```
!pip install branca plotly scikit-learn imblearn --quiet
!pip install xgboost --quiet
!pip install h3 folium --quiet
```

```
^C
ERROR: Operation cancelled by user
```

```
import numpy as np
from IPython.display import Image
import branca.colormap as cm
import pandas as pd
import seaborn as sns
import plotly.express as px
from plotly.offline import init_notebook_mode, iplot
import seaborn as sns
import matplotlib.pyplot as plt
from imblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline
from xgboost import XGBClassifier
import h3
import matplotlib
import imblearn
import os
```

```
data = pd.read_excel('Telco_customer_churn.xlsx')
data.sample(5)
```

	CustomerID	Count	Country	State	City	Zip Code	Lat Long	Latitude	Longitude	Gender	...	Contract	Paperless Billing
871	1513-XNPPH	1	United States	California	Los Angeles	90037	34.002642, -118.287596	34.002642	-118.287596	Female	...	Month-to-month	Yes
3103	1060-ENTOF	1	United States	California	Los Angeles	90028	34.099869, -118.326843	34.099869	-118.326843	Female	...	One year	Yes
1401	9526-JAWYF	1	United States	California	Blythe	92225	33.674583, -114.71612	33.674583	-114.716120	Male	...	Month-to-month	No
3219	6908-VVYHM	1	United States	California	Pasadena	91104	34.165383, -118.123752	34.165383	-118.123752	Male	...	Month-to-month	Yes
6245	2862-PFNIK	1	United States	California	San Leandro	94578	37.704384, -122.126703	37.704384	-122.126703	Male	...	Month-to-month	Yes

5 rows × 33 columns

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 33 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   CustomerID         7043 non-null   object 
 1   Count              7043 non-null   int64  
 2   Country            7040 non-null   object 
 3   State              7040 non-null   object 
 4   City               7040 non-null   object 
 5   Zip Code           7043 non-null   int64  
 6   Lat Long           7043 non-null   object 
 7   Latitude           7043 non-null   float64
 8   Longitude          7043 non-null   float64
 9   Gender             7043 non-null   object 
 10  Senior Citizen     7043 non-null   object 
 11  Partner            7043 non-null   object 
 12  Dependents         7043 non-null   object 
 13  Tenure Months      7043 non-null   int64  
 14  Phone Service      7043 non-null   object 
 15  Multiple Lines     7043 non-null   object 
 16  Internet Service   7043 non-null   object 
 17  Online Security    7035 non-null   object 
 18  Online Backup      7038 non-null   object 
 19  Device Protection  7043 non-null   object 
 20  Tech Support       7043 non-null   object 
 21  Streaming TV       7043 non-null   object 
 22  Streaming Movies   7043 non-null   object 
 23  Contract           7043 non-null   object 
 24  Paperless Billing  7043 non-null   object 
 25  Payment Method     7043 non-null   object 
 26  Monthly Charges    7043 non-null   float64
 27  Total Charges      7043 non-null   object 
 28  Churn Label        7043 non-null   object 
 29  Churn Value        7043 non-null   int64
 30  Churn Score        7043 non-null   int64
 31  CLTV               7043 non-null   int64
 32  Churn Reason       1869 non-null   object
dtypes: float64(3), int64(6), object(24)
memory usage: 1.8+ MB
```

- what services customers use,
- type of contract
- the lifetime of the client in the service
- payment method
- the amount of monthly payments of customers and their total costs in the service,
- customer locations,
- gender and age of the client
- reason for churn (for clients in the churn)

```
data['Total Charges']
```

```
0        108.15
1        151.65
2         820.5
3       3046.05
4        5036.3
         ...
7038     1419.4
7039     1990.5
7040     7362.9
7041     346.45
7042     6844.5
Name: Total Charges, Length: 7043, dtype: object
```

```
data['Total Charges'] = pd.to_numeric(data['Total Charges'], errors='coerce')
data['Total Charges']
```

```
0        108.15
1        151.65
2        820.50
3       3046.05
4       5036.30
         ...
7038    1419.40
7039    1990.50
7040    7362.90
7041     346.45
7042    6844.50
Name: Total Charges, Length: 7043, dtype: float64
```

```
data.isnull().sum()
```

```
CustomerID              0
Count                   0
Country                 3
State                   3
City                    3
Zip Code                0
Lat Long                0
Latitude                0
Longitude               0
Gender                  0
Senior Citizen          0
Partner                 0
Dependents              0
Tenure Months           0
Phone Service           0
Multiple Lines          0
Internet Service        0
Online Security         8
Online Backup           5
Device Protection       0
Tech Support            0
Streaming TV            0
Streaming Movies        0
Contract                0
Paperless Billing       0
Payment Method          0
Monthly Charges         0
Total Charges          11
Churn Label             0
Churn Value             0
Churn Score             0
CLTV                    0
Churn Reason         5174
dtype: int64
```

```
data.groupby('Churn Label')['CustomerID'].nunique()
```

```
Churn Label
No     5174
Yes    1869
Name: CustomerID, dtype: int64
```

```
data[data['Total Charges'].isna()]
```

	CustomerID	Count	Country	State	City	Zip Code	Lat Long	Latitude	Longitude	Gender	...	Co
2234	4472-LVYGI	1	United States	California	San Bernardino	92408	34.084909, -117.258107	34.084909	-117.258107	Female	...	
2438	3115-CZMZD	1	United States	California	Independence	93526	36.869584, -118.189241	36.869584	-118.189241	Male	...	
2568	5709-LVOEQ	1	United States	California	San Mateo	94401	37.590421, -122.306467	37.590421	-122.306467	Female	...	
2667	4367-NUYAO	1	United States	California	Cupertino	95014	37.306612, -122.080621	37.306612	-122.080621	Male	...	
2856	1371-DWPAZ	1	United States	California	Redcrest	95569	40.363446, -123.835041	40.363446	-123.835041	Female	...	
4331	7644-OMVMY	1	United States	California	Los Angeles	90029	34.089953, -118.294824	34.089953	-118.294824	Male	...	
4687	3213-VVOLG	1	United States	California	Sun City	92585	33.739412, -117.173334	33.739412	-117.173334	Male	...	
5104	2520-SGTTA	1	United States	California	Ben Lomond	95005	37.078873, -122.090386	37.078873	-122.090386	Female	...	
5719	2923-ARZLG	1	United States	California	La Verne	91750	34.144703, -117.770299	34.144703	-117.770299	Male	...	
6772	4075-WKNIU	1	United States	California	Bell	90201	33.970343, -118.171368	33.970343	-118.171368	Female	...	
6840	2775-SEFEE	1	United States	California	Wilmington	90744	33.782068, -118.262263	33.782068	-118.262263	Male	...	

11 rows × 33 columns

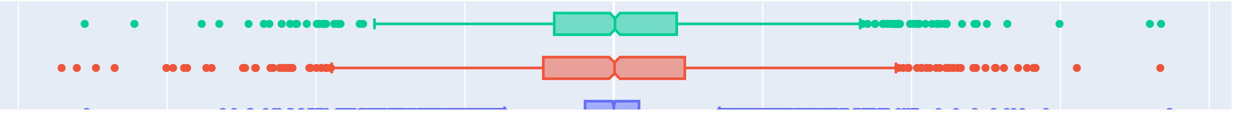
```
data.dropna(subset=['Country', 'State', 'City'], inplace=True)
```

```
data['calc_charges'] = data['Monthly Charges'] * data['Tenure Months']
```

calculating difference between Total Charges and calculated charges

```
data['diff_in_charges'] = data['Total Charges'] - data['calc_charges']
```

```
fig = px.histogram(data, x="diff_in_charges",color = 'Contract',marginal="box")
fig.show()
```



```
data.groupby('Contract')[['Total Charges','diff_in_charges']].quantile([.50,.80,.90,.95])
```

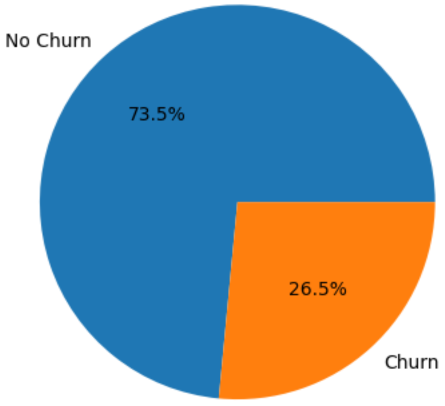
Contract		Total Charges	diff_in_charges
Month-to-month	0.50	679.9500	0.0000
	0.80	2492.4200	24.8400
	0.90	3846.7150	54.0800
	0.95	4968.0300	85.3450
One year	0.50	2657.5500	0.7750
	0.80	5286.4600	55.0500
	0.90	6341.2500	92.2000
	0.95	7072.4725	133.3375
Two year	0.50	3623.9500	0.5000
	0.80	6399.2400	61.5300
	0.90	7457.6100	97.5700
	0.95	7922.3400	139.1800

```
data['Total Charges'] = np.where(data['Total Charges'].isna() == True,data['calc_charges'], data['Total Charges'])
```

```
data = data.drop(['calc_charges','diff_in_charges'], axis=1)
```

```
plt.pie(data['Churn Label'].value_counts(), labels=['No Churn','Churn'], autopct='%1.1f%%')
```

```
([<matplotlib.patches.Wedge at 0x7927654e2ce0>,
  <matplotlib.patches.Wedge at 0x7927654e2380>],
 [Text(-0.7401686016429937, 0.8137262691727823, 'No Churn'),
  Text(0.740168677829542, -0.8137261998731931, 'Churn')],
 [Text(-0.4037283281689056, 0.44385069227606305, '73.5%'),
  Text(0.40372836972520465, -0.4438506544762871, '26.5%')])
```



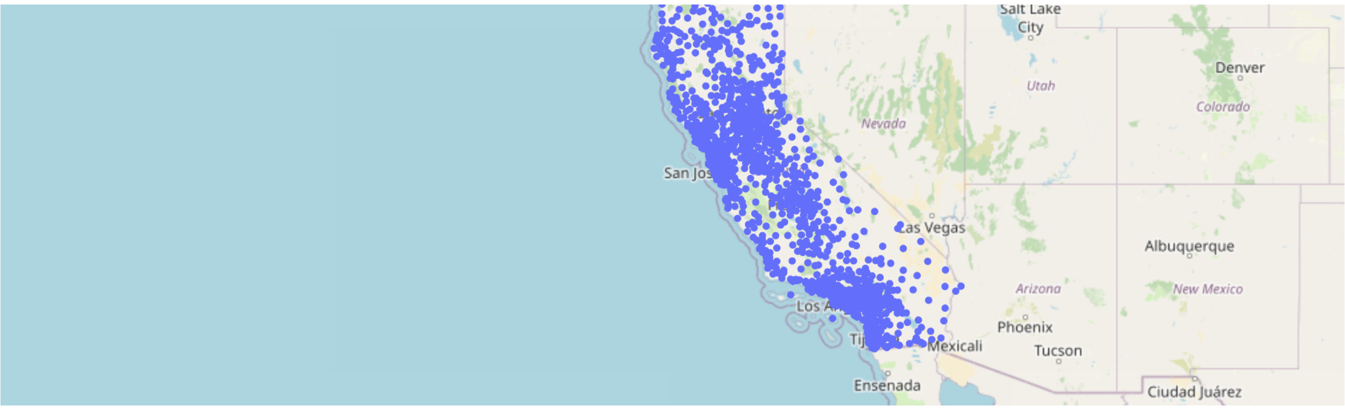
```
data.groupby(['Country','State'])['CustomerID'].count()
```

```
Country        State
United States  California    7040
Name: CustomerID, dtype: int64
```

```
data['City'].nunique()
```

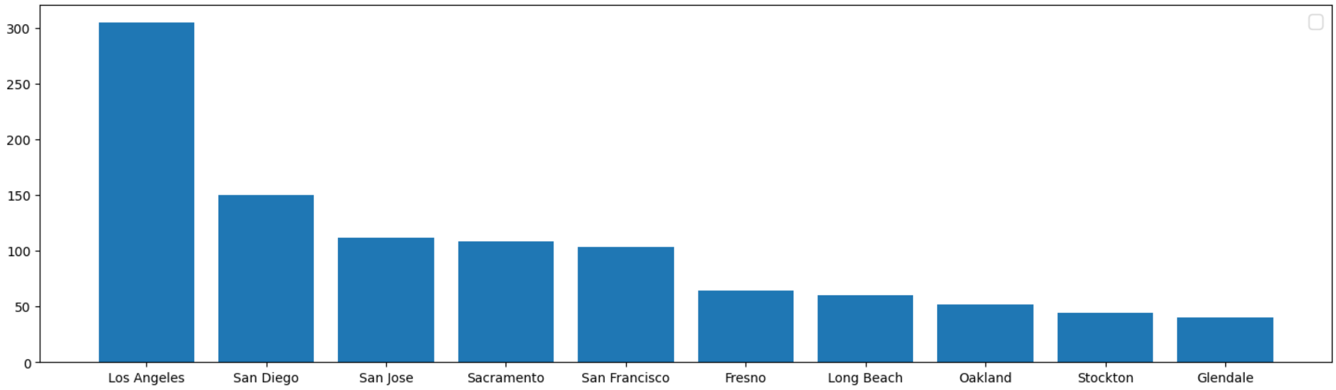
```
1129
```

```
fig = px.scatter_mapbox(data.groupby(['Latitude','Longitude'])['CustomerID'].count().reset_index(), lat="Latitude", lon=
fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```



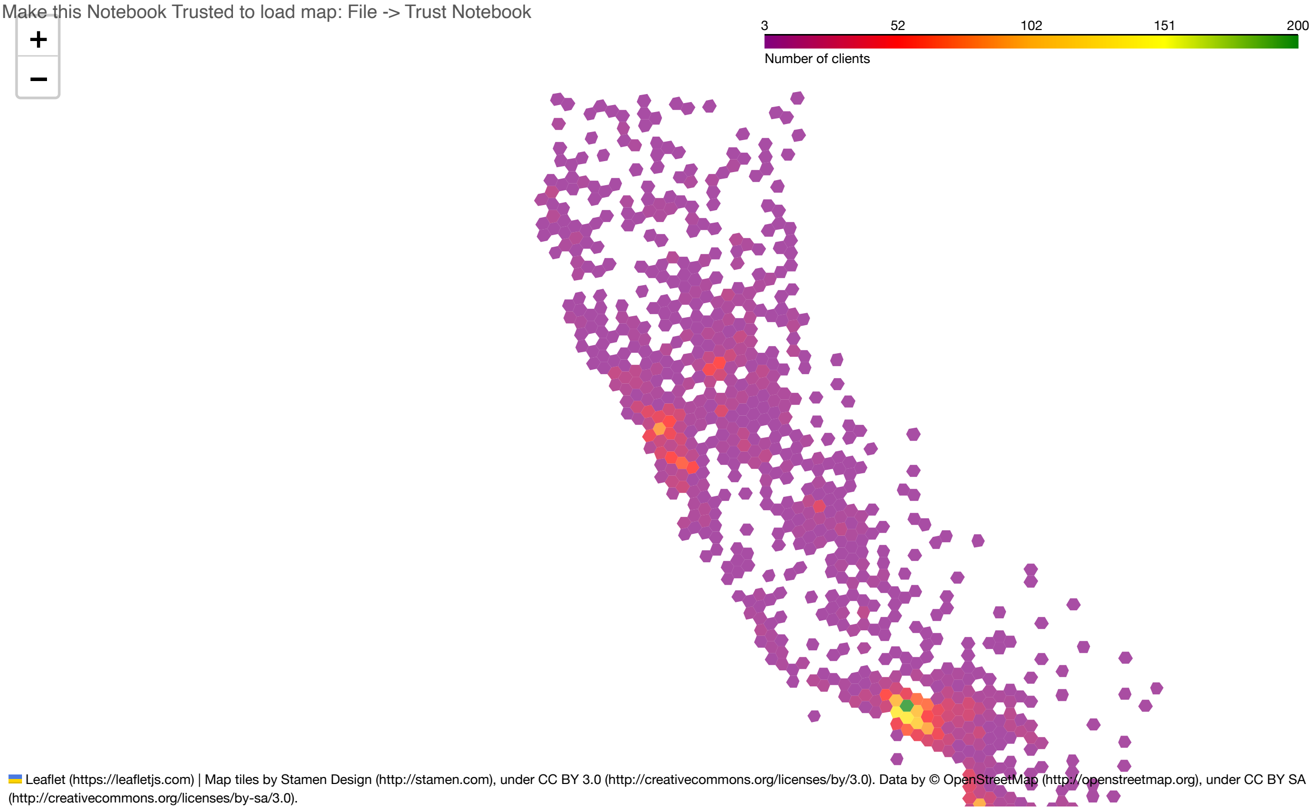
```
f, ax = plt.subplots(figsize=(18,5))
plt.bar(
  data.groupby('City')['CustomerID'].count().sort_values(ascending=False).head(
    10).index,
  data.groupby('City')['CustomerID'].count().sort_values(ascending=False).head(10).values,
)
ax.legend(fontsize = 14)

```

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that artists whose label start with an ur
<matplotlib.legend.Legend at 0x7927655c3a90>
```



```
hex_level = 5

data['hex_id'] = data.apply(lambda x: h3.geo_to_h3(x['Latitude'], x['Longitude'], hex_level), axis=1)

hex_counts = data.groupby('hex_id')['CustomerID'].count().reset_index(name='total_clients')
hex_counts['center'] = hex_counts['hex_id'].apply(lambda x: h3.h3_to_geo(x))


color_range = [hex_counts['total_clients'].min(), hex_counts['total_clients'].max()]
colormap = cm.LinearColormap(["purple","red","orange","yellow","green"],vmin = min(color_range), vmax = max(color_range)

mean_lat, mean_lon = hex_counts['center'].apply(lambda x: x[0]).mean(), hex_counts['center'].apply(lambda x: x[1]).mean(
map_center = [mean_lat, mean_lon]
m = folium.Map(location=map_center, zoom_start=6, tiles='Stamen Terrain')

for _, row in hex_counts.iterrows():
    folium.Polygon(
        locations=h3.h3_to_geo_boundary(row['hex_id']),
        fill=True,
        fill_color=colormap(row['total_clients']),
        fill_opacity=0.7,
        stroke=False,
        tooltip=f"Number of clients: {row['total_clients']}"
    ).add_to(m)

colormap.caption = 'Number of clients'
m.add_child(colormap)
```

Make this Notebook Trusted to load map: File -> Trust Notebook


+
−
3
52
102
151
200
Number of clients
Leaflet (https://leafletjs.com) | Map tiles by Stamen Design (http://stamen.com), under CC BY 3.0 (http://creativecommons.org/licenses/by/3.0). Data by © OpenStreetMap (http://openstreetmap.org), under CC BY SA (http://creativecommons.org/licenses/by-sa/3.0).

```
churn = data.assign(churn_clients = np.where(data['Churn Label']=='Yes',data['CustomerID'],None)).groupby(['hex_id']).ag
```

```
clients = data.groupby(['hex_id'])['CustomerID'].count().reset_index()
```

```
churn_data = clients.join(churn.set_index(['hex_id']), on=['hex_id'])
```

```
churn_data['churn_rate'] = churn_data['churn_clients']/churn_data['CustomerID']
```

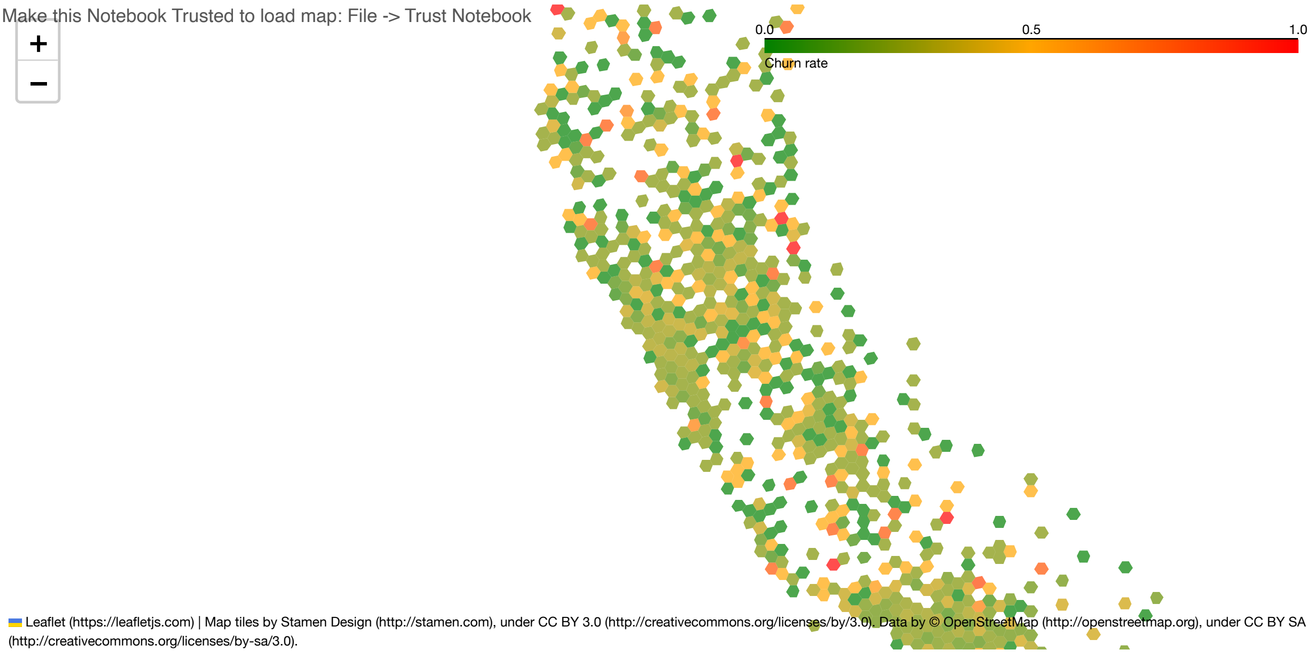
```
churn_data
```

	hex_id	CustomerID	churn_clients	churn_rate
0	85280043fffffff	4	1	0.25
1	8528004bfffffff	4	1	0.25
2	8528004ffffffff	4	1	0.25
3	85280207fffffff	4	1	0.25
4	8528020bfffffff	4	1	0.25
...	...	...	...	...
709	85485b03fffffff	5	1	0.20
710	85485b33fffffff	5	1	0.20
711	85485b63fffffff	5	0	0.00
712	85485babfffffff	5	3	0.60
713	85485bb7fffffff	10	3	0.30

714 rows × 4 columns

```
churn_data['center'] = churn_data['hex_id'].apply(lambda x: h3.h3_to_geo(x))


color_range = [churn_data['churn_rate'].min(), churn_data['churn_rate'].max()]
colormap = cm.LinearColormap(["green","orange","red"],vmin = min(color_range), vmax = max(color_range))

```

```
mean_lat, mean_lon = churn_data['center'].apply(lambda x: x[0]).mean(), churn_data['center'].apply(lambda x: x[1]).mean(
map_center = [mean_lat, mean_lon]
m = folium.Map(location=map_center, zoom_start=6,  width='100%', height='80%',tiles='Stamen Terrain')

for _, row in churn_data.iterrows():
    folium.Polygon(
        locations=h3.h3_to_geo_boundary(row['hex_id']),
        fill=True,
        fill_color=colormap(row['churn_rate']),
        fill_opacity=0.7,
        stroke=False,
        tooltip=f"Churn rate: {row['churn_rate']}<br>Number of customers: {row['CustomerID']}"
    ).add_to(m)

colormap.caption = 'Churn rate'
m.add_child(colormap)

m
```



Make this Notebook Trusted to load map: File -> Trust Notebook

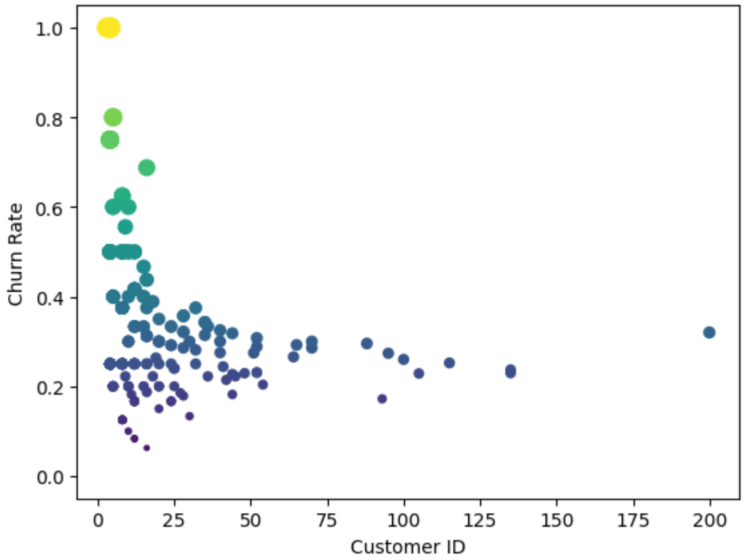
Leaflet (https://leafletjs.com) | Map tiles by Stamen Design (http://stamen.com), under CC BY 3.0 (http://creativecommons.org/licenses/by/3.0). Data by © OpenStreetMap (http://openstreetmap.org), under CC BY SA (http://creativecommons.org/licenses/by-sa/3.0).

```
plt.scatter(
  churn_data['CustomerID'],
  churn_data['churn_rate'],
)
```

```
<matplotlib.collections.PathCollection at 0x792763db7e50>
```

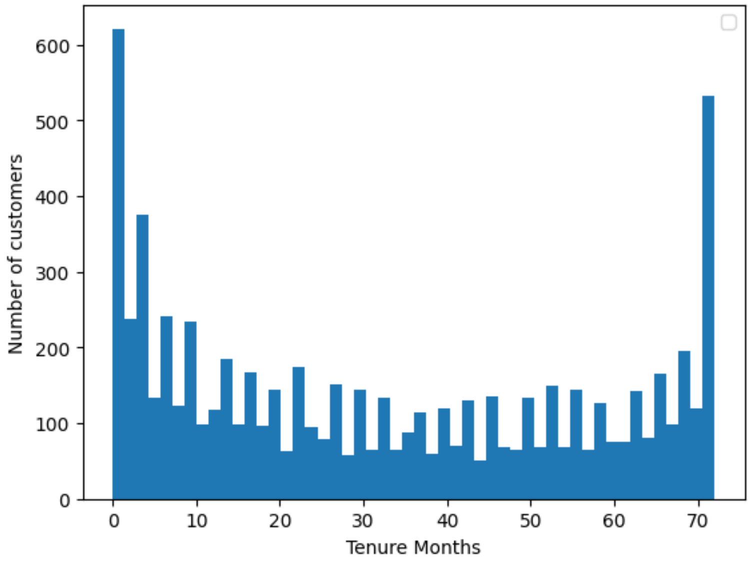
```
fig, ax = plt.subplots()
ax.scatter(churn_data['CustomerID'], churn_data['churn_rate'], s=churn_data['churn_rate']*100, c=churn_data['churn_rate'
ax.set_xlabel('Customer ID')
ax.set_ylabel('Churn Rate')
```

```
Text(0, 0.5, 'Churn Rate')
```



```
plt.hist(data['Tenure Months'], bins=50)
plt.legend()
plt.xlabel('Tenure Months')
plt.ylabel('Number of customers')
```

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that artists whose label start with an ur
Text(0, 0.5, 'Number of customers')
```



```
data.groupby('Churn Label')['Tenure Months'].quantile([.50,.75,.90,.95])
```

```
Churn Label
No           0.50    38.0
             0.75    61.0
             0.90    71.0
             0.95    72.0
Yes          0.50    10.0
             0.75    29.0
             0.90    51.0
             0.95    60.0
Name: Tenure Months, dtype: float64
```

```
data.groupby('Churn Label')['Tenure Months'].mean()
```

```
Churn Label
No     37.569965
Yes    18.006431
Name: Tenure Months, dtype: float64
```

```
grouped = data.groupby(['Churn Reason'])['CustomerID'].count().reset_index().sort_values('CustomerID',
                                                                                           ascending=False)
grouped
```

	Churn Reason	CustomerID
1	Attitude of support person	192
4	Competitor offered higher download speeds	189
5	Competitor offered more data	161
7	Don't know	153
3	Competitor made better offer	140
0	Attitude of service provider	135
2	Competitor had better devices	130
14	Network reliability	103
18	Product dissatisfaction	102
17	Price too high	98
19	Service dissatisfaction	89
10	Lack of self-service on Website	88
8	Extra data charges	56
13	Moved	53
11	Limited range of services	44
12	Long distance charges	44
9	Lack of affordable download/upload speed	44
16	Poor expertise of phone support	20
15	Poor expertise of online support	19
6	Deceased	6

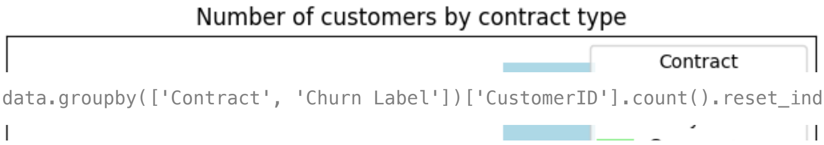
```
plt.bar(grouped['Churn Reason'], grouped['CustomerID'])
plt.xticks(rotation=90)
```

```
 Text(3, 0, 'Don't know'),
 Text(4, 0, 'Competitor made better offer'),
 Text(5, 0, 'Attitude of service provider'),
 Text(6, 0, 'Competitor had better devices'),
 Text(7, 0, 'Network reliability'),
 Text(8, 0, 'Product dissatisfaction'),
 Text(9, 0, 'Price too high'),
 Text(10, 0, 'Service dissatisfaction'),
 Text(11, 0, 'Lack of self-service on Website'),
 Text(12, 0, 'Extra data charges'),
 Text(13, 0, 'Moved'),
 Text(14, 0, 'Limited range of services'),
 Text(15, 0, 'Long distance charges'),
 Text(16, 0, 'Lack of affordable download/upload speed'),
 Text(17, 0, 'Poor expertise of phone support'),
 Text(18, 0, 'Poor expertise of online support'),
 Text(19, 0, 'Deceased')])
```

Contract types

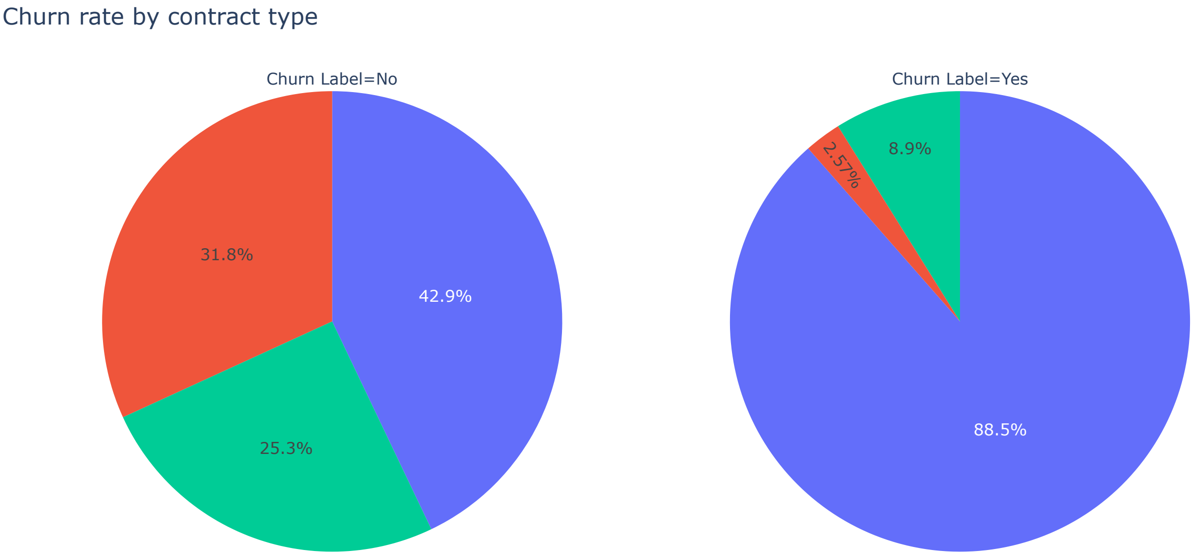
Let's see what types of contracts there are in the service and how this affects the churn rate.

```

# Assuming your data is in a DataFrame called 'data'
churn_labels = data['Churn Label'].unique()
contracts = data['Contract'].unique()

fig, ax = plt.subplots(figsize=(7, 5))

# Create a dictionary to map contract types to colors
contract_colors = {'Month-to-month': 'lightblue', 'One year': 'lightgreen', 'Two year': 'lightcoral'}

width = 0.2
x = np.arange(len(churn_labels))

for i, contract in enumerate(contracts):
    contract_data = data[data['Contract'] == contract]
    counts = [contract_data[contract_data['Churn Label'] == label]['Churn Label'].count() for label in churn_labels]
    ax.bar(x + i * width, counts, width=width, label=contract, color=contract_colors[contract])

ax.set_xlabel('Churn Label')
ax.set_ylabel('Count')
ax.set_title('Number of customers by contract type')
ax.set_xticks(x + width)
ax.set_xticklabels(churn_labels)
ax.legend(title='Contract', loc='upper right')

plt.tight_layout()
plt.show()
```



```
grouped = data.groupby(['Contract', 'Churn Label'])['CustomerID'].count().reset_index()
```

```
fig = px.pie(data.groupby(['Contract','Churn Label'])['CustomerID'].count().reset_index(),
             values='CustomerID',
            names='Contract',
            facet_col = 'Churn Label',
            title = 'Churn rate by contract type')

fig.show()
```



```
data.groupby(['Contract','Churn Label'])['Tenure Months'].mean()
```

```
Contract        Churn Label
Month-to-month  No             21.033333
                Yes            14.040557
One year        No             41.674063
                Yes            44.963855
Two year        No             56.602914
                Yes            61.270833
Name: Tenure Months, dtype: float64
```

Total charges

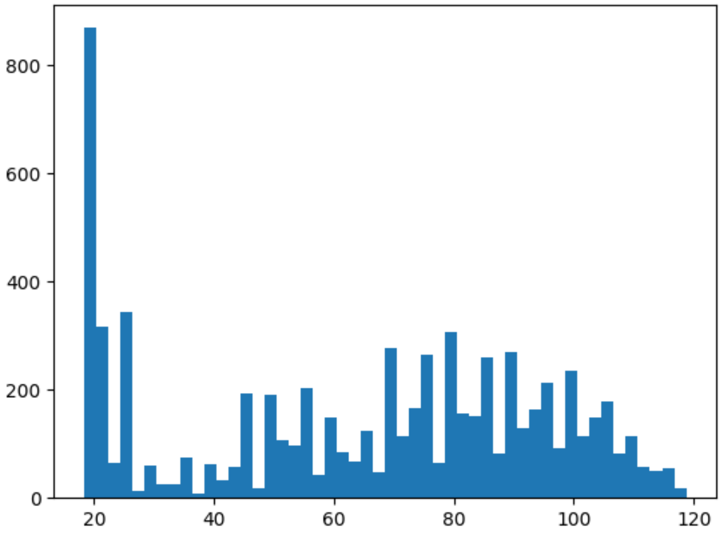
```
plt.hist(data['Total Charges'], bins=50)
```

```
(array([1098.,  547.,  404.,  337.,  312.,  270.,  272.,  270.,  235.,
         180.,  190.,  135.,  123.,  121.,  116.,  104.,   93.,  115.,
          99.,   92.,  100.,   79.,   99.,  100.,   82.,   80.,   86.,
          86.,  102.,   60.,   71.,   75.,   93.,   74.,   83.,   62.,
          73.,   62.,   58.,   56.,   57.,   44.,   44.,   44.,   35.,
          41.,   29.,   26.,   18.,    8.]),
 array([   0.   ,  173.696,  347.392,  521.088,  694.784,  868.48 ,
        1042.176, 1215.872, 1389.568, 1563.264, 1736.96 , 1910.656,
        2084.352, 2258.048, 2431.744, 2605.44 , 2779.136, 2952.832,
        3126.528, 3300.224, 3473.92 , 3647.616, 3821.312, 3995.008,
        4168.704, 4342.4  , 4516.096, 4689.792, 4863.488, 5037.184,
        5210.88 , 5384.576, 5558.272, 5731.968, 5905.664, 6079.36 ,
        6253.056, 6426.752, 6600.448, 6774.144, 6947.84 , 7121.536,
        7295.232, 7468.928, 7642.624, 7816.32 , 7990.016, 8163.712,
        8337.408, 8511.104, 8684.8  ]),
 <BarContainer object of 50 artists>)
```

Monthly Charges

```
plt.hist(data['Monthly Charges'], bins=50)
```

```
(array([868., 316.,  66., 343.,  13.,  59.,  25.,  25.,  74.,   8.,  62.,
         32.,  57., 194.,  19., 190., 107.,  96., 204.,  43., 148.,  85.,
         67., 125.,  48., 278., 115., 166., 265.,  66., 307., 156., 150.,
        259.,  82., 270., 130., 164., 213.,  93., 236., 114., 149., 177.,
         82., 115.,  58.,  50.,  54.,  17.]),
 array([ 18.25,  20.26,  22.27,  24.28,  26.29,  28.3 ,  30.31,  32.32,
         34.33,  36.34,  38.35,  40.36,  42.37,  44.38,  46.39,  48.4 ,
         50.41,  52.42,  54.43,  56.44,  58.45,  60.46,  62.47,  64.48,
         66.49,  68.5 ,  70.51,  72.52,  74.53,  76.54,  78.55,  80.56,
         82.57,  84.58,  86.59,  88.6 ,  90.61,  92.62,  94.63,  96.64,
         98.65, 100.66, 102.67, 104.68, 106.69, 108.7 , 110.71, 112.72,
        114.73, 116.74, 118.75]),
 <BarContainer object of 50 artists>)
```



```
data.groupby('Churn Label')['Monthly Charges'].quantile([.50,.75,.95,.99])
```

```
Churn Label
No           0.50     64.4250
             0.75     88.4000
             0.95    108.4175
             0.99    115.1000
Yes          0.50     79.6500
             0.75     94.2000
             0.95    105.6250
             0.99    111.1350
Name: Monthly Charges, dtype: float64
```

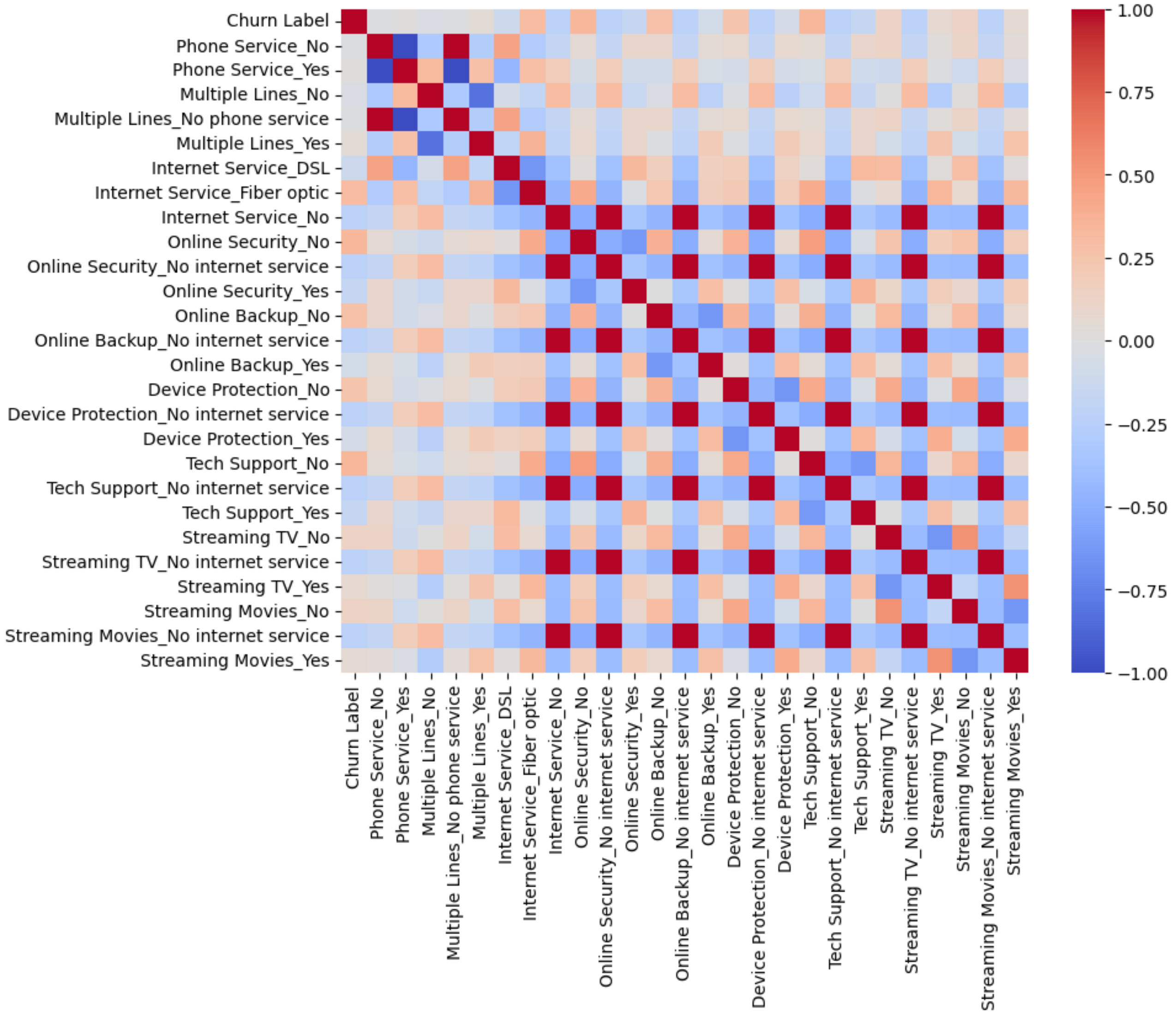
```
corr_df = data.copy()
```

```
corr_df['Churn Label'].replace(to_replace='Yes', value=1, inplace=True)
corr_df['Churn Label'].replace(to_replace='No',  value=0, inplace=True)
```

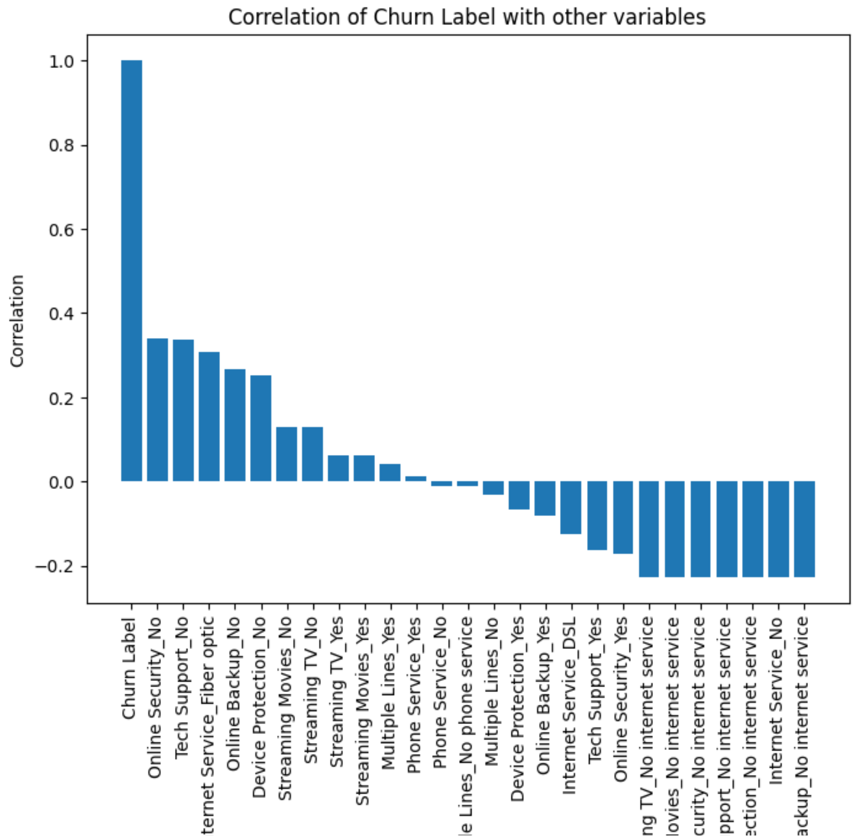
```
df_dummies = pd.get_dummies(corr_df[['Churn Label','Phone Service','Multiple Lines','Internet Service','Online Security'
                                 'Online Backup','Device Protection','Tech Support','Streaming TV',
                                 'Streaming Movies']])
df_dummies.head()
```

	Churn Label	Phone Service_No	Phone Service_Yes	Multiple Lines_No	Multiple Lines_No phone service	Multiple Lines_Yes	Internet Service_DSL	Internet Service_Fiber optic	Internet Service_No	Online Security_No	...
0	1	0	1	1	0	0	1	0	0	0	...
1	1	0	1	1	0	0	0	1	0	1	...
2	1	0	1	0	0	1	0	1	0	1	...
3	1	0	1	0	0	1	0	1	0	1	...
4	1	0	1	0	0	1	0	1	0	1	...

```
plt.figure(figsize=(9, 7))
sns.heatmap(df_dummies.corr(), annot=False, cmap='coolwarm')

plt.show()
```

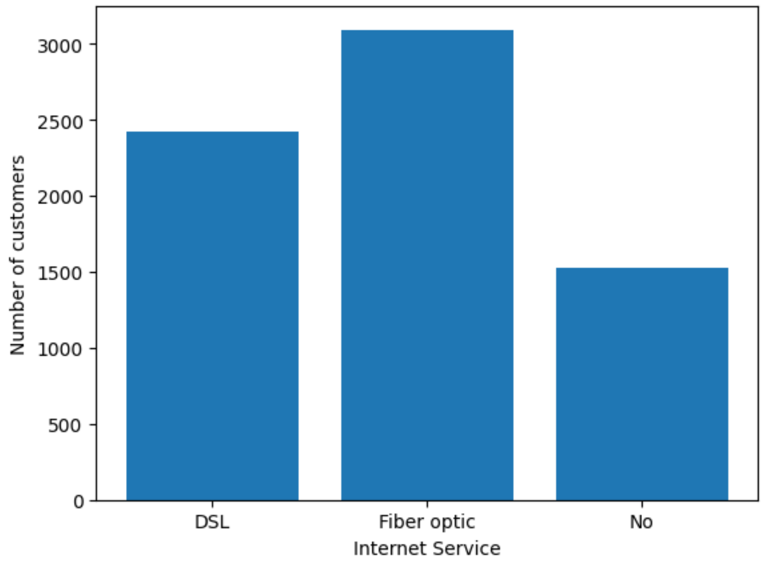


```
fig, ax = plt.subplots(figsize=(8, 6))
ax.bar(df_dummies.corr()['Churn Label'].sort_values(ascending=False).index,
       df_dummies.corr()['Churn Label'].sort_values(ascending=False).values)
ax.set_title('Correlation of Churn Label with other variables')
ax.set_xlabel('Variables')
ax.set_ylabel('Correlation')
plt.xticks(rotation=90)
plt.show()
```



```
internet_services = data.groupby('Internet Service')['CustomerID'].count().reset_index()
plt.bar(internet_services['Internet Service'], internet_services['CustomerID'])
plt.xlabel('Internet Service')
plt.ylabel('Number of customers')
```

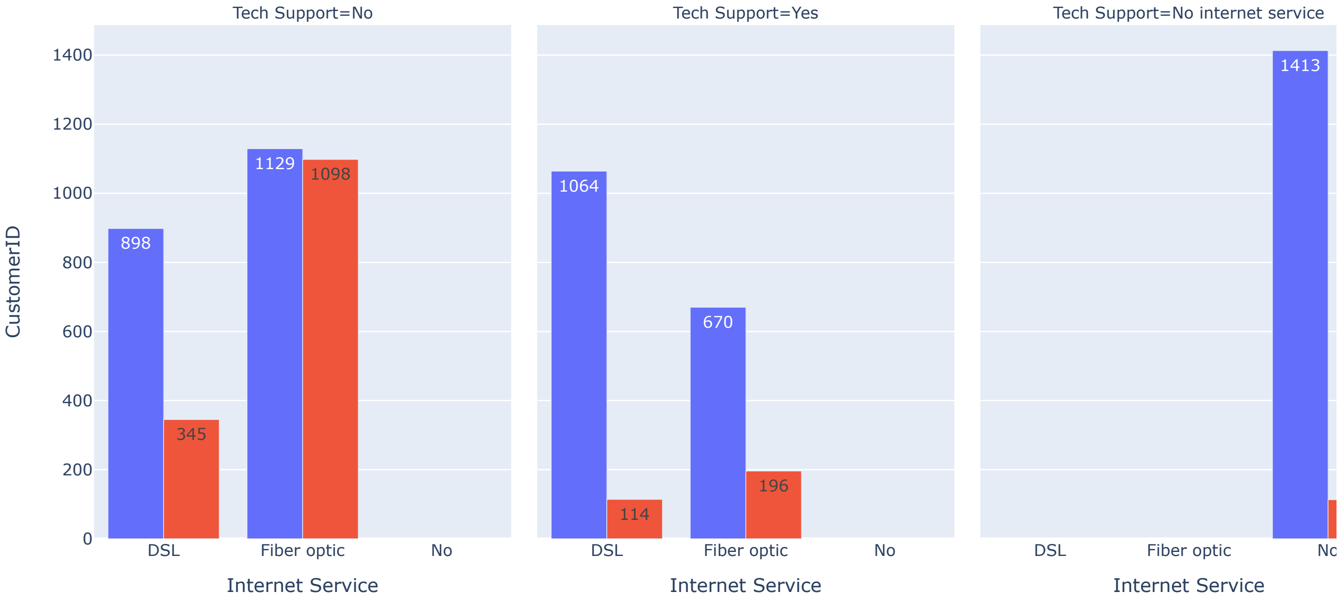
```
Text(0, 0.5, 'Number of customers')
```



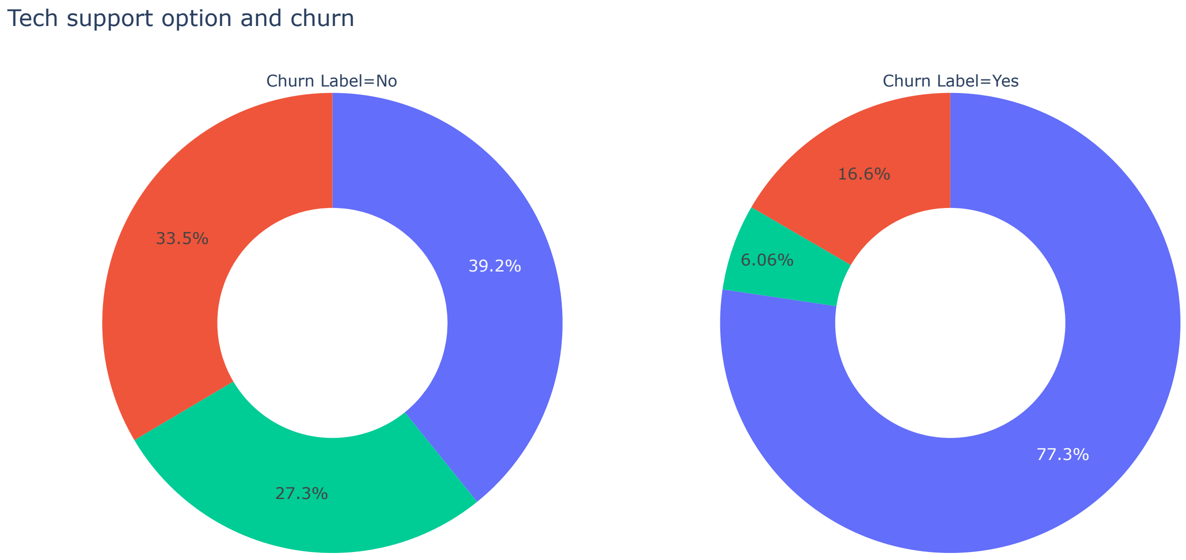
```
fig = px.pie(data.groupby(['Internet Service','Churn Label'])['CustomerID'].count().reset_index(),
             values='CustomerID',
             facet_col = 'Churn Label',
             names='Internet Service',
            title = 'What type of internet was connected to the clients who left the service?')
fig.show()
```

Tech Support

```
fig = px.bar(data.groupby(['Internet Service',
                                                'Tech Support',
                                                'Churn Label'])['CustomerID'].count().reset_index(),
             x="Internet Service",
             y="CustomerID",
             color="Churn Label",
             text = 'CustomerID',
             barmode="group",
             facet_col="Tech Support"
            )
fig.show()
```



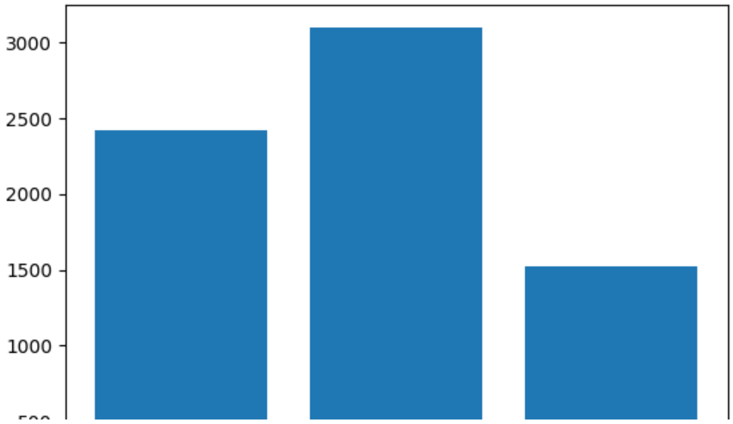
```
fig = px.pie(data.groupby(['Tech Support','Churn Label'])['CustomerID'].count().reset_index(),
             values='CustomerID',
             facet_col = 'Churn Label',
             hole = .5,
             names='Tech Support',
            title = 'Tech support option and churn')
fig.show()
```



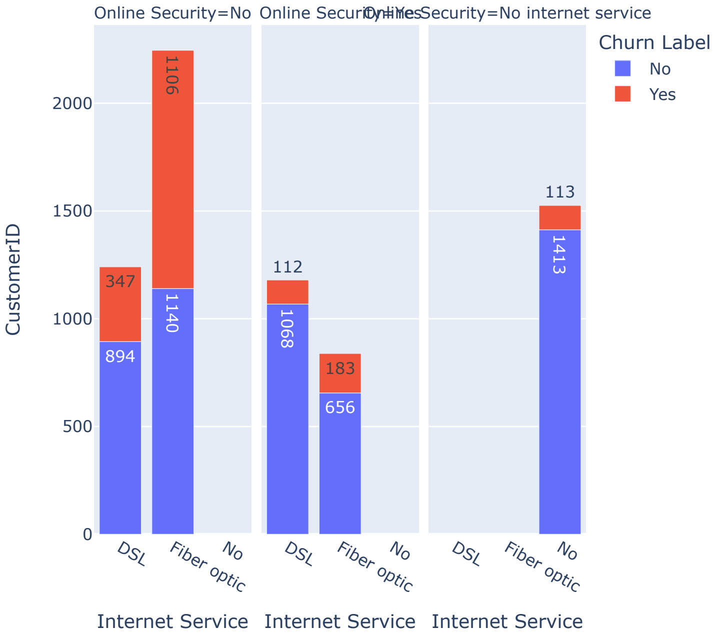
```
internet_services = data.groupby('Internet Service')['CustomerID'].count().reset_index()
```

```
plt.bar(internet_services['Internet Service'], internet_services['CustomerID'])
```

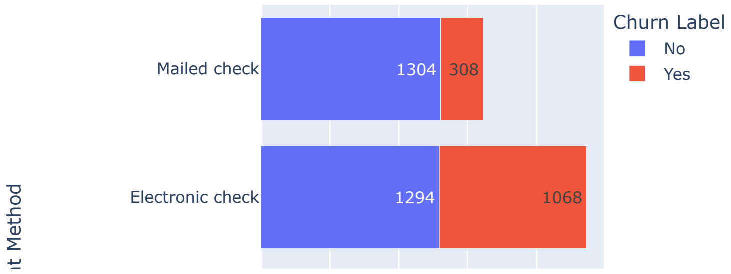
```
<BarContainer object of 3 artists>
```



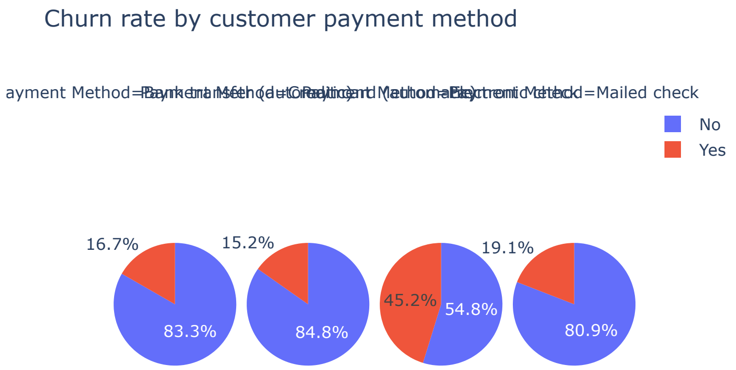
```
fig = px.bar(data.groupby(['Internet Service','Online Security',
                                                'Churn Label'])['CustomerID'].count().reset_index(),
             x="Internet Service",
             y="CustomerID",
             color="Churn Label",
             #barmode="group",
             text = 'CustomerID',
             facet_col = 'Online Security'
            )
fig.show()
```



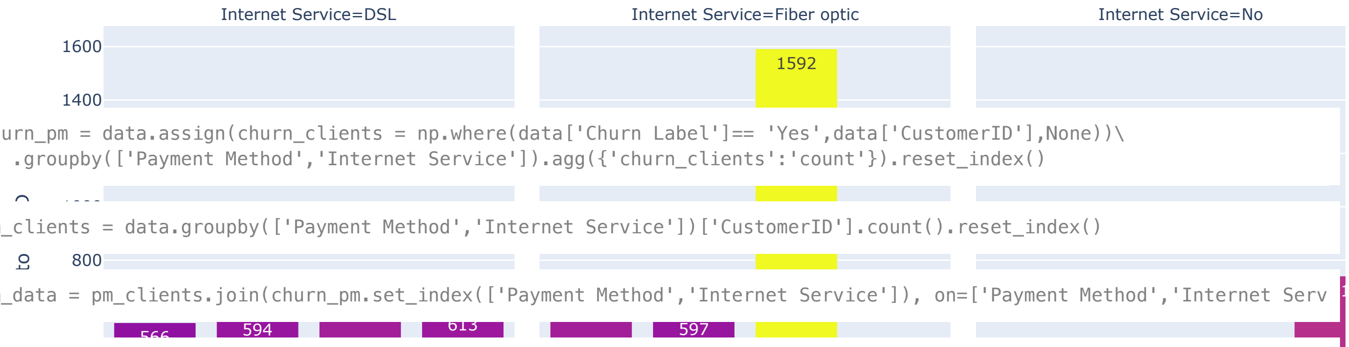
```
fig = px.bar(data.groupby(['Payment Method',
                                                'Churn Label'])['CustomerID'].count().reset_index(),
             x="CustomerID",
             y="Payment Method",
             color="Churn Label",
             text = 'CustomerID'
            )
fig.show()
```



```
fig = px.pie(data.groupby(['Payment Method','Churn Label'])['CustomerID'].count().reset_index(),
            values='CustomerID',
            names='Churn Label',
            facet_col = 'Payment Method',
            color = 'Churn Label',
            title = 'Churn rate by customer payment method')

fig.show()
```



```
fig = px.bar(data.groupby(['Payment Method','Internet Service'])['CustomerID'].count().reset_index(),
             x='Payment Method',
             y='CustomerID',
             facet_col = 'Internet Service',
             color = 'CustomerID',
             text = 'CustomerID')
fig.show()
```



```
churn_pm = data.assign(churn_clients = np.where(data['Churn Label']== 'Yes',data['CustomerID'],None))\
   .groupby(['Payment Method','Internet Service']).agg({'churn_clients':'count'}).reset_index()
```

```
pm_clients = data.groupby(['Payment Method','Internet Service'])['CustomerID'].count().reset_index()
```

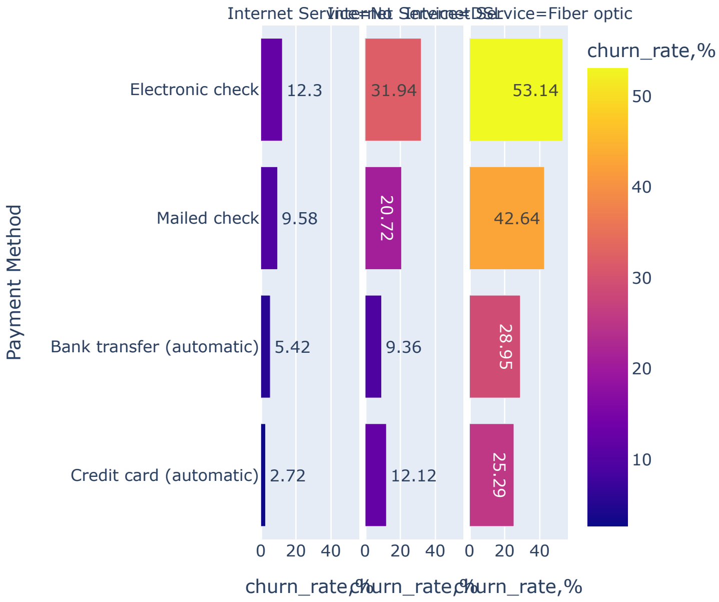
```
pm_data = pm_clients.join(churn_pm.set_index(['Payment Method','Internet Service']), on=['Payment Method','Internet Serv
```

```
pm_data
```

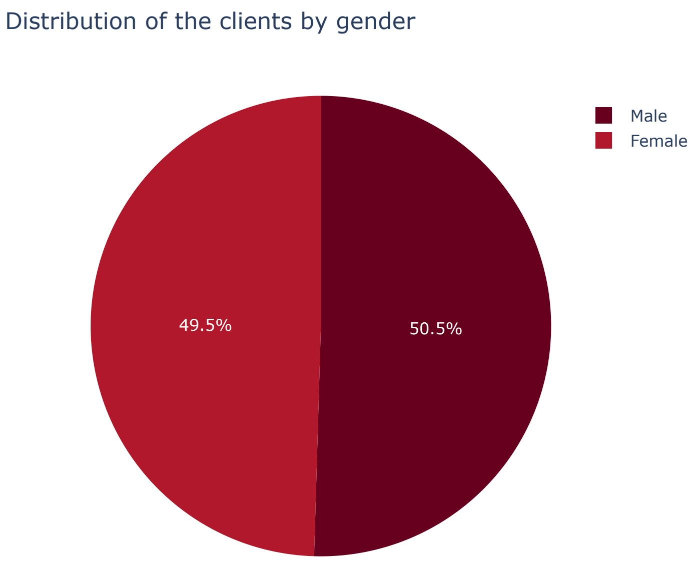
	Payment Method	Internet Service	CustomerID	churn_clients
0	Bank transfer (automatic)	DSL	566	53
1	Bank transfer (automatic)	Fiber optic	646	187
2	Bank transfer (automatic)	No	332	18
3	Credit card (automatic)	DSL	594	72
4	Credit card (automatic)	Fiber optic	597	151
5	Credit card (automatic)	No	331	9
6	Electronic check	DSL	648	207
7	Electronic check	Fiber optic	1592	846
8	Electronic check	No	122	15
9	Mailed check	DSL	613	127
10	Mailed check	Fiber optic	258	110
11	Mailed check	No	741	71

```
pm_data['churn_rate,%'] = round(((pm_data['churn_clients']/pm_data['CustomerID']) * 100),2)
```

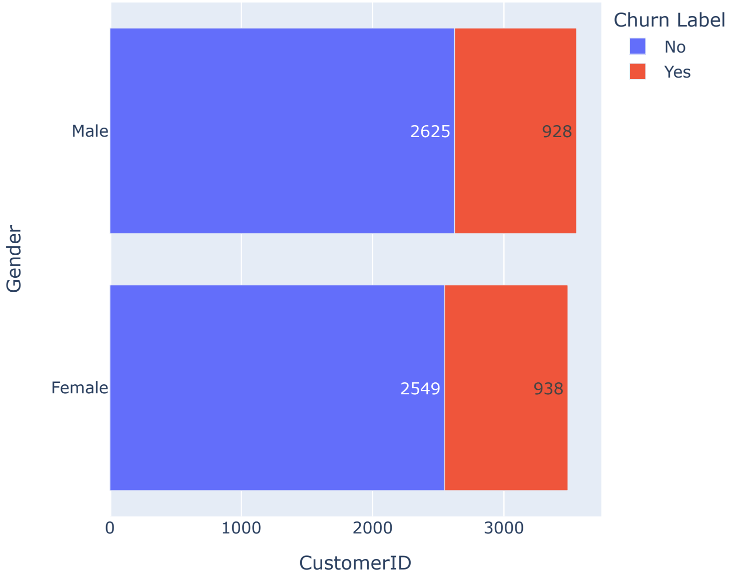
```
fig = px.bar(pm_data.sort_values('churn_rate,%'),
             x='churn_rate,%',
             y='Payment Method',
             facet_col = 'Internet Service',
             color = 'churn_rate,%',
             text = 'churn_rate,%')
fig.show()
```



```
fig = px.pie(data.groupby('Gender')['CustomerID'].count().reset_index(),
            values='CustomerID',
            names='Gender',
            color_discrete_sequence=px.colors.sequential.RdBu,
            title = 'Distribution of the clients by gender')

fig.show()
```

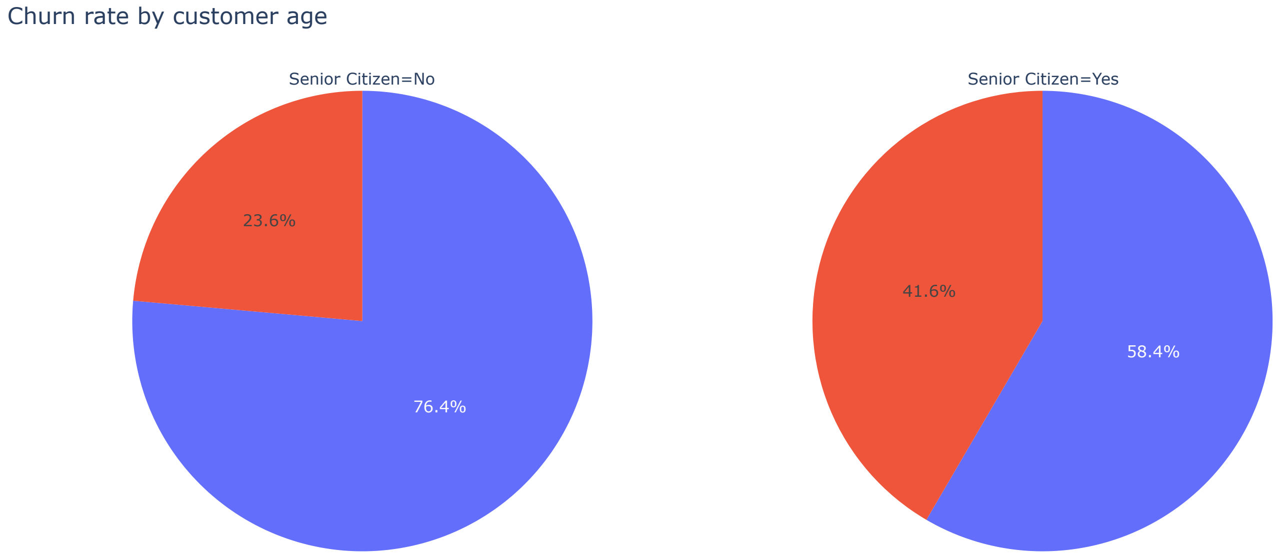


```
fig = px.bar(data.groupby(['Gender',
                                                 'Churn Label'])['CustomerID'].count().reset_index(),
             x="CustomerID",
             y="Gender",
             color="Churn Label",
             text = 'CustomerID'
            )
fig.show()
```



```
fig = px.pie(data.groupby(['Senior Citizen','Churn Label'])['CustomerID'].count().reset_index(),
            values='CustomerID',
            names='Churn Label',
            facet_col = 'Senior Citizen',
            color = 'Churn Label',
            title = 'Churn rate by customer age')

```

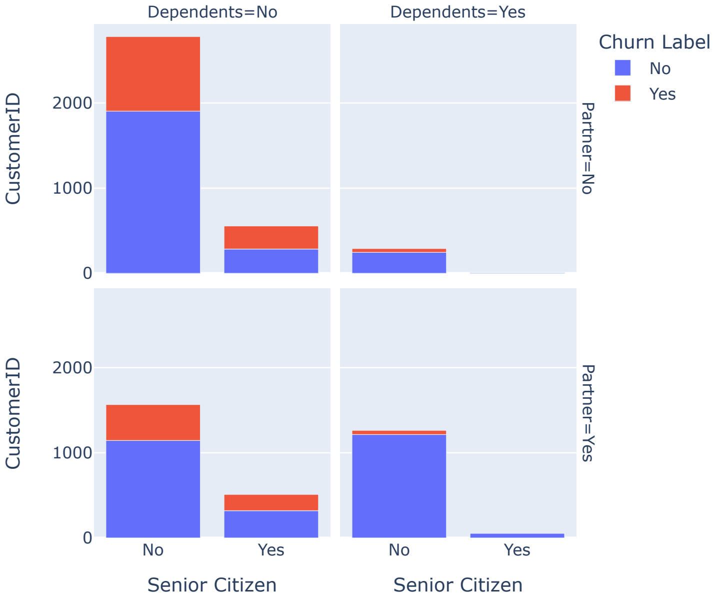
```
fig.show()

```



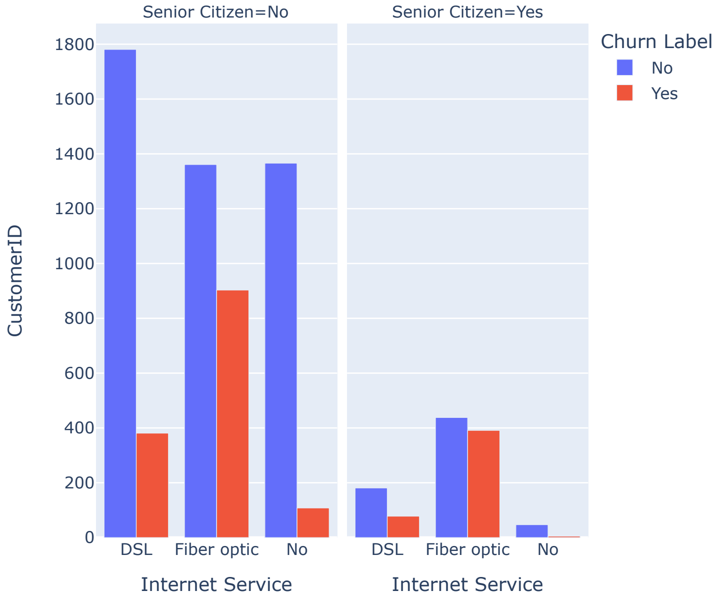
```
data.groupby('Senior Citizen')['CustomerID'].count()
```

```
Senior Citizen
No     5900
Yes    1140
Name: CustomerID, dtype: int64
```

```
fig = px.bar(data.groupby(['Senior Citizen','Partner',
                                        'Dependents','Churn Label'])['CustomerID'].count().reset_index(),
             x="Senior Citizen",
             y="CustomerID",
             color="Churn Label",
             #barmode="group",
             facet_row="Partner",
             facet_col = 'Dependents'
            )
fig.show()
```



```
fig = px.bar(data.groupby(['Senior Citizen','Internet Service','Churn Label'])['CustomerID'].count().reset_index(),
             x="Internet Service",
             y="CustomerID",
             color="Churn Label",
             barmode="group",
             facet_col = 'Senior Citizen'
            )
fig.show()
```



Double-click (or enter) to edit

Double-click (or enter) to edit