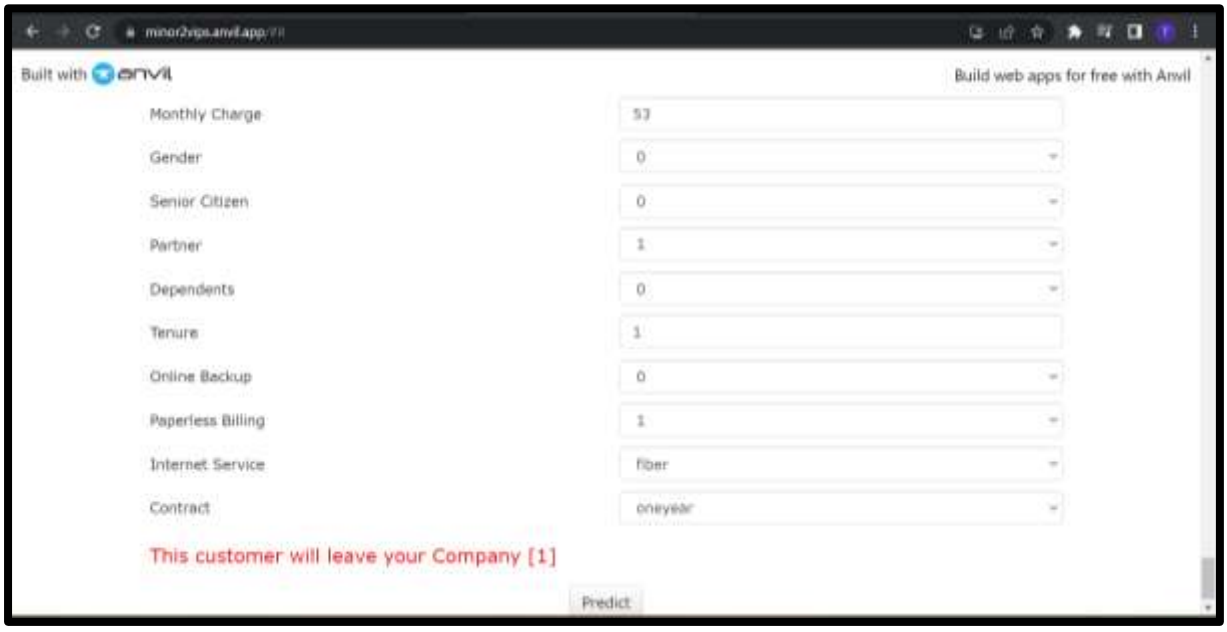
GOOGLE COLABORATORY

Colaboratory, or "Colab" for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use.

OUTPUT


Built with anvil
Build web apps for free with Anvil
Monthly Charge
Gender
Senior Citizen
Partner
Dependents
Tenure
Online Backup
Paperless Billing
Internet Service
DSL
Contract
monthly
Predict


Built with anvil
Build web apps for free with Anvil
Monthly Charge
29
Gender
Senior Citizen
Partner
Dependents
Tenure
Online Backup
Paperless Billing
Internet Service
DSL
Contract
monthly
This customer will not leave your Company [0]
Predict


Built with anvil
Build web apps for free with Anvil
Monthly Charge
53
Gender
Senior Citizen
Partner
Dependents
Tenure
Online Backup
Paperless Billing
Internet Service
fiber
Contract
oneyear
This customer will leave your Company [1]
Predict

CODE

```
import pandas as pd
url="https://raw.githubusercontent.com/tarlochan123/minor2/main/customer%20churn%20-%20TOM%20-%20Sheet1.csv"
df=pd.read_csv(url)
df.head(5)
df.dtypes
df.Partner.unique()
df.Churn.unique()
df.gender.unique()
df.InternetService.unique()
df.OnlineBackup.unique()
df.replace('No internet service','No',inplace=True)
df.replace('No phone service','No',inplace=True)
df.OnlineBackup.unique()
yes_no_columns = ['Partner','Dependents','OnlineBackup',
                  'Churn','PaperlessBilling']
for col in yes_no_columns:
    df[col].replace({'Yes': 1,'No': 0},inplace=True)
df['gender'].replace({'Female':1,'Male':0},inplace=True)
df.head(5)
df1 = pd.get_dummies(data=df, columns=['InternetService','Contract'])
df1.columns
df1.head(5)
import matplotlib.pyplot as plt
import seaborn as sns
corrmat=df1.corr()
top=corrmat.index
plt.figure(figsize=(15,10))
g=sns.heatmap(df1[top].corr(),annot=True,cmap='RdYlGn')
target_name="Churn"
X = df1.drop('Churn', axis=1)
y=df1[target_name]


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,)
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.model_selection import cross_val_score,StratifiedKFold
```

```
from sklearn import tree
from sklearn.ensemble import RandomForestClassifier
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
pre=logreg.predict(X_test)
def print_confusion_matrix(confusion_matrix, class_names, figsize = (10,7), fontsize=14):
    """Prints a confusion matrix, as returned by sklearn.metrics.confusion_matrix, as a heatmap.

    Arguments
    ---------
    confusion_matrix: numpy.ndarray
        The numpy.ndarray object returned from a call to sklearn.metrics.confusion_matrix.
        Similarly constructed ndarrays can also be used.
    class_names: list
        An ordered list of class names, in the order they index the given confusion matrix.
    figsize: tuple
        A 2-long tuple, the first value determining the horizontal size of the ouputted figure,
        the second determining the vertical size. Defaults to (10,7).
    fontsize: int
        Font size for axes labels. Defaults to 14.

    Returns
    -------
    matplotlib.figure.Figure
        The resulting confusion matrix figure
    """
    df_cm = pd.DataFrame(
        confusion_matrix, index=class_names, columns=class_names,
    )
    fig = plt.figure(figsize=figsize)
    try:

        heatmap = sns.heatmap(df_cm, annot=True, fmt="d",cmap="YlGnBu")
    except ValueError:
        raise ValueError("Confusion matrix values must be integers.")
    heatmap.yaxis.set_ticklabels(heatmap.yaxis.get_ticklabels(), rotation=0, ha='right', fontsize=fontsize)
    heatmap.xaxis.set_ticklabels(heatmap.xaxis.get_ticklabels(), rotation=45, ha='right', fontsize=fontsize)
    plt.ylabel('Actual or Truth')
    plt.xlabel('Prediction')
```

```
cm = confusion_matrix(y_test,pre)
print_confusion_matrix(cm,["not leave","leave"])

print(classification_report(y_test,pre))

treemodel = tree.DecisionTreeClassifier()
skk=StratifiedKFold(shuffle=True,n_splits=10)
treemodel.fit(X_train, y_train)

score3=cross_val_score(treemodel,X_train,y_train,cv=skk,scoring='accuracy') # Training score
print(score3.mean())

score2=cross_val_score(treemodel,X_test,y_test,cv=skk,scoring='accuracy') # Testing score
print(score2.mean())

pretree=treemodel.predict(X_test)

print(classification_report(y_test,pretree))

cm = confusion_matrix(y_test,pretree)
print_confusion_matrix(cm,["not leave","leave"])

rf= RandomForestClassifier()
rf.fit(X_train, y_train)

rpretree=rf.predict(X_test)

score7=cross_val_score(rf,X_train,y_train,cv=skk,scoring='accuracy') # Training score
print(score7.mean())

score8=cross_val_score(rf,X_test,y_test,cv=skk,scoring='accuracy') # Testing score
print(score8.mean())

print(classification_report(y_test,rpretree))

cm = confusion_matrix(y_test,rpretree)
print_confusion_matrix(cm,["not leave","leave"])

import pickle
    with open('Telecom','wb') as file:
 pickle.dump(logreg,file)
```

```
    with open('Telecom','rb') as file:
  Telecom= pickle.load(file)


Telecom.predict([[29,1,0,1,0,1,1,1,1,0,0,1,0,0]])
```

ANVIL DEPLOYMENT CODE

```
pip install anvil-uplink

import pickle
with open('Telecom','rb') as file:
    Telecom= pickle.load(file)

Telecom.predict([[29,1,0,1,0,1,1,1,1,0,0,1,0,0]])



import anvil.server

anvil.server.connect("452DA5QKXR3YPFG3WQEA4OKC-6UAISXLKVLJKPNJB")

@anvil.server.callable
def get_data(a,b,c,d,e,f,g,h,i,j):

  if i=="DSL" and j=="monthly":
    score=Telecom.predict([[int(a),int(b),int(c),int(d),int(e),int(f),int(g),int(h),1,0,0,1,0,0]])

  elif i=="DSL" and j=="oneyear":
    score=Telecom.predict([[int(a),int(b),int(c),int(d),int(e),int(f),int(g),int(h),1,0,0,0,1,0]])

  elif i=="DSL" and j=="twoyear":
    score=Telecom.predict([[int(a),int(b),int(c),int(d),int(e),int(f),int(g),int(h),1,0,0,0,0,1]])

  elif i=="fiber" and j=="monthly":
    score=Telecom.predict([[int(a),int(b),int(c),int(d),int(e),int(f),int(g),int(h),0,1,0,1,0,0]])

  elif i=="fiber" and j=="oneyear":
    score=Telecom.predict([[int(a),int(b),int(c),int(d),int(e),int(f),int(g),int(h),0,1,0,0,1,0]])

  elif i=="fiber" and j=="twoyear":
    score=Telecom.predict([[int(a),int(b),int(c),int(d),int(e),int(f),int(g),int(h),0,1,0,0,0,1]])

  elif i=="no" and j=="monthly":
    score=Telecom.predict([[int(a),int(b),int(c),int(d),int(e),int(f),int(g),int(h),0,0,1,1,0,0]])


     elif i=="no" and j=="oneyear":

    score=Telecom.predict([[int(a),int(b),int(c),int(d),int(e),f,int(g),int(h),0,0,1,0,1,0]])


  elif i=="no" and j=="twoyear":

    score=Telecom.predict([[int(a),int(b),int(c),int(d),int(e),int(f),int(g),int(h),0,0,1,0,0,1]])

  result=score
```

```
if result==1:
  return ('This customer will leave your Company [1]')
elif result==0:
  return ("This customer will not leave your Company [0]")
```

REFERENCES

[1] Hand on Machine Learning with Scikit- Learn Keras and Tensorflow By Aurelien Geron
[2] https://www.w3schools.com
[3] https://www.kaggle.com/
[4] https://towardsdatascience.com/
[5] https://www.analyticsvidhya.com/