Customer Churn Prediction

Data Visualization

Problem Definition:

Data visualization plays a pivotal role in customer churn prediction, a critical aspect of customer relationship management for businesses across various industries. By harnessing the power of visual representation, organizations can transform complex datasets into actionable insights. Through intuitive graphs, charts, and dashboards, data visualization enables decision-makers to identify trends, patterns, and potential churn triggers with remarkable clarity. These visual aids not only make it easier to spot at-risk customers but also facilitate the communication of these insights to various stakeholders within the company. Whether it's a heat map illustrating customer engagement over time or a predictive model's performance metrics displayed graphically, data visualization empowers businesses to proactively address churn, enhance customer retention strategies, and ultimately bolster their bottom line. In essence, it transforms raw data into a strategic asset, helping organizations stay ahead of the curve in an increasingly competitive market landscape.

Required steps:

a. Univariate Analysis: Start with univariate visualization to understand the distribution of individual features, for instance, using histograms or bar charts.

b. Bivariate Analysis: Explore relationships between pairs of variables to uncover potential correlations or patterns. This can be done through scatter plots, box plots, or correlation matrices.

c. Time Series Analysis: If applicable, create time series visualizations to track customer behavior and churn rates over time. Line charts or heatmaps can be useful here.

d. Dashboard Creation: Build interactive dashboards that provide a holistic view of the data. Tools like Tableau, Power BI, or Python libraries like Plotly and Dash can be used to create these dashboards.

e. Heatmaps: Visualize customer churn risk factors and correlations using heatmaps. This can help in identifying which combinations of variables are most influential.

f. Cohort Analysis: Visualize customer behavior within cohorts, allowing for comparisons between different groups of customers.

g. Geospatial Analysis: If applicable, use geospatial visualization to understand churn patterns across geographic regions.

h. Alerts and Monitoring: Create real-time dashboards and alerts to monitor customer churn in real-time, allowing for immediate intervention if necessary.

Python code :

(a)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')

Input data files are available in the "../input/" directory.

import os

print(os.listdir("../input"))

Any results you write to the current directory are saved as output.

(b)telecom_cust = pd.read_csv('/kaggle/input/churn-prediction/WA_Fn-UseC_-Telco-Customer-Churn.csv')

(c)Data exploration:

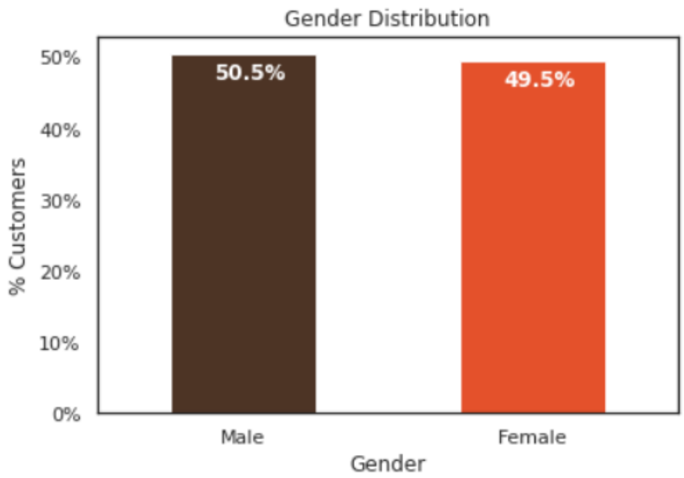
Demographics - Let us first understand the gender, age range, patner and dependent status of the customers

1.Gender Distribution - About half of the customers in our data set are male while the other half are female

```
colors = ['#4D3425','#E4512B']
ax = (telecom_cust['gender'].value_counts()*100.0 /len(telecom_cust)).plot(kind='bar',
                                                                           stacked = True,
                                                                           rot = 0,
                                                                           color = colors)
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers')
```

```
ax.set_xlabel('Gender')
ax.set_ylabel('% Customers')
ax.set_title('Gender Distribution')

# create a list to collect the plt.patches data
totals = []

# find the values and append to list
for i in ax.patches:
    totals.append(i.get_width())

# set individual bar lables using above list
total = sum(totals)

for i in ax.patches:
    # get_width pulls left or right; get_y pushes up or down
    ax.text(i.get_x()+.15, i.get_height()-3.5, \
            str(round((i.get_height()/total), 1))+'%',
            fontsize=12,
            color='white',
           weight = 'bold')
```

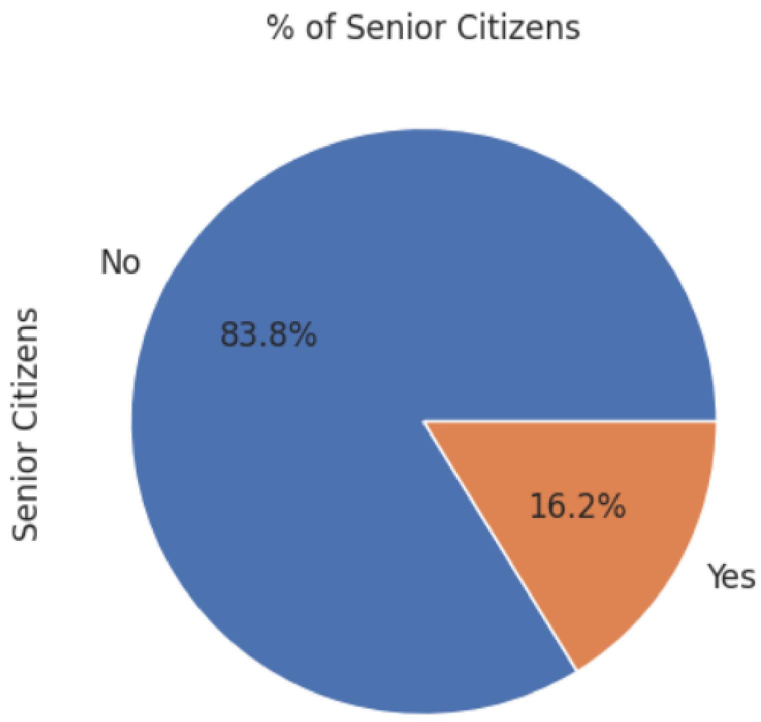
Output:



1. **% Senior Citizens** - There are only 16% of the customers who are senior citizens. Thus most of our customers in the data are younger people.

```
plt.figure(figsize=(12, 6))
sns.boxplot(x='churn', y='monthly_charges', data=data, palette='Set3')
plt.title('Monthly Charges vs. Churn')
plt.xlabel('Churn')
plt.ylabel('Monthly Charges')
plt.show()
```

Output:

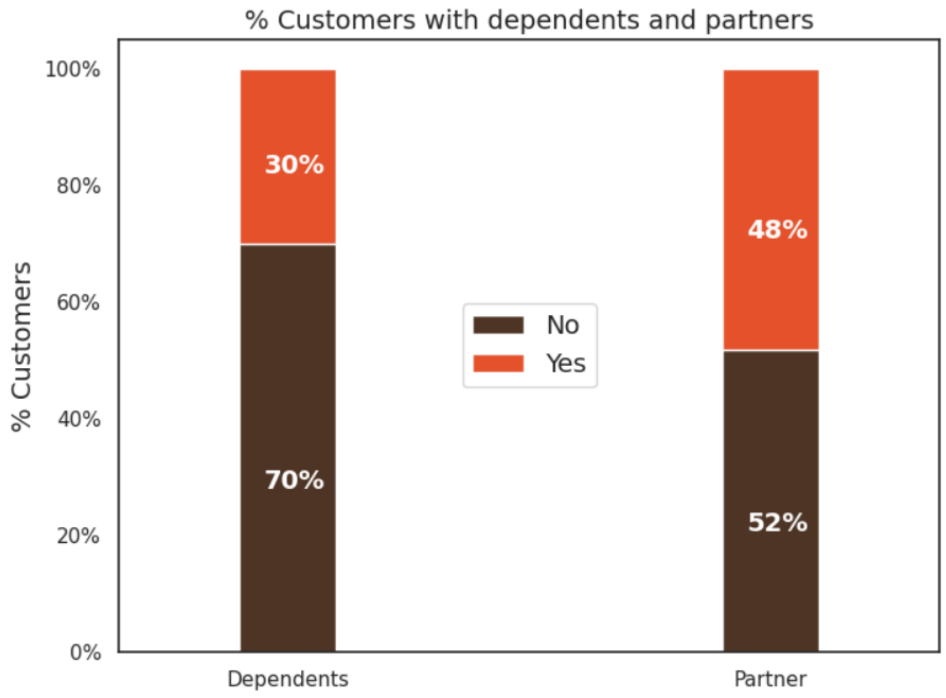


1. **Partner and dependent status** - About 50% of the customers have a partner, while only 30% of the total customers have dependents.

```
df2 = pd.melt(telecom_cust, id_vars=['customerID'], value_vars=['Dependents','Partner'])

df3 = df2.groupby(['variable','value']).count().unstack()

df3 = df3*100/len(telecom_cust)

colors = ['#4D3425','#E4512B']

ax = df3.loc[:,'customerID'].plot.bar(stacked=True, color=colors,

                                       figsize=(8,6),rot = 0,

                                      width = 0.2)
```

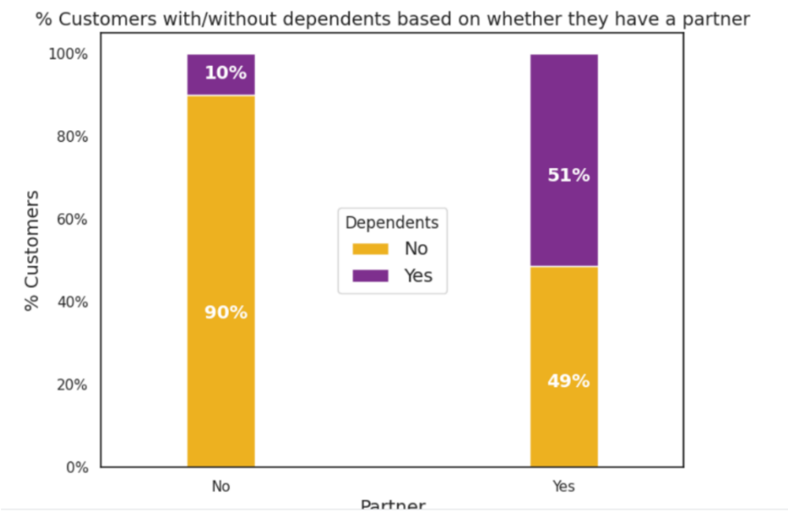
```
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers',size = 14)
ax.set_xlabel('')
ax.set_title('% Customers with dependents and partners',size = 14)
ax.legend(loc = 'center',prop={'size':14})

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)
```

Output:



```
colors = ['#4D3425','#E4512B']
partner_dependents = telecom_cust.groupby(['Partner','Dependents']).size().unstack()

ax = (partner_dependents.T*100.0 / partner_dependents.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 = 'Dependents',fontsize =14)
ax.set_ylabel('% Customers',size = 14)
ax.set_title('% Customers with/without dependents based on whether they have a partner',size = 14)
ax.xaxis.label.set_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)
```

Output:



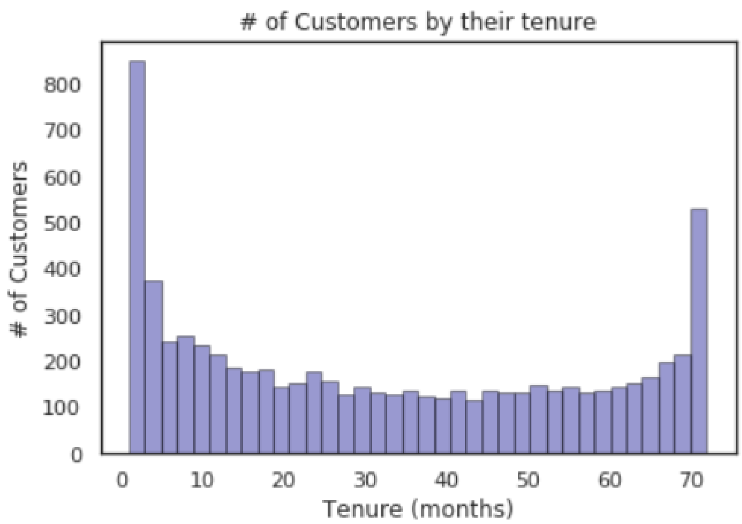
<u>Customer Account Information:</u>

1. Tenure: After looking at the below histogram we can see that a lot of customers have been with the telecom company for just a month, while quite a many are there for about 72 months. This could be potentially because different customers have different contracts. Thus based on the contract they are into it could be more/less easier for the customers to stay/leave the telecom company.

```
ax = sns.distplot(telecom_cust['tenure'], hist=True, kde=False,
            bins=int(180/5), color = 'darkblue',
            hist_kws={'edgecolor':'black'},
            kde_kws={'linewidth': 4})
ax.set_ylabel('# of Customers')
ax.set_xlabel('Tenure (months)')
ax.set_title('# of Customers by their tenure')
```

Output:

```
Text(0.5,1,'# of Customers by their tenure')
```

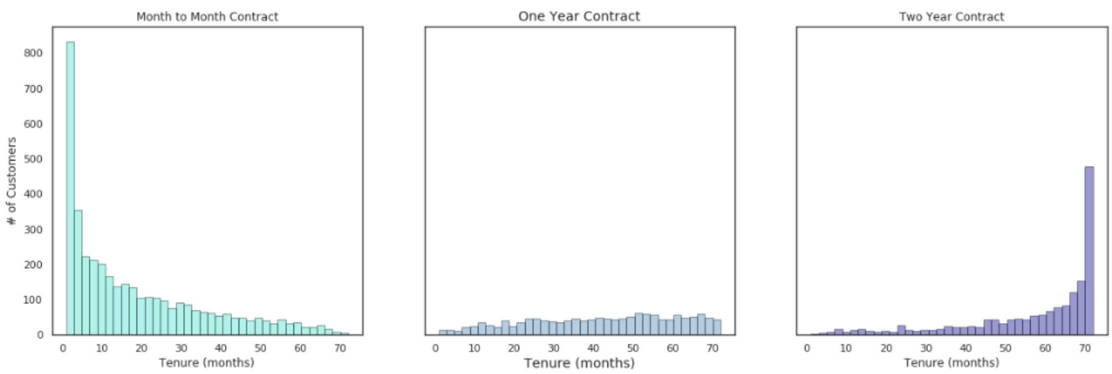


2.Contracts:

```
fig, (ax1,ax2,ax3) = plt.subplots(nrows=1, ncols=3, sharey = True, figsize = (20,6))
ax = sns.distplot(telecom_cust[telecom_cust['Contract']=='Month-to-month']['tenure'],
                hist=True, kde=False,
                bins=int(180/5), color = 'turquoise',
                hist_kws={'edgecolor':'black'},
                kde_kws={'linewidth': 4},
```

```
                ax=ax1)
ax.set_ylabel('# of Customers')
ax.set_xlabel('Tenure (months)')
ax.set_title('Month to Month Contract')
ax = sns.distplot(telecom_cust[telecom_cust['Contract']=='One year']['tenure'],
                  hist=True, kde=False,
                  bins=int(180/5), color = 'steelblue',
                  hist_kws={'edgecolor':'black'},
                  kde_kws={'linewidth': 4},
                ax=ax2)
ax.set_xlabel('Tenure (months)',size = 14)
ax.set_title('One Year Contract',size = 14)

ax = sns.distplot(telecom_cust[telecom_cust['Contract']=='Two year']['tenure'],
                  hist=True, kde=False,
                  bins=int(180/5), color = 'darkblue',
                  hist_kws={'edgecolor':'black'},
                  kde_kws={'linewidth': 4},
                ax=ax3)
ax.set_xlabel('Tenure (months)')
ax.set_title('Two Year Contract')
```

Output:

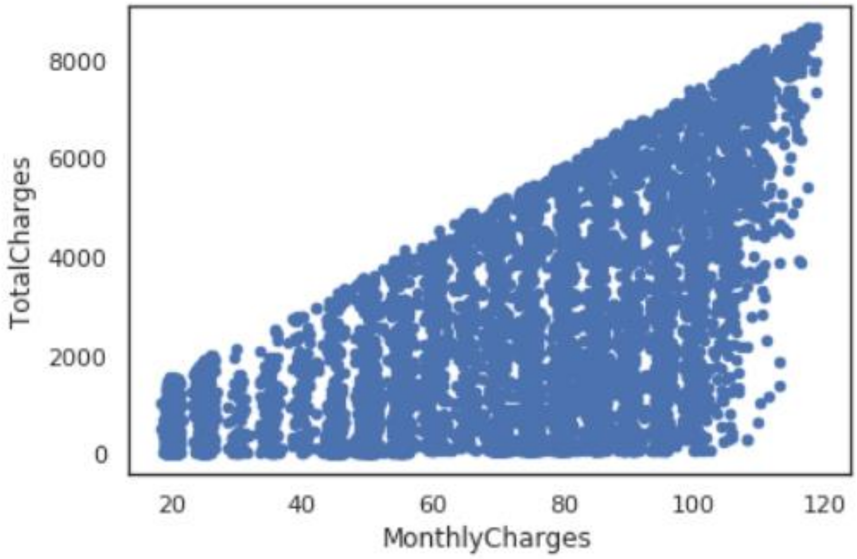


telecom_cust.columns.values

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

```
telecom_cust[['MonthlyCharges', 'TotalCharges']].plot.scatter(x = 'MonthlyCharges',
                                                              y='TotalCharges')
```

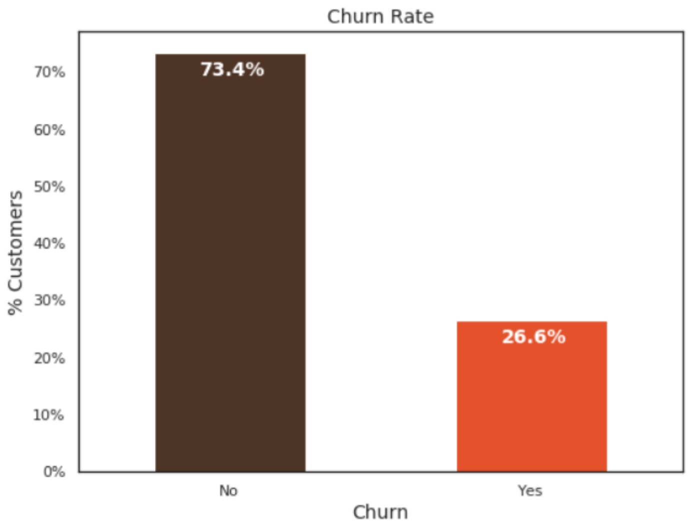
Output:



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.

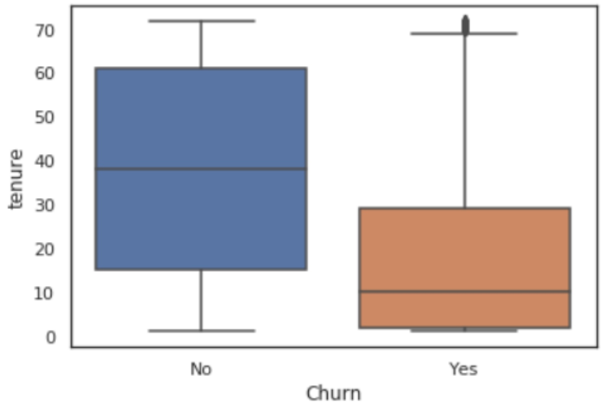
```
colors = ['#4D3425','#E4512B']
ax = (telecom_cust['Churn'].value_counts()*100.0 /len(telecom_cust)).plot(kind='bar',
                                                                           stacked = True,
                                                                           rot = 0,
                                                                           color = colors,
                                                                           figsize = (8,6))
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers',size = 14)
```

```
ax.set_xlabel('Churn',size = 14)
ax.set_title('Churn Rate', size = 14)

# create a list to collect the plt.patches data
totals = []

# find the values and append to list
for i in ax.patches:
    totals.append(i.get_width())

# set individual bar lables using above list
total = sum(totals)

for i in ax.patches:
    # get_width pulls left or right; get_y pushes up or down
    ax.text(i.get_x()+.15, i.get_height()-4.0, \
            str(round((i.get_height()/total), 1))+'%',
            fontsize=12,
            color='white',
           weight = 'bold',
           size = 14)
```



sns.boxplot(x = telecom_cust.Churn, y = telecom_cust.tenure)

Output:



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)

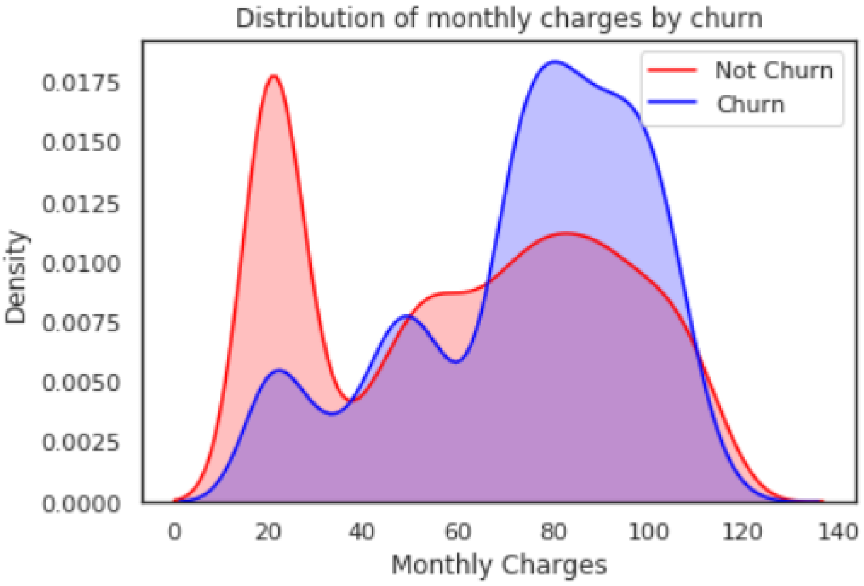
ax.legend(["Not Churn","Churn"],loc='upper right')

ax.set_ylabel('Density')

ax.set_xlabel('Monthly Charges')

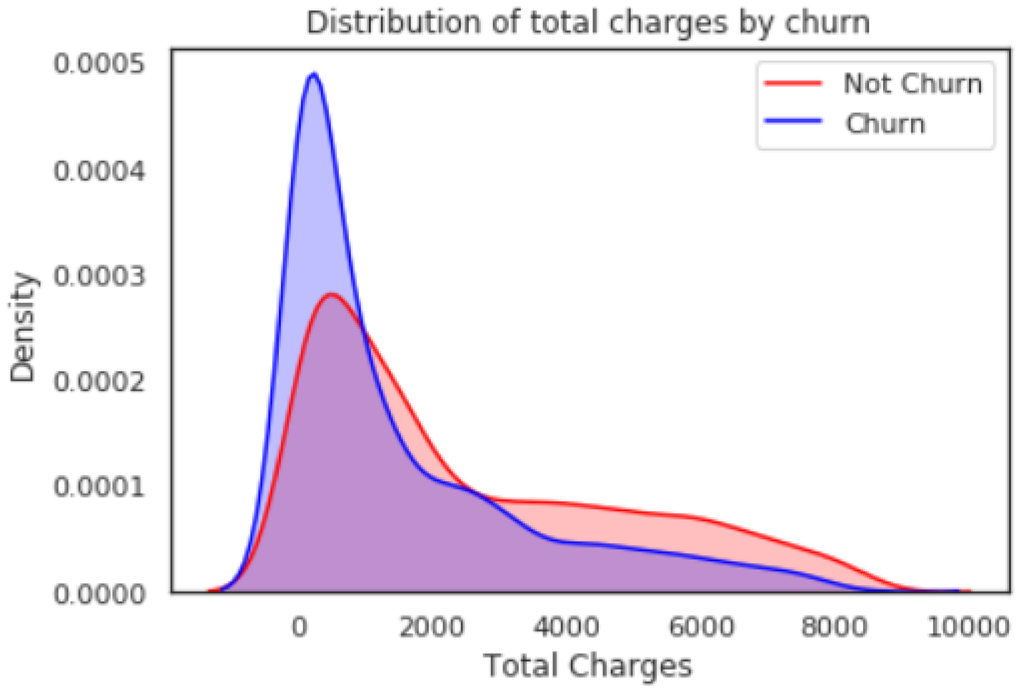
ax.set_title('Distribution of monthly charges by churn')

Output:




```
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')
```

Output:



<u>Conclusion:</u>

In conclusion, data visualization plays a pivotal role in customer churn prediction for telecom companies and various other industries. It serves as a powerful tool for understanding complex data patterns, communicating insights, and making informed decisions. Through visualizations, we can easily identify trends, correlations, and outliers within our customer data. This, in turn, empowers telecom businesses to take proactive steps in reducing churn rates. By creating visual representations of data, such as scatter plots, bar charts, heatmaps, and more, we gain a clearer understanding of customer behavior, enabling us to pinpoint areas that require attention, refine marketing strategies, and enhance customer retention efforts. Data visualization is a crucial component of the customer churn prediction process, enabling companies to stay competitive and responsive in today's data-driven landscape.