fit-globe-t2

August 3, 2024

[1]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

df = pd.read_csv(r'/content/fitness_tracker_dataset.csv')

df.head()
```

[1]:

	user_id	date	steps	calories_burned	distance_km	active_minutes	sleep_hours	heart_rate_avg	workout_type	weather_conditions	location	mood
0	468	2023-01-01	4530	2543.02	16.10	613	1.5	176.0	Walking	Clear	Park	Tired
1	879	2023-01-01	11613	1720.76	8.10	352	6.3	128.0	Cycling	Fog	Park	Happy
2	152	2023-01-01	27335	1706.35	3.57	236	6.7	134.0	Yoga	Snow	Park	Neutral
3	311	2023-01-01	13459	2912.38	6.41	1329	11.6	116.0	Swimming	Rain	Office	Tired
4	759	2023-01-01	15378	3344.51	17.88	52	7.4	84.0	Swimming	Rain	Office	Neutral

[2]:
```
df.describe()
```

[2]:

	user_id	steps	calories_burned	distance_km
count	376576.000000	376576.000000	376576.000000	376576.000000
mean	500.062811	15023.867434	2749.063799	9.984541
std	287.835191	8652.909781	721.393353	5.776236

```
min         1.000000       0.000000      1500.000000       0.000000
25%       251.000000    7518.000000      2125.700000       4.970000
50%       501.000000   15053.000000      2745.460000      10.000000
75%       749.000000   22518.000000      3373.292500      14.970000
max       999.000000   29999.000000      4000.000000      20.000000

       active_minutes    sleep_hours  heart_rate_avg
count   376576.000000  376576.000000   376575.000000
mean       719.740913       6.000922      119.430578
std        416.032043       3.462251       34.653814
min          0.000000       0.000000       60.000000
25%        359.000000       3.000000       89.000000
50%        720.000000       6.000000      119.000000
75%       1080.000000       9.000000      149.000000
max       1439.000000      12.000000      179.000000
```

[4]:
```
df.info()
print(df.isnull().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 376576 entries, 0 to 376575
Data columns (total 12 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   user_id             376576 non-null  int64
 1   date                376576 non-null  object
 2   steps               376576 non-null  int64
 3   calories_burned     376576 non-null  float64
 4   distance_km         376576 non-null  float64
 5   active_minutes      376576 non-null  int64
 6   sleep_hours         376576 non-null  float64
 7   heart_rate_avg      376575 non-null  float64
 8   workout_type        322717 non-null  object
 9   weather_conditions  376575 non-null  object
 10  location            376575 non-null  object
 11  mood                376575 non-null  object
dtypes: float64(4), int64(3), object(5)
memory usage: 34.5+ MB
user_id                  0
date                     0
steps                    0
calories_burned          0
distance_km              0
active_minutes           0
sleep_hours              0
heart_rate_avg           1
workout_type         53859
```

```
weather_conditions        1
location                  1
mood                      1
dtype: int64
```

Data Preprocessing

[20]:
```
df.dropna(inplace=True)
print(df.isnull().sum())
new_df = df.drop(columns=['date', 'location'],axis = 1)
print(new_df)

# Use .replace() to map values in the columns
mapping_workout = {'Walking': 1, 'Cycling': 2, 'Swimming': 3, 'Yoga': 4,'Gym␣
 ↪Workout':5,'Running':6}
mapping_weather = {'Clear': 0, "Fog": 1, 'Snow': 2, 'Rain': 3}
mapping_mood = {'Stressed': 0, "Happy": 1, 'Neutral': 2, 'Tired': 3}

new_df['workout_type'] = new_df['workout_type'].replace(mapping_workout)
new_df['weather_conditions'] = new_df['weather_conditions'].
 ↪replace(mapping_weather)
new_df['mood'] = new_df['mood'].replace(mapping_mood)

print(new_df)
new_df.info()
new_df.isnull().sum()
print(new_df['workout_type'].unique())
```

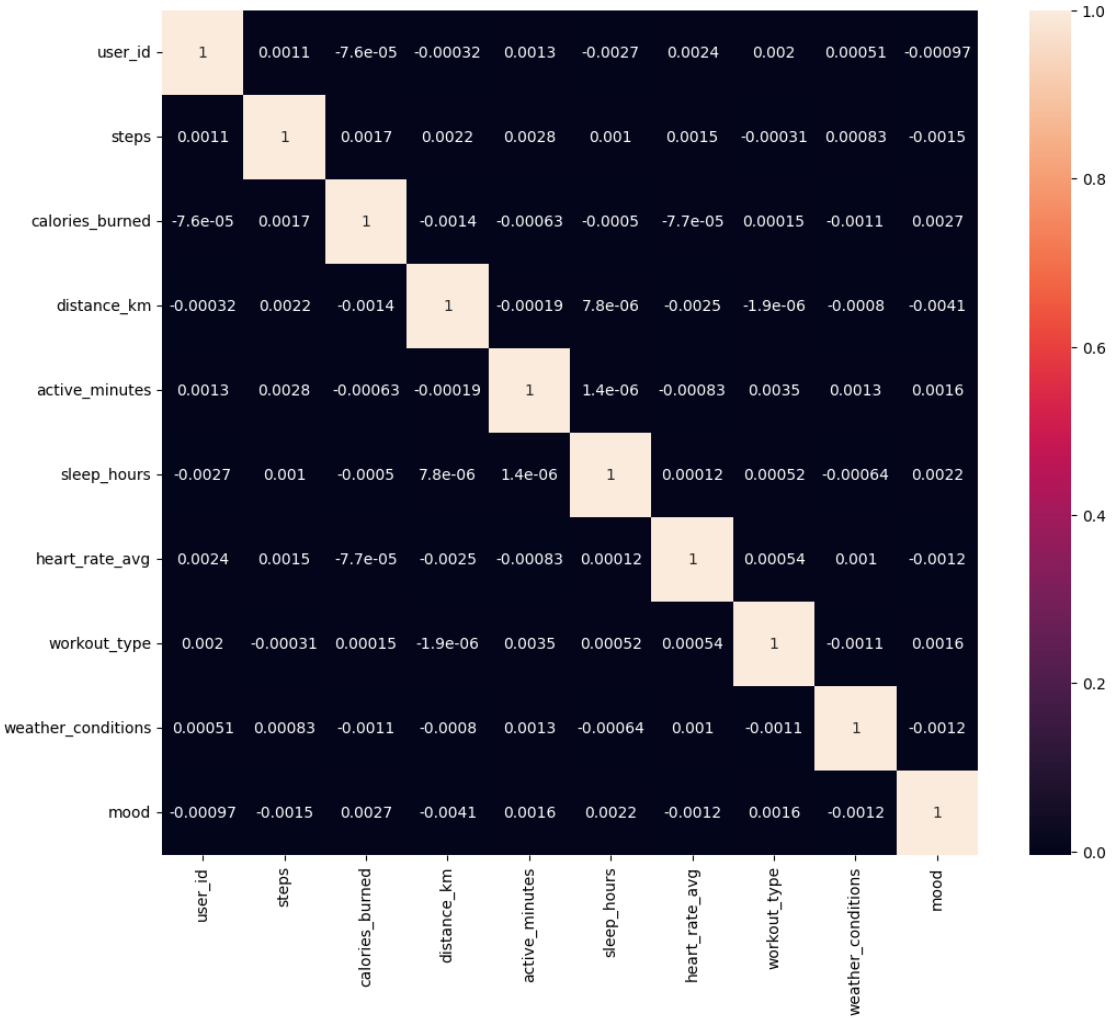
```
user_id               0
date                  0
steps                 0
calories_burned       0
distance_km           0
active_minutes        0
sleep_hours           0
heart_rate_avg        0
workout_type          0
weather_conditions    0
location              0
mood                  0
dtype: int64
        user_id  steps  calories_burned  distance_km  active_minutes  \
0           468   4530          2543.02        16.10             613
1           879  11613          1720.76         8.10             352
2           152  27335          1706.35         3.57             236
3           311  13459          2912.38         6.41            1329
4           759  15378          3344.51        17.88              52
…           …    …              …              …                …
```

```
376570      148   7736          2332.95         4.74             554
376571      384   8229          3286.88         8.10             830
376572      456   5425          1832.47         0.60             631
376573      980  26212          3994.09        16.72             428
376574      495  24728          3367.84         0.88             716

        sleep_hours  heart_rate_avg workout_type weather_conditions      mood
0               1.5           176.0      Walking              Clear     Tired
1               6.3           128.0      Cycling                Fog     Happy
2               6.7           134.0         Yoga               Snow   Neutral
3              11.6           116.0     Swimming               Rain     Tired
4               7.4            84.0     Swimming               Rain   Neutral
…               …              …           …                  …       …
376570          1.8           129.0      Running               Snow     Happy
376571          5.0           106.0      Cycling              Clear   Neutral
376572          5.0            79.0      Cycling               Snow  Stressed
376573          6.0            71.0     Swimming               Snow  Stressed
376574          5.0           117.0      Cycling               Snow  Stressed

[322717 rows x 10 columns]
        user_id  steps  calories_burned  distance_km  active_minutes  \
0           468   4530          2543.02        16.10             613
1           879  11613          1720.76         8.10             352
2           152  27335          1706.35         3.57             236
3           311  13459          2912.38         6.41            1329
4           759  15378          3344.51        17.88              52
…         …    …              …           …              …
376570      148   7736          2332.95         4.74             554
376571      384   8229          3286.88         8.10             830
376572      456   5425          1832.47         0.60             631
376573      980  26212          3994.09        16.72             428
376574      495  24728          3367.84         0.88             716

        sleep_hours  heart_rate_avg  workout_type  weather_conditions  mood
0               1.5           176.0             1                   0     3
1               6.3           128.0             2                   1     1
2               6.7           134.0             4                   2     2
3              11.6           116.0             3                   3     3
4               7.4            84.0             3                   3     2
…               …              …           …                   …   …
376570          1.8           129.0             6                   2     1
376571          5.0           106.0             2                   0     2
376572          5.0            79.0             2                   2     0
376573          6.0            71.0             3                   2     0
376574          5.0           117.0             2                   2     0

[322717 rows x 10 columns]
<class 'pandas.core.frame.DataFrame'>
```

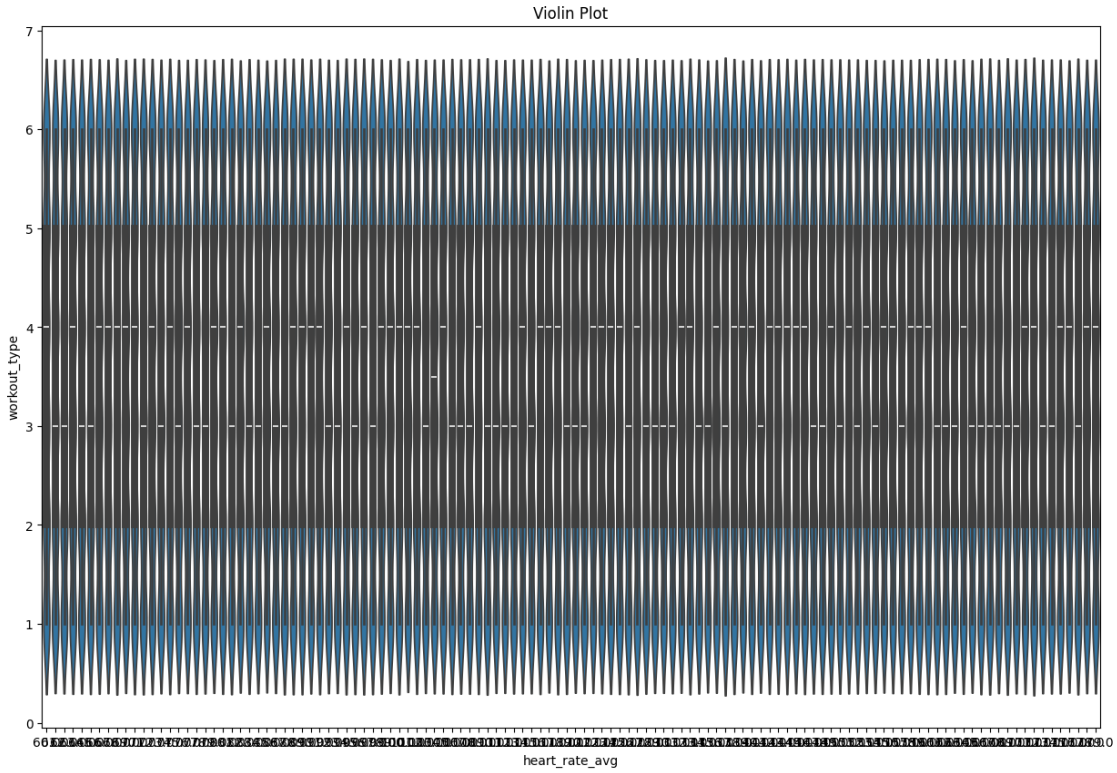
```
Index: 322717 entries, 0 to 376574
Data columns (total 10 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   user_id             322717 non-null  int64
 1   steps               322717 non-null  int64
 2   calories_burned     322717 non-null  float64
 3   distance_km         322717 non-null  float64
 4   active_minutes      322717 non-null  int64
 5   sleep_hours         322717 non-null  float64
 6   heart_rate_avg      322717 non-null  float64
 7   workout_type        322717 non-null  int64
 8   weather_conditions  322717 non-null  int64
 9   mood                322717 non-null  int64
dtypes: float64(4), int64(6)
memory usage: 27.1 MB
[1 2 4 3 5 6]
```

Data Visualization

[22]:
```python
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(12,10))
correlation_matrix = new_df.corr()
sns.heatmap(correlation_matrix, annot=True)
plt.show()
```



[32]:
```
import seaborn as sns
import matplotlib.pyplot as plt
plotsize = (15,10)
plt.figure(figsize=plotsize)
sns.violinplot(x="heart_rate_avg", y="workout_type", data=new_df)
plt.title('Violin Plot')
plt.show()
```



Application to Fitness Tracker Business:

Linear Growth Scenario

Let's assume: - Pricing of the fitness tracker (m = \$100) - Initial sales (b = 100) units - Rate of growth in sales (a = 20) units per year - Fixed costs (c = \$5000)

The linear financial model will be:

[y = 100 (20 t + 100) + 5000] [y = 2000 t + 10000 + 5000] [y = 2000 t + 15000]

Exponential Growth Scenario

Let's assume: - Pricing of the fitness tracker (m = \$100) - Initial sales (x_0 = 100) units - Growth rate (k = 0.1) (10% per year) - Fixed costs (c = \$5000)

The exponential financial model will be:

[y = 100 (100 e^{0.1t}) + 5000] [y = 10000 e^{0.1t} + 5000]

Summary

- **Linear Growth Model:** [y = 2000 t + 15000]
- **Exponential Growth Model:** [y = 10000 e^{0.1t} + 5000]

These models provide a framework for predicting total profit based on the assumed growth patterns of the fitness tracker market. Adjust the parameters as needed to reflect realistic scenarios for your business.