D207 Project Using the Cleaned Churn Data Set

In [1]:
```
import pandas as pd
df = pd.read_csv('churn_clean.csv')
```

In [2]:
```
df.head()
```

Out[2]:

	CaseOrder	Customer_id	Interaction	UID	City	State	C
0	1	K409198	aa90260b-4141-4a24-8e36-b04ce1f4f77b	e885b299883d4f9fb18e39c75155d990	Point Baker	AK	Pri \
1	2	S120509	fb76459f-c047-4a9d-8af9-e0f7d4ac2524	f2de8bef964785f41a2959829830fb8a	West Branch	MI	Og
2	3	K191035	344d114c-3736-4be5-98f7-c72c281e2d35	f1784cfa9f6d92ae816197eb175d3c71	Yamhill	OR	Y
3	4	D90850	abfa2b40-2d43-4994-b15a-989b8c79e311	dc8a365077241bb5cd5ccd305136b05e	Del Mar	CA	
4	5	K662701	68a861fd-0d20-4e51-a587-8a90407ee574	aabb64a116e83fdc4befc1fbab1663f9	Needville	TX	

5 rows × 50 columns

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 50 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   CaseOrder             10000 non-null  int64  
 1   Customer_id           10000 non-null  object 
 2   Interaction           10000 non-null  object 
 3   UID                   10000 non-null  object 
 4   City                  10000 non-null  object 
 5   State                 10000 non-null  object 
 6   County                10000 non-null  object 
 7   Zip                   10000 non-null  int64  
 8   Lat                   10000 non-null  float64
 9   Lng                   10000 non-null  float64
 10  Population            10000 non-null  int64  
 11  Area                  10000 non-null  object 
 12  TimeZone              10000 non-null  object 
 13  Job                   10000 non-null  object 
 14  Children              10000 non-null  int64  
 15  Age                   10000 non-null  int64  
 16  Income                10000 non-null  float64
 17  Marital               10000 non-null  object 
 18  Gender                10000 non-null  object 
 19  Churn                 10000 non-null  object 
 20  Outage_sec_perweek    10000 non-null  float64
 21  Email                 10000 non-null  int64  
 22  Contacts              10000 non-null  int64  
 23  Yearly_equip_failure  10000 non-null  int64  
 24  Techie                10000 non-null  object 
 25  Contract              10000 non-null  object 
 26  Port_modem            10000 non-null  object 
 27  Tablet                10000 non-null  object 
 28  InternetService       10000 non-null  object 
 29  Phone                 10000 non-null  object 
 30  Multiple              10000 non-null  object 
 31  OnlineSecurity        10000 non-null  object 
 32  OnlineBackup          10000 non-null  object 
 33  DeviceProtection      10000 non-null  object 
 34  TechSupport           10000 non-null  object 
 35  StreamingTV           10000 non-null  object 
 36  StreamingMovies       10000 non-null  object 
 37  PaperlessBilling      10000 non-null  object 
 38  PaymentMethod         10000 non-null  object 
 39  Tenure                10000 non-null  float64
 40  MonthlyCharge         10000 non-null  float64
 41  Bandwidth_GB_Year     10000 non-null  float64
 42  Item1                 10000 non-null  int64  
 43  Item2                 10000 non-null  int64  
 44  Item3                 10000 non-null  int64  
 45  Item4                 10000 non-null  int64  
 46  Item5                 10000 non-null  int64  
 47  Item6                 10000 non-null  int64  
 48  Item7                 10000 non-null  int64  
 49  Item8                 10000 non-null  int64  
```

```
dtypes: float64(7), int64(16), object(27)
memory usage: 3.8+ MB
```

Section A

1) Is there a correlation between customers who see themselves as Techie and the churn rate of the customer?

2) Stakeholders will value this information because they can see how people who view themselves in a certain way behave as customers.

3) In order to ansewr my question from part A1 I am going to use the Churn and Techie column. Both of these columns have an objecct data type where Churn is either yes or no and Techie which is also either yes or no.

Section B

1) Chi-Square: My hypothesis is that being a techie and churn rate are independent of each other. Even if you view yourself as a techie you still need an internet provider.

In [4]:
```
contingency = (df['Churn'], df['Techie'])
contingency
```

Out[4]:
```
(0        No
 1       Yes
 2        No
 3        No
 4       Yes
        ...
 9995     No
 9996     No
 9997     No
 9998     No
 9999     No
 Name: Churn, Length: 10000, dtype: object,
 0        No
 1       Yes
 2       Yes
 3       Yes
 4        No
        ...
 9995     No
 9996     No
 9997     No
 9998     No
 9999     No
 Name: Techie, Length: 10000, dtype: object)
```

In [5]:
```
contingency_pct = pd.crosstab(df['Churn'], df['Techie'], normalize ='index')
contingency_pct
```

Out[5]:

Techie	No	Yes
Churn		
No	0.847075	0.152925
Yes	0.790566	0.209434

In [6]:
```
from scipy.stats import chi2_contingency
```

In [7]:
```
c, p, dof, expected = chi2_contingency(contingency_pct)
p
```

Out[7]:
```
0.08325508175692804
```

2) The results show that p-value is 8.32% which is not enough to reject our null hypothesis that the Churn and Techie columns are independent of each other.

3) I decided to run a chi-square test because I was looking at two categorical data types.

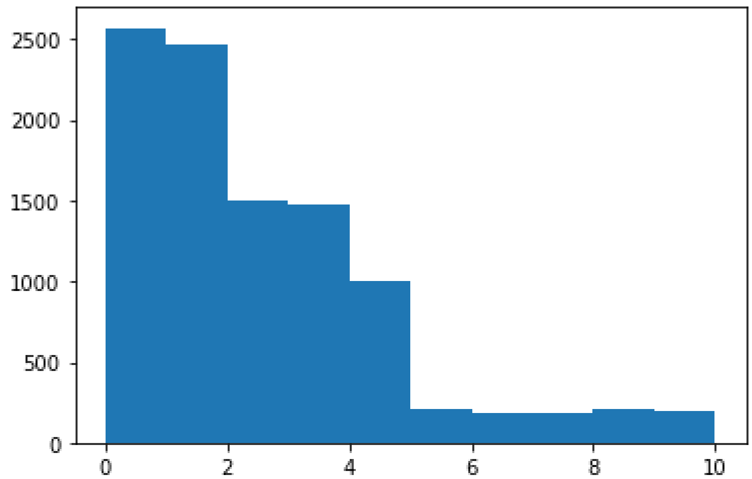
Section C

The columns we will be using for Parts C and D are: Children, Age, Monthly Charge, Outage_sec_perweek, Contacts, Churn and Techie.

In [8]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [9]:
```
plt.hist(df['Children'])
```

Out[9]:
```
(array([2570., 2472., 1495., 1472., 1006.,  212.,  187.,  185.,  210.,
         191.]),
 array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.]),
 <BarContainer object of 10 artists>)
```



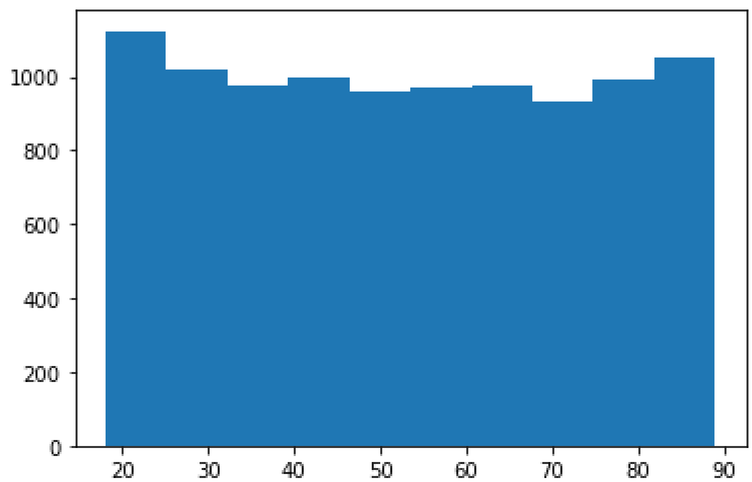
In [10]:
```
df['Children'].describe()
```

Out[10]:
```
count    10000.0000
mean         2.0877
std          2.1472
min          0.0000
25%          0.0000
50%          1.0000
75%          3.0000
max         10.0000
Name: Children, dtype: float64
```

The Children column is right-skewed with most of the data points coming on the left side of the histogram.

In [11]:
```
plt.hist(df['Age'])
```

Out[11]:
```
(array([1124., 1019.,  976.,  997.,  961.,  971.,  974.,  935.,  992.,
        1051.]),
 array([18. , 25.1, 32.2, 39.3, 46.4, 53.5, 60.6, 67.7, 74.8, 81.9, 89. ]),
 <BarContainer object of 10 artists>)
```



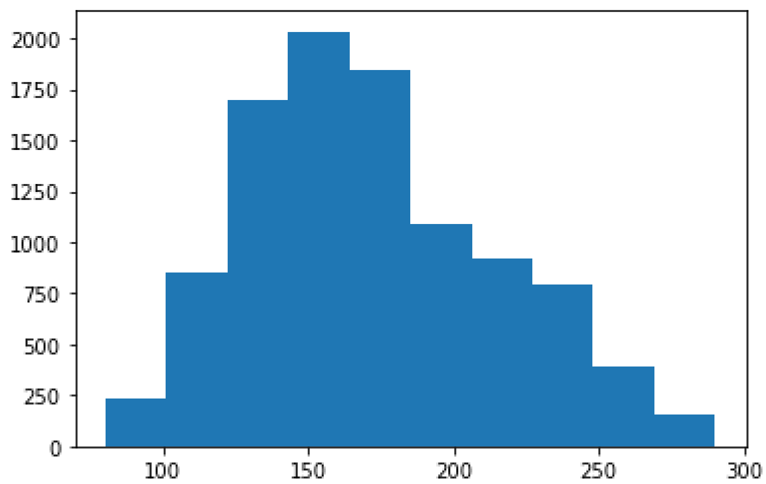
In [12]:
```
df['Age'].describe()
```

Out[12]:
```
count    10000.000000
mean        53.078400
std         20.698882
min         18.000000
25%         35.000000
50%         53.000000
75%         71.000000
max         89.000000
Name: Age, dtype: float64
```

The Age column appears to have a normal distribution with data points showing up evenly on both sides of the average.

In [13]:
```
plt.hist(df['MonthlyCharge'])
```

Out[13]:
```
(array([ 230.,  853., 1695., 2034., 1842., 1089.,  926.,  791.,  388.,
         152.]),
 array([ 79.97886  , 100.9970159, 122.0151718, 143.0333277, 164.0514836,
        185.0696395, 206.0877954, 227.1059513, 248.1241072, 269.1422631,
        290.160419 ]),
 <BarContainer object of 10 artists>)
```



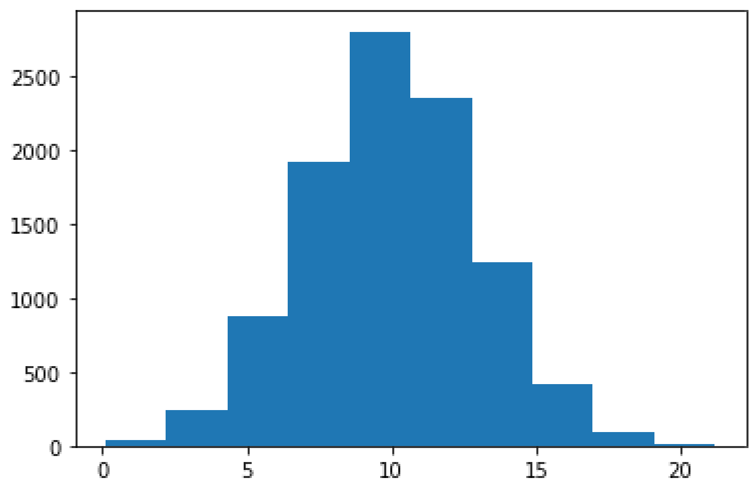
In [14]:
```
df['MonthlyCharge'].describe()
```

Out[14]:
```
count    10000.000000
mean       172.624816
std         42.943094
min         79.978860
25%        139.979239
50%        167.484700
75%        200.734725
max        290.160419
Name: MonthlyCharge, dtype: float64
```

The Monthly Charge column is left-skewed with most data points appearing on the right side of the average.

In [15]:
```
plt.hist(df['Outage_sec_perweek'])
```

Out[15]:
```
(array([  42.,  245.,  876., 1919., 2804., 2357., 1240.,  413.,   94.,
          10.]),
 array([ 0.09974694,  2.21049525,  4.32124355,  6.43199186,  8.54274016,
        10.65348847, 12.76423678, 14.87498508, 16.98573339, 19.09648169,
        21.20723   ]),
 <BarContainer object of 10 artists>)
```

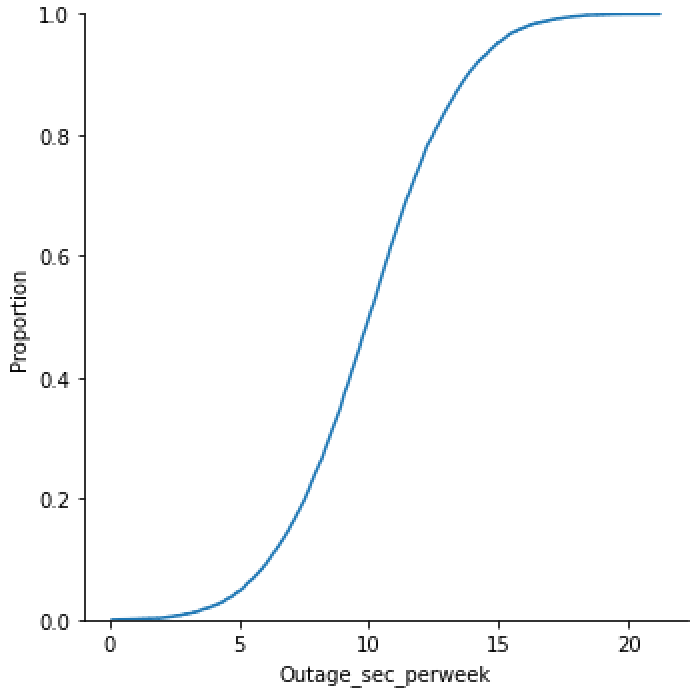


In [16]:
```
df['Outage_sec_perweek'].describe()
```

Out[16]:
```
count    10000.000000
mean        10.001848
std          2.976019
min          0.099747
25%          8.018214
50%         10.018560
75%         11.969485
max         21.207230
Name: Outage_sec_perweek, dtype: float64
```

In [17]:
```
sns.displot(df, x="Outage_sec_perweek",  kind="ecdf")
```

Out[17]: <seaborn.axisgrid.FacetGrid at 0x1efd77aaf70>



This displot certifies the conclusion made about the histogram above. As you can see by the curve most of the data lays between just before 5 up to about 15.

The Outage_sec_perweek column has a normal distribution with an even number of data points showing up on either side of the average.

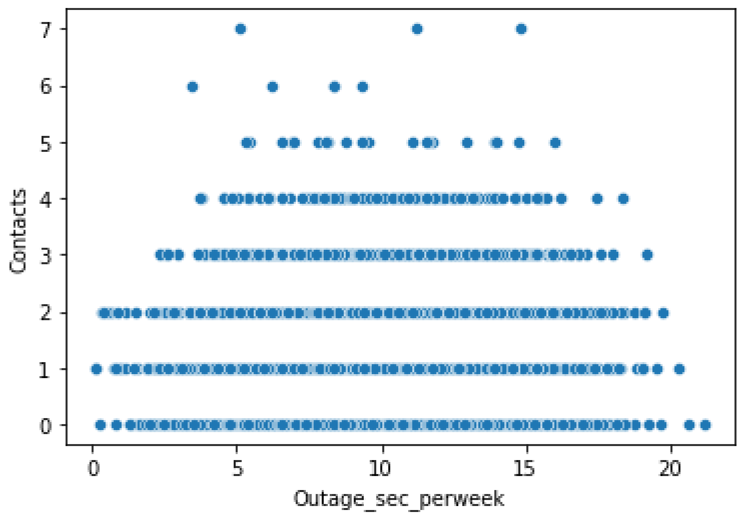
Section D

In [18]:
```
import numpy as np
import seaborn as sns
```

In [19]:
```
df_heatmap = df[['Contacts','Outage_sec_perweek']].copy()
```

In [20]:
```
sns.scatterplot(data=df, x='Outage_sec_perweek', y='Contacts')
```

Out[20]:
```
<AxesSubplot:xlabel='Outage_sec_perweek', ylabel='Contacts'>
```



In [21]:
```
df['Contacts'].describe()
```

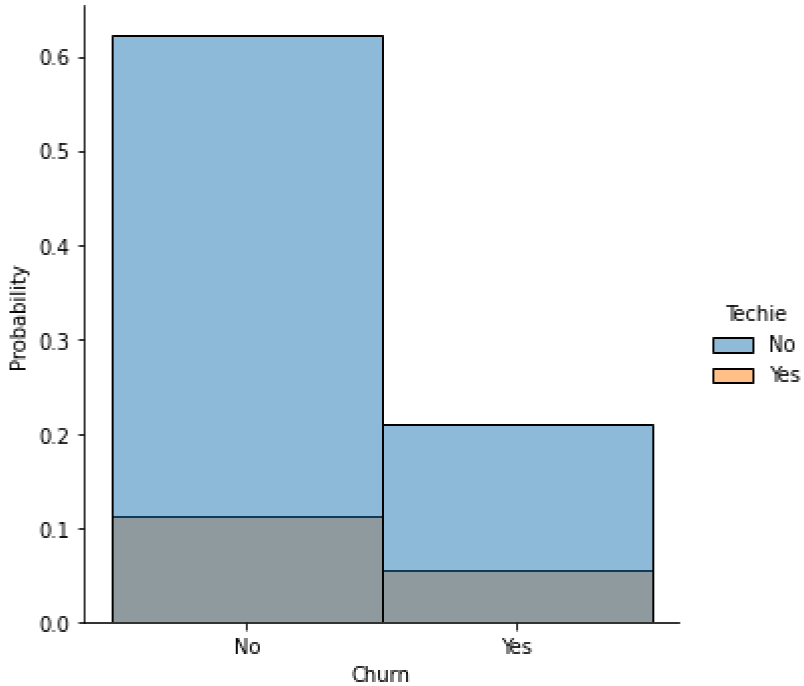
Out[21]:
```
count    10000.000000
mean         0.994200
std          0.988466
min          0.000000
25%          0.000000
50%          1.000000
75%          2.000000
max          7.000000
Name: Contacts, dtype: float64
```

In [22]:
```
df['Outage_sec_perweek'].describe()
```

Out[22]:
```
count    10000.000000
mean        10.001848
std          2.976019
min          0.099747
25%          8.018214
50%         10.018560
75%         11.969485
max         21.207230
Name: Outage_sec_perweek, dtype: float64
```

In [23]:
```
sns.displot(df, x="Churn", hue="Techie", stat="probability")
```

Out[23]:
```
<seaborn.axisgrid.FacetGrid at 0x1efd75962e0>
```



In [24]:
```
df['Churn'].describe()
```

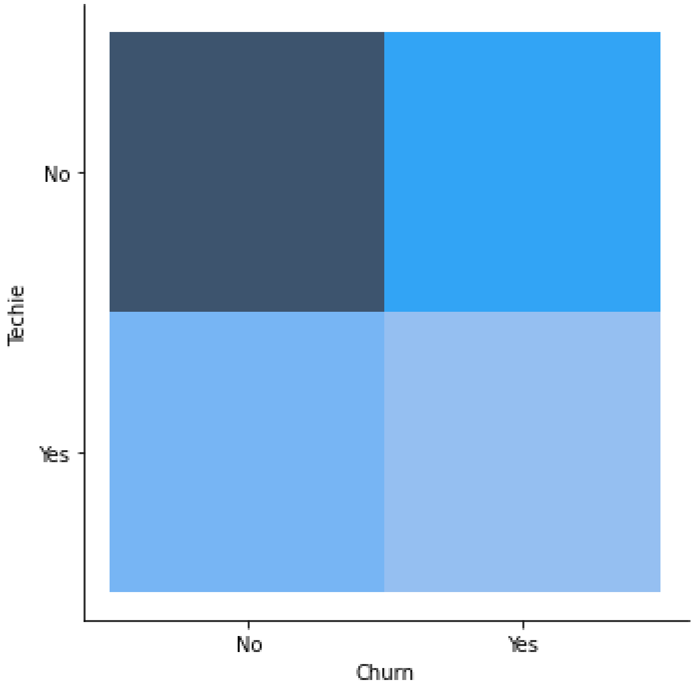
Out[24]:
```
count     10000
unique        2
top          No
freq       7350
Name: Churn, dtype: object
```

In [25]:
```
df['Techie'].describe()
```

Out[25]:
```
count     10000
unique        2
top          No
freq       8321
Name: Techie, dtype: object
```

In [26]:
```
sns.displot(df, x="Churn", y="Techie")
```

Out[26]: <seaborn.axisgrid.FacetGrid at 0x1efd7861be0>



In the above visualization we can see that most customers who said they were not techies stayed with the company. The darker the square the more data is in that square.

Section E

1) The results of my hypothesis test was that we accept the null hypothesis that churn and techie are independent of each other.

2) My data analysis could have been limited if I did not have access to the data or an incomplete data set.

3) I would recommended not paying attention to if a customer is a techie or not as there was no correlation between churn and if a customer viewed themselves as techie or not.

References

https://www.statology.org/two-sample-t-test-python/#:~:text=%20How%20to%20Conduct%20a%20Two%20Sample%20T-Test,3%20Step%203%3A%20Interpret%20the%20results.%20More%20

(https://www.statology.org/two-sample-t-test-python/#:~:text=%20How%20to%20Conduct%20a%20Two%20Sample%20T-Test,3%20Step%203%3A%20Interpret%20the%20results.%20More%20)

https://predictivehacks.com/how-to-run-chi-square-test-in-python/ (https://predictivehacks.com/how-to-run-chi-square-test-in-python/)

https://datagy.io/histogram-python/ (https://datagy.io/histogram-python/)

https://seaborn.pydata.org/tutorial/distributions.html (https://seaborn.pydata.org/tutorial/distributions.html)