**CUSTOMER CHURN PREDICTION**

Sachin S, Ramco Institute of Technology, 953621243046

**Import Libraries:**

The code begins by importing essential Python libraries like NumPy, Pandas, Seaborn, and Matplotlib. These libraries are used for data manipulation, data visualization, and analysis.

Load the Dataset: It loads a telecom customer dataset from a CSV file named 'Dataset.csv' into a Pandas DataFrame named 'telecom\_cust'.

**Data Exploration:**

It prints the first few rows of the dataset to get an initial look at the data.

Lists the column names present in the dataset using telecom\_cust.columns.values.

Checks the data types of each column using telecom\_cust.dtypes.

Converts the 'TotalCharges' column to a numeric data type. This is done to ensure that it can be used in numerical calculations.

**Data Preprocessing:**

It checks for missing values using telecom\_cust.isnull().sum().

Removes rows with missing values using telecom\_cust.dropna(inplace=True).

Drops the 'customerID' column, assuming it's not relevant to the analysis, by creating a new DataFrame 'df2' without it.

Binary Classification for 'Churn':

Converts the 'Churn' column to binary values (0 for 'No' and 1 for 'Yes'). This is typically done to prepare the target variable for a classification model.

Creating Dummy Variables:

Converts categorical variables into dummy variables using pd.get\_dummies(). This allows the machine learning model to work with categorical data by transforming it into a numerical format.

Data Visualization and Analysis:

Calculates the correlation of all variables with the 'Churn' variable and visualizes it as a bar plot to understand which features are most related to customer churn.

Data Exploration (sub-sections): The code includes comments indicating where specific data exploration and analysis tasks can be performed. These are placeholders, and you would typically fill them in with your own analysis.

Demographics: Gender distribution, senior citizen percentage, partner and dependent status.

Customer Account Information: Tenure distribution, contracts distribution, churn by contract type and tenure.

Distribution of various services.

Churn vs. Monthly Charges.

Churn vs. Total Charges.

Churn Rate.

Churn vs. Tenure.

Churn by Contract Type.

EDA and Prediction

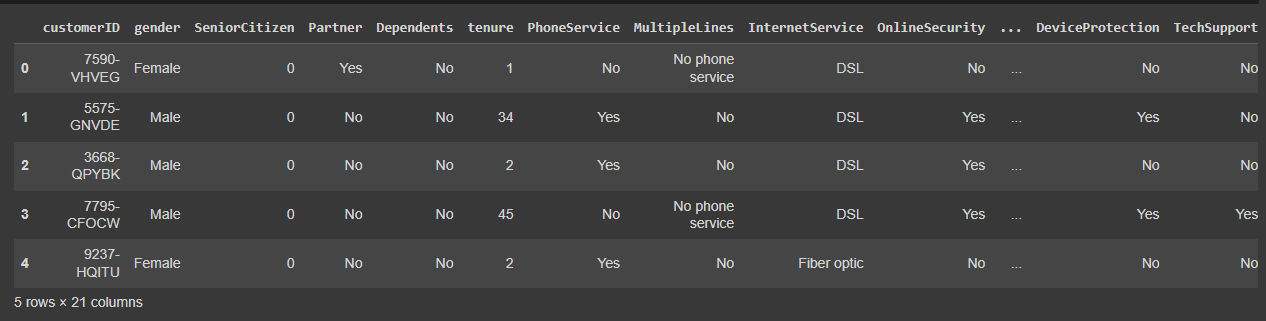
Churn is a one of the biggest problem in the telecom industry. Research has shown that the average monthly churn rate among the top 4 wireless carriers in the US is 1.9% - 2%.

import numpy as np # linear algebra  
import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)  
import seaborn as sns # For creating plots  
import matplotlib.ticker as mtick # For specifying the axes tick format   
import matplotlib.pyplot as plt  
  
sns.set(style = 'white')

Let us read the data file in the python notebook

telecom\_cust = pd.read\_csv('Dataset.csv')

telecom\_cust.head()



telecom\_cust.columns.values

array(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',  
 'tenure', 'PhoneService', 'MultipleLines', 'InternetService',  
 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',  
 'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract',  
 'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges',  
 'TotalCharges', 'Churn'], dtype=object)

Let's explore the data to see if there are any missing values.

# Checking the data types of all the columns  
telecom\_cust.dtypes

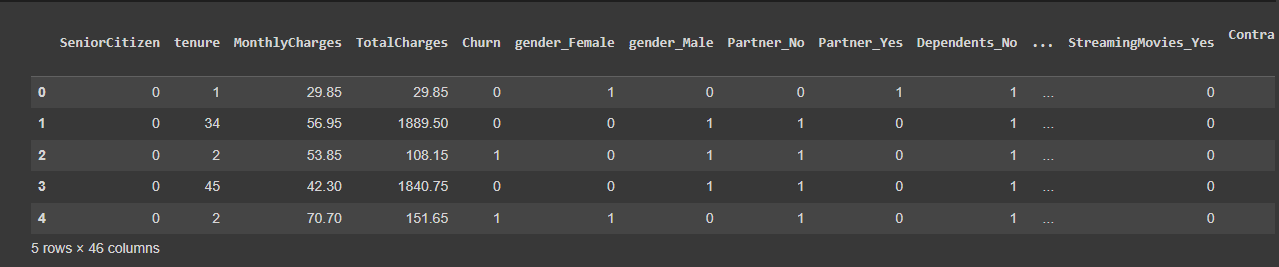
customerID object  
gender object  
SeniorCitizen int64  
Partner object  
Dependents object  
tenure int64  
PhoneService object  
MultipleLines object  
InternetService object  
OnlineSecurity object  
OnlineBackup object  
DeviceProtection object  
TechSupport object  
StreamingTV object  
StreamingMovies object  
Contract object  
PaperlessBilling object  
PaymentMethod object  
MonthlyCharges float64  
TotalCharges object  
Churn object  
dtype: object

# Converting Total Charges to a numerical data type.  
telecom\_cust.TotalCharges = pd.to\_numeric(telecom\_cust.TotalCharges, errors='coerce')  
telecom\_cust.isnull().sum()

customerID 0  
gender 0  
SeniorCitizen 0  
Partner 0  
Dependents 0  
tenure 0  
PhoneService 0  
MultipleLines 0  
InternetService 0  
OnlineSecurity 0  
OnlineBackup 0  
DeviceProtection 0  
TechSupport 0  
StreamingTV 0  
StreamingMovies 0  
Contract 0  
PaperlessBilling 0  
PaymentMethod 0  
MonthlyCharges 0  
TotalCharges 11  
Churn 0  
dtype: int64

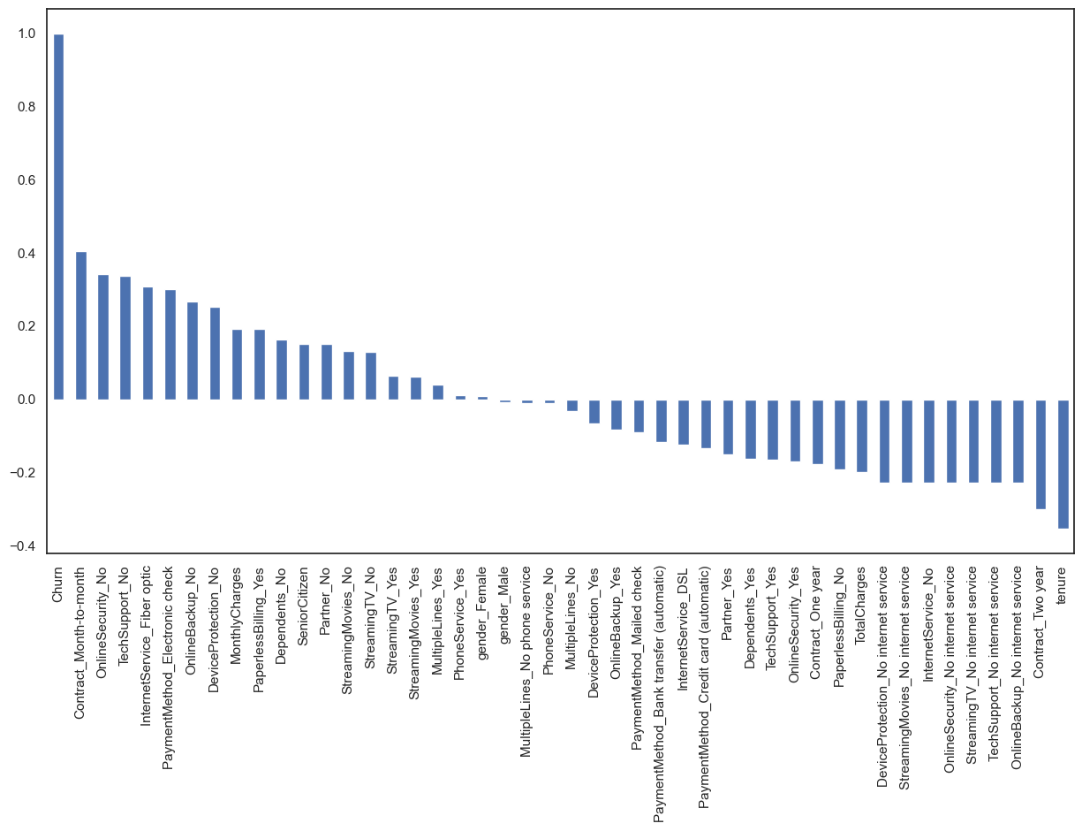
After looking at the above output, we can say that there are 11 missing values for Total Charges. Let us replace remove these 11 rows from our data set

#Removing missing values   
telecom\_cust.dropna(inplace = True)  
#Remove customer IDs from the data set  
df2 = telecom\_cust.iloc[:,1:]  
#Convertin the predictor variable in a binary numeric variable  
df2['Churn'].replace(to\_replace='Yes', value=1, inplace=True)  
df2['Churn'].replace(to\_replace='No', value=0, inplace=True)  
  
#Let's convert all the categorical variables into dummy variables  
df\_dummies = pd.get\_dummies(df2)  
df\_dummies.head()



#Get Correlation of "Churn" with other variables:  
plt.figure(figsize=(15,8))  
df\_dummies.corr()['Churn'].sort\_values(ascending = False).plot(kind='bar')

<Axes: >



Month to month contracts, absence of online security and tech support seem to be positively correlated with churn. While, tenure, two year contracts seem to be negatively correlated with churn.

Interestingly, services such as Online security, streaming TV, online backup, tech support, etc. without internet connection seem to be negatively related to churn.

We will explore the patterns for the above correlations below before we delve into modelling and identifying the important variables.

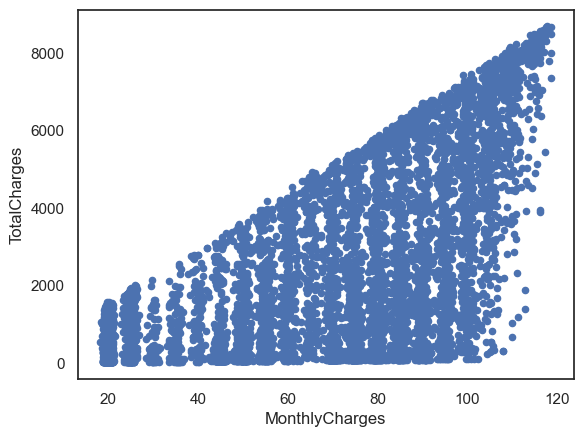
Now let's take a quick look at the relation between monthly and total charges

We will observe that the total charges increases as the monthly bill for a customer increases.

telecom\_cust[['MonthlyCharges', 'TotalCharges']].plot.scatter(x = 'MonthlyCharges',  
 y='TotalCharges')

C:\ProgramData\anaconda3\lib\site-packages\pandas\plotting\\_matplotlib\core.py:1070: UserWarning: No data for colormapping provided via 'c'. Parameters 'cmap' will be ignored  
 scatter = ax.scatter(

<Axes: xlabel='MonthlyCharges', ylabel='TotalCharges'>



Finally, let's take a look at out predictor variable (Churn) and understand its interaction with other important variables as was found out in the correlation plot.

Lets first look at the churn rate in our data

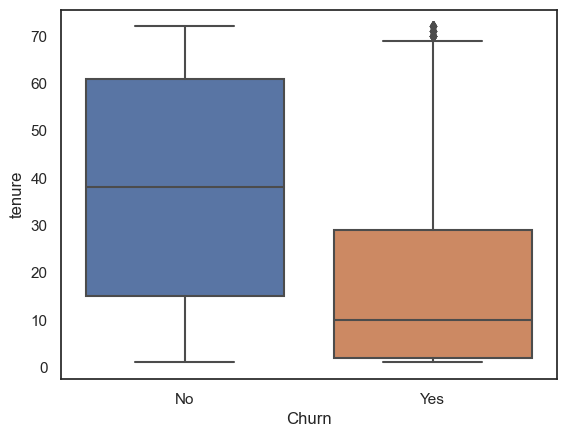
In our data, 74% of the customers do not churn. Clearly the data is skewed as we would expect a large majority of the customers to not churn. This is important to keep in mind for our modelling as skeweness could lead to a lot of false negatives. We will see in the modelling section on how to avoid skewness in the data.

Lets now explore the churn rate by tenure, seniority, contract type, monthly charges and total charges to see how it varies by these variables.

i.) Churn vs Tenure: As we can see form the below plot, the customers who do not churn, they tend to stay for a longer tenure with the telecom company.

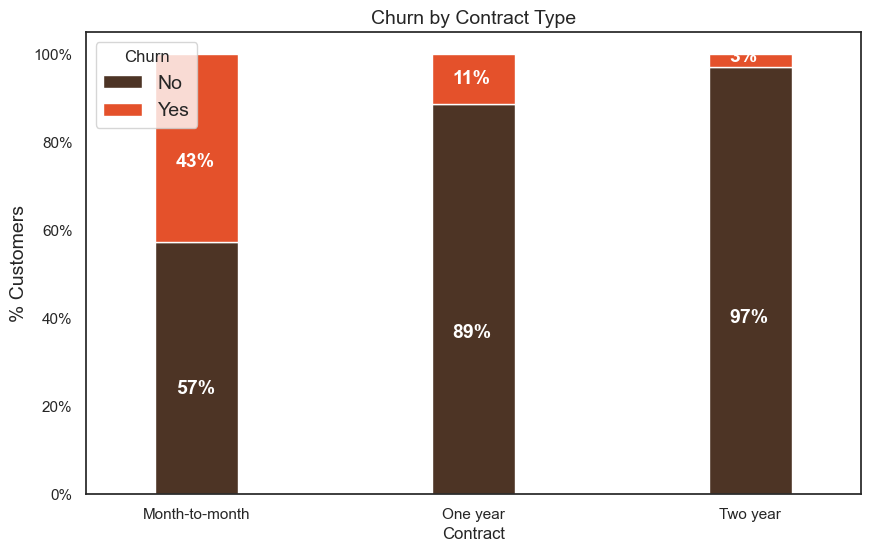
sns.boxplot(x = telecom\_cust.Churn, y = telecom\_cust.tenure)

<Axes: xlabel='Churn', ylabel='tenure'>



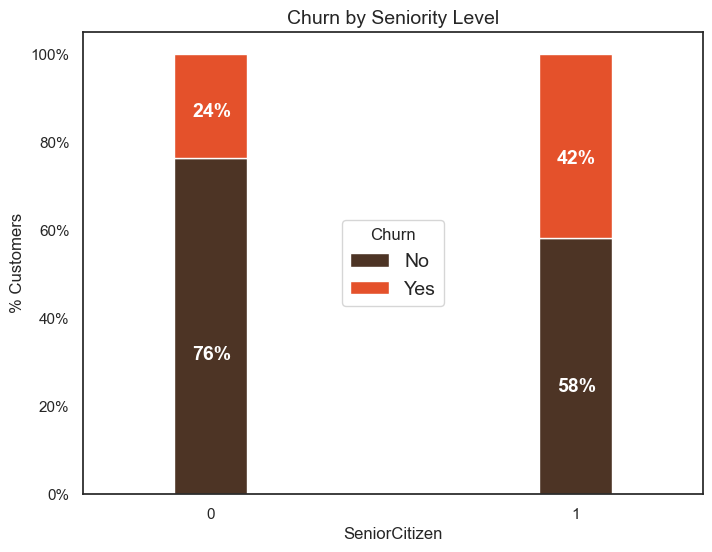
ii.) Churn by Contract Type: Similar to what we saw in the correlation plot, the customers who have a month to month contract have a very high churn rate.

colors = ['#4D3425','#E4512B']  
contract\_churn = telecom\_cust.groupby(['Contract','Churn']).size().unstack()  
  
ax = (contract\_churn.T\*100.0 / contract\_churn.T.sum()).T.plot(kind='bar',  
 width = 0.3,  
 stacked = True,  
 rot = 0,   
 figsize = (10,6),  
 color = colors)  
ax.yaxis.set\_major\_formatter(mtick.PercentFormatter())  
ax.legend(loc='best',prop={'size':14},title = 'Churn')  
ax.set\_ylabel('% Customers',size = 14)  
ax.set\_title('Churn by Contract Type',size = 14)  
  
# Code to add the data labels on the stacked bar chart  
for p in ax.patches:  
 width, height = p.get\_width(), p.get\_height()  
 x, y = p.get\_xy()   
 ax.annotate('{:.0f}%'.format(height), (p.get\_x()+.25\*width, p.get\_y()+.4\*height),  
 color = 'white',  
 weight = 'bold',  
 size = 14)



iii.) Churn by Seniority: Senior Citizens have almost double the churn rate than younger population.

colors = ['#4D3425','#E4512B']  
seniority\_churn = telecom\_cust.groupby(['SeniorCitizen','Churn']).size().unstack()  
  
ax = (seniority\_churn.T\*100.0 / seniority\_churn.T.sum()).T.plot(kind='bar',  
 width = 0.2,  
 stacked = True,  
 rot = 0,   
 figsize = (8,6),  
 color = colors)  
ax.yaxis.set\_major\_formatter(mtick.PercentFormatter())  
ax.legend(loc='center',prop={'size':14},title = 'Churn')  
ax.set\_ylabel('% Customers')  
ax.set\_title('Churn by Seniority Level',size = 14)  
  
# Code to add the data labels on the stacked bar chart  
for p in ax.patches:  
 width, height = p.get\_width(), p.get\_height()  
 x, y = p.get\_xy()   
 ax.annotate('{:.0f}%'.format(height), (p.get\_x()+.25\*width, p.get\_y()+.4\*height),  
 color = 'white',  
 weight = 'bold',size =14)

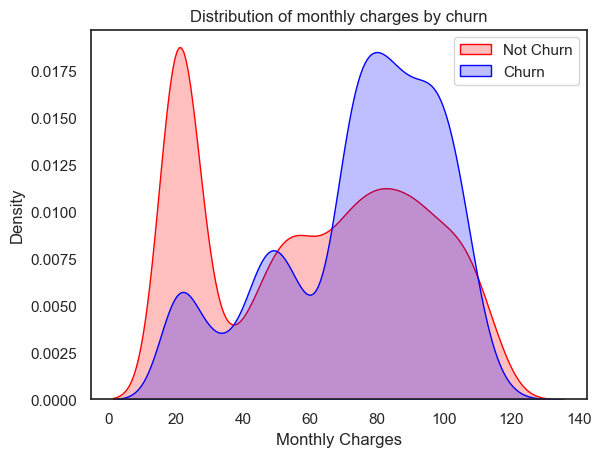


iv.) Churn by Monthly Charges: Higher % of customers churn when the monthly charges are high.

ax = sns.kdeplot(telecom\_cust.MonthlyCharges[(telecom\_cust["Churn"] == 'No') ],  
 color="Red", shade = True)  
ax = sns.kdeplot(telecom\_cust.MonthlyCharges[(telecom\_cust["Churn"] == 'Yes') ],  
 ax =ax, color="Blue", shade= True)  
ax.legend(["Not Churn","Churn"],loc='upper right')  
ax.set\_ylabel('Density')  
ax.set\_xlabel('Monthly Charges')  
ax.set\_title('Distribution of monthly charges by churn')

C:\Users\Administrator\AppData\Local\Temp\ipykernel\_14052\2862132292.py:1: FutureWarning:   
  
`shade` is now deprecated in favor of `fill`; setting `fill=True`.  
This will become an error in seaborn v0.14.0; please update your code.  
  
 ax = sns.kdeplot(telecom\_cust.MonthlyCharges[(telecom\_cust["Churn"] == 'No') ],  
C:\Users\Administrator\AppData\Local\Temp\ipykernel\_14052\2862132292.py:3: FutureWarning:   
  
`shade` is now deprecated in favor of `fill`; setting `fill=True`.  
This will become an error in seaborn v0.14.0; please update your code.  
  
 ax = sns.kdeplot(telecom\_cust.MonthlyCharges[(telecom\_cust["Churn"] == 'Yes') ],

Text(0.5, 1.0, 'Distribution of monthly charges by churn')

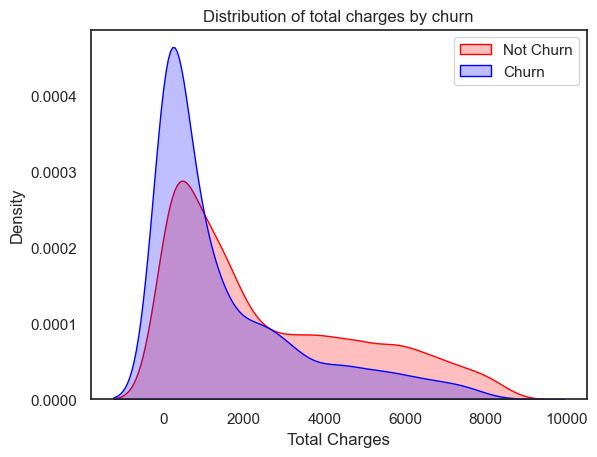


v.) Churn by Total Charges: It seems that there is higer churn when the total charges are lower.

ax = sns.kdeplot(telecom\_cust.TotalCharges[(telecom\_cust["Churn"] == 'No') ],  
 color="Red", shade = True)  
ax = sns.kdeplot(telecom\_cust.TotalCharges[(telecom\_cust["Churn"] == 'Yes') ],  
 ax =ax, color="Blue", shade= True)  
ax.legend(["Not Churn","Churn"],loc='upper right')  
ax.set\_ylabel('Density')  
ax.set\_xlabel('Total Charges')  
ax.set\_title('Distribution of total charges by churn')

C:\Users\Administrator\AppData\Local\Temp\ipykernel\_14052\3097405637.py:1: FutureWarning:   
  
`shade` is now deprecated in favor of `fill`; setting `fill=True`.  
This will become an error in seaborn v0.14.0; please update your code.  
  
 ax = sns.kdeplot(telecom\_cust.TotalCharges[(telecom\_cust["Churn"] == 'No') ],  
C:\Users\Administrator\AppData\Local\Temp\ipykernel\_14052\3097405637.py:3: FutureWarning:   
  
`shade` is now deprecated in favor of `fill`; setting `fill=True`.  
This will become an error in seaborn v0.14.0; please update your code.  
  
 ax = sns.kdeplot(telecom\_cust.TotalCharges[(telecom\_cust["Churn"] == 'Yes') ],

Text(0.5, 1.0, 'Distribution of total charges by churn')



After going through the above EDA we will develop some predictive models and compare them.

We will develop Logistic Regression, Random Forest, SVM, ADA Boost and XG Boost

1. Logistic Regression

# We will use the data frame where we had created dummy variables  
y = df\_dummies['Churn'].values  
X = df\_dummies.drop(columns = ['Churn'])  
  
# Scaling all the variables to a range of 0 to 1  
from sklearn.preprocessing import MinMaxScaler  
features = X.columns.values  
scaler = MinMaxScaler(feature\_range = (0,1))  
scaler.fit(X)  
X = pd.DataFrame(scaler.transform(X))  
X.columns = features

It is important to scale the variables in logistic regression so that all of them are within a range of 0 to 1. This helped me improve the accuracy from 79.7% to 80.7%. Further, you will notice below that the importance of variables is also aligned with what we are seeing in Random Forest algorithm and the EDA we conducted above.

# Create Train & Test Data  
from sklearn.model\_selection import train\_test\_split  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=101)

# Running logistic regression model  
from sklearn.linear\_model import LogisticRegression  
model = LogisticRegression()  
result = model.fit(X\_train, y\_train)

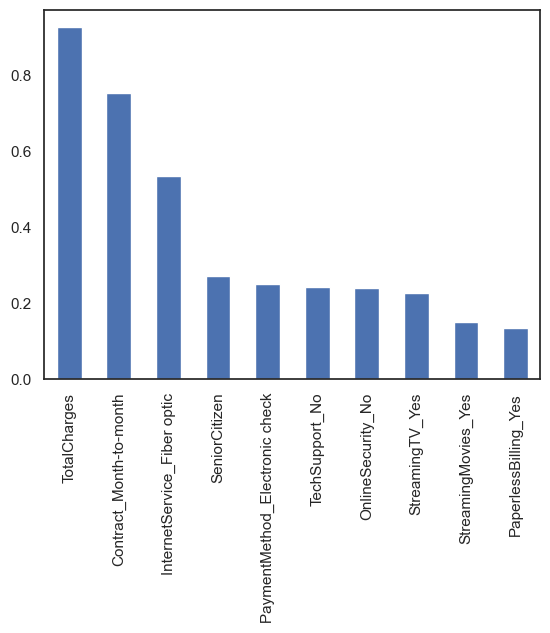
from sklearn import metrics  
prediction\_test = model.predict(X\_test)  
# Print the prediction accuracy  
print (metrics.accuracy\_score(y\_test, prediction\_test))

0.8075829383886256

logit\_model = metrics.accuracy\_score(y\_test, prediction\_test)

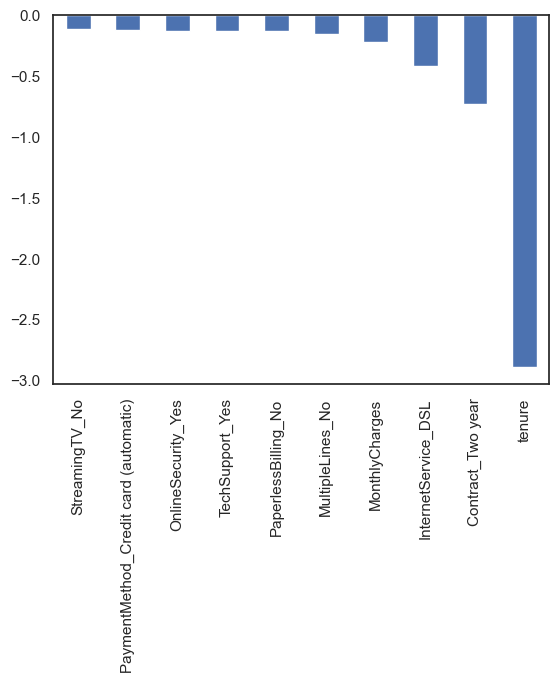
# To get the weights of all the variables  
weights = pd.Series(model.coef\_[0],  
 index=X.columns.values)  
print (weights.sort\_values(ascending = False)[:10].plot(kind='bar'))

Axes(0.125,0.11;0.775x0.77)



print(weights.sort\_values(ascending = False)[-10:].plot(kind='bar'))

Axes(0.125,0.11;0.775x0.77)



Observations

We can see that some variables have a negative relation to our predicted variable (Churn), while some have positive relation. Negative relation means that likeliness of churn decreases with that variable. Let us summarize some of the interesting features below:

As we saw in our EDA, having a 2 month contract reduces chances of churn. 2 month contract along with tenure have the most negative relation with Churn as predicted by logistic regressions

Having DSL internet service also reduces the proability of Churn

Lastly, total charges, monthly contracts, fibre optic internet services and seniority can lead to higher churn rates. This is interesting because although fibre optic services are faster, customers are likely to churn because of it. I think we need to explore more to better understad why this is happening.

Any hypothesis on the above would be really helpful!

2. Random Forest

from sklearn.ensemble import RandomForestClassifier  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=101)  
model\_rf = RandomForestClassifier(n\_estimators=1000 , oob\_score = True, n\_jobs = -1,  
 random\_state =50, max\_features = "auto",  
 max\_leaf\_nodes = 30)  
model\_rf.fit(X\_train, y\_train)  
  
# Make predictions  
prediction\_test = model\_rf.predict(X\_test)  
print (metrics.accuracy\_score(y\_test, prediction\_test))

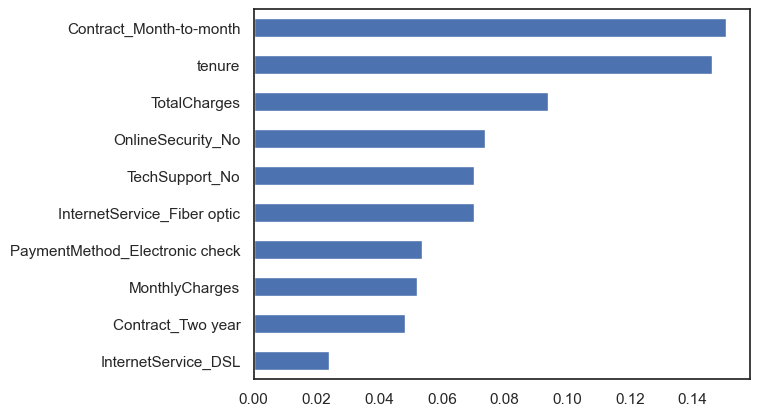
C:\ProgramData\anaconda3\lib\site-packages\sklearn\ensemble\\_forest.py:424: FutureWarning: `max\_features='auto'` has been deprecated in 1.1 and will be removed in 1.3. To keep the past behaviour, explicitly set `max\_features='sqrt'` or remove this parameter as it is also the default value for RandomForestClassifiers and ExtraTreesClassifiers.  
 warn(

0.8088130774697939

rf\_model = metrics.accuracy\_score(y\_test, prediction\_test)

importances = model\_rf.feature\_importances\_  
weights = pd.Series(importances,  
 index=X.columns.values)  
weights.sort\_values()[-10:].plot(kind = 'barh')

<Axes: >



Observations:

From random forest algorithm, monthly contract, tenure and total charges are the most important predictor variables to predict churn.

The results from random forest are very similar to that of the logistic regression and in line to what we had expected from our EDA

3. Support Vecor Machine (SVM)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=99)

from sklearn.svm import SVC  
  
model.svm = SVC(kernel='linear')   
model.svm.fit(X\_train,y\_train)  
preds = model.svm.predict(X\_test)  
metrics.accuracy\_score(y\_test, preds)

0.820184790334044

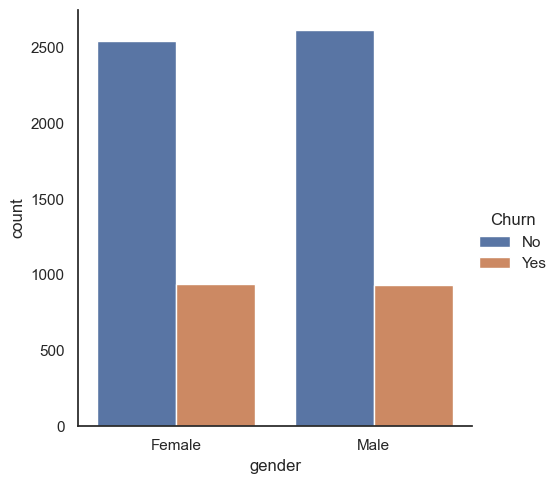
svm\_model = metrics.accuracy\_score(y\_test, preds)

# Create the Confusion matrix  
from sklearn.metrics import classification\_report, confusion\_matrix   
print(confusion\_matrix(y\_test,preds))

[[953 89]  
 [164 201]]

Wth SVM I was able to increase the accuracy to upto 82%. However, we need to take a deeper look at the true positive and true negative rates, including the Area Under the Curve (AUC) for a better prediction. I will explore this soon. Stay Tuned!

ax1 = sns.catplot(x="gender", kind="count", hue="Churn", data=telecom\_cust,  
 estimator=lambda x: sum(x==0)\*100.0/len(x))  
#ax1.yaxis.set\_major\_formatter(mtick.PercentFormatter())



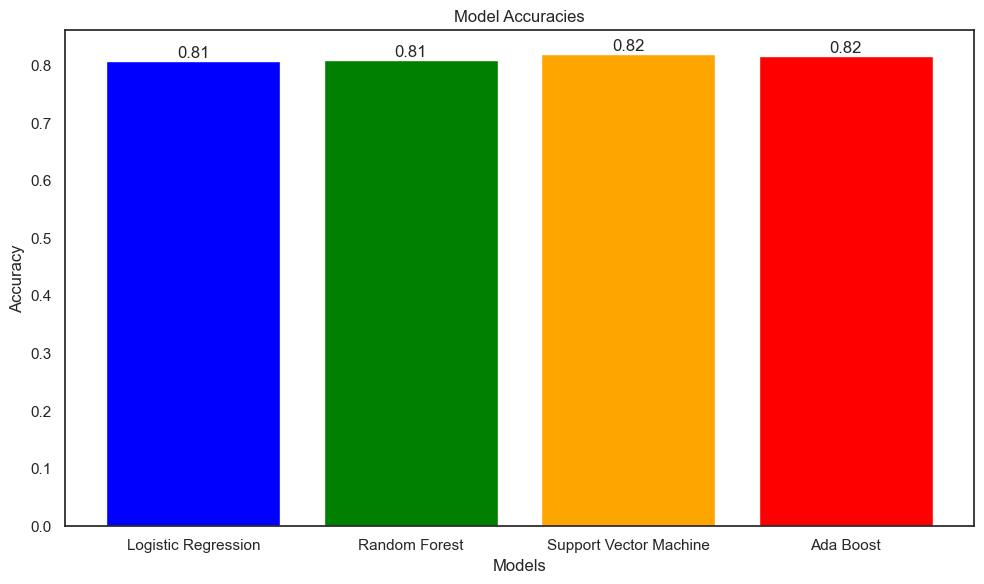
4. ADA Boost

# AdaBoost Algorithm  
from sklearn.ensemble import AdaBoostClassifier  
model = AdaBoostClassifier()  
# n\_estimators = 50 (default value)   
# base\_estimator = DecisionTreeClassifier (default value)  
model.fit(X\_train,y\_train)  
preds = model.predict(X\_test)  
metrics.accuracy\_score(y\_test, preds)

0.8159203980099502

adab\_model = metrics.accuracy\_score(y\_test, preds)

import matplotlib.pyplot as plt  
  
# Model names and accuracies  
models = ["Logistic Regression", "Random Forest", "Support Vector Machine", "Ada Boost"]  
accuracies = [logit\_model, rf\_model, svm\_model, adab\_model]  
  
# Create a bar chart  
plt.figure(figsize=(10, 6))  
plt.bar(models, accuracies, color=['blue', 'green', 'orange', 'red'])  
plt.xlabel("Models")  
plt.ylabel("Accuracy")  
plt.title("Model Accuracies")  
  
# Adding accuracy values on top of the bars  
for i, accuracy in enumerate(accuracies):  
 plt.text(i, accuracy, f"{accuracy:.2f}", ha='center', va='bottom')  
  
# Show the chart  
plt.tight\_layout()  
plt.show()



EDA and Prediction

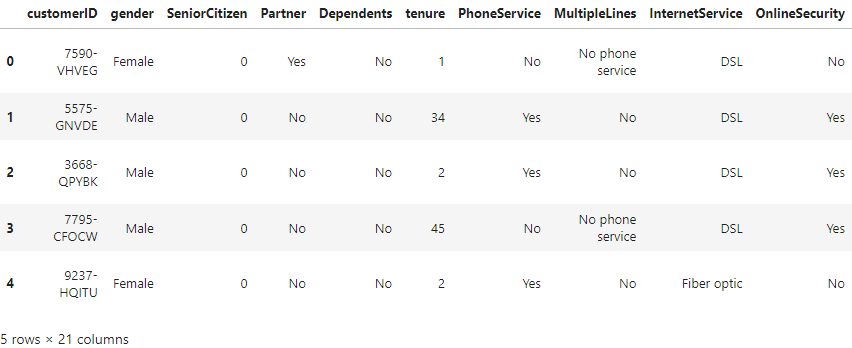
Churn is a one of the biggest problem in the telecom industry. Research has shown that the average monthly churn rate among the top 4 wireless carriers in the US is 1.9% - 2%.

import numpy as np # linear algebra  
import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)  
import seaborn as sns # For creating plots  
import matplotlib.ticker as mtick # For specifying the axes tick format   
import matplotlib.pyplot as plt  
  
sns.set(style = 'white')

Let us read the data file in the python notebook

telecom\_cust = pd.read\_csv('Dataset.csv')

telecom\_cust.head()



telecom\_cust.columns.values

array(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',  
 'tenure', 'PhoneService', 'MultipleLines', 'InternetService',  
 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',  
 'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract',  
 'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges',  
 'TotalCharges', 'Churn'], dtype=object)

Let's explore the data to see if there are any missing values.

# Checking the data types of all the columns  
telecom\_cust.dtypes

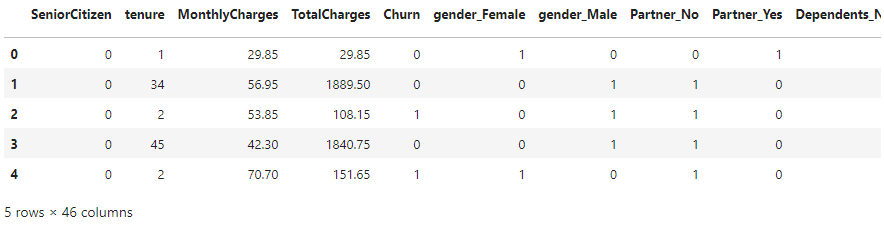
customerID object  
gender object  
SeniorCitizen int64  
Partner object  
Dependents object  
tenure int64  
PhoneService object  
MultipleLines object  
InternetService object  
OnlineSecurity object  
OnlineBackup object  
DeviceProtection object  
TechSupport object  
StreamingTV object  
StreamingMovies object  
Contract object  
PaperlessBilling object  
PaymentMethod object  
MonthlyCharges float64  
TotalCharges object  
Churn object  
dtype: object

# Converting Total Charges to a numerical data type.  
telecom\_cust.TotalCharges = pd.to\_numeric(telecom\_cust.TotalCharges, errors='coerce')  
telecom\_cust.isnull().sum()

customerID 0  
gender 0  
SeniorCitizen 0  
Partner 0  
Dependents 0  
tenure 0  
PhoneService 0  
MultipleLines 0  
InternetService 0  
OnlineSecurity 0  
OnlineBackup 0  
DeviceProtection 0  
TechSupport 0  
StreamingTV 0  
StreamingMovies 0  
Contract 0  
PaperlessBilling 0  
PaymentMethod 0  
MonthlyCharges 0  
TotalCharges 11  
Churn 0  
dtype: int64

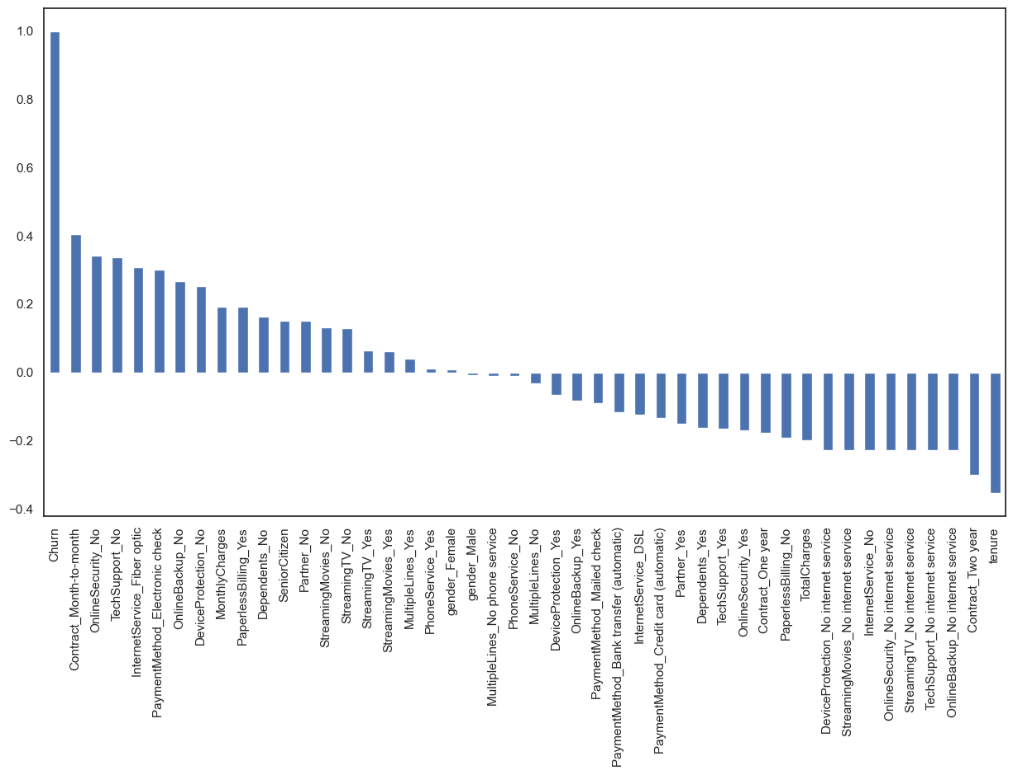
After looking at the above output, we can say that there are 11 missing values for Total Charges. Let us replace remove these 11 rows from our data set

#Removing missing values   
telecom\_cust.dropna(inplace = True)  
#Remove customer IDs from the data set  
df2 = telecom\_cust.iloc[:,1:]  
#Convertin the predictor variable in a binary numeric variable  
df2['Churn'].replace(to\_replace='Yes', value=1, inplace=True)  
df2['Churn'].replace(to\_replace='No', value=0, inplace=True)  
  
#Let's convert all the categorical variables into dummy variables  
df\_dummies = pd.get\_dummies(df2)  
df\_dummies.head()



#Get Correlation of "Churn" with other variables:  
plt.figure(figsize=(15,8))  
df\_dummies.corr()['Churn'].sort\_values(ascending = False).plot(kind='bar')

<Axes: >



Month to month contracts, absence of online security and tech support seem to be positively correlated with churn. While, tenure, two year contracts seem to be negatively correlated with churn.

Interestingly, services such as Online security, streaming TV, online backup, tech support, etc. without internet connection seem to be negatively related to churn.

We will explore the patterns for the above correlations below before we delve into modelling and identifying the important variables.

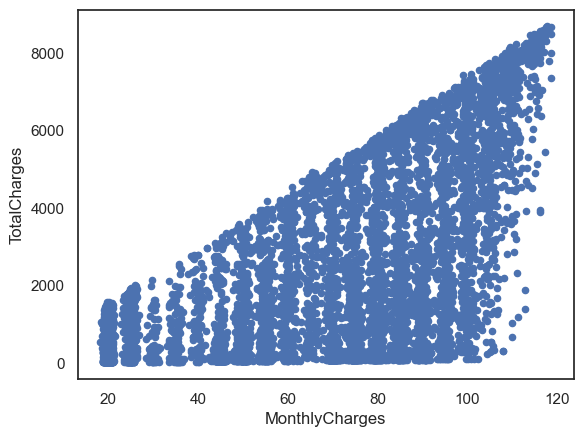
Now let's take a quick look at the relation between monthly and total charges

We will observe that the total charges increases as the monthly bill for a customer increases.

telecom\_cust[['MonthlyCharges', 'TotalCharges']].plot.scatter(x = 'MonthlyCharges',  
 y='TotalCharges')

C:\ProgramData\anaconda3\lib\site-packages\pandas\plotting\\_matplotlib\core.py:1070: UserWarning: No data for colormapping provided via 'c'. Parameters 'cmap' will be ignored  
 scatter = ax.scatter(

<Axes: xlabel='MonthlyCharges', ylabel='TotalCharges'>



Finally, let's take a look at out predictor variable (Churn) and understand its interaction with other important variables as was found out in the correlation plot.

Lets first look at the churn rate in our data

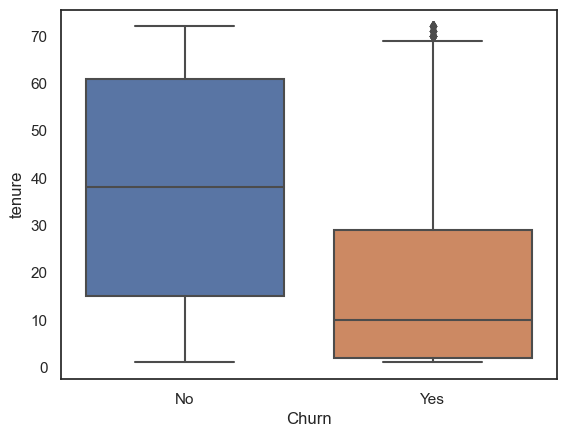
In our data, 74% of the customers do not churn. Clearly the data is skewed as we would expect a large majority of the customers to not churn. This is important to keep in mind for our modelling as skeweness could lead to a lot of false negatives. We will see in the modelling section on how to avoid skewness in the data.

Lets now explore the churn rate by tenure, seniority, contract type, monthly charges and total charges to see how it varies by these variables.

i.) Churn vs Tenure: As we can see form the below plot, the customers who do not churn, they tend to stay for a longer tenure with the telecom company.

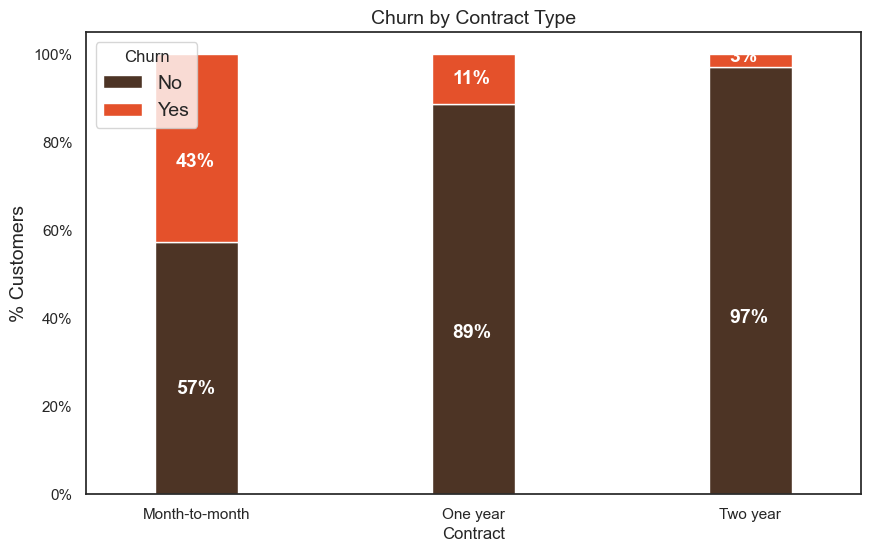
sns.boxplot(x = telecom\_cust.Churn, y = telecom\_cust.tenure)

<Axes: xlabel='Churn', ylabel='tenure'>



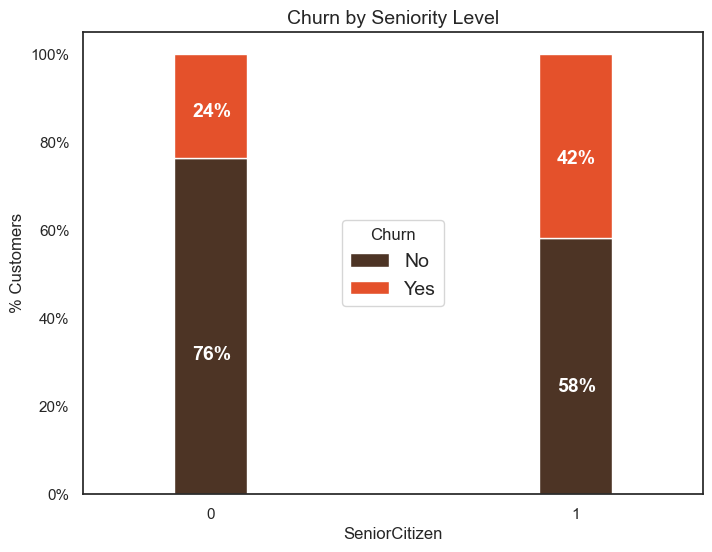
ii.) Churn by Contract Type: Similar to what we saw in the correlation plot, the customers who have a month to month contract have a very high churn rate.

colors = ['#4D3425','#E4512B']  
contract\_churn = telecom\_cust.groupby(['Contract','Churn']).size().unstack()  
  
ax = (contract\_churn.T\*100.0 / contract\_churn.T.sum()).T.plot(kind='bar',  
 width = 0.3,  
 stacked = True,  
 rot = 0,   
 figsize = (10,6),  
 color = colors)  
ax.yaxis.set\_major\_formatter(mtick.PercentFormatter())  
ax.legend(loc='best',prop={'size':14},title = 'Churn')  
ax.set\_ylabel('% Customers',size = 14)  
ax.set\_title('Churn by Contract Type',size = 14)  
  
# Code to add the data labels on the stacked bar chart  
for p in ax.patches:  
 width, height = p.get\_width(), p.get\_height()  
 x, y = p.get\_xy()   
 ax.annotate('{:.0f}%'.format(height), (p.get\_x()+.25\*width, p.get\_y()+.4\*height),  
 color = 'white',  
 weight = 'bold',  
 size = 14)



iii.) Churn by Seniority: Senior Citizens have almost double the churn rate than younger population.

colors = ['#4D3425','#E4512B']  
seniority\_churn = telecom\_cust.groupby(['SeniorCitizen','Churn']).size().unstack()  
  
ax = (seniority\_churn.T\*100.0 / seniority\_churn.T.sum()).T.plot(kind='bar',  
 width = 0.2,  
 stacked = True,  
 rot = 0,   
 figsize = (8,6),  
 color = colors)  
ax.yaxis.set\_major\_formatter(mtick.PercentFormatter())  
ax.legend(loc='center',prop={'size':14},title = 'Churn')  
ax.set\_ylabel('% Customers')  
ax.set\_title('Churn by Seniority Level',size = 14)  
  
# Code to add the data labels on the stacked bar chart  
for p in ax.patches:  
 width, height = p.get\_width(), p.get\_height()  
 x, y = p.get\_xy()   
 ax.annotate('{:.0f}%'.format(height), (p.get\_x()+.25\*width, p.get\_y()+.4\*height),  
 color = 'white',  
 weight = 'bold',size =14)

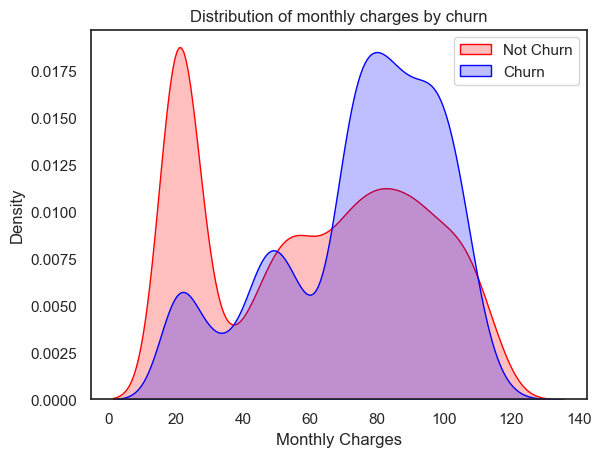


iv.) Churn by Monthly Charges: Higher % of customers churn when the monthly charges are high.

ax = sns.kdeplot(telecom\_cust.MonthlyCharges[(telecom\_cust["Churn"] == 'No') ],  
 color="Red", shade = True)  
ax = sns.kdeplot(telecom\_cust.MonthlyCharges[(telecom\_cust["Churn"] == 'Yes') ],  
 ax =ax, color="Blue", shade= True)  
ax.legend(["Not Churn","Churn"],loc='upper right')  
ax.set\_ylabel('Density')  
ax.set\_xlabel('Monthly Charges')  
ax.set\_title('Distribution of monthly charges by churn')

C:\Users\Administrator\AppData\Local\Temp\ipykernel\_14052\2862132292.py:1: FutureWarning:   
  
`shade` is now deprecated in favor of `fill`; setting `fill=True`.  
This will become an error in seaborn v0.14.0; please update your code.  
  
 ax = sns.kdeplot(telecom\_cust.MonthlyCharges[(telecom\_cust["Churn"] == 'No') ],  
C:\Users\Administrator\AppData\Local\Temp\ipykernel\_14052\2862132292.py:3: FutureWarning:   
  
`shade` is now deprecated in favor of `fill`; setting `fill=True`.  
This will become an error in seaborn v0.14.0; please update your code.  
  
 ax = sns.kdeplot(telecom\_cust.MonthlyCharges[(telecom\_cust["Churn"] == 'Yes') ],

Text(0.5, 1.0, 'Distribution of monthly charges by churn')

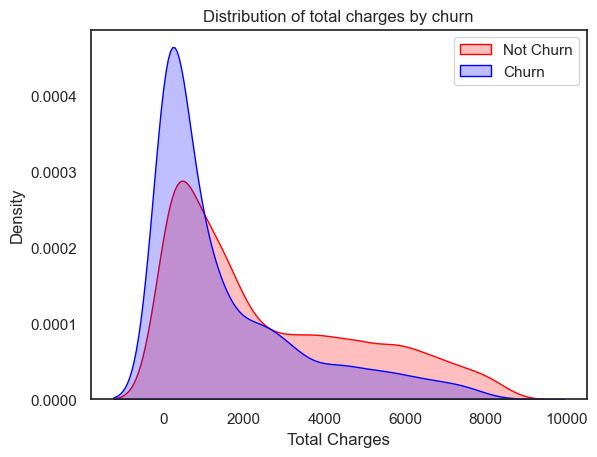


v.) Churn by Total Charges: It seems that there is higer churn when the total charges are lower.

ax = sns.kdeplot(telecom\_cust.TotalCharges[(telecom\_cust["Churn"] == 'No') ],  
 color="Red", shade = True)  
ax = sns.kdeplot(telecom\_cust.TotalCharges[(telecom\_cust["Churn"] == 'Yes') ],  
 ax =ax, color="Blue", shade= True)  
ax.legend(["Not Churn","Churn"],loc='upper right')  
ax.set\_ylabel('Density')  
ax.set\_xlabel('Total Charges')  
ax.set\_title('Distribution of total charges by churn')

C:\Users\Administrator\AppData\Local\Temp\ipykernel\_14052\3097405637.py:1: FutureWarning:   
  
`shade` is now deprecated in favor of `fill`; setting `fill=True`.  
This will become an error in seaborn v0.14.0; please update your code.  
  
 ax = sns.kdeplot(telecom\_cust.TotalCharges[(telecom\_cust["Churn"] == 'No') ],  
C:\Users\Administrator\AppData\Local\Temp\ipykernel\_14052\3097405637.py:3: FutureWarning:   
  
`shade` is now deprecated in favor of `fill`; setting `fill=True`.  
This will become an error in seaborn v0.14.0; please update your code.  
  
 ax = sns.kdeplot(telecom\_cust.TotalCharges[(telecom\_cust["Churn"] == 'Yes') ],

Text(0.5, 1.0, 'Distribution of total charges by churn')



After going through the above EDA we will develop some predictive models and compare them.

We will develop Logistic Regression, Random Forest, SVM, ADA Boost and XG Boost

1. Logistic Regression

# We will use the data frame where we had created dummy variables  
y = df\_dummies['Churn'].values  
X = df\_dummies.drop(columns = ['Churn'])  
  
# Scaling all the variables to a range of 0 to 1  
from sklearn.preprocessing import MinMaxScaler  
features = X.columns.values  
scaler = MinMaxScaler(feature\_range = (0,1))  
scaler.fit(X)  
X = pd.DataFrame(scaler.transform(X))  
X.columns = features

It is important to scale the variables in logistic regression so that all of them are within a range of 0 to 1. This helped me improve the accuracy from 79.7% to 80.7%. Further, you will notice below that the importance of variables is also aligned with what we are seeing in Random Forest algorithm and the EDA we conducted above.

# Create Train & Test Data  
from sklearn.model\_selection import train\_test\_split  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=101)

# Running logistic regression model  
from sklearn.linear\_model import LogisticRegression  
model = LogisticRegression()  
result = model.fit(X\_train, y\_train)

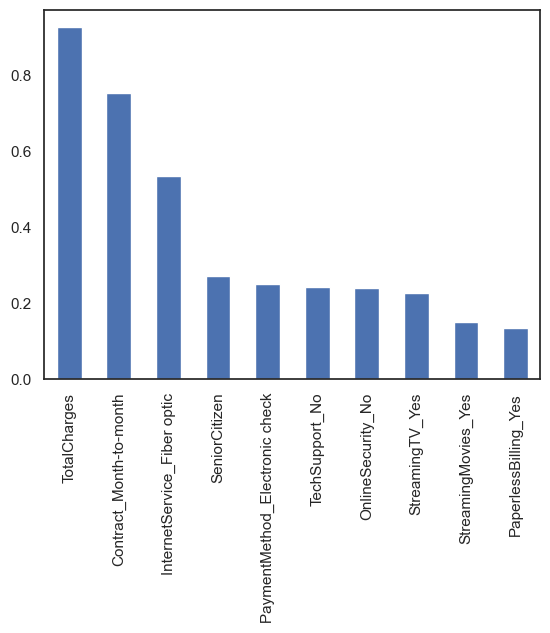
from sklearn import metrics  
prediction\_test = model.predict(X\_test)  
# Print the prediction accuracy  
print (metrics.accuracy\_score(y\_test, prediction\_test))

0.8075829383886256

logit\_model = metrics.accuracy\_score(y\_test, prediction\_test)

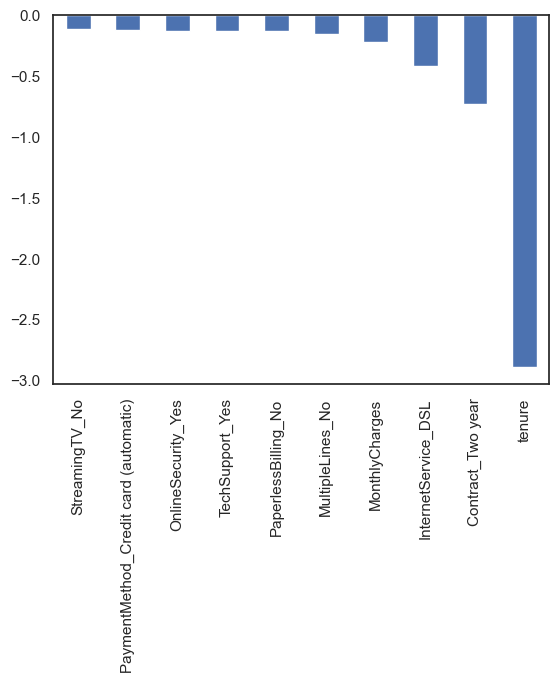
# To get the weights of all the variables  
weights = pd.Series(model.coef\_[0],  
 index=X.columns.values)  
print (weights.sort\_values(ascending = False)[:10].plot(kind='bar'))

Axes(0.125,0.11;0.775x0.77)



print(weights.sort\_values(ascending = False)[-10:].plot(kind='bar'))

Axes(0.125,0.11;0.775x0.77)



Observations

We can see that some variables have a negative relation to our predicted variable (Churn), while some have positive relation. Negative relation means that likeliness of churn decreases with that variable. Let us summarize some of the interesting features below:

As we saw in our EDA, having a 2 month contract reduces chances of churn. 2 month contract along with tenure have the most negative relation with Churn as predicted by logistic regressions

Having DSL internet service also reduces the proability of Churn

Lastly, total charges, monthly contracts, fibre optic internet services and seniority can lead to higher churn rates. This is interesting because although fibre optic services are faster, customers are likely to churn because of it. I think we need to explore more to better understad why this is happening.

Any hypothesis on the above would be really helpful!

2. Random Forest

from sklearn.ensemble import RandomForestClassifier  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=101)  
model\_rf = RandomForestClassifier(n\_estimators=1000 , oob\_score = True, n\_jobs = -1,  
 random\_state =50, max\_features = "auto",  
 max\_leaf\_nodes = 30)  
model\_rf.fit(X\_train, y\_train)  
  
# Make predictions  
prediction\_test = model\_rf.predict(X\_test)  
print (metrics.accuracy\_score(y\_test, prediction\_test))

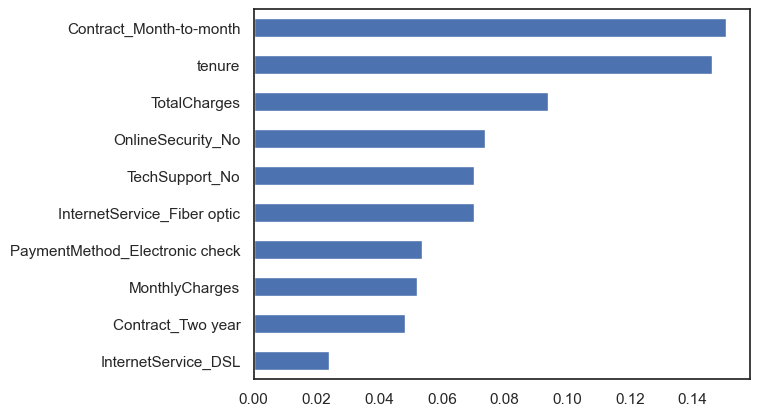
C:\ProgramData\anaconda3\lib\site-packages\sklearn\ensemble\\_forest.py:424: FutureWarning: `max\_features='auto'` has been deprecated in 1.1 and will be removed in 1.3. To keep the past behaviour, explicitly set `max\_features='sqrt'` or remove this parameter as it is also the default value for RandomForestClassifiers and ExtraTreesClassifiers.  
 warn(

0.8088130774697939

rf\_model = metrics.accuracy\_score(y\_test, prediction\_test)

importances = model\_rf.feature\_importances\_  
weights = pd.Series(importances,  
 index=X.columns.values)  
weights.sort\_values()[-10:].plot(kind = 'barh')

<Axes: >



Observations:

From random forest algorithm, monthly contract, tenure and total charges are the most important predictor variables to predict churn.

The results from random forest are very similar to that of the logistic regression and in line to what we had expected from our EDA

3. Support Vecor Machine (SVM)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=99)

from sklearn.svm import SVC  
  
model.svm = SVC(kernel='linear')   
model.svm.fit(X\_train,y\_train)  
preds = model.svm.predict(X\_test)  
metrics.accuracy\_score(y\_test, preds)

0.820184790334044

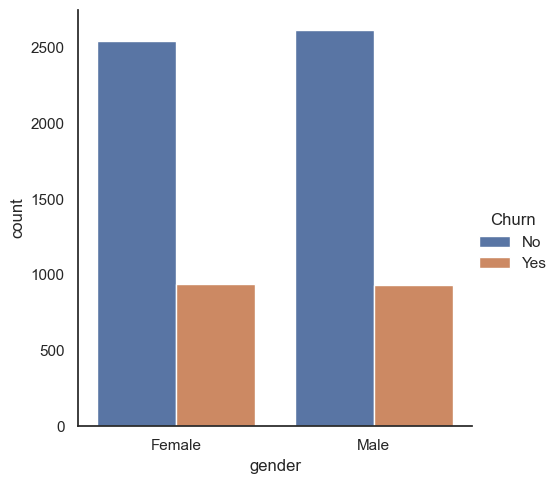
svm\_model = metrics.accuracy\_score(y\_test, preds)

# Create the Confusion matrix  
from sklearn.metrics import classification\_report, confusion\_matrix   
print(confusion\_matrix(y\_test,preds))

[[953 89]  
 [164 201]]

Wth SVM I was able to increase the accuracy to upto 82%. However, we need to take a deeper look at the true positive and true negative rates, including the Area Under the Curve (AUC) for a better prediction. I will explore this soon. Stay Tuned!

ax1 = sns.catplot(x="gender", kind="count", hue="Churn", data=telecom\_cust,  
 estimator=lambda x: sum(x==0)\*100.0/len(x))  
#ax1.yaxis.set\_major\_formatter(mtick.PercentFormatter())



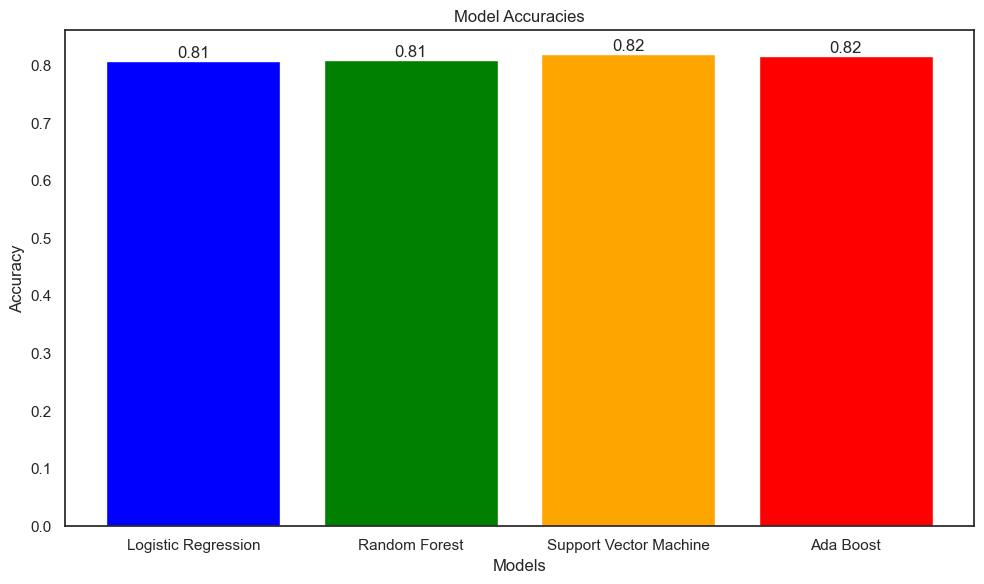
4. ADA Boost

# AdaBoost Algorithm  
from sklearn.ensemble import AdaBoostClassifier  
model = AdaBoostClassifier()  
# n\_estimators = 50 (default value)   
# base\_estimator = DecisionTreeClassifier (default value)  
model.fit(X\_train,y\_train)  
preds = model.predict(X\_test)  
metrics.accuracy\_score(y\_test, preds)

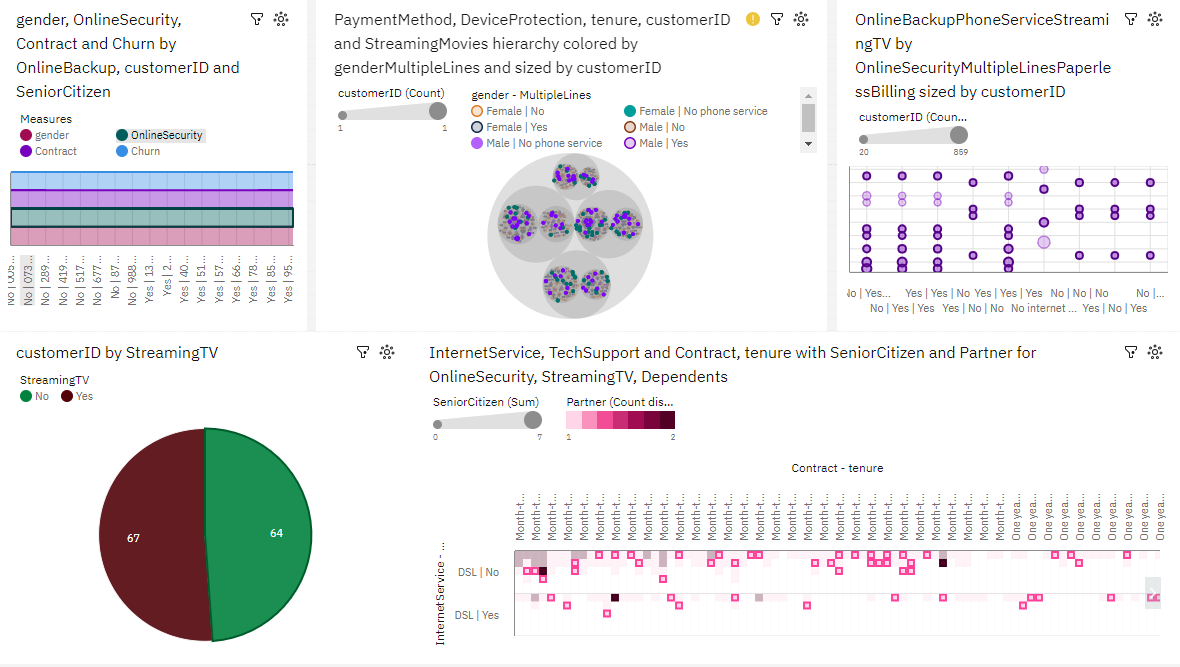
0.8159203980099502

adab\_model = metrics.accuracy\_score(y\_test, preds)

import matplotlib.pyplot as plt  
  
# Model names and accuracies  
models = ["Logistic Regression", "Random Forest", "Support Vector Machine", "Ada Boost"]  
accuracies = [logit\_model, rf\_model, svm\_model, adab\_model]  
  
# Create a bar chart  
plt.figure(figsize=(10, 6))  
plt.bar(models, accuracies, color=['blue', 'green', 'orange', 'red'])  
plt.xlabel("Models")  
plt.ylabel("Accuracy")  
plt.title("Model Accuracies")  
  
# Adding accuracy values on top of the bars  
for i, accuracy in enumerate(accuracies):  
 plt.text(i, accuracy, f"{accuracy:.2f}", ha='center', va='bottom')  
  
# Show the chart  
plt.tight\_layout()  
plt.show()



IBM COGNOS ANALYTICS :



I. Introduction

The dataset under examination pertains to a telecommunications company's customer information and their churn status. Churn, in this context, refers to the phenomenon where customers discontinue their services with the company. Understanding the factors that influence churn is crucial for business sustainability and growth.

II. Customer Demographics and Churn

Gender: Analysis reveals that gender does not have a significant impact on churn. The churn rate is relatively similar for both male and female customers.

Senior Citizen: The churn rate among senior citizens is notably higher than that of non-senior customers. This suggests that the company may need to explore strategies to retain senior citizen customers.

Partner and Dependents: Customers with partners and dependents are indeed less likely to churn. Those who have a partner and dependents tend to have a stronger commitment to the service, resulting in lower churn rates.

III. Customer Tenure and Churn

Tenure Distribution: The distribution of customer tenure reveals that a significant number of customers are relatively new, with shorter tenures, while a smaller proportion have long tenures.

Churn vs. Tenure: There is a clear negative correlation between customer tenure and churn. Newer customers are more likely to churn, while longer-tenured customers tend to stay with the company.

IV. Service Usage and Churn

Phone Service: Surprisingly, having or not having phone service does not significantly impact churn. The churn rates are fairly similar for both groups.

Multiple Lines: Having multiple phone lines also does not seem to be a major factor in churn. Churn rates are similar for those with and without multiple lines.

Internet Service: The type of internet service plays a crucial role. Customers with fiber-optic internet are less likely to churn compared to those with DSL or no internet service.

V. Online Services and Churn

Online Security: Customers with online security services are indeed less likely to churn. This is a significant factor influencing churn.

Online Backup: Similarly, customers with online backup services have a lower churn rate.

Device Protection: Customers with device protection services also exhibit a lower churn rate.

Tech Support: Access to tech support services positively impacts customer retention.

VI. Entertainment Services and Churn

Streaming TV: Subscribing to streaming TV services does not significantly affect churn. The churn rates are similar for both users and non-users of this service.

Streaming Movies: Likewise, streaming movies do not seem to have a significant impact on customer churn.

VII. Contract and Billing Options and Churn

Contract Types: Customers on a month-to-month contract have higher churn rates compared to those on one-year or two-year contracts. Long-term contracts are more effective in retaining customers.

Paperless Billing: Opting for paperless billing has a slightly positive impact on churn reduction.

Payment Method: The choice of payment method does not appear to be a major factor affecting churn.

VIII. Financial Information and Churn

Monthly Charges Distribution: Monthly charges are positively correlated with churn. Customers with higher monthly charges are more likely to churn.

Churn vs. Monthly Charges: Analysis indicates a strong correlation between higher monthly charges and increased churn rates.

Total Charges: The distribution of total charges is influenced by customer tenure and contract type.

IX. Key Features to Churn

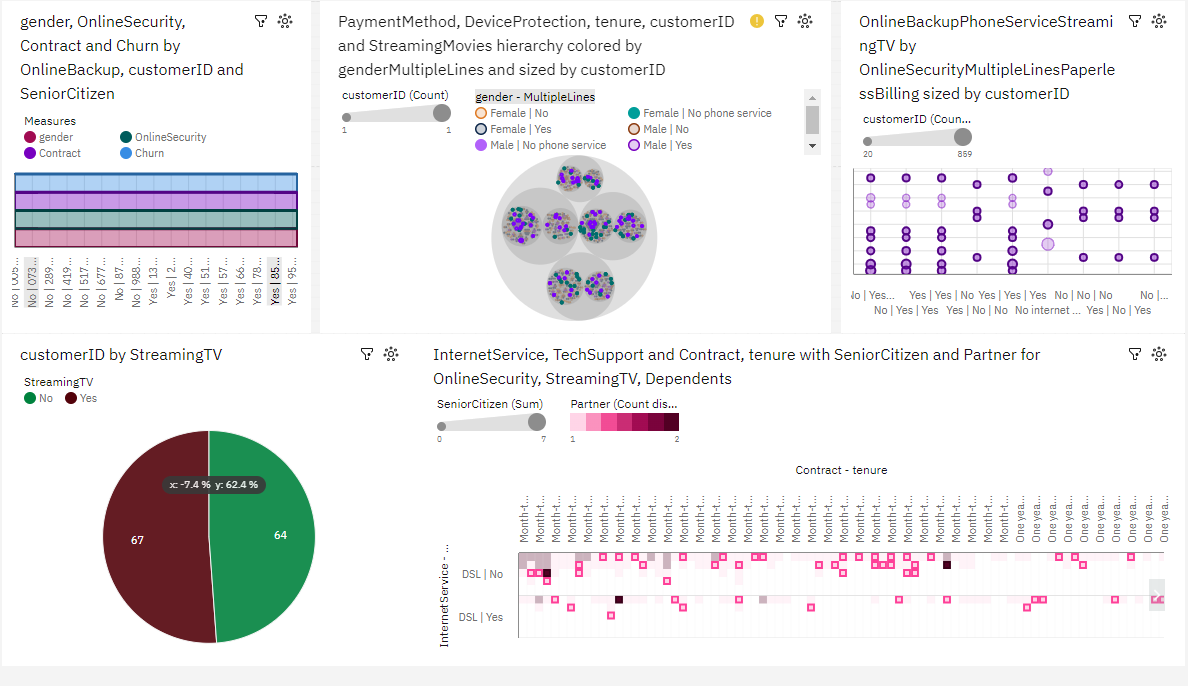
Key features that strongly influence churn include customer tenure, the availability of online security, online backup, device protection, and access to tech support services. Monthly charges also play a significant role in customer churn.

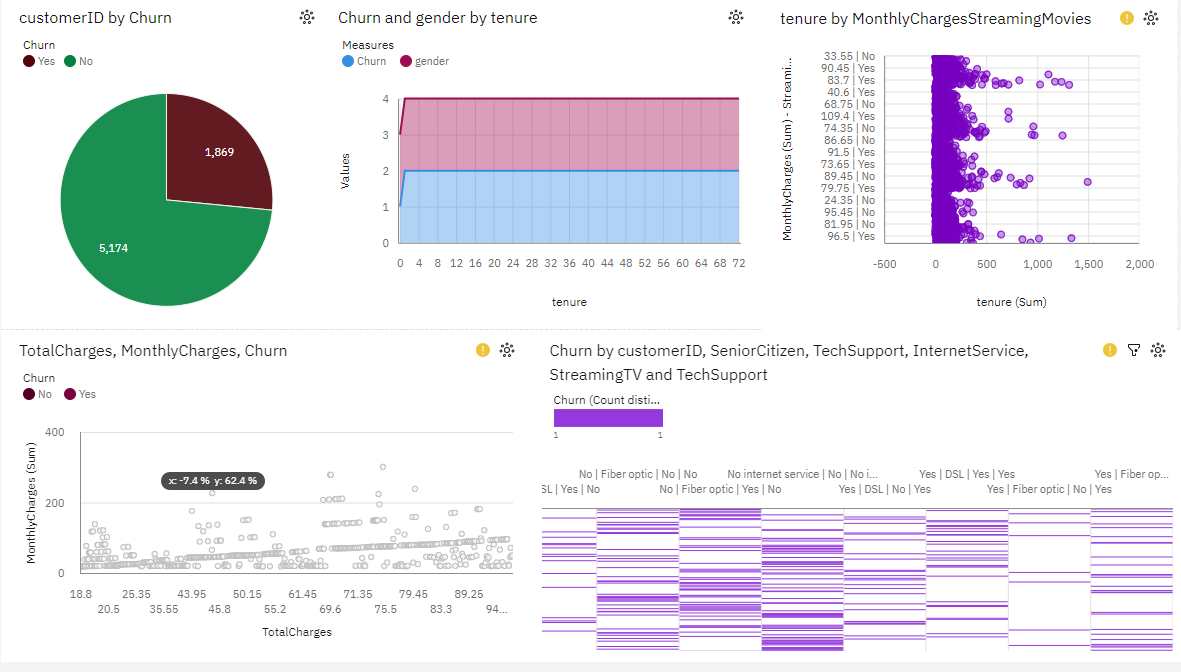
X. Recommendations

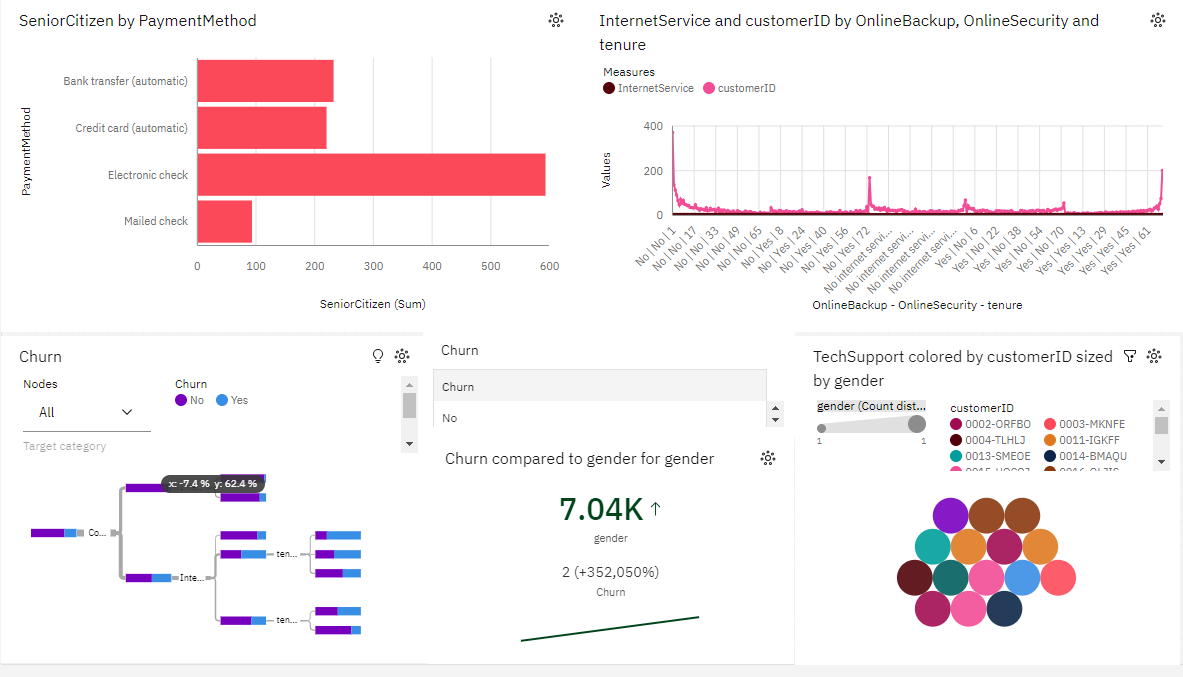
Implement customer retention strategies that specifically target senior citizens, as they have a higher churn rate.

Promote long-term contracts and the use of online security, online backup, device protection, and tech support services to reduce churn.

Consider pricing strategies to mitigate the impact of high monthly charges on churn.







XI. Conclusion

In conclusion, the analysis reveals that customer demographics, service usage, contract types, online services, and financial factors all contribute to customer churn. To enhance customer retention, it is crucial for the company to focus on these key factors and implement targeted strategies to address them