outages

February 20, 2020

1 Power Outages

This project uses major power outage data in the continental U.S. from January 2000 to July 2016. Here, a major power outage is defined as a power outage that impacted at least 50,000 customers or caused an unplanned firm load loss of atleast 300MW. Interesting questions to consider include: - Where and when do major power outages tend to occur? - What are the characteristics of major power outages with higher severity? Variables to consider include location, time, climate, land-use characteristics, electricity consumption patterns, economic characteristics, etc. What risk factors may an energy company want to look into when predicting the location and severity of its next major power outage? - What characteristics are associated with each category of cause? - How have characteristics of major power outages changed over time? Is there a clear trend?

1.0.1 Getting the Data

The data is downloadable here.

A data dictionary is available at this article under *Table 1. Variable descriptions*.

1.0.2 Cleaning and EDA

- Note that the data is given as an Excel file rather than a CSV. Open the data in Excel or another spreadsheet application and determine which rows and columns of the Excel spreadsheet should be ignored when loading the data in pandas.
- Clean the data.
  - The power outage start date and time is given by `OUTAGE.START.DATE` and `OUTAGE.START.TIME`. It would be preferable if these two columns were combined into one datetime column. Combine `OUTAGE.START.DATE` and `OUTAGE.START.TIME` into a new datetime column called `OUTAGE.START`. Similarly, combine `OUTAGE.RESTORATION.DATE` and `OUTAGE.RESTORATION.TIME` into a new datetime column called `OUTAGE.RESTORATION`.
- Understand the data in ways relevant to your question using univariate and bivariate analysis of the data as well as aggregations.

Hint 1: pandas can load multiple filetypes: `pd.read_csv`, `pd.read_excel`, `pd.read_html`, `pd.read_json`, etc.

Hint 2: `pd.to_datetime` and `pd.to_timedelta` will be useful here.

Tip: To visualize geospatial data, consider Folium or another geospatial plotting library.

1.0.3 Assessment of Missingness

- Assess the missingness of a column that is not missing by design.

1.0.4 Hypothesis Test

Find a hypothesis test to perform. You can use the questions at the top of the notebook for inspiration.

2 Summary of Findings

2.0.1 Introduction

Power outage dataset includes data of power outage information in the continental U.S. from January 2000 to July 2016. A major power outage, for the purpose of this project, is defined as a power outage that impacted at least 50,000 customers or caused an unplanned firm load loss of atleast 300MW.

This data provides valuable information that can be used to conduct future research in various paradigms, such as—state-level power outage risk maps for the continental U.S., predicting demand load loss, analyzing vulnerability of the U.S. states to frequent major power outages, and studying historical trends of major power outages.

Other than basic information of state, climate, time, this dataset also includes electricity consumption patterns, economic characteristics, and land-use characteristics, each with many sub-sections to signify it's characteristics.

Main question to concern

- Where and when do major power outages tend to occur?
  - Tested under **Exploratory Data Analysis (EDA)**
  - The question of "where" includes [states, climate region]
  - The question of "when" includes [year, month, time]
- What are the characteristics of major power outages with higher severity?
  - Missingness of Demand Loss and Customers are assessed under **Assessment of Missingness**
  - Demand Loss and Customers Affected are combined using PCA to create a single measurement for Severity
  - Relationship with Location, and Climate Category, Time, Climate anomaly level, land-use characteristics, electricity consumption patterns, and economic characteristics are tested under **Exploratory Data Analysis**
    * Differnt columns within characteristics are combined using PCA to create a single measurement for each characterstic
- What characteristics are associated with each category of cause?

    - Tested under **Exploratory Data Analysis**
- How have characteristics of major power outages changed over time? Is there a clear trend?
    - Tested under **Exploratory Data Analysis**
- What characteristics are associated with each category of cause?
    - Is the cause category 1) Demand Loss; 2) Customer Affected; 3) Outage Duration of major power outage similar to that of non-major power outage?
        * Tested under **Hypothesis Testing** using Permutation Tests

2.0.2 Cleaning

- Load data with useful columns of interest
- Combine START & RESTORATION Date + Time into one column respectively
- PCA (knowledge from DSC 40A) for elec, econ and land characteristic
- Get major power outage (n > 50,000 & MW > 300)

2.0.3 Exploratory Data Analysis (EDA)

1) Where does power outage tend to occur?

- CA has the highest number of power outage, while TX has the highest number of major power outage
- MA seems to have both the highest average demand loss (MW) for both power outage and major power outage
- Northeast area has the highest number of power outage, while southeast area has the highest number of major power outage
- Southeast area has both the highest average amount of demand loss for power outage and major power outage
- Normal climate tend to have more power outage and more major power outage than cold or warm climate
- Warm climate has the highest average amount of demand loss (MW) for power outage, while both normal and warm climate seem to have the same average amount of demand loss (MW) for major power outage

2) When does power outage tend to occur?

By Year

- Year 2011 has the highest number of power outage (way more than the second place), while year 2004 and 2008 have leading number of major power outage
- Year 2014 and Year 2003 have similar average amount of demand loss (MW) for power outage, while Year 2003 has the highest average amount of demand loss (MW) for major power outage ##### By Month
- June seems to be the month in which people encounter the highest number of both normal power outage and major power outage

- August seems to be the month in which people encounter the highest average amount of demand loss (MW) for both normal power outage and major power outage ##### By Time
- People tend to encounter normal power outage from 12pm - 4pm, but they tend to encounter major power outage from 4pm - 7pm
- When encountered normal power outage, those in 4pm - 7pm have the highest average amount of demand loss. But when encountered major power outage, those in 12pm - 4pm have the highest average amount of demand loss

3) Correlation of characteristics with outage severity

- Demand Loss and Customers Affected are moderately correalted with each other (r = 0.52)
- However, even though the three characteristics (electricity, economic, land-use) are highly correalted with each other, none of them are even moderately correalted with Demand Loss or Number of Customers Affected, which may imply that those three characteristics do not really affect the severity of power outage a lot
- Anomaly level, which intuitively thinking would potentially be correlated with severity of power outage, actually shows really weak association with Demand Loss and Number of Customers Affected
- Outage duration also shows a weak correlation with Demand Loss and Number of Customers Affected

4) Characteristics of major outage over time

- Verifies the above correlation that Demand Loss and Number of Customers Affected are moderately correlated with each other. Whenever there is a spike (increase) in Demand Loss, there would usually be an increase in the Number of Customers Affected shown in the graph
- Outage duration does not show highly consistent change with Demand Loss and Number of Customers Affected, which can be verified in the above heat map too
- Electricity, economic, and land-use characteristic show highly consistent changes through time

2.0.4 Assessment of Missingness

1) Missingness of Demand Loss

- Demand Loss (MW) is MAR dependent on Year, State, Climate Regions, Anomaly levels, Cause Category, Outage start/restoration, Electricity consumption, Economic characteristic, and Land-use characteristic

2) Missingness of Customers Affected

- Number of Customers Affected is MAR dependent on Year, Month, State, Climate Regions, Anomaly levels, Cause Category, Demand Loss (MW), Outage start/restoration, and Economic characteristic

2.0.5 Hypothesis Test

Permutation Test 1 - Demand Loss

- p value > 0.05. Fail to reject the null hypothesis that Demand Loss of cause category for both groups come from the same distribution

Permutation Test 2 - Customer Affected

- p value < 0.05. Reject the null hypothesis that Number of Customers Affected of cause category for both groups come from the same distribution

Permutation Test 3 - Outage Duration

- p value > 0.05. Fail to reject the null hypothesis that Outage duration of cause category for both groups come from the same distribution

3 Code

[1]:
```
%load_ext autoreload
%autoreload 2
```

[2]:
```
import matplotlib.pyplot as plt
import numpy as np
import os
import pandas as pd
import seaborn as sns
import folium
# import json
# import warnings
# warnings.simplefilter(action="ignore", category=RuntimeWarning)
import calendar
from scipy.cluster.vq import whiten
from sklearn.decomposition import PCA
%matplotlib inline
%config InlineBackend.figure_format = 'retina'  # Higher resolution figures

import util
```

3.1 Cleaning

3.1.1 1) Load data with useful columns of interest

[3]:
```
# Load data
to_drop = ['variables', 'OBS', 'CAUSE.CATEGORY.DETAIL', 'HURRICANE.NAMES']
fp = os.path.join('data', 'outage.xlsx')
df = pd.read_excel(fp, header=0, skiprows=[0, 1, 2, 3, 4, 6]).
↪drop(columns=to_drop) # Load df, skip unuseful rows
df.head()
```

[3]:
```
   YEAR  MONTH U.S._STATE POSTAL.CODE NERC.REGION       CLIMATE.REGION  \
0  2011    7.0  Minnesota          MN         MRO  East North Central
1  2014    5.0  Minnesota          MN         MRO  East North Central
2  2010   10.0  Minnesota          MN         MRO  East North Central
3  2012    6.0  Minnesota          MN         MRO  East North Central
4  2015    7.0  Minnesota          MN         MRO  East North Central

   ANOMALY.LEVEL CLIMATE.CATEGORY OUTAGE.START.DATE OUTAGE.START.TIME  \
0           -0.3           normal        2011-07-01          17:00:00
1           -0.1           normal        2014-05-11          18:38:00
2           -1.5             cold        2010-10-26          20:00:00
3           -0.1           normal        2012-06-19          04:30:00
4            1.2             warm        2015-07-18          02:00:00

         …       POPPCT_URBAN POPPCT_UC POPDEN_URBAN  POPDEN_UC  \
0        …              73.27     15.28       2279.0     1700.5
1        …              73.27     15.28       2279.0     1700.5
2        …              73.27     15.28       2279.0     1700.5
3        …              73.27     15.28       2279.0     1700.5
4        …              73.27     15.28       2279.0     1700.5

   POPDEN_RURAL  AREAPCT_URBAN  AREAPCT_UC   PCT_LAND  PCT_WATER_TOT  \
0          18.2           2.14         0.6  91.592666       8.407334
1          18.2           2.14         0.6  91.592666       8.407334
2          18.2           2.14         0.6  91.592666       8.407334
3          18.2           2.14         0.6  91.592666       8.407334
4          18.2           2.14         0.6  91.592666       8.407334

   PCT_WATER_INLAND
0          5.478743
1          5.478743
2          5.478743
3          5.478743
4          5.478743

[5 rows x 53 columns]
```

[ ]:

```

```

3.1.2 2) Combine START & RESTORATION Date + Time into one column respectively

Date and Time of START and RESTORATION can be combined into one column of datetime object to represent the date.

[4]:

```
# Combine date and time into datetime
df['OUTAGE.START'] = (df['OUTAGE.START.DATE'] +
                      pd.to_timedelta(df['OUTAGE.START.TIME']
                      .astype(str))) # Combine START into date
df['OUTAGE.RESTORATION'] = (df['OUTAGE.RESTORATION.DATE'] +
                            pd.to_timedelta(df['OUTAGE.RESTORATION.TIME']
                            .astype(str))) # Combine RESTORATION into date
outage = df.drop(columns=['OUTAGE.START.DATE', 'OUTAGE.START.TIME',
                          'OUTAGE.RESTORATION.DATE', 'OUTAGE.RESTORATION.
 ↪TIME']) # Drop columns
outage.head()
```

[4]:

```
   YEAR  MONTH U.S._STATE POSTAL.CODE NERC.REGION      CLIMATE.REGION  \
0  2011    7.0  Minnesota          MN         MRO  East North Central
1  2014    5.0  Minnesota          MN         MRO  East North Central
2  2010   10.0  Minnesota          MN         MRO  East North Central
3  2012    6.0  Minnesota          MN         MRO  East North Central
4  2015    7.0  Minnesota          MN         MRO  East North Central

   ANOMALY.LEVEL CLIMATE.CATEGORY      CAUSE.CATEGORY  OUTAGE.DURATION  \
0           -0.3           normal      severe weather           3060.0
1           -0.1           normal  intentional attack              1.0
2           -1.5             cold      severe weather           3000.0
3           -0.1           normal      severe weather           2550.0
4            1.2             warm      severe weather           1740.0

         …            POPDEN_URBAN  POPDEN_UC  POPDEN_RURAL  AREAPCT_URBAN  \
0        …                  2279.0     1700.5          18.2           2.14
1        …                  2279.0     1700.5          18.2           2.14
2        …                  2279.0     1700.5          18.2           2.14
3        …                  2279.0     1700.5          18.2           2.14
4        …                  2279.0     1700.5          18.2           2.14

   AREAPCT_UC   PCT_LAND  PCT_WATER_TOT  PCT_WATER_INLAND        OUTAGE.START  \
0         0.6  91.592666       8.407334          5.478743 2011-07-01 17:00:00
1         0.6  91.592666       8.407334          5.478743 2014-05-11 18:38:00
2         0.6  91.592666       8.407334          5.478743 2010-10-26 20:00:00
3         0.6  91.592666       8.407334          5.478743 2012-06-19 04:30:00
```

```
4         0.6  91.592666       8.407334         5.478743 2015-07-18 02:00:00

   OUTAGE.RESTORATION
0 2011-07-03 20:00:00
1 2014-05-11 18:39:00
2 2010-10-28 22:00:00
3 2012-06-20 23:00:00
4 2015-07-19 07:00:00

[5 rows x 51 columns]
```

[ ]:

3.1.3 3) PCA (knowledge from DSC 40A) for elec, econ and land characteristic

Electricity consumption has 18 subsections, economic characteristic has 9 subsections, and land-use characteristic has 11 subsections. These subsections/columns would be too much for future analysis. Since we need to further examine the relationship of land-use characteristics, electricity consumption patterns, and economic characteristics with 1) outage severity, 2) category cause, and 3) time, it is better that we combine all the subsections within on characteristic into one wholistic value to represent each characteristic.

In order to do so, Pricipal Component Analysis (PCA) would be a good way for reducing dimensions of charactersitics, since all the subsections under each characteristic are somehow correlated with that characteristic.

[5]:

```python
# Cols of location, time, climate, elect, econ, land characteristics
location = ['U.S._STATE', 'POSTAL.CODE', 'NERC.REGION']
time = ['OUTAGE.DURATION', 'OUTAGE.START', 'OUTAGE.RESTORATION', 'TIME.TILE']
climate = ['CLIMATE.REGION', 'ANOMALY.LEVEL', 'CLIMATE.CATEGORY']
elec_chara = (['RES.PRICE', 'COM.PRICE', 'IND.PRICE',
               'TOTAL.PRICE', 'RES.SALES', 'COM.SALES',
               'IND.SALES', 'TOTAL.SALES', 'RES.PERCEN',
               'COM.PERCEN', 'IND.PERCEN', 'RES.CUSTOMERS',
               'COM.CUSTOMERS', 'IND.CUSTOMERS', 'TOTAL.CUSTOMERS',
               'RES.CUST.PCT', 'COM.CUST.PCT', 'IND.CUST.PCT'])
econ_chara = (['PC.REALGSP.STATE', 'PC.REALGSP.USA', 'PC.REALGSP.REL',
               'PC.REALGSP.CHANGE', 'UTIL.REALGSP', 'TOTAL.REALGSP',
               'UTIL.CONTRI', 'PI.UTIL.OFUSA'])
land_chara = (['POPULATION', 'POPPCT_URBAN', 'POPPCT_UC',
               'POPDEN_URBAN', 'POPDEN_UC', 'POPDEN_RURAL',
               'AREAPCT_URBAN', 'AREAPCT_UC', 'PCT_LAND',
               'PCT_WATER_TOT', 'PCT_WATER_INLAND'])
# 'YEAR', 'MONTH', 'CAUSE.CATEGORY', 'DEMAND.LOSS.MW', 'CUSTOMERS.AFFECTED'
```

[6]:

```python
# Function to compute PCA
def pca_fit(df, names):
```

```
    # pull out required data
    dms = []
    for name in names:
        dms.append(df[name])

    # initialize PCA object
    pca = PCA(n_components=2, whiten=True)
    # combine data
    pca_data = np.array(dms).T
    # fit the dimensionality reduction model
    pca_fit = pca.fit_transform(pca_data)

    return pca_fit
```

[7]:
```
# Get the null index of elec, land for future refill
elec_isnull = outage[elec_chara].isnull().any(axis=1)
elec_todrop = outage[elec_isnull].index # Null index of

land_isnull = outage[land_chara].isnull().any(axis=1)
land_todrop = outage[land_isnull].index
```

[8]:
```
pca_elec = pca_fit(outage.fillna(0), elec_chara) # Combined PCA of elec
pca_econ = pca_fit(outage, econ_chara) # Combined PCA of econ
pca_land = pca_fit(outage.fillna(0), land_chara) # Combined PCA of land
outage = (outage.assign(ELEC_CHARA=[elec[0] for elec in pca_elec], # Add PCA␣
 ↪cols
                        ECON_CHARA=[econ[0] for econ in pca_econ],
                        LAND_CHARA=[land[0] for land in pca_land])
                .drop(columns=elec_chara+econ_chara+land_chara)) # Drop chara␣
 ↪cols
outage.loc[elec_todrop, 'ELEC_CHARA'] = np.nan # Refill NaN for elec
outage.loc[land_todrop, 'LAND_CHARA'] = np.nan # Refill NaN for land
outage.head()
```

[8]:
```
   YEAR  MONTH U.S._STATE POSTAL.CODE NERC.REGION      CLIMATE.REGION  \
0  2011    7.0  Minnesota          MN         MRO  East North Central
1  2014    5.0  Minnesota          MN         MRO  East North Central
2  2010   10.0  Minnesota          MN         MRO  East North Central
3  2012    6.0  Minnesota          MN         MRO  East North Central
4  2015    7.0  Minnesota          MN         MRO  East North Central

   ANOMALY.LEVEL CLIMATE.CATEGORY      CAUSE.CATEGORY  OUTAGE.DURATION  \
0           -0.3           normal      severe weather           3060.0
1           -0.1           normal  intentional attack              1.0
2           -1.5             cold      severe weather           3000.0
3           -0.1           normal      severe weather           2550.0
4            1.2             warm      severe weather           1740.0
```

```
   DEMAND.LOSS.MW  CUSTOMERS.AFFECTED        OUTAGE.START  OUTAGE.RESTORATION  \
0             NaN             70000.0 2011-07-01 17:00:00 2011-07-03 20:00:00
1             NaN                 NaN 2014-05-11 18:38:00 2014-05-11 18:39:00
2             NaN             70000.0 2010-10-26 20:00:00 2010-10-28 22:00:00
3             NaN             68200.0 2012-06-19 04:30:00 2012-06-20 23:00:00
4           250.0            250000.0 2015-07-18 02:00:00 2015-07-19 07:00:00

   ELEC_CHARA  ECON_CHARA  LAND_CHARA
0   -0.616168   -0.626816   -0.679064
1   -0.726823   -0.598510   -0.669629
2   -0.736153   -0.636831   -0.682285
3   -0.684369   -0.621319   -0.676266
4   -0.661444   -0.598405   -0.666819
```

[ ]:

3.1.4 4) Get major power outage (n > 50,000 & MW > 300)

Major power outage, which is defined as demand loss > 300MW and number of customers affected > 50,000, is the major analysis subject of my project. Therefore, getting a dataframe beforehand would be helpful.

[9]:
```
# Maybe need to check the missingness before analysis
major_outage = (outage[(outage['DEMAND.LOSS.MW'] > 300) & # Firm load loss >␣
 ↪300MW
                        (outage['CUSTOMERS.AFFECTED'] > 50000)] # Customers >␣
 ↪50000
                        .reset_index(drop=True)) # Reset index
major_outage.head()
```

[9]:
```
   YEAR  MONTH     U.S._STATE POSTAL.CODE NERC.REGION      CLIMATE.REGION  \
0  2011    4.0      Tennessee          TN        SERC             Central
1  2009    6.0      Tennessee          TN        SERC             Central
2  2005    9.0      Wisconsin          WI         RFC  East North Central
3  2014    6.0      Wisconsin          WI         MRO  East North Central
4  2012    6.0  West Virginia          WV         RFC             Central

   ANOMALY.LEVEL CLIMATE.CATEGORY  CAUSE.CATEGORY  OUTAGE.DURATION  \
0           -0.5             cold  severe weather           5054.0
1            0.4           normal  severe weather            136.0
2            0.0           normal  severe weather           4410.0
3            0.0           normal  severe weather            538.0
4           -0.1           normal  severe weather           9576.0

   DEMAND.LOSS.MW  CUSTOMERS.AFFECTED        OUTAGE.START  OUTAGE.RESTORATION  \
```

```
0          359.0          63000.0 2011-04-04 11:47:00 2011-04-08 00:01:00
1          860.0         136000.0 2009-06-12 16:37:00 2009-06-12 18:53:00
2          600.0         110000.0 2005-09-13 18:30:00 2005-09-16 20:00:00
3          424.0         120000.0 2014-06-30 17:55:00 2014-07-01 02:53:00
4          700.0         265000.0 2012-06-29 18:24:00 2012-07-06 10:00:00

   ELEC_CHARA  ECON_CHARA  LAND_CHARA
0   -0.522612   -0.651248   -0.588161
1   -0.475129   -0.669702   -0.596157
2   -0.656703   -0.663657   -0.661923
3   -0.665971   -0.635500   -0.643465
4   -1.072693   -0.958495   -0.981284
```

[ ]:

```

```

[ ]:

```

```

3.2 Exploratory Data Analysis

3.2.1 1) Where does power outage tend to occur?

State differences in normal power outage and major power outage

[10]:

```python
tot = outage['U.S._STATE'].unique()
major = major_outage['U.S._STATE'].unique()
diff = list(set(tot) - set(major))
print(np.array(diff))
```

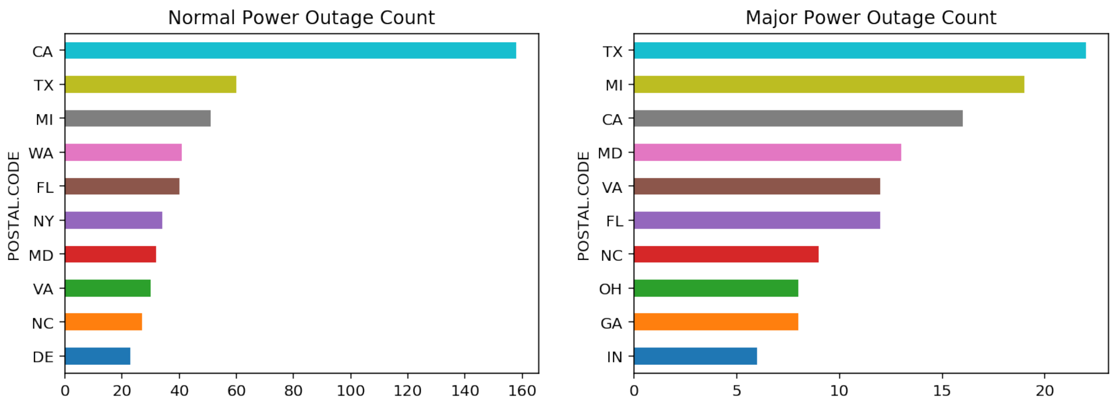
```
['Colorado' 'Nevada' 'Wyoming' 'North Dakota' 'Delaware' 'Alaska' 'Idaho'
 'South Dakota' 'Maine' 'Vermont' 'New Hampshire' 'Connecticut'
 'Mississippi' 'Montana' 'Minnesota' 'Missouri']
```

Count number of times normal vs. major power outage occur across STATE

[11]:

```python
fig, axes = plt.subplots(1, 2, figsize=(12,4))
count = outage.groupby('POSTAL.CODE').count()['DEMAND.LOSS.MW']
(count.nlargest(10).sort_values(ascending=True)
                   .plot(kind='barh', ax=axes[0],
                         title='Normal Power Outage Count')) # Highest 10␣
 ↪demand loss count

count_m = major_outage.groupby('POSTAL.CODE').count()['DEMAND.LOSS.MW']
(count_m.nlargest(10).sort_values(ascending=True)
                     .plot(kind='barh', ax=axes[1],
                           title='Major Power Outage Count')) # Highest 10␣
 ↪demand loss count
```

[11]: <matplotlib.axes._subplots.AxesSubplot at 0x7f195bfa7a90>



- Normal Power Outage Count (Choropleth)

[12]:
```
util.choropleth(count, 'POSTAL.CODE','DEMAND.LOSS.MW', 'Demand Loss (MW)',␣
↪'YlGn')
```

[12]: <folium.folium.Map at 0x7f195bf71048>

- Major Power Outage Count (Choropleth)

[13]:
```
util.choropleth(count_m, 'POSTAL.CODE','DEMAND.LOSS.MW', 'Demand Loss (MW)',␣
↪'YlGn')
```

[13]: <folium.folium.Map at 0x7f19599fd6d8>

CA has the highest number of power outage throughout the whole data set. However, TX has the highest number of major power outage throughout the whole dataset. This means that number of outage does not necessarily mean that the outage is severe. As you can see from the choropleth above, some of the states (in back color), including 'Nevada' 'Idaho' 'Alaska' 'Maine' 'North Dakota' 'Delaware' 'Montana' 'Wyoming' 'Minnesota' 'Mississippi' 'Connecticut' 'New Hampshire' 'Colorado' 'Vermont' 'South Dakota' 'Missouri'], do not even have any major power outage.

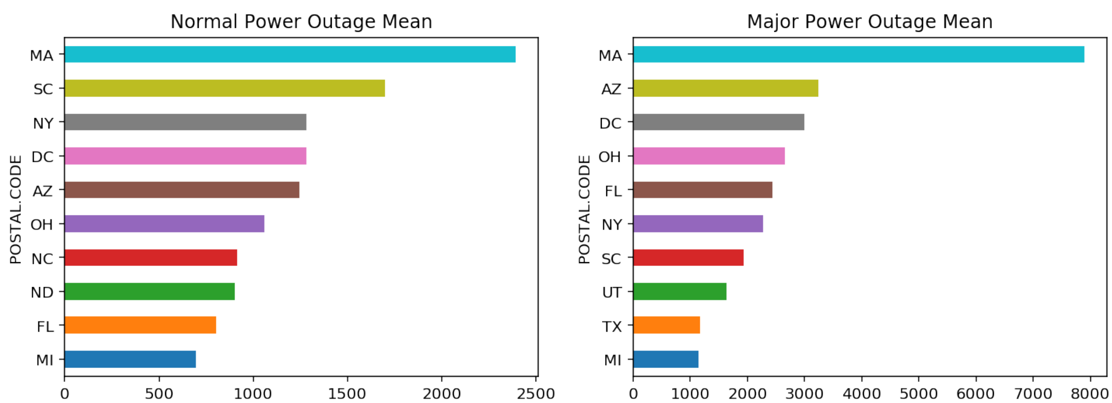
[ ]:
```

```

Average normal vs. major power outage MW across STATE

[14]:
```
fig, axes = plt.subplots(1, 2, figsize=(12,4))
mean_loss = outage.groupby('POSTAL.CODE').mean()['DEMAND.LOSS.MW']
(mean_loss.nlargest(10).sort_values(ascending=True)
                        .plot(kind='barh', ax=axes[0],
                              title='Normal Power Outage Mean')) # Highes 10 mean
```

```
mean_loss_m = major_outage.groupby('POSTAL.CODE').mean()['DEMAND.LOSS.MW']
(mean_loss_m.nlargest(10).sort_values(ascending=True)
                            .plot(kind='barh', ax=axes[1],
                                  title='Major Power Outage Mean')) # Highes 10␣
 ↪mean
```

[14]: `<matplotlib.axes._subplots.AxesSubplot at 0x7f195c313ef0>`



- Normal Power Outage Mean (Choropleth)

[15]:
```
util.choropleth(mean_loss, 'POSTAL.CODE','DEMAND.LOSS.MW', 'Demand Loss (MW)',␣
 ↪'YlGn')
```

[15]: `<folium.folium.Map at 0x7f195c0b8c50>`

- Major Power Outage Mean (Choropleth)

[16]:
```
util.choropleth(mean_loss_m, 'POSTAL.CODE','DEMAND.LOSS.MW', 'Demand Loss␣
 ↪(MW)', 'YlGn') #
```

[16]: `<folium.folium.Map at 0x7f195c0ebc50>`

MA seems to have both the highest average demand loss (MW) for both power outage and major power outage.

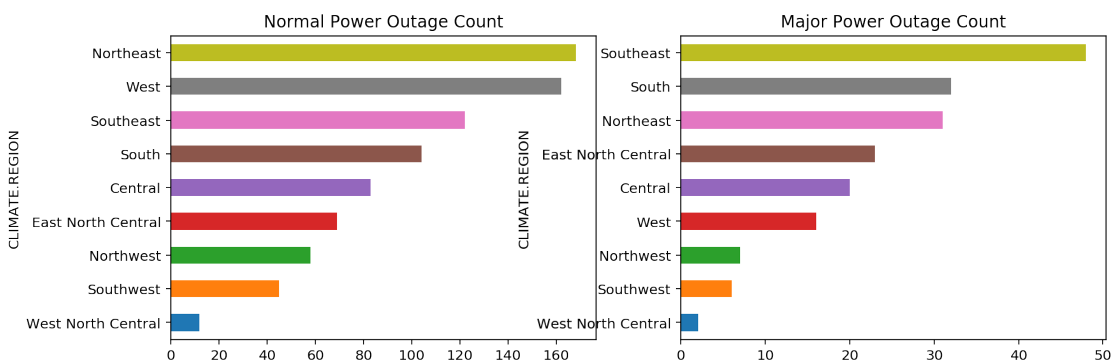
[ ]:
```

```

[ ]:
```

```

Count number of times normal vs. major power outage occur across CLIMATE REGION

[17]:
```
fig, axes = plt.subplots(1, 2, figsize=(12,4))
count = outage.groupby('CLIMATE.REGION').count()['DEMAND.LOSS.MW']
(count.nlargest(10).sort_values(ascending=True)
                   .plot(kind='barh', ax=axes[0],
                        title='Normal Power Outage Count'))# Highest 10 demand␣
 ↪loss count

count_m = major_outage.groupby('CLIMATE.REGION').count()['DEMAND.LOSS.MW']
(count_m.nlargest(10).sort_values(ascending=True)
                     .plot(kind='barh', ax=axes[1],
                          title='Major Power Outage Count')) # Highest 10␣
 ↪demand loss count
```

[17]: <matplotlib.axes._subplots.AxesSubplot at 0x7f195c1a5630>



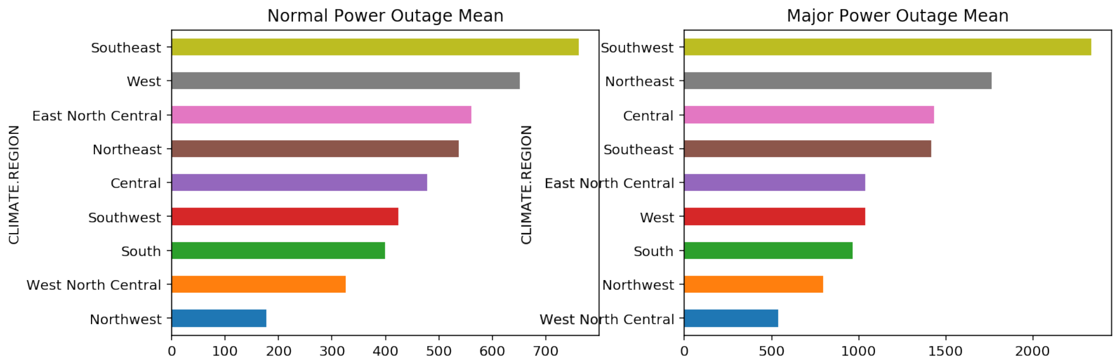
Northeast area has the highest number of power outage, while southeast area has the highest number of major power outage.

[ ]:

Average normal vs. major power outage MW across CLIMATE REGION

[18]:
```
fig, axes = plt.subplots(1, 2, figsize=(12,4))
mean_loss = outage.groupby('CLIMATE.REGION').mean()['DEMAND.LOSS.MW']
(mean_loss.nlargest(10).sort_values(ascending=True)
                       .plot(kind='barh', ax=axes[0],
                            title='Normal Power Outage Mean'))# Highest 10 mean

mean_loss_m = major_outage.groupby('CLIMATE.REGION').mean()['DEMAND.LOSS.MW']
(mean_loss_m.nlargest(10).sort_values(ascending=True).
 ↪sort_values(ascending=True)
                         .plot(kind='barh', ax=axes[1],
                              title='Major Power Outage Mean')) # Highest 10␣
 ↪mean
```

[18]: <matplotlib.axes._subplots.AxesSubplot at 0x7f195837e908>



Southeast area has both the highest average amount of demand loss for power outage and major power outage.

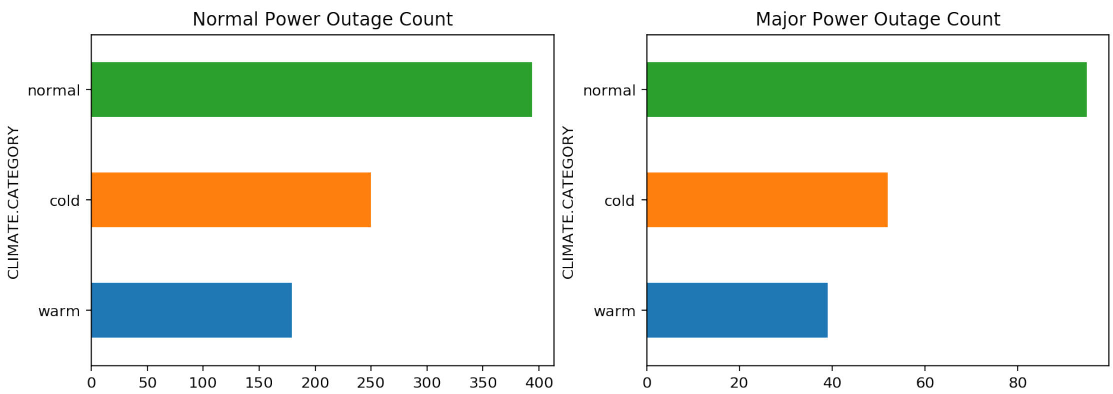
[ ]:

[ ]:

Count number of times normal vs. major power outage occur across CLIMATE CATEGORY

[19]:
```
fig, axes = plt.subplots(1, 2, figsize=(12,4))
count = outage.groupby('CLIMATE.CATEGORY').count()['DEMAND.LOSS.MW']
(count.nlargest(10).sort_values(ascending=True)
                    .plot(kind='barh', ax=axes[0],
                          title='Normal Power Outage Count'))# Highest 10 demand␣
↪loss count

count_m = major_outage.groupby('CLIMATE.CATEGORY').count()['DEMAND.LOSS.MW']
(count_m.nlargest(10).sort_values(ascending=True)
                      .plot(kind='barh', ax=axes[1],
                            title='Major Power Outage Count')) # Highest 10␣
↪demand loss count
```

[19]: <matplotlib.axes._subplots.AxesSubplot at 0x7f19582eb0f0>



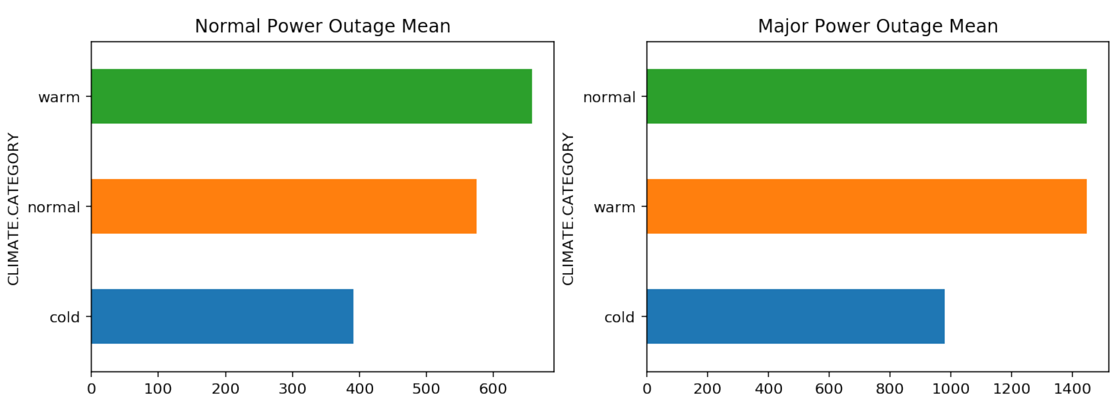
Normal climate tend to have more power outage and more major power outage than cold or warm climate.

```
[ ]:
```

Average normal vs. major power outage MW across CLIMATE CATEGORY

```
[20]: fig, axes = plt.subplots(1, 2, figsize=(12,4))
mean_loss = outage.groupby('CLIMATE.CATEGORY').mean()['DEMAND.LOSS.MW']
(mean_loss.nlargest(10).sort_values(ascending=True)
                        .plot(kind='barh', ax=axes[0],
                              title='Normal Power Outage Mean'))# Highest 10 mean

mean_loss_m = major_outage.groupby('CLIMATE.CATEGORY').mean()['DEMAND.LOSS.MW']
(mean_loss_m.nlargest(10).sort_values(ascending=True).
 ↪sort_values(ascending=True)
                          .plot(kind='barh', ax=axes[1],
                                title='Major Power Outage Mean')) # Highest 10␣
 ↪mean
```

```
[20]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1958224cc0>
```



Warm climate has the highest average amount of demand loss (MW) for power outage, while both normal and warm climate seem to have the same average amount of demand loss (MW) for major power outage.

[ ]:

```

```

[ ]:

```

```

[ ]:

```

```

3.2.2 2) When does power outage tend to occur?

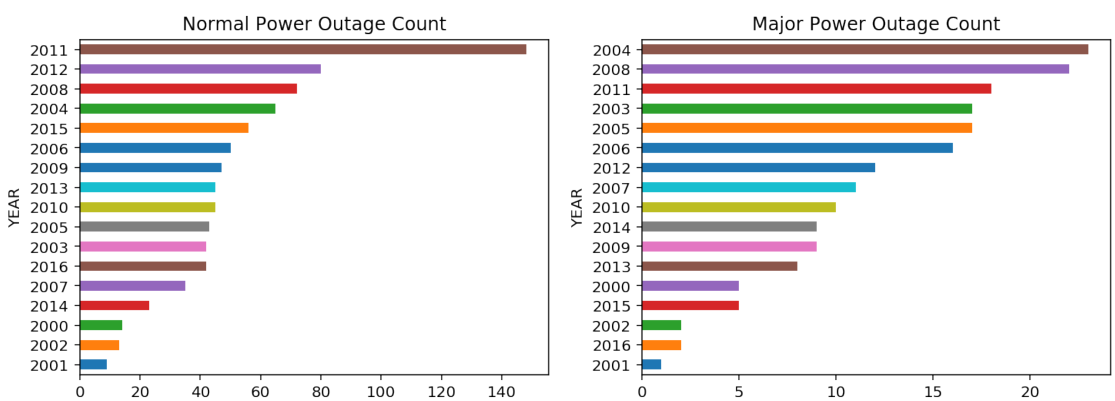
By YEAR

- Count number of times normal vs. major power outage occur across YEAR

[21]:

```
fig, axes = plt.subplots(1, 2, figsize=(12,4))
count = outage.groupby('YEAR').count()['DEMAND.LOSS.MW']
(count.sort_values(ascending=True)
      .plot(kind='barh', ax=axes[0],
            title='Normal Power Outage Count')) # Highest 10 demand loss count

count_m = major_outage.groupby('YEAR').count()['DEMAND.LOSS.MW']
(count_m.sort_values(ascending=True)
        .plot(kind='barh', ax=axes[1],
              title='Major Power Outage Count')) # Highest 10 demand loss count
```

[21]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1958166b38>



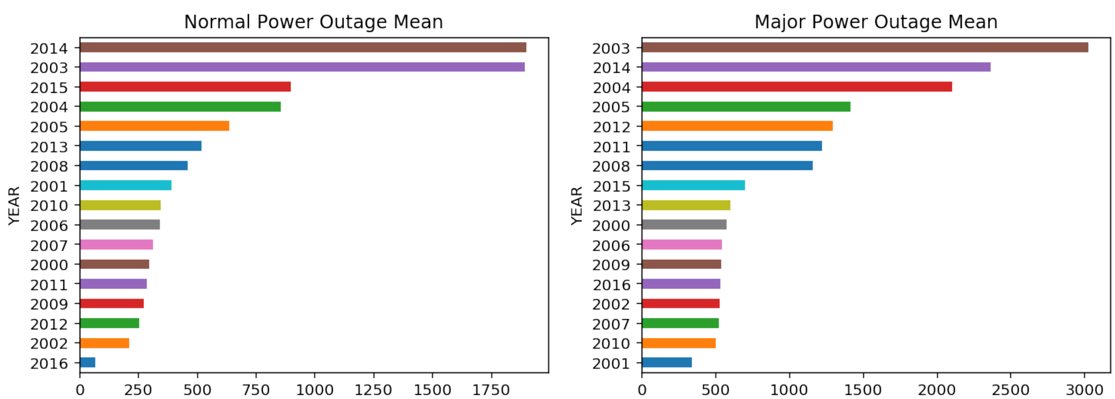
Year 2011 has the highest number of power outage (way more than the second place), while year 2004 and 2008 have leading number of major power outage.

```
[ ]:
```

- Average normal vs. major power outage MW across YEAR

```
[22]: fig, axes = plt.subplots(1, 2, figsize=(12,4))
mean_loss = outage.groupby('YEAR').mean()['DEMAND.LOSS.MW']
(mean_loss.sort_values(ascending=True)
          .plot(kind='barh', ax=axes[0],
                title='Normal Power Outage Mean')) # Highest 10 mean

mean_loss_m = major_outage.groupby('YEAR').mean()['DEMAND.LOSS.MW']
(mean_loss_m.sort_values(ascending=True)
            .plot(kind='barh', ax=axes[1],
                title='Major Power Outage Mean')) # Highest 10 mean
```

```
[22]: <matplotlib.axes._subplots.AxesSubplot at 0x7f19583d3978>
```



Year 2014 and Year 2003 have similar average amount of demand loss (MW) for power outage, while Year 2003 has the highest average amount of demand loss (MW) for major power outage.

```
[ ]:
```

```
[ ]:
```

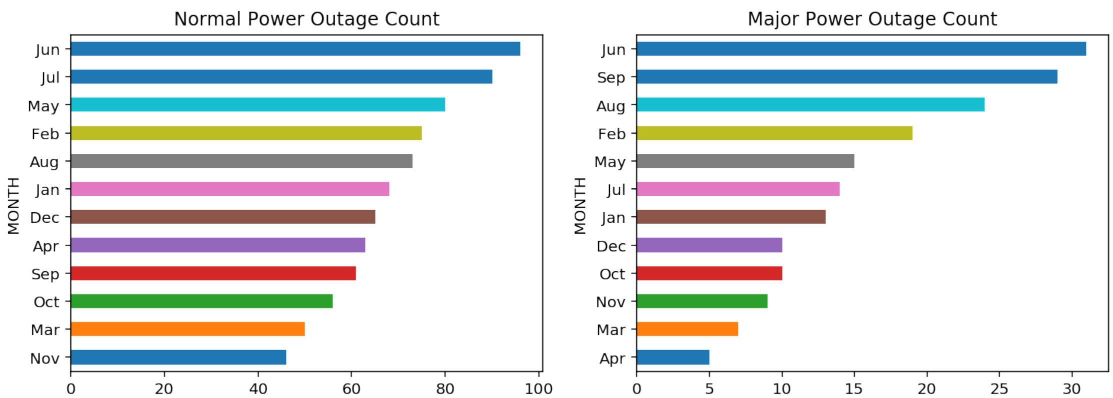
By MONTH

- Count number of times normal vs. major power outage occur across MONTH

```
[23]: fig, axes = plt.subplots(1, 2, figsize=(12,4))
count = outage.groupby('MONTH').count()['DEMAND.LOSS.MW']
count.index = count.index.map(lambda m: calendar.month_abbr[int(m)])
(count.sort_values(ascending=True)
      .plot(kind='barh', ax=axes[0],
```

```
              title='Normal Power Outage Count')) # Highest 10 demand loss count

count_m = major_outage.groupby('MONTH').count()['DEMAND.LOSS.MW']
count_m.index = count_m.index.map(lambda m: calendar.month_abbr[int(m)])
(count_m.sort_values(ascending=True)
        .plot(kind='barh', ax=axes[1],
              title='Major Power Outage Count')) # Highest 10 demand loss count
```

[23]: <matplotlib.axes._subplots.AxesSubplot at 0x7f195368cef0>



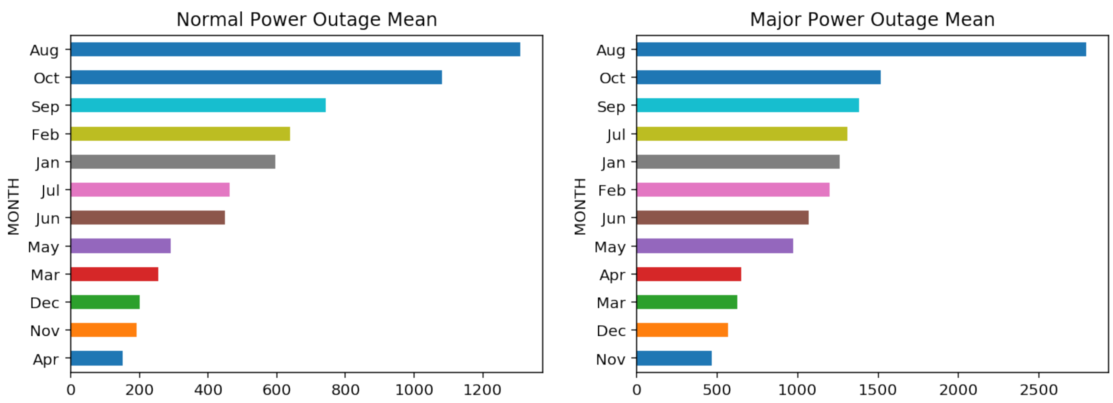
June seems to be the month in which people encounter the highest number of both normal power outage and major power outage.

[ ]:

- Average normal vs. major power outage MW across MONTH

[24]:
```
fig, axes = plt.subplots(1, 2, figsize=(12,4))
mean_loss = outage.groupby('MONTH').mean()['DEMAND.LOSS.MW']
mean_loss.index = mean_loss.index.map(lambda m: calendar.month_abbr[int(m)])
(mean_loss.sort_values(ascending=True)
          .plot(kind='barh', ax=axes[0],
                title='Normal Power Outage Mean')) # Highest 10 mean

mean_loss_m = major_outage.groupby('MONTH').mean()['DEMAND.LOSS.MW']
mean_loss_m.index = mean_loss_m.index.map(lambda m: calendar.month_abbr[int(m)])
(mean_loss_m.sort_values(ascending=True)
            .plot(kind='barh', ax=axes[1],
                  title='Major Power Outage Mean')) # Highest 10 mean
```

[24]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1953598c50>



August seems to be the month in which people encounter the highest average amount of demand loss (MW) for both normal power outage and major power outage.

[ ]:

```

```

[ ]:

```

```

By TIME

- Count number of times normal vs. major power outage occur across TIME

[25]:

```
def get_tile(outage):
    """get the tile of datetime time"""
    qtime_out = outage.copy().set_index('OUTAGE.START') # Deep copy
    tiles = (pd.to_datetime(['00:00:00', '03:59:59', '07:59:59', '11:59:59',
                             '15:59:59', '19:59:59', '23:59:59']).time) # Time␣
↪intervals
    for i in range(len(tiles) - 1): # Get interval category
        lower, upper = tiles[i], tiles[i+1]
        indices = qtime_out.between_time(lower, upper).index
        for idx in indices:
            qtime_out.loc[idx, 'TIME.TILE'] = str(lower) + ' - ' +str(upper)

    outage['TIME.TILE'] = qtime_out['TIME.TILE'].reset_index(drop=True) #␣
↪Append col to outage
    return outage
```

[26]:

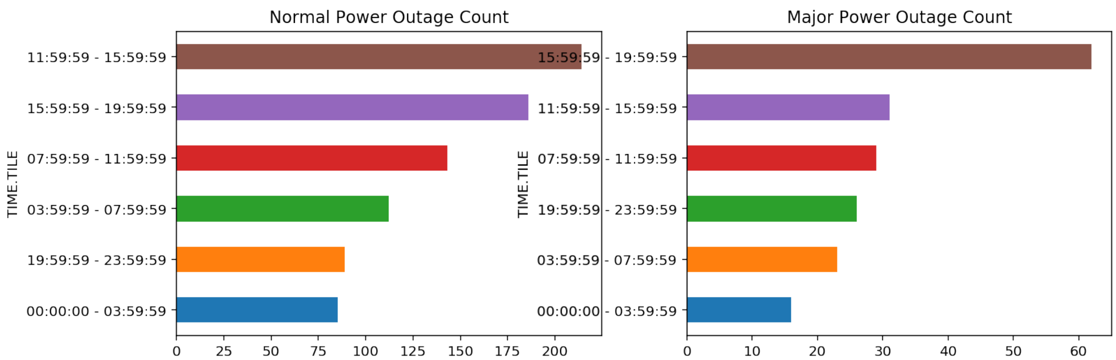
```
outage, major_outage = get_tile(outage), get_tile(major_outage)
```

[27]:

```
fig, axes = plt.subplots(1, 2, figsize=(12,4))
count = outage.groupby('TIME.TILE').count()['DEMAND.LOSS.MW']
(count.sort_values(ascending=True)
```

```
      .plot(kind='barh', ax=axes[0],
            title='Normal Power Outage Count')) # Highest 10 demand loss count

count_m = major_outage.groupby('TIME.TILE').count()['DEMAND.LOSS.MW']
(count_m.sort_values(ascending=True)
       .plot(kind='barh', ax=axes[1],
             title='Major Power Outage Count')) # Highest 10 demand loss count
```

[27]: <matplotlib.axes._subplots.AxesSubplot at 0x7f195340c048>



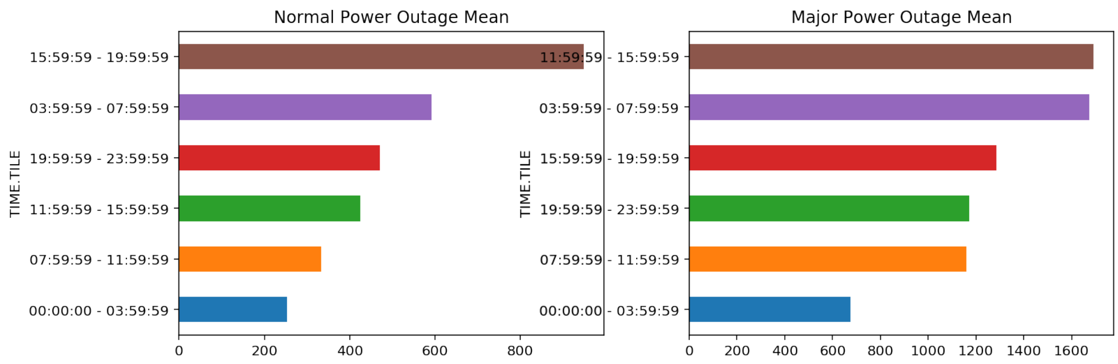
People tend to encounter normal power outage from 12pm - 4pm, but they tend to encounter major power outage from 4pm - 7pm.

[ ]:

- Average normal vs. major power outage MW across TIME

[28]:
```
fig, axes = plt.subplots(1, 2, figsize=(12,4))
mean_loss = outage.groupby('TIME.TILE').mean()['DEMAND.LOSS.MW']
(mean_loss.sort_values(ascending=True)
          .plot(kind='barh', ax=axes[0],
                title='Normal Power Outage Mean')) # Highest 10 mean

mean_loss_m = major_outage.groupby('TIME.TILE').mean()['DEMAND.LOSS.MW']
(mean_loss_m.sort_values(ascending=True)
            .plot(kind='barh', ax=axes[1],
                  title='Major Power Outage Mean')) # Highest 10 mean
```

[28]: <matplotlib.axes._subplots.AxesSubplot at 0x7f195335a588>



When encountered normal power outage, those in 4pm - 7pm have the highest average amount of demand loss. But when encountered major power outage, those in 12pm - 4pm have the highest average amount of demand loss.

[ ]:

```

```

[ ]:

```

```

[ ]:

```

```

3.2.3 3) Correlation of characteristics with outage severity

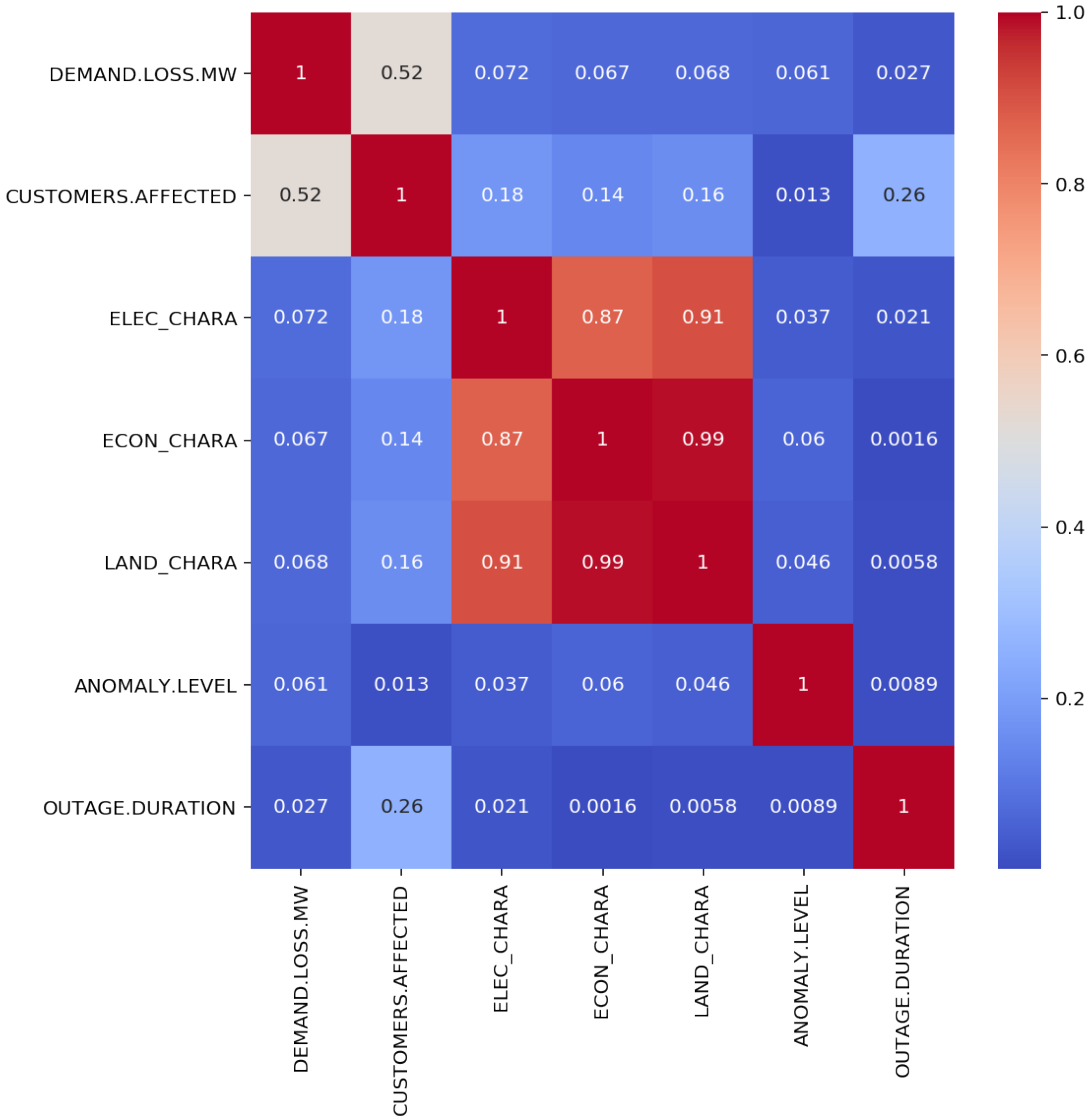
land-use characteristics, electricity consumption patterns, economic characteristics, and Climate anomaly level

[382]:

```
corr_col = ['DEMAND.LOSS.MW', 'CUSTOMERS.AFFECTED',
            'ELEC_CHARA', 'ECON_CHARA', 'LAND_CHARA',
            'ANOMALY.LEVEL', 'OUTAGE.DURATION']
fig, ax = plt.subplots(figsize=(8,8))
sns.heatmap(outage[corr_col].corr(), cmap='coolwarm', annot=True, ax=ax)
```

[382]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x173a5f43128>
```



Outage severity, possibly defined by both Demand Loss and Number of Customers Affected, has the correlation shown above. - Demand Loss and Customers Affected are moderately correalted with each other (r = 0.52) - However, even though the three characteristics (electricity, economic, land-use) are highly correalted with each other, none of them are even moderately correalted with Demand Loss or Number of Customers Affected, which may imply that those three characteristics do not really affect the severity of power outage a lot - Anomaly level, which intuitively thinking would potentially be correlated with severity of power outage, actually shows really weak association with Demand Loss and Number of Customers Affected - Outage duration also shows a weak correlation with Demand Loss and Number of Customers Affected

```
[ ]:
```

[ ]:

[ ]:

3.2.4 4) Characteristics of major outage over time

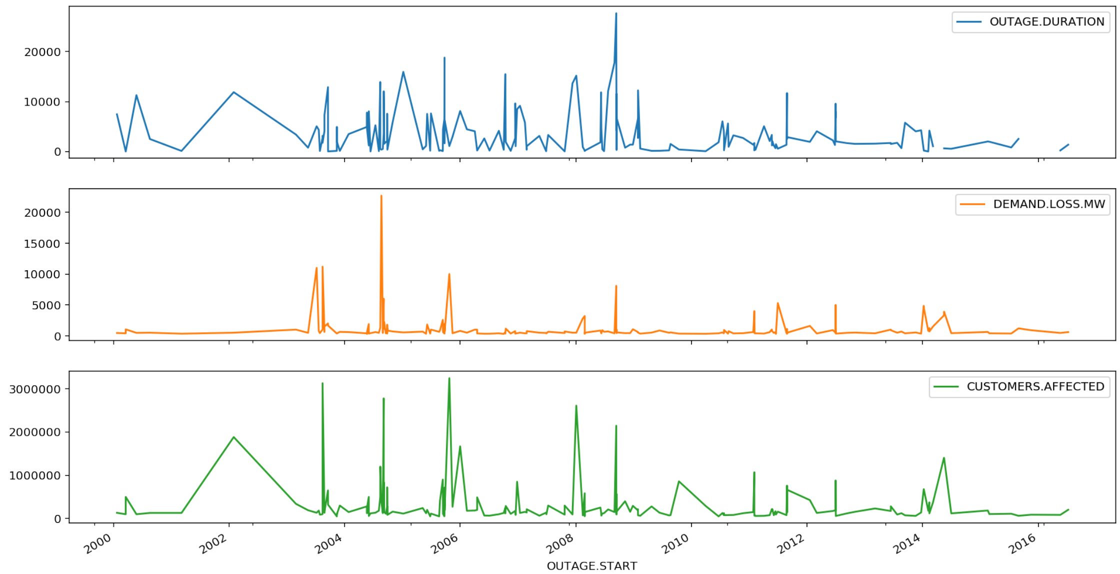
Demand Loss, and Number of Customers Affected over time

[218]:

```
major_outage.plot(subplots=True,
                  x='OUTAGE.START',
                  y=['OUTAGE.DURATION', 'DEMAND.LOSS.MW', 'CUSTOMERS.AFFECTED'],
                  style='-',
                  figsize=(16,9))
```

[218]:

```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x0000017397DE5C88>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x0000017397E0F5C0>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x0000017397E3EA20>],
      dtype=object)
```



Verifies the above correlation that Demand Loss and Number of Customers Affected are moderately correlated with each other. Whenever there is a spike (increase) in Demand Loss, there would usually be an increase in the number of customers affected shown in the graph

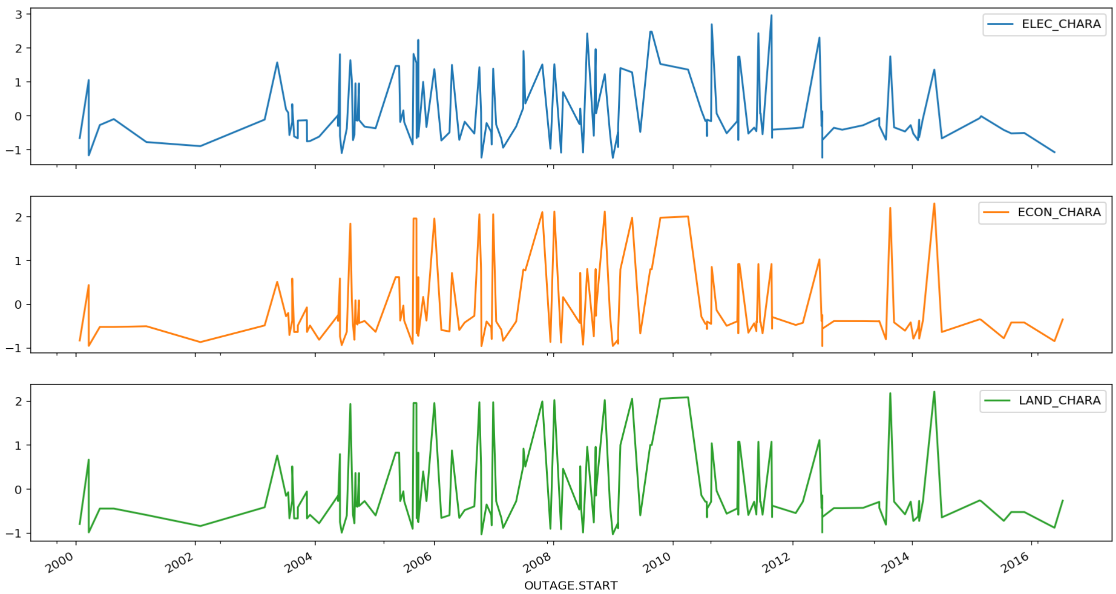
Outage duration does not show highly consistent change with Demand Loss and Number of Customers Affected, which can be verified in the above heat map too.

[ ]:

Land-use characteristics, electricity consumption patterns, and economic characteristics over time

[219]:
```
major_outage.plot(subplots=True,
                  x='OUTAGE.START',
                  y=['ELEC_CHARA','ECON_CHARA', 'LAND_CHARA'],
                  style='-',
                  figsize=(16,9))
```

[219]:
```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x0000017397F65668>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x0000017397F7DF98>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x0000017397FB6438>],
      dtype=object)
```



Electricity, economic, and land-use characteristic show highly consistent changes through time, and this can be verified with the high correaltion between these three characteristics in the correlation heatmap above.

[ ]:

[ ]:

[ ]:

3.3 Assessment of Missingness

[194]:
```python
# Function to check the missingess
def missingness_check(outage, column, title, N=1000):
    """check the missingness of columns on all other cols in outage"""
    demand_miss = outage.assign(IS_NULL=outage[column].isnull())
    cols = outage.columns.drop([column]) # Columns to check

    for i in range(len(cols)): # Check missingness
        col = cols[i] # Get the columns

        # Different cols have different functions
        if demand_miss[col].dtype == int or demand_miss[col].dtype == float: #␣
↪Number type
            func = util.diff_in_means # Difference in means
            stats, obs = (util.permutation_test(demand_miss, # Perform␣
↪permutation test
                                                col, 'IS_NULL', # Dep col, check col
                                                func, N)) # Function, trials
            p_val = np.min([np.count_nonzero(np.array(stats) <= obs) / N,
                            np.count_nonzero(np.array(stats) >= obs) / N]) #␣
↪P-value

        elif demand_miss[col].dtype == object: # String Type
            func = util.tvd # Total Variation Distance
            stats, obs = (util.permutation_test(demand_miss, # Perform␣
↪permutation test
                                                col, 'IS_NULL', # Dep col, check col
                                                func, N)) # Function, trials
            p_val = np.count_nonzero(np.array(stats) >= obs) / N # P-value

        else: # Datetime continuous category type
            func = util.ks # KS statistic
            stats, obs = (util.permutation_test(demand_miss, # Perform␣
↪permutation test
                                                col, 'IS_NULL', # Dep col, check col
                                                func, N)) # Function, trials
            p_val = np.count_nonzero(np.array(stats) >= obs) / N # P-value

        if p_val < 0.05:
            # util.plot_distribution(stats, obs, i, title, col, p_val) # Plot␣
↪distribution
            print(title + col + ', p_val is ' + str(p_val))
    return
```

3.3.1 1) Missingness of Demand Loss

[233]:
```
# Missingness of Demand Loss
missingness_check(outage, 'DEMAND.LOSS.MW', 'Demand Loss Dependence on ')
```

```
Demand Loss Dependence on YEAR, p_val is 0.0
Demand Loss Dependence on U.S._STATE, p_val is 0.0
Demand Loss Dependence on POSTAL.CODE, p_val is 0.0
Demand Loss Dependence on NERC.REGION, p_val is 0.0
Demand Loss Dependence on CLIMATE.REGION, p_val is 0.0
Demand Loss Dependence on ANOMALY.LEVEL, p_val is 0.014
Demand Loss Dependence on CAUSE.CATEGORY, p_val is 0.0
Demand Loss Dependence on OUTAGE.START, p_val is 0.0
Demand Loss Dependence on OUTAGE.RESTORATION, p_val is 0.0
Demand Loss Dependence on ELEC_CHARA, p_val is 0.001
Demand Loss Dependence on ECON_CHARA, p_val is 0.0
Demand Loss Dependence on LAND_CHARA, p_val is 0.0
```

Demand Loss (MW) is MAR dependent on Year, State, Climate Regions, Anomaly levels, Cause Category, Outage start/restoration, Electricity consumption, Economic characteristic, and Land-use characteristic.

[ ]:
```
```

3.3.2 2) Missingness of Customers Affected

[234]:
```
# Missingness of Customers Affected
missingness_check(outage, 'CUSTOMERS.AFFECTED', 'Affected Customers Dependence␣
↪on ')
```

```
Affected Customers Dependence on YEAR, p_val is 0.0
Affected Customers Dependence on MONTH, p_val is 0.0
Affected Customers Dependence on U.S._STATE, p_val is 0.0
Affected Customers Dependence on POSTAL.CODE, p_val is 0.0
Affected Customers Dependence on NERC.REGION, p_val is 0.0
Affected Customers Dependence on CLIMATE.REGION, p_val is 0.0
Affected Customers Dependence on ANOMALY.LEVEL, p_val is 0.005
Affected Customers Dependence on CAUSE.CATEGORY, p_val is 0.0
Affected Customers Dependence on OUTAGE.DURATION, p_val is 0.006
Affected Customers Dependence on DEMAND.LOSS.MW, p_val is 0.0
Affected Customers Dependence on OUTAGE.START, p_val is 0.0
Affected Customers Dependence on OUTAGE.RESTORATION, p_val is 0.0
Affected Customers Dependence on ECON_CHARA, p_val is 0.032
Affected Customers Dependence on TIME.TILE, p_val is 0.0
```

Number of Customers Affected is MAR dependent on Year, Month, State, Climate Regions, Anomaly levels, Cause Category, Demand Loss (MW), Outage start/restoration, and Economic characteristic.

[ ]:

```

```

[ ]:

```

```

3.4 Hypothesis Test

[366]:

```
# Functions to be used
# Total Variation Distance (TVD)
def total_variation_distance(dist1, dist2):
    '''Given two empirical distributions,
    both sorted with same categories, calculates the TVD'''
    return np.sum(np.abs(dist1 - dist2)) / 2

def permutation_test(value, N):
    """Conduct permutation test"""
    obs_count = (cause_out
        .pivot_table(
        index='CAUSE.CATEGORY',
        columns='MAJOR',
        values=value,
        aggfunc='mean',
        fill_value=0
    ))
    obs = total_variation_distance(obs_count[True], obs_count[False])

    tvds = []
    for i in range(N):
        shuffled_maj = (
            cause_out['MAJOR']
            .sample(replace=False, frac=1)
            .reset_index(drop=True)
        ) # Shuffle MAJOR column

        shuffled_major = cause_out.assign(**{'shuffled major': shuffled_maj}).
↪drop(columns='MAJOR') # Assign to dataframe

        value_counts = (shuffled_major
            .pivot_table(
            index='CAUSE.CATEGORY',
            columns='shuffled major',
            values=value,
            aggfunc='mean',
        ))
```

```
        test_stat = total_variation_distance(value_counts[True],␣
↪value_counts[False]) # Test statistic
        tvds.append(test_stat) # Append to lst
        p_val = np.count_nonzero(np.array(tvds) >= obs) / N

    return np.array(tvds), obs, p_val

def plot_distribution(stats, obs):
    """Plot distributions"""
    pd.Series(stats).hist(bins = 10, alpha = 0.5)
    plt.scatter(obs, 0, s=25, c='r', zorder=10)# tvds
    return
```

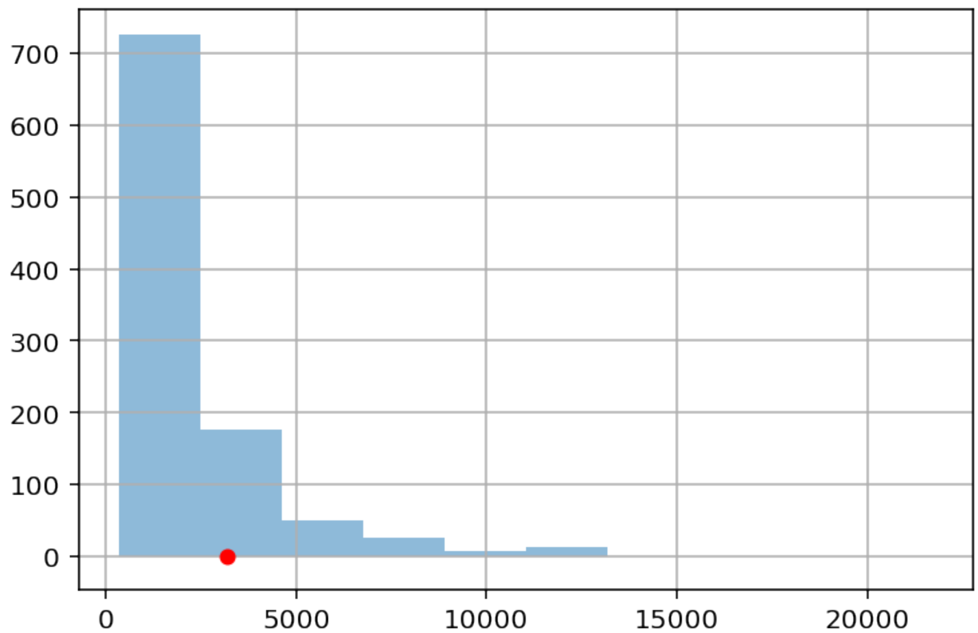
[367]:
```
cause_out = (outage.assign(MAJOR=((outage['DEMAND.LOSS.MW'] > 300) & # Firm␣
↪load loss > 300MW
                                  (outage['CUSTOMERS.AFFECTED'] > 50000))) #␣
↪Customers > 50000
             [['CAUSE.CATEGORY', 'MAJOR', 'DEMAND.LOSS.MW', 'CUSTOMERS.
↪AFFECTED', 'OUTAGE.DURATION']]) # Select needed column
cause_out.head()
```

[367]:
```
        CAUSE.CATEGORY  MAJOR  DEMAND.LOSS.MW  CUSTOMERS.AFFECTED  \
0       severe weather  False             NaN             70000.0
1  intentional attack  False             NaN                 NaN
2       severe weather  False             NaN             70000.0
3       severe weather  False             NaN             68200.0
4       severe weather  False           250.0            250000.0

   OUTAGE.DURATION
0           3060.0
1              1.0
2           3000.0
3           2550.0
4           1740.0
```

3.4.1 Permutation Test 1 - Demand Loss

- **Question**: Is the cause category Demand Loss of major power outage similar to that of non-major power outage?
- **Null hypothesis**: Demand Loss of cause category for both groups come from the same distribution.
- **Alternative hypothesis**: Demand Loss of cause category are different among both groups.
- **Test-statistics**: Total Variation Distance.

[368]:
```
stats1, obs1, p_val1 = permutation_test('DEMAND.LOSS.MW', 1000)
plot_distribution(stats1, obs1)
```



[369]:
```
p_val1
```

[369]: 0.181

p value > 0.05. Fail to reject the null hypothesis that Demand Loss of cause category for both groups come from the same distribution.

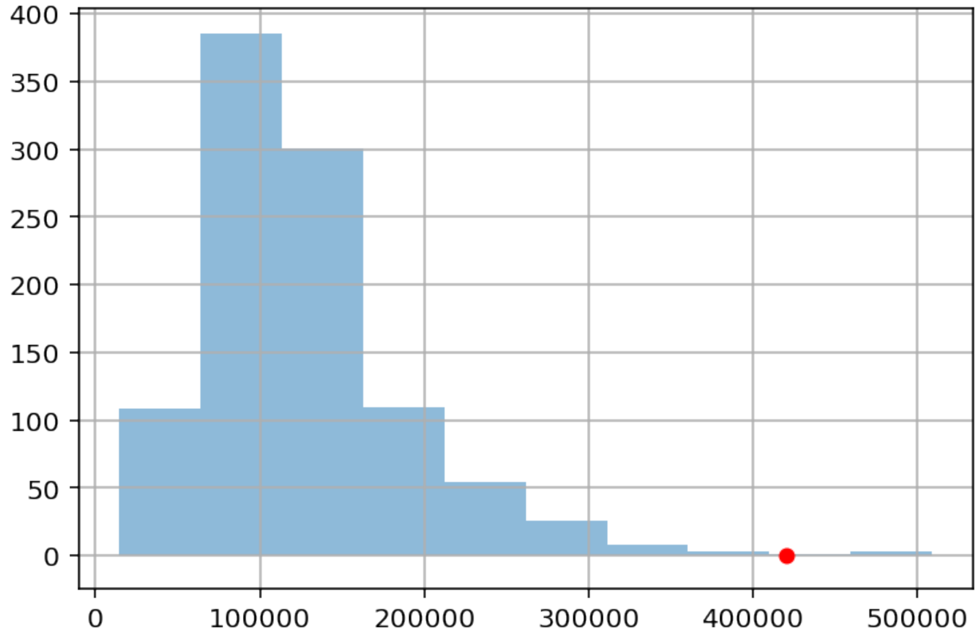
[ ]:
```

```

[ ]:
```

```

3.4.2 Permutation Test 2 - Customers Affected

- **Question**: Is the cause category Customer Affected of major power outage similar to that of non-major power outage?
- **Null hypothesis**: Customer Affected of cause category for both groups come from the same distribution.
- **Alternative hypothesis**: Customer Affected of cause category are different among both groups.
- **Test-statistics**: Total Variation Distance.

[370]:
```
stats2, obs2, p_val2 = permutation_test('CUSTOMERS.AFFECTED', 1000)
plot_distribution(stats2, obs2)
```



[371]:
```
p_val2
```

[371]: 0.004

p value < 0.05. Reject the null hypothesis that Number of Customers Affected of cause category for both groups come from the same distribution.

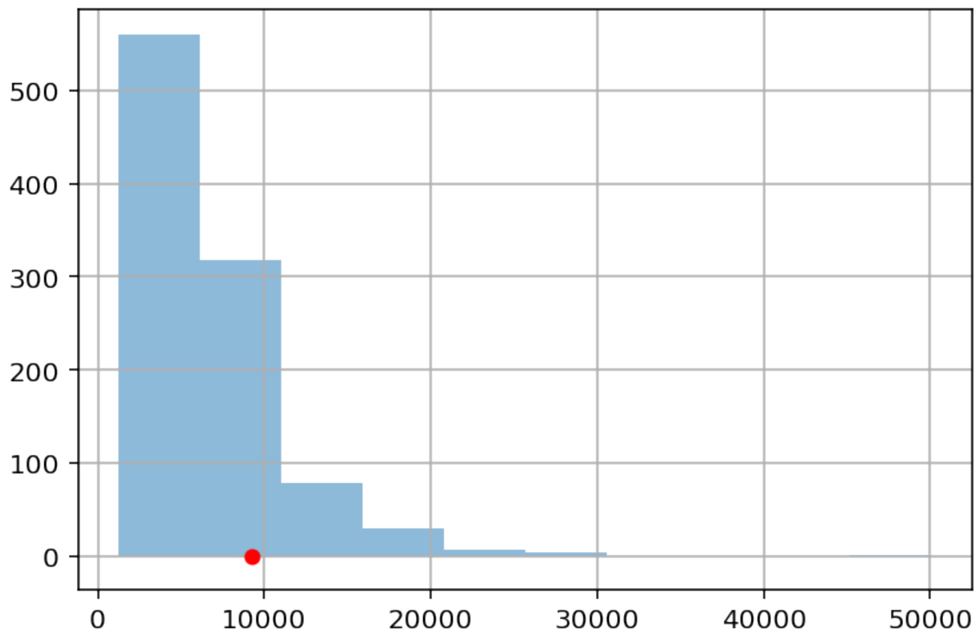
[ ]:
```
```

[ ]:
```
```

3.4.3 Permutation Test 3 - Outage Duration

- **Question**: Is the cause category Outage Duration of major power outage similar to that of non-major power outage?
- **Null hypothesis**: Outage Duration of cause category for both groups come from the same distribution.
- **Alternative hypothesis**: Outage Duration of cause category are different among both groups.
- **Test-statistics**: Total Variation Distance.

[372]:
```
stats3, obs3, p_val3 = permutation_test('OUTAGE.DURATION', 1000)
plot_distribution(stats3, obs3)
```



[373]:
```
p_val3
```

[373]: 0.177

p value > 0.05. Fail to reject the null hypothesis that Outage duration of cause category for both groups come from the same distribution.

[ ]:
```
```