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?

Getting the Data

The data is downloadable [here (https://engineering.purdue.edu/LASCI/research-data/outages/outagerisks)](https://engineering.purdue.edu/LASCI/research-data/outages/outagerisks).

A data dictionary is available at this [article (https://www.sciencedirect.com/science/article/pii/S2352340918307182)](https://www.sciencedirect.com/science/article/pii/S2352340918307182) under *Table 1. Variable descriptions*.

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 (https://python-visualization.github.io/folium/)](https://python-visualization.github.io/folium/) or another geospatial plotting library.

Assessment of Missingness

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

Hypothesis Test

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

Summary of Findings

Introduction

Have you ever wondered if there was a reason why massive power outages in your community happen? Is it related to the weather? Too many people using the electricity? Maybe your region has poor electricity maintenance! In this project my partner and I attempt to tackle these questions. We utilize the dataset on these power outage events that dates back from January 2000 to July 2016.

The dataset presents a multitude of variables and other valuable information that can help in future research on power outages, and possiblities on how they can be prevented. The dataset has 1540 rows, pertaining to a power outage and 57 columns of variables pertaining to that specific power outage. If you didn't know the prevalence of this issue, this number should astonish you, as this dataset includes hundreds of outages that involve a power outage that impacted over 50,000 customers and deprived a demand of 300 megawatts of electricity. The amount of people who are affected by these outages are even higher. We narrowed down the relevant columns to the following:

YEAR : The year of the specific power outage

MONTH : The month of the specific power outage

U.S._STATE : The state where the power outage took place in

POSTAL.CODE : Represents the postal code of the U.S. states

NERC.REGION : The North American Electric Reliability Corporation (NERC) regions involved in the outage event

CLIMATE.REGION : U.S. Climate regions as specified by National Centers for Environmental Information (nine climatically consistent regions in continental U.S.A.)

ANOMALY.LEVEL : This represents the oceanic El Niño/La Niña (ONI) index referring to the cold and warm episodes by season. It is estimated as a 3-month running mean of ERSST.v4 SST anomalies in the Niño 3.4 region (5°N to 5°S, 120–170°W)

CLIMATE.CATEGORY : This represents the climate episodes corresponding to the years. The categories—“Warm”, “Cold” or “Normal” episodes of the climate are based on a threshold of ± 0.5 °C for the Oceanic Niño Index (ONI)

OUTAGE.START.DATE : This variable indicates the day of the year when the outage event started (as reported by the corresponding Utility in the region)

OUTAGE.START.TIME : This variable indicates the time of the day when the outage event started (as reported by the corresponding Utility in the region)

OUTAGE.RESTORATION.DATE : This variable indicates the day of the year when power was restored to all the customers (as reported by the corresponding Utility in the region)

OUTAGE.RESTORATION.TIME : This variable indicates the time of the day when power was restored to all the customers (as reported by the corresponding Utility in the region)

CAUSE.CATEGORY : Categories of all the events causing the major power outages

CAUSE.CATEGORY.DETAIL : Detailed description of the event categories causing the major power outage

HURRICANE.NAMES : If the outage is due to a hurricane, then the hurricane name is given by this variable

```
OUTAGE.DURATION : Duration of outage events (in minutes)

DEMAND.LOSS.MW  : Amount of peak demand lost during an outage event
(in Megawatt) [but in many cases, total demand is reported]

CUSTOMERS.AFFECTED : Number of customers affected by the power outa
ge event
```

The main questions that we hope to answer through an analyses of this dataset are:

```
What are the characteristics of major power outages of higher sever
ity?
Does the climate category affect whether an outage is classified as
major?
```

Cleaning and EDA

To explain our data cleaning and EDA process, we will break it down into steps:

1. We first loaded the excel file and narrowed down the columns to only the relevant ones to our analyses and questions.
2. After taking a look at the data types for the columns, we wanted to add certain columns and clean some of the existing ones up. We accomplish this with our data_cleaning function.
3. Once we finished the data cleaning process we moved onto the univariate, bivariate, and aggregate analyses.
4. Univariate Analysis: We first wrote a function to compute the following univariate statistics in table form:

   ```
   Min year
   Max year
   Unique states
   Unique states
   Unique climate regions
   Unique climate categories
   Most common outage start date/time
   Most common outage restoration date/time
   Max outage duration
   Min outage duration
   Average outage duration
   Min customers affected
   Max customers affected
   Average customers affected
   ```

   There weren't that many meaningful insights that we could get from this table, but we noticed that there were cases where 0 customers were affected, which is odd given that these major outages mean a minimum of 50000 customers were affected. How could only 0 customers be affected? We addres this issue later on in our missingness section. We were interested in determining the proportion of these major outages grouped by climate category, climate region, and season. We noticed that there was a higher proportion of normal outages, with nearly half of them being normal. However, we want to note that these numbers could just be because more of the US consists of normal climates. When grouped by Climate Region we noticed that the Northeast region had a large proportion of outages, followed by the South, West, and Central regions. We surmise that the population and urban density in these areas possibly had an effect on this. Finally, we noticed that in the summer and winter there is a higher proportion of outages. This is interesting to us because our hypothesis ties into if climate has an effect on the severity of an outage. From this number it is possible that the more extreme weather conditions in the summer and winter affect the outage severity.

5. Bivariate Analysis

   To help us answer the question of how a climate can affect the severity we make scatter plots of the Season vs (Outage Duration and Demand Loss). In Season vs Outage Duration, the spread of these numbers across the seasons were relatively similar and hard to determine any major differences, albeit there were a few outliers. In Season vs Demand we could see that there was larger spread of Demand Loss in the summer. Finally, we plotted the metrics of severity with each other and calculated the correlation coefficient to determine if there was a relationship between the severities. We notice here that there is somewhat of a positive association between the demand loss and the customers affected, and between outage duration and customers affected. However, these correlation coefficients are all relatively small, so when determining severity of an outage it might be a better alternative to develop a model that places weights using all three of these metrics, as there isn't really one metric between the three that is best.

6. Aggregate Analysis

   In the aggregate analysis we wanted to determine if there were large differences between the averages of the three main measures of severity when aggregated across State, Climate, Year, and Cause Category. We made groupbys for each of the following aggregations and solved for the mean, min and max of each of the aggregations. We then generated bar plots to help us visualize these differences. There weren't many meaningful insights to be drawn from the state, cause category, and year aggregations, since there wasn't a clear trend for us to see. However, we noticed that warm climates tend to be very high in the severity metrics. It was the highest for average duration of outage, second highest for average customers affected, and highest for demand loss. From here, we wanted to see if the climate category would have a meaningful effect on the severity of a power outage.

Assessment of Missingness

For our data set we decided to assess the missingness of the "CUSTOMERS.AFFECTED" column. We did not believe that this column was 'Not Missing at Random', thinking that the column could potentially be MAR dependent on "OUTAGE.DURATION" and "DEMAND.LOSS.MW" and not MAR dependent on "ANOMALY.LEVEL". Our thought process behind the missing of the customers affected being MAR dependent on demand loss and outage duration resulted from observation. We noticed that the customers affected column often was missing for power outages with a low duration and low demand loss. We thought that this might potentially mean that the customers affected were not reported for power outages that did last long and did not affect demand by much. We also figured that anomaly level and the customers affected were most likely not related in any way in regards to missingness.

In our order to test our theories we ran three permutation tests using the ks statistic. In our first test we tested if the missingness of customers affected was MAR dependent on outage duration. After running 500 repetitions of the test we calculated a pval of 0. In our second test we tested if the missingness of customers affected was MAR dependent on demand loss and similarly got a pval of 0. These results give us reason to believe that the distributions in the two tests were not the same, supporting our theory that the missingness of customers affected was in fact MAR dependent on outage duration and demand loss. In our last we tested if the missingness of customers affected was MAR dependent on anomaly level. After running 500 repetitions of the test we calculated a pval of 0.432. This result differed from our previous two - we failed to reject the null hypothesis that the distributions were the same, leading us to conclude that the missingness of customers affected is most likely not affected by anomaly level.

Hypothesis Test

For our hypothesis test we wanted to look deeper into the relationship between climate category and the severity of power outages. One might believe that the type of climate has an affect on how ‘severe’ a power outage is. For instance, snow storms (‘cold’ climate) might foster longer power outages or tropical storms (‘warm’ climate) might also lead to long power outages. In order to define severity we tried to look at whether a particular power outage was defined as ‘Major’ or not. 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 at least 300MW. We performed a permutation test to see if in our cleaned data set, the distribution of climate category among those power outages that were classified as 'major' is the same as among those that were not classified as 'major'. Our null and alternative hypotheses were defined as the following:

Null: In the US power outages data set, the distribution of climate category among those power outages that were classified as 'major' is the same as among those that were not classified as 'major'. The difference between the two samples is due to chance.

Alternative: In the US power outages data set, the distribution of climate category among the two groups of power outages is different.

To do this we first looked at our observed distribution of power outages in each climate category conditional on whether the power outage was defined as ‘major’ or not major. We noticed there was a slight difference in the number of major vs non major power outages in each climate. We ran our simulation to see if this difference was just due to noise. For our test statistic we used the TVD and set a significance level of 5%.

After running our simulation 1000 times we computed a pval of 0.206. We failed to reject our null hypothesis at the 5% significance level. We could not conclude that there was a significant difference in the classification of 'major' power outages vs. non 'major' power outages in different climate categories. The results of this test tell us that climate might not be the best factor to consider when determining where the next major power outages will occur for a power company.

Code

In [3]:
```
import matplotlib.pyplot as plt
import numpy as np
import os
import pandas as pd
import seaborn as sns
from scipy.stats import pearsonr
sns.set_theme(style= 'whitegrid')
%matplotlib inline
%config InlineBackend.figure_format = 'retina'  # Higher resolution figures
```

Cleaning and EDA

In [4]:
```
#We first read in the file from excel and keep the columns relevant to our
def read_data(fp):
    """
    This is a function that reads in the Excel file with outages data.
    Only the appropriate rows/columns are taken in.
    fp = os.path.join('data', 'outage.xlsx')
    """

    #reads the excel file
    data = pd.read_excel(fp, header = 5, usecols = "B:T")

    # drop the first row with the descriptions
    data = data.drop([0])

    return data
```

In [7]:
```
fp = os.path.join('data', 'outage.xlsx')
data = read_data(fp).head()
```

In [8]:
```
#checking the dtata types of the relevant columns
data.dtypes
```

Out[8]:
```
OBS                           float64
YEAR                          float64
MONTH                         float64
U.S._STATE                     object
POSTAL.CODE                    object
NERC.REGION                    object
CLIMATE.REGION                 object
ANOMALY.LEVEL                  object
CLIMATE.CATEGORY               object
OUTAGE.START.DATE              object
OUTAGE.START.TIME              object
OUTAGE.RESTORATION.DATE        object
OUTAGE.RESTORATION.TIME        object
CAUSE.CATEGORY                 object
CAUSE.CATEGORY.DETAIL          object
HURRICANE.NAMES                object
OUTAGE.DURATION                object
DEMAND.LOSS.MW                 object
CUSTOMERS.AFFECTED            float64
dtype: object
```

In [11]:
```
## helper function to convert into seasons
def season_helper(obs):
    """
    Helper functions converts month into
    respective season.
    """
    if obs >= 3 and obs <= 5:
        return 'Spring'
    elif obs >= 6 and obs <= 8:
        return 'Summer'
    elif obs >= 9 and obs <= 11:
        return 'Fall'
    elif pd.isnull(obs):
        return np.NaN
    else:
        return 'Winter'
def data_cleaning(data):
    """
    Takes in a table like the one produced by read_data(fp) and
    does the necessary stuff to clean the data set. This includes:
    -> combining the outage time/date columns into one
    -> combining the restoration time/date columns into one
    -> type casting the values in columns to their proper types
    (year needs to be an int, month should be an int, customers affected sh
    be an int because you can't have half a customer)
    """
    data_copy = data.copy(deep = True)

    # combine outage start time/date
    data_copy["OUTAGE.START.DATE"] = pd.to_datetime(data_copy["OUTAGE.START
    data_copy["OUTAGE.START.TIME"] = pd.to_timedelta(data_copy["OUTAGE.STAR
    data_copy["OUTAGE.START"] = data_copy["OUTAGE.START.DATE"] +  data_copy
    data_copy = data_copy.drop(columns = ["OUTAGE.START.DATE","OUTAGE.START

    # combine outage restoration time/date
    data_copy["OUTAGE.RESTORATION.DATE"] = pd.to_datetime(data_copy["OUTAGE
    data_copy["OUTAGE.RESTORATION.TIME"] = pd.to_timedelta(data_copy["OUTAG
    data_copy["OUTAGE.RESTORATION"] = data_copy["OUTAGE.RESTORATION.DATE"]
    data_copy = data_copy.drop(columns = ["OUTAGE.RESTORATION.DATE", "OUTAG

    # cast values to 'correct' types + set index
    data_copy = data_copy.set_index('OBS')
    data_copy["YEAR"] = data_copy["YEAR"].astype(int)
    data_copy["OUTAGE.DURATION"] = data_copy["OUTAGE.DURATION"].astype(floa
    data_copy['DEMAND.LOSS.MW'] = data_copy['DEMAND.LOSS.MW'].astype(float)
    #add on a season column based on the month
    data_copy['SEASON'] = data_copy['MONTH'].apply(season_helper)


    data_copy = data_copy.reset_index().drop(columns = ['OBS'])

    return data_copy
```

In [12]:
```
# now let us read in the clean data
fp = os.path.join('data', 'outage.xlsx')
data = read_data(fp)
clean_data = data_cleaning(data)
clean_data.head()
```

Out[12]:

	YEAR	MONTH	U.S._STATE	POSTAL.CODE	NERC.REGION	CLIMATE.REGION	ANOMALY.LEVEL
0	2011	7.0	Minnesota	MN	MRO	East North Central	-0.3
1	2014	5.0	Minnesota	MN	MRO	East North Central	-0.1
2	2010	10.0	Minnesota	MN	MRO	East North Central	-1.5
3	2012	6.0	Minnesota	MN	MRO	East North Central	-0.1
4	2015	7.0	Minnesota	MN	MRO	East North Central	1.2

Univariate Analysis

In [13]:
```
# function to compute univariatet stats
def compute_univariate_stats(data):
    """
    Takes in the cleaned data and computes some
    univariate statistics and returns them in a dataframe.
    -> min year in data set
    -> max year in data set
    -> # unique states
    -> # unique climate regions
    -> # unique climate categories
    -> most common outage start date/time
    -> most common outage restoration date/time
    -> max outage duration
    -> min outage duration
    -> average outage duration
    -> min customers affected
    -> max customers affected
    -> average customers affected
    """
    data_copy = data.copy(deep = True)

    min_year = data_copy["YEAR"].min()
    max_year = data_copy["YEAR"].max()

    numunique_states = data_copy["U.S._STATE"].nunique()
    numunique_climregions = data_copy["CLIMATE.REGION"].nunique()
    numunique_climcats = data_copy["CLIMATE.CATEGORY"].nunique() # normal,

    most_com_start = data_copy["OUTAGE.START"].describe()['top']
    most_com_restoration = data_copy["OUTAGE.RESTORATION"].describe()['top'

    max_outage_dur = data_copy["OUTAGE.DURATION"].max()/60
    min_outage_dur = data_copy["OUTAGE.DURATION"].min()/60
    avg_outage_dur = data_copy["OUTAGE.DURATION"].mean()/60

    max_cust_affected = int(data_copy["CUSTOMERS.AFFECTED"].max())
    min_cust_affected = int(data_copy["CUSTOMERS.AFFECTED"].min())
    avg_cust_affected = int(np.round(data_copy["CUSTOMERS.AFFECTED"].mean()

    table_data = [min_year, max_year, numunique_states, numunique_climregio
                  most_com_start, most_com_restoration, max_outage_dur, min_
                  max_cust_affected, min_cust_affected, avg_cust_affected]

    table_index = ["Min Year", "Max Year", "# Unique States", "# Unique Cli
                   "Most Common Start Date/Time", "Most Common Rest. Date/Ti
                   "Min Outage Duration (hrs)", "Avg Outage Duration (hrs)",
                   "Min # Customers Affected", "Avg # Customers Affected"]

    table = pd.DataFrame(data = table_data, index = table_index, columns =

    return table
```

In [15]:
```
# Here we produced a table of relevant univariate statistics
fp = os.path.join('data', 'outage.xlsx')
data = read_data(fp)
clean_data = data_cleaning(data)
compute_univariate_stats(clean_data)
```

```
<ipython-input-13-51e2fdb76411>:29: FutureWarning: Treating datetime data
as categorical rather than numeric in `.describe` is deprecated and will
be removed in a future version of pandas. Specify `datetime_is_numeric=Tr
ue` to silence this warning and adopt the future behavior now.
  most_com_start = data_copy["OUTAGE.START"].describe()['top']
<ipython-input-13-51e2fdb76411>:30: FutureWarning: Treating datetime data
as categorical rather than numeric in `.describe` is deprecated and will
be removed in a future version of pandas. Specify `datetime_is_numeric=Tr
ue` to silence this warning and adopt the future behavior now.
  most_com_restoration = data_copy["OUTAGE.RESTORATION"].describe()['to
p']
```

Out[15]:

	Univariate Statistics
Min Year	2000
Max Year	2016
# Unique States	50
# Unique Climate Regions	9
# Unique Climate Cats.	3
Most Common Start Date/Time	2010-08-02 12:45:00
Most Common Rest. Date/Time	2010-08-04 11:00:00
Max Outage Duration (hrs)	1810.883333
Min Outage Duration (hrs)	0.0
Avg Outage Duration (hrs)	43.75664
Max # Customers Affected	3241437
Min # Customers Affected	0
Avg # Customers Affected	143456

In [16]:
```
#proportions of outages by climate category
clean_data['CLIMATE.CATEGORY'].value_counts(normalize = True)
```

Out[16]:
```
normal     0.487869
cold       0.310164
warm       0.201967
Name: CLIMATE.CATEGORY, dtype: float64
```

In [17]:
```
#proportions of outages by climate region
clean_data['CLIMATE.REGION'].value_counts(normalize = True)
```

Out[17]:
```
Northeast             0.229058
South                 0.149869
West                  0.142016
Central               0.130890
Southeast             0.100131
East North Central    0.090314
Northwest             0.086387
Southwest             0.060209
West North Central    0.011126
Name: CLIMATE.REGION, dtype: float64
```

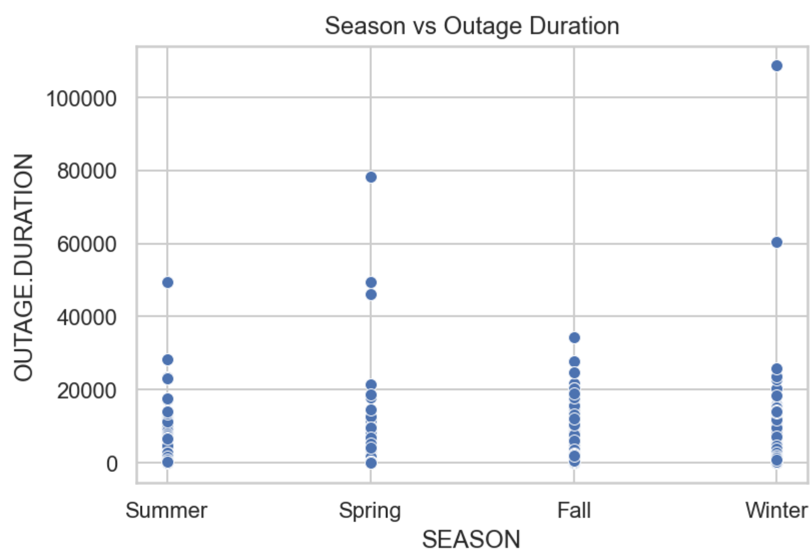
In [18]:
```
#proportions of outages by season
clean_data['SEASON'].value_counts(normalize = True)
```

Out[18]:
```
Summer    0.346885
Winter    0.251148
Spring    0.221639
Fall      0.180328
Name: SEASON, dtype: float64
```

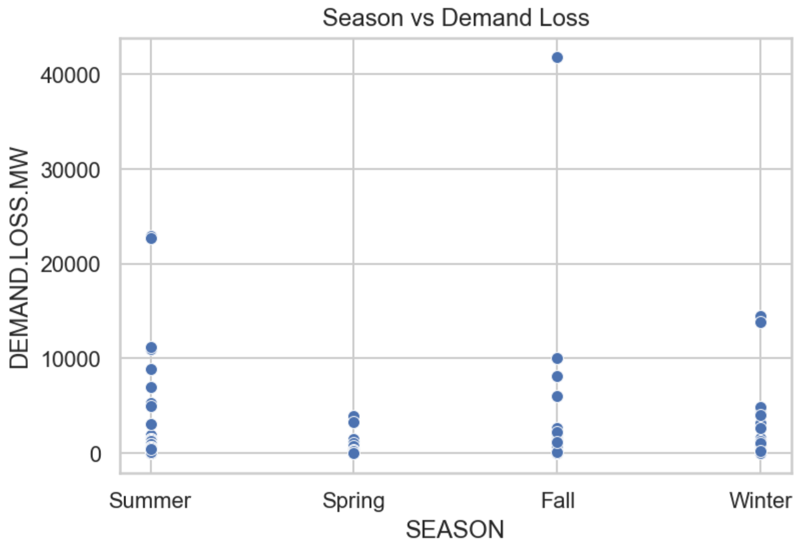
Bivariate Analysis

In [19]:
```
# season vs outage duration scatterplot
season_duration = sns.scatterplot(x = clean_data['SEASON'], y = clean_data[
plt.title('Season vs Outage Duration')
```

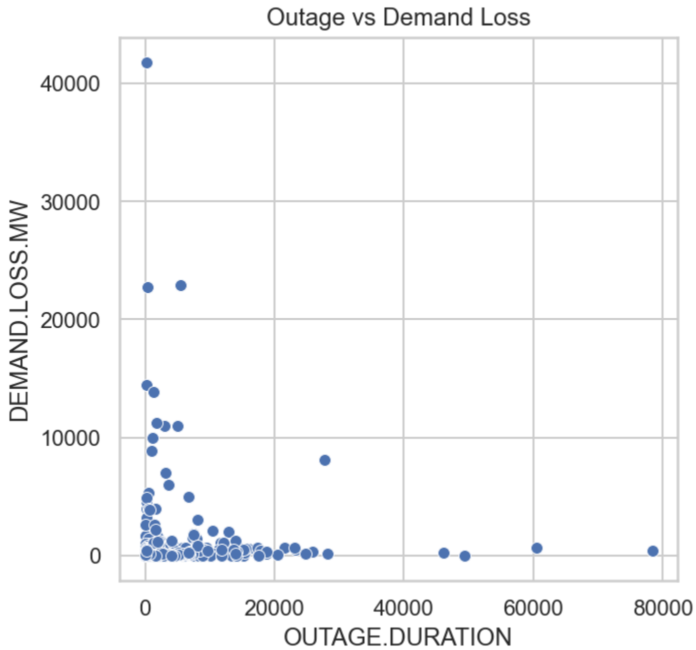
Out[19]:
```
Text(0.5, 1.0, 'Season vs Outage Duration')
```



In [20]:
```
#season vs demand loss plot
season_demand = sns.scatterplot(x = clean_data['SEASON'], y = clean_data['D
plt.title('Season vs Demand Loss')
plt.show()
```

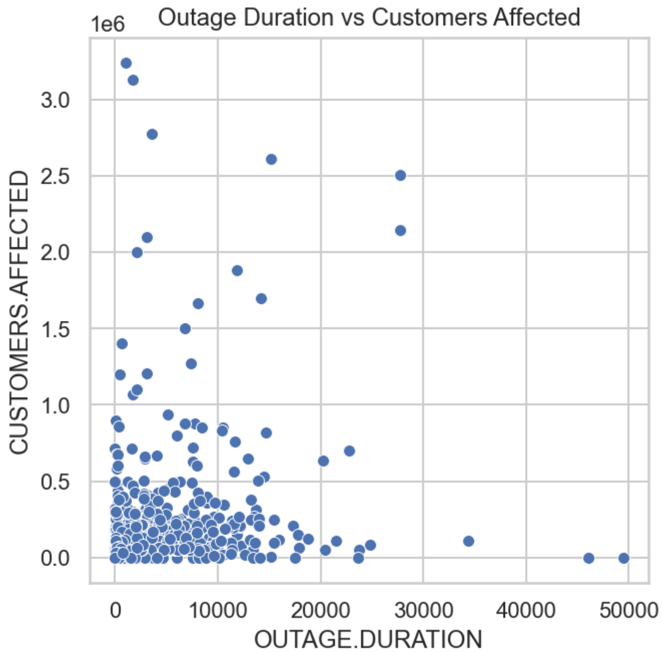


In [21]:
```
# demand loss vs duration to see relationship
plt.figure(figsize = (5,5))
dl_duration = sns.scatterplot(x = clean_data['OUTAGE.DURATION'], y = clean_
plt.title('Outage vs Demand Loss')
plt.show()
corr = pearsonr(clean_data['OUTAGE.DURATION'].fillna(0), clean_data['DEMAND
print('correlation coefficient: ' + str(corr))
```



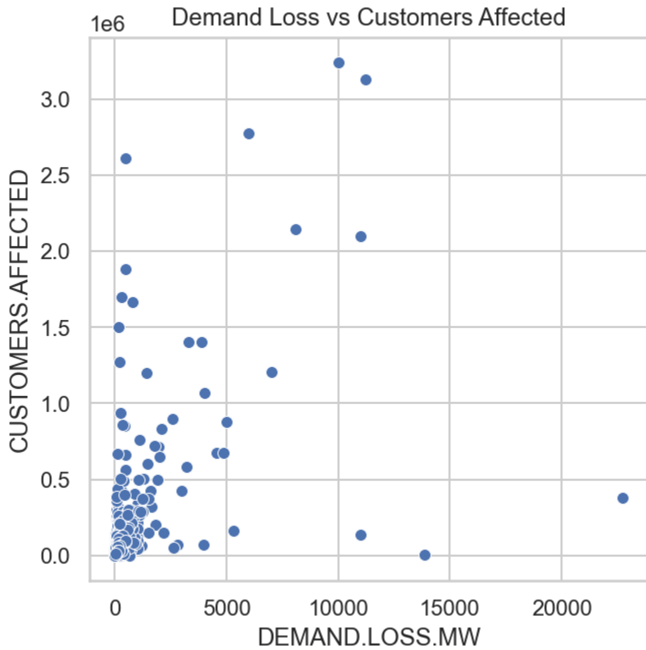

```
correlation coefficient: 0.01862128765201535
```

In [22]:
```
# customers affected vs customers affected scatterplot
plt.figure(figsize = (5,5))
dl_duration = sns.scatterplot(x = clean_data['OUTAGE.DURATION'], y = clean_
plt.title('Outage Duration vs Customers Affected')
plt.show()
corr = pearsonr(clean_data['OUTAGE.DURATION'].fillna(0), clean_data['CUSTOM
print('correlation coefficient: ' + str(corr))
```




```
correlation coefficient: 0.1740290539994629
```

In [23]:
```
# demand loss vs customers affected scatterplot
plt.figure(figsize = (5,5))
dl_duration = sns.scatterplot(x = clean_data['DEMAND.LOSS.MW'], y = clean_d
plt.title('Demand Loss vs Customers Affected')
plt.show()
corr = pearsonr(clean_data['DEMAND.LOSS.MW'].fillna(0), clean_data['CUSTOME
print('correlation coefficient: ' + str(corr))
```




```
correlation coefficient: 0.26770031980190656
```

Aggregate Analysis

State Level

In [28]:
```
# helper function to determine if a power outage is major or not
def detect_major(row):
    if row["CUSTOMERS.AFFECTED"] > 50000:
        return True
    elif row["DEMAND.LOSS.MW"] > 300:
        return True
    else:
        return False
```

In [29]:
```
fp = os.path.join('data', 'outage.xlsx')
data = read_data(fp)
clean_data = data_cleaning(data)
```

In [30]:
```
# map building process
clean_data["Major_Outage"] = clean_data.apply(detect_major, axis = 1)
grouped = clean_data.groupby("U.S._STATE").count()
grouped = grouped.drop(columns = ["YEAR", "MONTH", "POSTAL.CODE", "NERC.REG
                                  "ANOMALY.LEVEL", "CLIMATE.CATEGORY", "CAUS
                                  "CAUSE.CATEGORY.DETAIL", "HURRICANE.NAMES"
                                  "DEMAND.LOSS.MW", "CUSTOMERS.AFFECTED", "O
                                  "SEASON"])
rhode_island = pd.DataFrame(data = {"Major_Outage": 0}, index = ["Rhode Isl
state_data = pd.concat([grouped, rhode_island])
state_data = state_data.drop(index = 'Alaska')
state_data = state_data.drop(index = 'Hawaii')
state_data = state_data.sort_index()
```

In [31]:
```
# map building process
import json

states_geo = 'data/united-states.geojson'

with open(states_geo) as states_file:
    states_json = json.load(states_file)


denominations_json = []

for index in range(len(states_json['features'])):
    denominations_json.append(states_json['features'][index]['properties'][

#denominations_json
```

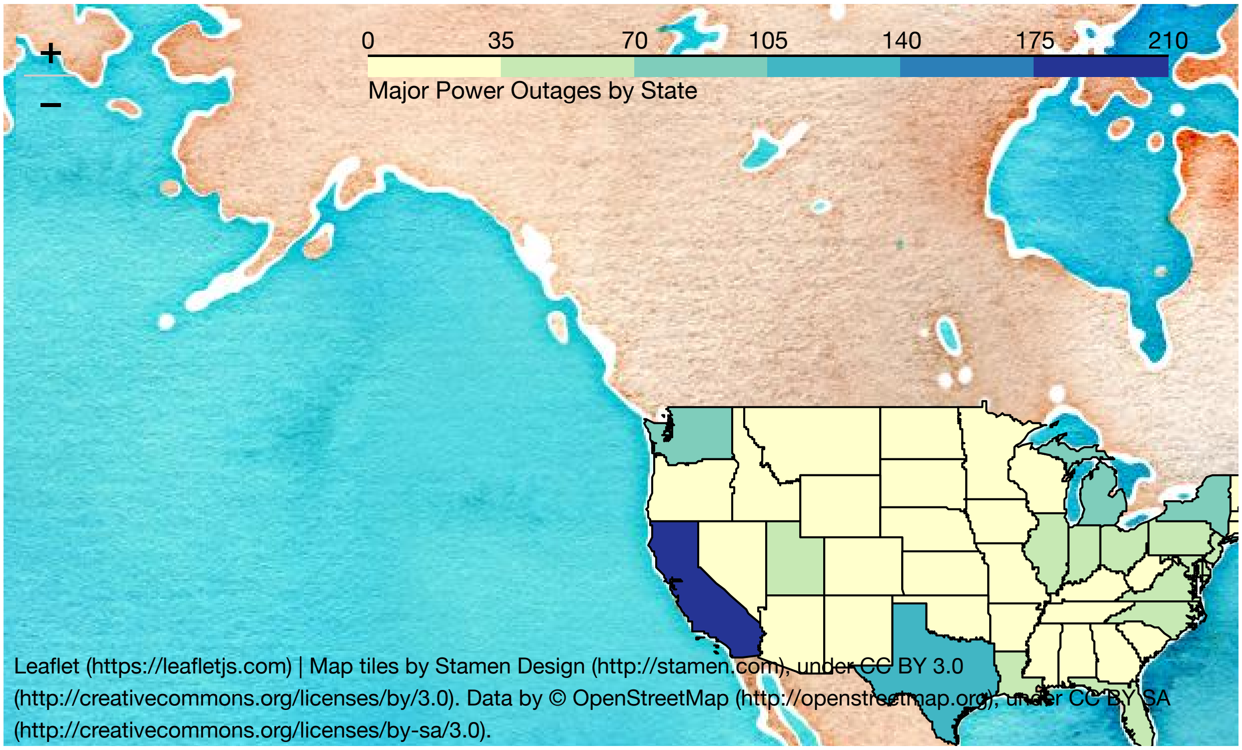
In [32]:
```
# map bulding process
df_names = state_data.index.tolist()
geojson_names = denominations_json

state_data.replace(dict(zip(df_names, geojson_names)), inplace = True)
```

In [33]:
```
state_data = state_data.reset_index()
```

In [38]:
```python
# plot the map ("Number of Major Power Outages by State")
import folium

states_geo = 'data/united-states.geojson'

states_map = folium.Map(location=[32.949683, -117.075116], zoom_start=4, ti
states_map.choropleth(
    geo_data = states_geo,
    data = state_data,
    columns = ["index","Major_Outage"],
    key_on = 'feature.properties.name',
    fill_color = 'YlGnBu',
    fill_opacity = 1,
    line_opacity = 1,
    legend_name = "Major Power Outages by State",
    smooth_factor = 0)

# our beatiful map does not show in the PDF :(
states_map
```

```
/Users/devonromero/Library/Python/3.8/lib/python/site-packages/folium/fol
ium.py:409: FutureWarning: The choropleth  method has been deprecated. In
stead use the new Choropleth class, which has the same arguments. See the
example notebook 'GeoJSON_and_choropleth' for how to do this.
  warnings.warn(
```

Out[38]:



Climate Level

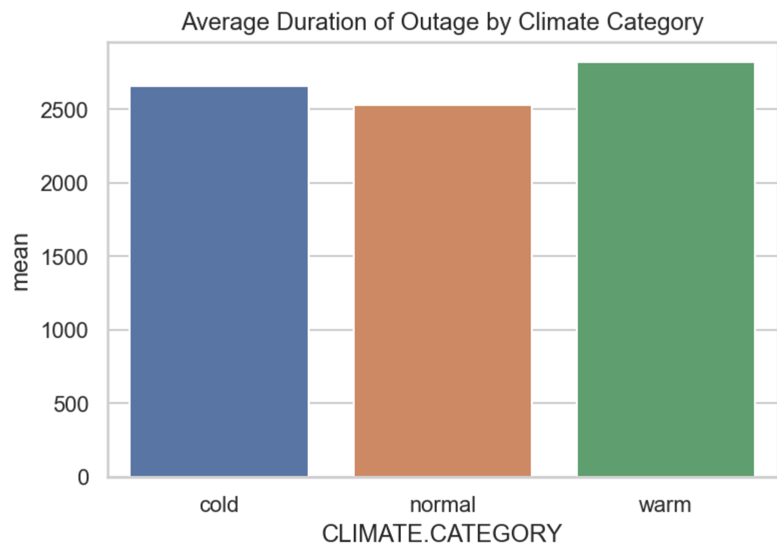
In [25]:
```
def climate_level_eda(data):
    """
    Takes in a cleaned power outages data frame and groups by
    climate category to find the average duration of power outages in each
    Creates a bar chart by region.
    """
    data_copy = data.copy(deep = True)
    data_copy = data_copy.drop(columns = ["YEAR", "MONTH"])
    grouped_data = data_copy.groupby("CLIMATE.CATEGORY").agg(['mean', 'min'

    return grouped_data
```

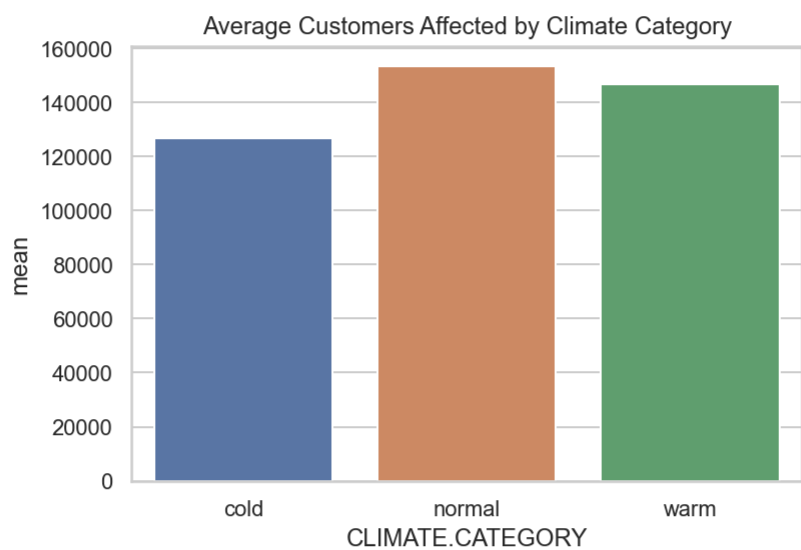
In [26]:
```
by_climates = climate_level_eda(clean_data)
by_climates
```

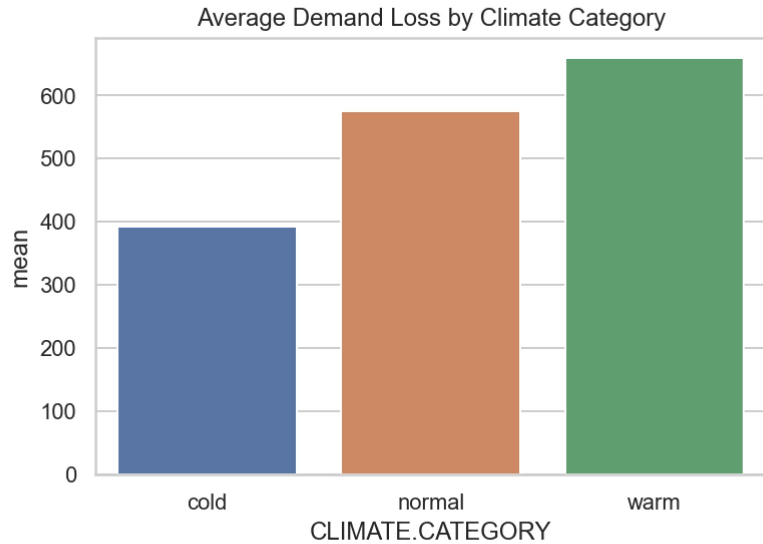
Out[26]:

	OUTAGE.DURATION			DEMAND.LOSS.MW			CUSTOMERS.AFFECTEI		
	mean	min	max	mean	min	max	mean	min	n
CLIMATE.CATEGORY									
cold	2656.956803	0.0	108653.0	391.028000	0.0	14435.0	126840.066869	0.0	2
normal	2530.980822	0.0	78377.0	574.796954	0.0	22934.0	153182.834286	0.0	3
warm	2817.318021	0.0	49427.0	657.854749	0.0	41788.0	146843.895652	0.0	2

In [27]:
```
#We plot the climates by average outage duration and customers affected her

climates_duration = sns.barplot(x = by_climates.index,
                               y = by_climates['OUTAGE.DURATION']['mean'])
plt.title('Average Duration of Outage by Climate Category')
plt.show()
# add one for category by customers, category by demand loss
climates_customer = sns.barplot(x = by_climates.index,
                               y = by_climates['CUSTOMERS.AFFECTED']['mean'
plt.title('Average Customers Affected by Climate Category')
plt.show()
#demand loss by climate category
climates_demand = sns.barplot(x = by_climates.index,
                               y = by_climates['DEMAND.LOSS.MW']['mean'])
plt.title('Average Demand Loss by Climate Category')
plt.show()
```





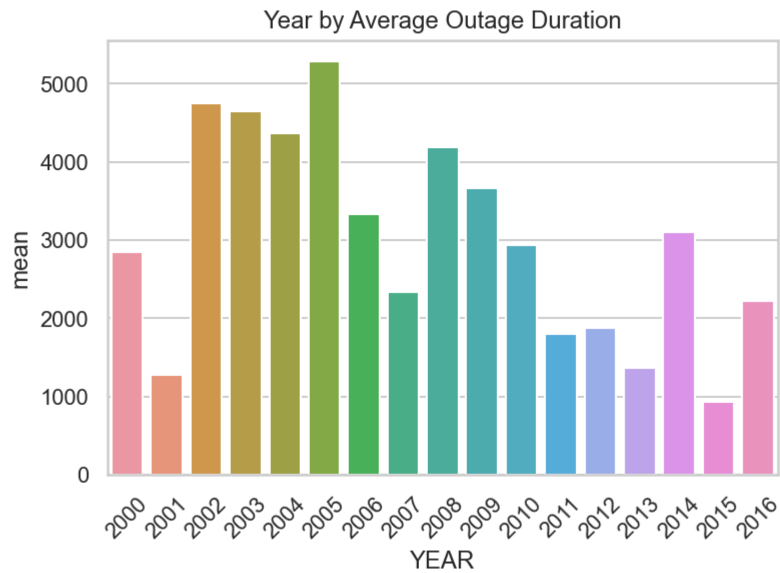


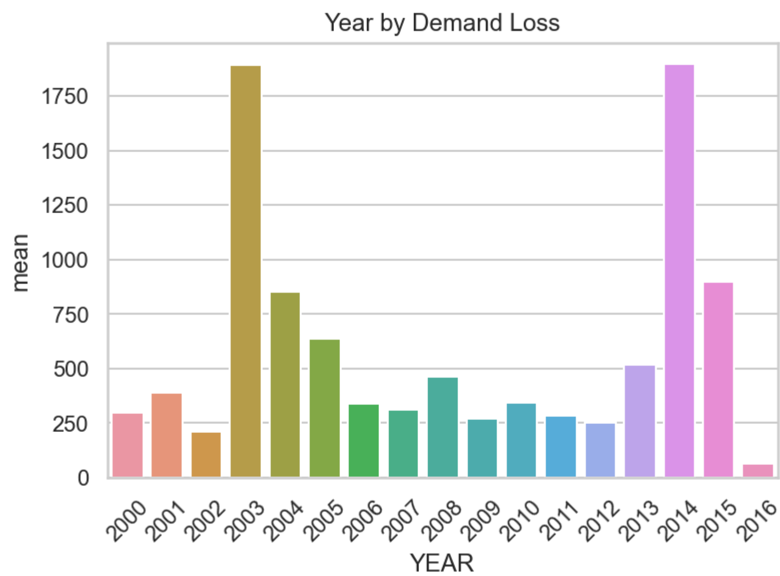
Year Level

In [35]:
```
def year_eda(data):
    """
    Plots the average duration of power outages per year by year.
    """
    data_copy = data.copy(deep = True)
    data_copy = data_copy.drop(columns = ["MONTH"])
    grouped_data = data_copy.groupby("YEAR").agg(['mean', 'min', 'max'])

    return grouped_data
```

In [36]:
```
# Let us aggregate by year and compute the mean min and max.
by_year = year_eda(clean_data)
```

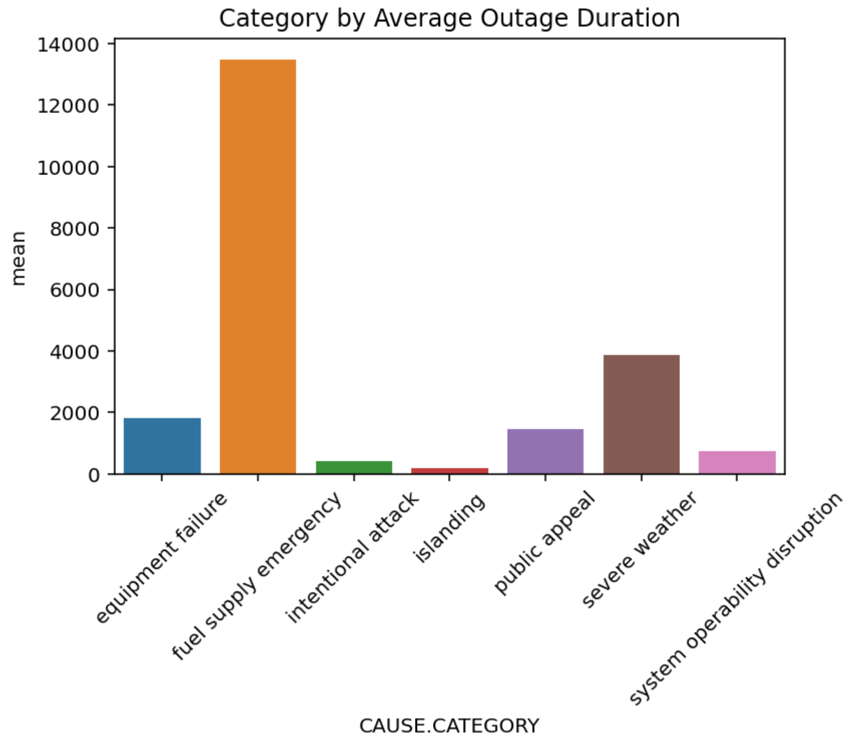
In [37]:
```
# Let us plot the average outage duration and demand loss by year
category_duration = sns.barplot(x = by_year.index,
                                y = by_year['OUTAGE.DURATION']['mean'])
plt.title('Year by Average Outage Duration')
plt.xticks(rotation = 45)
plt.show()

category_dl = sns.barplot(x = by_year.index,
                                y = by_year['DEMAND.LOSS.MW']['mean'])
plt.title('Year by Demand Loss')
plt.xticks(rotation = 45)
plt.show()
```

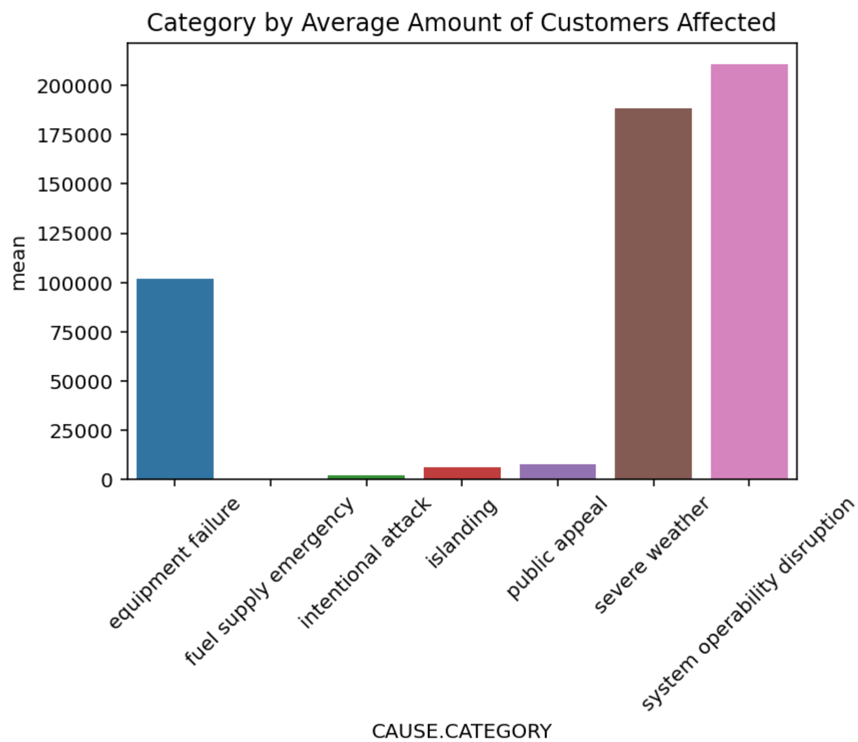


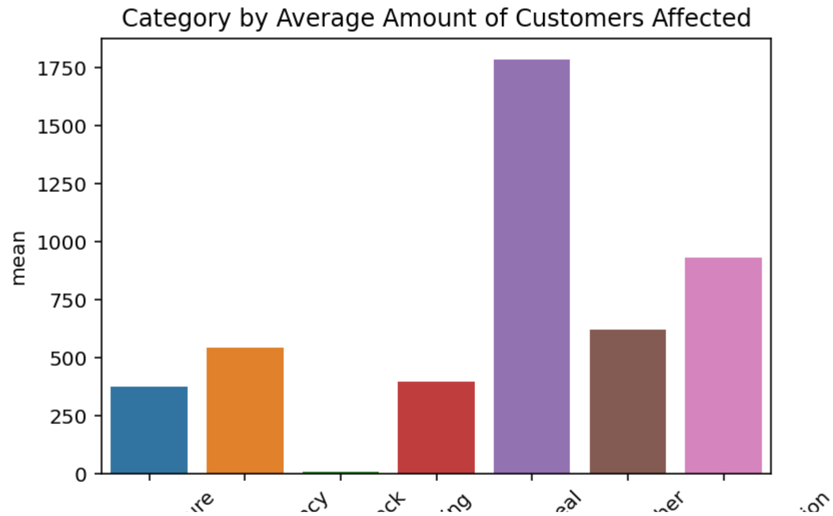


Category Level

In [22]:
```
def category_eda(data):
    """
    This function computes the aggregated statistics
    for each category of outage causation
    """
    data_copy = data.copy()
    data_copy = data_copy[['OUTAGE.DURATION', 'CUSTOMERS.AFFECTED', 'DEMAND
    data_copy['DEMAND.LOSS.MW'] = data_copy['DEMAND.LOSS.MW'].astype(float)

    return data_copy.groupby('CAUSE.CATEGORY').agg(['min', 'max', 'mean'])
# make bar plots for outage by outage duration, demand loss, and customers

by_category = category_eda(clean_data)
category_duration = sns.barplot(x = by_category.index,
                                y = by_category['OUTAGE.DURATION']['mean'])
plt.title('Category by Average Outage Duration')
plt.xticks(rotation = 45)
plt.show()
#Customers affected plot
category_customers = sns.barplot(x = by_category.index,
                                y = by_category['CUSTOMERS.AFFECTED']['mean
plt.title('Category by Average Amount of Customers Affected')
plt.xticks(rotation = 45)
plt.show()

#demand loss by category
category_demand = sns.barplot(x = by_category.index,
                                y = by_category['DEMAND.LOSS.MW']['mean'])
plt.title('Category by Average Amount of Customers Affected')
plt.xticks(rotation = 45)
plt.show()
```


Category by Average Outage Duration
14000
12000
10000
8000
6000
4000
2000
0
mean
equipment failure
fuel supply emergency
intentional attack
islanding
public appeal
severe weather
system operability disruption
CAUSE.CATEGORY


Category by Average Amount of Customers Affected
200000
175000
150000
125000
100000
75000
50000
25000
0
mean
equipment failure
fuel supply emergency
intentional attack
islanding
public appeal
severe weather
system operability disruption
CAUSE.CATEGORY



Assessment of Missingness

In [26]:
```
fp = os.path.join('data', 'outage.xlsx')
data = read_data(fp)
clean_data = data_cleaning(data)
```

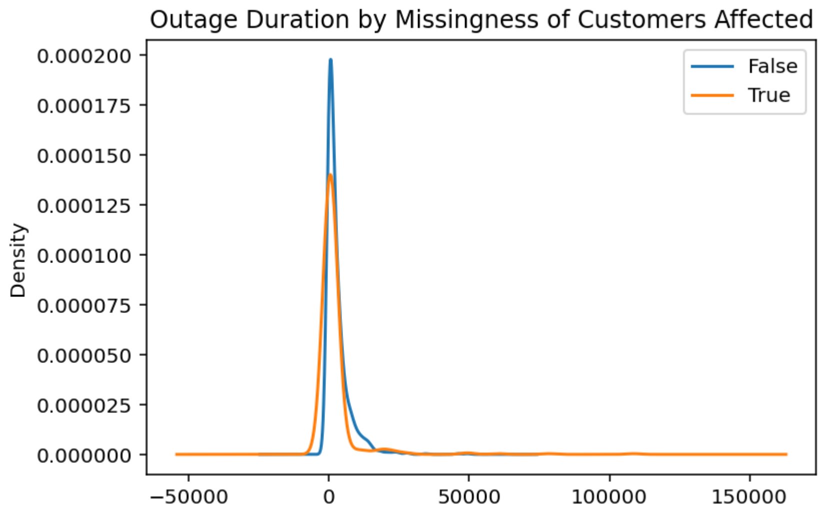
In [73]:
```
# customers effected column MAR dependent on outage duration
# idea: if the outage duration is low the # of customers effected might not
```

Missingness With KS Statistic

Is CUSTOMERS.AFFECTED MAR Dependent on OUTAGE.DURATION?

In [92]:
```
# plot the observed distribution
(clean_data.assign(is_null = clean_data["CUSTOMERS.AFFECTED"].isnull())
 .groupby('is_null')["OUTAGE.DURATION"].plot(kind = 'kde', legend = True, t
)
```

Out[92]:
```
is_null
False    AxesSubplot(0.125,0.125;0.775x0.755)
True     AxesSubplot(0.125,0.125;0.775x0.755)
Name: OUTAGE.DURATION, dtype: object
```

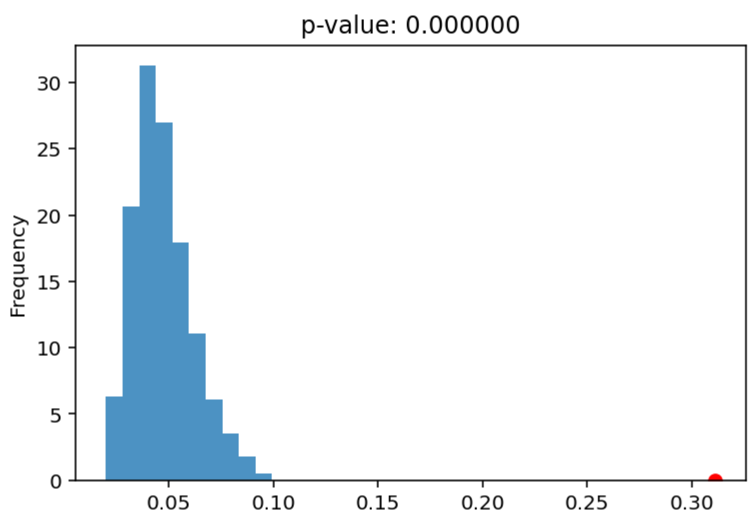


In [93]:
```
# compute the observed ks statistic
copy_clean = clean_data.copy(deep = True)
copy_clean = copy_clean.assign(is_null = copy_clean["CUSTOMERS.AFFECTED"].i
gpA = copy_clean.loc[copy_clean["is_null"] == True, "OUTAGE.DURATION"]
gpB = copy_clean.loc[copy_clean["is_null"] == False, "OUTAGE.DURATION"]

obs = ks_2samp(gpA, gpB).statistic
obs
```

Out[93]:
```
0.31113998588906155
```

In [94]:
```
# simulate the null hypothesis
n_repetitions = 500

ks_list = []

for _ in range(n_repetitions):

    shuffled_col = (
    copy_clean["OUTAGE.DURATION"]
    .sample(replace = False, frac = 1)
    .reset_index(drop = True)
    )

    shuffled = (
    copy_clean
    .assign(**{
        "OUTAGE.DURATION" : shuffled_col,
    })
    )

    grps = shuffled.groupby("is_null")["OUTAGE.DURATION"]
    ks = ks_2samp(grps.get_group(True), grps.get_group(False)).statistic

    ks_list.append(ks)

# compute the pval
pval = np.mean(np.array(ks_list) > obs)


pd.Series(ks_list).plot(kind = 'hist', density = True, alpha = 0.8, title =
plt.scatter(obs, 0, color = 'red', s = 40)
```

Out[94]: <matplotlib.collections.PathCollection at 0x7f8979a35fa0>

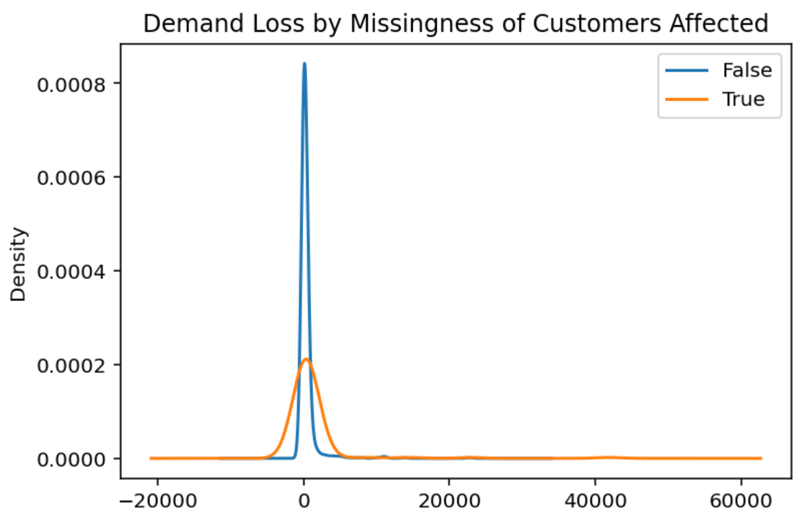


Is CUSTOMERS.AFFECTED MAR Dependent on DEMAND.LOSS.MW?

In [95]:
```
fp = os.path.join('data', 'outage.xlsx')
data = read_data(fp)
clean_data = data_cleaning(data)
```

In [96]:
```
# plot the observed distribution
(clean_data.assign(is_null = clean_data["CUSTOMERS.AFFECTED"].isnull())
 .groupby('is_null')["DEMAND.LOSS.MW"].plot(kind = 'kde', legend = True, ti
)
```

Out[96]:
```
is_null
False    AxesSubplot(0.125,0.125;0.775x0.755)
True     AxesSubplot(0.125,0.125;0.775x0.755)
Name: DEMAND.LOSS.MW, dtype: object
```

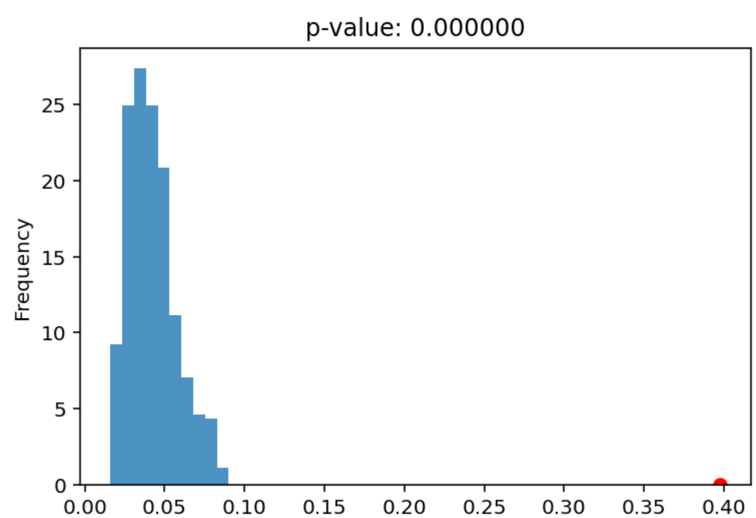


In [97]:
```
# compute the observed statistic
copy_clean = clean_data.copy(deep = True)
copy_clean = copy_clean.assign(is_null = copy_clean["CUSTOMERS.AFFECTED"].i
gpA = copy_clean.loc[copy_clean["is_null"] == True, "DEMAND.LOSS.MW"]
gpB = copy_clean.loc[copy_clean["is_null"] == False, "DEMAND.LOSS.MW"]

obs = ks_2samp(gpA, gpB).statistic
obs
```

Out[97]: 0.3975332755377985

In [98]:
```
# simulate the null hypothesis
n_repetitions = 500

ks_list = []

for _ in range(n_repetitions):

    shuffled_col = (
    copy_clean["DEMAND.LOSS.MW"]
    .sample(replace = False, frac = 1)
    .reset_index(drop = True)
    )

    shuffled = (
    copy_clean
    .assign(**{
        "DEMAND.LOSS.MW" : shuffled_col,
    })
    )

    grps = shuffled.groupby("is_null")["DEMAND.LOSS.MW"]
    ks = ks_2samp(grps.get_group(True), grps.get_group(False)).statistic

    ks_list.append(ks)

pval = np.mean(np.array(ks_list) > obs)


pd.Series(ks_list).plot(kind = 'hist', density = True, alpha = 0.8, title =
plt.scatter(obs, 0, color = 'red', s = 40)
```

Out[98]: <matplotlib.collections.PathCollection at 0x7f8958a29730>

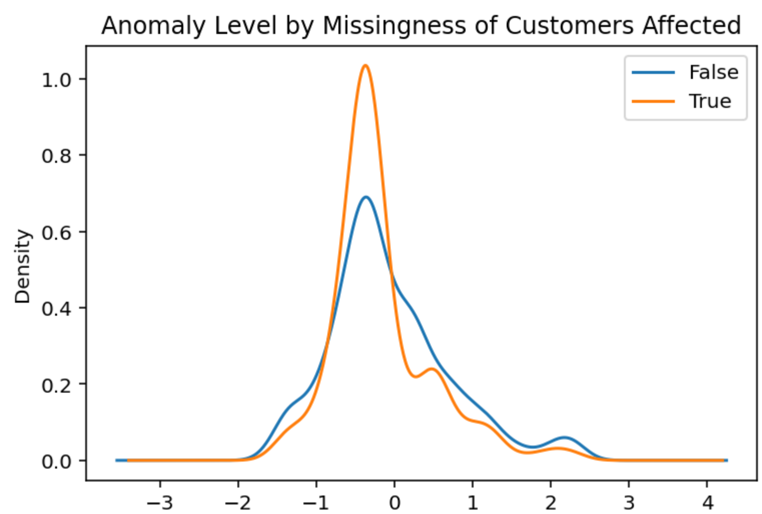


Is CUSTOMERS.AFFECTED MAR Dependent on ANOMALY LEVEL?

In [99]:
```
fp = os.path.join('data', 'outage.xlsx')
data = read_data(fp)
clean_data = data_cleaning(data)
```

In [100]:
```
# plot the observed distribution
(clean_data.assign(is_null = clean_data["CUSTOMERS.AFFECTED"].isnull())
 .groupby('is_null')["ANOMALY.LEVEL"].plot(kind = 'kde', legend = True, tit
)
```

Out[100]:
```
is_null
False    AxesSubplot(0.125,0.125;0.775x0.755)
True     AxesSubplot(0.125,0.125;0.775x0.755)
Name: ANOMALY.LEVEL, dtype: object
```

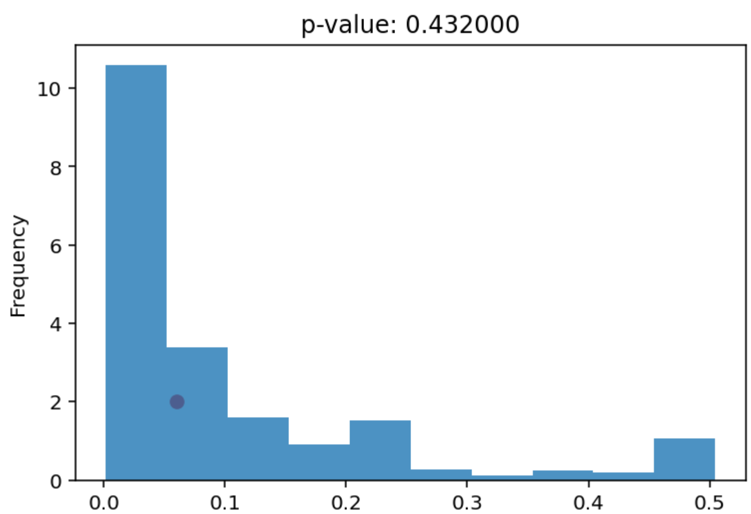


In [101]:
```
# compute the observed statistic
copy_clean = clean_data.copy(deep = True)
copy_clean = copy_clean.assign(is_null = copy_clean["CUSTOMERS.AFFECTED"].i
gpA = copy_clean.loc[copy_clean["is_null"] == True, "ANOMALY.LEVEL"]
gpB = copy_clean.loc[copy_clean["is_null"] == False, "ANOMALY.LEVEL"]

obs = ks_2samp(gpA, gpB).statistic
obs
```

Out[101]:
```
0.06012046024005148
```

In [102]:
```
# simulate the null hypothesis
n_repetitions = 500

ks_list = []

for _ in range(n_repetitions):

    shuffled_col = (
    copy_clean["ANOMALY.LEVEL"]
    .sample(replace = False, frac = 1)
    .reset_index(drop = True)
    )

    shuffled = (
    copy_clean
    .assign(**{
        "ANOMALY.LEVEL" : shuffled_col,
    })
    )

    grps = shuffled.groupby("is_null")["ANOMALY.LEVEL"]
    ks = ks_2samp(grps.get_group(True), grps.get_group(False)).statistic

    ks_list.append(ks)

pval = np.mean(np.array(ks_list) > obs)


pd.Series(ks_list).plot(kind = 'hist', density = True, alpha = 0.8, title =
plt.scatter(obs, 2, color = 'red', s = 40)
```

Out[102]:
```
<matplotlib.collections.PathCollection at 0x7f897996a6a0>
```



Hypothesis Test

Question: (just a reminder)

Is the distribution of climate category different for those outages that were classified as 'major' than those that were not classified as 'major'?

In [69]:
```
# conditional function to determine if a power outage is classified as 'maj
def detect_major(row):
    if row["CUSTOMERS.AFFECTED"] > 50000:
        return True
    elif row["DEMAND.LOSS.MW"] > 300:
        return True
    else:
        return False
```

In [87]:
```
fp = os.path.join('data', 'outage.xlsx')
data = read_data(fp)
clean_data = data_cleaning(data)
```

In [127]:
```
# compute the distribution of outages in a certain climate conditional on w
clean_data["Major_Outage"] = clean_data.apply(detect_major, axis = 1)
climate_counts = clean_data.pivot_table(index = "CLIMATE.CATEGORY", columns
climate_counts
```

Out[127]:

Major_Outage	False	True
CLIMATE.CATEGORY		
cold	236	237
normal	367	377
warm	140	168

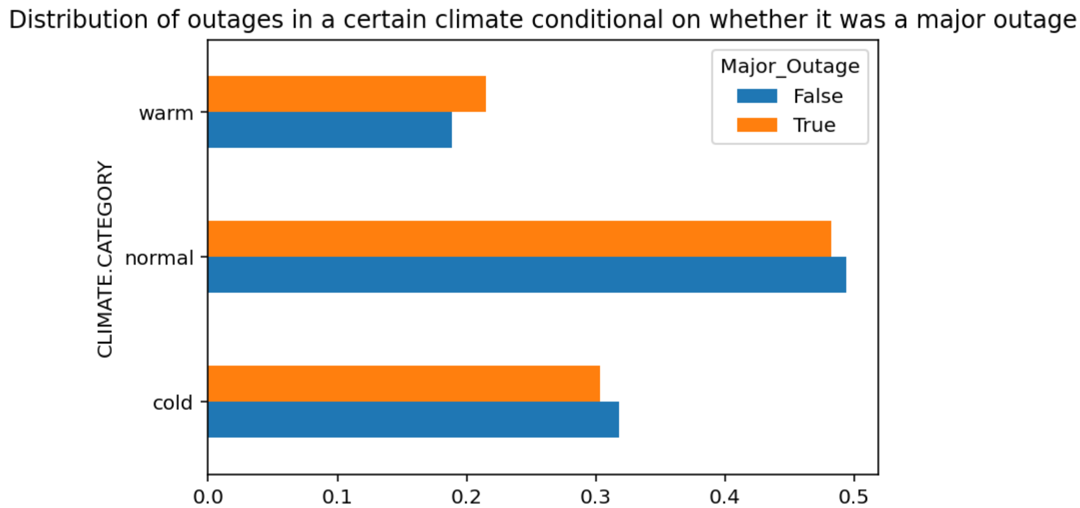
In [89]:
```
climate_counts.sum().to_frame().T
cond_distr = climate_counts / climate_counts.sum()
cond_distr
```

Out[89]:

Major_Outage	False	True
CLIMATE.CATEGORY		
cold	0.317631	0.303069
normal	0.493943	0.482097
warm	0.188425	0.214834

In [90]:
```
# plot the observed distribution
title = "Distribution of outages in a certain climate conditional on whethe
cond_distr.plot(kind = 'barh', title = title)
```

Out[90]:
```
<AxesSubplot:title={'center':'Distribution of outages in a certain climat
e conditional on whether it was a major outage'}, ylabel='CLIMATE.CATEGOR
Y'>
```



In [91]:
```
# find the observed TVD
obs = cond_distr.diff(axis =1).iloc[-1].abs().sum() / 2
obs
```

Out[91]: 0.013204228382206654

In [92]:
```
# helper function to compute the TVD
def compute_tvd(df):

    counts = df.pivot_table(index = "CLIMATE.CATEGORY", columns = "Major_Ou
    distr = counts / counts.sum()

    return distr.diff(axis = 1).iloc[-1].abs().sum() / 2
```

In [93]:
```
# ensure that the helper function works
observed = compute_tvd(clean_data)
observed
```

Out[93]: 0.013204228382206654

In [95]:
```
# simulate null
N = 1000

tvds = []

for _ in range(N):

    s = clean_data["Major_Outage"].sample(frac = 1, replace = False).reset_
    shuffled = clean_data.loc[:, ["CLIMATE.CATEGORY"]].assign(Major_Outage

    tvds.append(compute_tvd(shuffled))

tvds = pd.Series(tvds)
```

In [98]:
```
# compute the pval
pval = (tvds >= obs).sum() / N
pval
```

Out[98]: 0.206

In [101]:
```
# plot the simulated tvds with the observed
tvds.plot(kind = 'hist', title = 'p-value: %f' % pval)

perc = np.percentile(tvds, 95)
plt.axvline(x = perc, color = 'y')
```

Out[101]: <matplotlib.lines.Line2D at 0x7fd414b5ea30>