Analysis of Chris Brooks' Exercise Data

I have followed the following et al.'s rules among the ten rules.

1. Know Your Audience - The library is explained with necessary definitions and charts so that reader can understand easily
2. Identify Your Message - Introduction and Conclusion part
3. Adapt the Figure to the Support Medium - I have used the Plotly library so that reader can zoom in/out the graph to understand better. Moreover, I used the most straightforward visualization technique as basic plots(scatter, line, and bar) with the necessary feature so that audience can understand instant what the data is representing. The basic plots will help to have an understanding of the relations between variables in the advanced plot.
4. Captions Are Not Optional - I have added a title, legend, and label for each plot so that audience can easily understand the plot.
5. Do Not Trust the Defaults- I changed many features of visualization techniques from default settings based on necessity.
6. Use Color Effectively - Maintained it in every plot to distinguish between variables
7. Get the Right Tool - Unsed Plotly mostly to make plots interactive and matploblib to visualize easily.

Introduction

To explore Chris Brooks' exercise data, I analyzed the data collected over the summer of 2019. The data was collected using various devices, including a heart rate monitor, watch, and bicycle, and was published to the social sharing site Strava. My analysis was performed using Python 3.8.5 and used the following libraries: pandas, numpy, Plotly, and matplotlib.

Due to the unavailability of any health-related information, I assumed Mr. Brooks was a 40-year-old male. There are no pre-existing medical conditions, injuries, or health concerns that may affect his ability to exercise.

I analyzed and visualized some essential columns of the data frame to get insight and provide a recommendation.

In [1]:
```
#Install necessary library
#pip install pandas
#pip install numpy
#pip install matplotlib
#pip install plotly
```

In [2]:
```
#Import necessary library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from typing import Dict
import plotly.express as px
import plotly.graph_objects as go
```

Data Cleaning and Preprocessing

Before analyzing the data, I performed the following cleaning and preprocessing steps:

1. Imported the raw data from Strava into a pandas data frame.
2. Created a new data frame with selected columns. There were two columns named "Cadence" and "cadence". I selected the one with most data.
3. Converted the timestamps to the correct timezone.
4. Generally, a 40-year-old recreational male runner might run at an average speed between 2.24 and 2.98 m/s and cycle at an average speed between 5.81 and 8.05 m/s. But this range varies from person to person depending on fitness level, experience, training, and terrain. So, I considered the running speed range (1, 4.99) and the cycling speed range (5,12) for Mr.Brooks.
5. Created two new DataFrame, "running_df" and "cycling_df," with respective running and cycling data.
6. To summarize and analyze daily data for running and cycling activities, grouped the data by date and calculated the mean for each day, using fillna(df.mean()) function to handle missing data. This approach helped to reduce noise and variations caused by individual activities or events within a single day. It helps track progress and identify patterns.
7. Then, finally merged two data frames to compare performance between running and cycling.

In [3]:
```
#Import raw data from "strave.csv"
df= pd.read_csv(r"C:\Users\nuzha\Downloads\strava.csv")
```

In [4]:
```
# Convert the 'timestamp' column to a datetime object
df['timestamp'] = pd.to_datetime(df['timestamp'])

# Extract the date and time from the 'timestamp' column
df['date'] = df['timestamp'].dt.date
df['time'] = df['timestamp'].dt.time
```

There are 24 columns in the data frame. While avoiding the complexity of analyzing many columns, I used 'cadence', 'enhanced_speed', 'heart_rate,' and 'date' columns for the new DataFrame.

'cadence,' 'enhanced_speed,' 'heart_rate,' and 'date' are helpful as they provide critical insights into the performance and fitness of an individual. Here's a brief explanation of why these columns are relevant:

'cadence': Cadence represents the number of steps (in the running) or pedal strokes (in cycling) per minute. Analyzing cadence can help determine the efficiency of the individual's movement and identify opportunities for improvement in form or technique.

'enhanced_speed': Enhanced speed typically refers to a more accurate speed measurement that considers additional factors like elevation changes and GPS accuracy. This column is essential for understanding the overall performance of Mr.Brooks during his activity and can be used to track progress over time.

'heart_rate': Heart rate is a crucial indicator of the intensity of the exercise and the individual's fitness level. By analyzing heart rate data, I can assess how hard Mr.Brooks works during his activity and monitor his progress.

'date': The date column helps put the data in a temporal context, allowing me to analyze trends and performance changes over time. This information can be valuable for understanding the impact of training programs and identifying patterns in the individual's performance.

In [5]:
```
# Select a subset of columns and create a new copy of the dataframe
df = df[['cadence', 'enhanced_speed', 'heart_rate', 'date']].copy()
```

In [6]:
```
# Convert the 'date' column to a datetime object
df['date'] = pd.to_datetime(df['date'], format='%Y-%m-%d', errors='coerce')
```

In [7]:
```
# Define speed ranges (in m/s) for running and cycling
running_speed_ranges = [(1, 4.99)]
cycling_speed_ranges = [(5, 12)]

# Check if a speed falls within any of the given speed ranges
def is_in_speed_ranges(speed: float, speed_ranges: list) -> bool:
    for lower, upper in speed_ranges:
        if lower <= speed <= upper:
            return True
    return False
```

In [8]:
```
# Filter the data for running and cycling based on speed ranges
running_df = df[df['enhanced_speed'].apply(lambda x: is_in_speed_ranges(x, running_speed_ranges))]
cycling_df = df[df['enhanced_speed'].apply(lambda x: is_in_speed_ranges(x, cycling_speed_ranges))]
```

In [9]:
```
# Calculate daily mean values for heart rate, cadence, and enhanced speed for running data
daily_running_data = running_df.fillna(df.mean()).groupby('date').agg({
    'heart_rate': 'mean',
    'cadence': 'mean',
    'enhanced_speed': 'mean'
})
daily_running_data = daily_running_data.reset_index()

# Calculate daily mean values for heart rate, cadence, and enhanced speed for cycling data
daily_cycling_data = cycling_df.fillna(df.mean()).groupby('date').agg({
    'heart_rate': 'mean',
    'cadence': 'mean',
    'enhanced_speed': 'mean'
})
daily_cycling_data = daily_cycling_data.reset_index()
```

```
C:\Users\nuzha\AppData\Local\Temp\ipykernel_27416\148297740.py:2: FutureWarning: DataFrame.mean and Data
Frame.median with numeric_only=None will include datetime64 and datetime64tz columns in a future versio
n.
  daily_running_data = running_df.fillna(df.mean()).groupby('date').agg({
C:\Users\nuzha\AppData\Local\Temp\ipykernel_27416\148297740.py:10: FutureWarning: DataFrame.mean and Dat
aFrame.median with numeric_only=None will include datetime64 and datetime64tz columns in a future versio
n.
  daily_cycling_data = cycling_df.fillna(df.mean()).groupby('date').agg({
```

In [10]:
```
# Merge the running and cycling dataframes based on date
data = pd.merge(daily_running_data, daily_cycling_data, on='date', how='outer', suffixes=('_running', '_c
```

In [11]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 49 entries, 0 to 48
Data columns (total 7 columns):
 #   Column                  Non-Null Count  Dtype         
---  ------                  --------------  -----         
 0   date                    49 non-null     datetime64[ns]
 1   heart_rate_running      49 non-null     float64       
 2   cadence_running         49 non-null     float64       
 3   enhanced_speed_running  49 non-null     float64       
 4   heart_rate_cycling      10 non-null     float64       
 5   cadence_cycling         10 non-null     float64       
 6   enhanced_speed_cycling  10 non-null     float64       
dtypes: datetime64[ns](1), float64(6)
memory usage: 3.1 KB
```

Exploratory Data Analysis

In order to gain insights into Christopher Brooks' exercise habits, an exploratory data analysis was conducted. Four different visualization techniques were utilized to better understand the data.

Firstly, a scatter plot was created to compare the average speed per day during running and cycling activities. The plot helped us to identify the days where he ran at a slower speed than he cycled, and vice versa.

Secondly, a line plot was generated to understand the trend in average heart rate per day during the time period. This allowed us to understand any patterns or trends in his heart rate during running and cycling activities.

Thirdly, a bar chart was plotted to understand the trend and compare the difference in running and cycling cadence per day. By comparing the running and cycling cadence, we could identify the days where he had a higher cadence during running as opposed to cycling and vice versa.

Finally, a parallel coordinate plot was created to understand the relationship between heart rate, speed, and cadence during running and cycling. This plot allowed us to identify any trends or relationships between the variables.

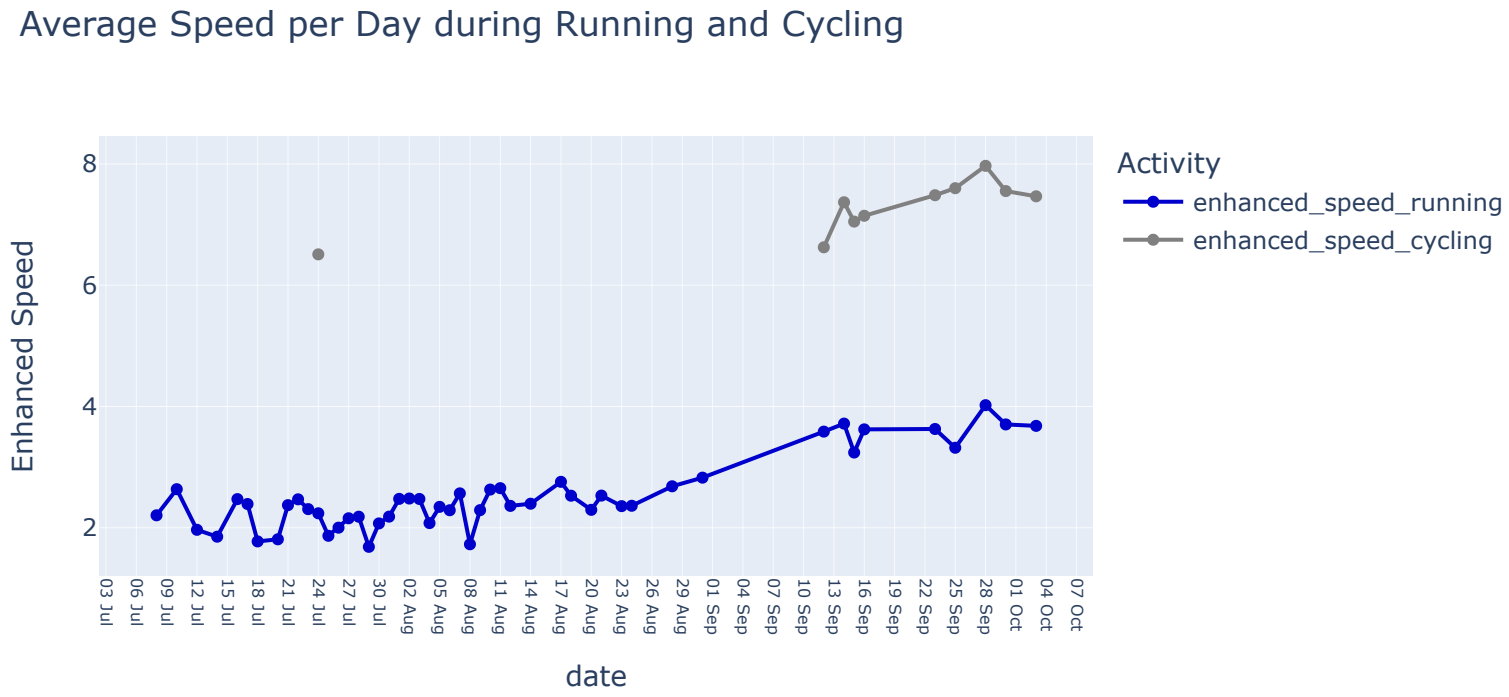
The data was analyzed using Python libraries such as Pandas, Numpy, Matplotlib, Plotly Express, and Plotly Graph Objects. The visualization techniques used were chosen based on their ability to effectively display the data and help us better understand Christopher Brooks' exercise habits.

Scatter Plot

Here I chose Plotly library to plot interactive visualization. So that user can hover over the data points to see more information or zoom in on specific areas of the plot for a closer look. Again, a scatter chart also effectively shows any trends or patterns in the data, such as whether the average speed is increasing or decreasing over time. It also helps to identify if there are any outliers or clusters in the data.

In [12]:

```
# Create a scatter plot to compare the average speed per day during running and cycling activities
fig = px.scatter(data_frame=data,
                 x='date',
                 y=['enhanced_speed_running', 'enhanced_speed_cycling'],
                 labels=dict(variable='Activity', value='Enhanced Speed'),
                 color_discrete_sequence=['mediumblue', 'gray'],
                 title='Average Speed per Day during Running and Cycling',
                 width=800, height=400)

# Update the trace mode to include markers and lines
fig.update_traces(mode='markers+lines')

# Update x-axis properties
fig.update_layout(
    xaxis=dict(
        tickformat='%d %b',
        tickangle=90,
        tickfont=dict(size=8),
        nticks=len(data)  # Set the number of ticks to the number of data points
    )
)

# Show the figure
fig.show()
```



Based on the scatter plot analysis, we can observe Mr. Brooks' average speed differences between running and cycling exercises during the given period. Here's a summary of the findings:

1. Mr. Brooks exhibits a higher average speed during cycling than running. Specifically, his average cycling speed ranges between 6.51 m/s and 7.97 m/s, while his average running speed ranges between 1.68 m/s and 4.02 m/s.
2. It's important to note that a man's typical minimum average running speed is around 2.24 m/s. On days when Mr. Brooks' average speed is below this threshold, he may be walking rather than running.
3. During most days in July and August, Mr. Brooks' average running speed fell between 2 m/s and 3 m/s. After this period, there is a noticeable increase in his running speed. However, given the limited data available, we cannot determine the exact cause of this improvement. Various factors can influence the running speed, such as fitness level, running technique, body composition, terrain, and weather conditions.
4. The data for cycling is limited, with Mr. Brooks cycling sporadically between July and October. Despite the limited data, it appears his cycling performance improved over time.

In conclusion, the scatter plot offers valuable insights into Mr. Brooks' running and cycling performance, highlighting the difference in

Line Chart

I chose Plotly library for this plot because it offers interactive, hover feature and publication-quality graphs and charts. And plotted data with line chart to understand trends and changes over time.

I added MHR zone in this visualization.It is an extra layer of information, helping to analyze the data in terms of heart rate zones.

MHR stands for Maximum Heart Rate, which is the maximum number of times our heart can beat per minute during exercise. One common method for determining MHR is to subtract our age from 220. For example, for 40-year-old Mr.Brooks, his estimated MHR would be 180 beats per minute (220 - 40 = 180).

The five heart rate zones are based on a percentage of maximum heart rate (MHR). Each zone corresponds to a different intensity level and has specific benefits and training purposes. Here's an overview of the five zones:

1. **Zone 1 (50-60% of MHR): Light Intensity** Purpose: Warm-up, cool-down, and active recovery, Benefits: Promotes recovery, improves fat metabolism, and increases overall fitness base

2. **Zone 2 (60-70% of MHR): Moderate Intensity** Purpose: Aerobic endurance training, Benefits: Enhances endurance, increases aerobic capacity, and improves fat-burning efficiency
3. **Zone 3 (70-80% of MHR): Moderate to High Intensity** Purpose: Aerobic conditioning and improving lactate threshold, Benefits: Increases cardiovascular fitness, lactate threshold, and overall endurance
4. **Zone 4 (80-90% of MHR): High Intensity** Purpose: Anaerobic conditioning and improving VO2 max, Benefits: Boosts speed, power, and VO2 max, and enhances anaerobic endurance
5. **Zone 5 (90-100% of MHR) - Maximum Effort/VO2 Max Zone** Purpose: Maximum effort training and improving overall athletic performance, Benefits: Enhances cardiovascular capacity, increases mental and physical tolerance to high-intensity exercise, and improves overall athletic performance

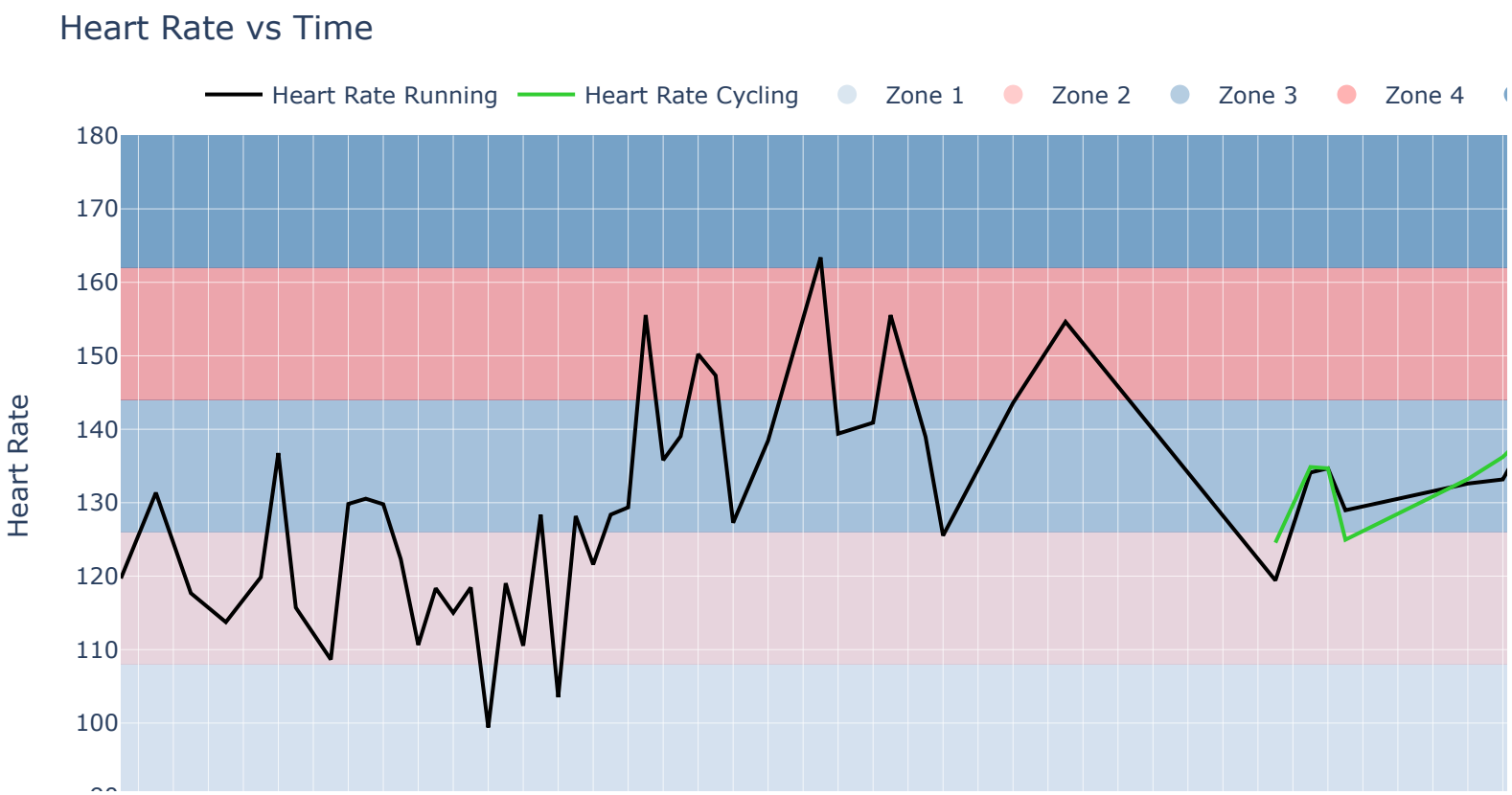
5 MHR zone for Mr.Brooks Zone 1 Range: 90-108 BPM, Zone 2 Range: 108-126 BPM, Zone 3 Range: 126-144 BPM, Zone 4 Range: 144-162 BPM, Zone 5 Range: 162-180 BPM

Reference - https://l.facebook.com/l.php?u=https%3A%2F%2Fmarathonhandbook.com%2Faverage-heart-rate-while-running%2F%3Ffbclid%3DIwAR3g7yDcFk5bS230MPZgpWp3chKZcgU9Sxu8p6ty8VzA_JeOdlT_WeX_gdc&h=AT058RtjdStdEpVz5Ee5 (https://l.facebook.com/l.php?u=https%3A%2F%2Fmarathonhandbook.com%2Faverage-heart-rate-while-running%2F%3Ffbclid%3DIwAR3g7yDcFk5bS230MPZgpWp3chKZcgU9Sxu8p6ty8VzA_JeOdlT_WeX_gdc&h=AT058RtjdStdEpVz5Ee5

In [13]:

```
import plotly.graph_objects as go

fig = go.Figure()

# Plot heart_rate_running
fig.add_trace(go.Scatter(x=data['date'], y=data['heart_rate_running'], mode='lines', name='Heart Rate Run

# Plot heart_rate_cycling
fig.add_trace(go.Scatter(x=data['date'], y=data['heart_rate_cycling'], mode='lines', name='Heart Rate Cyc

# Define heart rate zones
zone_1_min = 90
zone_1_max = 108
zone_2_min = 108
zone_2_max = 126
zone_3_min = 126
zone_3_max = 144
zone_4_min = 144
zone_4_max = 162
zone_5_min = 162
zone_5_max = 180

# Create shaded areas for heart rate zones
for i, (min_val, max_val, color, opacity) in enumerate([(zone_1_min, zone_1_max, "steelblue", 0.1),
                                                        (zone_2_min, zone_2_max, "red", 0.1),
                                                        (zone_3_min, zone_3_max, "steelblue", 0.4),
                                                        (zone_4_min, zone_4_max, "red", 0.3),
                                                        (zone_5_min, zone_5_max, "steelblue", 0.7)]):

    fig.add_shape(type="rect", xref="x", yref="y", x0=data['date'].min(), x1=data['date'].max(),
                  y0=min_val, y1=max_val, fillcolor=color, opacity=opacity, layer="below", line_width=0)

    # Add a scatter trace for the legend
    fig.add_trace(go.Scatter(x=[None], y=[None], mode='markers', marker=dict(size=10, color=color, opacit
                             showlegend=True, name=f'Zone {i+1}'))

fig.update_layout(
    xaxis=dict(
        tickformat='%d %b',
        tickangle=90,
        tickfont=dict(size=8),
        nticks=len(data)  # Set the number of ticks to the number of data points
    ),
```

```
    yaxis=dict(title="Heart Rate"),
    title="Heart Rate vs Time",
    legend=dict(orientation="h", yanchor="bottom", y=1.02, xanchor="right", x=1)
)

fig.show()
```



Based on the line chart analysis, we can observe Mr. Brooks' heart rate zones during his running and cycling exercises in 2019. Here's a summary of the findings:

1. From July to the first week of August, Mr. Brooks' average daily heart rate while running was mostly in Zone 2. This suggests that his primary focus during this period was on aerobic endurance training.
2. From August 5th to August 31st, his average heart rate shifted to Zones 3 and 4. This indicates a change in his training focus towards aerobic conditioning, VO2 max improvement, and lactate threshold enhancement.
3. On days when Mr. Brooks participated in both running and cycling, he was able to maintain a similar average heart rate in Zone 3 for both activities. This implies that his primary benefits during this period were derived from aerobic conditioning and lactate threshold training.
4. Despite the similarities in heart rate zones, the line chart shows that Mr. Brooks' running heart rate data points were slightly higher or equal than his cycling data points. **Karin Sofia Elisabeth Olsson, Hans Rosdahl, and Peter Schantz mentioned in their journal article"Interchangeability and optimization of heart rate methods for estimating oxygen uptake in ergometer cycling, level treadmill walking and running", which reported that oxygen consumption levels for walking are slightly higher (about 5-12%) compared to cycling**. Having more data could help to prove with it.

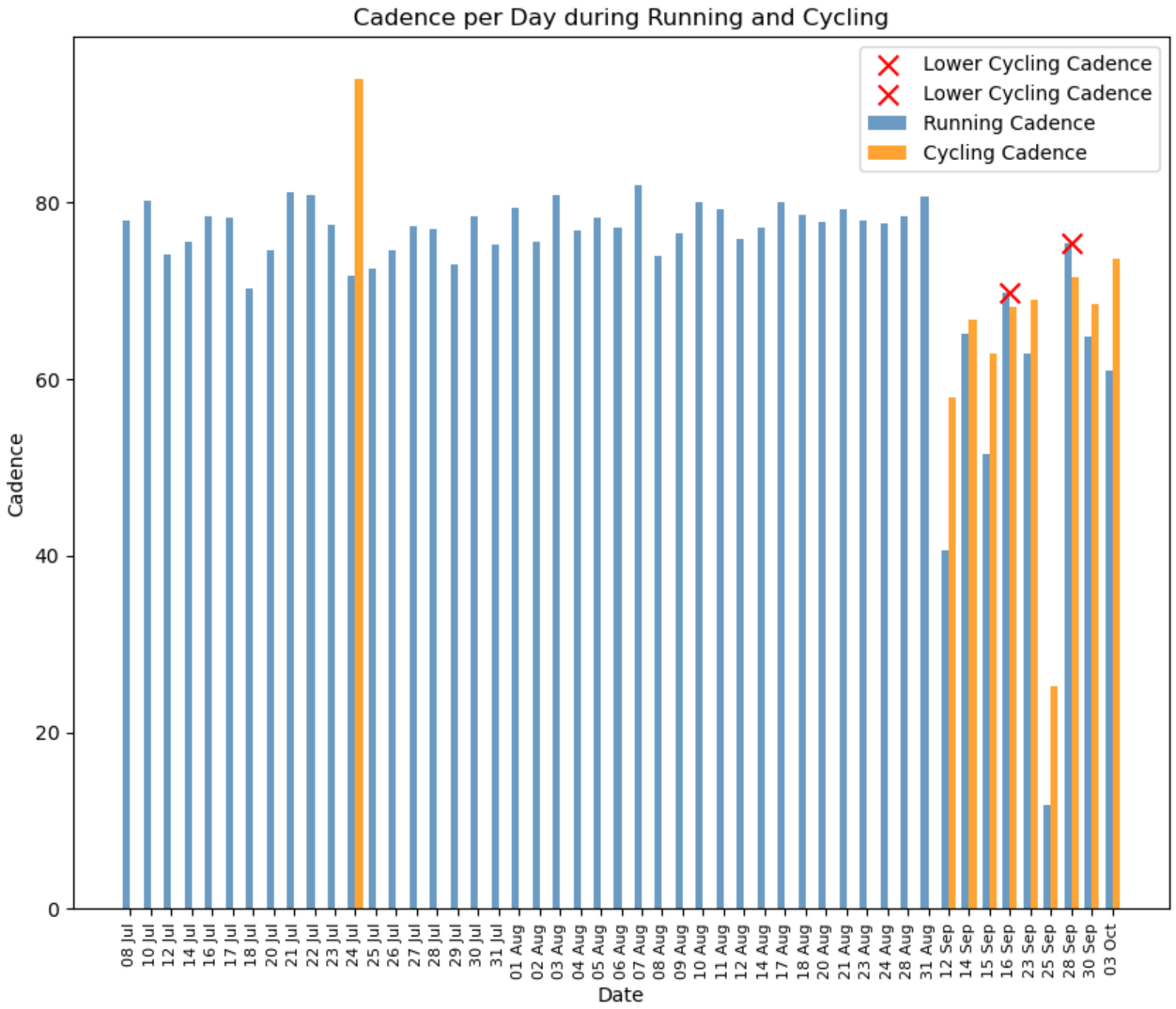
In conclusion, the line chart helps us understand Mr. Brooks' training patterns and heart rate zones over time, revealing his focus on aerobic endurance, aerobic conditioning, and lactate threshold improvement during the analyzed period.

Bar Chart

I chose Matplotlib for this plot cause it is versatile, easy-to-use, and offers a wide range of plotting capabilities, making it a suitable choice for generating the desired visualization.I intend to to present comparison between running cadence and cycling cadence values across different dates.A bar chart allows us to easily compare the values side by side for each date.

In [14]:

```
fig, axs = plt.subplots(figsize=(10,8))

bar_width = 0.35
opacity = 0.8

# Create a bar chart for running cadence
running_cadence_bars = plt.bar(np.arange(len(data['date'])), data['cadence_running'], bar_width, alpha=op

# Create a bar chart for cycling cadence, offset by bar_width
cycling_cadence_bars = plt.bar(np.arange(len(data['date'])) + bar_width, data['cadence_cycling'], bar_wid

# Find dates where cycling cadence is lower than running cadence
lower_cycling_cadence_dates = data[data['cadence_cycling'] < data['cadence_running']]['date']

# Mark those dates on the bar chart with red 'X' markers
for date in lower_cycling_cadence_dates:
    index = data[data['date'] == date].index[0]
    plt.scatter(index + bar_width / 2, data.loc[index, 'cadence_running'], marker='x', color='red', s=100
    #plt.scatter(index + bar_width * 3 / 2, data.loc[index, 'cadence_cycling'], marker='x', color='red', 

# Set x and y axis labels and chart title
plt.xlabel('Date')
plt.ylabel('Cadence')
plt.title('Cadence per Day during Running and Cycling')

# Customize x-axis ticks with dates
plt.xticks(np.arange(len(data['date'])) + bar_width / 2, data['date'].dt.strftime('%d %b'), rotation=90, 

# Add legend to the chart
plt.legend()

# Display the chart
plt.show()
```


Cadence per Day during Running and Cycling
Lower Cycling Cadence
Lower Cycling Cadence
Running Cadence
Cycling Cadence
Cadence
0
20
40
60
80
08 Jul
10 Jul
12 Jul
14 Jul
16 Jul
17 Jul
18 Jul
20 Jul
21 Jul
22 Jul
23 Jul
24 Jul
25 Jul
26 Jul
27 Jul
28 Jul
29 Jul
30 Jul
31 Jul
01 Aug
02 Aug
03 Aug
04 Aug
05 Aug
06 Aug
07 Aug
08 Aug
09 Aug
10 Aug
11 Aug
12 Aug
14 Aug
17 Aug
18 Aug
20 Aug
21 Aug
23 Aug
24 Aug
28 Aug
31 Aug
12 Sep
14 Sep
15 Sep
16 Sep
23 Sep
25 Sep
28 Sep
30 Sep
03 Oct
Date

The bar chart displays the average cadence per day during Mr. Brooks' running and cycling activities. It reveals a few key findings about his cadence patterns:

1. Mr. Brooks' average running cadence is between 70-80 spm, which is significantly lower than the recommended 180 spm by running coach Jack Daniels. However, this recommendation may not be a one-size-fits-all standard, as individual factors influence cadence.
2. Interestingly, Mr. Brooks' running cadence decreased when he started cycling. Typically, cycling can help improve running cadence, but in this case, the opposite occurred.
3. Mr. Brooks' cycling cadence is also lower than the average value of 80-90 spm for a fit cyclist. This suggests that he may not be as efficient in both activities as he could be.
4. The red 'X' markers indicate the dates when the cycling cadence was lower than the running cadence, making it easier to visualize and identify these instances.

From the scatter plot analysis, we found that Mr. Brooks' speed increased on days when he engaged in both running and cycling. The lower cadence in these cases could be attributed to an increase in stride length, which may result in a higher fatigue rate.

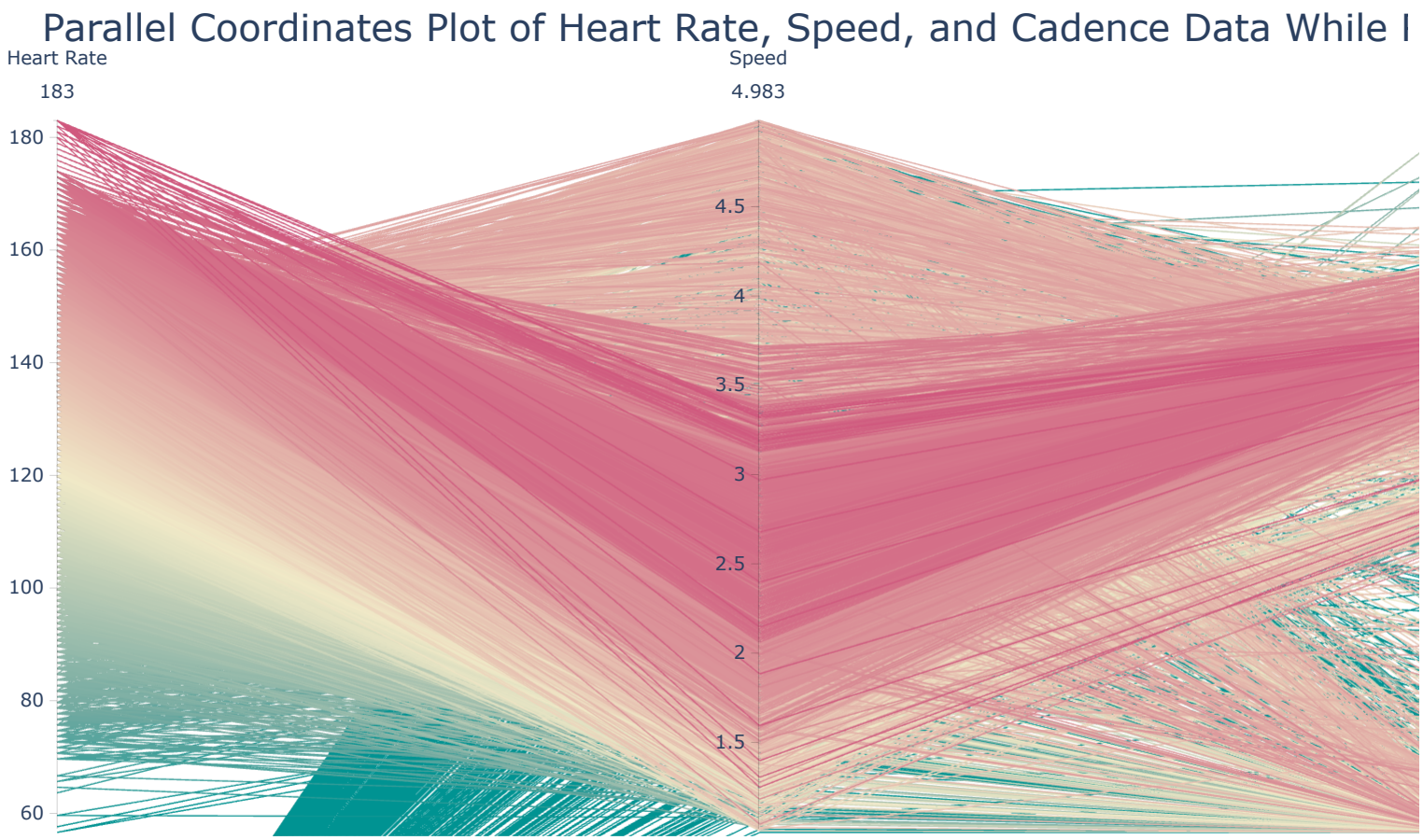
Reference - https://hvmn.com/blogs/blog/training-optimize-running-cadence-to-improve-performance?fbclid=IwAR21eQOJVlmpzGfFPx---sJwsCXW59QQFkK04kujZo0FKOYPWqS29WDmvKE (https://hvmn.com/blogs/blog/training-optimize-running-cadence-to-improve-performance?fbclid=IwAR21eQOJVlmpzGfFPx---sJwsCXW59QQFkK04kujZo0FKOYPWqS29WDmvKE) https://www.trainerroad.com/blog/whats-the-most-efficient-cycling-cadence-and-how-cadence-drills-can-make-you-faster/ (https://www.trainerroad.com/blog/whats-the-most-efficient-cycling-cadence-and-how-cadence-drills-can-make-you-faster/)

Parallel Coordinates Plot

I chose Plotly Express because it is a powerful and interactive visualization library that provides an easy-to-use interface for creating complex plots whenhave a huge number of Data.A parallel coordinates plot was chosen to display the relationship between multiple continuous variables (in this case, heart rate, speed, and cadence) simultaneously. It helps to identify any trends, patterns, or correlations among these variables for the running data.

In [15]:

```
# Create a parallel coordinates plot using Plotly Express
fig = px.parallel_coordinates(running_df, dimensions=['heart_rate', 'enhanced_speed', 'cadence'],
                              labels={'heart_rate': 'Heart Rate', 'enhanced_speed': 'Speed', 'cadence': '
                              color='heart_rate',  # Use the 'heart_rate' column to define the color
                              color_continuous_scale=px.colors.diverging.Tealrose,  # Choose a color scal
                              range_color=[min(running_df['heart_rate']), max(running_df['heart_rate'])])

# Update the layout of the plot
fig.update_layout(
    title='Parallel Coordinates Plot of Heart Rate, Speed, and Cadence Data While Running',  # Set the pl
    title_x=0.5,  # Center the title horizontally
    title_y=.99,
    title_font=dict(size=20)
)

# Show the plot
fig.show()
```



In this specific parallel coordinates plot, the heart rate, speed, and cadence data while running are visualized, with the heart rate values defining the color scale. This color scale helps to identify how the heart rate correlates with speed and cadence and reveals trends or clusters in the data that may provide insights into Mr. Brooks' running performance and efficiency.

The plot provides several insights into Mr. Brooks' running performance:

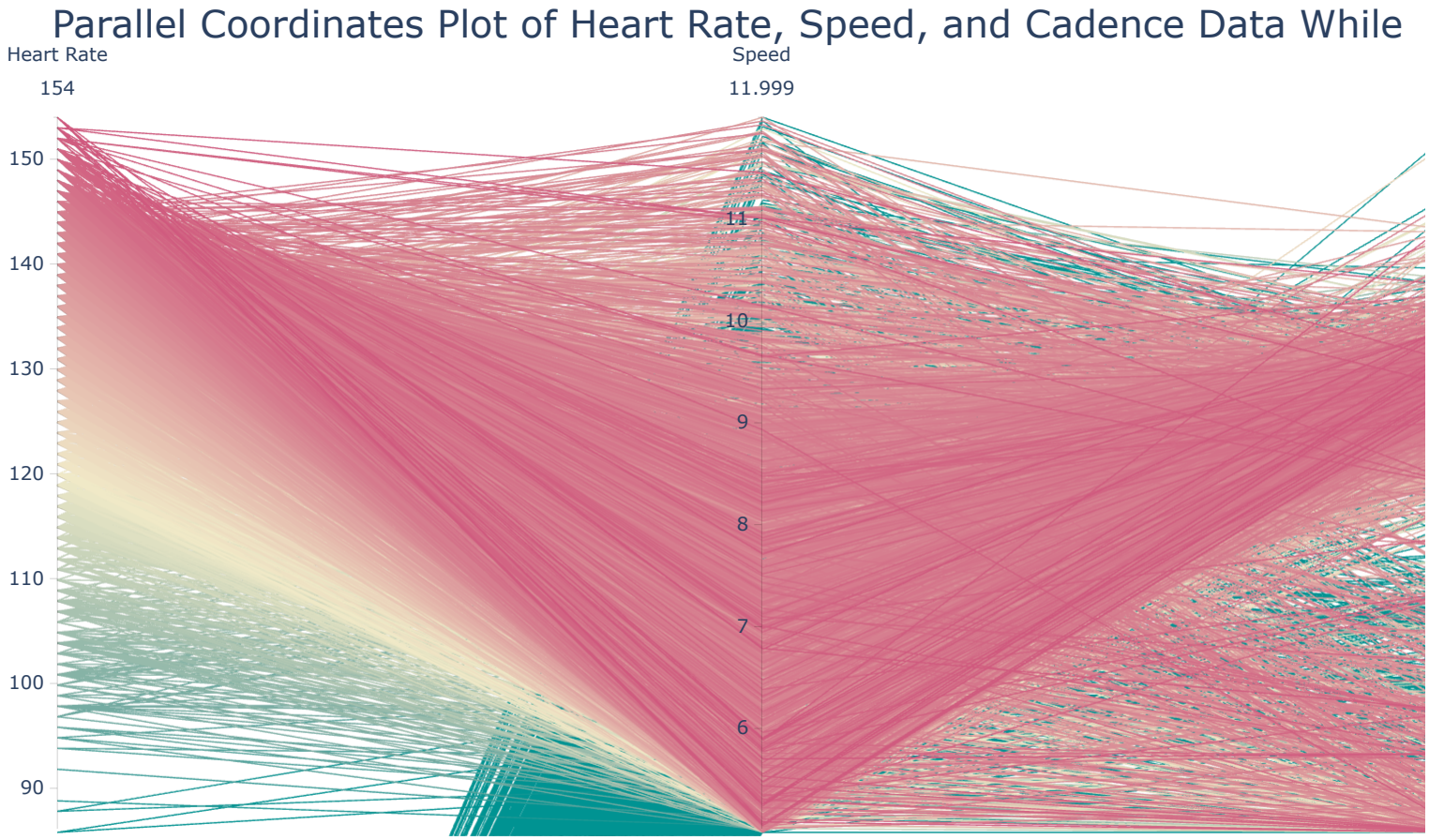
1. Heart rate and speed: There is a mix of positive and negative correlations between heart rate and speed. Mr. Brooks' MHR is highest (in zone 5) when the running speed range is between 2.2 to 3.7 m/s, indicating that cadence should be higher. We can see that cadence is higher for this speed and HR. When the speed is highest for MHR range 3 & 4, it indicates that cadence should be lower. From the graph, it is proven that we have lower cadence for this HR and speed, meaning the stride length is larger, which can lead to injury.
2. Heart rate and cadence: The relationship between heart rate and cadence is less clear, but there seems to be a slight positive correlation. From the bar chart, we concluded that he needs to increase his cadence. Here we can see his HR reaches the maximum threshold to achieve this cadence. As heart rate increases, cadence tends to increase, though not as consistently as with speed. This suggests that cadence plays a role in the intensity of the exercise, but other factors like stride length may also have a significant impact on heart rate.
3. Speed and cadence: The plot indicates a mixed correlation between speed and cadence. However, the relationship is not perfectly linear, suggesting that other factors like stride length may also contribute to variations in speed.
4. Clustering: There are some clusters of data points - cluster 1 (heart rates = zone 5, cadence = (62-92), and speed = (2.2 - 3.7 m/s)), cluster 2 (heart rates = zone 1, cadence = (10-60), and speed = (0 - 4.983 m/s)), and cluster 3 (heart rates = zone 3 & 4,

In [ ]:

I chose Plotly Express because it is a powerful and interactive visualization library that provides an easy-to-use interface for creating complex plots whenhave a huge number of Data.A parallel coordinates plot was chosen to display the relationship between multiple continuous variables (in this case, heart rate, speed, and cadence) simultaneously. It helps to identify any trends, patterns, or correlations among these variables for the running data.

In [134]:

```
# Create a parallel coordinates plot using Plotly Express for cycling_df
fig = px.parallel_coordinates(cycling_df, dimensions=['heart_rate', 'enhanced_speed', 'cadence'],
                              labels={'heart_rate': 'Heart Rate', 'enhanced_speed': 'Speed', 'cadence': '
                              color='heart_rate',  # Use the 'heart_rate' column to define the color
                              color_continuous_scale=px.colors.diverging.Tealrose,  # Choose a color scal
                              range_color=[min(cycling_df['heart_rate']), max(cycling_df['heart_rate'])])

# Update the layout of the plot
fig.update_layout(
    title='Parallel Coordinates Plot of Heart Rate, Speed, and Cadence Data While Cycling',  # Set the pl
    title_font=dict(size=20),
    title_x=0.5,  # Center the title horizontally
    title_y=0.99,  # Increase the gap between the title and the axis labels
)

# Show the plot
fig.show()
```



In this specific parallel coordinates plot, the heart rate, speed, and cadence data whilecycling are visualized, with the heart rate values defining the color scale. This color scale helps to identify how the heart rate correlates with speed and cadence and reveals trends or clusters in the data that may provide insights into Mr. Brooks' cycling performance and efficiency.

The plot provides several insights into Mr. Brooks' running performance:

1. Heart rate and speed: There is a negative correlations between heart rate and speed.Proving when speed is lower his HR gets higher cause his cadence is higher. Vice versa.

2. Heart rate and cadence: The relationship between heart rate and cadence is less clear, but there seems to be a slight positive correlation. For MHR in zone he hets higher cadence.
3. Speed and cadence: The plot indicates a mixed correlation between speed and cadence. However, the relationship is not perfectly linear, suggesting that other factors like stride length may also contribute to variations in speed.
4. Clustering: There are some clusters of data points - cluster 1 (heart rates = zone 4 & 5, cadence = (60-92), and speed = (5.001 - 10 m/s)), cluster 2 (heart rates = zone 1, cadence = (0-10), and speed = (5.01 - 11.99 m/s)), These clusters indicate that Mr. Brooks tends to run at certain combinations of these parameters, which can help understand his preferred running patterns and identify areas for improvement.

Recommendations for Mr. Brooks:

1. Work on improving cadence for both running and cycling by incorporating specific training exercises that focus on increasing stride frequency and maintaining a consistent pace.
2. Consider monitoring the relationship between heart rate and stride rate during exercise, known as "cardiolocomotor synchronization," to improve running efficiency and performance.

In [ ]: