HUANG

March 20, 2021

1 SIADS 521 Assignment 4

Student's name: "Liwen Huang"

Version: 1.2

1.1 Table of Contents

1.2 Description

This assignment is a real-world one, and Professor Chris Brooks is your client! In short, he started increasing his exercise over the summer of 2019 and started collecting data on what he was doing. Throughout the summer he bought a variety of devices (heart rate monitor, watch, bicycle, etc.), and began publishing this data to the social sharing site strava. Your job in this assignment is to explore his strava data dump and say something interesting about it.

1. (20%) Are you making a compelling computational narrative, judged in part by Rule et al's ten rules for computational analyses?
2. (45%) Have you demonstrated that you have a solid grasp of at least three of the basic visual analysis techniques in this class (scatter, box, line, violin, histograms, heatmaps, probability plots, treemaps, sploms) and that they were appropriate for the analysis/data you were investigating?
   - Get equal grades for each plot type (15% each), and grades for a given plot will be broken down into three equal categories (5% each):
     1. The mechanics of generating a reasonable plot from the data you are working with.
     2. The justification for the plot and the insight as a result, as described by your computational narrative.
     3. Making the plot rock visually, by embedding advanced features ranging from the aesthetic (color, form) to the informational (callouts, annotations).
3. (15%) Have you demonstrated that you have a solid grasp of at least one of the more advanced visual analysis techniques in this class (time series, 3d, geographic/mapping, spatial) and that it was appropriate for the analysis/data you were investigating?
4. (20%) Are you able to provide an interesting and defensible analysis that helps Professor Brooks understand what this data means in the context of his activities?

1.3 Introduction

I will use Strava dataset by Mr. Brooks to generate several figures/visualizations to analysis his work-out performances and the amount of exercise roughly.

1.3.1 Explanation of Strava dataset

- Running **cadence** is a number of steps that you will do in one minute. More steps in a minute mean that you are running more effectively.
- Air power, Cadence, Form Power, Ground Time, Leg Spring Stiffness are variables of calculating the quantity of power.
- Altitude is a distance measurement, usually in the vertical or "up" direction, between a reference datum and a point or object.

Units of the data * Cadence: rpm * Ground time: milliseconds * Vertical oscillation: centimeters * Distance, Altitude, and Enhanced Altitude: meters * Longitude and Latitude: semicircles (radians) * Air and Form Power: watts * Leg Spring Stiffness: kN/m * Speed: m/s

1.3.2 Visualization Library

For assignment 4, I will use Plotly and Folium Visualization libraries to generate charts and map.

Plotly graphing library makes interactive, open-source plotting visualization library. It has **user interactive web-based visualizations** that can be displayed in Jupyter Notebooks, saved to standalone HTML files, or served as part of pure Python-build web applications using Dash Plotly charts allow you to hover over values and zoom in/out graphs, like identifying outliers among a large number of data points or detecting anomalies in time series plots.

`folium` makes interactive maps with Python and `folium.map` creates a custom pane to hold map elements.

Installation

- `plotly` may be installed using `pip`:

  `$ pip install plotly==4.14.3`

  or conda:

  `$ conda install -c plotly plotly=4.14.3`

- `folium` may be installed using `pip`: `$ pip install folium` or conda: `$ conda install folium -c conda-forge`

1.4 Demostration

1.4.1 Import Libraries

```
[1]: import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     %matplotlib inline
     import matplotlib as mpl
     import plotly.graph_objects as go
```

```
from plotly.subplots import make_subplots
import chart_studio as py
from plotly import tools
from chart_studio.plotly import plot, iplot
import plotly.express as px
import math
import folium
```

1.4.2 Read csv file and import dataset

I will generate three datasets:

- For general analysis using.
- For map figure using.
- For time-series data using.

```
[2]: def read_csv(filename):
         df_1 = pd.read_csv(filename)
         # Convert timstamp to datetime
         df_1['datetime'] = pd.to_datetime(df_1['timestamp'])
         df_1.drop('timestamp', axis=1, inplace=True)
         # Drop NaN values rows
         df_1.dropna(how='all', inplace=True)
         # Remove useless columns
         columns = ['Cadence', 'unknown_87', 'unknown_88', 'unknown_90',
                    'Air Power', 'Form Power', 'Ground Time', 'Leg Spring␣
     ↪Stiffness',
                    'Vertical Oscillation', 'enhanced_altitude',␣
     ↪'enhanced_speed',
                    'fractional_cadence', 'altitude', 'speed','datafile']

         d = {0:'Monday',1:'Tuesday',2:'Wednesday',3:'Thursday',4:'Friday',5:
     ↪'Saturday',6:'Sunday'}
         df_1['day'] = df_1['datetime'].dt.weekday
         df_1['day'].replace(d, inplace=True)
         df_1.drop(columns, axis=1, inplace=True)
         return df_1
     df = read_csv('assets/strava.csv')
```

1.4.3 General dataset df

```
[3]: df = df[~(df['distance'] == 0)]
     df.reset_index(drop=True, inplace=True)
     df.head()
```

```
[3]:    Power  cadence  distance  heart_rate  position_lat  position_long  \
     0    NaN     54.0      1.32        71.0           NaN            NaN
     1    NaN     77.0     12.19        77.0   504432050.0   -999063637.0
```

```
2    NaN     77.0     14.08        80.0   504432492.0   -999064534.0
3    NaN     77.0     14.08        83.0   504432667.0   -999064622.0
4    NaN     77.0     14.99        83.0   504432736.0   -999064796.0

             datetime     day
0 2019-      21:04     Monday
1 2019-      21:04     Monday
2 2019-      21:04     Monday
3 2019-      21:04     Monday
4 2019-      21:04     Monday
```

1.4.4 Map figure using dataset `df_map`

Convert lat_degrees and lonog_degrees to latitudes and longitudes.

[4]:
```python
df_map = df[['Power','position_lat','position_long']]
df_map["position_lat_degrees"] = df_map["position_lat"] * ( 180 / 2**31 )
df_map["position_long_degrees"] = df_map["position_long"] * ( 180 / 2**31 )

def lat2y(a):
  return 180.0/math.pi*math.log(math.tan(math.pi/4.0+a*(math.pi/180.0)/2.0))

df_map["position_lat_degrees_mercantor"]=df_map["position_lat_degrees"].
 ↪apply(lat2y)
df_map.drop(columns=['position_lat','position_long'], axis=1, inplace=True)

df_map.dropna(inplace=True)
```

```
<ipython-input-4-475e6bc623ff>:2: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

<ipython-input-4-475e6bc623ff>:3: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

<ipython-input-4-475e6bc623ff>:8: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\huang\AppData\Local\Programs\Python\Python39\lib\site-
packages\pandas\core\frame.py:4308: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

<ipython-input-4-475e6bc623ff>:11: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

[5]:
```
df.drop(columns=['position_lat','position_long','Power'], axis=1, inplace=True)
```

1.4.5 Resample time-series dataset df_resample

A time series is a series of data points indexed in time order. Resampling generates a unique sampling distribution on the basis of the actual data.

[6]:
```
df_resample = df.resample('D', on='datetime').mean()
```

1.4.6 Review datasets

Please uncomment below code and review datasets.

[7]:
```
# df.head()
# df_map.head()
# df_resample.head()
```

1.5 Data Cleaning

- Calculate average speed
- Resample data to get mean values to variables
- Remove NaN values

[8]:
```
df_resample.dropna(inplace=True)
```

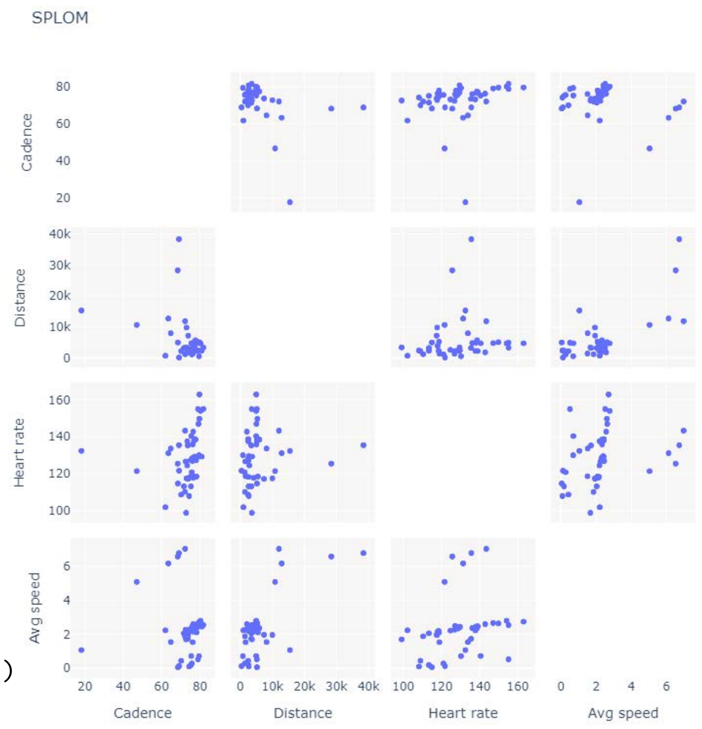
```
[9]: def avg_speed(df, datetime):
         mask = (df['datetime'].dt.year == datetime.year) & (df['datetime'].dt.month␣
      ↪== datetime.month) & (df['datetime'].dt.day == datetime.day)
         df = df[mask]
         min_time = df.iloc[0]['datetime']
         max_time = df.iloc[-1]['datetime']
         duration = (max_time - min_time).total_seconds()
         dist_min = df.iloc[0]['distance']
         dist_max = df.iloc[-1]['distance']
         dist = dist_max - dist_min
         avg_speed = dist / duration
         return duration, dist, avg_speed, min_time, max_time

     values = []
     for i in range(df_resample.shape[0]):
         values.append(avg_speed(df, df_resample.index[i]))
     df_resample['time'] = [x for x, _, _, _, _ in values]
     df_resample['distance'] = [x for _, x, _, _, _ in values]
     df_resample['avg_speed'] = [x for _, _, x, _, _ in values]
     # df_resample
```

1.6 Scatterplot

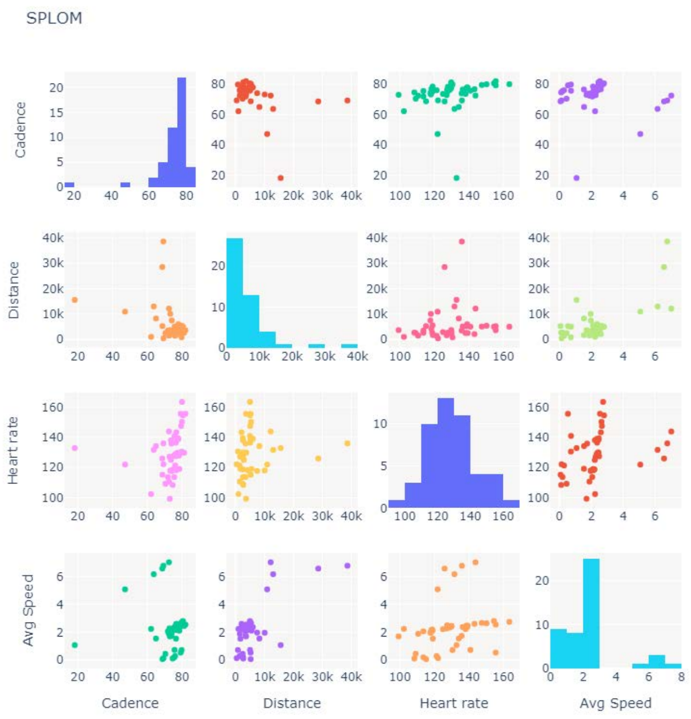
The Scatterplot will be used to explore the main variables from the dataset. It allows us to see where the relationship may be between the variables and visualize the trends.

- A `plotly.graph_objects.Splom` trace is a graph object in the figure's `data` list with any of the named arguments or attirbutes. Splom traces generate scatter plot matrix visualizations. The Plotly splom trace implementation for the scatterplot matrix does not require to set $x = Xi$, and $y = Xj$, for each scatter plot. All arrays, $X_1, X_2, ..., X_n$, are passed once, through a list of dicts called dimensions, i.e. each array/variable represents a dimension.
- The `plotly.figure_factory` module contains dedicated functions for creating very specific types of plots that were at the time of their creation difficult to create with graph objects and prior to the existence of Plotly Express. As new functionality gets added to Plotly.js and to Plotly Express, certain Figure Factories become unnecessary and are therefore deprecated as "legacy", but remain in the module for backwards-compatibility reasons.

```
[10]: fig = go.Figure(data=go.Splom(
          dimensions=[dict(label='Cadence', values=df_resample['cadence']),
                      dict(label='Distance', values=df_resample['distance']),
                      dict(label='Heart rate', values=df_resample['heart_rate']),
                      dict(label='Avg speed', values=df_resample['avg_speed'])],
          # showupperhalf=False,
          diagonal=dict(visible=False)
      ))

      fig.update_layout(
    title='SPLOM',
    width=800,
    height=800,
    plot_bgcolor='rgb(247, 246, 245)'
)

fig.show()
```



[11]:
```
import plotly.figure_factory as ff

df_new = df_resample[['cadence','distance','heart_rate','avg_speed']]
df_new.rename(columns={'cadence':'Cadence','distance':'Distance','heart_rate':
 ↪'Heart rate','avg_speed':'Avg Speed'}, inplace=True)

fig = ff.create_scatterplotmatrix(df_new, diag='histogram', height=800,␣
 ↪width=800)

fig.update_layout(
    title="SPLOM",
    plot_bgcolor='rgb(247, 246, 245)'
)
fig.show()
```



1.7 Line Chart

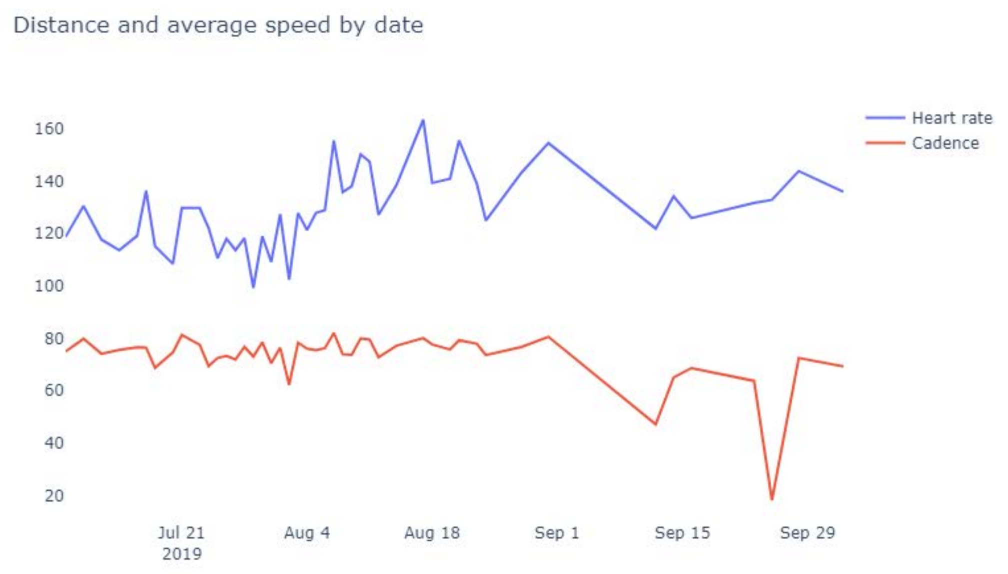
- Use line charts to create the visualization among variables of *Cadence, Heart rate, Distance, and Average Speed.*
- Line charts can be visually interpreted by human and describe relationships with variables.
- Training heart rate would increase your athletic performance and overall level of fitness. To find a steady heart rate – a level at which you feel like you're working hard, but your heart rate doesn't jump up over the time you're trainning.

Brief Summary

From this chart, we can see that the heart rate of Mr. Brooks' heart rate had been changing along with the change of cadences. However, there's really nothing more the chart can talk about.

[12]:
```
trace1 = go.Scatter(
    x = df_resample.heart_rate.index,
    y = df_resample.heart_rate,
    name="Heart rate"
)

trace2 = go.Scatter(
    x=df_resample.cadence.index,
    y=df_resample.cadence,
    name="Cadence"
)
```

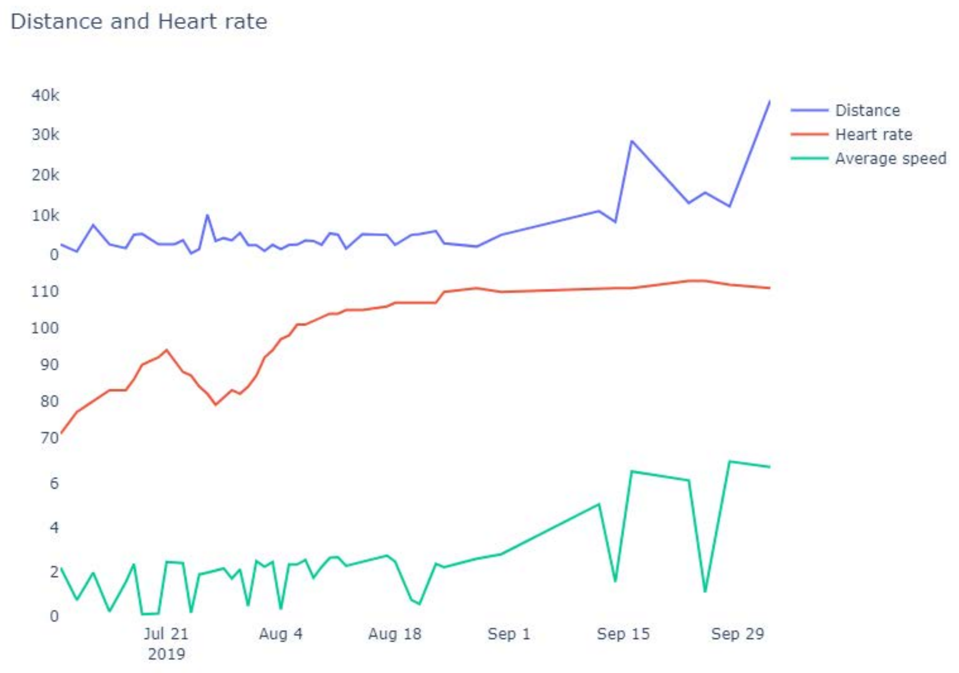
```
# layout = go.Layout(dict(title='Distance by date',␣
↪xaxis=dict(title='Date'),yaxis=dict(title='Distance')))
fig=go.Figure()
fig.add_trace(trace1)
fig.add_trace(trace2)
fig.update_layout(
    height=500, width=800,
    title_text = "Distance and average speed by date",
    plot_bgcolor='white'
)
fig.show()
```



Brief Summary

From this chart, we can tell that the heart rate remains stably while he's been doing work-out continually and after running around the same distance for few months, he changed to do cycling, combined with running after September. This is the reason of the sharp increasing of distance.

[13]:
```
trace1 = go.Scatter(
    x=df_resample.distance.index,
    y=df_resample.distance,
    name="Distance"
)

trace2 = go.Scatter(
    x=df_resample.heart_rate.index,
    y=df.heart_rate,
    name="Heart rate"
)

trace3 = go.Scatter(
    x=df_resample.avg_speed.index,
    y=df_resample.avg_speed,
    name="Average speed"
)

fig = make_subplots(rows=3, cols=1, shared_xaxes=True, vertical_spacing=0.02)

fig.add_trace(trace1, row=1, col=1)
fig.add_trace(trace2, row=2, col=1)
fig.add_trace(trace3, row=3, col=1)

fig.update_layout(height=600, width=800, title_text="Distance and Heart rate",␣
 ↪plot_bgcolor='white')
fig.show()
```



1.8 Bar Chart and Mixed bar-line Chart

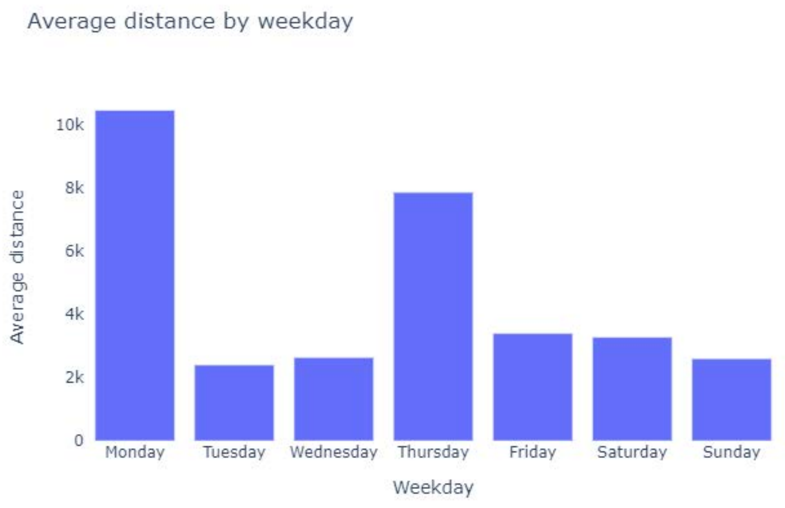
- Bar Chart are used to compare things between differnt groups or to track changes over time.

Brief Summary

From this chart, we can see that Mr. Brooks oftenly work out on Monday and Thursday.

[14]:
```
a = df.groupby('day',as_index=False)['distance'].mean()
b = df.groupby('day', as_index=False)['cadence'].mean()
c = df.groupby('day', as_index=False)['heart_rate'].mean()

weekdays =␣
 ↪['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday']
a['day'] = pd.Categorical(a['day'], categories=weekdays, ordered=True)
a = a.sort_values('day')
b['day'] = pd.Categorical(b['day'], categories=weekdays, ordered=True)
b = b.sort_values('day')
c['day'] = pd.Categorical(c['day'], categories=weekdays, ordered=True)
c = c.sort_values('day')
```

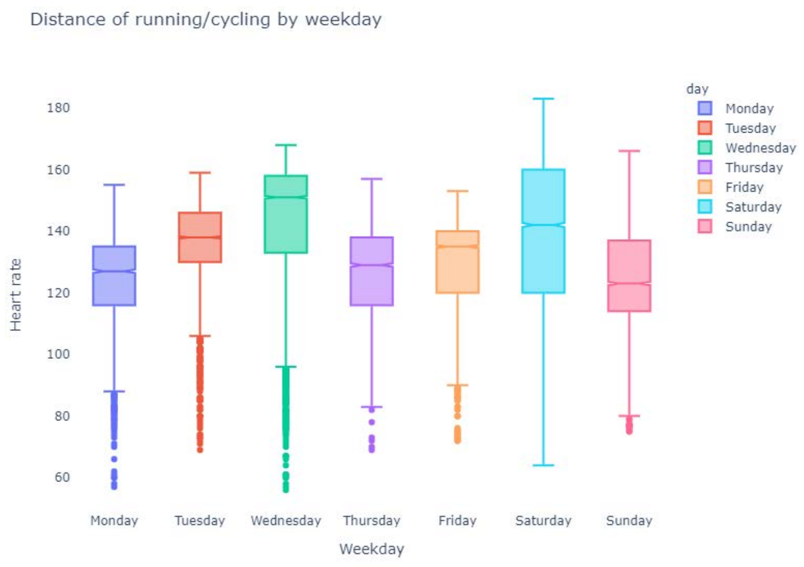
[15]:
```
fig = px.bar(a, x='day',y='distance', title='Average distance by weekday')

fig.update_layout(
    xaxis=dict(title='Weekday'),
    yaxis=dict(title='Average distance'),
    plot_bgcolor='white'
)
fig.show()
```



Brief Summary

This chart tells us that even though Mr. Brooks usually works out on Monday and Thursday, he's running in nearly the same steps in a minute.

[16]:
```
fig = go.Figure()
fig.add_trace(
    go.Bar(
        x=c['day'],
        y=c['heart_rate'],
        name='Heart rate',
        marker_color='pink'
    )
)

fig.add_trace(
    go.Scatter(
        x=b['day'],
        y=b['cadence'],
        name='Cadence',
        marker_color='orange'
    )
)

fig.update_layout(
    xaxis=dict(title='Weekday'),
    title_text='Heart rate and Cadence by weekday',
    plot_bgcolor='white'
)
fig.show()
```



1.10 Map Chart

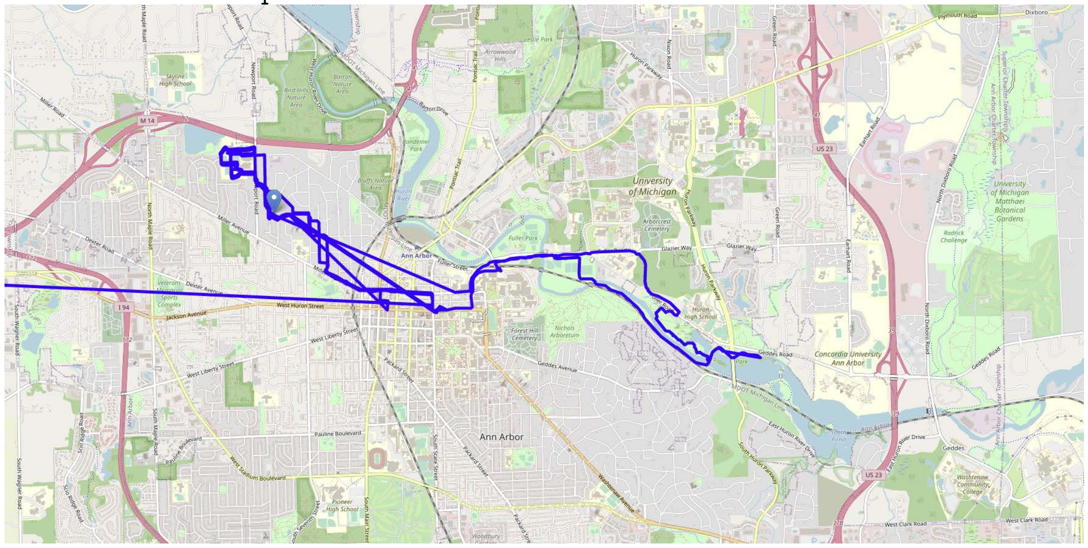
```
[19]: # pip install folium
```

```
[20]: m=folium.Map(location=[42.296,-83.768], zoom_start=15)
      folium.Marker([df_map["position_lat_degrees"].
       ↪iloc[0],df_map["position_long_degrees"].iloc[0]],
                    popup="Start Point").add_to(m)
      folium.Marker([df_map["position_lat_degrees"].
       ↪iloc[-1],df_map["position_long_degrees"].iloc[-1]],
                    popup="Stop Point").add_to(m)
```

```
[20]: <folium.map.Marker at 0x26426329f70>
```

```
[21]: route=folium.
       ↪PolyLine(locations=zip(df_map["position_lat_degrees"],df_map["position_long_degrees"]),
                          weight=5,color='blue').add_to(m)

      display(m)
```

```
<folium.folium.Map at 0x264348d5dc0>
```



1.11 Summary

In a conclusion, Mr. Brooks started to work out in Ann Arbor, MI since July 2019 until now. He usually does work-out on Monday and Thursday. From the scatterplot, I can see that he switched different types of work-out, but I cannot point out which sport he's been doing since I miss the variable of work-out type. Besides that, he's doing good on heart-rate trainning which stays at a stable level, around 130BPM.