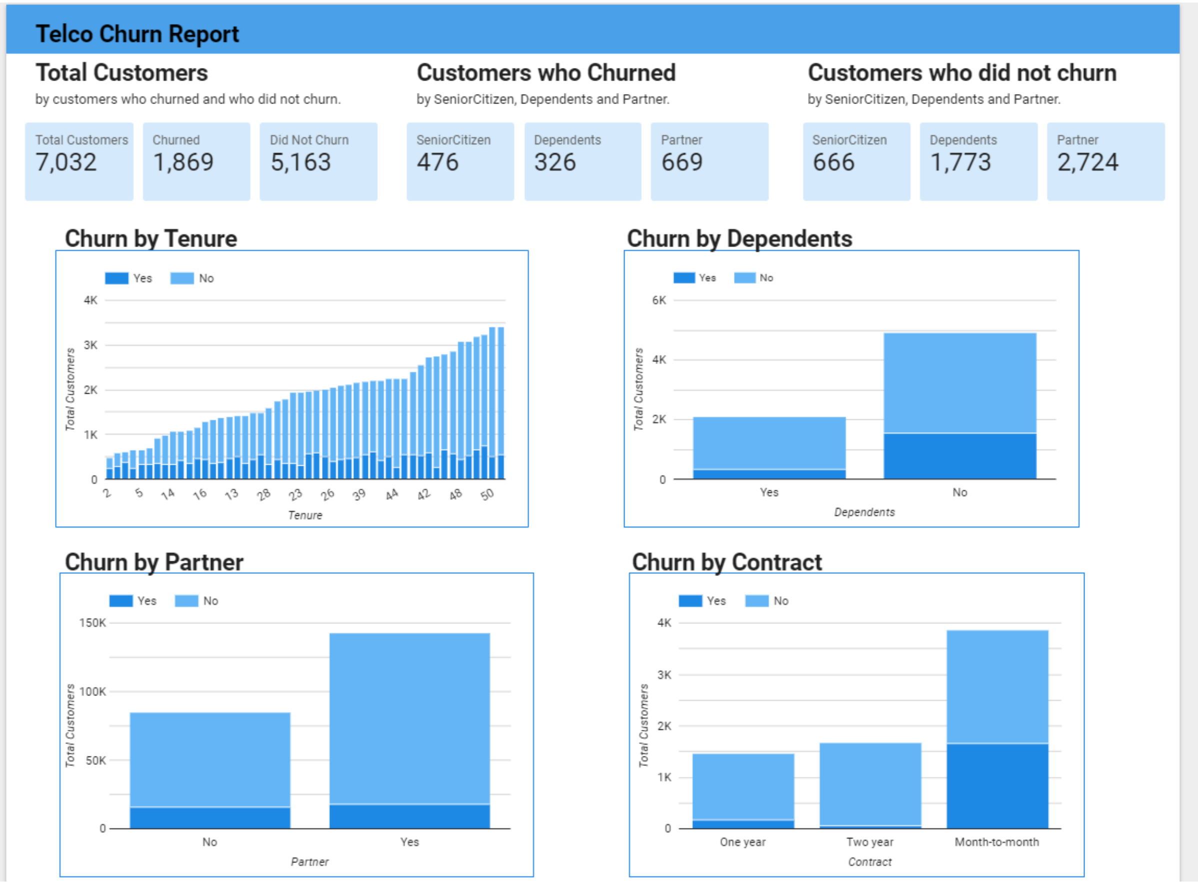
Name: Hurma Mahmood

ERPN: 14855

Telco Churn Data Assignment

Dashboard:

Link: https://datastudio.google.com/reporting/302afa40-b6a8-4744-a039-8cda74326624



Telco Churn Report

Total Customers

by customers who churned and who did not churn.

Total Customers	Churned	Did Not Churn
7,032	1,869	5,163

Customers who Churned

by SeniorCitizen, Dependents and Partner.

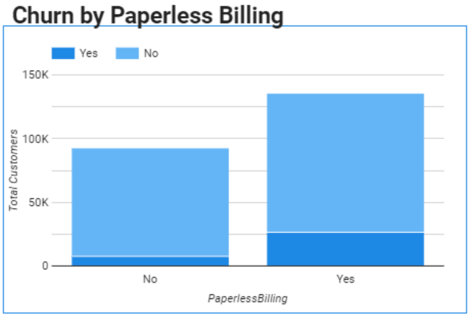
SeniorCitizen	Dependents	Partner
476	326	669

Customers who did not churn

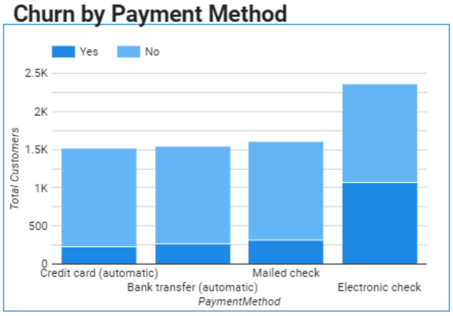
by SeniorCitizen, Dependents and Partner.

SeniorCitizen	Dependents	Partner
666	1,773	2,724

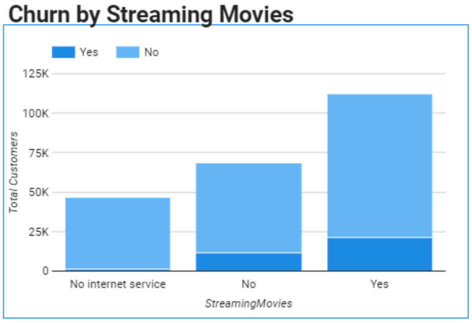
Churn by Paperless Billing



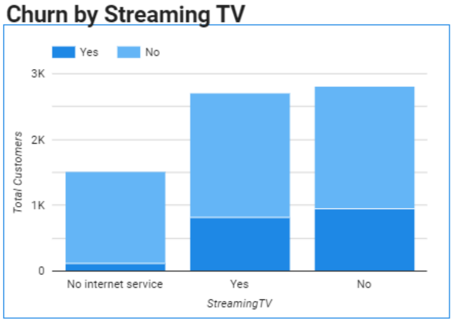
Churn by Payment Method



Churn by Streaming Movies



Churn by Streaming TV



Telco Churn Report

Total Customers

by customers who churned and who did not churn.

Total Customers	Churned	Did Not Churn
7,032	1,869	5,163

Customers who Churned

by SeniorCitizen, Dependents and Partner.

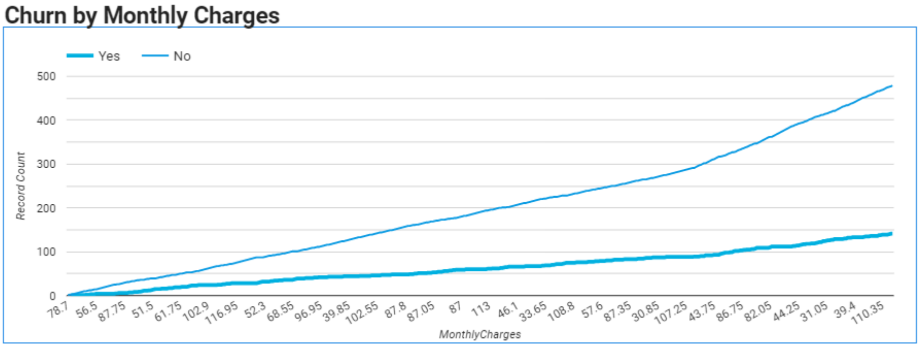
SeniorCitizen	Dependents	Partner
476	326	669

Customers who did not churn

by SeniorCitizen, Dependents and Partner.

SeniorCitizen	Dependents	Partner
666	1,773	2,724

Churn by Monthly Charges



Churn by Total Charges

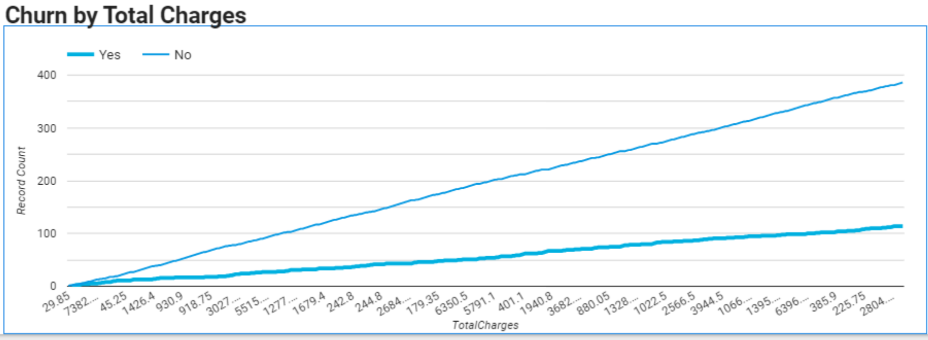


Chart Explanation:

1. Churn by Tenure: This graph shows that as the tenure of customers increases, churn decrease therefore this means that tenure and churn are indirectly proportional to each other. Therefore, customers that have been with the company for a long time are most likely to stay their loyal customers. The company should invest in a campaign that will target new customer and persuade these new customers to stay for a certain period of time by giving them good deals and various promotional offers. On the other hand, the company should also work towards rewarding their loyal customers so that their churn rate further decreases.
2. Churn by Dependents: As the graph shows, customer who do not have any dependents are most likely to churn, this may be because they may have time to look for better offers and are easily persuaded to join other companies. This indicates that the company should work on creating offers and deals that will persuade these customers to stay.
3. Churn by Partner: As the graph shows, customer who do not have any partner are most likely to churn. This indicates that the company should work on creating offers and deals that will persuade these customers to stay.
4. Churn by Contract: The chart shows that, customers who have 'month-to-month' contracts are most likely to churn out of the other contract options. The should focus on working on schemes that will persuade these customers on buying another contract. Other than this they should determine the root of customers leaving after month-to-month contract.
5. Churn by Paperless Billing: As the chart shows, more people prefer paperless billing however, there is also a significantly higher churn among these customers. The company should determine the cause of this problem and work on it to bring down their churn significantly.
6. Churn by Payment Method: As the pervious chart shows, more people prefer paperless billing, so they are paying through electronic check as this chart displays. Again, there is also a significantly higher churn among these customers. The company should work on determining this problem which will bring down their churn significantly.
7. Churn by Streaming Movies: This graph shows that customers who stream movies may churn may be due to other streaming services offer better variety of movies.
8. Churn by Streaming TV: This graph shows that people who do or do not watch TV have a high churn compared to other customers in this category. People who do watch TV and churn may be because other streaming services offer better variety of channels which the company should immediately work on and also determine the cause of the people who do not watch TV and still churn.
9. Churn by Monthly Charges: The graph displays that, when monthly charges are high, churn is also high which indicates that customers do not prefer high monthly charges.
10. Churn by Total Charges: The graph displays that, when total charges are low, churn is high which indicates that customers prefer high total charges when all services are included.

Appendix:

jupyter TelcoDataChurn_HurmaMahmood-14855 Last Checkpoint: a minute ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

In [1]:
```
#importing the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from scipy import stats
import seaborn as sns

import statsmodels.api as sm
from scipy import stats
from statsmodels.formula.api import ols
```

In [2]:
```
#import the data
churndf = pd.read_csv("churndata.csv")
```

In [3]:
```
#printing the top 5 row of the dataset
churndf.head(5)
```

Out[3]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity	...	DeviceProtection	Tecl
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL	No	...	No	
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL	Yes	...	Yes	
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL	Yes	...	No	
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL	Yes	...	Yes	
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic	No	...	No	

5 rows × 21 columns

In [4]:
```
#checking the total number of rows and columns
churndf.shape
```

Out[4]: (7043, 21)

```
churndf.dtypes
```

Out[5]:
```
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 [6]:
```
#checking null values
churndf.isnull().sum()
```

Out[6]:
```
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
```

```
OnlineBackup        0
DeviceProtection    0
TechSupport         0
StreamingTV         0
StreamingMovies     0
Contract            0
PaperlessBilling    0
PaymentMethod       0
MonthlyCharges      0
TotalCharges        0
Churn               0
dtype: int64
```

Replacing Incorrect Values

In [7]:
```
#Replacing inccrect values will null values also correcting the datatype of the columns
churndf.dtypes
```

Out[7]:
```
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 [8]:
```
#replacing incorrect Total Charges values with NaN and changing column datatype to float
churndf['TotalCharges'] = pd.to_numeric(churndf['TotalCharges'], downcast='float', errors='coerce')
```

In [9]:
```
churndf.dtypes
```

Out[9]:
```
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        float32
Churn                object
dtype: object
```

Dropping Unnecessary Columns/Data & Dealing with Missing/Null Values

In [10]:
```
#Checking missung/Null values in the columns
churndf.isnull().sum()
```

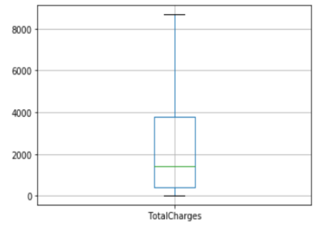
Out[10]:
```
customerID          0
gender              0
SeniorCitizen       0
Partner             0
Dependents          0
tenure              0
PhoneService        0
```

```
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 [11]:

```
#to check any outliers
churndf.boxplot(column='TotalCharges', sym='o', return_type='axes')
```

Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x1fab98b9b88>



With the above box plot it can be seen that the data is not normally distributed but it is skewed also, that there are no outliers in the data.

In [12]:

```
churndf['TotalCharges'].value_counts()
```

Out[12]:

```
20.200001      11
19.750000       9
20.049999       8
19.900000       8
19.650000       8
               ..
1451.599976     1
1173.349976     1
5589.450195     1
3810.550049     1
1024.000000     1
Name: TotalCharges, Length: 6530, dtype: int64
```

In [13]:

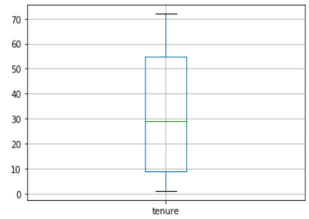
```
#Since this columns has too maany different values and the number of missing/null values only makeup to less than 0.5% of
#the data hence the best option is to remove the rows containing these null values.
churndf = churndf.drop(churndf[churndf['TotalCharges'].isnull()].index)
```

Now lets check for outliers in all the numerical data columns by using box plot

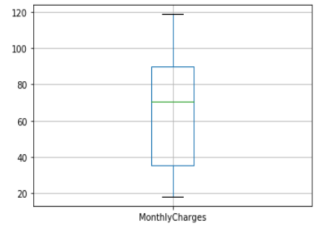
In [14]:

```
#to check any outliers
churndf.boxplot(column='tenure', sym='o', return_type='axes')
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x1faba042b88>



In [15]:
```
churndf.boxplot(column='MonthlyCharges', sym='o', return_type='axes')
```
Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x1faba0af348>



As the above box plots shows there are no outliers in the numerical data

In [16]:
```
#Along with this the column 'customerID' contains the ID which will not be usefull in our analysis hence dropping this column
churndf = churndf.drop('customerID', axis=1)
```

t-test

In [17]:
```
types_map = churndf.dtypes.to_dict()
num_columns = []
for k,v in types_map.items():
    if np.issubdtype(np.int64, v) or np.issubdtype(np.float64, v) or np.issubdtype(np.float32, v):
        num_columns.append(k)

print(num_columns)

for i in range(len(num_columns)-1):
    for j in range(i+1,len(num_columns)):
        col1 = num_columns[i]
```

```
['SeniorCitizen', 'tenure', 'MonthlyCharges', 'TotalCharges']
(SeniorCitizen,tenure) => t-value=-110.19930682001105, p-value=0.0
(SeniorCitizen,MonthlyCharges) => t-value=-180.14238265109182, p-value=0.0
(SeniorCitizen,TotalCharges) => t-value=-84.46249234456064, p-value=0.0
(tenure,MonthlyCharges) => t-value=-69.92300630034109, p-value=0.0
(tenure,TotalCharges) => t-value=-83.26420730219256, p-value=0.0
(MonthlyCharges,TotalCharges) => t-value=-82.06412611168557, p-value=0.0
```

With the above t-test we can see that all the numerical columns have p-values < 0.05, which means that the alternate hypothesis is true. Therefore it can be said that there is a statistically significant difference between them.

anova

In [18]:
```
#anova
model = ols('tenure ~ C(Q("Churn"))', data=churndf).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => tenure - Churn")
display(anova_table)

model = ols('SeniorCitizen ~ C(Q("Churn"))', data=churndf).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => Senior Citizen - Churn")
display(anova_table)

model = ols('MonthlyCharges ~ C(Q("Churn"))', data=churndf).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => Monthly Charges - Churn")
display(anova_table)

model = ols('TotalCharges ~ C(Q("Churn"))', data=churndf).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => Total Charges - Churn")
display(anova_table)
```

Anova => tenure - Churn

	sum_sq	df	F	PR(>F)

```
Anova => tenure - Churn
```

	sum_sq	df	F	PR(>F)
C(Q("Churn"))	5.309822e+05	1.0	1007.509431	9.437650e-207
Residual	3.704983e+06	7030.0	NaN	NaN

```
Anova => Senior Citizen - Churn
```

	sum_sq	df	F	PR(>F)
C(Q("Churn"))	21.677662	1.0	163.012426	6.377295e-37
Residual	934.861018	7030.0	NaN	NaN

```
Anova => Monthly Charges - Churn
```

	sum_sq	df	F	PR(>F)
C(Q("Churn"))	2.367127e+05	1.0	271.57699	6.760843e-60
Residual	6.127508e+06	7030.0	NaN	NaN

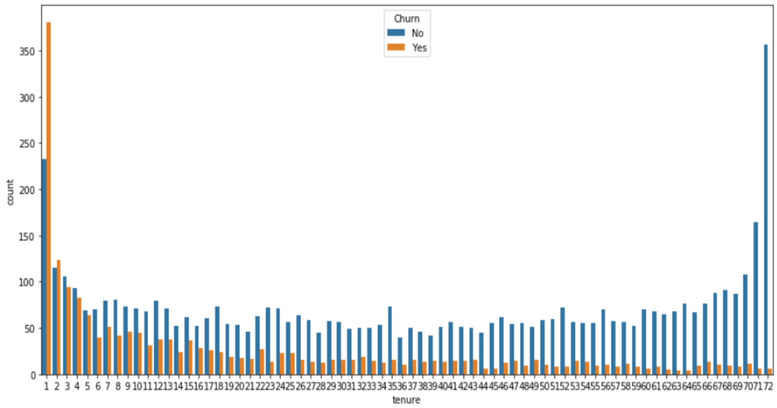
```
Anova => Total Charges - Churn
```

	sum_sq	df	F	PR(>F)
C(Q("Churn"))	1.437636e+09	1.0	291.344864	4.876862e-64
Residual	3.468942e+10	7030.0	NaN	NaN

As the above results shows, there is a significant difference between all above mentioned columns.

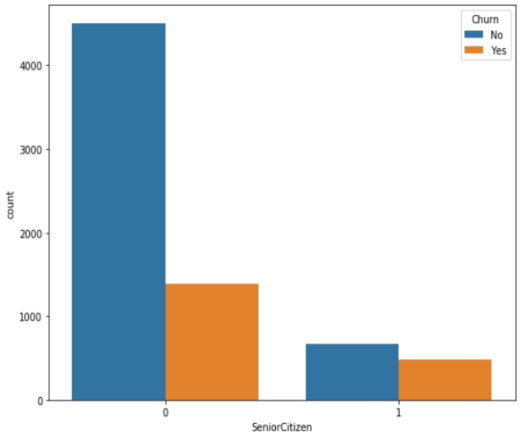
Data Exploration:

In [19]:
```python
#plotting bar charts for better understanding the relation b/w numerical columns and churn
a4_dims = (15, 7)
fig, ax = plt.subplots(figsize=a4_dims)
Tenure = sns.countplot(ax=ax, data=churndf, x=churndf.tenure, hue='Churn')
```



The above graphs shows that as the tunre increases, there is a significant drop in the churn rate.

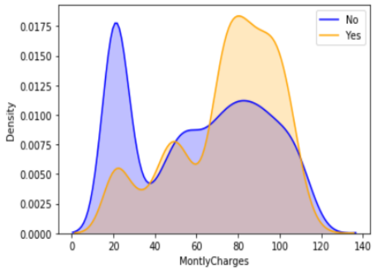
In [20]:
```python
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
SeniorCitizen = sns.countplot(ax=ax, data=churndf, x=churndf.SeniorCitizen, hue='Churn')
```



As the above chart shows, people who are not senior citizens have a low churn rate. While if the customer is a senior citzen, there is a approximately 60-40 ratio that the customer will not churn.

In [21]:
```python
#Making the monthly and total charges with kde since there are too many values which couldnot be displayed by countplot
MonthlyCharges = sns.kdeplot(churndf.MonthlyCharges[(churndf["Churn"] == "No") ], color="Blue", shade = True, label="No")
MonthlyCharges = sns.kdeplot(churndf.MonthlyCharges[(churndf["Churn"] == "Yes") ], color="Orange", shade= True, label="Yes")
MonthlyCharges.set_ylabel('Density')
MonthlyCharges.set_xlabel('MontlyCharges')
```

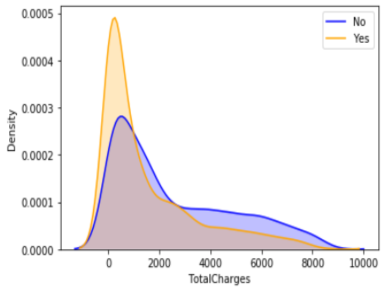
Out[21]: Text(0.5, 0, 'MontlyCharges')



The above graph displays that, when monthly charges are high, churn is also high which indicates that customers do not prefer high monthly charges.

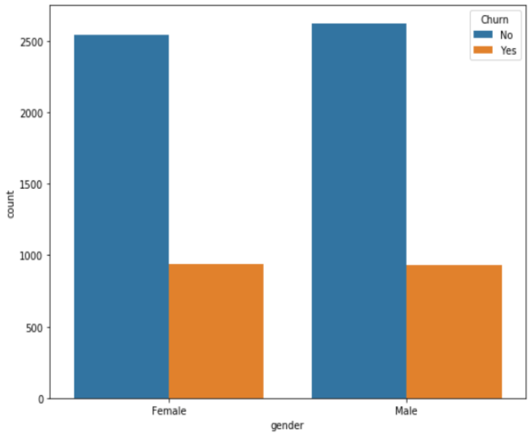
In [22]:
```python
TotlaCharges = sns.kdeplot(churndf.TotalCharges[(churndf["Churn"] == "No") ], color="Blue", shade = True, label="No")
TotalCharges = sns.kdeplot(churndf.TotalCharges[(churndf["Churn"] == "Yes") ], color="Orange", shade= True, label="Yes")
TotalCharges.set_ylabel('Density')
TotalCharges.set_xlabel('TotalCharges')
```

Out[22]: Text(0.5, 0, 'TotalCharges')



The above graph displays that, when total charges are low, churn is high which indicates that customers prefer high total charges when all services are included.

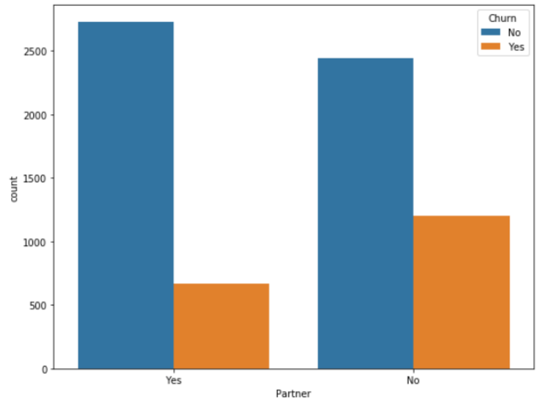
In [23]:
```python
#plotting bar charts for better understanding the relation b/w categorical columns and churn
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
Gender = sns.countplot(ax=ax, data=churndf, x=churndf.gender, hue='Churn')
```



As the above chart shows, there is no significant affect on churn by Gender.

In [24]:

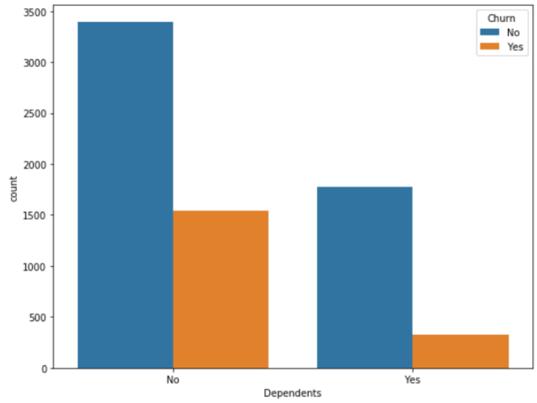
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
Partner = sns.countplot(ax=ax, data=churndf, x=churndf.Partner, hue='Churn')
```



As the above chart shows, people who don't have a partner are likely to churn more than people who do.

In [25]:

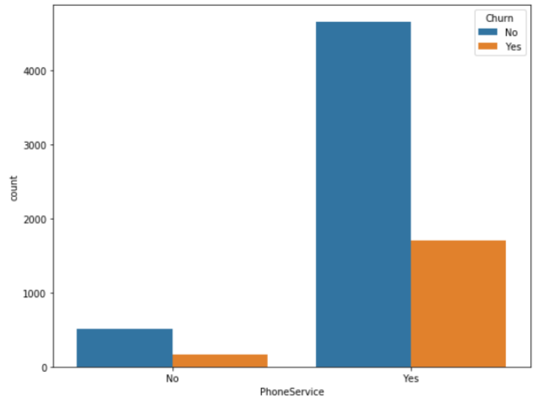
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
Dependents = sns.countplot(ax=ax, data=churndf, x=churndf.Dependents, hue='Churn')
```



As the above chart shows, people who don't have dependents are likely to churn more than people who do.

In [26]:

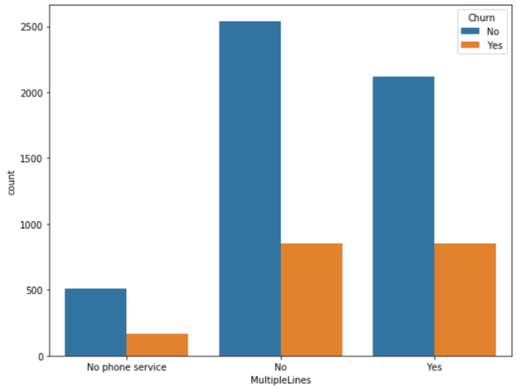
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
PhoneService = sns.countplot(ax=ax, data=churndf, x=churndf.PhoneService, hue='Churn')
```



As the above chart shows, there is no significant affect on churn by PhoneService.

In [27]:

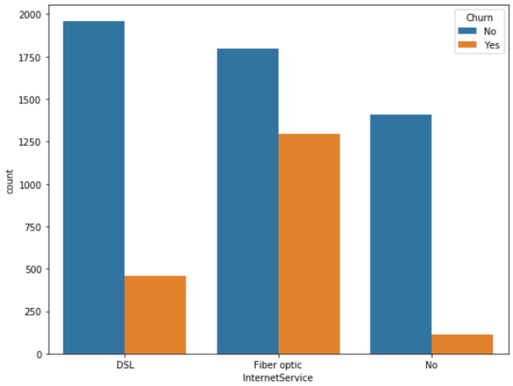
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
MultipleLines = sns.countplot(ax=ax, data=churndf, x=churndf.MultipleLines, hue='Churn')
```



As the above chart shows, there is no significant affect on churn by MultipleLines.

In [28]:

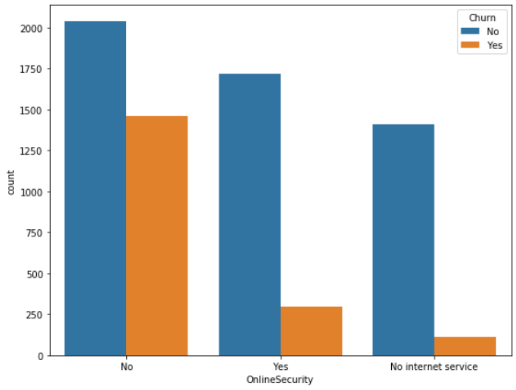
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
InternetService = sns.countplot(ax=ax, data=churndf, x=churndf.InternetService, hue='Churn')
```



As the above chart shows, People with no internet service are very much less likely to churn, whereas, people with Fiber optic are more likely to churn as compared to DSL.

In [29]:

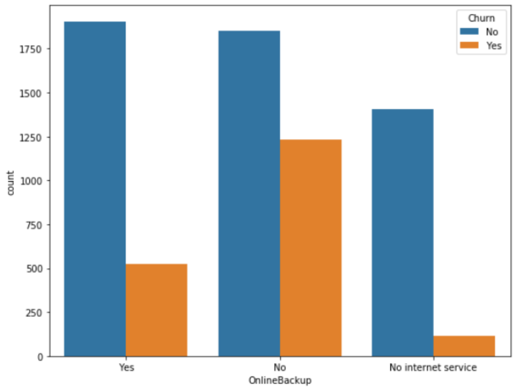
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
OnlineSecurity = sns.countplot(ax=ax, data=churndf, x=churndf.OnlineSecurity, hue='Churn')
```



There doesn't seem to be have much affect on churning.

In [30]:

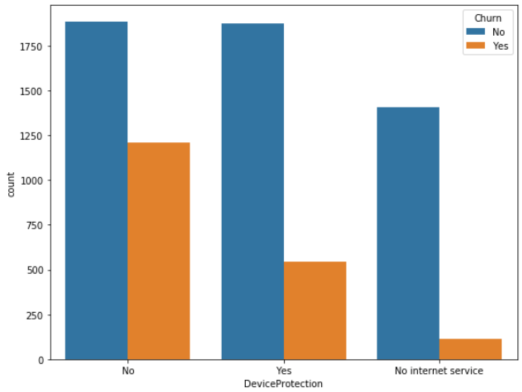
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
OnlineBackup = sns.countplot(ax=ax, data=churndf, x=churndf.OnlineBackup, hue='Churn')
```



There doesn't seem to be have much affect on churning.

In [31]:

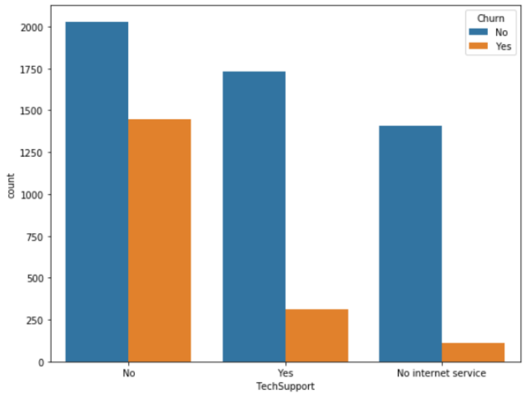
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
DeviceProtection = sns.countplot(ax=ax, data=churndf, x=churndf.DeviceProtection, hue='Churn')
```



There doesn't seem to be have much affect on churning.

In [32]:

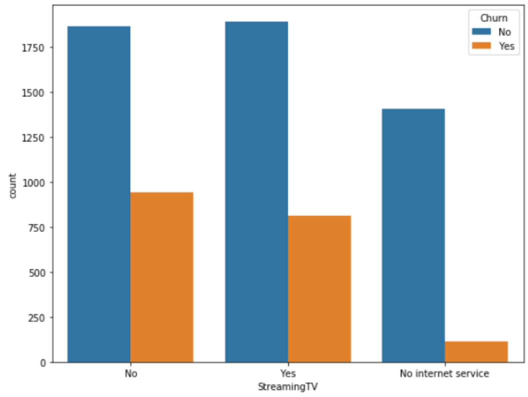
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
TechSupport = sns.countplot(ax=ax, data=churndf, x=churndf.TechSupport, hue='Churn')
```



There doesn't seem to be have much affect on churning.

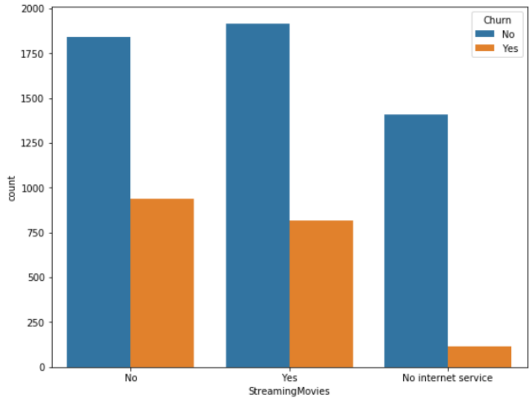
In [33]:

```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
StreamingTV = sns.countplot(ax=ax, data=churndf, x=churndf.StreamingTV, hue='Churn')
```



In [34]:

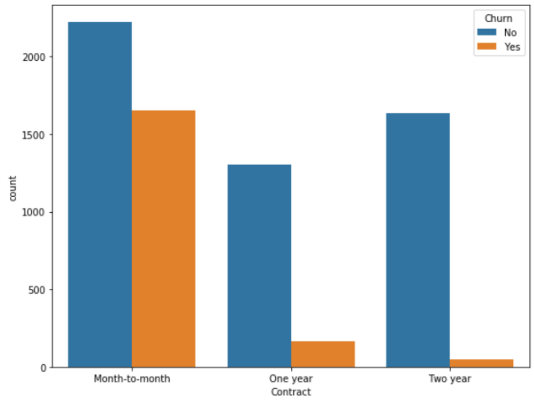
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
StreamingMovies = sns.countplot(ax=ax, data=churndf, x=churndf.StreamingMovies, hue='Churn')
```




```
Churn rate is same for StreamingTV and StreamingMovies
```

In [35]:

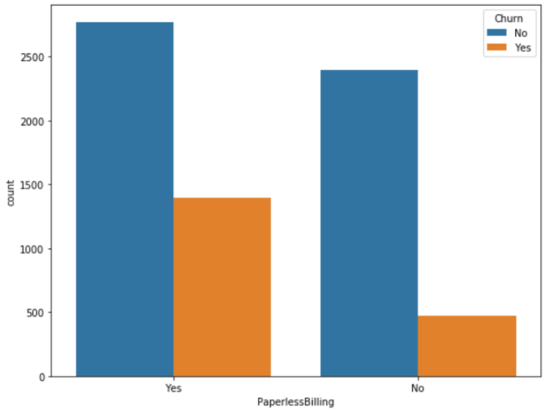
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
Contract = sns.countplot(ax=ax, data=churndf, x=churndf.Contract, hue='Churn')
```




```
As the above figure shows, people with month-to-month contracts are way more likely to churn as these people can be easily
persuaded to go to other companies.
```

In [36]:

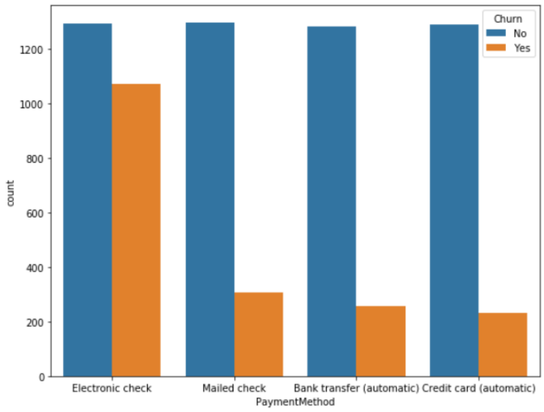
```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
PaperlessBilling = sns.countplot(ax=ax, data=churndf, x=churndf.PaperlessBilling, hue='Churn')
```



As the above figure shows, people with paperlessBilling tend to churn slightly higher than people who don't have paperless Billing.

In [37]:

```
a4_dims = (9, 7)
fig, ax = plt.subplots(figsize=a4_dims)
PaymentMethod = sns.countplot(ax=ax, data=churndf, x=churndf.PaymentMethod, hue='Churn')
```



As the above chart shows, people with Electronic check as thier payment method are the one the are most likely to churn, this makes sense as the pervious chart confirms this.

In [ ]:

```
churndf.to_csv("TelcoChurnData.csv")
```