In [1]:
```
print("hello world")
```
hello world

In [2]:
```
import numpy as np
```
In [3]:
```
import pandas as pd
```
In [4]:
```
import seaborn as sns
```
In [5]:
```
import matplotlib.pyplot as plt
```
In [6]:
```
import matplotlib.ticker as mtick
```
In [7]:
```
telecom_cust_churn = pd.read_csv('F:/PRADEEP/CASE STUDY PYTHON/Churn.csv')
```
In [8]:
```
telecom_cust_churn
```
Out[8]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetS
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fibe
...	...	...	...	...	...	...	...	...	
7038	6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes	
7039	2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes	Fibe
7040	4801-JZAZL	Female	0	Yes	Yes	11	No	No phone service	
7041	8361-LTMKD	Male	1	Yes	No	4	Yes	Yes	Fibe
7042	3186-AJIEK	Male	0	No	No	66	Yes	No	Fibe

7043 rows × 21 columns

In [39]:
```
telecom_cust_churn.head()
```
Out[39]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetServi
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	D
1	5575-GNVDE	Male	0	No	No	34	Yes	No	D
2	3668-QPYBK	Male	0	No	No	2	Yes	No	D
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	D
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber op

5 rows × 21 columns

In [40]:
```
telecom_cust_churn.columns.values
```
Out[40]:
```
array(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
       'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract',
       'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges',
       'TotalCharges', 'Churn'], dtype=object)
```
In [14]:
```
telecom_cust_churn.head()
```
Out[14]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetServi
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	D
1	5575-GNVDE	Male	0	No	No	34	Yes	No	D
2	3668-QPYBK	Male	0	No	No	2	Yes	No	D
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	D
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber op

5 rows × 21 columns

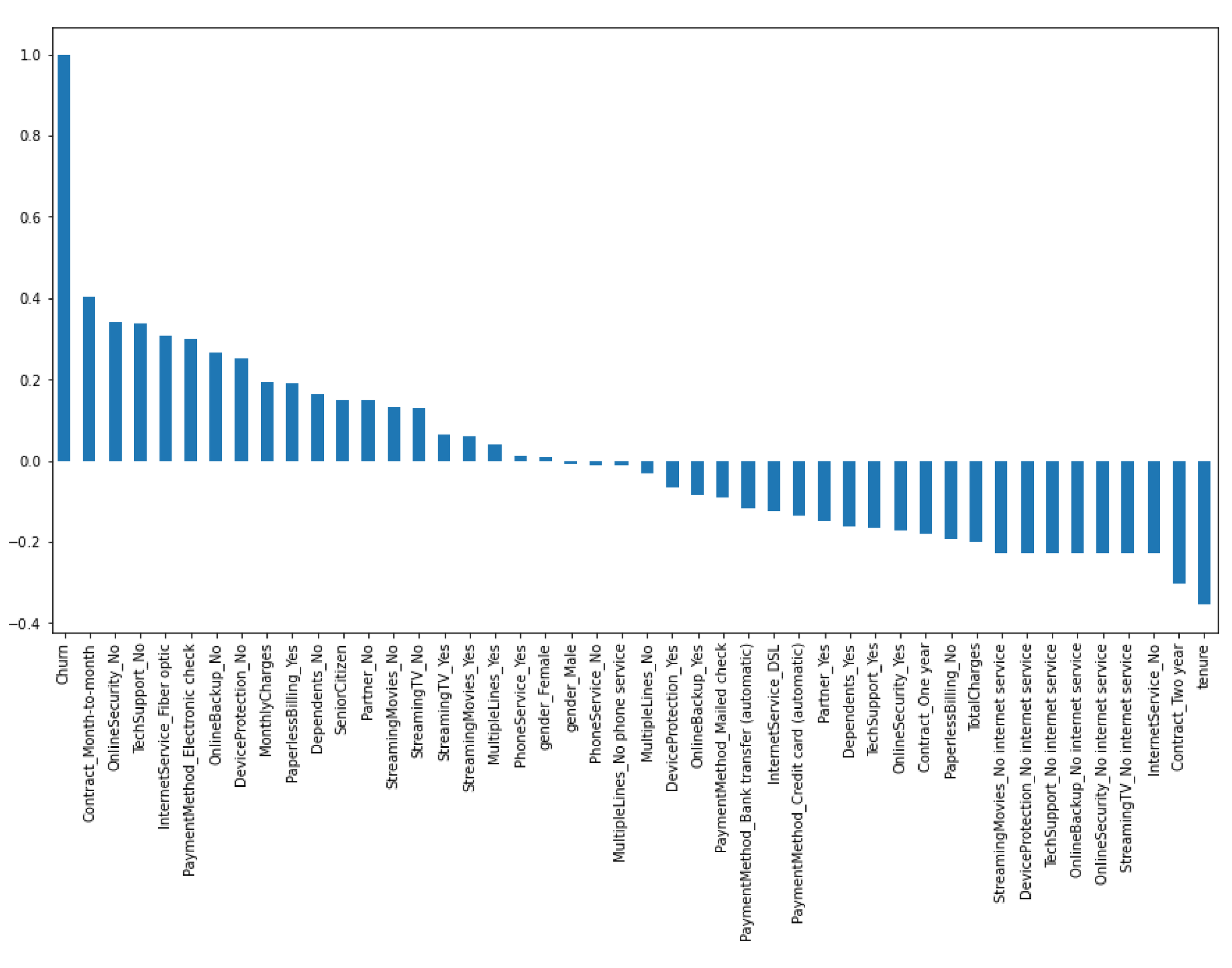
In [15]:
```
telecom_cust_churn.columns.values
```
Out[15]:
```
array(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
       'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract',
       'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges',
       'TotalCharges', 'Churn'], dtype=object)
```
In [16]:
```
# Checking the data types of all the columns
telecom_cust_churn.dtypes
```
Out[16]:
```
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
```
In [19]:
```
# Converting Total Charges to a numerical data type.
telecom_cust_churn.TotalCharges = pd.to_numeric(telecom_cust_churn.TotalCharges, erro
telecom_cust_churn.isnull().sum()
```
Out[19]:
```
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
```
In [20]:
```
#Removing missing values
telecom_cust_churn.dropna(inplace = True)
#Remove customer IDs from the data set
df2 = telecom_cust_churn.iloc[:,1:]
#Converting 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()
```
Out[20]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	Churn	gender_Female	gender_Male	Partner_No	Part
0	0	1	29.85	29.85	0	1	0	0	
1	0	34	56.95	1889.50	0	0	1	1	
2	0	2	53.85	108.15	1	0	1	1	
3	0	45	42.30	1840.75	0	0	1	1	
4	0	2	70.70	151.65	1	1	0	1	

5 rows × 46 columns

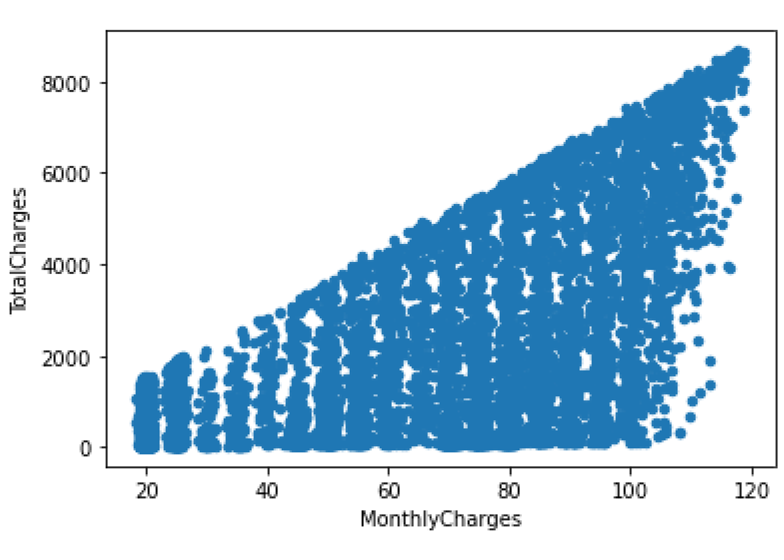
In [21]:
```
#Get Correlation of "Churn" with other variables:
plt.figure(figsize=(15,8))
df_dummies.corr()['Churn'].sort_values(ascending = False).plot(kind='bar')
```
Out[21]: <AxesSubplot:>



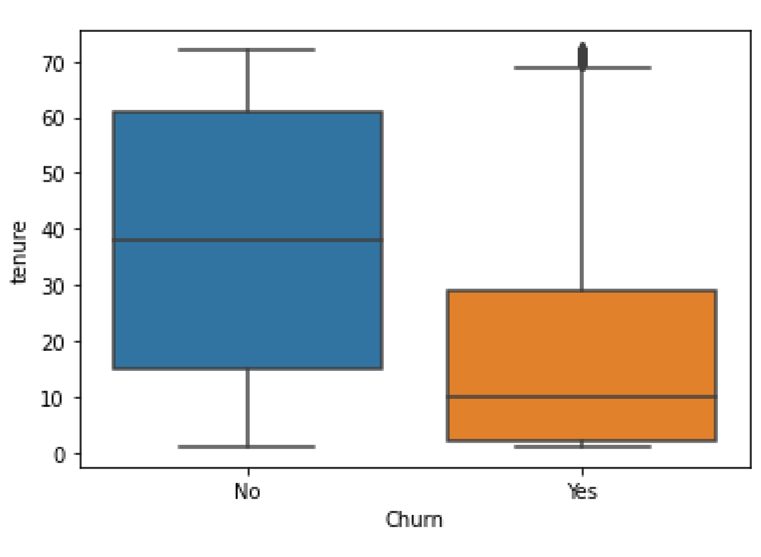
In [ ]:
```
# Observation
Month to month contracts, absence of online security and tech support seem to be posi

Interestingly, services such as Online security, streaming TV, online backup, tech su

We will explore the patterns for the above correlations below before we delve into mo
```
In [22]:
```
# Relation Between Monthly and Total Charges
telecom_cust_churn[['MonthlyCharges', 'TotalCharges']].plot.scatter(x = 'MonthlyCharge
                                                                    y='TotalCharges')
```
Out[22]: <AxesSubplot:xlabel='MonthlyCharges', ylabel='TotalCharges'>



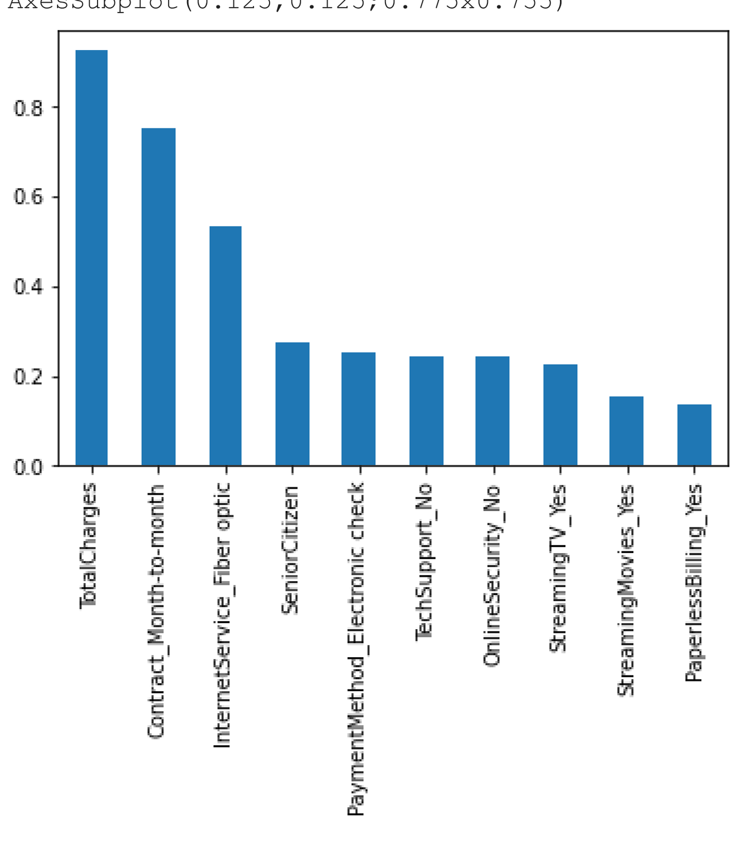
In [23]:
```
sns.boxplot(x = telecom_cust_churn.Churn, y = telecom_cust_churn.tenure)
```
Out[23]: <AxesSubplot:xlabel='Churn', ylabel='tenure'>



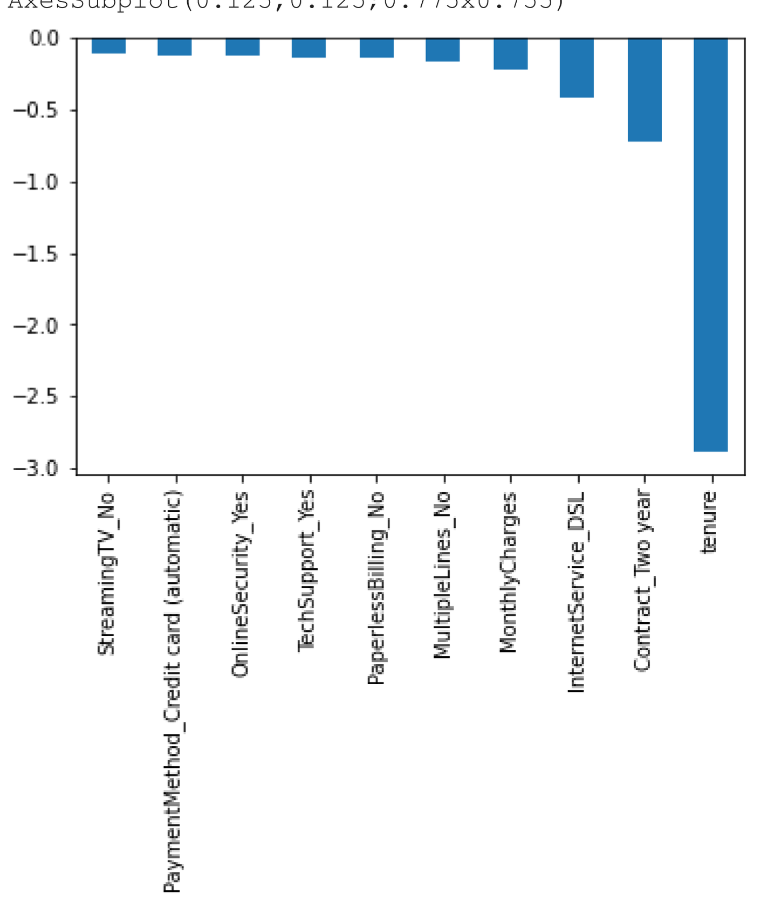
In [24]:
```
#1 Logistic Regression
# We will use the data frame where we had created dummy variables
y = df_dummies['Churn'].values
X = df_dummies.drop(columns = ['Churn'])

# 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
```
In [25]:
```
# 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=
```
In [26]:
```
# Running logistic regression model
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
result = model.fit(X_train, y_train)
```
In [27]:
```
from sklearn import metrics
prediction_test = model.predict(X_test)
# Print the prediction accuracy
print (metrics.accuracy_score(y_test, prediction_test))
```
0.8075829383886256

In [28]:
```
# 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'))
```
AxesSubplot(0.125,0.125;0.775x0.755)



In [30]:
```
print(weights.sort_values(ascending = False)[-10:].plot(kind='bar'))
```
AxesSubplot(0.125,0.125;0.775x0.755)



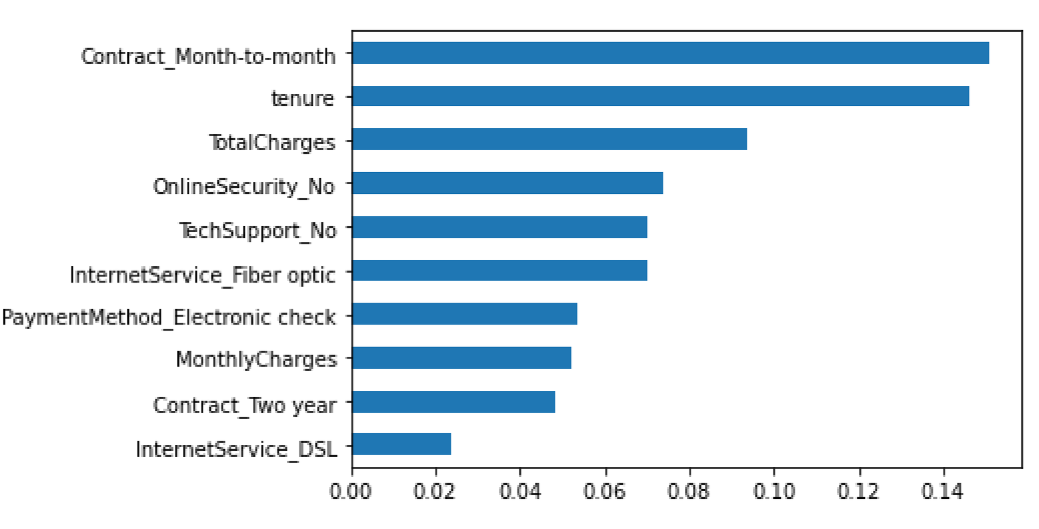
In [ ]:
```
# Observations

We can see that some variables have a negative relation to our predicted variable (Chu

1) As we saw in our EDA, having a 2 month contract reduces chances of churn. 2 month c
2) Having DSL internet service also reduces the proability of Churn
3) Lastly, total charges, monthly contracts, fibre optic internet services and senior
Any hypothesis on the above would be really helpful!
```
In [31]:
```
#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=
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))
```
0.8088130774697939

In [32]:
```
importances = model_rf.feature_importances_
weights = pd.Series(importances,
                 index=X.columns.values)
weights.sort_values()[-10:].plot(kind = 'barh')
```
Out[32]: <AxesSubplot:>



In [ ]:
```
# Observations:

1) From random forest algorithm, monthly contract, tenure and total charges are the mo
2) The results from random forest are very similar to that of the logistic regression
```
In [33]:
```
#3 Support Vector Machine (SVM)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=
```
In [35]:
```
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)
```
Out[35]: 0.820184790334044

In [36]:
```
# Create the Confusion matrix
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,preds))
```
```
[[953  89]
 [164 201]]
```
In [ ]:
```
# Observation
Wth SVM I was able to increase the accuracy to upto 82%.
```