```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
import seaborn as sbn
import statsmodels.api as sm

churn_df =
pd.read_csv('D:\WGU\D208_PredictiveModelling\d9rkejv84kd9rk30fi2l\churn_clean.csv')

print(churn_df.head())

df = churn_df.columns
print(df)

#Drop unwanted column
df = churn_df.drop(churn_df.columns[0], axis=1)
print(df.head())

#Rename columns to a more logical name
df.rename(columns={'Item1':'Timely Response',
                   'Item2':'Timely Fixes',
                   'Item3':'Timely Replacements',
                   'Item4':'Reliability',
                   'Item5':'Options',
                   'Item6':'Respectfulness',
                   'Item7':'Courteous',
                   'Item8':'Active Listening'},
          inplace=True)

#Get the number of rows and columns
print(df.shape)
print(df.columns)
#Get the description of the data in the dataframe
print(df.describe())

#Re-validate column data type and missing values
print(df.columns.to_series().groupby(df.dtypes).groups)

#Examine columns having categorical data to see if any mis spellings are present
print(df['State'].unique())
print(df['Area'].unique())
print(df['TimeZone'].unique())
print(df['Children'].unique())
print(sorted(df['Age'].unique()))
print(df['Marital'].unique())
print(df['Gender'].unique())
print(df['Contract'].unique())
print(df['PaymentMethod'].unique())
print(df['Techie'].unique())
print(df['Port_modem'].unique())
print(df['Tablet'].unique())
print(df['InternetService'].unique())
print(df['Phone'].unique())
print(df['Multiple'].unique())
print(df['OnlineSecurity'].unique())
print(df['OnlineBackup'].unique())
print(df['DeviceProtection'].unique())
print(df['TechSupport'].unique())
```

```
print(df['StreamingTV'].unique())
print(df['StreamingMovies'].unique())
print(df['PaperlessBilling'].unique())

#Print any duplicate rows in the data frame
data_duplicates = df.loc[df.duplicated()]
print(data_duplicates)

#Find missing values
print(df.isnull())

#Get the total missing values for each column
data_nulls = df.isnull().sum()
print(data_nulls)

print(df.describe())
df['InternetService'] = df['InternetService'].fillna('None')

#Anomaly Detection and Data Visualization
#Create Histogram of important variables
df[['Outage_sec_perweek','Yearly_equip_failure','Bandwidth_GB_Year','Tenure']].hist
()
plt.savefig('D:\WGU\D208_PredictiveModelling\histogram1.jpg')
#plt.tight_layout()
plt.show()
df[['MonthlyCharge']].hist()
plt.show()

#Finding Outliers using boxplot
df[['Population']].boxplot()
plt.show()
df[['Children']].boxplot()
plt.show()
df[['Age']].boxplot()
plt.show()
df[['Income']].boxplot()
plt.show()
df[['Outage_sec_perweek']].boxplot()
plt.show()
df[['Email']].boxplot()
plt.show()
df[['Contacts']].boxplot()
plt.show()
df[['Yearly_equip_failure']].boxplot()
plt.show()
df[['Tenure']].boxplot()
plt.show()
df[['MonthlyCharge']].boxplot()
plt.show()
df[['Bandwidth_GB_Year']].boxplot()
plt.show()

print(df[['MonthlyCharge']].describe())
print(df[['Outage_sec_perweek']].describe())
print(df[['Yearly_equip_failure']].describe())
print(df[['Tenure']].describe())
print(df[['Bandwidth_GB_Year']].describe())

print(df[['Port_modem']].value_counts())
```

```
print(df[['Phone']].value_counts())
print(df[['OnlineSecurity']].value_counts())
print(df[['OnlineBackup']].value_counts())
print(df[['DeviceProtection']].value_counts())
print(df[['TechSupport']].value_counts())
print(df[['StreamingTV']].value_counts())
print(df[['StreamingMovies']].value_counts())
print(df[['Contract']].value_counts())
print(df[['InternetService']].value_counts())

#Univariate visualization of independent variables
df[['Outage_sec_perweek']].hist()
plt.show()
df[['Yearly_equip_failure']].hist()
plt.show()
df[['Tenure']].hist()
plt.show()
df[['Bandwidth_GB_Year']].hist()
plt.show()

sbn.countplot(x='Port_modem', data=df)
plt.show()
sbn.countplot(x='Contract', data=df)
plt.show()

sbn.countplot(x='InternetService', data=df)
plt.show()

sbn.countplot(x='Phone', data=df)
plt.show()
sbn.countplot(x='OnlineSecurity', data=df)
plt.show()
sbn.countplot(x='OnlineBackup', data=df)
plt.show()
sbn.countplot(x='DeviceProtection', data=df)
plt.show()

sbn.countplot(x='TechSupport', data=df)
plt.show()
sbn.countplot(x='StreamingTV', data=df)
plt.show()
sbn.countplot(x='StreamingMovies', data=df)
plt.show()

#Bivariate analysis
sbn.heatmap(df[['MonthlyCharge','Bandwidth_GB_Year']].corr(), annot=True)
plt.show()

sbn.heatmap(df[['MonthlyCharge','Outage_sec_perweek']].corr(), annot=True)
plt.show()

sbn.heatmap(df[['MonthlyCharge','Yearly_equip_failure']].corr(), annot=True)
plt.show()

sbn.heatmap(df[['MonthlyCharge','Tenure']].corr(), annot=True)
plt.show()

df.boxplot(column='MonthlyCharge', by='InternetService', figsize=(8,8))
plt.show()
```

```
df.boxplot(column='MonthlyCharge', by='Port_modem', figsize=(8,8))
plt.show()

df.boxplot(column='MonthlyCharge', by='Contract', figsize=(8,8))
plt.show()

df.boxplot(column='MonthlyCharge', by='Phone', figsize=(8,8))
plt.show()

df.boxplot(column='MonthlyCharge', by='OnlineBackup', figsize=(8,8))
plt.show()

df.boxplot(column='MonthlyCharge', by='OnlineSecurity', figsize=(8,8))
plt.show()

df.boxplot(column='MonthlyCharge', by='DeviceProtection', figsize=(8,8))
plt.show()

df.boxplot(column='MonthlyCharge', by='TechSupport', figsize=(8,8))
plt.show()

df.boxplot(column='MonthlyCharge', by='StreamingTV', figsize=(8,8))
plt.show()

df.boxplot(column='MonthlyCharge', by='StreamingMovies', figsize=(8,8))
plt.show()

print(df.columns)

df = df.drop(columns=['Customer_id', 'Interaction', 'UID', 'City', 'State',
'County', 'Zip', 'Lat', 'Lng',
                      'Population', 'Area', 'TimeZone', 'Job', 'Marital',
'PaymentMethod'])
print(df.describe())

#Discover missing data points withing dataset
data_nulls = df.isnull().sum()
print(data_nulls)

df['Churn'] = [1 if v == 'Yes' else 0 for v in df['Churn']]
df['Port_modem'] = [1 if v == 'Yes' else 0 for v in df['Port_modem']]
df['Techie'] = [1 if v == 'Yes' else 0 for v in df['Techie']]
df['Tablet'] = [1 if v == 'Yes' else 0 for v in df['Tablet']]
df['Phone'] = [1 if v == 'Yes' else 0 for v in df['Phone']]
df['Multiple'] = [1 if v == 'Yes' else 0 for v in df['Multiple']]
df['OnlineSecurity'] = [1 if v == 'Yes' else 0 for v in df['OnlineSecurity']]
df['OnlineBackup'] = [1 if v == 'Yes' else 0 for v in df['OnlineBackup']]
df['DeviceProtection'] = [1 if v == 'Yes' else 0 for v in df['DeviceProtection']]
df['TechSupport'] = [1 if v == 'Yes' else 0 for v in df['TechSupport']]
df['StreamingTV'] = [1 if v == 'Yes' else 0 for v in df['StreamingTV']]
df['StreamingMovies'] = [1 if v == 'Yes' else 0 for v in df['StreamingMovies']]
df['PaperlessBilling'] = [1 if v == 'Yes' else 0 for v in df['PaperlessBilling']]
df['InternetService'] = [1 if v == 'Fiber Optic' else 0 for v in
df['InternetService']]


df = df[['Children', 'Age', 'Income', 'Gender', 'Churn', 'Outage_sec_perweek',
'Email', 'Contacts',
```

```
  'Yearly_equip_failure', 'Techie', 'Contract', 'Port_modem', 'Tablet',
'InternetService',
  'Phone', 'Multiple', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
'TechSupport', 'StreamingTV',
  'StreamingMovies', 'PaperlessBilling', 'Tenure', 'Bandwidth_GB_Year',
  'Timely Response', 'Timely Fixes', 'Timely Replacements', 'Reliability',
'Options', 'Respectfulness', 'Courteous',
  'Active Listening', 'MonthlyCharge']]

print(df.columns)
sbn.heatmap(df[['MonthlyCharge','StreamingTV']].corr(), annot=True)
plt.show()
df.to_csv('D:\WGU\D208_PredictiveModelling\churn_prepared.csv')
df['Intercept'] = 1
churn_regression_df =
df[['MonthlyCharge','Outage_sec_perweek','Yearly_equip_failure','Port_modem','Inter
netService','Phone','OnlineSecurity',

'OnlineBackup','DeviceProtection','TechSupport','StreamingTV','StreamingMovies','Te
nure','Bandwidth_GB_Year','Intercept']]

lm_monthly_charge =
sm.OLS(churn_regression_df['MonthlyCharge'],churn_regression_df[['Outage_sec_perwee
k','Yearly_equip_failure','Port_modem','InternetService','Phone',

'OnlineSecurity','OnlineBackup','DeviceProtection','TechSupport','StreamingTV',

'StreamingMovies','Tenure','Bandwidth_GB_Year','Intercept']]).fit()
print(lm_monthly_charge.summary())
print(lm_monthly_charge.params['Intercept'])

lm_monthly_charge_reduced =
sm.OLS(churn_regression_df['MonthlyCharge'],churn_regression_df[['Outage_sec_perwee
k','Yearly_equip_failure','Port_modem',

'InternetService','OnlineSecurity','OnlineBackup',

'DeviceProtection','TechSupport','StreamingTV','StreamingMovies',

'Tenure','Bandwidth_GB_Year','Intercept']]).fit()
print(lm_monthly_charge_reduced.summary())
print(lm_monthly_charge_reduced.params['Intercept'])

lm_monthly_charge_reduced =
sm.OLS(churn_regression_df['MonthlyCharge'],churn_regression_df[['Outage_sec_perwee
k','Port_modem',

'InternetService','OnlineSecurity','OnlineBackup',

'DeviceProtection','TechSupport','StreamingTV','StreamingMovies',

'Tenure','Bandwidth_GB_Year','Intercept']]).fit()
print(lm_monthly_charge_reduced.summary())
print(lm_monthly_charge_reduced.params['Intercept'])

lm_monthly_charge_reduced =
sm.OLS(churn_regression_df['MonthlyCharge'],churn_regression_df[['Port_modem',

'InternetService','OnlineSecurity','OnlineBackup',
```

```
'DeviceProtection','TechSupport','StreamingTV','StreamingMovies',

'Tenure','Bandwidth_GB_Year','Intercept']]).fit()
print(lm_monthly_charge_reduced.summary())

lm_monthly_charge_reduced =
sm.OLS(churn_regression_df['MonthlyCharge'],churn_regression_df[['InternetService',
'OnlineSecurity','OnlineBackup',

'DeviceProtection','TechSupport','StreamingTV','StreamingMovies',

'Tenure','Bandwidth_GB_Year','Intercept']]).fit()
print(lm_monthly_charge_reduced.summary())

lm_monthly_charge_reduced =
sm.OLS(churn_regression_df['MonthlyCharge'],churn_regression_df[['InternetService',
'OnlineBackup',

'DeviceProtection','TechSupport','StreamingTV','StreamingMovies',

'Intercept']]).fit()
print(lm_monthly_charge_reduced.summary())


sbn.heatmap(df[['MonthlyCharge','InternetService','OnlineSecurity','OnlineBackup','
DeviceProtection','TechSupport','StreamingTV','StreamingMovies',
                'Tenure','Bandwidth_GB_Year','Intercept']].corr(), annot=True)
plt.show()

#Model Evaluation Metric for reduced model
residuals = churn_regression_df['MonthlyCharge'] -
lm_monthly_charge_reduced.predict(churn_regression_df[['InternetService','OnlineBac
kup','StreamingTV','StreamingMovies',

'DeviceProtection','TechSupport','Tenure','Bandwidth_GB_Year','Intercept']])
sbn.scatterplot(x = churn_regression_df['MonthlyCharge'], y = residuals,
color='red')
plt.show()
```