Churn Prediction

April 18, 2020

0.0.1 Telco Churn Prediction (Part 1 of 2)

Telco is a fictional telecommunication company which provides home phone and Internet services. The dataset clearly indicates the status of each customer in the `Churn` column. Our goal is to train a model to predict customer churn using the Telco dataset. The dataset description indicates that churn refers to the customer leaving within the last month. The first part of the solution focuses on data visualization and exploration. The second part dives into the predictive modeling.

In [1]:
```
# load required libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# used for formatting the PDF
from IPython.display import display, Latex
```

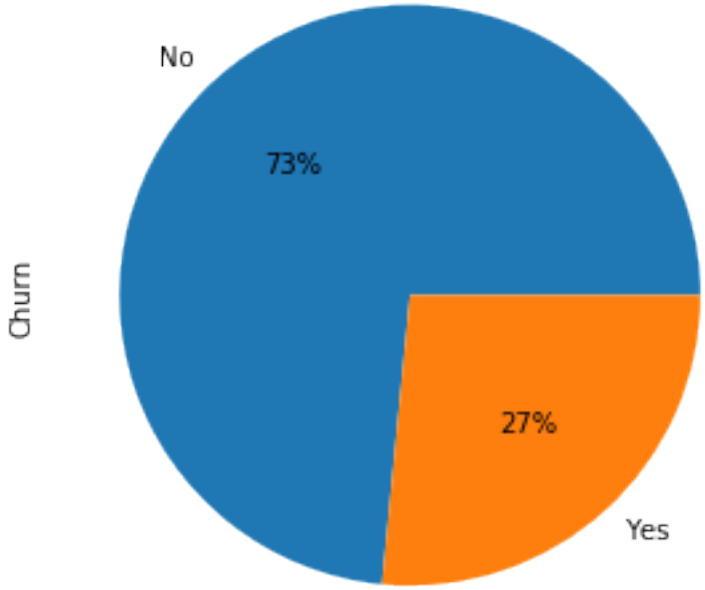
In [2]:
```
# import the data as a DataFrame
data = pd.read_csv('Telco-Customer-Churn.csv')

# let's take a look at the first row
data.iloc[0]
```

Out[2]:
```
customerID                7590-VHVEG
gender                        Female
SeniorCitizen                      0
Partner                          Yes
Dependents                        No
tenure                             1
PhoneService                      No
MultipleLines       No phone service
InternetService                  DSL
OnlineSecurity                    No
OnlineBackup                     Yes
DeviceProtection                  No
TechSupport                       No
StreamingTV                       No
StreamingMovies                   No
Contract              Month-to-month
```

```
PaperlessBilling                 Yes
PaymentMethod       Electronic check
MonthlyCharges                 29.85
TotalCharges                   29.85
Churn                             No
Name: 0, dtype: object
```

Inspecting the Dataset This dataset is fairly clean with no missing entries. There is a total of 7043 observations. Around 27% of the customers churned.

```
In [4]: plot = data.Churn.value_counts().plot.pie(y='Churn',
                                                  figsize=(5, 5),
                                                  autopct='%1.0f%%')
```



```
In [6]: data.info()
        data.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
customerID          7043 non-null object
gender              7043 non-null object
SeniorCitizen       7043 non-null int64
Partner             7043 non-null object
Dependents          7043 non-null object
tenure              7043 non-null int64
PhoneService        7043 non-null object
MultipleLines       7043 non-null object
InternetService     7043 non-null object
OnlineSecurity      7043 non-null object
OnlineBackup        7043 non-null object
DeviceProtection    7043 non-null object
TechSupport         7043 non-null object
StreamingTV         7043 non-null object
StreamingMovies     7043 non-null object
Contract            7043 non-null object
PaperlessBilling    7043 non-null object
PaymentMethod       7043 non-null object
MonthlyCharges      7043 non-null float64
TotalCharges        7043 non-null object
Churn               7043 non-null object
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

Out[6]:

	SeniorCitizen	tenure	MonthlyCharges
count	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	64.761692
std	0.368612	24.559481	30.090047
min	0.000000	0.000000	18.250000
25%	0.000000	9.000000	35.500000
50%	0.000000	29.000000	70.350000
75%	0.000000	55.000000	89.850000
max	1.000000	72.000000	118.750000

Visualizing and Inspecting the Dataset We can use the seaborn library to visualize and inspect the dataset.

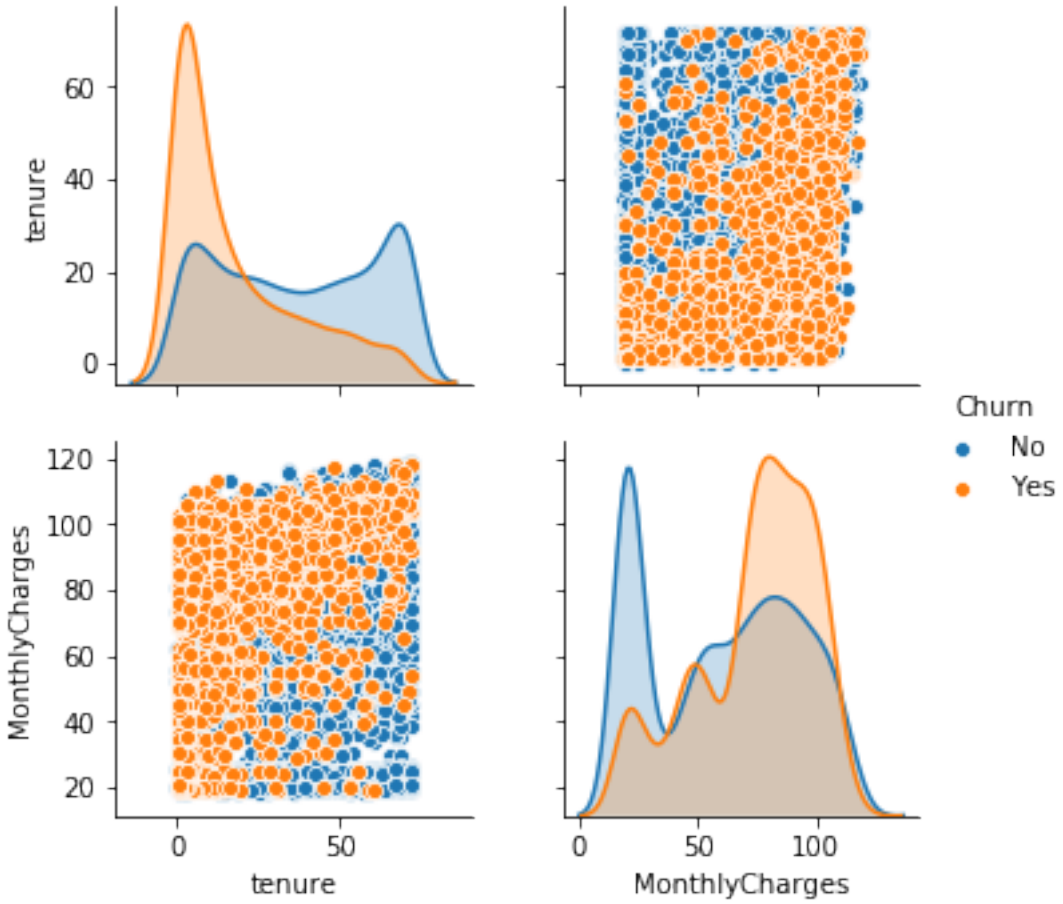
We begin by examining the continuous variables `tenure` and `MonthlyCharges`. Since `TotalCharges` can be approxiamated as a function of these two variables, we can exclude it from the plot.

The plots below show that:

- churned clients pay higher monthly charges than unchurned clients
- as tenure increases, the porportion of churned clients decrease

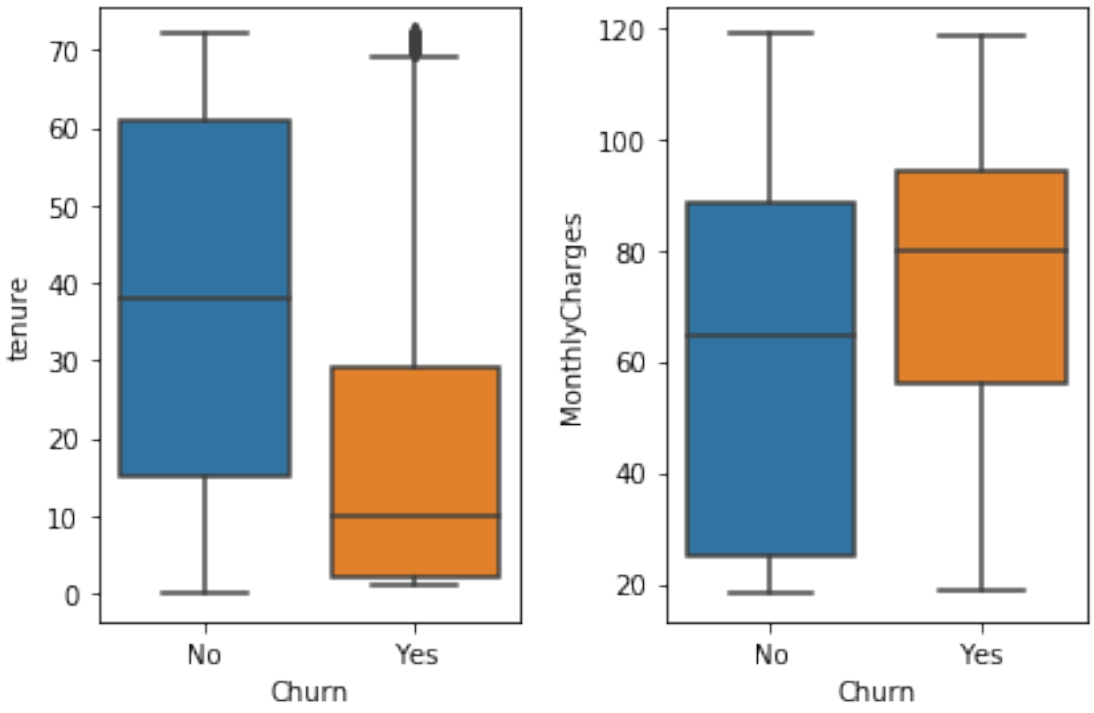
```
In [8]: # pairplot plots pairwise relationships in the data
        sns.pairplot(data[['Churn', 'tenure', 'MonthlyCharges']], hue="Churn")
```

```
Out[8]: <seaborn.axisgrid.PairGrid at 0x7f89acce2c88>
```



We can also draw some boxplots to illustrate the quantile differences between customers who churned and customers who did not churn. Understanding this relationship can help us pick the right promotions to incentive customers into staying.

```
In [10]: # this helps organize our plots and keep them in the same figure
         fig, axs = plt.subplots(ncols=2)

         sns.boxplot(x='Churn', y='tenure', data=data, ax=axs[0])
         sns.boxplot(x='Churn', y='MonthlyCharges', data=data, ax=axs[1])

         # this prevents plots from overlapping
         plt.tight_layout()
```



s

Next, we examine the categorical variables. From these plots, we see that:
Demographics:

- no difference in churn porportions between gender
- higher porportion of churn amongst SeniorCitizens, customers with no Partners and no Dependents

Services:

- higher porportion of churn amongst customers with Fiber optic, no OnlineSecurity, no OnlineBackup, no DeviceProtection, no TechSupport

Billing:

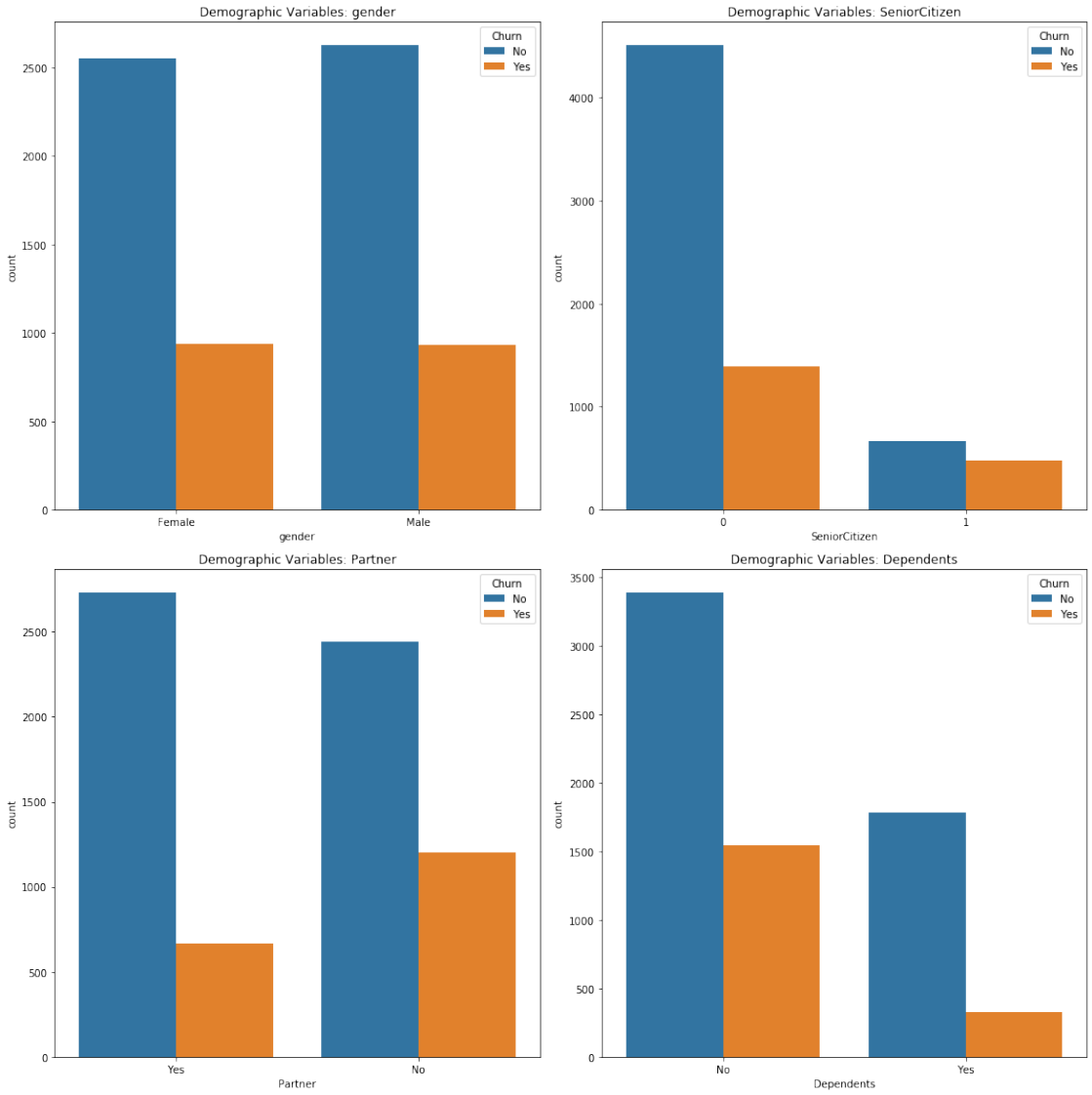
- higher porportion of churn for customers on a month-to-month plan, Paperless Billing and who pay by ElectronicCheck

This information tells us a lot about Telco's products and customers. We see that customers in certain demographics have much higher rates of churn than others. Customers less reliant on Telco's other services also seem to churn more. Surprisingly, customers that pay by electronic cheques see a much higher churn rate.

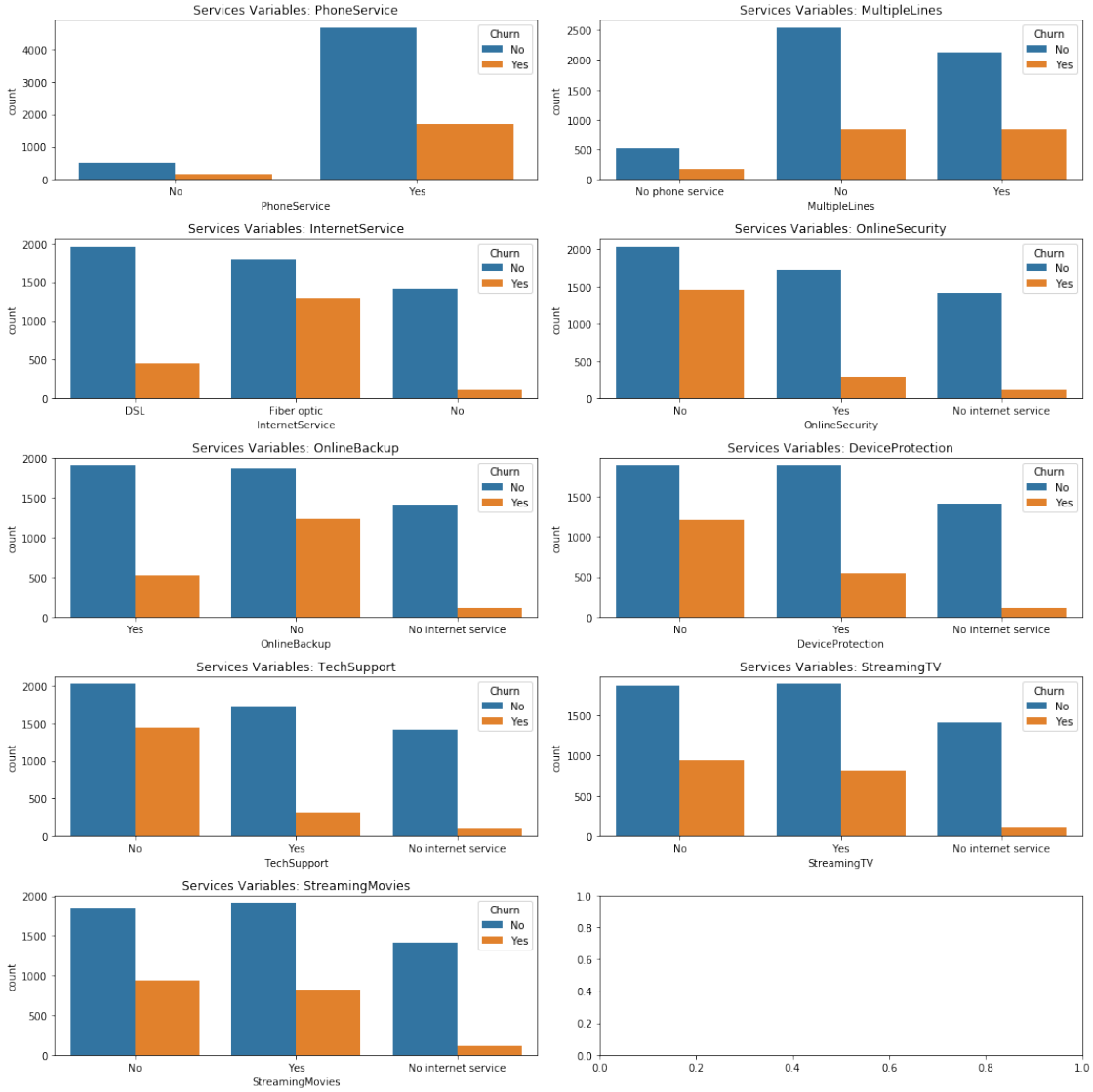
Certain services, such as Fiber optic, also sees a higher churn rate. It would be worthwhile to investigate to understand *why* this is the case, in particular if there are any gaps in these offerings.

```
In [12]: '''
         this helps organize our plots and keep them in the same figure
             columns: List[str] - each column is its own chart
             title: str - title is used as the title of each chart
         function only works for 3 or more columns
         '''
         def plot_categoricals(columns, title):
             fig, axs = plt.subplots(ncols=2, nrows=int(len(columns) / 2) + len(columns) % 2)
             fig.set_size_inches(15, 15)

             row = col = 0
             for column in columns:
                 plot_title = '{}: {}'.format(title, column)
                 sns.countplot(x=column, hue="Churn",
                               data=data, ax=axs[row][col]).set_title(plot_title)
                 if col == 1:
                     col = 0
                     row += 1
                 else:
                     col += 1

             # this prevents plots from overlapping
             plt.tight_layout()
```

```
In [13]: # plot demographic features
         plot_categoricals(['gender', 'SeniorCitizen', 'Partner', 'Dependents'],
                           'Demographic Variables')
```



```
In [14]: # plot services features
         plot_categoricals(['PhoneService', 'MultipleLines', 'InternetService',
                            'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
                            'TechSupport', 'StreamingTV','StreamingMovies'],
                           'Services Variables')
```



```
In [15]: # plot billing features
         plot_categoricals(['Contract', 'PaperlessBilling', 'PaymentMethod'],
                           'Billing Variables')
```