Marathon

Aidan Carrier

1. Problem Formulation and Significance

Problem Statement:

The Boston Marathon is one of the most prestigious marathons in the world, but marathon runners as a whole may fail to realize the significance of using machine learning methods to improve their times.

Significance:

Using data from a previous year, we can identify patterns and predictors of better times that could be used in helping future runners achieve faster runs.

Dataset Description:

The data was collected from 2015, and contains 26,598 rows and 25 columns. The columns include split times every 5K, demographics such as age and gender, and overall/gender finish time. All 26,598 rows represent a runner who finished the race.

2. Exploratory Data Analysis

Data Exploration

```
import pandas as pd
import numpy as np
```

```
# Read in data, sometimes "-" is used instead of NA
marathon = pd.read_csv("marathon_results_2015.csv", na_values=["-"])
```

I noticed that a lot of columns didn't use NA values, but instead used "-" to show empty values.

```
print("Shape: ", marathon.shape)
#print("Data Types for each column: ",marathon.dtypes)
#print("NA counts: \n", marathon.isnull().sum())
```

```
Shape:  (26598, 25)
```

```
def time_to_sec(time):
  if pd.isna(time):
    return np.nan
  else:
    time = str(time)
    lst = time.split(":")
    return int(lst[0])*3600 + int(lst[1])*60 + int(lst[2])

time_cols = ['5K', '10K', '15K', '20K', 'Half', '25K',
             '30K', '35K', '40K', 'Official Time']

for col in time_cols:
  marathon[col] = marathon[col].apply(time_to_sec)
```

The values of the split times were in H:MM:SS format, and have to be converted to numeric values to be used in models.

```
marathon = marathon.drop(columns=['Unnamed: 0', 'Bib', 'Name', 'City', 'State', 'Country', '
```

Drop columns with either too many NA values, or not relevant to the problem.

```
marathon.describe().T.round(1)
```

	count	mean	std	min	25%	50%	75%	max
Age	26598.0	42.1	11.3	18.0	33.0	42.0	50.0	82.0
5K	26446.0	1532.1	242.4	883.0	1356.0	1498.0	1658.0	3183.0

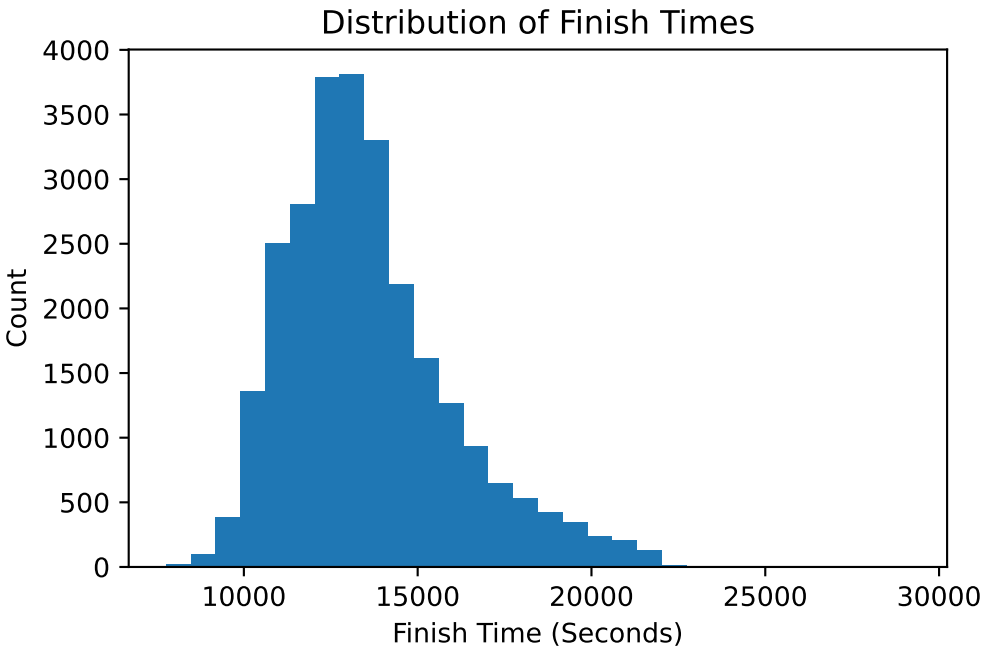
	count	mean	std	min	25%	50%	75%	max
10K	26567.0	3045.6	478.8	1783.0	2702.0	2979.0	3299.0	6436.0
15K	26580.0	4575.8	728.7	2697.0	4057.0	4469.0	4958.0	9705.0
20K	26569.0	6139.5	1001.0	3628.0	5434.0	5985.0	6649.0	13027.0
Half	26570.0	6478.3	1057.5	3841.0	5733.0	6315.0	7017.0	13701.0
25K	26567.0	7729.4	1294.7	4565.0	6823.0	7521.0	8374.0	16566.0
30K	26559.0	9396.8	1622.3	5519.0	8262.0	9125.0	10190.0	20624.0
35K	26547.0	11119.7	1961.9	6479.0	9754.0	10791.0	12074.5	24691.0
40K	26542.0	12834.6	2290.9	7359.0	11255.0	12448.0	13956.8	27688.0
Official Time	26598.0	13585.6	2428.1	7757.0	11918.0	13180.0	14785.0	29161.0

This shows the summary statistics of the numeric data. This can be used to see the means, standard deviations, and distributions of data such as min/max and percentiles.

Data Visualization

```
import matplotlib.pyplot as plt

plt.figure()
plt.hist(marathon["Official Time"], bins=30)
plt.xlabel("Finish Time (Seconds)")
plt.ylabel("Count")
plt.title("Distribution of Finish Times")
plt.show()
```



Here is a plot of the distribution of finish times. It is interesting to see how this is right skewed.

Handling of Missing/Imbalanced Data

```
print("NA counts: \n", marathon.isnull().sum())
```

```
NA counts:
 Age                0
M/F                 0
5K                152
10K                31
15K                18
20K                29
Half               28
25K                31
30K                39
35K                51
40K                56
Official Time       0
dtype: int64
```

```
marathon = marathon.dropna(subset=time_cols)
```

With over 25000 observations, we can simply remove the split times with incomplete values, without it causing too much harm to the model. It is important to note that I am dropping the whole row when removing the NA values, even if a runner only had one incomplete data point.

```
#Check
#print("NA counts: \n", marathon.isnull().sum())
```