In [4]:
```
from IPython.display import clear_output
```

In [5]:
```
%pip install kagglehub
%pip install pandas
%pip install openpyxl
%pip install seaborn
%pip install plotly
%pip install scikit-learn
%pip install geopandas
%pip install "nbformat>=4.2.0"
clear_output()
```

In [6]:
```
import kagglehub
from pathlib import Path
import pandas as pd
import seaborn as sns
import plotly.express as px
import numpy as np
from scipy.stats import skew, kurtosis
import geopandas as gpd
import matplotlib.pyplot as plt
import scipy.stats as stats
```

In [7]:
```
# Download latest version
path = kagglehub.dataset_download("ajitjadhav1/strava-running-activity-data")

try:
    Path(path).rename(r"data\Strava Running Data.xlsx")
except FileExistsError:
    print(
        "Path to dataset files:",
        r"data\Strava Running Data.xlsx\Strava Running Data.xlsx",
    )
```

```
Warning: Looks like you're using an outdated `kagglehub` version, please consider up
dating (latest version: 0.3.5)
Path to dataset files: data\Strava Running Data.xlsx\Strava Running Data.xlsx
Path to dataset files: data\Strava Running Data.xlsx\Strava Running Data.xlsx
```

Exploratory Data Analysis for Machine Learning

Brief description of the data set and a summary of its attributes

The data contains running activity data from March 2022 to December 2023. The data has been downloaded using the Strava API.

The data consists of 105 rows and 19 variables. With 4 float variables, 8 integer variables, and 7 categorical variables.

The key variables are:

- start_date_local The date and time the activity begins.
- type The type of activity (Run, Hike, Walkout, Workout).
- distance The total distance traveled during the activity.
- moving_time The total time of the activity where the individual was moving.
- elapsed_time The total time of the activity.
- total_elevation_gain The total amount of uphill movement during the activity.
- start_latlng The starting latitude and longitude.
- end_latlng The ending latitude and longitude.
- sport_type The type of sport (Run, Hike, Tennis, Walk, Workout).
- start_date The date and tome the activity begins.
- timezone The timezone the activity begins in.
- achievement_count The number of achievements revceived from the activity.
- kudos_count The number of kudos received from other people for the activity.
- comment_count The number of comments received for the activity.
- athlete_count The number of people close enough to be grouped together in the activity.
- photo_count The number of photographs saved to the activity.
- average_speed The average speed during the activity.
- max_speed The maximum speed during the activity.

In [8]:
```
data = pd.read_excel(r"data\Strava Running Data.xlsx\Strava Running Data.xlsx")
data
```

Out[8]:

	Sr. no.	start_date_local	type	distance	moving_time	elapsed_time	total_elevation_gain
0	1	2023-12-09T09:09:19Z	Run	10879.7	4023	4617	91.4
1	2	2023-12-07T17:31:50Z	Run	1304.4	722	62993	0.0
2	3	2023-12-03T09:18:13Z	Run	17503.0	7370	7462	68.4
3	4	2023-12-02T09:41:14Z	Run	3457.8	1791	2170	3.9
4	5	2023-12-01T17:06:05Z	Run	10108.2	4128	4221	6.3
...	...	...	...	...	...	...	...
100	101	2022-06-12T09:20:44Z	Run	7857.2	3900	4113	116.6
101	102	2022-05-28T11:11:38Z	Run	6067.2	3334	3555	88.2
102	103	2022-05-22T07:13:30Z	Run	4587.6	2263	2478	56.6
103	104	2022-05-21T07:27:40Z	Run	4313.6	2182	2426	55.0
104	105	2022-03-16T12:12:54Z	Run	2764.0	955	1009	38.5

105 rows × 19 columns

In [9]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 105 entries, 0 to 104
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Sr. no.               105 non-null    int64
 1   start_date_local      105 non-null    object
 2   type                  105 non-null    object
 3   distance              105 non-null    float64
 4   moving_time           105 non-null    int64
 5   elapsed_time          105 non-null    int64
 6   total_elevation_gain  105 non-null    float64
 7   start_latlng          105 non-null    object
 8   end_latlng            105 non-null    object
 9   sport_type            105 non-null    object
 10  start_date            105 non-null    object
 11  timezone              105 non-null    object
 12  achievement_count     105 non-null    int64
 13  kudos_count           105 non-null    int64
 14  comment_count         105 non-null    int64
 15  athlete_count         105 non-null    int64
 16  photo_count           105 non-null    int64
 17  average_speed         105 non-null    float64
 18  max_speed             105 non-null    float64
dtypes: float64(4), int64(8), object(7)
memory usage: 15.7+ KB
```

In [10]:
```
data.describe()
```

Out[10]:

	Sr. no.	distance	moving_time	elapsed_time	total_elevation_gain	achiever
count	105.000000	105.000000	105.000000	105.000000	105.000000	
mean	53.000000	7067.378095	3554.019048	4753.238095	31.277143	
std	30.454885	5785.399832	2627.200905	6956.141991	50.915702	
min	1.000000	0.000000	85.000000	217.000000	0.000000	
25%	27.000000	3181.300000	1880.000000	2042.000000	0.000000	
50%	53.000000	5266.200000	2564.000000	3058.000000	5.000000	
75%	79.000000	10137.900000	4768.000000	5181.000000	55.000000	
max	105.000000	32192.300000	14536.000000	62993.000000	290.400000	

Initial plan for data exploration

The data will be analyzed over various variables per type. A scatter plot of distance, moving time, elapsed time, total elevation gain, kudos count, athlete count, average speed, and max speed will be compared over the range of dates for each type.

The same variables are used to create histograms to determine the count of each variable across the activite types.

The skewness and kurtosis is evaluated for each variable to see if a transformation may be needed.

With different activity types the data will be explored on a per type basis. Running, walking, hiking, etc. a different activities that will have different results. There can also be missing or incomplete data that will need to be evaluated.

In [11]:
```
cont_cols = (
    "distance",
    "moving_time",
    "elapsed_time",
    "total_elevation_gain",
    "kudos_count",
    "athlete_count",
    "average_speed",
    "max_speed",
)
for col in cont_cols:
    fig1 = px.scatter(data, "start_date", col, color="type")
    fig1.show()
    fig2 = px.histogram(data, "type", col)
    fig2.show()
```

Exploratory Analysis results

The scatter plots and histograms show that there are significantly more running activities than all the other activities combined. The running activities cover a much longer distance, moving time, and elapsed time than the other activities. The running activies typically have aa greater elevation gain than other activities but there is a hiking activity that has by far the greatest amount of elevation gain. Running activities have greater kudos and athelete counts than the other activities. Since running is more of a social event this is to be expected. The average speed and max speed are also greatest for running activities as expected.

Check for skewness and kurtosis. Skewness is the measure of asymmetry of the probability distribution of a continous variable. A skewness of -1 to 1 is considered acceptable. Kurtosis is the measure of tailedness of the probability distribution of a continouse variable. A kurtosis value of -2 to 2.

In [12]:
```
for col in cont_cols:
    print(f"Skewness of {col}:", skew(data[col]))
    print(f"Kurtosis of {col}:", kurtosis(data[col]))
```

```
Skewness of distance: 1.5934895889435066
Kurtosis of distance: 3.252392540915764
Skewness of moving_time: 1.404366354542567
Kurtosis of moving_time: 2.260747478757996
Skewness of elapsed_time: 6.360598252087873
Kurtosis of elapsed_time: 47.5401833619346
Skewness of total_elevation_gain: 2.1649192823144796
Kurtosis of total_elevation_gain: 5.7073612532609665
Skewness of kudos_count: 1.4218115145156738
Kurtosis of kudos_count: 2.390504995033231
Skewness of athlete_count: 7.144311171585414
Kurtosis of athlete_count: 53.301740891221364
Skewness of average_speed: -0.3281567753891057
Kurtosis of average_speed: 6.647197383202823
Skewness of max_speed: 1.5477080073010587
Kurtosis of max_speed: 5.548894957555968
```

Apply Log Transformation to evaluate impact on skewness and kurtosis.

In [13]:
```
for col in cont_cols:
    print(f"Skewness of {col}:", skew(np.log(data[data[col] > 0][col])))
    print(f"Kurtosis of {col}:", kurtosis(np.log(data[data[col] > 0][col])))
```

```
Skewness of distance: -0.732391837424972
Kurtosis of distance: 1.3243355403890913
Skewness of moving_time: -0.9826159268824273
Kurtosis of moving_time: 2.326464261467911
Skewness of elapsed_time: -0.09438798360714662
Kurtosis of elapsed_time: 2.0047473049324616
Skewness of total_elevation_gain: 0.14670236014215177
Kurtosis of total_elevation_gain: -1.5431234170410872
Skewness of kudos_count: 0.5757576562408753
Kurtosis of kudos_count: -0.2764166416483054
Skewness of athlete_count: 3.709570894478009
Kurtosis of athlete_count: 15.957711178731671
Skewness of average_speed: -1.0039616666331685
Kurtosis of average_speed: 7.357739216433007
Skewness of max_speed: 1.1536706878469605
Kurtosis of max_speed: 1.6751444422215576
```

Apply Square Root Transformation to evaluate impact on skewness and kurtosis.

In [14]:
```
for col in cont_cols:
    print(f"Skewness of {col}:", skew(np.sqrt(data[data[col] > 0][col])))
    print(f"Kurtosis of {col}:", kurtosis(np.sqrt(data[data[col] > 0][col])))
```

```
Skewness of distance: 0.6475068973500417
Kurtosis of distance: 0.5824194631416924
Skewness of moving_time: 0.48035441982506044
Kurtosis of moving_time: 0.043862820834660976
Skewness of elapsed_time: 2.962151072940719
Kurtosis of elapsed_time: 14.806947773639155
Skewness of total_elevation_gain: 0.7202721241797174
Kurtosis of total_elevation_gain: -0.5952934016173193
Skewness of kudos_count: 1.1438995920738122
Kurtosis of kudos_count: 1.446491249020875
Skewness of athlete_count: 5.618078598796969
Kurtosis of athlete_count: 35.07519477664853
Skewness of average_speed: 0.6081591254042092
Kurtosis of average_speed: 9.612991003243007
Skewness of max_speed: 1.8381537009549378
Kurtosis of max_speed: 4.3718317187733575
```

Categorical variables will be analyzed by checking the number of unique variables. Plots of the catrgorical variables will include histograms and boxplots.

In [15]:
```
for col in ("type", "sport_type", "timezone"):
    if data[col].dtype == "object":
        print(col, data[col].unique())
```

```
type ['Run' 'Hike' 'Workout' 'Walk']
sport_type ['Run' 'Hike' 'Tennis' 'Walk' 'Workout']
timezone ['(GMT-05:00) America/New_York' '(GMT-04:00) America/Anguilla'
 '(GMT+00:00) GMT']
```

Histogram plots of sport_type show that running is by far the most popular activity. The majority of activities occur in the GMT-5:00 timezone.

In [16]:
```
px.histogram(data, x="sport_type")
```

In [17]:
```
px.histogram(data, x="timezone")
```

Boxplots of the sport_type with elapsed_time and distance are shown. Running as the most varied amount of workouts via time and distance. There are a coupple of extreme data points. Running also is the most likely activity to receive a kudos.

In [18]:
```
px.box(data, x="sport_type", y="elapsed_time")
```

In [19]:
```
px.box(data, x="sport_type", y="distance")
```

In [20]:
```
px.box(data, x="sport_type", y="kudos_count")
```

Actions taken for data cleaning and feature engineering

Each column is confirmed to have no null values. The start_latlng and end_latlng are split to have a start latitude, start longitude, end latitude, and end longitude. There are some rows with empty strings which are removed for generating the routes on a map. The string variables are also converted to floats.

Outliers are identified and removed.

In [21]:
```
data.isnull().any()
```

Out[21]:
```
Sr. no.                 False
start_date_local        False
type                    False
distance                False
moving_time             False
elapsed_time            False
total_elevation_gain    False
start_latlng            False
end_latlng              False
sport_type              False
start_date              False
timezone                False
achievement_count       False
kudos_count             False
comment_count           False
athlete_count           False
photo_count             False
average_speed           False
max_speed               False
dtype: bool
```

In [22]:
```
px.scatter(data, "elapsed_time", "distance", color="type")
```

Outliers

There are 2 data points that are to the extreme for elapsed_time. These data points will be removed.

In [23]:
```
data = data.drop(data[data["elapsed_time"] == 33260].index)
data = data.drop(data[data["elapsed_time"] == 62993].index)
data
```

Out[23]:

	Sr. no.	start_date_local	type	distance	moving_time	elapsed_time	total_elevation_gain
0	1	2023-12-09T09:09:19Z	Run	10879.7	4023	4617	91.4
2	3	2023-12-03T09:18:13Z	Run	17503.0	7370	7462	68.4
3	4	2023-12-02T09:41:14Z	Run	3457.8	1791	2170	3.9
4	5	2023-12-01T17:06:05Z	Run	10108.2	4128	4221	6.3
5	6	2023-11-29T17:22:51Z	Run	6694.7	2913	2983	9.8
...	...	...	...	...	...	...	...
100	101	2022-06-12T09:20:44Z	Run	7857.2	3900	4113	116.6
101	102	2022-05-28T11:11:38Z	Run	6067.2	3334	3555	88.2
102	103	2022-05-22T07:13:30Z	Run	4587.6	2263	2478	56.6
103	104	2022-05-21T07:27:40Z	Run	4313.6	2182	2426	55.0
104	105	2022-03-16T12:12:54Z	Run	2764.0	955	1009	38.5

103 rows × 19 columns

After removing the 2 outliers there is still one extreme value but it will remain in the data set.

In [24]:
```
px.scatter(data, "elapsed_time", "distance", color="type")
```

Graph location of activities.

In [25]:
```
data["start_latlng"] = data["start_latlng"].str.replace("[", "").str.replace("]", "
data["end_latlng"] = data["end_latlng"].str.replace("[", "").str.replace("]", "")
data[["start_lat", "start_long"]] = data["start_latlng"].str.split(",", expand=True
data[["end_lat", "end_long"]] = data["end_latlng"].str.split(",", expand=True)

geo_df = data[data["start_lat"] != ""]
geo_df.loc[:, "start_lat"] = geo_df.loc[:, "start_lat"].astype(float)
geo_df.loc[:, "start_long"] = geo_df.loc[:, "start_long"].astype(float)
geo_df.loc[:, "end_lat"] = geo_df.loc[:, "end_lat"].astype(float)
geo_df.loc[:, "end_long"] = geo_df.loc[:, "end_long"].astype(float)
```

In [26]:
```
color_scale = [(0, "orange"), (1, "red")]

fig = px.scatter_mapbox(
    geo_df,
    lat="start_lat",
    lon="start_long",
    color="type",
    color_continuous_scale=color_scale,
    size="distance",
    zoom=8,
    height=800,
    width=800,
)
for i, row in geo_df.iterrows():
    fig.add_trace(
        px.line_mapbox(
            lat=[row["start_lat"], row["end_lat"]],
            lon=[row["start_long"], row["end_long"]],
            color_discrete_sequence=["red"],
        ).data[0]
    )
fig.add_trace(
    px.scatter_mapbox(geo_df, lat="end_lat", lon="end_long", size="distance").data[
)
fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(margin={"r": 0, "t": 0, "l": 0, "b": 0})
fig.show()
```

Key Insights and Findings

The most popular activity is running. It is no surprise that the running activities have the longest moving times and distance. The majority of activites are in the New York City areas.

The months of January and February have the least amount of activities.

There are not a log of hiking activities but the largest elevation gain belongs to a hiking activity.

April 27, 2022 has the highest athlete count which indicates that a number of runners were near each other indicating this may have been a race of a group run. This date also has the highest max speed and average speed.

3 Hypothesis

1. The longer the moving time the larger the distance traveled is.
2. The higher the athlete count the faster the average speed is.
3. The higher the kudos count the larger the distance traveled is.

Hypothesis test number 3 will be tested. A higher kudos count will be considered as any activity with 2 or more kudos.

$H_0 : \mu \leq x$

$H_a : \mu > x$

In [27]:
```
large_kudos = geo_df[geo_df["kudos_count"] >= 2]
small_kudos = geo_df[geo_df["kudos_count"] < 2]


t_stat, p_value = stats.ttest_ind(
    a=large_kudos["distance"],
    b=small_kudos["kudos_count"],
    equal_var=False,
    alternative="greater",
)
if p_value < 0.05:
    print(
        "Total distance of high kudos activities is significantly greater than thos
    )
else:
    print("There is no significant difference between the samples.")
print(f"The pvalue is {p_value}.")
```

```
Total distance of high kudos activities is significantly greater than those will les
s than 2 kudos.
The pvalue is 2.867044127885265e-08.
```

Hypothesis Test Results

The ttest with confidence interval of 95% rejects the null hypothesis that the distance traveled per activity has no impact on the number of kudos received. Since the pvalue is $2.86e^{-8} which is less than 0.05 we can reject the null hypothesis. The number of kudos receievd for an activity based on the distance traveled is statistically significant.

Next Steps

A next step could be to predict if an activity would receive a kudos based on location, distance, avg speed, elevation gain or any other variable. Route suggestions could be created based on requested start location, distance, and elevation gains.

Conclusion

The dataset is suitable for this objective but a larger dataset would allow for more exploration. The data is mostly centered around New York City with only a few activities. With data from throughout the United States or the world would allow for a better comparison of running styles and abilities.