MGT-415: Data Science in Practice

Solutions to Problem Set 1
March 15, 2019

Do Carmo Maria Genc Murat Nyambuu Lkham Trichilo Giulio Xi Fan

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

I Problem Set 1 Data Analysis

In [2]:
```
import numpy as np
import pandas as pd

import seaborn as sns
import matplotlib as mpl

import matplotlib.pyplot as plt
from IPython.display import display

import warnings
warnings.filterwarnings("ignore")
pd.set_option('display.max_colwidth', 250)
sns.set()

pd.set_option('display.notebook_repr_html', True)

def _repr_latex_(self):
    return "\centering{%s}" % self.to_latex()

pd.DataFrame._repr_latex_ = _repr_latex_  # monkey patch pandas DataFrame
```

I Introductory Information about the dataset

In [3]:
```
## Check data

dsdata = pd.read_csv("Telco-Customer-Churn.csv")
dsdata = dsdata.drop(dsdata.columns[0], axis=1) #dont need ID
ds = dsdata.copy()
ds['SeniorCitizen'] = ds['SeniorCitizen'].astype(str) #objects are implicitly 'str'
ds['TotalCharges']  = pd.to_numeric(ds.TotalCharges, errors='coerce')
#idx = ds.select_dtypes(include='object').columns.values
#ds[idx] = ds[idx].astype('str')


## No lack of data , however Senior citizen is 'int' and 'TotalCharges' is object
(string!)
```

In [4]:
```
display(ds.head(n=5))
```

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineI
0	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes
1	Male	0	No	No	34	Yes	No	DSL	Yes	No
2	Male	0	No	No	2	Yes	No	DSL	Yes	Yes
3	Male	0	No	No	45	No	No phone service	DSL	Yes	No
4	Female	0	No	No	2	Yes	No	Fiber optic	No	No

In [5]:
```
ds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 20 columns):
gender              7043 non-null object
```

```
SeniorCitizen       7043 non-null object
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        7032 non-null float64
Churn               7043 non-null object
dtypes: float64(2), int64(1), object(17)
memory usage: 1.1+ MB
```

1. *Functions for computation of relevant statistics of categorical variables*

```
In [6]: # counts
        def count_categorical(df):
            dsfeature = df.columns
            dfprint = pd.DataFrame(columns=['Variable Name','Category/Count'])
            for i in dsfeature:
                #if (i != 'customerID') and (i !='tenure') and (i !='MonthlyCharges') and (i
        !='TotalCharges') :
                if type(df[i].values[0]) == str:
                    dfprint = dfprint.append({'Variable Name':i,'Category/Count':list(zip(list(d
        f[i].value_counts().index),df[i].value_counts().values))}, ignore_index=True)

            return dfprint

        def get_num_cols(df):
            idx = df.select_dtypes(exclude='object').columns.values
            dF = df[idx].dropna() #remove NaNs or else it cant plot
            return dF.columns

        def plot_hist2(df,df2,df_col):
            df = df.dropna()
            df2 = df2.dropna()
            for d in df_col:
                print("Empirical Distribution of Variable "+d)
                fig, axes = plt.subplots(1,2,figsize=(15,9))
                sns.distplot(df[d],ax=axes[0])
                sns.distplot(df2[d],ax=axes[1])
                axes[0].set_ylabel("Probability")
                axes[1].set_ylabel("Probability")
                plt.suptitle("Empirical Probability Distribution of Numerical Variable "+d)
                plt.show()
```

```
        def plot_hist(df,df_col):
            df = df.dropna()
            for d in df_col:
                #print("Empirical Distribution of Variable "+d)
                plt.figure(figsize=(15,9))
                sns.distplot(df[d])
                plt.ylabel("Probability")
                plt.title("Empirical Probability Distribution of Numerical Variable "+d)
                plt.show()
```

```
In [7]: print("For categorical variables:")
        dfprint = count_categorical(ds)
        dfprint
```

```
For categorical variables:
```

Out[7]:

	Variable Name	Category/Count
0	gender	[(Male, 3555), (Female, 3488)]
1	SeniorCitizen	[(0, 5901), (1, 1142)]
2	Partner	[(No, 3641), (Yes, 3402)]
3	Dependents	[(No, 4933), (Yes, 2110)]
4	PhoneService	[(Yes, 6361), (No, 682)]
5	MultipleLines	[(No, 3390), (Yes, 2971), (No phone service, 682)]
6	InternetService	[(Fiber optic, 3096), (DSL, 2421), (No, 1526)]
7	OnlineSecurity	[(No, 3498), (Yes, 2019), (No internet service, 1526)]
8	OnlineBackup	[(No, 3088), (Yes, 2429), (No internet service, 1526)]
9	DeviceProtection	[(No, 3095), (Yes, 2422), (No internet service, 1526)]
10	TechSupport	[(No, 3473), (Yes, 2044), (No internet service, 1526)]
11	StreamingTV	[(No, 2810), (Yes, 2707), (No internet service, 1526)]
12	StreamingMovies	[(No, 2785), (Yes, 2732), (No internet service, 1526)]
13	Contract	[(Month-to-month, 3875), (Two year, 1695), (One year, 1473)]
14	PaperlessBilling	[(Yes, 4171), (No, 2872)]
15	PaymentMethod	[(Electronic check, 2365), (Mailed check, 1612), (Bank transfer (automatic), 1544), (Credit card (auto
16	Churn	[(No, 5174), (Yes, 1869)]

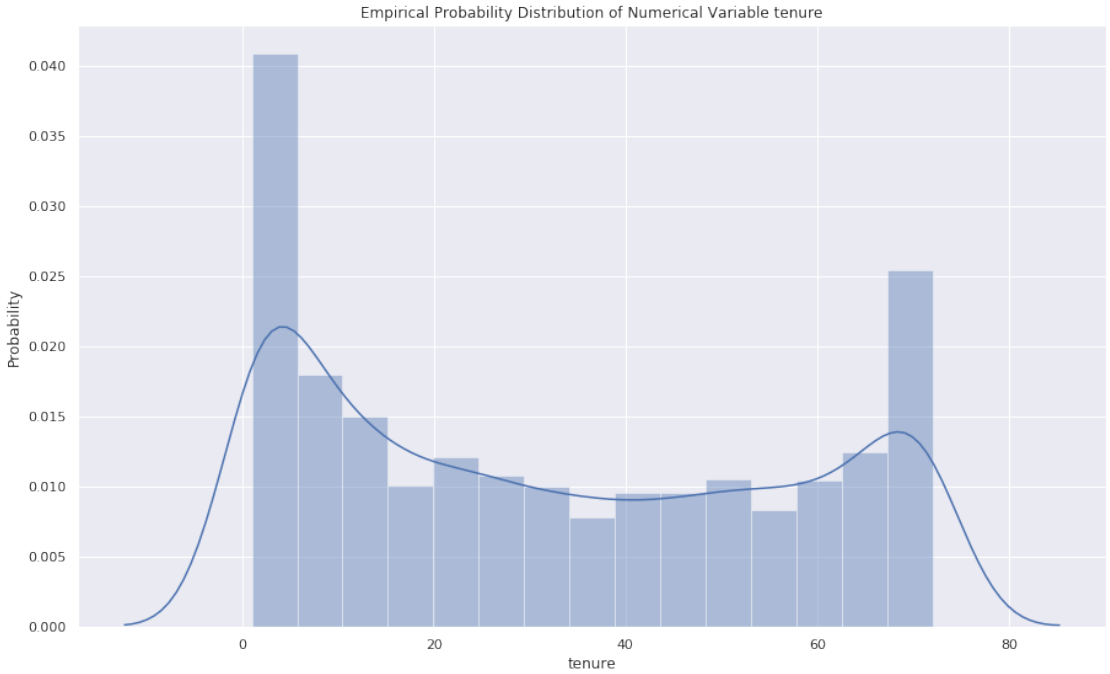
```
In [8]: print("For numerical variables:")
        ds.describe().round(4).T
```

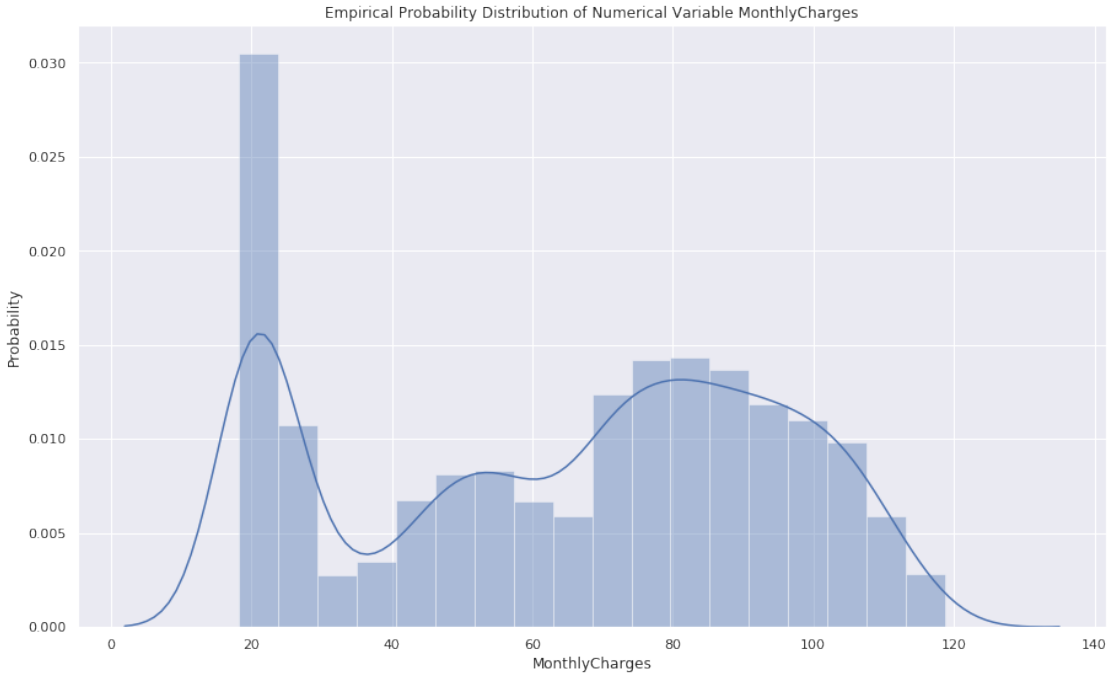
```
For numerical variables:
```

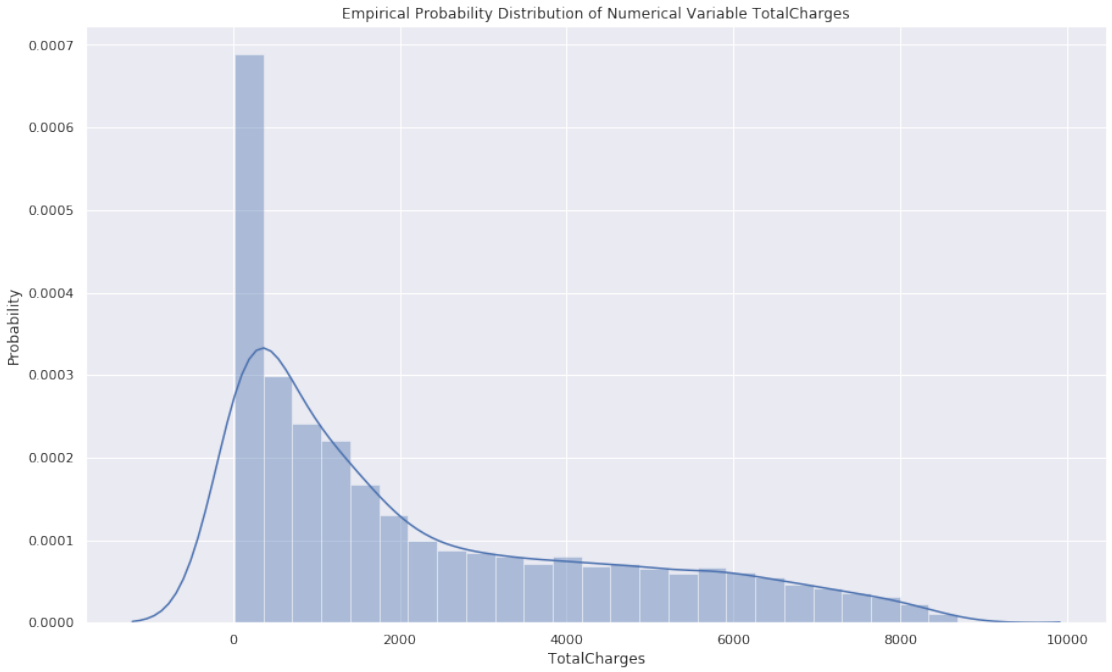
Out[8]:

	count	mean	std	min	25%	50%	75%	max
tenure	7043.0	32.3711	24.5595	0.00	9.00	29.000	55.0000	72.00
MonthlyCharges	7043.0	64.7617	30.0900	18.25	35.50	70.350	89.8500	118.75
TotalCharges	7032.0	2283.3004	2266.7714	18.80	401.45	1397.475	3794.7375	8684.80

```
In [9]: plot_hist(ds,get_num_cols(ds))
```


Empirical Probability Distribution of Numerical Variable tenure
Probability
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
20
40
60
80
tenure


Empirical Probability Distribution of Numerical Variable MonthlyCharges
Probability
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
20
40
60
80
100
120
140
MonthlyCharges



I Analysis for the Loyal customer base

```
In [10]: ## Analysis of product preference
         def plot_bars(df):
             # Visual Service
             #plt.figure()
             fig, axes = plt.subplots(1,4,figsize=(18, 5))
             sns.countplot(x=df['PhoneService'],hue=df['InternetService'], palette="Reds_d",
         ax=axes[0])
             sns.countplot(x=df['PhoneService'],hue=df['MultipleLines'], palette="Blues_d",
         ax=axes[1])
             sns.countplot(x=df['InternetService'],hue=df['StreamingMovies'],palette="Greens_d",
         ax=axes[2])
             sns.countplot(x=df['InternetService'],hue=df['StreamingTV'],palette="Oranges_d",
         ax=axes[3])
             plt.show()

             # Visual Support
             #plt.figure()
             fig, axes = plt.subplots(1,4,figsize=(18, 5))
             sns.countplot(x=df['InternetService'],hue=df['OnlineSecurity'],palette="Reds_d",
         ax=axes[0])
             sns.countplot(x=df['InternetService'],hue=df['OnlineBackup'],palette="Blues_d",
         ax=axes[1])
             sns.countplot(x=df['InternetService'],hue=df['TechSupport'],palette="Greens_d",
         ax=axes[2])
         sns.countplot(x=df['InternetService'],hue=df['DeviceProtection'],palette="Oranges_d",
         ax=axes[3])
             plt.show()
```

```
        def plot_scatter_bw(df):
            #Segmentation

            fig, axes = plt.subplots(1,4,figsize=(18, 10))
            sns.stripplot(x=df['gender'], y=df['TotalCharges'],jitter=True, ax=axes[0])
            sns.stripplot(x=df['SeniorCitizen'], y=df['TotalCharges'],jitter=True, ax=axes[1])
            sns.stripplot(x=df['Partner'], y=df['TotalCharges'],jitter=True, ax=axes[2])
            sns.stripplot(x=df['Dependents'], y=df['TotalCharges'],jitter=True, ax=axes[3])
            plt.show()

            fig, axes = plt.subplots(1,4,figsize=(18, 8))
            sns.boxplot(x=df['gender'], y=df['TotalCharges'], ax=axes[0])
            sns.boxplot(x=df['SeniorCitizen'], y=df['TotalCharges'], ax=axes[1])
            sns.boxplot(x=df['Partner'], y=df['TotalCharges'], ax=axes[2])
            sns.boxplot(x=df['Dependents'], y=df['TotalCharges'],ax=axes[3])
            plt.show()
            #Apparently gender & dependent do not matter much

        ## Analsysis of revenue contribution, i.e. charges and tenure
        # Screen of 'Loyal Customer'
        def revenue_contr(df):
            dscus=df.copy()
            dscus['SeniorCitizen'] = pd.to_numeric(dscus.SeniorCitizen, errors='coerce')
            partner_mapping = {
                       'Yes': 1,
                       'No': 0}
            dscus['Partner'] = dscus['Partner'].map(partner_mapping)
            dscus['Loyal'] = dscus['SeniorCitizen'] + dscus['Partner']

            dscus=dscus[(dscus[u'Loyal']>=1)]
            dscus['Loyal'].value_counts()
            display(dscus.head(n=15))
            return dscus



        def plot_box(df,dscus):
            fig, axes = plt.subplots(1,3,figsize=(18, 5))
            sns.countplot(x=dscus['Loyal'],hue=df['InternetService'], palette="Reds_d",
        ax=axes[0])
            sns.countplot(x=dscus['Loyal'],hue=df['PhoneService'], palette="Reds_d", ax=axes[1])
            sns.countplot(x=dscus['Loyal'],hue=df['Contract'], palette="Reds_d", ax=axes[2])
            plt.show()
```

In [11]:
```
ds_loyal = ds.copy()

        #ignore the churn
        print(ds_loyal['Churn'].value_counts())
        print("Overview of customers who have not churned this year")
        ds_loyal=ds_loyal[(ds_loyal[u'Churn']=='No')]
        ds_loyal.head(n=5)
```

```
No     5174
Yes    1869
Name: Churn, dtype: int64
Overview of customers who have not churned this year
```

Out[11]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	OnlineI
0	Female	0	Yes	No	1	No	No phone service	DSL	No	Yes
1	Male	0	No	No	34	Yes	No	DSL	Yes	No
3	Male	0	No	No	45	No	No phone service	DSL	Yes	No
6	Male	0	No	Yes	22	Yes	Yes	Fiber optic	No	Yes
7	Female	0	No	No	10	No	No phone service	DSL	Yes	No

```
In [12]: print("For categorical variables:")
         dfprint_l = count_categorical(ds_loyal)
         dfprint_l
```

```
For categorical variables:
```

Out[12]:

	Variable Name	Category/Count
0	gender	[(Male, 2625), (Female, 2549)]
1	SeniorCitizen	[(0, 4508), (1, 666)]
2	Partner	[(Yes, 2733), (No, 2441)]
3	Dependents	[(No, 3390), (Yes, 1784)]
4	PhoneService	[(Yes, 4662), (No, 512)]
5	MultipleLines	[(No, 2541), (Yes, 2121), (No phone service, 512)]
6	InternetService	[(DSL, 1962), (Fiber optic, 1799), (No, 1413)]
7	OnlineSecurity	[(No, 2037), (Yes, 1724), (No internet service, 1413)]
8	OnlineBackup	[(Yes, 1906), (No, 1855), (No internet service, 1413)]
9	DeviceProtection	[(No, 1884), (Yes, 1877), (No internet service, 1413)]
10	TechSupport	[(No, 2027), (Yes, 1734), (No internet service, 1413)]
11	StreamingTV	[(Yes, 1893), (No, 1868), (No internet service, 1413)]
12	StreamingMovies	[(Yes, 1914), (No, 1847), (No internet service, 1413)]
13	Contract	[(Month-to-month, 2220), (Two year, 1647), (One year, 1307)]
14	PaperlessBilling	[(Yes, 2771), (No, 2403)]
15	PaymentMethod	[(Mailed check, 1304), (Electronic check, 1294), (Credit card (automatic), 1290), (Bank transfer (auto
16	Churn	[(No, 5174)]

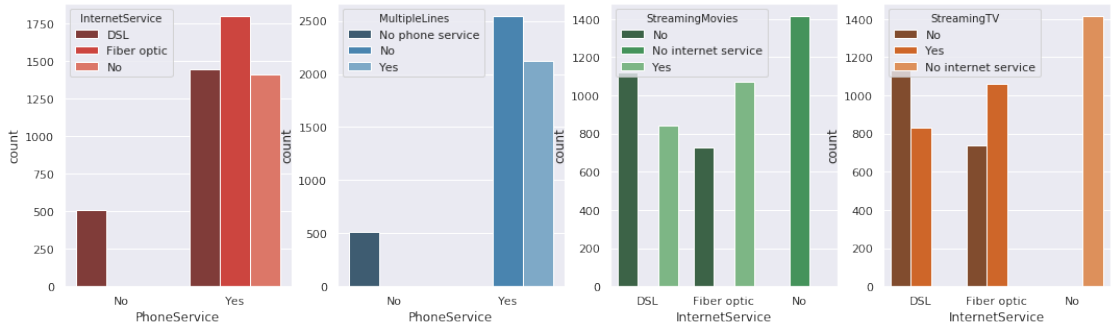
```
In [13]: print("For numerical variables:")
         ds_loyal.describe().round(4).T
```

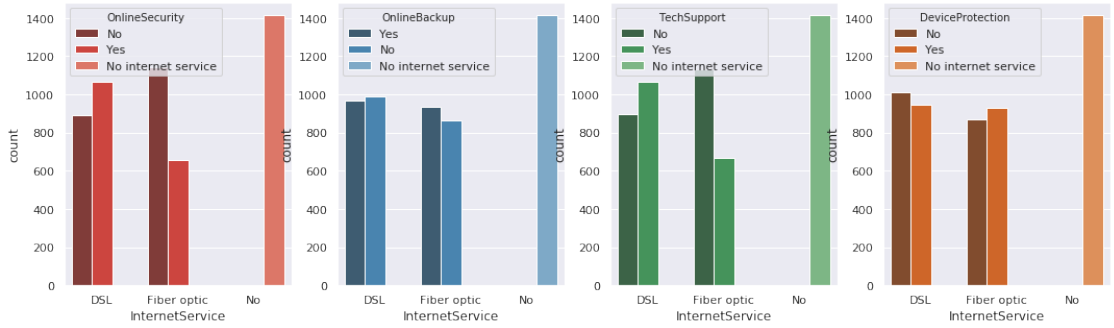
```
For numerical variables:
```

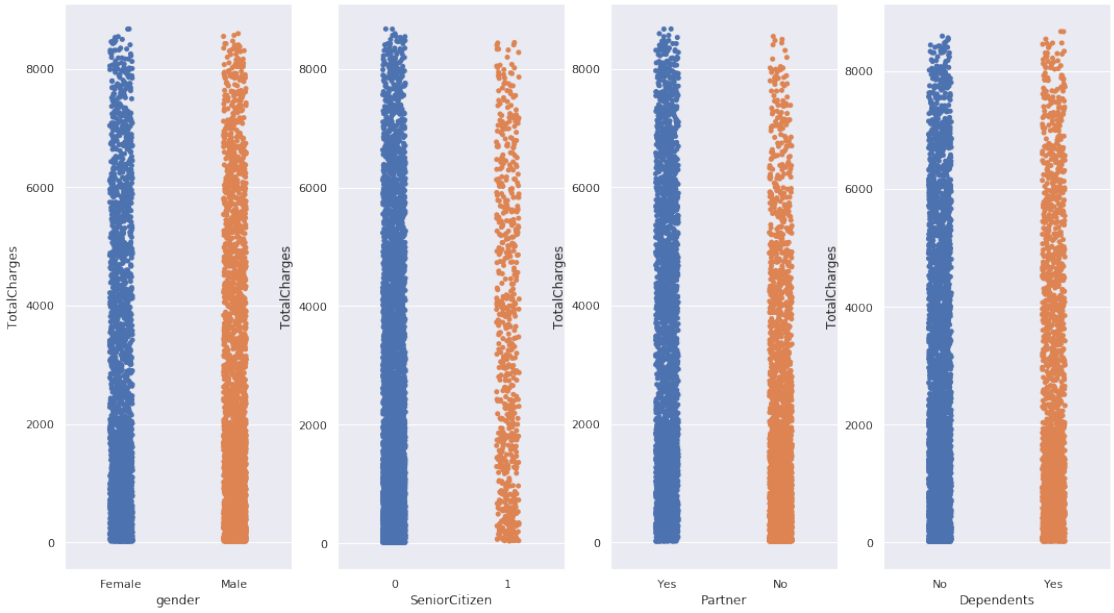
Out[13]:

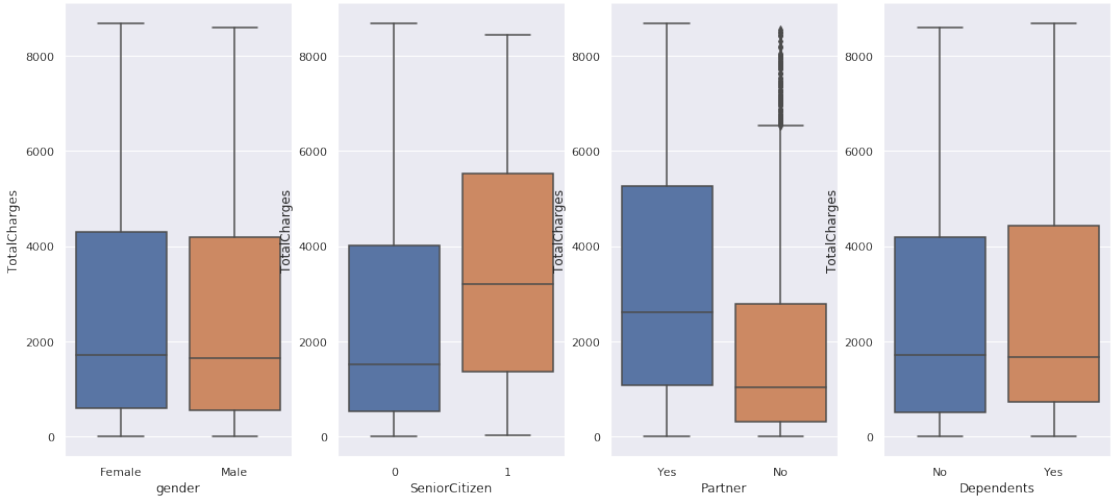
	count	mean	std	min	25%	50%	75%	max
tenure	5174.0	37.5700	24.1138	0.00	15.000	38.000	61.000	72.00
MonthlyCharges	5174.0	61.2651	31.0926	18.25	25.100	64.425	88.400	118.75
TotalCharges	5163.0	2555.3441	2329.4570	18.80	577.825	1683.600	4264.125	8672.45

```
In [14]: plot_bars(ds_loyal)
         plot_scatter_bw(ds_loyal)
         dscus = revenue_contr(ds_loyal)
         plot_box(ds_loyal,dscus)
         plot_hist(ds_loyal,get_num_cols(ds))
```

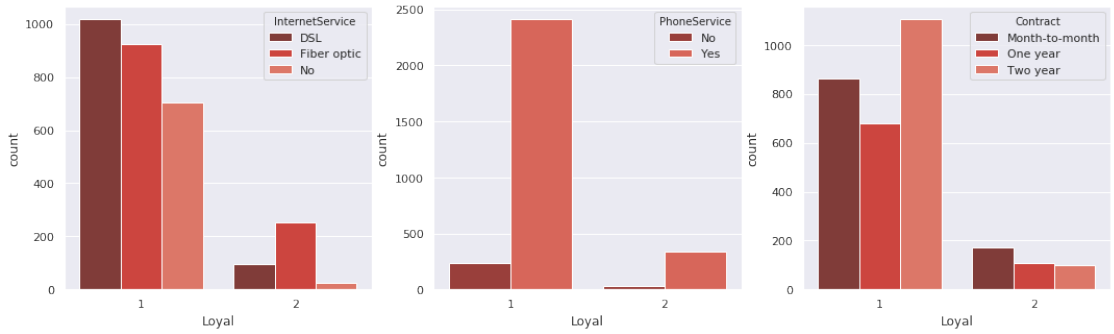

InternetService
DSL
Fiber optic
No
count
PhoneService
MultipleLines
No phone service
No
Yes
StreamingMovies
No internet service
InternetService
StreamingTV

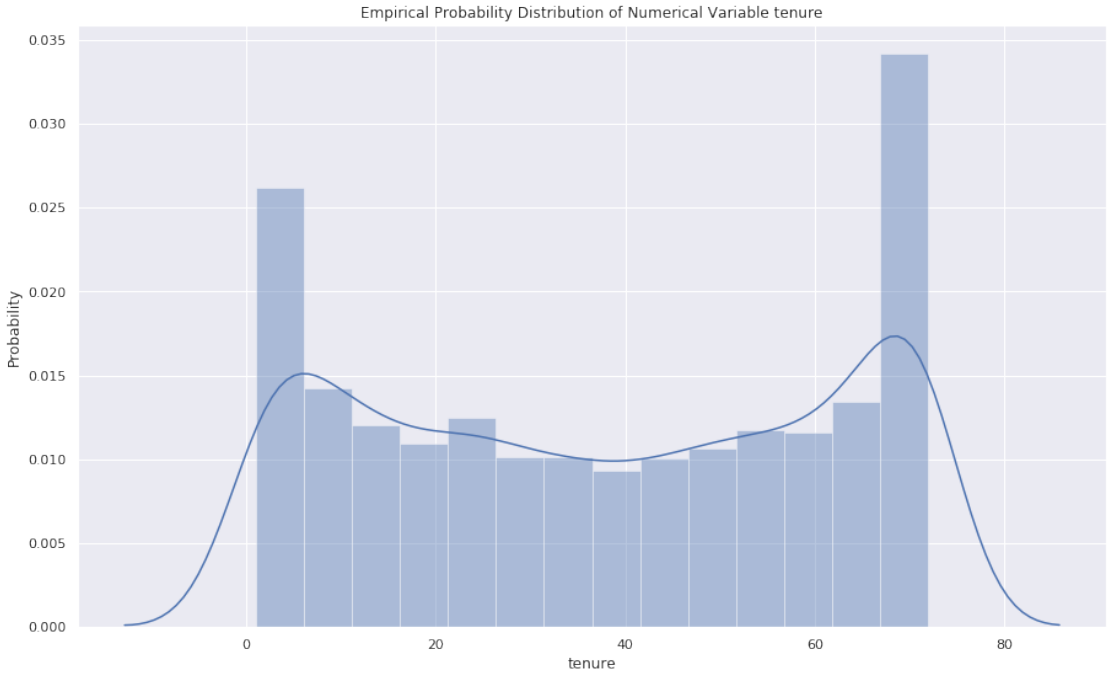

OnlineSecurity
OnlineBackup
TechSupport
DeviceProtection
InternetService

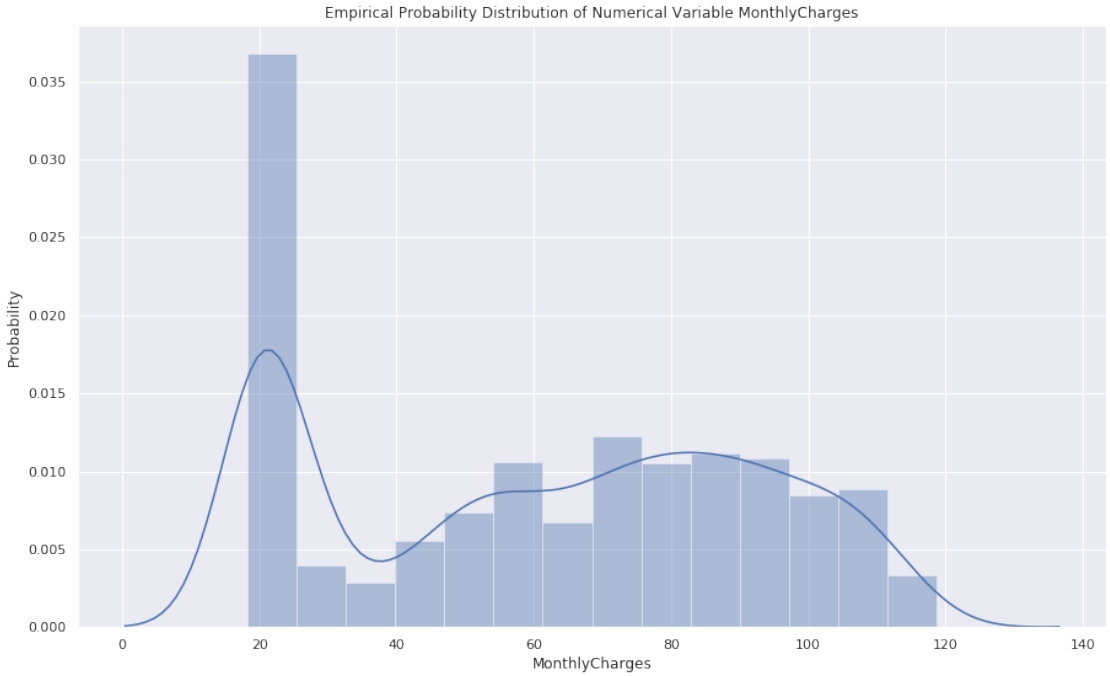

TotalCharges
gender
Female
Male
SeniorCitizen
Partner
Dependents

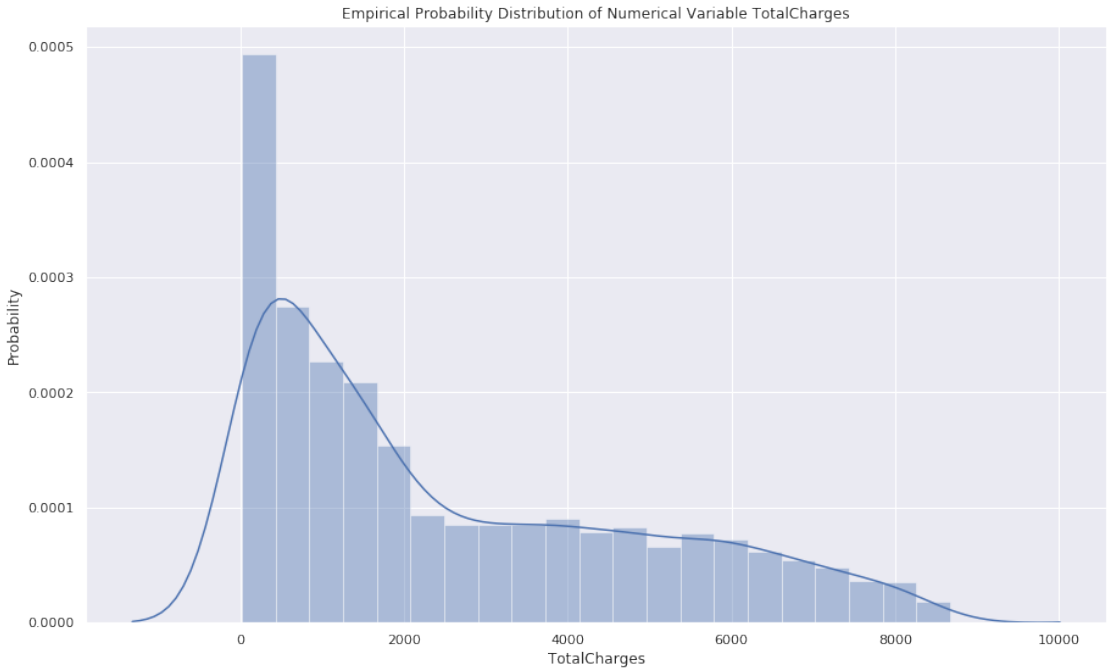


	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	O
0	Female	0	1	No	1	No	No phone service	DSL	No	Ye
10	Male	0	1	Yes	13	Yes	No	DSL	Yes	N
12	Male	0	1	No	58	Yes	Yes	Fiber optic	No	N
15	Female	0	1	Yes	69	Yes	Yes	Fiber optic	Yes	Ye
21	Male	0	1	No	12	Yes	No	No	No internet service	N
23	Female	0	1	No	58	Yes	Yes	DSL	No	Ye
24	Male	0	1	Yes	49	Yes	No	DSL	Yes	Ye
28	Male	0	1	No	72	Yes	Yes	DSL	Yes	Ye
30	Female	1	1	No	71	Yes	Yes	Fiber optic	Yes	Ye
31	Male	1	1	No	2	Yes	No	Fiber optic	No	N
32	Female	0	1	Yes	27	Yes	No	DSL	Yes	Ye
34	Male	1	0	No	1	Yes	No	DSL	No	N
35	Female	0	1	Yes	72	Yes	Yes	Fiber optic	Yes	Ye
40	Male	0	1	Yes	10	Yes	No	DSL	No	Ye
41	Female	0	1	Yes	70	Yes	Yes	DSL	Yes	Ye




Empirical Probability Distribution of Numerical Variable tenure
Probability
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
20
40
60
80
tenure


Empirical Probability Distribution of Numerical Variable MonthlyCharges
Probability
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
20
40
60
80
100
120
140
MonthlyCharges



In [ ]:

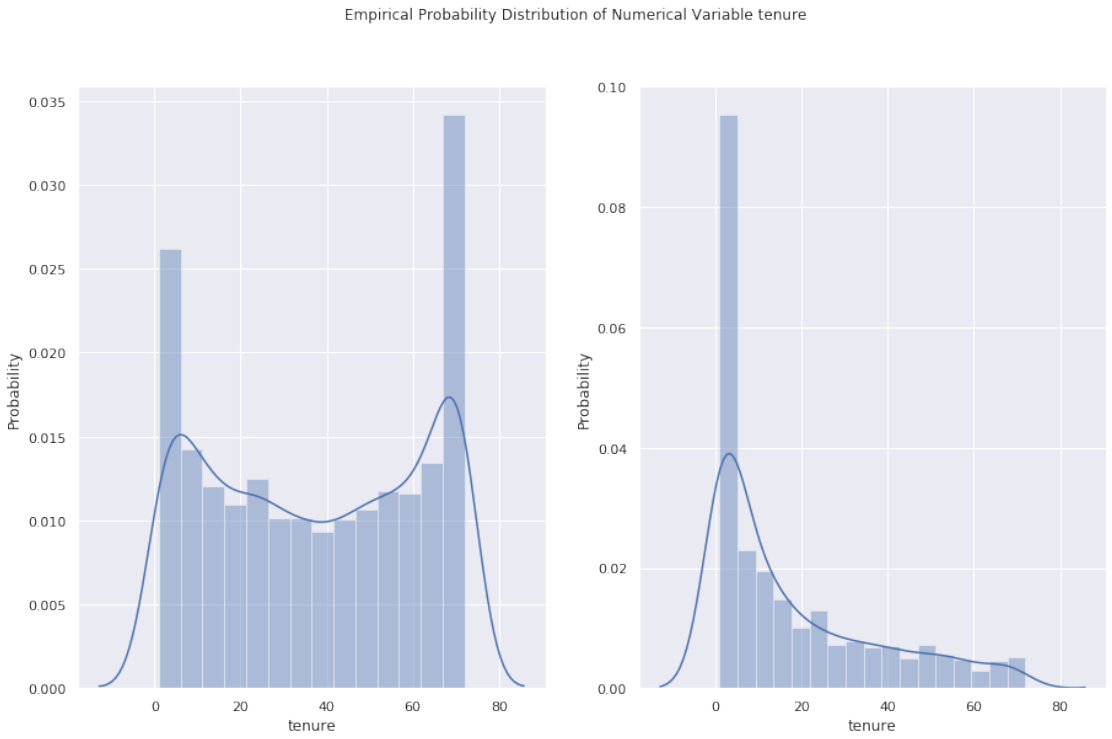
I Compare Empirical Distribution of Loyal vs Churn

In [15]:
```
ds_churn = ds.copy()

        #ignore the churn
        print(ds_churn['Churn'].value_counts())
        print("Overview of customers who have churned this year")
        ds_churn=ds_churn[(ds_churn[u'Churn']=='Yes')]
        ds_churn.head(n=5)

        def plot_more_bars_dis(df, ds_churn):
            fig, axes = plt.subplots(1,3,figsize=(18, 5))
            sns.countplot(x=ds_churn['Churn'],hue=df['InternetService'], palette="Reds_d",
        ax=axes[0])
            sns.countplot(x=ds_churn['Churn'],hue=df['PhoneService'], palette="Reds_d",
        ax=axes[1])
            sns.countplot(x=ds_churn['Churn'],hue=df['Contract'], palette="Reds_d", ax=axes[2])
            plt.show()
```

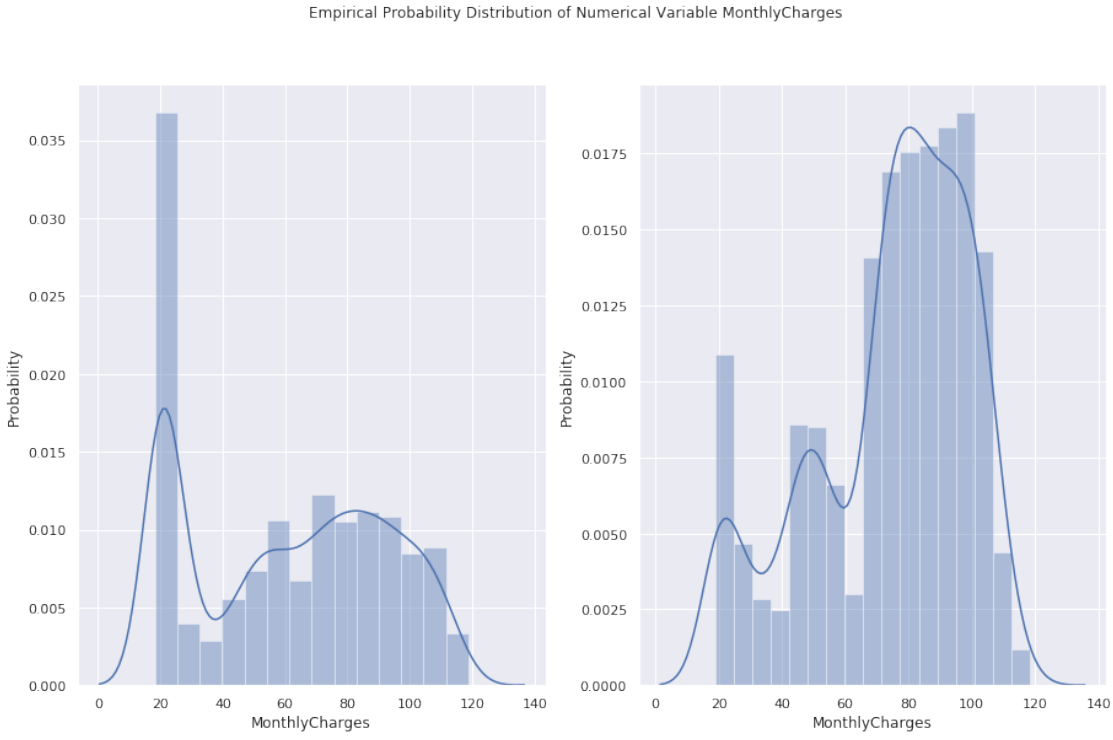
```
No     5174
Yes    1869
Name: Churn, dtype: int64
Overview of customers who have churned this year
```

In [16]:
```
plot_hist2(ds_loyal,ds_churn,get_num_cols(ds))
```

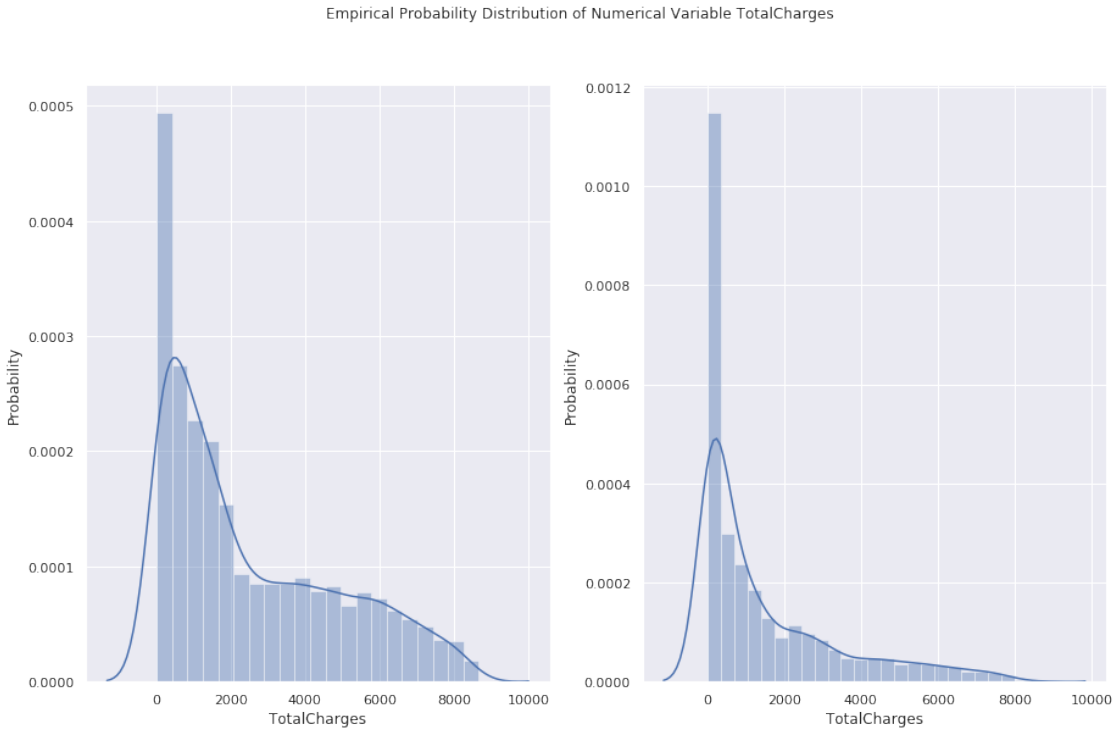
```
Empirical Distribution of Variable tenure
```



Empirical Distribution of Variable MonthlyCharges

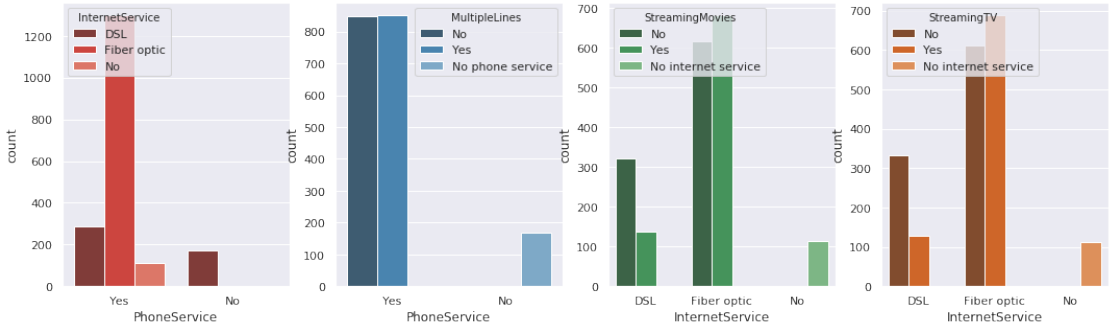


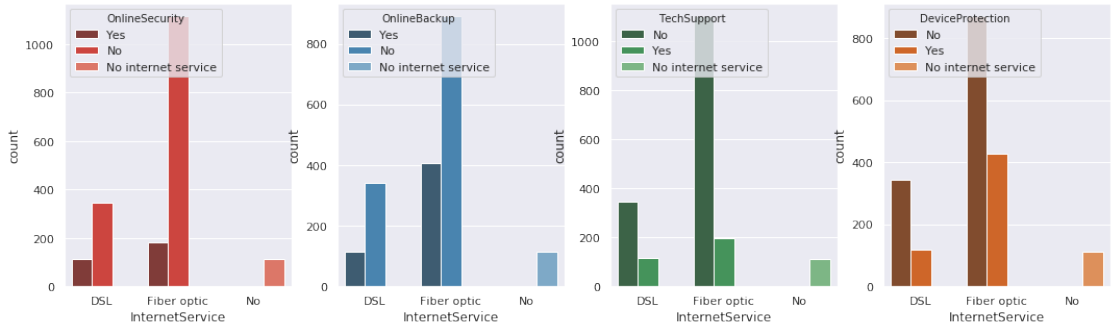
Empirical Distribution of Variable TotalCharges

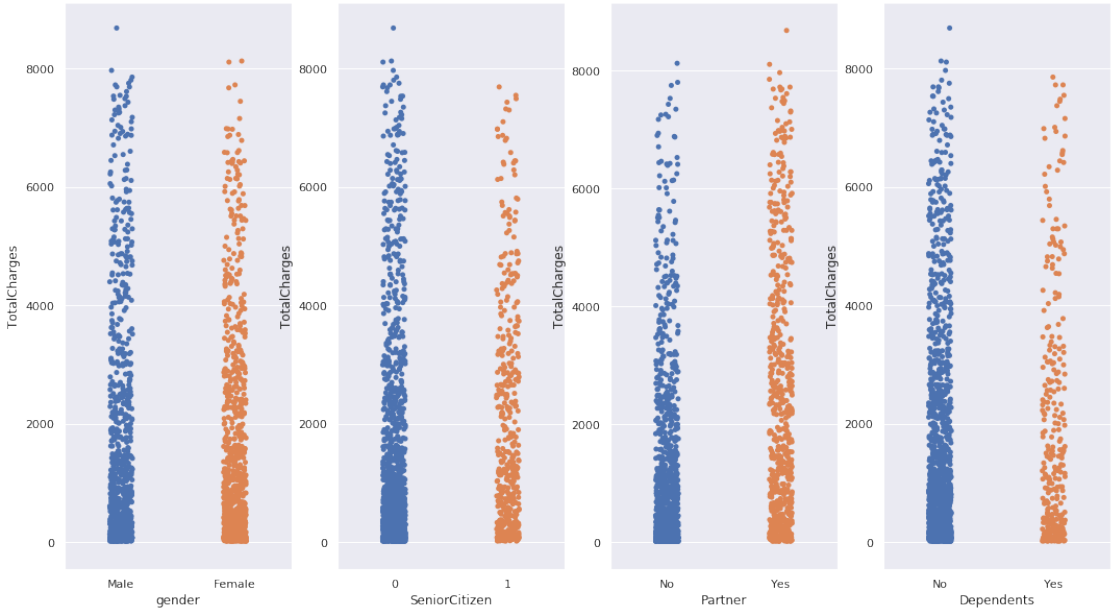


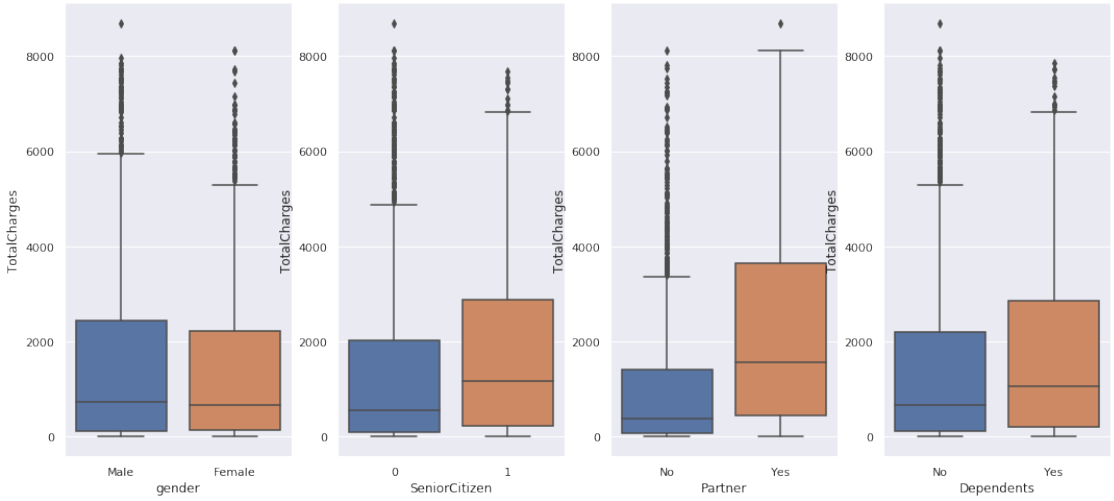
I Distribution of Categorical Variables related to Churn

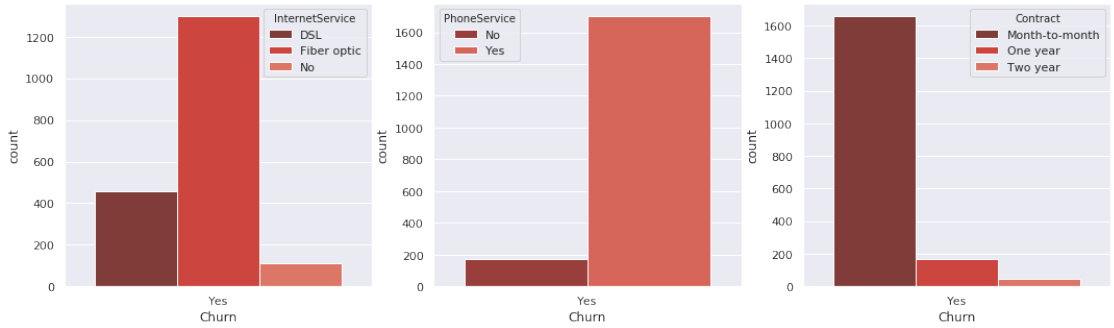
```
In [31]: plot_bars(ds_churn)
         plot_scatter_bw(ds_churn)
         plot_more_bars_dis(ds, ds_churn)
```




OnlineSecurity
Yes
No
No internet service
count
InternetService
DSL
Fiber optic
No
OnlineBackup
Yes
No
No internet service
TechSupport
No
Yes
No internet service
DeviceProtection
No
Yes
No internet service


TotalCharges
gender
Male
Female
SeniorCitizen
0
1
Partner
No
Yes
Dependents


TotalCharges
gender
Male
Female
SeniorCitizen
Partner
Dependents



I Lifetime value of loyal customers

```
In [18]: data = ds_loyal.drop(["gender","SeniorCitizen","Partner","Dependents","PaperlessBilling"
         ,"PaymentMethod","Churn"], axis=1)
```

```
In [19]: data = data.dropna()
```

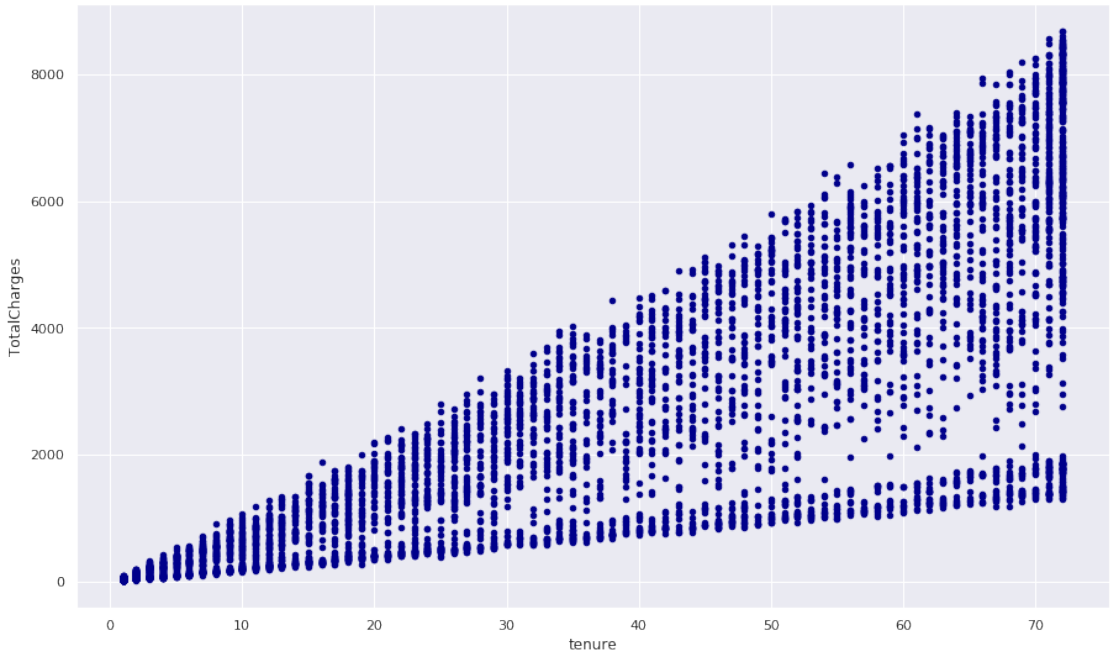
```
In [20]: data = data.replace('No phone service', 'No')
         data = data.replace('No internet service', 'No')
         data = data.replace('DSL', 'Yes')
         data = data.replace('Fiber optic', 'Yes')
```

```
In [21]: data['ServiceCount'] = (data.iloc[:,1:] == 'Yes').sum(axis=1)
```

```
In [22]: data = data.drop(['PhoneService','MultipleLines','InternetService','OnlineSecurity','Onl
         ineBackup','DeviceProtection','TechSupport','StreamingTV','StreamingMovies','Contract'],
         axis=1)
```

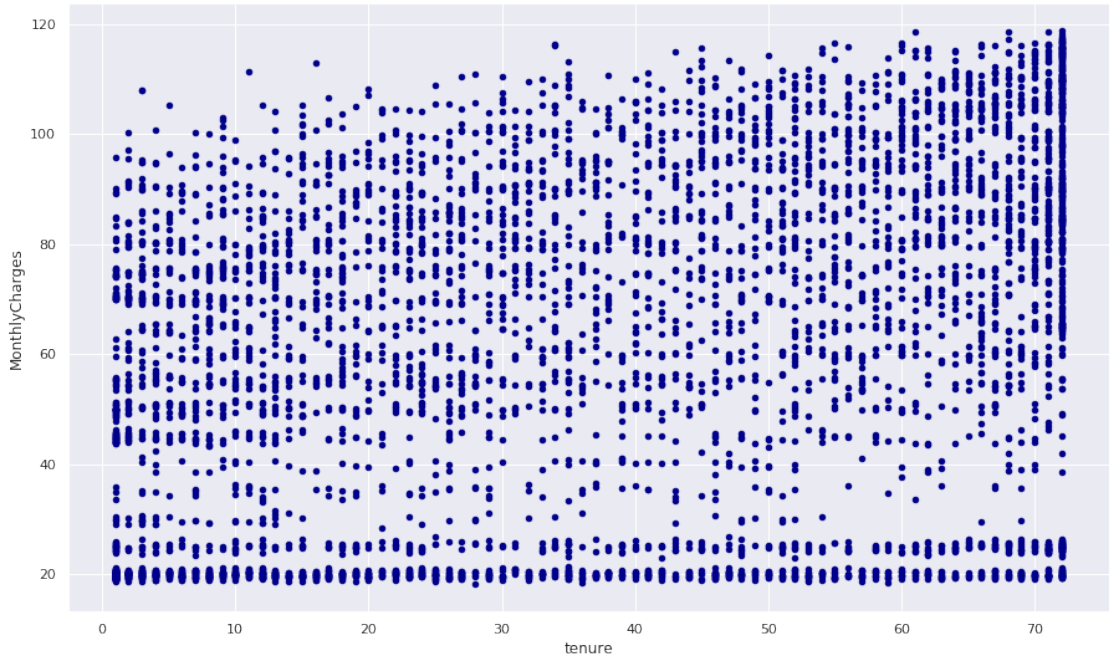
```
In [23]: data = data.astype(float)
```

```
In [24]: #plt.figure()
         data.plot.scatter(x='tenure',y='TotalCharges',color='DarkBlue',figsize=(15,9))
         plt.show()
```



The above graph shows that there is a linear relationship between the tenure and the total charges. However, the graph also shows that there are many users who use the basic service consistently for long period of time. This is illustrated by the lower part of the plot which has high density.

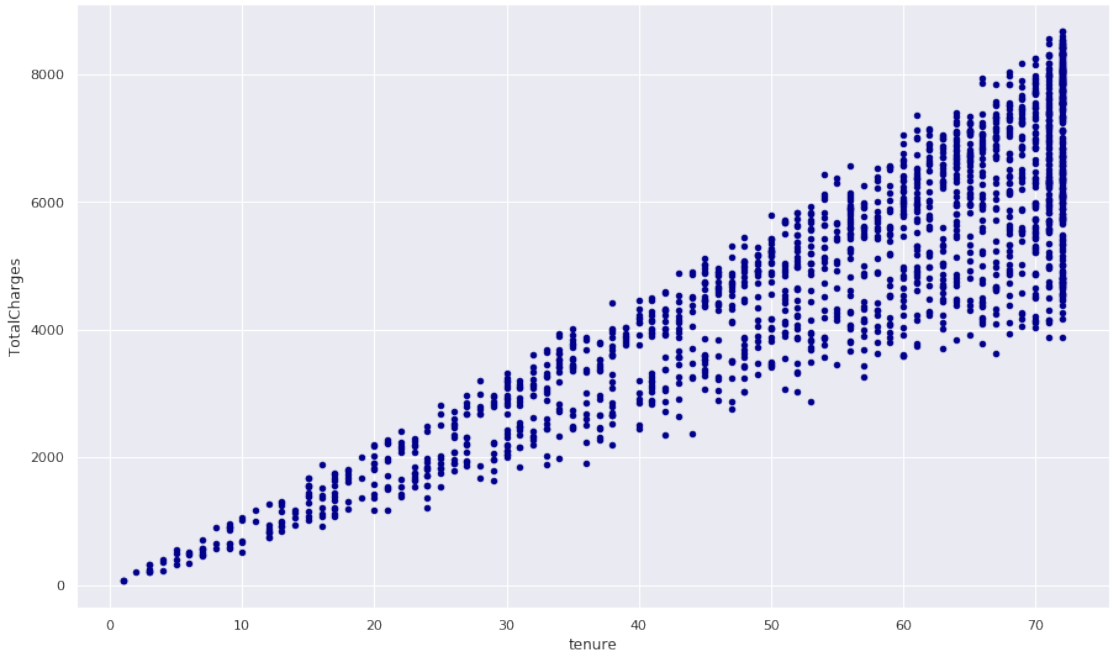
```
In [25]: scatter2 =
         data.plot.scatter(x='tenure',y='MonthlyCharges',color='DarkBlue',figsize=(15,9))
         plt.show()
```



The above plot for monthly charges does not show clear visual representation of the data. Therefore, we divide the data into two segments. People who use either more or less than five services.

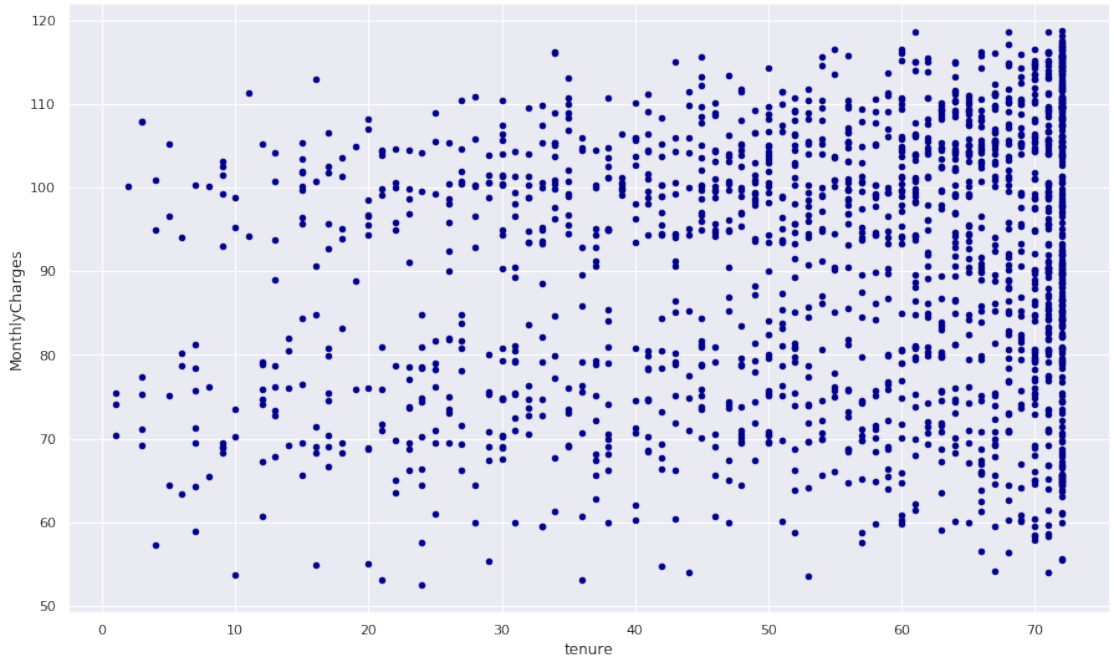
```
In [26]: data1 = data[data.ServiceCount > 5]
         data2 = data[data.ServiceCount < 5]
```

```
In [27]: scatter3 =
         data1.plot.scatter(x='tenure',y='TotalCharges',color='DarkBlue',figsize=(15,9))
         plt.show()
```



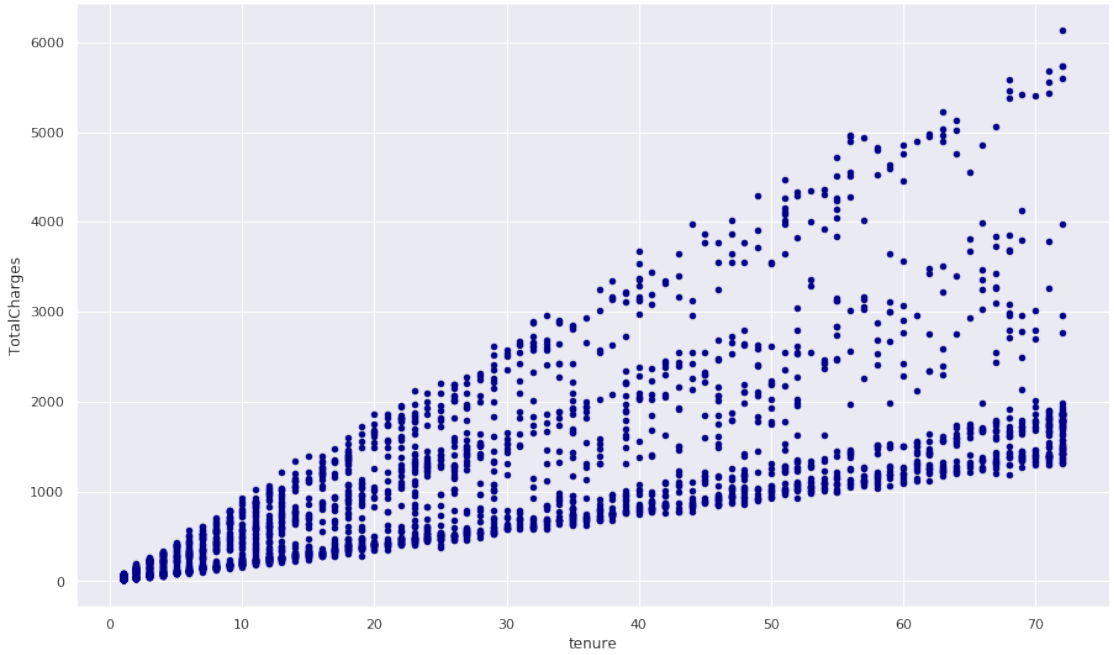
The above plot illustrated linear increase of total charges of customers who use more than five services. It is clear in the plot that longer they use more they pay.

```
In [28]: scatter4 =
         data1.plot.scatter(x='tenure',y='MonthlyCharges',color='DarkBlue',figsize=(15,9))
         plt.show()
```



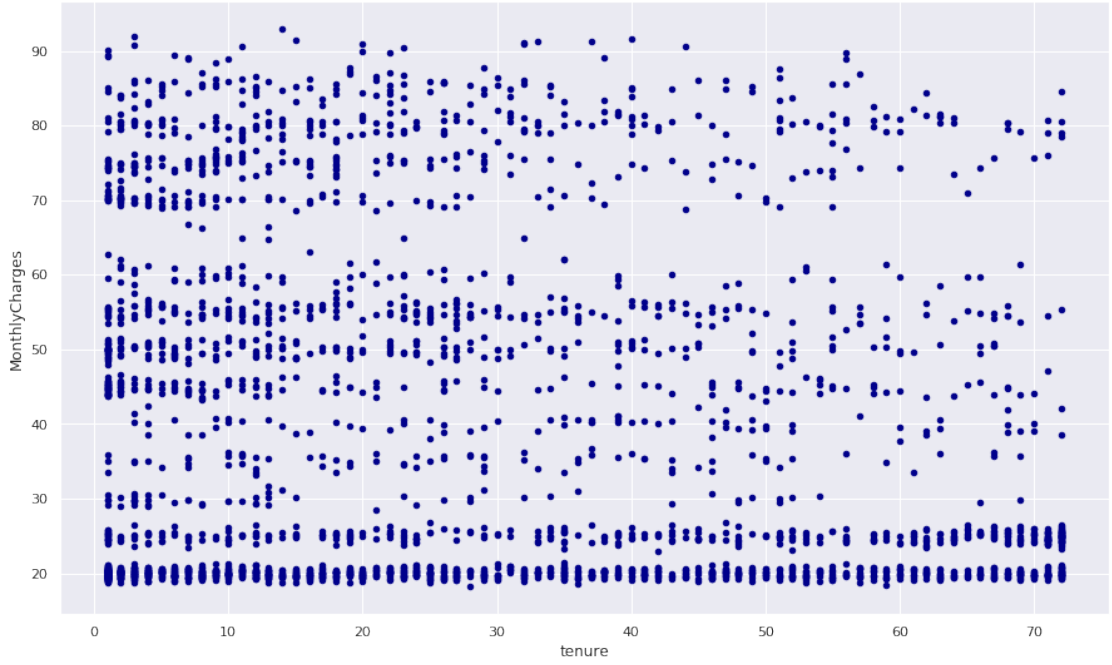
The above plot shows monthly charges of the customers who use more than five services. It is clear from the graph that loyal customers tend to take more services. Thus, the plot is more dense on the right hand side.

```
In [29]: scatter5 =
         data2.plot.scatter(x='tenure',y='TotalCharges',color='DarkBlue',figsize=(15,9))
         plt.show()
```



The above plot represents the total charges of customers who use less than five services. The graph is more dense at the beginning which means newer customers use less services. Also, there are many customers who stick to the basic services for the long term as the lower part of the graph is consistently thick from beginning untill the end.

```
In [30]: scatter6 =
         data2.plot.scatter(x='tenure',y='MonthlyCharges',color='DarkBlue',figsize=(15,9))
         plt.show()
```



The above plot illustrates the monthly charges of customers who use less than five services. As it is shown on the graph, the customers who use less than five services are more likely to be newer customers.