# Analayzing Workout Patterns: Comparing Calorie Burn in Cardio and Strength Training

Programming with Data Assignment

# Introduction to the Research Space:

Having recently started going to the gym and experimenting with various workout routines, including cardio and strength training, I aim to analyze how different types of workouts influence the average calorie burn. This research focuses on comparing cardio and strength training, utilizing data from 'Kaggle' to uncover trends and insights. The primary objective is to provide a deeper understanding of how these exercises impact calorie burn.

# Aims and Objectives

The aim of this research is to analyze how different types of workout routines, specifically cardio and strength training, influence the average calorie burn per session among fitness enthusiasts. By focusing on this topic, the research seeks to understand the variations in calorie expenditure between these two workout styles and identify patterns or trends in user behavior. A key objective is to provide actionable insights that can help individuals make informed decisions about their fitness routines, particularly in selecting exercises that align with their personal fitness goals. Through this analysis, I aim to contribute valuable knowledge for anyone looking to optimize their workout routines and better understand the impact of different types of exercises.

# Acquiring the Dataset

I will acquire datasets from Kaggle using the Kaggle API, which offers structured data on fitness metrics like calorie burn and workout types. Also, I will use the Python requests module to fetch data programmatically for further analysis.

# Utilizing the Dataset

The dataset will be utilized to examine the relationship between workout types—cardio and strength training—and their effect on average calorie burn. By leveraging Python in a Jupyter Notebook, I will carry out exploratory data analysis (EDA) to uncover trends, including variations in calorie burn by workout type and session length. In addition to that, a t-test will be conducted if required to evaluate the statistical significance of the observed patterns.

# Writing Style

The writing style for this research will be analytical and explanatory to ensure clarity and accessibility for a broader audience as this research not only aims to educate but also help people with their own personal fitness goals. To guide readers effectively, the research will include detailed references to specific datasets and their relevance. For instance, when discussing weight loss or weight gain implications, the writing will explicitly direct readers to the corresponding sections or datasets. This approach ensures that readers can easily navigate the research and locate the information most pertinent to their interests or objectives.

# Summary of the Area of the Research

This research will explore how cardio and strength training impact calorie burn, using datasets from Kaggle API and using those datasets for the exploratory data analysis (EDA). The goal is to uncover trends, provide actionable insights, and help individuals make informed fitness decisions based on their goals.

```
!pip install kaggle
```

```
Collecting kaggle
  Downloading kaggle-1.6.17.tar.gz (82 kB)
  Preparing metadata (setup.py) ... ent already satisfied: six>=1.10
in /opt/anaconda3/lib/python3.12/site-packages (from kaggle) (1.16.0)
Requirement already satisfied: certifi>=2023.7.22 in
/opt/anaconda3/lib/python3.12/site-packages (from kaggle) (2024.12.14)
Requirement already satisfied: python-dateutil in
/opt/anaconda3/lib/python3.12/site-packages (from kaggle)
(2.9.0.post0)
Requirement already satisfied: requests in
/opt/anaconda3/lib/python3.12/site-packages (from kaggle) (2.32.3)
Requirement already satisfied: tqdm in
/opt/anaconda3/lib/python3.12/site-packages (from kaggle) (4.66.5)
Requirement already satisfied: python-slugify in
/opt/anaconda3/lib/python3.12/site-packages (from kaggle) (5.0.2)
Requirement already satisfied: urllib3 in
/opt/anaconda3/lib/python3.12/site-packages (from kaggle) (2.2.3)
Requirement already satisfied: bleach in
/opt/anaconda3/lib/python3.12/site-packages (from kaggle) (4.1.0)
Requirement already satisfied: packaging in
/opt/anaconda3/lib/python3.12/site-packages (from bleach->kaggle)
(24.1)
Requirement already satisfied: webencodings in
/opt/anaconda3/lib/python3.12/site-packages (from bleach->kaggle)
(0.5.1)
Requirement already satisfied: text-unidecode>=1.3 in
/opt/anaconda3/lib/python3.12/site-packages (from python-slugify-
>kaggle) (1.3)
```

```
Requirement already satisfied: charset-normalizer<4,>=2 in
/opt/anaconda3/lib/python3.12/site-packages (from requests->kaggle)
(3.3.2)
Requirement already satisfied: idna<4,>=2.5 in
/opt/anaconda3/lib/python3.12/site-packages (from requests->kaggle)
(3.7)
Building wheels for collected packages: kaggle
  Building wheel for kaggle (setup.py) ... e=kaggle-1.6.17-py3-none-
any.whl size=105786
sha256=91d6f8fe384af5da26fffeb4d6e0fd9fbcc4541737783ebeb3e901488ba81c0
1
  Stored in directory:
/Users/shivanishri/Library/Caches/pip/wheels/46/d2/26/84d0a1acdb9c6bac
cf7d28cf06962ec80529fe1ad938489983
Successfully built kaggle
Installing collected packages: kaggle
Successfully installed kaggle-1.6.17
```

## Import Requirements

```
# The imports

#basic libraries for data manipulation and analysis
import pandas as pd
import zipfile
import io

import numpy as np

#data visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns

#statistical testing
from scipy.stats import ttest_ind

#json
import os
import json
```

# Why the Sourced Data is Relevant

This study utilizes two datasets from Kaggle to compare specific key variables, aiming to perform a concise yet simple analysis of cardio and strength training exercises. The Kaggle API serves as a resource for accessing datasets relevant to this analysis.

## The identifiable case for working with this data

The datasets mentioned above contain essential but concise information, including activity types and specific details like calories burned for cardio exercises and the weightlifting capacity of different age groups for strength training workouts.

## How the format of data is suitable for analysis

The Kaggle API facilitates reproducible access to the cardio and strength workout datasets, provided in CSV format. The use of ZipFile ensures the extraction of compressed files, and the statistical tool, SciPy will allow for hypothesis testing. These will make sure that the analysis is efficient using the right datasets.

## Consideration of two other datasets

The two other datasets that were considered were- Google Dataset Reseatrch and Dataworld. While Google Dataset Search offers a broad range of datasets, the search results often require additional effort to identify reliable and structured data sources, which can be time-consuming. On the other hand, Data World, though highly collaborative and flexible, does not always focus on fitness-related datasets or provide the level of curation and metadata that Kaggle offers. Kaggle's platform, with its structured CSV format, integrated API, and ready-to-use datasets, provided a more efficient and reliable foundation for analyzing cardio and strength training workout data.

# Ethics of data used

## Where the data comes from

Kaggle itself is not open-source; it is a proprietary platform owned by Google. However, many of the datasets hosted on Kaggle are open-sourced or provided under permissive licenses like CC0 (Public Domain) or CC BY (Creative Commons), which allow free use, modification, and sharing. These datasets are typically contributed by individuals, researchers, organizations, or government institutions. Also, Kaggle provides access to public kernels (Jupyter notebooks), code, and discussions, which are shared by the community and can often be reused under their respective licenses.

## Considerations about the usage/reuse of data

Kaggle is not an open-source platform; however, the datasets hosted on Kaggle are open-source. The datasets I have utilized from Kaggle are also open-source.

## Consideration around implications of utilising data for purpose

This research serves multiple purposes, including enabling others to reuse the data I have compiled for further analysis and providing a reference point for individuals starting their workout journey, among other applications. The study focuses on cardio and strength training workouts, offering detailed insights into each type. By analyzing common workouts from both categories, it aims to help users discover their workout preferences through an unbiased and accurate exploration of the data.

## Considerations of the data processing pipeline

Since Kaggle offers access to public kernels (Jupyter notebooks), code, and discussions, this ensures that the data is both easily accessible and ethically handled.

## Potential biases of the dataset

The data utilized in this research comes from two CSV files, each containing workout information for both cardio and strength training.

Demographic imbalance: A potential bias in the dataset is that it may disproportionately represent certain demographics, such as specific age groups, genders, or geographic regions.

Workout type representation: Another possible bias is an imbalance between cardio and strength training data. For instance, commonly performed exercises may be overrepresented, while niche or less popular workouts might be underrepresented, limiting the diversity and breadth of insights.

# Project Background

## Why the field of interest/relevant

This research aligns with my field of interest as it combines my personal fitness journey and my passion for data analysis. Having recently started going to the gym and experimenting with different workout routines, I have developed a curiosity about how various exercises, such as cardio and strength training, influence outcomes. By analyzing data from Kaggle, I can bridge my interest in fitness with my skills in data processing and visualization, uncovering meaningful trends and insights.

## Previous exploration of this topic

Many individuals have utilized the Kaggle API to retrieve datasets for further exploration; however, this research specifically focuses on analyzing selected columns and rows within the dataset to provide a more detailed and refined analysis.

## Scope of work

Analysis part 1- I will compare calorie burn: For cardio workouts, I will be using the 'calories burned' column to analyze how cardio activities contribute to calorie expenditure. For strength training workout, I will be focusing on creating a correlation between the Body weight Kg and the Weight Class Kg to assess how strength training intensity might influence calorie burn.

I will then compare these two insights.

Analysis part 2- I will compare workout duration: For cardio workouts, I will utilize the 'Duration' column to examine how workout length impacts calorie burn. For strength training workouts, I will compare with 'Event' (such as SBD lifts) and the corresponding weights lifted to assess how workout duration and intensity differ between cardio and strength training.

I will then again compare these two insights.

For the final analysis(analysis part 3), I will create a t-test to check whether there is a significant difference between cardio and strength workouts.

## Steps in analytical data processing pipeline

1. Install Kaggle.
2. Obtain relevant datasets from the Kaggle API for analysis.
3. Organise retrieved data into pandas dataframes.
4. Clean dataframes for high-quality results.
5. Use visualisation tools (e.g. histograms, bar charts, and more) to show and compare specific data insights.
6. Summarise study findings and discuss their practical consequences.

# The Technical exploration of the dataset

# The Data used

## Origin of Data

The data for this research project will be CSV files from Kaggle, which can be retrieved using the Python requests module via the Kaggle API request. This is done so that any csv files needed for further analysis can be requested using API requests.

## Setting up Kaggle API

```
with open("/Users/shivanishri/Desktop/Jason/kaggle.json", "r") as f:
    data = json.load(f)
```

Moving the file

```
#the .kaggle directory
os.makedirs(os.path.expanduser("~/.kaggle"), exist_ok=True)

#moving the kaggle.json file to the .kaggle directory
!mv "/Users/shivanishri/Desktop/Jason/kaggle.json" ~/.kaggle/

#setting up the permissions for the kaggle.json file
os.chmod(os.path.expanduser("~/.kaggle/kaggle.json"), 0o600)

!kaggle datasets list
```

```
ref                                                            title
size  lastUpdated          downloadCount  voteCount  usabilityRating
-------------------------------------------------------------------
-----------------------------------------------  -----
------------------  -------------  ---------  ---------------
oktayrdeki/heart-disease                                       Heart
Disease                                          568KB  2024-12-29
13:26:49            931         29  1.0
fatmanur12/new-york-air-quality                                New
York Air Quality                                   166KB  2025-01-01
19:19:59            597         24  1.0
bhadramohit/customer-shopping-latest-trends-dataset
Customer Shopping (Latest Trends) Dataset           76KB  2024-11-23
15:26:12          20592        402  1.0
stealthtechnologies/predict-student-performance-dataset
Predict Student Performance                         12KB  2024-12-26
12:57:04           1147         31  1.0
mhassansaboor/toyota-motors-stock-data-2980-2024
Toyota Motors Stock Data (1980-2024)               189KB  2024-12-28
20:26:56            867         37  1.0
hopesb/student-depression-dataset
Student Depression Dataset.                        454KB  2024-11-22
17:56:03          16771        241  1.0
govindaramsriram/sleep-time-prediction                         Sleep
Time Prediction                                  28KB  2024-12-28
17:08:56           1310         31  1.0
abdulmalik1518/the-ultimate-cars-dataset-2024                  The
Ultimate Cars Dataset 2024                         25KB  2024-12-30
14:45:41            805         29  1.0
govindaramsriram/crop-yield-of-a-farm                          Crop
Yield of a Farm                                  28KB  2024-12-28
18:12:16            876         28  1.0
willianoliveiragibin/water-sanitation-and-hygiene
Water, Sanitation and Hygiene                       33KB  2024-12-30
20:41:52            402         22  1.0
taimoor888/top-100-youtube-channels-in-2024                    Top
100 YouTube Channels in 2024                        3KB  2024-12-15
16:09:25           1484         42  1.0
```

```
oktayrdeki/houses-in-london
Houses in London                                          21KB  2024-12-15
19:27:42           1697         37  1.0
denkuznetz/food-delivery-time-prediction                          Food
Delivery Time Prediction                                  12KB  2024-12-23
13:12:14           1435         39  1.0
mujtabamatin/air-quality-and-pollution-assessment                 Air
Quality and Pollution Assessment                           84KB  2024-12-04
15:29:51           7877        117  1.0
yaaryiitturan/global-tech-salary-dataset
Global Tech Salary Dataset                                43KB  2024-12-30
12:52:44            571         28  1.0
michaellanurias/spotify-playlist-origins
Spotify Playlist-ORIGINS                                  25KB  2024-12-15
01:16:34           1087         30  1.0
mhassansaboor/intel-stock-data-1980-2024                          Intel
Stock Data (1980-2024)                                   281KB  2024-12-25
16:12:36            771         29  1.0
hassanelfattmi/why-do-customers-leave-can-you-spot-the-churners  Why
Do Customers Leave? Can You Spot the Churners?    3MB  2024-12-21
11:49:00            728         21  1.0
kanchana1990/world-internet-usage-data-2023-updated               World
Internet Usage Data (2023 Updated)                4KB  2024-12-21
09:41:41            813         50  1.0
olaflundstrom/vaccination-against-covid-19
Vaccination against Covid-19                              55MB  2024-12-31
10:29:06            227         48  0.9411765
```

# Data Collecting

## Collecting data on Cardio Workouts and loading into Pandas dataframe

```
!kaggle datasets download -d deependraverma13/cardio-activities

with zipfile.ZipFile("cardio-activities.zip", "r") as z:
    with z.open("cardioActivities.csv") as f:
        cardio_df = pd.read_csv(f)

Dataset URL: https://www.kaggle.com/datasets/deependraverma13/cardio-
activities
License(s): apache-2.0
cardio-activities.zip: Skipping, found more recently modified local
copy (use --force to force download)

print("Cardio Workouts - Before Cleaning:")
print(cardio_df.head())
```

```
Cardio Workouts - Before Cleaning:
                  Date                           Activity Id     Type
\
0  2018-11-11 14:05:12  c9627fed-14ac-47a2-bed3-2a2630c63c15  Running

1  2018-11-09 15:02:35  be65818d-a801-4847-a43b-2acdf4dc70e7  Running

2  2018-11-04 16:05:00  c09b2f92-f855-497c-b624-c196b3ef036c  Running

3  2018-11-01 14:03:58  bc9b612d-3499-43ff-b82a-9b17b71b8a36  Running

4  2018-10-27 17:01:36  972567b2-1b0e-437c-9e82-fef8078d6438  Running


  Route Name  Distance (km) Duration Average Pace  Average Speed
(km/h)  \
0        NaN          10.44    58:40         5:37
10.68
1        NaN          12.84  1:14:12         5:47
10.39
2        NaN          13.01  1:15:16         5:47
10.37
3        NaN          12.98  1:14:25         5:44
10.47
4        NaN          13.02  1:12:50         5:36
10.73

   Calories Burned  Climb (m)  Average Heart Rate (bpm)  Friend's
Tagged  \
0            774.0        130                     159.0
NaN
1            954.0        168                     159.0
NaN
2            967.0        171                     155.0
NaN
3            960.0        169                     158.0
NaN
4            967.0        170                     154.0
NaN

  Notes               GPX File
0   NaN  2018-11-11-140512.gpx
1   NaN  2018-11-09-150235.gpx
2   NaN  2018-11-04-160500.gpx
3   NaN  2018-11-01-140358.gpx
4   NaN  2018-10-27-170136.gpx
```

The code below checks and cleans the specific data from the whole dataset to make sure there are no errors.

```
#cleaning
cardio_df['Calories Burned'] = cardio_df['Calories
Burned'].astype(str).str.replace(r'[^\d.]', '',
regex=True).astype(float)

cardio_df['Duration'] = pd.to_timedelta(cardio_df['Duration'],
errors='coerce')

cardio_df['Distance (km)'] = cardio_df['Distance
(km)'].astype(str).str.replace(r'[^\d.]', '',
regex=True).astype(float)

cardio_df = cardio_df.dropna(subset=['Calories Burned', 'Duration',
'Distance (km)'])

#cleaned dataset
print("Cleaned Cardio Dataset:")
print(cardio_df.head())
```

```
Cleaned Cardio Dataset:
                  Date                           Activity Id     Type
\
1  2018-11-09 15:02:35  be65818d-a801-4847-a43b-2acdf4dc70e7  Running

2  2018-11-04 16:05:00  c09b2f92-f855-497c-b624-c196b3ef036c  Running

3  2018-11-01 14:03:58  bc9b612d-3499-43ff-b82a-9b17b71b8a36  Running

4  2018-10-27 17:01:36  972567b2-1b0e-437c-9e82-fef8078d6438  Running

6  2018-10-14 17:28:56  96acedc9-d3d5-4aac-8df4-f549a6418c1d  Running


  Route Name  Distance (km)        Duration Average Pace  \
1        NaN          12.84 0 days 01:14:12         5:47
2        NaN          13.01 0 days 01:15:16         5:47
3        NaN          12.98 0 days 01:14:25         5:44
4        NaN          13.02 0 days 01:12:50         5:36
6        NaN          12.93 0 days 01:10:16         5:26

   Average Speed (km/h)  Calories Burned  Climb (m)  Average Heart
Rate (bpm)  \
1                 10.39            954.0        168
159.0
2                 10.37            967.0        171
155.0
3                 10.47            960.0        169
158.0
```

```
4                  10.73             967.0        170
154.0
6                  11.04             953.0        159
158.0

   Friend's Tagged Notes                GPX File
1              NaN   NaN  2018-11-09-150235.gpx
2              NaN   NaN  2018-11-04-160500.gpx
3              NaN   NaN  2018-11-01-140358.gpx
4              NaN   NaN  2018-10-27-170136.gpx
6              NaN   NaN  2018-10-14-172856.gpx
```

## Collecting data on Strength Training Workouts and loading into Pandas dataframe

```
!kaggle datasets download -d docgenki/powerlifting-dataset

with zipfile.ZipFile("powerlifting-dataset.zip", "r") as z:
    with z.open("openpowerlifting-2023-04-08-a090afd8.csv") as f:
        strength_df = pd.read_csv(f, low_memory=False)

print("Strength Workouts - Before Cleaning:")
print(strength_df.head())
```

```
Dataset URL: https://www.kaggle.com/datasets/docgenki/powerlifting-
dataset
License(s): CC0-1.0
powerlifting-dataset.zip: Skipping, found more recently modified local
copy (use --force to force download)
Strength Workouts - Before Cleaning:
                    Name Sex Event Equipment   Age AgeClass
BirthYearClass  \
0             Alona Vladi   F   SBD       Raw  33.0    24-34
24-39
1      Galina Solovyanova   F   SBD       Raw  43.0    40-44
40-49
2          Daniil Voronin   M   SBD       Raw  15.5    16-17
14-18
3          Aleksey Krasov   M   SBD       Raw  35.0    35-39
24-39
4  Margarita Pleschenkova   M   SBD       Raw  26.5    24-34
24-39

  Division  BodyweightKg WeightClassKg  ...  Tested  Country  State  \
0        O         58.30            60  ...     Yes   Russia    NaN
1       M1         73.10            75  ...     Yes   Russia    NaN
2        T         67.40            75  ...     Yes   Russia    NaN
3        O         66.65            75  ...     Yes   Russia    NaN
```

```
4         0         72.45             75  ...      Yes   Russia    NaN

   Federation  ParentFederation        Date  MeetCountry  MeetState
MeetTown  \
0         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
1         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
2         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
3         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
4         GFP               NaN  2019-05-11       Russia        NaN
Bryansk

           MeetName
0  Open Tournament
1  Open Tournament
2  Open Tournament
3  Open Tournament
4  Open Tournament

[5 rows x 41 columns]
```

The code below checks and cleans the specific data from the whole dataset to make sure there are no errors.

```python
#debug
print("Columns in DataFrame:", strength_df.columns)

#debug
print("\nInitial Rows:")
print(strength_df.head())

#cleaning
try:
    strength_df['BodyweightKg'] = (
        strength_df['BodyweightKg']
        .astype(str)
        .str.replace(r'[^\d.]', '', regex=True)
        .apply(pd.to_numeric, errors='coerce')
    )
except KeyError:
    print("Column 'BodyweightKg' not found!")

try:
    strength_df['WeightClassKg'] = (
        strength_df['WeightClassKg']
        .astype(str)
```

```
        .str.replace(r'[^\d.]', '', regex=True)
        .apply(pd.to_numeric, errors='coerce')
    )
except KeyError:
    print("Column 'WeightClassKg' not found!")

try:
    strength_df['Event'] = strength_df['Event'].fillna('Unknown')
except KeyError:
    print("Column 'Event' not found!")

try:
    strength_df['Age'] = pd.to_numeric(strength_df['Age'],
errors='coerce')
    strength_df = strength_df[(strength_df['Age'] > 10) &
(strength_df['Age'] < 100)]
except KeyError:
    print("Column 'Age' not found!")

critical_columns = ['BodyweightKg', 'WeightClassKg', 'Event', 'Age']
strength_df = strength_df.dropna(subset=critical_columns)

#debug
print("\nStrength Workouts - After Cleaning:")
print(strength_df.head())

#cleaned dataset
strength_df.to_csv('cleaned_strength.csv', index=False)
print("\nCleaned dataset saved as 'cleaned_strength.csv'")
```

```
Columns in DataFrame: Index(['Name', 'Sex', 'Event', 'Equipment',
'Age', 'AgeClass',
       'BirthYearClass', 'Division', 'BodyweightKg', 'WeightClassKg',
       'Squat1Kg', 'Squat2Kg', 'Squat3Kg', 'Squat4Kg', 'Best3SquatKg',
       'Bench1Kg', 'Bench2Kg', 'Bench3Kg', 'Bench4Kg', 'Best3BenchKg',
       'Deadlift1Kg', 'Deadlift2Kg', 'Deadlift3Kg', 'Deadlift4Kg',
       'Best3DeadliftKg', 'TotalKg', 'Place', 'Dots', 'Wilks',
'Glossbrenner',
       'Goodlift', 'Tested', 'Country', 'State', 'Federation',
       'ParentFederation', 'Date', 'MeetCountry', 'MeetState',
'MeetTown',
       'MeetName'],
      dtype='object')

Initial Rows:
                    Name Sex Event Equipment   Age AgeClass
BirthYearClass  \
0             Alona Vladi   F   SBD       Raw  33.0    24-34
24-39
1      Galina Solovyanova   F   SBD       Raw  43.0    40-44
```

```
40-49
2          Daniil Voronin   M   SBD       Raw  15.5    16-17
14-18
3          Aleksey Krasov   M   SBD       Raw  35.0    35-39
24-39
4  Margarita Pleschenkova   M   SBD       Raw  26.5    24-34
24-39

  Division  BodyweightKg  WeightClassKg  ...  Tested  Country
State  \
0        O         58.30           60.0  ...     Yes   Russia    NaN

1       M1         73.10           75.0  ...     Yes   Russia    NaN

2        T         67.40           75.0  ...     Yes   Russia    NaN

3        O         66.65           75.0  ...     Yes   Russia    NaN

4        O         72.45           75.0  ...     Yes   Russia    NaN


   Federation  ParentFederation        Date  MeetCountry  MeetState
MeetTown  \
0         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
1         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
2         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
3         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
4         GFP               NaN  2019-05-11       Russia        NaN
Bryansk

          MeetName
0  Open Tournament
1  Open Tournament
2  Open Tournament
3  Open Tournament
4  Open Tournament

[5 rows x 41 columns]

Strength Workouts - After Cleaning:
                    Name Sex Event Equipment   Age AgeClass
BirthYearClass  \
0             Alona Vladi   F   SBD       Raw  33.0    24-34
24-39
1      Galina Solovyanova   F   SBD       Raw  43.0    40-44
40-49
```

```
2          Daniil Voronin   M   SBD       Raw  15.5    16-17
14-18
3          Aleksey Krasov   M   SBD       Raw  35.0    35-39
24-39
4  Margarita Pleschenkova   M   SBD       Raw  26.5    24-34
24-39

  Division  BodyweightKg  WeightClassKg  ...  Tested  Country
State  \
0        O         58.30           60.0  ...     Yes   Russia    NaN

1       M1         73.10           75.0  ...     Yes   Russia    NaN

2        T         67.40           75.0  ...     Yes   Russia    NaN

3        O         66.65           75.0  ...     Yes   Russia    NaN

4        O         72.45           75.0  ...     Yes   Russia    NaN


   Federation  ParentFederation        Date  MeetCountry  MeetState
MeetTown  \
0         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
1         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
2         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
3         GFP               NaN  2019-05-11       Russia        NaN
Bryansk
4         GFP               NaN  2019-05-11       Russia        NaN
Bryansk

          MeetName
0  Open Tournament
1  Open Tournament
2  Open Tournament
3  Open Tournament
4  Open Tournament

[5 rows x 41 columns]

Cleaned dataset saved as 'cleaned_strength.csv'
```

# Checking for out of bound values

Out-of-Bound Values: Minimum and maximum values for numeric fields like Calories Burned, Distance (km), BodyweightKg, and Age are checked to ensure data validity and remove unrealistic values.

Categorization: Data is divided into 10 quantiles using 'pd.qcut' to identify patterns and trends across different ranges, such as Top 10% and Bottom 10%.

```
#for cardio workouts

print("Min and Max values for Cardio Workouts:")
print("Calories Burned - Min:", cardio_df['Calories Burned'].min(),
"Max:", cardio_df['Calories Burned'].max())
print("Distance (km) - Min:", cardio_df['Distance (km)'].min(),
"Max:", cardio_df['Distance (km)'].max())

cardio_df['Calories Category'] = pd.qcut(cardio_df['Calories Burned'],
q=10, labels=[f"Group {i+1}" for i in range(10)])
print("\nCardio Data Categorized by Calories Burned (Top 10%, Bottom
10% etc.):")
print(cardio_df[['Calories Burned', 'Calories Category']].head())

cardio_df['Distance Category'] = pd.qcut(cardio_df['Distance (km)'],
q=10, labels=[f"Group {i+1}" for i in range(10)])
print("\nCardio Data Categorized by Distance (Top 10%, Bottom 10%
etc.):")
print(cardio_df[['Distance (km)', 'Distance Category']].head())
```

```
Min and Max values for Cardio Workouts:
Calories Burned - Min: 306.0 Max: 2587.9999985093104
Distance (km) - Min: 2.48 Max: 49.18

Cardio Data Categorized by Calories Burned (Top 10%, Bottom 10% etc.):
   Calories Burned Calories Category
1            954.0           Group 6
2            967.0           Group 6
3            960.0           Group 6
4            967.0           Group 6
6            953.0           Group 6

Cardio Data Categorized by Distance (Top 10%, Bottom 10% etc.):
   Distance (km) Distance Category
1          12.84           Group 4
2          13.01           Group 5
3          12.98           Group 5
4          13.02           Group 5
6          12.93           Group 5
```

```
#for strength training workouts

print("Min and Max values for Strength Workouts:")
print("Bodyweight (kg) - Min:", strength_df['BodyweightKg'].min(),
"Max:", strength_df['BodyweightKg'].max())
print("Weight Class (kg) - Min:", strength_df['WeightClassKg'].min(),
"Max:", strength_df['WeightClassKg'].max())
print("Age - Min:", strength_df['Age'].min(), "Max:",
strength_df['Age'].max())

strength_df['Bodyweight Category'] =
pd.qcut(strength_df['BodyweightKg'], q=10, labels=[f"Group {i+1}" for
i in range(10)])
print("\nStrength Data Categorized by Bodyweight (Top 10%, Bottom 10%
etc.):")
print(strength_df[['BodyweightKg', 'Bodyweight Category']].head())

strength_df['Age Category'] = pd.qcut(strength_df['Age'], q=10,
labels=[f"Group {i+1}" for i in range(10)])
print("\nStrength Data Categorized by Age (Top 10%, Bottom 10%
etc.):")
print(strength_df[['Age', 'Age Category']].head())
```

```
Min and Max values for Strength Workouts:
Bodyweight (kg) - Min: 18.0 Max: 285.0
Weight Class (kg) - Min: 1.0 Max: 155.0
Age - Min: 10.5 Max: 98.0

Strength Data Categorized by Bodyweight (Top 10%, Bottom 10% etc.):
   BodyweightKg Bodyweight Category
0         58.30              Group 2
1         73.10              Group 4
2         67.40              Group 3
3         66.65              Group 3
4         72.45              Group 4

Strength Data Categorized by Age (Top 10%, Bottom 10% etc.):
    Age Age Category
0  33.0      Group 7
1  43.0      Group 9
2  15.5      Group 1
3  35.0      Group 7
4  26.5      Group 5
```

## Check if the data is in the correct format

```
#for cardio workouts

try:
    assert isinstance(cardio_df, pd.DataFrame), "Cardio data is not in
```

```
DataFrame format."

    assert 'Calories Burned' in cardio_df.columns, "'Calories Burned'
column is missing in cardio dataset."
    assert 'Duration' in cardio_df.columns, "'Duration' column is
missing in cardio dataset."
    assert 'Distance (km)' in cardio_df.columns, "'Distance (km)'
column is missing in cardio dataset."

    print("Cardio data is in the correct format for analysis!")
except AssertionError as e:
    print(f"Format validation error for cardio data: {e}")
```

```
Cardio data is in the correct format for analysis!
```

```
#for strength training workouts

try:
    assert isinstance(strength_df, pd.DataFrame), "Strength data is
not in DataFrame format."

    assert 'BodyweightKg' in strength_df.columns, "'BodyweightKg'
column is missing in strength dataset."
    assert 'WeightClassKg' in strength_df.columns, "'WeightClassKg'
column is missing in strength dataset."
    assert 'Event' in strength_df.columns, "'Event' column is missing
in strength dataset."
    assert 'Age' in strength_df.columns, "'Age' column is missing in
strength dataset."

    print("Strength data is in the correct format for analysis!")
except AssertionError as e:
    print(f"Format validation error for strength data: {e}")
```

```
Strength data is in the correct format for analysis!
```

# Depth of exploration

Using the Kaggle API;

The depth of exploration in this analysis focuses on uncovering meaningful insights by comparing two distinct datasets and analyzing key variables within each. For cardio workouts, the 'Calories Burned' column is used to evaluate how different activities contribute to energy expenditure. Strength training analysis, on the other hand, examines the relationship between 'BodyweightKg' and 'WeightClassKg' to understand how intensity correlates with calorie burn. This allows for a comparison between the two workout types in terms of energy expenditure.

For cardio workouts, the 'Duration' column is used to investigate the impact of workout length on calorie burn and intensity. In strength training, variables like 'Event' (e.g., SBD lifts) and the corresponding weights lifted are assessed to understand how duration and intensity vary

compared to cardio. These comparisons provide a comprehensive view of how cardio and strength training differ in terms of effort, intensity, and outcomes, highlighting valuable insights about workout efficiency and performance.

## Data is in the correct format for analysis

The code uses assert statements to check that the data is in a Pandas Dataframe and that all the needed columns are included. If the data format is wrong or a column is missing, clear error messages are shown to explain the problem.The data can be used for further analysis such as creating charts, correlations and many more.

# Clear rhetoric for modifications to data

## Combining key variables in datasets for accuracy

```
#selecting key variables from cardio_df
cardio_table = cardio_df[['Calories Burned', 'Duration', 'Distance
(km)']].dropna().copy()
cardio_table['Workout Type'] = 'Cardio'

#selecting key variables from strength_df
strength_table = strength_df[['BodyweightKg', 'WeightClassKg',
'Event', 'Age']].dropna().copy()
strength_table['Workout Type'] = 'Strength'

#combining both tables into a new DataFrame
combined_table = pd.concat([cardio_table, strength_table],
ignore_index=True)

print("Combined Table for Analysis:")
print(combined_table.head())

combined_table.to_csv('combined_table.csv', index=False)
print("Combined table saved as 'combined_table.csv'")
```

```
Combined Table for Analysis:
   Calories Burned        Duration  Distance (km) Workout Type
BodyweightKg  \
0            954.0 0 days 01:14:12          12.84       Cardio
NaN
1            967.0 0 days 01:15:16          13.01       Cardio
NaN
2            960.0 0 days 01:14:25          12.98       Cardio
NaN
3            967.0 0 days 01:12:50          13.02       Cardio
NaN
4            953.0 0 days 01:10:16          12.93       Cardio
NaN
```

```
   WeightClassKg Event  Age
0            NaN   NaN  NaN
1            NaN   NaN  NaN
2            NaN   NaN  NaN
3            NaN   NaN  NaN
4            NaN   NaN  NaN
Combined table saved as 'combined_table.csv'
```

# Exploratory Data Analysis

## Analysis Part 1

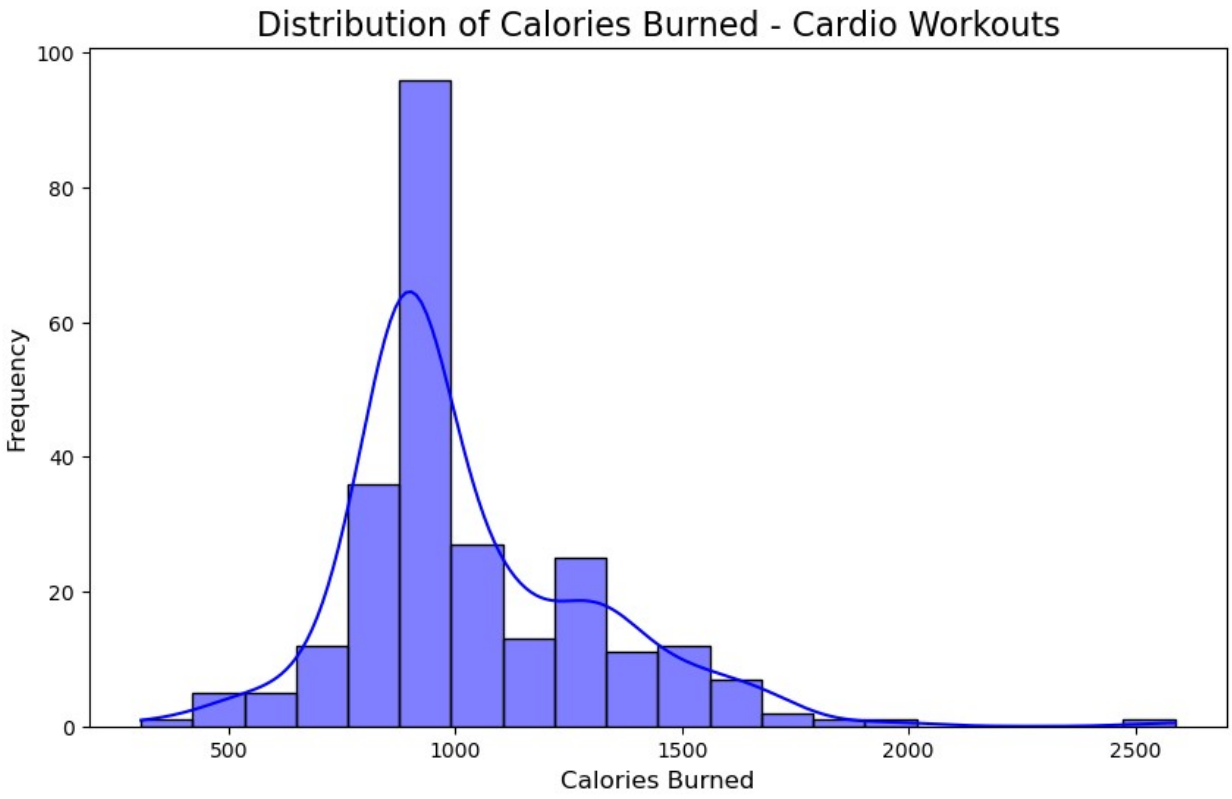
```
# for cardio workouts: analyzing calories burned
def analyze_cardio_calorie_burn(cardio_df):
    print("\n--- Cardio Workouts: Calorie Burn Analysis ---\n")
    print(cardio_df['Calories Burned'].describe())

    #calories burned
    plt.figure(figsize=(10, 6))
    sns.histplot(cardio_df['Calories Burned'], kde=True, bins=20,
color='blue')
    plt.title('Distribution of Calories Burned - Cardio Workouts',
fontsize=16)
    plt.xlabel('Calories Burned', fontsize=12)
    plt.ylabel('Frequency', fontsize=12)
    plt.show()

analyze_cardio_calorie_burn(cardio_df)
```

```
--- Cardio Workouts: Calorie Burn Analysis ---

count     255.000000
mean     1027.983124
std       281.431357
min       306.000000
25%       883.500000
50%       916.999999
75%      1148.500000
max      2587.999999
Name: Calories Burned, dtype: float64
```




A histogram is shown above: the frequency of calorie burn across cardio activities(such as running and cycling).

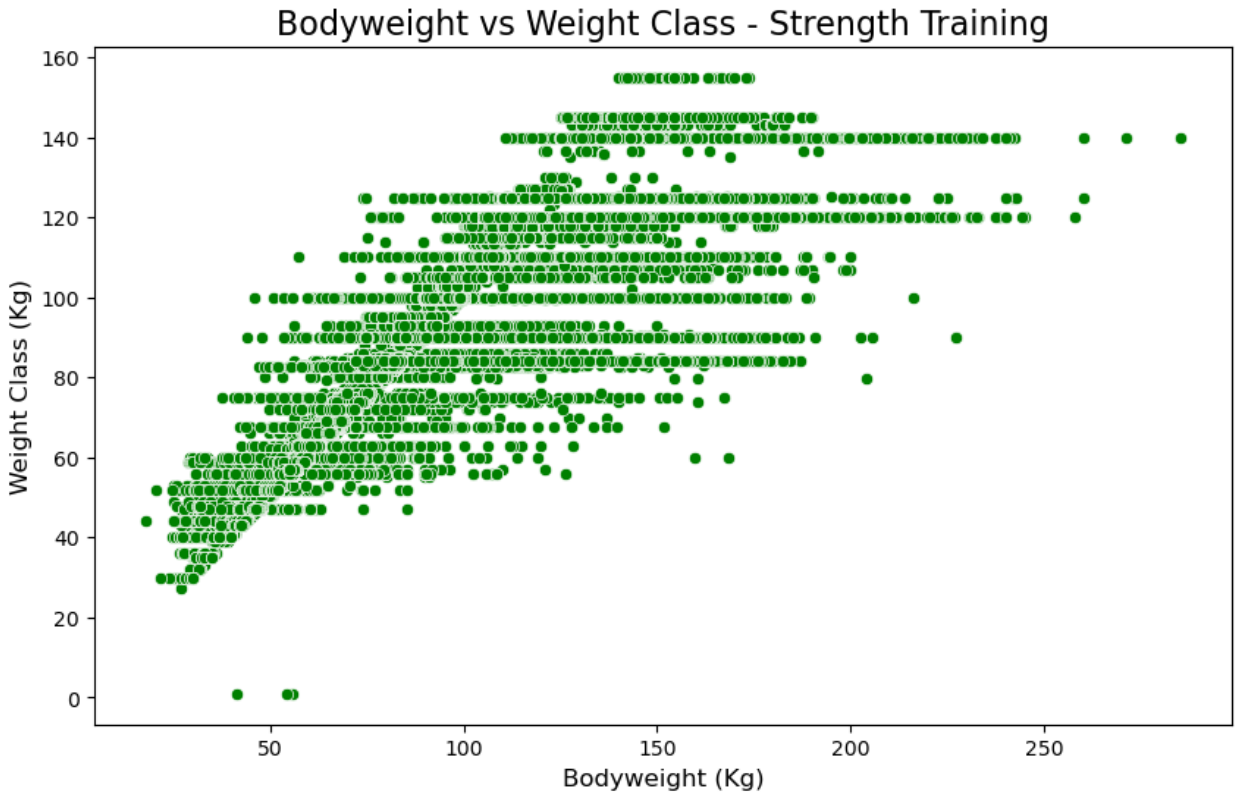
```
#for Strength Training: analyzing correlation between BodyweightKg and
WeightClassKg
def analyze_strength_correlation(strength_df):
    print("\n--- Strength Training: Correlation Analysis ---\n")
    correlation = strength_df[['BodyweightKg',
'WeightClassKg']].corr()
    print("Correlation Matrix:\n", correlation)

    # scatter Plot for Bodyweight vs WeightClass
    plt.figure(figsize=(10, 6))
    sns.scatterplot(x='BodyweightKg', y='WeightClassKg',
data=strength_df, color='green')
    plt.title('Bodyweight vs Weight Class - Strength Training',
fontsize=16)
    plt.xlabel('Bodyweight (Kg)', fontsize=12)
    plt.ylabel('Weight Class (Kg)', fontsize=12)
    plt.show()

analyze_strength_correlation(strength_df)


--- Strength Training: Correlation Analysis ---
```

```
Correlation Matrix:
                BodyweightKg  WeightClassKg
BodyweightKg        1.000000       0.962115
WeightClassKg       0.962115       1.000000
```




A scatter plot is shown above: for strength showing how body weight correlates with weight class.

## Quick comparison for analysis part 1

```
# combined insights
def compare_cardio_and_strength(cardio_df, strength_df):
    print("\n--- Combined Insights: Cardio vs Strength ---\n")

    # compare average calorie burn
    avg_cardio_calories = cardio_df['Calories Burned'].mean()
    avg_strength_intensity = (strength_df['WeightClassKg'] /
strength_df['BodyweightKg']).mean()

    print(f"Average Calories Burned (Cardio):
{avg_cardio_calories:.2f}")
    print(f"Average Intensity (Strength - WeightClass/Bodyweight):
{avg_strength_intensity:.2f}")
```

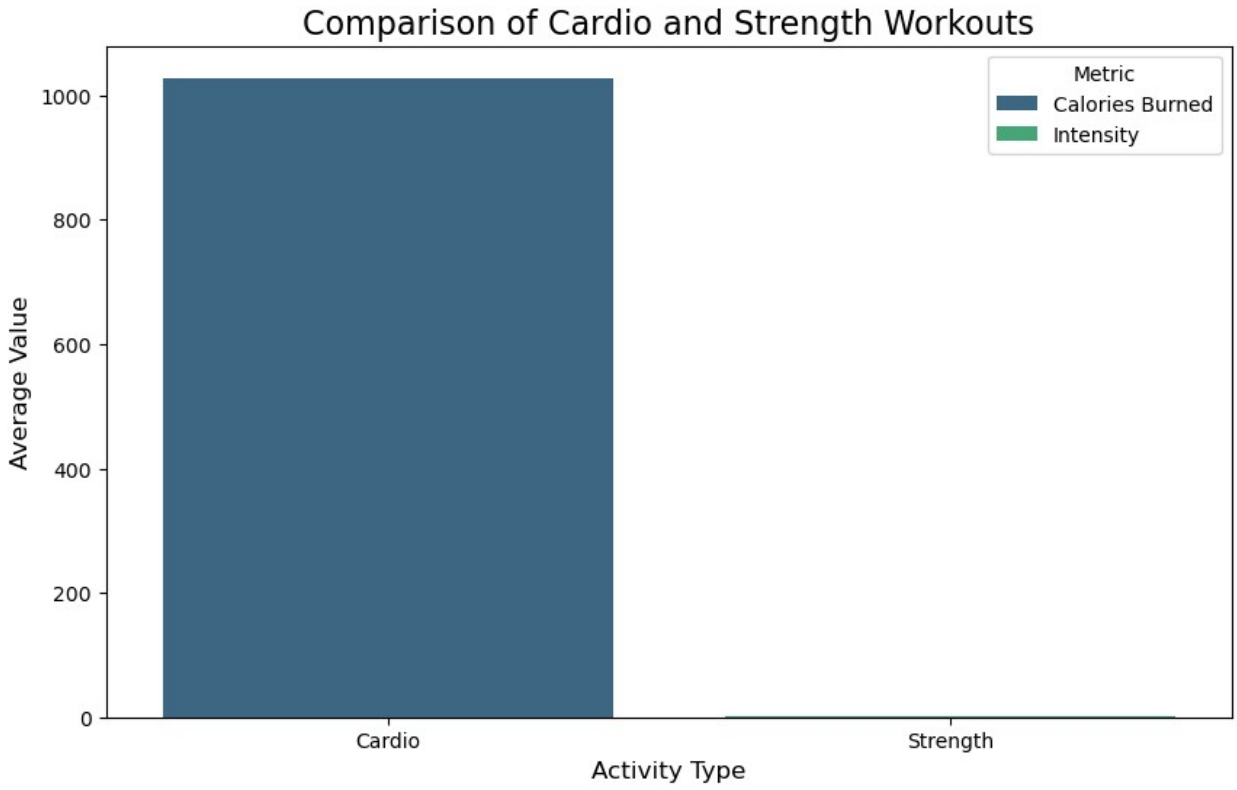
```
    # bar plot comparison
    comparison_df = pd.DataFrame({
        'Activity': ['Cardio', 'Strength'],
        'Metric': ['Calories Burned', 'Intensity'],
        'Value': [avg_cardio_calories, avg_strength_intensity]
    })

    plt.figure(figsize=(10, 6))
    sns.barplot(x='Activity', y='Value', hue='Metric',
data=comparison_df, palette='viridis')
    plt.title('Comparison of Cardio and Strength Workouts',
fontsize=16)
    plt.xlabel('Activity Type', fontsize=12)
    plt.ylabel('Average Value', fontsize=12)
    plt.legend(title='Metric')
    plt.show()

compare_cardio_and_strength(cardio_df, strength_df)


--- Combined Insights: Cardio vs Strength ---

Average Calories Burned (Cardio): 1027.98
Average Intensity (Strength - WeightClass/Bodyweight): 1.02
```



The chart above enables a side-by-side comparison of cardio and strength workouts, highlighting their physical demands. A taller cardio bar indicates higher average calorie burn, while a taller strength bar reflects greater workout intensity relative to body weight.

# Evaluation of analysis part 1

The histogram of calories burned provides a clear distribution of calorie expenditure during cardio activities, showing that most workouts burn around the mean (around 1028 kcal), with a slight skew toward higher calorie burns for more intense sessions. The statistical summary supports this, with a maximum calorie burn (around 2587 kcal) indicating outliers from extended or highly intense sessions, and a minimum (around 306 kcal) reflecting low-effort activities.

The scatter plot of BodyweightKg vs. WeightClassKg reveals a strong positive correlation (correlation coefficient ~0.96), showing that heavier individuals tend to train in higher weight classes. This alignment with expected strength training patterns highlights that heavier individuals typically train at higher intensities, emphasizing the critical relationship between body weight and training intensity.

The combined analysis reveals that cardio workouts generally burn more calories on average, while strength training emphasizes intensity relative to body weight. The bar chart comparing "Calories Burned" for cardio and "Intensity" for strength effectively highlights the contrasting nature of these two workout types.

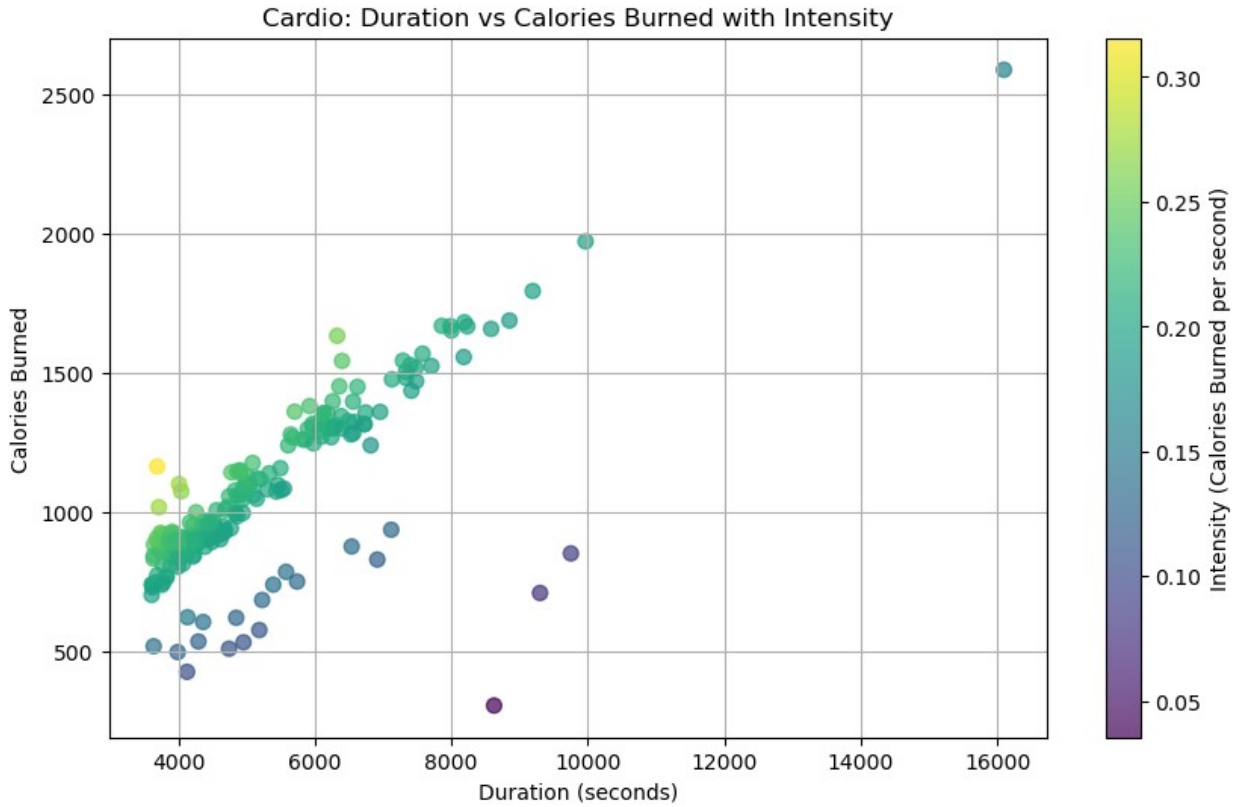
# Analysis Part 2

```
#cardio workout
plt.figure(figsize=(10, 6))

#matplotlib scatter
scatter = plt.scatter(
    cardio_df['Duration'].dt.total_seconds(),
    cardio_df['Calories Burned'],
    c=cardio_df['Intensity'],
    cmap='viridis',
    s=50,
    alpha=0.7
)

cbar = plt.colorbar(scatter)
cbar.set_label('Intensity (Calories Burned per second)')

plt.title("Cardio: Duration vs Calories Burned with Intensity")
plt.xlabel("Duration (seconds)")
plt.ylabel("Calories Burned")
plt.grid(True)
plt.show()
```




The scatter plot above visualizes the relationship between workout duration (in seconds) and calories burned during cardio workouts.

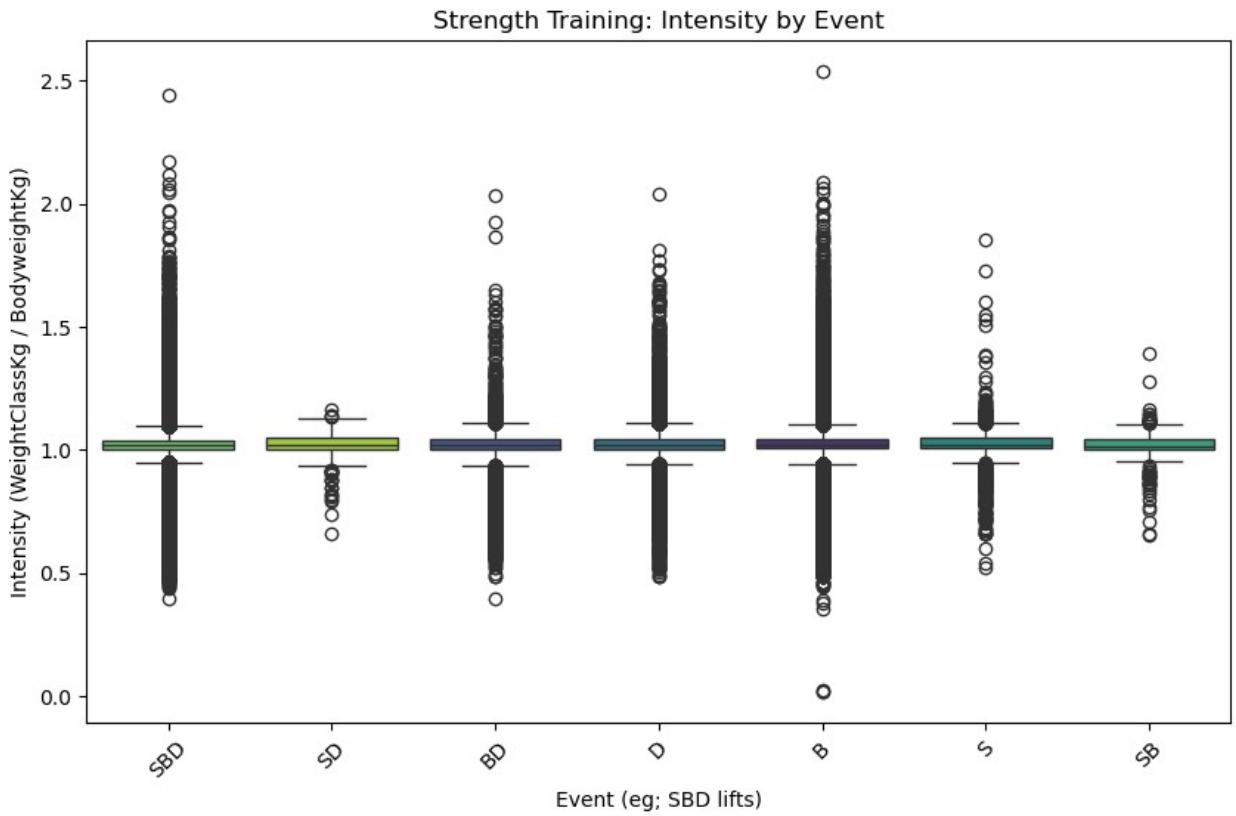
```
#for strength workout
unique_events = sorted(strength_df['Event'].unique())

color_palette = sns.color_palette("viridis", len(unique_events))

event_color_mapping = {event: color for event, color in
zip(unique_events, color_palette)}

#box plot
plt.figure(figsize=(10, 6))
sns.boxplot(
    data=strength_df,
    x='Event',
    y='Intensity',
    hue='Event',
    dodge=False,
    palette=event_color_mapping
)
plt.title("Strength Training: Intensity by Event")
plt.xlabel("Event (eg; SBD lifts)")
plt.ylabel("Intensity (WeightClassKg / BodyweightKg)")
plt.xticks(rotation=45)
```

```
plt.legend([], [], frameon=False)
plt.show()
```




The box plot above compares the intensity (calculated as WeightClassKg / BodyweightKg) across different strength training events (e.g., SBD lifts).

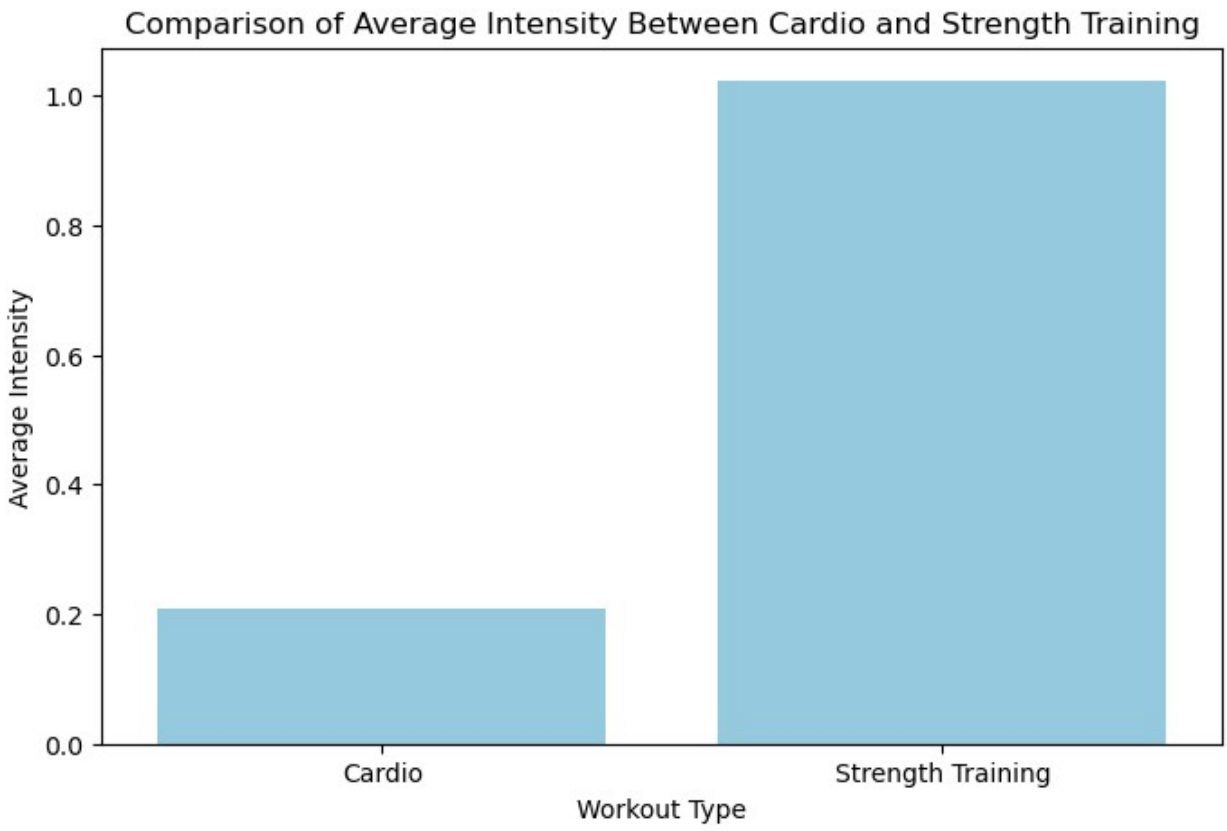
## Comparing both cardio workouts and strength training workouts

```
#bar chart
avg_cardio_intensity = cardio_df['Intensity'].mean()
avg_strength_intensity = strength_df['Intensity'].mean()

#data for bar chart
comparison_df = pd.DataFrame({
    'Workout Type': ['Cardio', 'Strength Training'],
    'Average Intensity': [avg_cardio_intensity,
avg_strength_intensity]
})

# Plot Bar Chart
```

```
plt.figure(figsize=(8, 5))
sns.barplot(data=comparison_df, x='Workout Type', y='Average
Intensity', color='skyblue')
plt.title("Comparison of Average Intensity Between Cardio and Strength
Training")
plt.ylabel("Average Intensity")
plt.xlabel("Workout Type")
plt.show()
```




This chart directly compares the average intensity between cardio and strength workouts.

## Evaluation of analysis part 2

The scatter plot demonstrates a positive correlation between workout duration and calories burned, with longer workouts generally resulting in higher calorie burns. The intensity, represented by a color gradient, provides insights into variations in workout effort across durations, allowing for the identification of outliers, such as sessions with exceptionally high or low intensity.

The box plot categorizes the intensity of strength training across different events, such as SBD lifts, effectively highlighting variations through the use of colors. It illustrates the diversity in

training intensities, showing how certain events demand more effort relative to body weight, as reflected in the intensity metric.

The bar chart comparing average intensities between cardio and strength training highlights their contrasting nature, with cardio emphasizing sustained calorie burn over time and strength training focusing on intensity relative to body weight.

# Analysis part 3- Creating a t-test

## For this analysis part, we will use the combined data to create a t-test-

```
cardio_data = cardio_table.select_dtypes(include=['number']).copy()
strength_data =
strength_table.select_dtypes(include=['number']).copy()

cardio_data.rename(columns={'Calories Burned': 'Calories'},
inplace=True)
strength_data.rename(columns={'BodyweightKg': 'Calories'},
inplace=True)

common_columns =
list(set(cardio_data.columns).intersection(set(strength_data.columns))
)
print("Updated Common Columns:", common_columns)

# performing the t-test
print("T-Test Results (Cardio vs. Strength):")
for column in common_columns:
    try:
        t_stat, p_value = ttest_ind(cardio_data[column],
strength_data[column], nan_policy='omit')
        print(f"{column}: t-statistic = {t_stat:.4f}, p-value =
{p_value:.4f}")
    except Exception as e:
        print(f"Error with column {column}: {e}")

#the interpretation
print("\nInterpretation:")
print("If p-value < 0.05, there is a statistically significant
difference between Cardio and Strength for the respective variable.")
```

```
Updated Common Columns: ['Calories']
T-Test Results (Cardio vs. Strength):
Calories: t-statistic = 661.3809, p-value = 0.0000

Interpretation:
If p-value < 0.05, there is a statistically significant difference
between Cardio and Strength for the respective variable.
```

## Evalution of analysis part 3

T-statistic is a measure of the difference between the means of two samples in terms of standard error.

Here, t-statistic=661.3809, where a high absolute value of the t-statistic indicates a larger difference between the two groups.

The p-value indicates the probability of observing the data (or more extreme) assuming the null hypothesis is true (i.e., no difference between the groups).

Also also since p-value=0.0000, it suggests that there is a statistically significant difference between the two groups for that variable.

The results of the t-test reveal a statistically significant difference between cardio and strength workouts in terms of the analyzed variable, "Calories Burned." With a t-statistic of 661.3809 and a p-value of 0.0000, the difference between the two groups is highly significant.

# Conclusion

This research provides valuable insights into the relationship between cardio and strength training and their impact on calorie burn, utilizing a robust dataset and analytical methods.

The calorie burn analysis (Part 1) highlighted that cardio workouts generally burn more calories on average, with a significant focus on sustained calorie expenditure. In contrast, strength training demonstrated a strong positive correlation between body weight and training intensity, revealing its emphasis on high-intensity effort relative to body weight. These insights align with the objective of understanding the distinct focuses of these two workout types and their implications for fitness planning.

The duration and intensity analysis (Part 2) further deepened the understanding of workout patterns. Cardio's positive correlation between session length and calorie burn emphasized the value of longer, consistent efforts, while strength training analysis underscored the variability in intensity across events. The statistical significance analysis (Part 3) confirmed these patterns, with the t-test providing robust evidence of the distinct impacts of cardio and strength training on calorie burn.

These findings achieved the research aims by providing actionable insights to guide individuals in effectively incorporating cardio and strength training into their fitness routines, supporting goals like weight loss, strength building, and overall health improvement.

# Practical applications based on conclusion

Cardio Workouts: Cardio is optimal for sustained calorie burn, making it suitable for individuals aiming for weight loss or endurance improvement. Longer durations contribute to higher calorie expenditure, with intensity varying depending on session length.

Strength Training: Strength workouts focus on increasing intensity relative to body weight, making them effective for building strength and muscle mass. The relationship between body weight and weight class highlights the tailored nature of strength training to individual physical capabilities.

Comparison of Workout Types: Cardio workouts prioritize calorie burn over time, while strength training emphasizes intensity. Both workout types offer unique benefits, and the choice between them should align with individual fitness goals.

# References


Your machine learning and Data Science Community (no date) Kaggle. Available at: https://www.kaggle.com/ (Accessed: 05 January 2025).

Verma, D. (2023) Cardio activities, Kaggle. Available at: https://www.kaggle.com/datasets/deependraverma13/cardio-activities (Accessed: 05 January 2025).

Dm, H. (2023) Powerlifting dataset, Kaggle. Available at: https://www.kaggle.com/datasets/docgenki/powerlifting-dataset (Accessed: 05 January 2025).

(No date) Welcome to the Apache Software Foundation! Available at: https://www.apache.org/licenses/LICENSE-2.0 (Accessed: 05 January 2025).