PowerOutageAnalysis

Name(s): (Jiayi zhu, Yihuan Wang)

Website Link: (https://yhw086.github.io/PowerOutageAnalysis/ (https://yhw086.github.io/PowerOutageAnalysis/))

Code

In [1]:
```
import plotly
```

In [2]:
```
import plotly.offline as pyo
pyo.init_notebook_mode(connected=True)
```

In [3]:
```
import plotly.io as pio
pio.renderers.default='notebook'
```

In [4]:
```
import pandas as pd
import numpy as np
import os

import plotly.express as px
pd.options.plotting.backend = 'plotly'
```

In [5]:
```
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
```

Research Question **"Does the occurrence of 'Extreme' climate conditions (including both El Niño and La Niña), as defined by the Oceanic Niño Index, versus 'Normal' conditions significantly influence the average duration of power outages in the United States?"**.

Power outage represents a significant challenge, which impacts to our environment. Understanding the factors influencing power outage duration is significant for developing effective strategies to enhance the time to restoration and minimize its disruptions. In this project, we aim to investigate the relationship between the normal and extreme conditions and the duration of power outages.

Before investigating our project, we read the excel data into the notebook. The dataset is sourced from the Purdue University laboratory, titled Major Power Outage Risks in the U.S.. The dataset has 1534 rows which represents the number of time the power outage happen in the U.S. The dataset has 55 columns. Since we aim to learn the relationship between the climate and power outages, we select 8 columns to use, including `Year` , `Anomaly.level` , `Climate.category` , `Cause.category` , `Cause.category.detail` , `Outage.duration` , `Customers.affected` , `Climate` .

- `Year` : Indicates the year when the outage event occurred.
- `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).
- `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 outages.
- `Outage.duration` : Duration of outage events (in minutes).
- `Customers.affected` : Number of customers affected by the power outage event.
- `Climate` : Replaces all 'warm' and 'cold' conditions in `Climate.category` column by 'extreme', keep 'normal' as it is, according to Oceanic Niño Index.

After setting the dataset, we begin to process the data. Firstly, we will do the Data cleaning and conduct the exploratory data analysis, including Univariate Analysis, Bivariate Analysis, and interesting aggregates.

Later then, we are going to assess the missingness analysis to analysis the column's missing dependency. In the missingness analysis, we will first discuss the NMAR, which is `Cause.category.detail` . Then we will discuss the MCAR and MAR analysis, and we mainly implement the test to explore the dependency of the missing power outage duration on the `Climate` and `Customers.affected` columns.

Moreover, we generate the hypothesis test according to the research question. We would analyze if the average duration of power outages during 'Extreme' climate conditions is the same as during 'Normal' climate conditions. Our research question is important, since it addresses the immediate impact of climate on power outages and also contributes valuable insights for proactive planning and risk reduction in the face of improving climate patterns. This investigation can enhance the resilience of power infrastructures in diverse climatic scenarios.

Cleaning and EDA

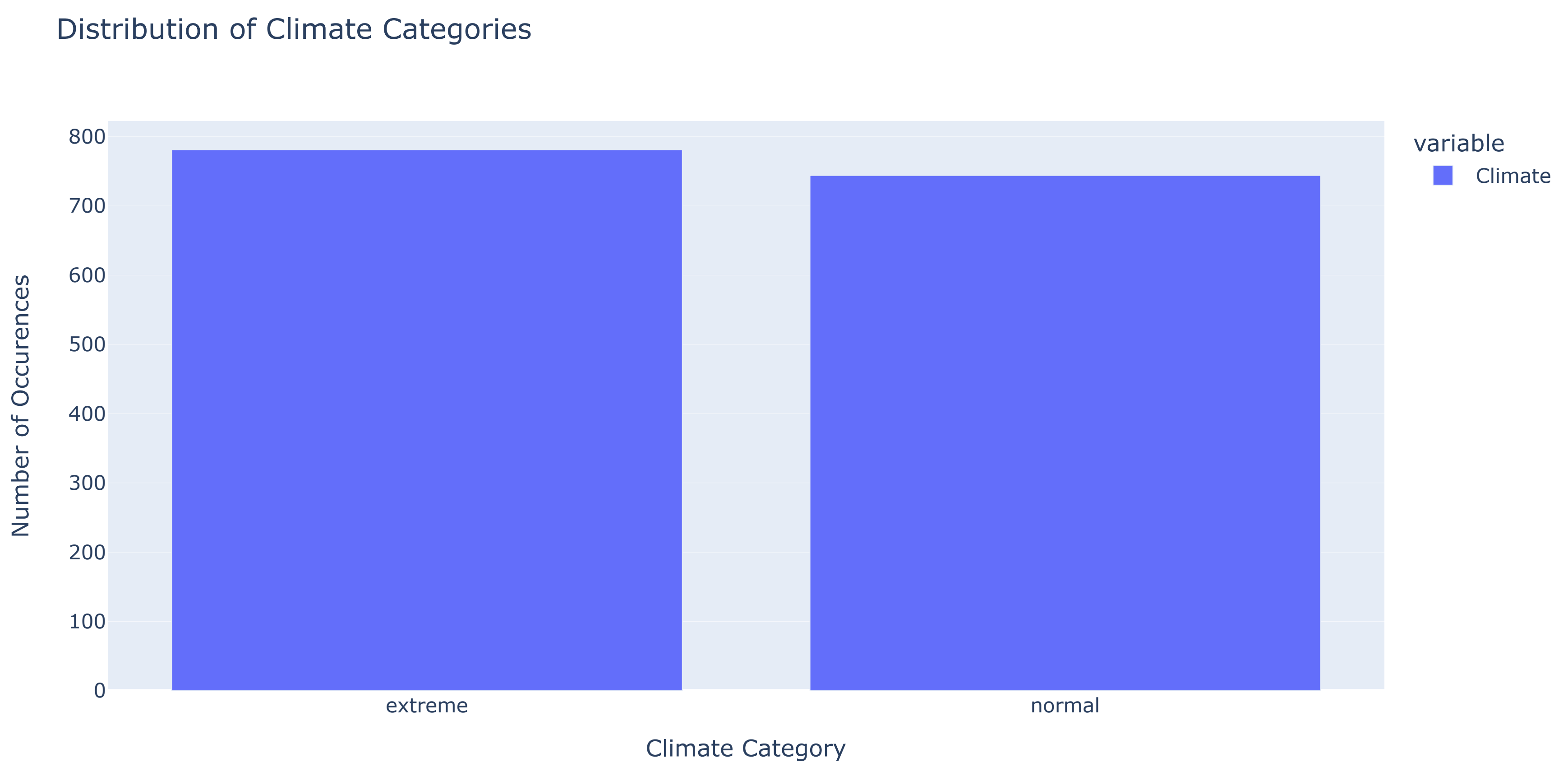
In [6]:
```python
# skip first 5 rows and drop unrelevant columns and rows
# rename variable names
# create new column `Climate` and change Year type
# keep only the relevant columns
file_path = 'outage.xlsx'
df = pd.read_excel(file_path, skiprows=5)
df = df.drop(columns=['variables', 'OBS'])[1:]
columns = [ea.capitalize() for ea in df.columns]
df.columns = columns
df['Climate'] = df['Climate.category'].replace({'warm': 'extreme', 'cold': 'extreme', 'normal': 'normal'})
df['Year'] = df['Year'].astype(int)
columns_of_interest = ['Year', 'Anomaly.level', 'Climate.category', 'Cause.category', 'Cause.category.detail',
                       'Outage.duration', 'Customers.affected', 'Climate']
df = df[columns_of_interest]
df.head(5)
```

Out[6]:

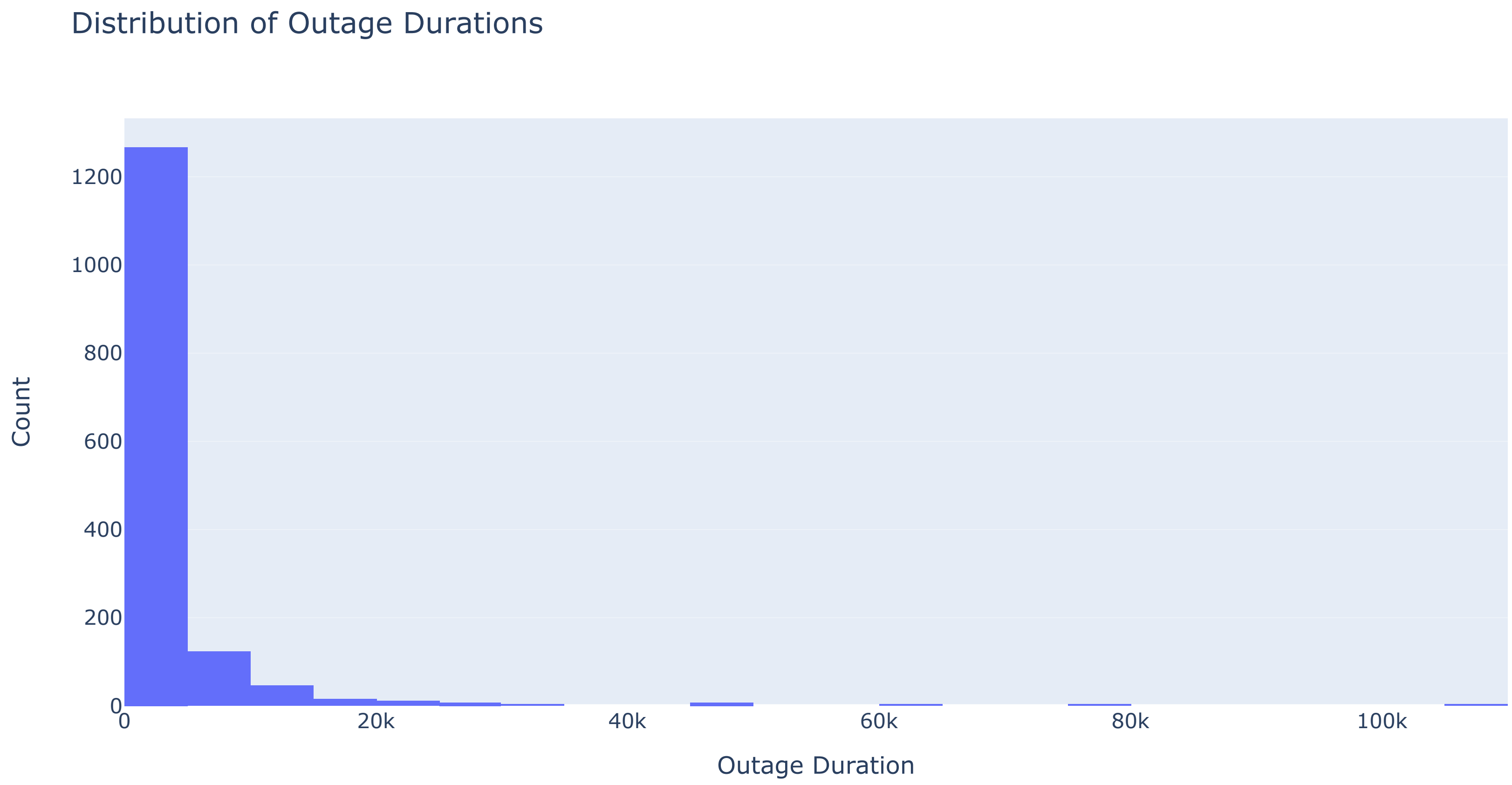
	Year	Anomaly.level	Climate.category	Cause.category	Cause.category.detail	Outage.duration	Customers.affected	Climate
1	2011	-0.3	normal	severe weather	NaN	3060	70000.0	normal
2	2014	-0.1	normal	intentional attack	vandalism	1	NaN	normal
3	2010	-1.5	cold	severe weather	heavy wind	3000	70000.0	extreme
4	2012	-0.1	normal	severe weather	thunderstorm	2550	68200.0	normal
5	2015	1.2	warm	severe weather	NaN	1740	250000.0	extreme

#outage['ANOMALY.LEVEL'].isna().sum() nan_proportions = df.iloc[:,1:].isna().mean() nan_proportions

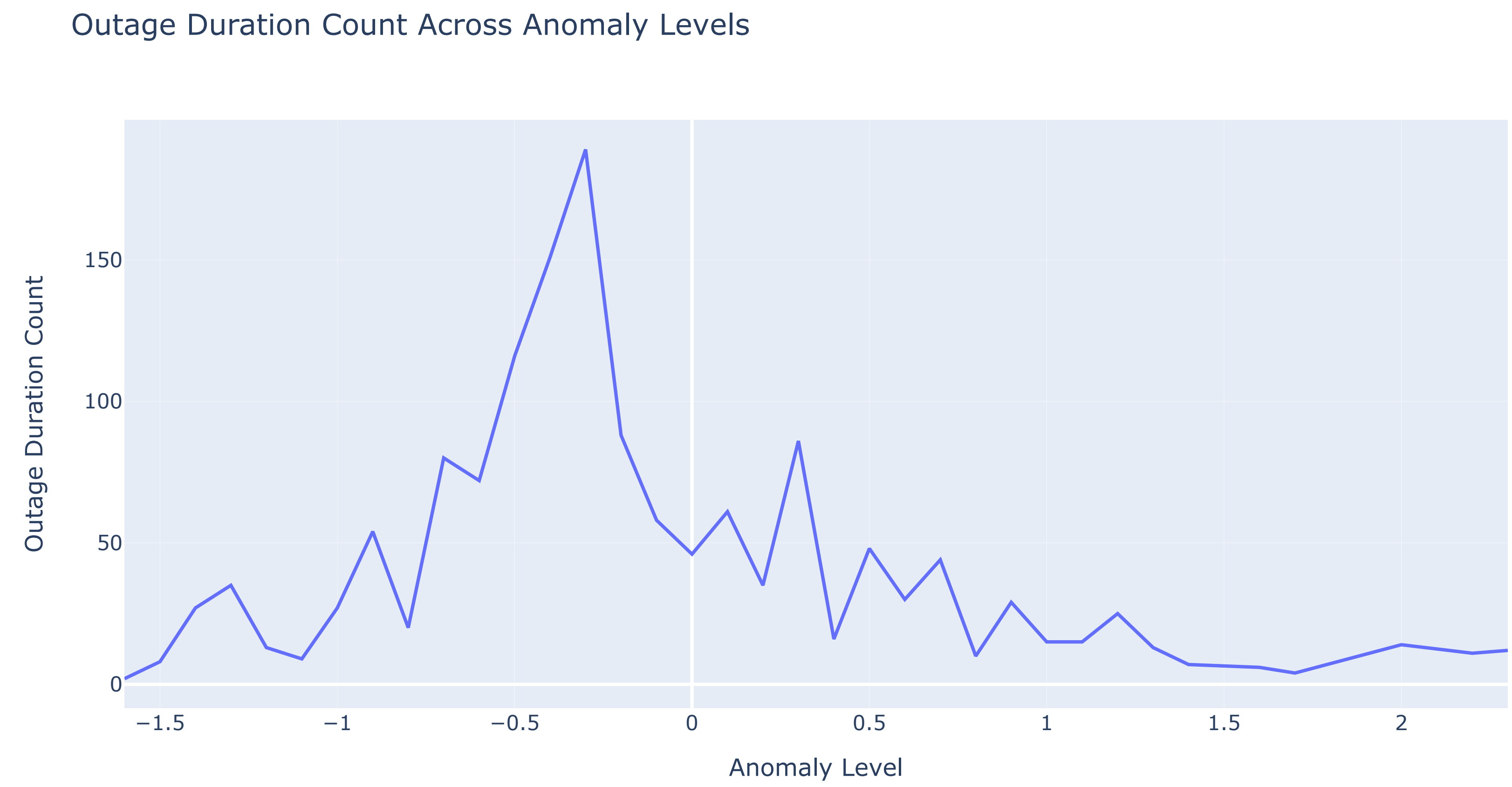
In [7]:
```python
# Univariate Analysis for 'Climate'
# Distribution of Climate Categories
climate_count_fig = px.bar(df['Climate'].value_counts(), title='Distribution of Climate Categories')
climate_count_fig.update_xaxes(title='Climate Category')
climate_count_fig.update_yaxes(title='Number of Occurences')
climate_count_fig.show()
climate_count_fig.write_html('uni_climate.html', include_plotlyjs='cdn')
```



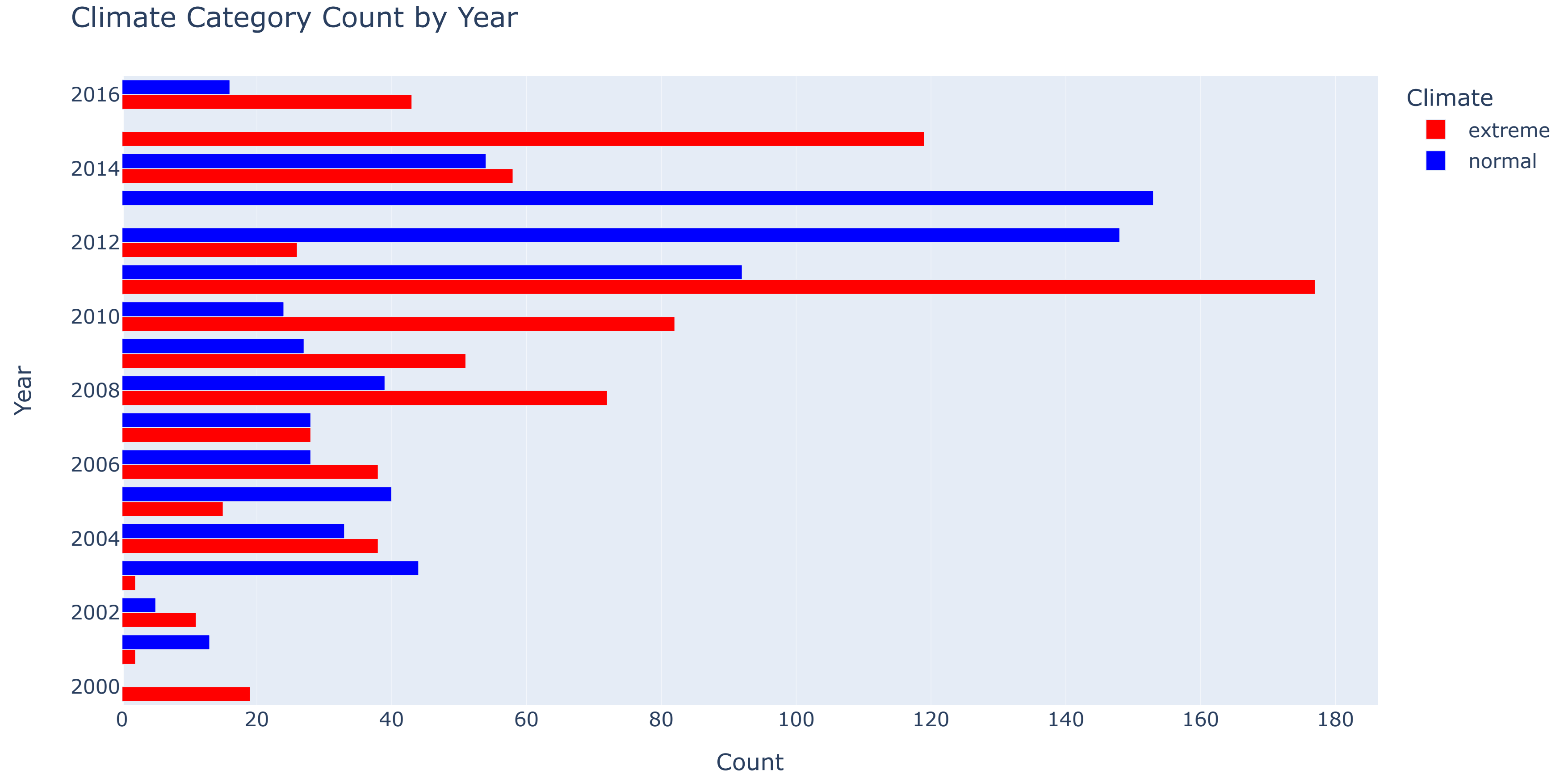
In [8]:
```
# Univariate Analysis for 'Outage.duration'
# Histogram for 'Outage.duration'
outage_duration_fig = px.histogram(df, x='Outage.duration', nbins=30, title='Distribution of Outage Durations')
outage_duration_fig.update_xaxes(title='Outage Duration')
outage_duration_fig.update_yaxes(title='Count')
outage_duration_fig.write_html('uni_duration.html', include_plotlyjs='cdn')
outage_duration_fig.show()
```



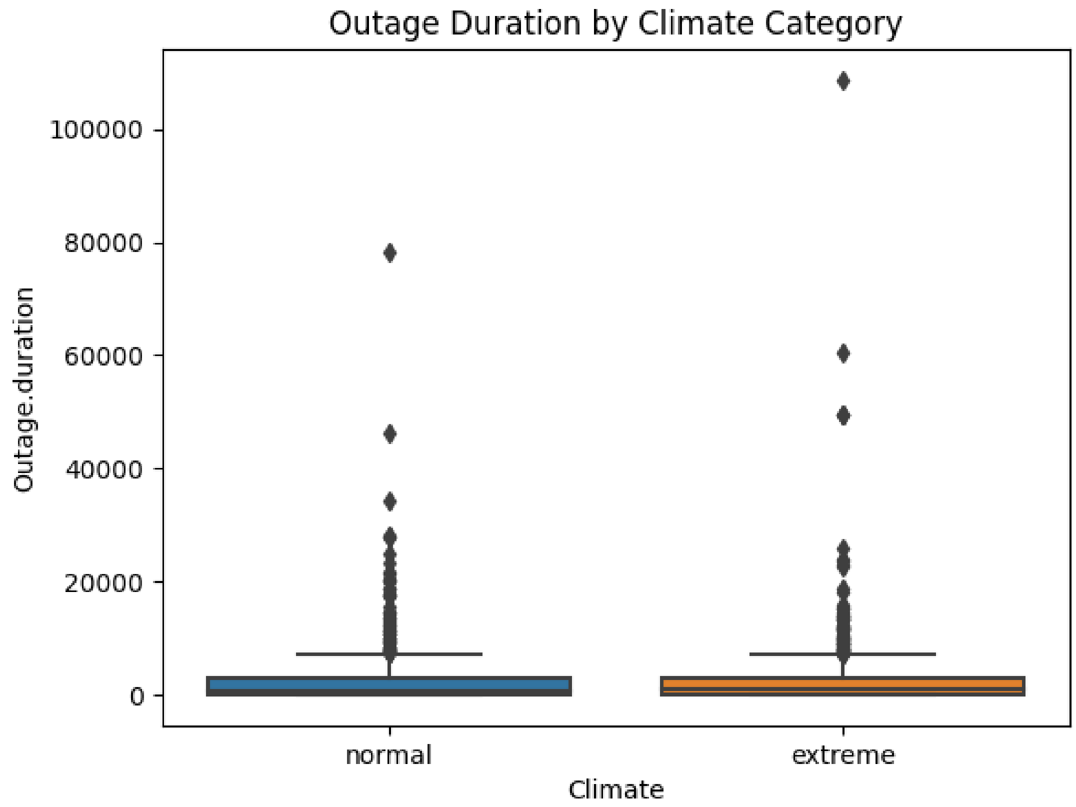
In [9]:
```
# Bivariate line plot
# Outage Duration Count Across Anomaly Levels
outage_counts = df.groupby('Anomaly.level')['Outage.duration'].count().reset_index()

bivar_lin_fig = px.line(outage_counts, x='Anomaly.level', y='Outage.duration',
              title='Outage Duration Count Across Anomaly Levels',
              labels={'Anomaly.level': 'Anomaly Level', 'Outage.duration': 'Outage Duration Count'})
bivar_lin_fig.update_xaxes(title='Anomaly Level')
bivar_lin_fig.update_yaxes(title='Outage Duration Count')
bivar_lin_fig.show()
bivar_lin_fig.write_html('bivar_line.html', include_plotlyjs='cdn')
```



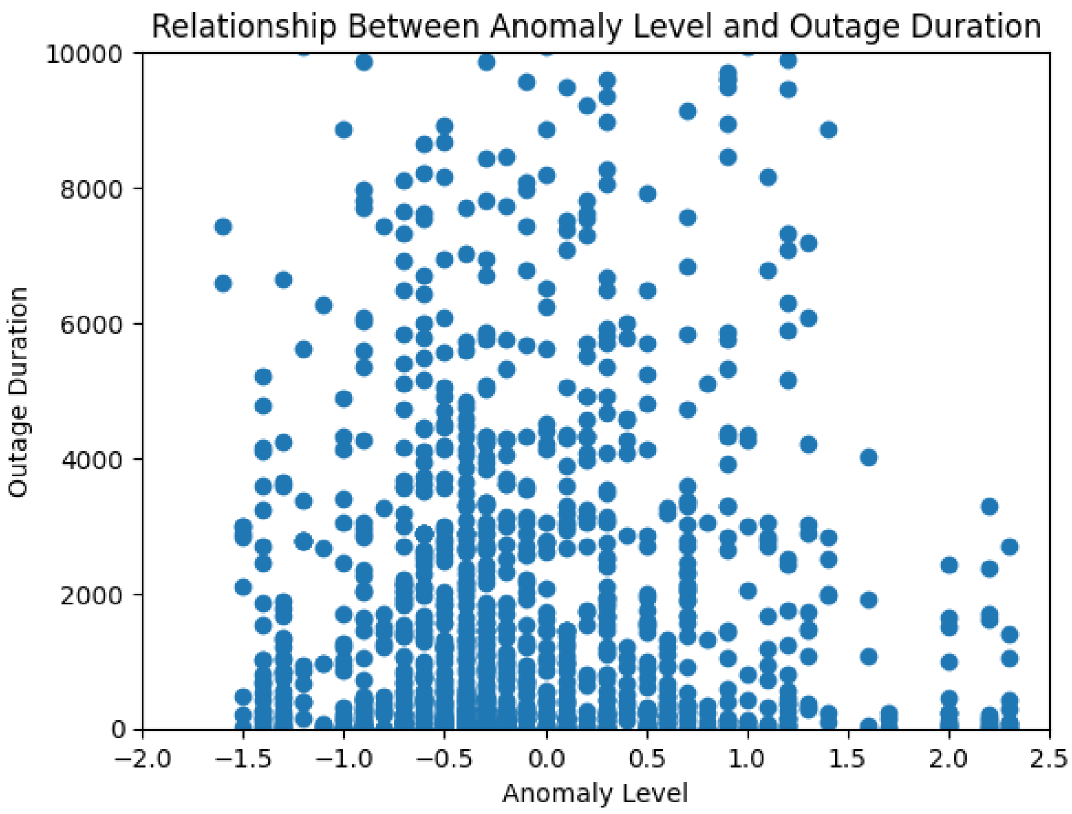
In [10]:
```
# Bivariate Analysis
# Group data by 'Year' and 'Climate' and count occurrences
climate_counts = df.groupby(['Year', 'Climate']).size().reset_index(name='Count')
color_map = {'extreme': 'red', 'normal': 'blue'}
bivar_group_fig = px.bar(climate_counts, x='Count', y='Year', color='Climate',
                         orientation='h', barmode='group',
                         color_discrete_map=color_map)
bivar_group_fig.update_layout(title='Climate Category Count by Year', xaxis_title='Count', yaxis_title='Year')
bivar_group_fig.show()
bivar_group_fig.write_html('bivar_group.html', include_plotlyjs='cdn')
```



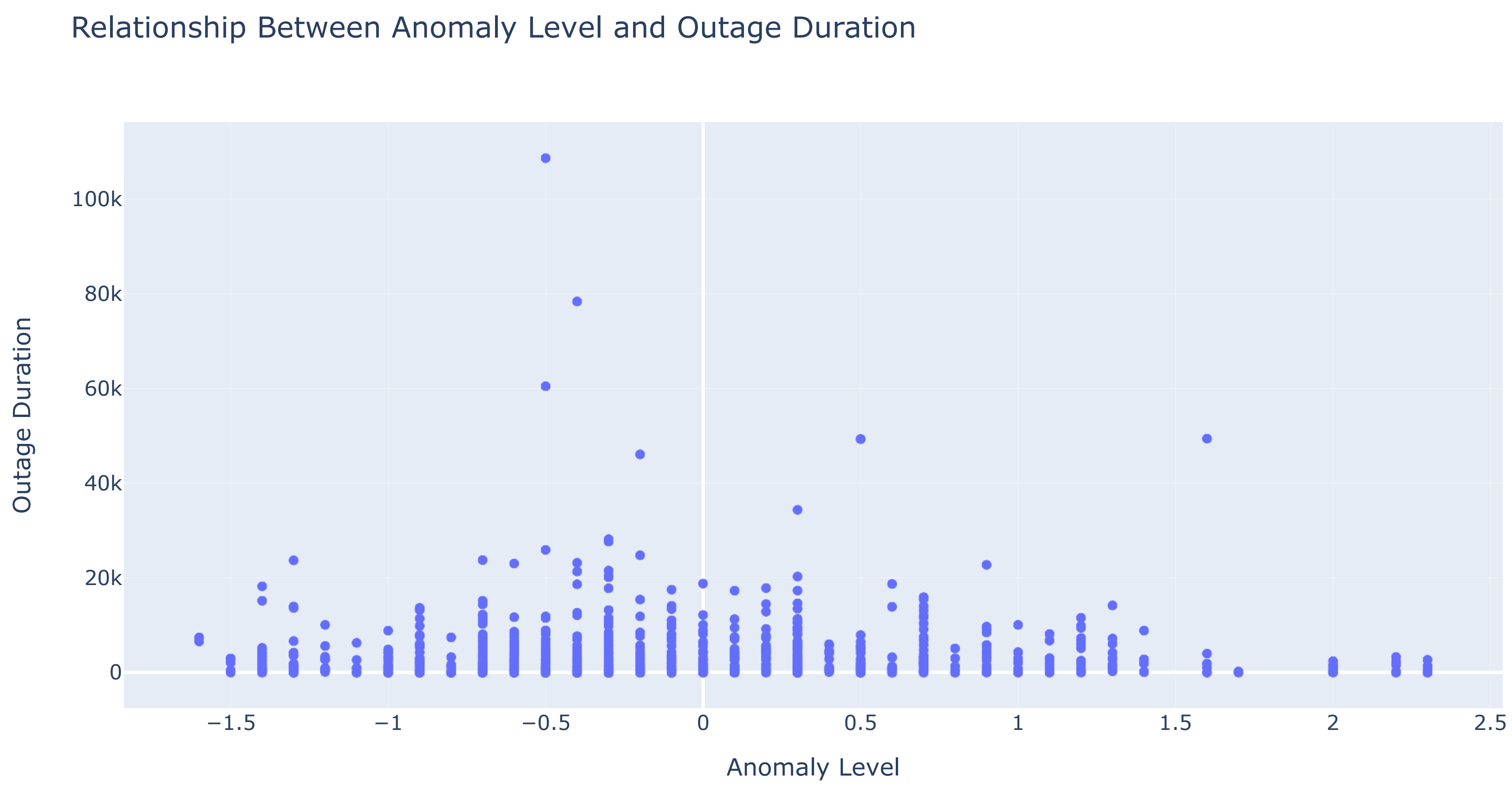
In [11]:
```
# Bivariate Analysis
# Grouped Box Plot
sns.boxplot(x='Climate', y='Outage.duration', data=df)
plt.title('Outage Duration by Climate Category')
plt.show()
```



In [12]:
```
# Bivariate Analysis
# Scatter plot

plt.scatter(df['Anomaly.level'], df['Outage.duration'])

plt.xlim(-2, 2.5)
plt.ylim(0, 10000)
plt.title('Relationship Between Anomaly Level and Outage Duration')
plt.xlabel('Anomaly Level')
plt.ylabel('Outage Duration')

plt.show()
```



In [13]:
```
# Bivariate Analysis
# Scatter plot
fig = px.scatter(df, x='Anomaly.level', y='Outage.duration',
                 title='Relationship Between Anomaly Level and Outage Duration',
                 labels={'Anomaly.level': 'Anomaly Level', 'Outage.duration': 'Outage Duration'})
fig.show()
```



In [14]:
```
# aggregate by 'Climate.category' and 'Outage.duration'

aggregated_data = (df.groupby('Climate.category')['Outage.duration']
                   .agg(['mean', 'median', 'std', 'count', 'min', 'max']).reset_index())

# Rename columns for clarity
aggregated_data.columns = ['Climate Category', 'Mean Duration', 'Median Duration', 'Standard Deviation',
                           'Count of Outages', 'Min Duration', 'Max Duration']

aggregated_data
```

Out[14]:

	Climate Category	Mean Duration	Median Duration	Standard Deviation	Count of Outages	Min Duration	Max Duration
0	cold	2656.956803	816.0	6736.666752	463	0	108653
1	normal	2530.980822	563.0	5365.118871	730	0	78377
2	warm	2817.318021	881.0	5990.164205	283	0	49427

Assessment of Missingness

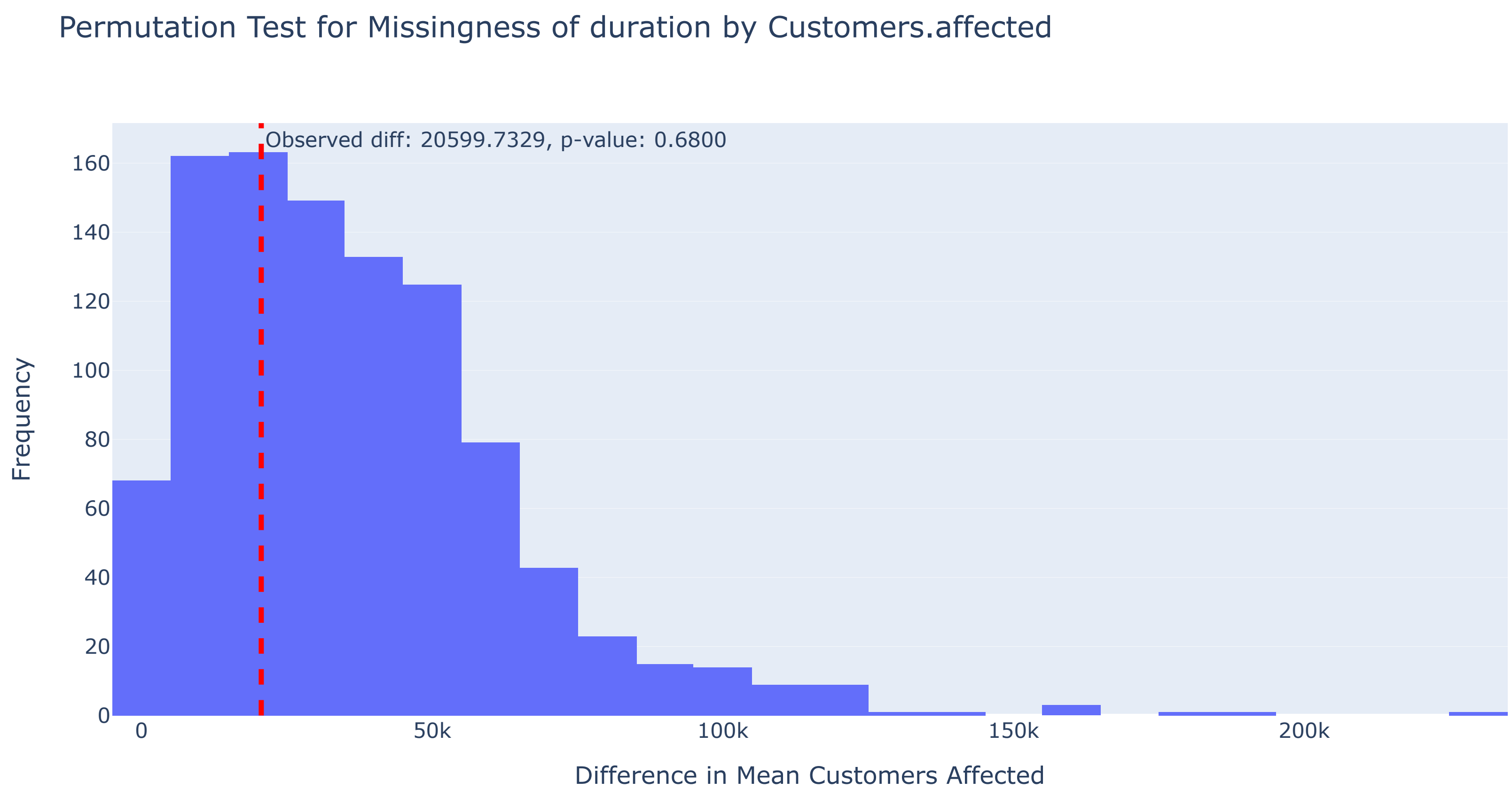
In [15]:
```
# check dependency of Outage.duration on Customers.affected
diffs = []
for i in range(1000):
    shuffled = df.copy()
    shuffled['Missing_duration'] = np.random.permutation(shuffled['Outage.duration'].isna())
    group_means = shuffled.groupby('Missing_duration')['Customers.affected'].mean()
    groupby_diff = abs(group_means.diff().iloc[-1])
    diffs.append(groupby_diff)

shuffled['Missing_duration'] = shuffled['Outage.duration'].isna()
obs_diff = abs(shuffled.groupby('Missing_duration')['Customers.affected'].mean().diff().iloc[-1])

p_val = (diffs >= obs_diff).mean()
p_val
```

Out[15]: 0.68

In [16]:
```
obs_diff = abs(shuffled.groupby('Missing_duration')['Customers.affected'].mean().diff().iloc[-1])
obs_diff
```

Out[16]: 20599.732900432893

In [17]:
```
perm_test_fig = px.histogram(x=diffs, nbins=30, title='Permutation Test for Missingness of duration by Customers.affecte
perm_test_fig.add_vline(x=obs_diff, line_width=3, line_dash="dash", line_color="red", annotation_text=f"Observed diff: {
perm_test_fig.update_layout(
    xaxis_title='Difference in Mean Customers Affected',
    yaxis_title='Frequency',
    showlegend=False
)
perm_test_fig.show()

perm_test_fig.write_html('perm_test_diff.html', include_plotlyjs='cdn')
```



In [18]:
```
out_dist = (
    df
    .assign(outage_missing=df['Outage.duration'].isna())
    .pivot_table(index='Climate', columns='outage_missing', aggfunc='size')
)

out_dist.columns = ['out_missing = False', 'out_missing = True']

out_dist = out_dist / out_dist.sum()
out_dist
```

Out[18]:

Climate	out_missing = False	out_missing = True
extreme	0.50542	0.714286
normal	0.49458	0.285714

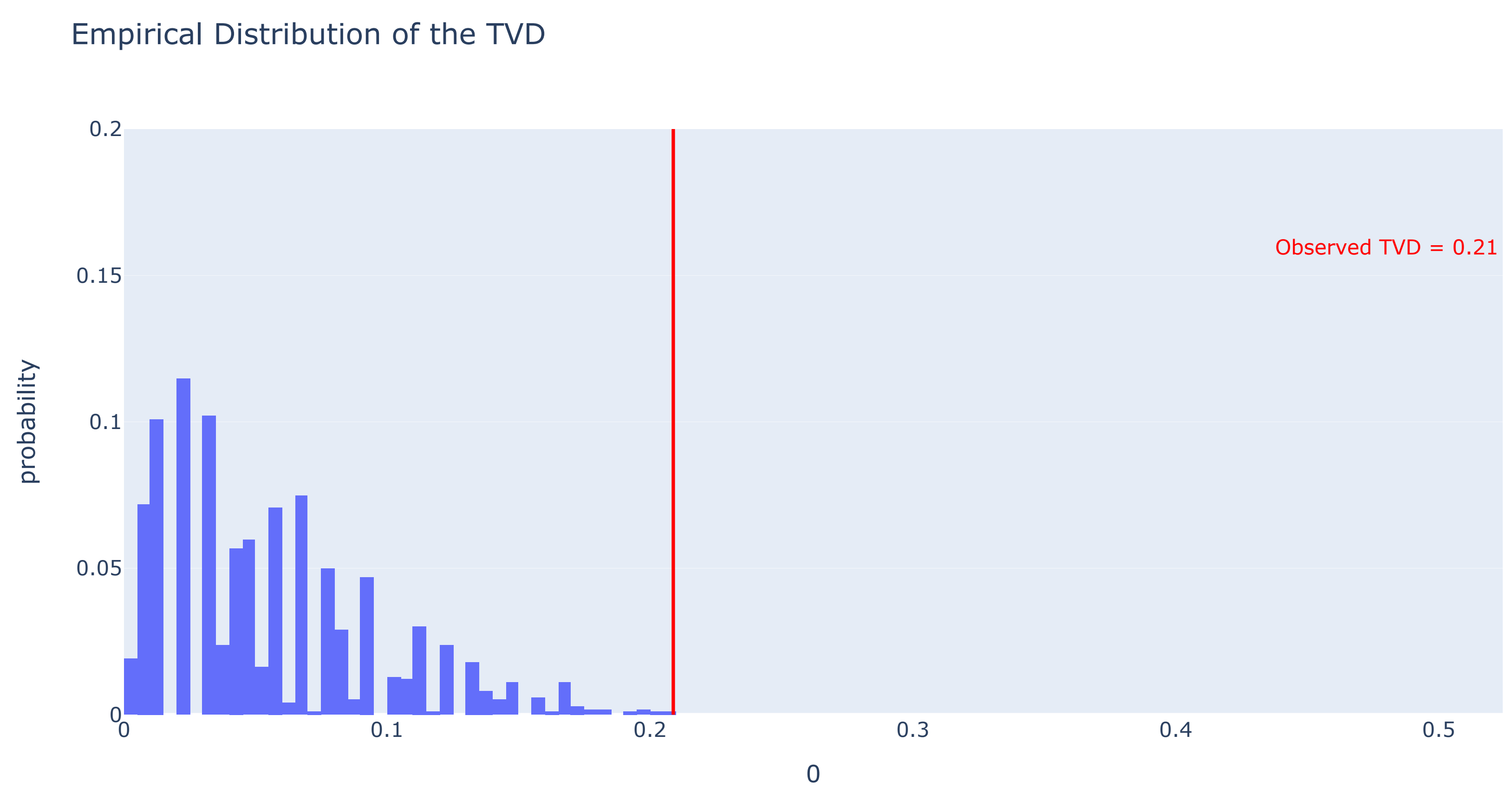
In [19]:
```
# check dependency of Outage.duration on Climate

n_repetitions = 1000
shuffled = df.copy()
shuffled['out_missing'] = shuffled['Outage.duration'].isna()

tvds = []
for _ in range(n_repetitions):

    shuffled['out_missing'] = np.random.permutation(shuffled['out_missing'])

    pivoted = (
        shuffled
        .pivot_table(index='Climate', columns='out_missing', aggfunc='size')
        .apply(lambda x: x / x.sum())
    )

    tvd = pivoted.diff(axis=1).iloc[:, -1].abs().sum() / 2
    tvds.append(tvd)

observed_tvd = out_dist.diff(axis=1).iloc[:, -1].abs().sum() / 2
observed_tvd
```

Out[19]: 0.2088656600851723

In [20]:
```
pval = np.mean(np.array(tvds) >= observed_tvd)
pval
```

Out[20]: 0.001

In [21]:
```
mar_fig = px.histogram(pd.DataFrame(tvds), x=0, nbins=50, histnorm='probability',
                   title='Empirical Distribution of the TVD')
mar_fig.add_vline(x=observed_tvd, line_color='red')
mar_fig.add_annotation(text=f'<span style="color:red">Observed TVD = {round(observed_tvd, 2)}</span>',
                   x=2.3 * observed_tvd, showarrow=False, y=0.16)
mar_fig.update_layout(yaxis_range=[0, 0.2])
mar_fig.show()
mar_fig.write_html('mar.html', include_plotlyjs='cdn')
```



Hypothesis Testing

Null Hypothesis (H0): The average duration of power outages during 'Extreme' climate conditions is the same as during 'Normal' climate conditions. Alternative

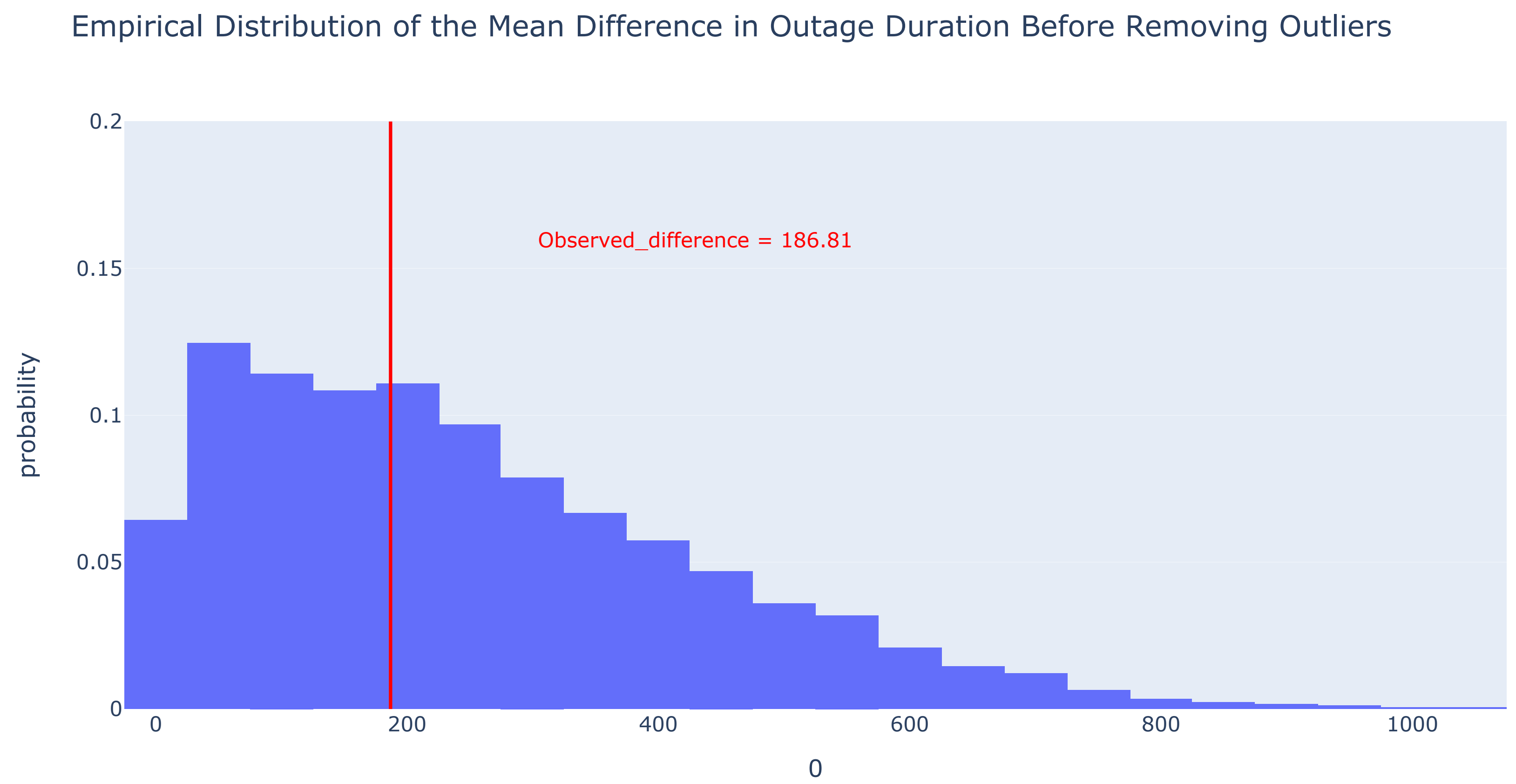
In [22]:
```
# test statistic

n_repetitions = 10000

differences = []
for _ in range(n_repetitions):
    copy = df[['Outage.duration', 'Climate']].copy()
    with_shuffled = copy.assign(Shuffled_duration=np.random.permutation(copy['Outage.duration']))

    group_means = (
        with_shuffled
        .groupby('Climate')
        .mean()
        .loc[:, 'Shuffled_duration']
    )
    difference = abs(group_means.diff().iloc[-1])
    differences.append(difference)

observed_difference = abs(df.groupby('Climate')['Outage.duration'].mean().diff().iloc[-1])

p_value= np.mean([np.abs(diff) >= np.abs(observed_difference) for diff in differences])
p_value
```

Out[22]: 0.5618

In [23]:
```
observed_difference = abs(df.groupby('Climate')['Outage.duration'].mean().diff().iloc[-1])
observed_difference
```

Out[23]: 186.8100628006905

In [25]:
```
hypo1_fig = px.histogram(pd.DataFrame(differences), x=0, nbins=30, histnorm='probability',
            title='Empirical Distribution of the Mean Difference in Outage Duration Before Removing Outliers')
hypo1_fig.add_vline(x=observed_difference, line_color='red')
hypo1_fig.add_annotation(text=f'<span style="color:red">Observed_difference = {round(observed_difference, 2)}</span>',
                         x=2.3 * observed_difference, showarrow=False, y=0.16)
hypo1_fig.update_layout(yaxis_range=[0, 0.2])
hypo1_fig.show()

hypo1_fig.write_html('hypo1.html', include_plotlyjs='cdn')
```



In [26]:
```
# detect how many outliers the outage.duration have
def detect_outliers(series, factor=1.5):
    Q1 = series.quantile(0.25)
    Q3 = series.quantile(0.75)
    IQR = Q3 - Q1
    outlier_condition = ((series < (Q1 - factor * IQR)) | (series > (Q3 + factor * IQR)))
    return outlier_condition

outliers = detect_outliers(df['Outage.duration'])
count_outliers = outliers.sum()

print(f'Number of outliers in Outage.duration: {count_outliers}')
```

Number of outliers in Outage.duration: 144

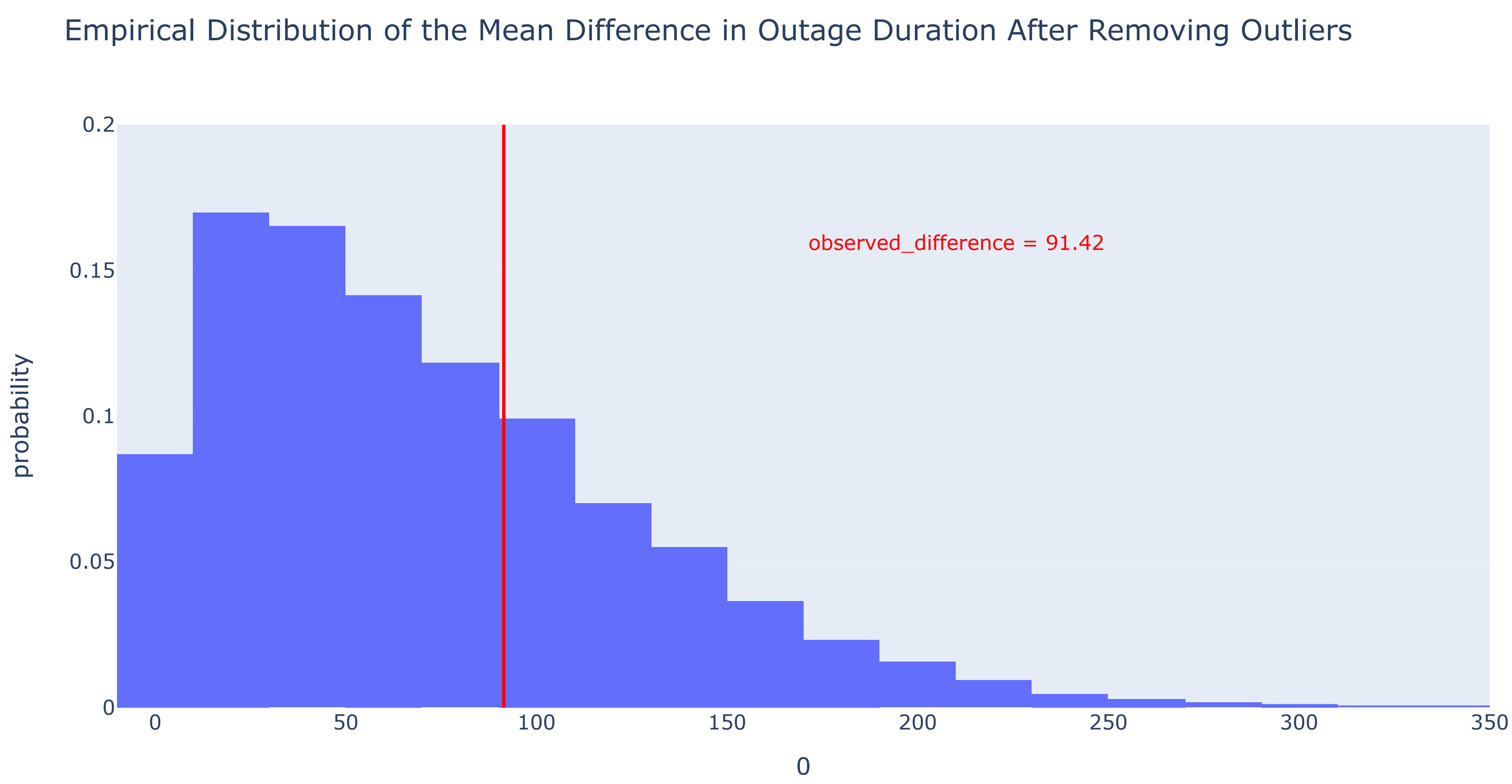
In [27]:
```
# remove outliers and perform permutation test

def remove_outliers(series, factor=1.5):
    Q1 = series.quantile(0.25)
    Q3 = series.quantile(0.75)
    IQR = Q3 - Q1
    return series[~((series < (Q1 - factor * IQR)) | (series > (Q3 + factor * IQR)))]

out = df.copy()
out['Outage.duration'] = remove_outliers(out['Outage.duration'])

n_repetitions = 10000

differences = []
for _ in range(n_repetitions):
    copy = out[['Outage.duration', 'Climate']].copy()
    with_shuffled = copy.assign(Shuffled_duration=np.random.permutation(copy['Outage.duration']))

    group_means = (
        with_shuffled
        .groupby('Climate')
        .mean()
        .loc[:, 'Shuffled_duration']
    )
    difference = abs(group_means.diff().iloc[-1])
    differences.append(difference)

observed_difference = abs(out.groupby('Climate')['Outage.duration'].mean().diff().iloc[-1])

p_value= np.mean([np.abs(diff) >= np.abs(observed_difference) for diff in differences])
p_value
```

Out[27]: 0.3105

In [28]:
```
observed_difference = abs(out.groupby('Climate')['Outage.duration'].mean().diff().iloc[-1])
observed_difference
```

Out[28]: 91.4227077701446

In [30]:
```
hypo_fig = px.histogram(pd.DataFrame(differences), x=0, nbins=30, histnorm='probability',
        title='Empirical Distribution of the Mean Difference in Outage Duration After Removing Outliers')
hypo_fig.add_vline(x=observed_difference, line_color='red')
hypo_fig.add_annotation(text=f'<span style="color:red">observed_difference = {round(observed_difference, 2)}</span>',
                        x=2.3 * observed_difference, showarrow=False, y=0.16)
hypo_fig.update_layout(yaxis_range=[0, 0.2])
hypo_fig.show()

hypo_fig.write_html('hypo.html', include_plotlyjs='cdn')
```



In [ ]: