CS210 PROJECT

Motivation:

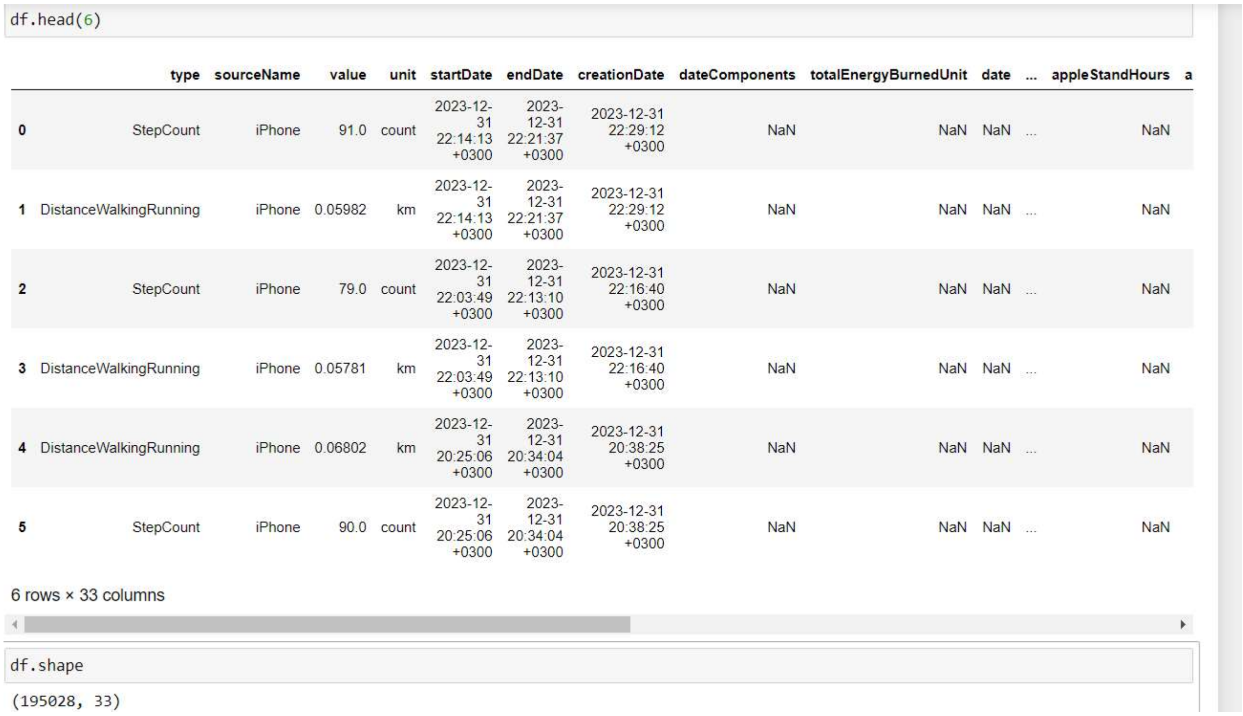
My motivation is to find how pandemic affected my daily walking routine (daily steps). My hypothesis is that I walked less during the pandemic.

Data source:

Health app of my phone that I have used for more than 7 years. I extracted the data as a xml file, then turned it to a csv file.

Data analysis:

It includes three types of movement: step counts, flights climbed and walking/running distance. The records was not divided based on days, they were based on minutes or seconds. There were hundreds of different records per day divided based on different amount of minutes or seconds according to the time period between the start and end of a nonstop movement such as 50 seconds, 5 minutes, 10 minutes, etc. In the beginning it was like that:



```
df.head(6)
```

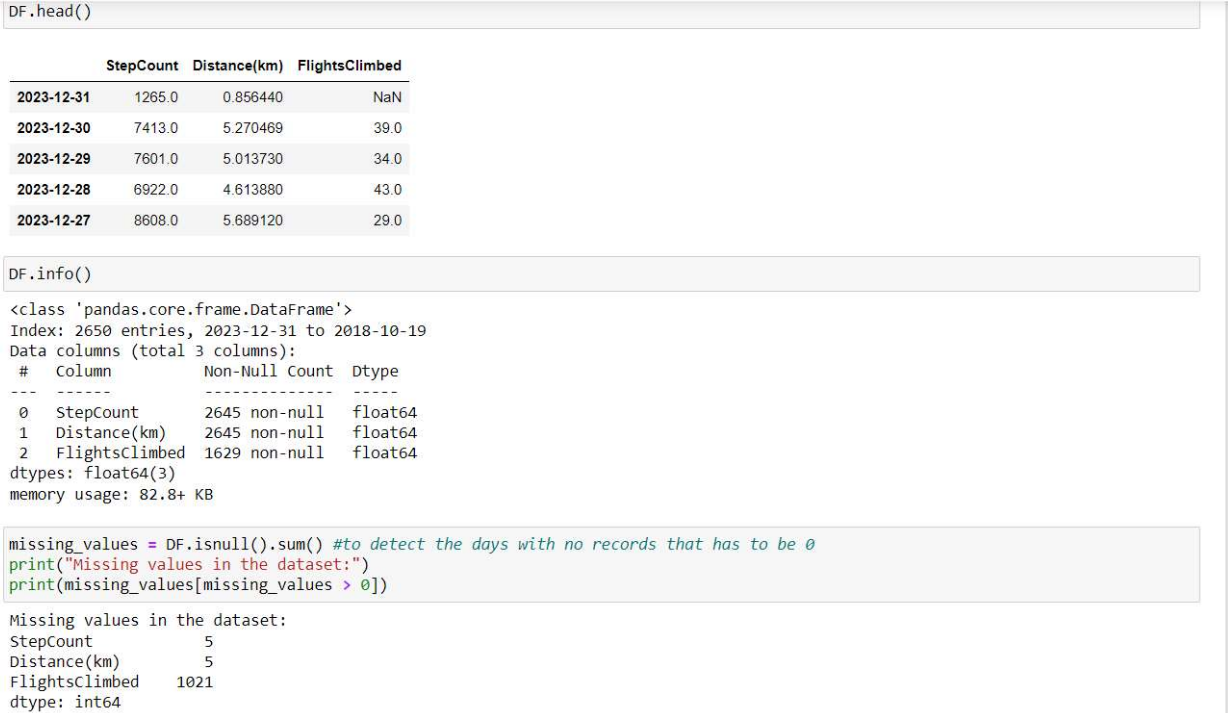
	type	sourceName	value	unit	startDate	endDate	creationDate	dateComponents	totalEnergyBurnedUnit	date	...	appleStandHours	a
0	StepCount	iPhone	91.0	count	2023-12-31 22:14:13 +0300	2023-12-31 22:21:37 +0300	2023-12-31 22:29:12 +0300	NaN	NaN	NaN	...	NaN	
1	DistanceWalkingRunning	iPhone	0.05982	km	2023-12-31 22:14:13 +0300	2023-12-31 22:21:37 +0300	2023-12-31 22:29:12 +0300	NaN	NaN	NaN	...	NaN	
2	StepCount	iPhone	79.0	count	2023-12-31 22:03:49 +0300	2023-12-31 22:13:10 +0300	2023-12-31 22:16:40 +0300	NaN	NaN	NaN	...	NaN	
3	DistanceWalkingRunning	iPhone	0.05781	km	2023-12-31 22:03:49 +0300	2023-12-31 22:13:10 +0300	2023-12-31 22:16:40 +0300	NaN	NaN	NaN	...	NaN	
4	DistanceWalkingRunning	iPhone	0.06802	km	2023-12-31 20:25:06 +0300	2023-12-31 20:34:04 +0300	2023-12-31 20:38:25 +0300	NaN	NaN	NaN	...	NaN	
5	StepCount	iPhone	90.0	count	2023-12-31 20:25:06 +0300	2023-12-31 20:34:04 +0300	2023-12-31 20:38:25 +0300	NaN	NaN	NaN	...	NaN	

6 rows × 33 columns

```
df.shape
```

```
(195028, 33)
```

Then, I categorized them based on days and types of movement and created a new dataframe with the rows are the days and the columns are the types of movement:



```
DF.head()
```

	StepCount	Distance(km)	FlightsClimbed
2023-12-31	1265.0	0.856440	NaN
2023-12-30	7413.0	5.270469	39.0
2023-12-29	7601.0	5.013730	34.0
2023-12-28	6922.0	4.613880	43.0
2023-12-27	8608.0	5.689120	29.0

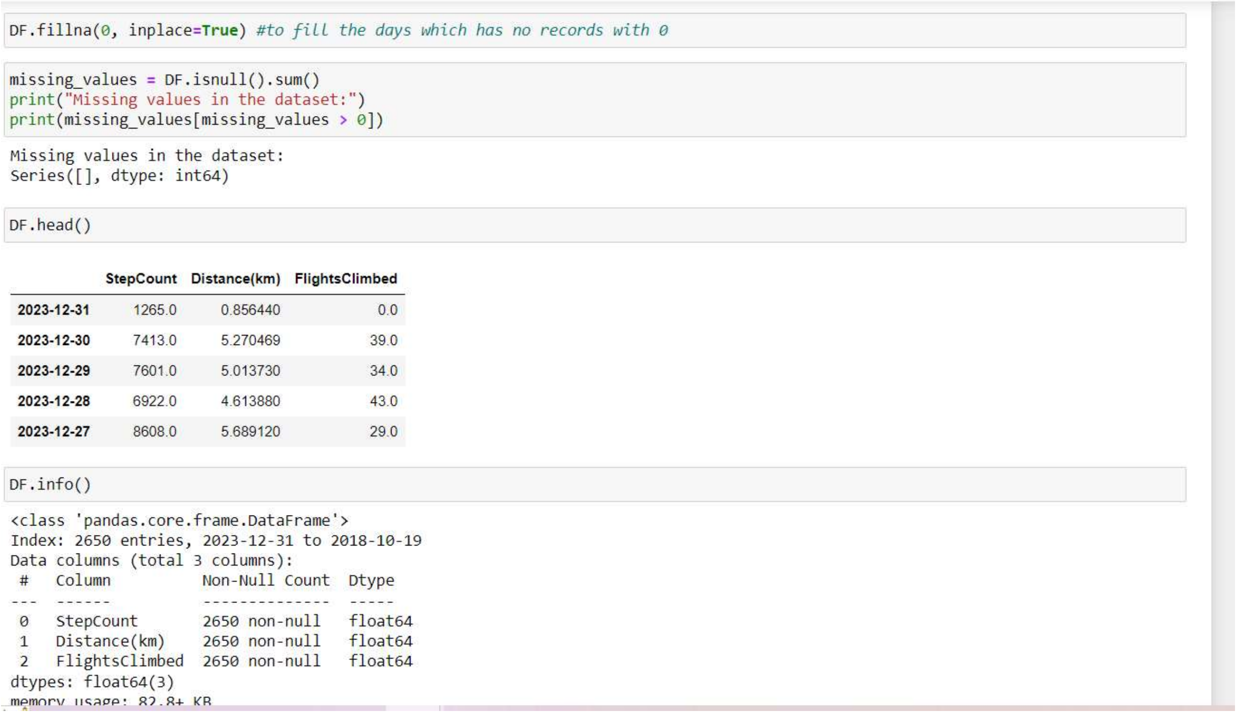
```
DF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2650 entries, 2023-12-31 to 2018-10-19
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   StepCount       2645 non-null   float64
 1   Distance(km)    2645 non-null   float64
 2   FlightsClimbed  1629 non-null   float64
dtypes: float64(3)
memory usage: 82.8+ KB
```

```
missing_values = DF.isnull().sum() #to detect the days with no records that has to be 0
print("Missing values in the dataset:")
print(missing_values[missing_values > 0])
```

```
Missing values in the dataset:
StepCount           5
Distance(km)        5
FlightsClimbed    1021
dtype: int64
```

Then, I realized that there are some missing values with no records which means that they have to be 0 so I filled them with 0:



```
DF.fillna(0, inplace=True) #to fill the days which has no records with 0
```

```
missing_values = DF.isnull().sum()
print("Missing values in the dataset:")
print(missing_values[missing_values > 0])
```

```
Missing values in the dataset:
Series([], dtype: int64)
```

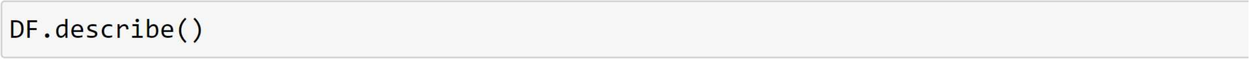
```
DF.head()
```

	StepCount	Distance(km)	FlightsClimbed
2023-12-31	1265.0	0.856440	0.0
2023-12-30	7413.0	5.270469	39.0
2023-12-29	7601.0	5.013730	34.0
2023-12-28	6922.0	4.613880	43.0
2023-12-27	8608.0	5.689120	29.0

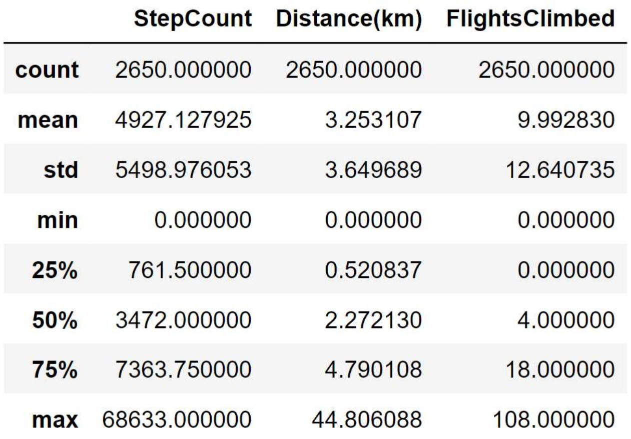
```
DF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2650 entries, 2023-12-31 to 2018-10-19
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   StepCount       2650 non-null   float64
 1   Distance(km)    2650 non-null   float64
 2   FlightsClimbed  2650 non-null   float64
dtypes: float64(3)
memory usage: 82.8+ KB
```

Then, I found the sample mean and the standard deviation:

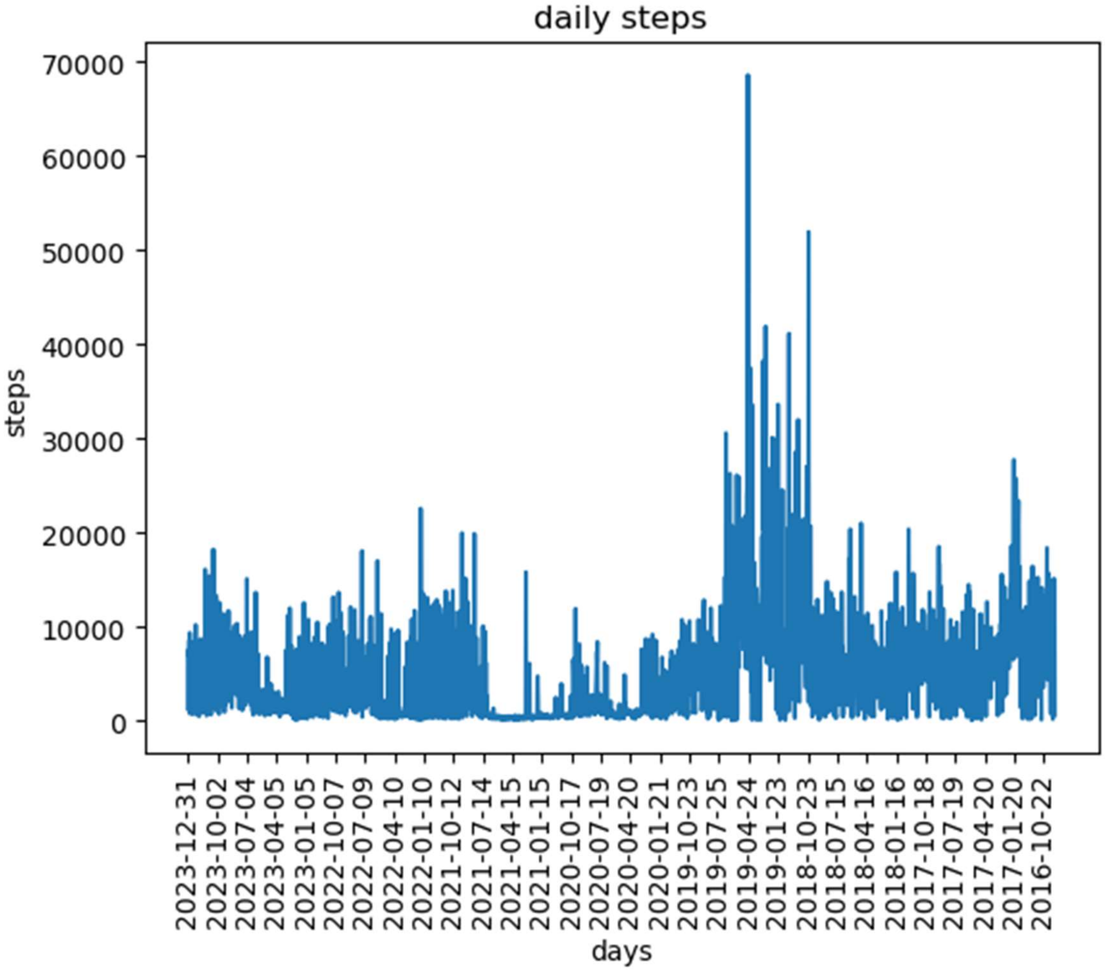


```
DF.describe()
```

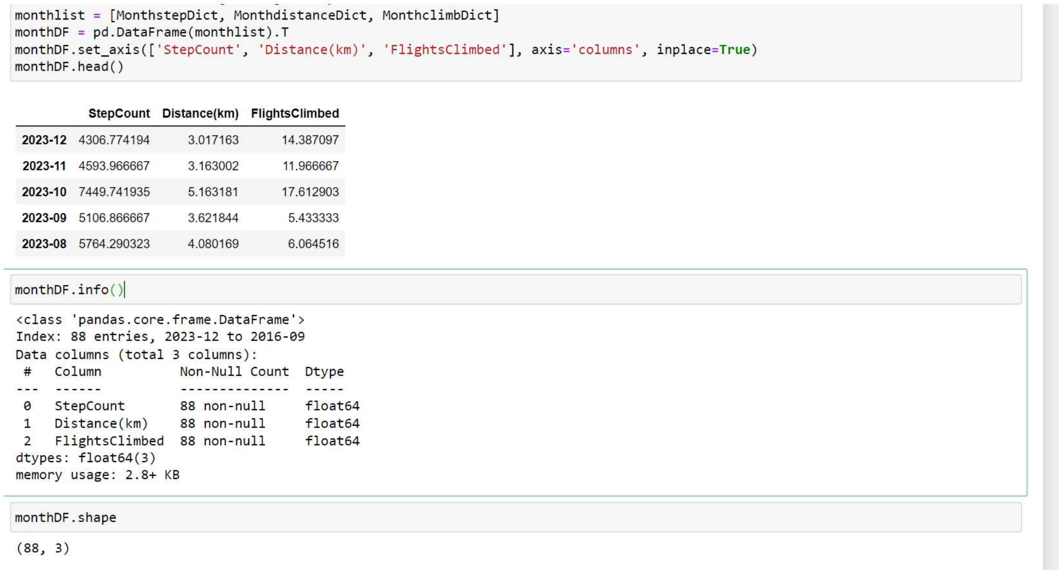


	StepCount	Distance(km)	FlightsClimbed
count	2650.000000	2650.000000	2650.000000
mean	4927.127925	3.253107	9.992830
std	5498.976053	3.649689	12.640735
min	0.000000	0.000000	0.000000
25%	761.500000	0.520837	0.000000
50%	3472.000000	2.272130	4.000000
75%	7363.750000	4.790108	18.000000
max	68633.000000	44.806088	108.000000

Then, I visualized the data by line chart like that:



Then, I created a new dataframe for the average numbers for months with the rows are the months and the columns are the types of movement:



```
monthlist = [MonthstepDict, MonthdistanceDict, MonthclimbDict]
monthDF = pd.DataFrame(monthlist).T
monthDF.set_axis(['StepCount', 'Distance(km)', 'FlightsClimbed'], axis='columns', inplace=True)
monthDF.head()
```

	StepCount	Distance(km)	FlightsClimbed
2023-12	4306.774194	3.017163	14.387097
2023-11	4593.966667	3.163002	11.966667
2023-10	7449.741935	5.163181	17.612903
2023-09	5106.866667	3.621844	5.433333
2023-08	5764.290323	4.080169	6.064516

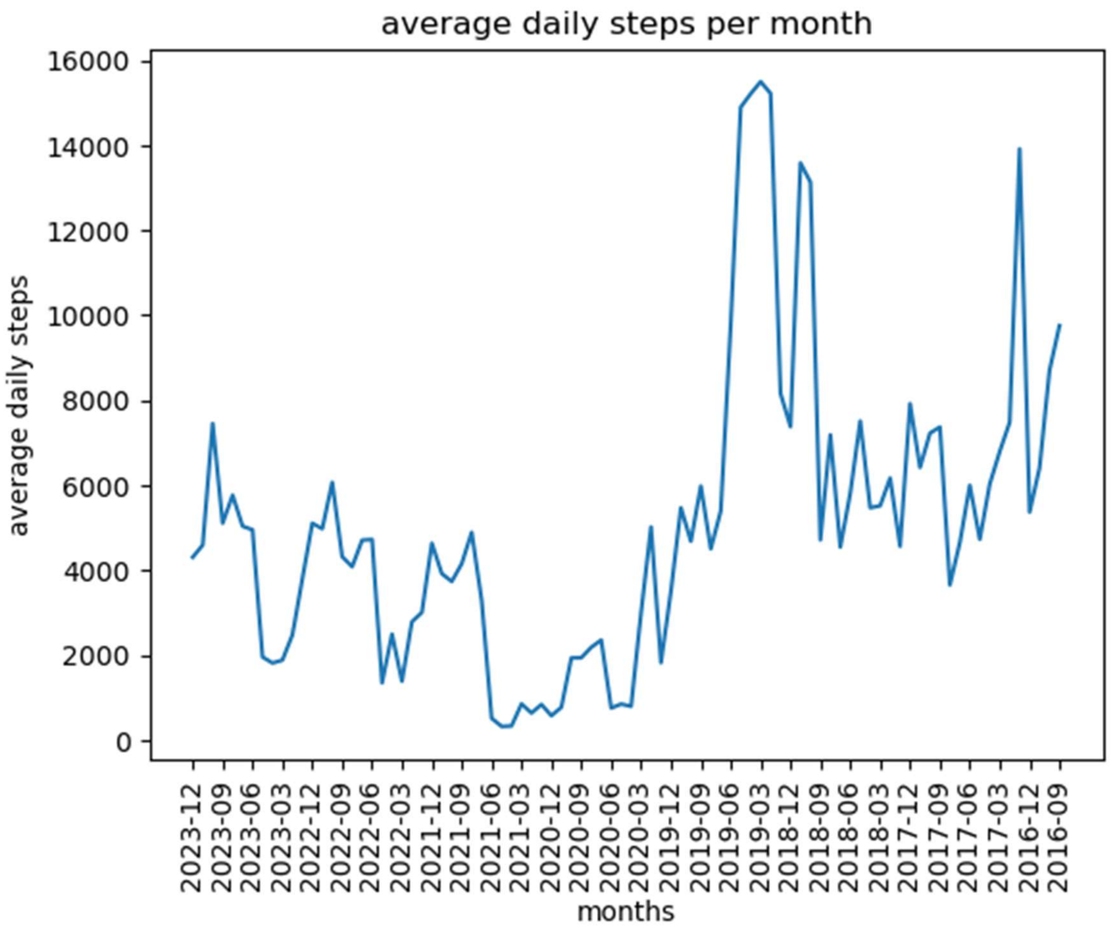
```
monthDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 88 entries, 2023-12 to 2016-09
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   StepCount       88 non-null     float64
 1   Distance(km)    88 non-null     float64
 2   FlightsClimbed  88 non-null     float64
dtypes: float64(3)
memory usage: 2.8+ KB
```

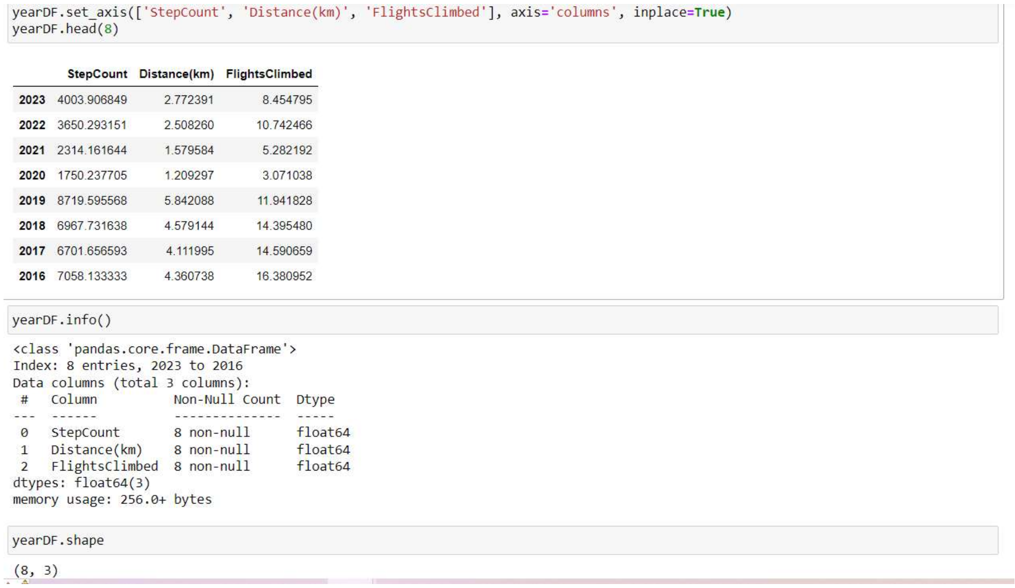
```
monthDF.shape
```

```
(88, 3)
```

Then, I visualized the data by line chart like that:



Then, I created a new dataframe for the average numbers for years with the rows are the years and the columns are the types of movement:



```
yearDF.set_axis(['StepCount', 'Distance(km)', 'FlightsClimbed'], axis='columns', inplace=True)
yearDF.head(8)
```

	StepCount	Distance(km)	FlightsClimbed
2023	4003.906849	2.772391	8.454795
2022	3650.293151	2.508260	10.742466
2021	2314.161644	1.579584	5.282192
2020	1750.237705	1.209297	3.071038
2019	8719.595568	5.842088	11.941828
2018	6967.731638	4.579144	14.395480
2017	6701.656593	4.111995	14.590659
2016	7058.133333	4.360738	16.380952

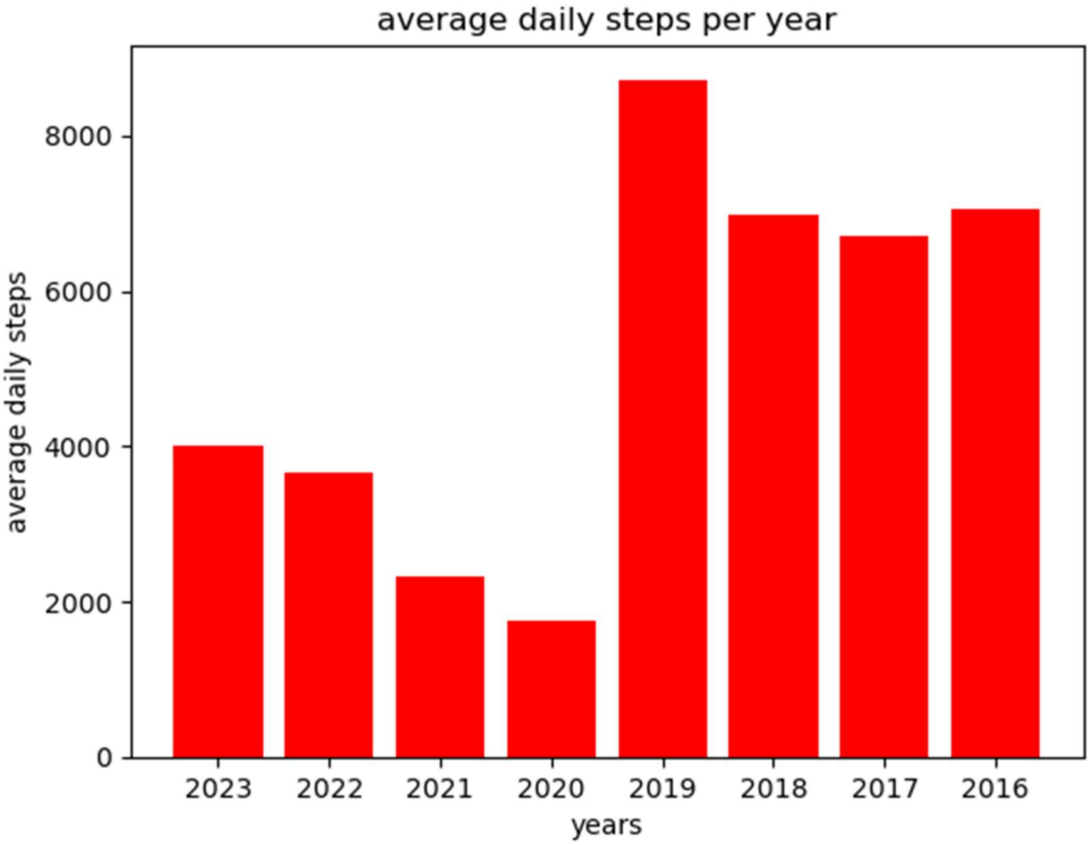
```
yearDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 8 entries, 2023 to 2016
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   StepCount       8 non-null      float64
 1   Distance(km)    8 non-null      float64
 2   FlightsClimbed  8 non-null      float64
dtypes: float64(3)
memory usage: 256.0+ bytes
```

```
yearDF.shape
```

```
(8, 3)
```

Then, I visualized the data by histogram like that:



Findings:

If we examine the last histogram above which shows the average daily step per year, it is obvious that I walked less during the pandemic. Pandemic started at the beginning of 2020 but when it ended is not clear. To show it by hypothesis testing, I assume that pandemic ended at the end of 2020. We had found that in 2020, my average daily step count was 1750. Therefore:

Null Hypothesis: My average daily step (mean) is equal to 1750.

Alternative Hypothesis: My average daily step (mean) is higher than 1750.

We had found that sample mean is 4927 and standard deviation is 5499.

Even though we do not know the variance, since n (total number of days in the data) is higher than 30, we will apply the Z test.

Test statistic Z = (sample mean – mean) / (standardard deviation / sqrt(n))

Observed Z = (4927 – 1750) / (5499 / sqrt(2650)) = 29.74

p value = P(Z > 29.74) = very close to 0 like 0.000000000000000000000000000000000001

If p value is less than the significance level, null hypothesis will be rejected. Even if we assume a low significance level like 0.01, p value is less than the significance level. Therefore null hypothesis is rejected. My average daily step is higher than 1750. On average, I walk more than the average of 2020.

Limitations and Future Work:

Instead of assuming pandemic as only 2020, I could determine it in a more detailed way but it would not change the result of the hypothesis test.

If we examine the charts, it seems like there can be differences between the seasons of the year so I can compare them too.

Özgür Kıyak 31051