# Analyze the *Old Faithful Geyser* Data

Ruochen Kong

**Abstract**

Old Faithful Geyser is a popular viewpoint in Yellowstone national park, near which tourists aim to observe its eruption. To better suggests when should a tourist comes to the viewing area, an analysis of the relationship between eruption and waiting time is important. This project aims to investigate this relationship based on a dataset with 272 observations with plots. The result shows a positive correlation between the duration of eruptions and waiting time.

## 1 Introduction

Old Faithful Geyser, located in the southwest section of Yellowstone national park, is the most popular geyser in North America. People from all over the world have come to watch its eruptions, so an accurate method of predicting when it will erupt could significantly benefit the tourists. The first step of developing a prediction model is to analyze the available dataset. The dataset used in this project, collected in 1985 (Härdle et al., 1991; Azzalini and Bowman, 1990), contains 272 observations on 2 variables: (i) the duration of eruptions, and (ii) the interval time between consecutive eruptions. The dataset contains several errored or noisy data points, and thus requires a pre-processing step before plotting and analyzing. After a cleaned dataset is generated, several plots are generated to support interpretation.

## 2 Methods

### 2.1 Pre-Process Data

After briefly reading through the data file, several properties are found: (i) comments are formatted with "#" at the beginning, (ii) eruption times are floats with three decimal places, and (iii) waiting times are two-digit integers. With these properties, a new file with cleaned data is created by checking each line with (i) whether it is a comment, (ii) whether the first 5 characters form a 3 decimal places float, and (iii) whether the last 2 characters form a 2 digit integer. The first run showed that typos exist in line 89 where "l.667" should be "1.667", line 96 where "6O" should be "60", line 207 where "3,333" should be "3.333", line 264 where "41083" should be "4.083", and line 295 where "4.5E3" should be "4.533". Then the final cleaned file is created by applying all these corrections. With the cleaned dataset, Pandas are used to load the data points. Considering that the possibility of two eruptions having exactly the same eruption time and waiting time is extremely low, duplicated data points are dropped.

### 2.2 Determine Outliers

The outliers of the processed dataset are determined by the interquartile range (IQR) method. Let $Q_1$ denote the 25 percentile and $Q_3$ denote the 75 percentile, then the upper and lower fences are calculated as:

$$IQR = Q_3 - Q_1$$
$$upper\ fence = Q_3 + 1.5 * IQR$$
$$lower\ fence = Q_1 - 1.5 * IQR$$

With the calculation, the only outlier of this dataset is find in line 148, which is (2.317,05).

### 2.3 Generate Plots

Three different types of plots are generated with the processed data: (i) 2 histogram plots for eruption and waiting times to show the distributions, (ii) 2 box plots for the two variables without the outlier to futher present the distributions, and (iii) 1 scatter plot to show the relationship between the two variables.

## 3 Results

The plots are generated with the matplotlib package, using functions *hist()*, *boxplot()*, and *scatter()*.

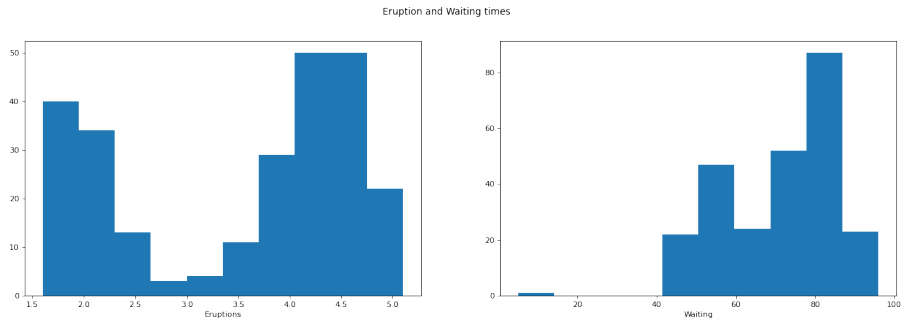



Figure 1: Histograms for eruption durations and waiting times

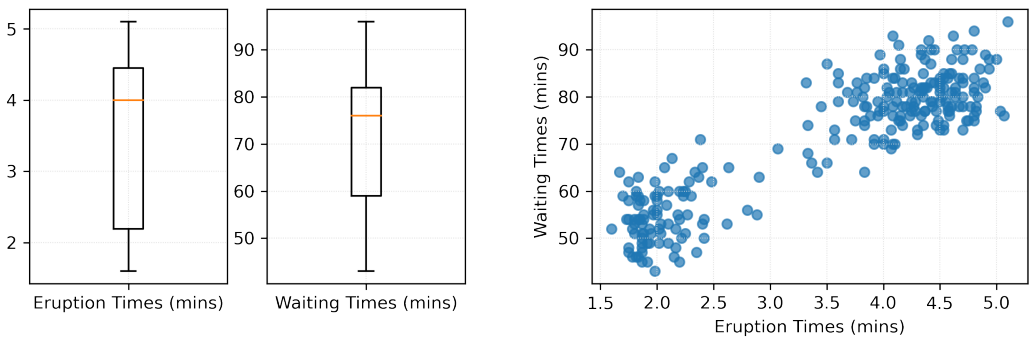



Figure 2: Plots without outliers. The left plots represent the distribution of eruption duration and waiting times, while the right plot shows the relationship between these two variables

## 4 Discussion

From the left part of Figure 1, we can see that the duration time of eruptions are within the range of 1.5 minutes to 5.2 minutes and are distributed densely on the two edges; from the right of the figure, we can see that the waiting time are in range of 40 minutes to 100 minutes except the outlier. Since the outlier locates far away from other data points, I decided not to contain it in the following plots suspecting that it is a noisy data point.

The left two box plots of Figure 2 further present the distribution of the duration time of eruptions and the waiting times, while the right of the figure is a scatter plot that clearly shows two clusters of data points and a positive correlation between the two variables. The correlation and the capability of clustering provide a possibility to develop accurate models for predicting the time or the duration of the next eruption.

## 5 Conclusions

The plots shows a clear relationship between the duration of eruptions and the waiting time, but the detailed relationship still requires further investigation with mathematical or machine learning methods, and developing a prediction model. Although this work only provides superficial analysis among the relationship between the duration of eruptions and the waiting time, its results help in constructing a blueprint of accuratly predict the eruption of Old Faithful Geyser.

## References

Azzalini, A. and Bowman, A. W. (1990). A look at some data on the old faithful geyser, *Journal of the Royal Statistical Society. Series C (Applied Statistics)* **39**(3): 357–365.

Härdle, W. K. et al. (1991). *Smoothing techniques: with implementation in S*, Springer Science & Business Media.

# Appendix: Codes

[1]:
```
import os
import pandas as pd
import matplotlib.pyplot as plt
```

[2]:
```
def process_data():
    f = open('geyser.csv','r')
    fout = open('geyser_cleaned.csv','w')

    i = 0
    for line in f.readlines():

        if '#' in line:
            continue

        if i == 0:
            fout.write(line) # head
            i += 1
            continue

        line = line.replace('l','1').replace('O','0').replace('E','3')
        strnums = [line[:5],line[6:]]
        if '.' not in strnums[0]:
            strnums[0] = strnums[0][:1] + '.' + strnums[0][2:]

        nums = None
        try:
            nums = [float(strnums[0]),int(strnums[1])]
            fout.write('%.3f,%02d\n'%(nums[0],nums[1]))
        except:
            print('Error in line %d: %s'%(i+1,strnums))

        i += 1
    f.close()
    fout.close()
```

[3]:
```
if not os.path.exists('geyser_cleaned.csv'):
    process_data()
data = pd.read_csv('geyser_cleaned.csv').drop_duplicates()
data_describe = data.describe()
data_describe
```

[3]:

| | eruptions | waiting |
|---|---|---|
| count | 256.00000 | 256.000000 |
| mean | 3.49868 | 70.886719 |
| std | 1.12986 | 14.045265 |
| min | 1.60000 | 5.000000 |
| 25% | 2.19575 | 59.000000 |

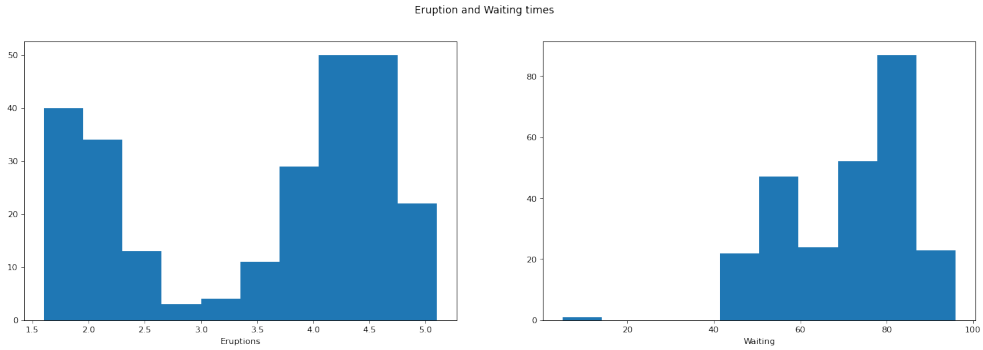
```
50%      4.00000   76.000000
75%      4.45000   82.000000
max      5.10000   96.000000
```

[4]:
```
def fences(Q1,Q3):
    IQR = Q3-Q1
    upper = Q3 + (1.5*IQR)
    lower = Q1 - (1.5*IQR)

    return lower, upper
```

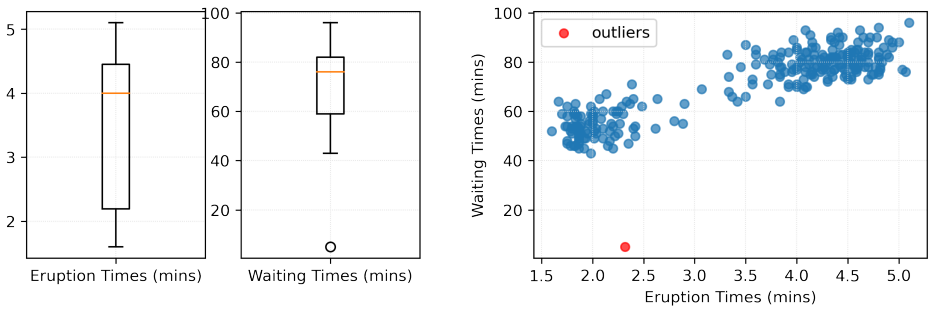
[5]:
```
eruption_fences = fences(data_describe.loc['25%','eruptions'],data_describe.
 ↪loc['75%','eruptions'])
waiting_fences = fences(data_describe.loc['25%','waiting'],data_describe.
 ↪loc['75%','waiting'])

out_erup_1 = data['eruptions'] < eruption_fences[0]
out_erup_2 = data['eruptions'] > eruption_fences[1]

out_wait_1 = data['waiting'] < waiting_fences[0]
out_wait_2 = data['waiting'] > waiting_fences[1]

outliers = data[out_erup_1 | out_erup_2 | out_wait_1 | out_wait_2]
outliers
```

[5]:
```
     eruptions  waiting
116      2.317        5
```

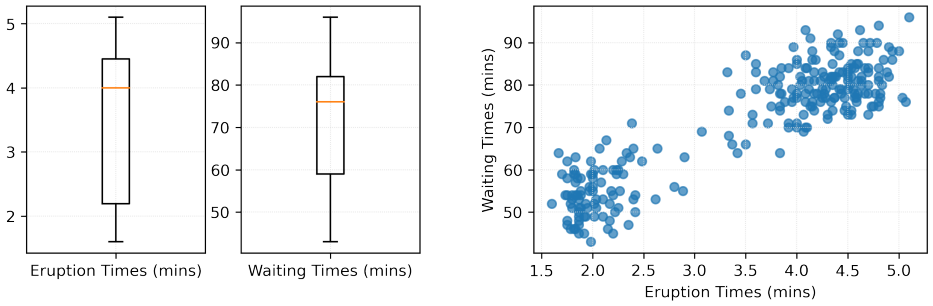
[6]:
```
fig = plt.figure(figsize=(20,6),dpi=80)
ax = fig.subplots(1,2)

ax[0].hist(data['eruptions'],bins=10)
ax[1].hist(data['waiting'],bins=10)
ax[0].set_xlabel('Eruptions')
ax[1].set_xlabel('Waiting')

fig.suptitle('Eruption and Waiting times')
fig.savefig('bad_plot.png')
```




[7]:
```
fig = plt.figure(figsize=(9,3),dpi=300)
fig_box,fig_scatter = fig.subfigures(1,2)

ax_box = fig_box.subplots(1,2)
ax_box[0].boxplot(data['eruptions'], labels = ['Eruption Times (mins)'])
ax_box[1].boxplot(data['waiting'], labels = ['Waiting Times (mins)'])
ax_box[0].grid(color='#DDDDDD', linestyle=':', linewidth=0.5)
ax_box[1].grid(color='#DDDDDD', linestyle=':', linewidth=0.5)

subdata = data.drop(outliers.index)

ax_scatter = fig_scatter.subplots(1,1)
ax_scatter.scatter(subdata['eruptions'],subdata['waiting'], s = 30, alpha = 0.7)
ax_scatter.scatter(outliers['eruptions'],outliers['waiting'], s = 30, alpha = 0.
 ↪7, color = 'r', label = 'outliers')
ax_scatter.set_xlabel('Eruption Times (mins)')
ax_scatter.set_ylabel('Waiting Times (mins)')
ax_scatter.legend()
ax_scatter.grid(color='#DDDDDD', linestyle=':', linewidth=0.5)

plt.subplots_adjust(bottom=0.15)
fig.savefig('good_plot_w.png')
```



[8]:
```
fig = plt.figure(figsize=(9,3),dpi=300)
fig_box,fig_scatter = fig.subfigures(1,2)

ax_box = fig_box.subplots(1,2)
ax_box[0].boxplot(subdata['eruptions'], labels = ['Eruption Times (mins)'])
ax_box[1].boxplot(subdata['waiting'], labels = ['Waiting Times (mins)'])
ax_box[0].grid(color='#DDDDDD', linestyle=':', linewidth=0.5)
ax_box[1].grid(color='#DDDDDD', linestyle=':', linewidth=0.5)



ax_scatter = fig_scatter.subplots(1,1)
ax_scatter.scatter(subdata['eruptions'],subdata['waiting'], s = 30, alpha = 0.7)
ax_scatter.set_xlabel('Eruption Times (mins)')
ax_scatter.set_ylabel('Waiting Times (mins)')
ax_scatter.grid(color='#DDDDDD', linestyle=':', linewidth=0.5)

plt.subplots_adjust(bottom=0.15)
fig.savefig('good_plot_wo.png')
```




[ ]: