In [ ]:

```
# import churn dataset

import pandas as pd
import numpy as np
import scipy.stats as sp
df = pd.read_csv("C:/Users/hkeim/OneDrive/Documents/School/D206/churn_raw_data.csv")
df.info()
```

In [ ]:

```
# return the column datatypes
print(df.dtypes)
```

In [ ]:

```
# change categorical data to numeric data
df['area_n']=df['area']
df['area_n'].value_counts()
dict_area_n = {'area_n': {'Rural': 1, 'Suburban': 2, 'Urban': 3}}
df = df.replace(dict_area_n)

df['employment_n']=df['employment']
df['employment_n'].value_counts()
dict_emp_n = {'employment_n': {'Student': 1, 'Unemployed': 2, 'Part Time': 3, 'Full Time': 4,
df = df.replace(dict_emp_n)

df['marital_n']=df['marital']
df['marital_n'].value_counts()
dict_marital_n = {'marital_n': {'Married': 1, 'Never Married': 2, 'Separated': 3, 'Widowed':
df = df.replace(dict_marital_n)

df['gender_n']=df['gender']
df['gender_n'].value_counts()
dict_gender_n = {'gender-_n': {'Female': 1, 'Male': 2, 'Prefer not to answer': 3}}
df = df.replace(dict_gender_n)

df['churn_n']=df['churn']
df['churn_n'].value_counts()
dict_churn_n = {'churn_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_churn_n)

df['techie_n']=df['techie']
df['techie_n'].value_counts()
dict_techie_n = {'techie_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_techie_n)

df['contract_n']=df['contract']
df['contract_n'].value_counts()
dict_contract_n = {'contract_n': {'Month-to-month': 1, 'Two Year': 2, 'One year': 3}}
df = df.replace(dict_contract_n)

df['port_modem_n']=df['port_modem']
df['port_modem_n'].value_counts()
dict_port_n = {'port_modem_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_port_n)

df['tablet_n']=df['tablet']
df['tablet_n'].value_counts()
dict_tablet_n = {'tablet_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_tablet_n)

df['internetservice_n']=df['internetservice']
```

```
df['internetservice_n'].value_counts()
dict_intserv_n = {'internetservice_n': {'None': 1, 'DSL': 2, 'Fiber Optic': 3}}
df = df.replace(dict_intserv_n)

df['phone_n']=df['phone']
df['phone_n'].value_counts()
dict_phone_n = {'phone_n': {'Yes': 1, 'No': 2},}
df = df.replace(dict_phone_n)

df['multiple_n']=df['multiple']
df['multiple_n'].value_counts()
dict_mult_n = {'multiple_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_mult_n)

df['onlinesecurity_n']=df['onlinesecurity']
df['onlinesecurity_n'].value_counts()
dict_onlinesec_n = {'onlinesecurity_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_onlinesec_n)

df['deviceprotection_n']=df['deviceprotection']
df['deviceprotection_n'].value_counts()
dict_devpro_n = {'deviceprotection_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_devpro_n)

df['techsupport_n']=df['techsupport']
df['techsupport_n'].value_counts()
dict_techsup_n = {'techsupport_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_techsup_n)

df['streamingtv_n']=df['streamingtv']
df['streamingtv_n'].value_counts()
dict_streamtv_n = {'streamingtv_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_streamtv_n)

df['streamingmovies_n']=df['streamingmovies']
df['streamingmovies_n'].value_counts()
dict_streammovies_n = {'streamingmovies_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_streammovies_n)

df['paperlessbilling_n']=df['paperlessbilling']
df['paperlessbilling_n'].value_counts()
dict_paper_n = {'paperlessbilling_n': {'Yes': 1, 'No': 2}}
df = df.replace(dict_paper_n)

df['paymentmethod_n']=df['paymentmethod']
df['paymentmethod_n'].value_counts()
dict_pay_n = {'paymentmethod_n': {'Electronic Check': 1, 'Mailed Check': 2, 'Bank Transfer(au
df = df.replace(dict_pay_n)

df.info()
```

In [ ]:

```
# standardize numeric fields

df['population_z'] = sp.zscore(df['population'])
df['children_z'] = sp.zscore(df['children'])
df['age_z'] = sp.zscore(df['age'])
df['income_z'] = sp.zscore(df['income'])
df['outage_sec_perweek_z'] = sp.zscore(df['outage_sec_perweek'])
df['email_z'] = sp.zscore(df['email'])
df['contacts_z'] = sp.zscore(df['contacts'])
df['yearly_equip_failure_z'] = sp.zscore(df['yearly_equip_failure'])
```

```
df['tenure_z'] = sp.zscore(df['tenure'])
df['monthlycharge_z'] = sp.zscore(df['monthlycharge'])
df['bandwidth_gb_year_z'] = sp.zscore(df['bandwidth_gb_year'])
df.info()
```

In [ ]:
```
# query outliers based on zscores

df.query('population_z > 3 | population_z < -3')
```

In [ ]:
```
df.query('children_z > 3 | children_z < -3')
```

In [ ]:
```
df.query('age_z > 3 | age_z < -3')
```

In [ ]:
```
 df.query('income_z > 3 | income_z < -3')
```

In [ ]:
```
df.query('outage_sec_perweek_z > 3 | outage_sec_perweek_z < -3')
```

In [ ]:
```
df.query('email_z > 3 | email_z < -3')
```

In [ ]:
```
df.query('contacts_z > 3 | contacts_z < -3')
```

In [ ]:
```
df.query('yearly_equip_failure_z > 3 | yearly_equip_failure_z < -3')
```

In [ ]:
```
df.query('tenure_z > 3 | tenure_z < -3')
```

In [ ]:
```
df.query('monthlycharge_z > 3 | monthlycharge_z < -3')
```

In [ ]:
```
df.query('bandwidth_gb_year_z > 3 | bandwidth_gb_year_z < -3')
```

In [ ]:
```
#drop outliers

df.drop(df[(df.population_z > 3) | (df.population_z < -3)].index, inplace=True)
df.drop(df[(df.outage_sec_perweek_z > 3) | (df.outage_sec_perweek_z < -3)].index, inplace=Tru
df.drop(df[(df.email_z > 3) | (df.email_z < -3)].index, inplace=True)
df.drop(df[(df.contacts_z > 3) | (df.contacts_z < -3)].index, inplace=True)
df.drop(df[(df.yearly_equip_failure_z > 3) | (df.yearly_equip_failure_z < -3)].index, inplace
df.drop(df[(df.monthlycharge_z > 3) | (df.monthlycharge_z < -3)].index, inplace=True)
df.info()
```

In [ ]:
```
# query outliers based on zscores

df.query('population_z > 3 | population_z < -3')
```

In [ ]:
```
df.query('children_z > 3 | children_z < -3')
```

In [ ]:
```
df.query('age_z > 3 | age_z < -3')
```

In [ ]:
```
df.query('income_z > 3 | income_z < -3')
```

In [ ]:
```
df.query('outage_sec_perweek_z > 3 | outage_sec_perweek_z < -3')
```

In [ ]:
```
df.query('email_z > 3 | email_z < -3')
```

In [ ]:
```
df.query('contacts_z > 3 | contacts_z < -3')
```

In [ ]:
```
df.query('yearly_equip_failure_z > 3 | yearly_equip_failure_z < -3')
```

In [ ]:
```
df.query('tenure_z > 3 | tenure_z < -3')
```

In [ ]:
```
df.query('monthlycharge_z > 3 | monthlycharge_z < -3')
```

In [ ]:
```
df.query('bandwidth_gb_year_z > 3 | bandwidth_gb_year_z < -3')
```

In [ ]:
```
# count the number of null values in each column

df.isna().sum()
```

In [ ]:
```
# count the number of null values in the data set

df.isnull().sum().sum()
```

In [ ]:
```
# drop columns with all null values

df.dropna(axis=1, how='all', inplace=True)
df.info()
print(df.isna().sum().sum())
```

In [ ]:
```
# drop rows with all extra services null

df.dropna(how='all', subset=['phone', 'techsupport'], inplace=True)
df.info()
print(df.isna().sum().sum())
```

In [ ]:
```
# fill remaining nulls with forward and backward fill

df.fillna(method='ffill', inplace=True)
df.fillna(method='bfill', inplace=True)
df.info()
print(df.isna().sum().sum())
```

In [ ]:

```
# create clean dataframe

df_clean = df.filter(['customer_id',
                      'interaction',
                      'city',
                      'state',
                      'county',
                      'zip',
                      'lat',
                      'lng',
                      'population',
                      'area',
                      'timezone',
                      'job',
                      'children',
                      'age',
                      'education',
                      'employment',
                      'income',
                      'marital',
                      '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',
                      'paymentmethod',
                      'tenure',
                      'monthlycharge',
                      'bandwidth_gb_year',
                      'item1',
                      'item2',
                      'item3',
                      'item4',
                      'item5',
                      'item6',
                      'item7',
                      'item8',])
df_clean.reset_index(inplace=True)
df_clean.info()
print(df_clean)
```

In [ ]:

```
# export df_clean to .csv file
```

```
df_clean.to_csv('C:/Users/hkeim/OneDrive/Documents/School/D206/D206 Keim Task One Clean Churn
```

In [ ]:

```

```