CALORIES BURNT PREDICTION APP AND FITNESS TRACKERS MARKET ANALYSIS

Presented By: Nitya Ravi

GitHub Account: NityaRavi12 (github.com)

Project Link: NityaRavi12/Feynn-Labs-Task-2- (github.com)

"Some people call this artificial Artificial intelligence, but the reality is this technology will enhance us. So instead of Artificial Intelligence, I think we will augment our intelligence"

-Ginni Rometty

Abstract:

People are less conscious of their mental stability and overall health in this age of rapidly advancing technology. Because of time constraints, they consume more junk food than wholesome alternatives, which raises their body's overall calorie intake which is one of the main contributors to obesity. A calorie is the rate at which energy is consumed and stored. The goal of the "Calorie Burnt Prediction App by Machine Learning Algorithm" is to use machine learning techniques to forecast how many calories a person will burn while engaging in physical activity. The app typically utilizes data such as the user's age, weight, height, gender, and the type/duration of activity to calculate the calories burnt. I have employed a range of machine learning such as random forest, SVM, XGBoost, linear regression, XG boost and decision tree to estimate the number of calories burned. The findings show that the XGBoost model has a minimum mean absolute error of calories when it comes to accurately predicting calorie burn. This work has potential applications for personalized health coaching and wellness tracking, and it adds to the growing body of research on the use of machine learning for fitness and health applications.

Keywords- ML (Machine learning), XGBOOST, Decision tree, linear Regression, Support Vector Machine(SVM), Random Forest.

Problem Statement:

A methodological approach to analyze the calories burnt using the concept of machine learning to train the model accordingly and predict whenever required via an app. In today's sedentary lifestyle, individuals often struggle to accurately estimate the calories burnt during physical activities, which can hinder their fitness goals and overall health. To address this challenge, the aim is to develop a machine learning-based app that predicts the calories burnt by an individual during various physical activities. This project will leverage machine learning algorithms to analyze user-specific data, including biometric information (such as age, weight, height, and gender) and activity parameters (such as type of exercise and duration) to build a calorie burn prediction app. The goal is to create an app that accurately estimates calorie expenditure, enabling users to make informed decisions about their exercise routines and dietary habits. This app will not only empower individuals to track their fitness progress more effectively but also provide valuable insights for healthcare professionals and fitness enthusiasts alike.

Introduction:

In today's fast-paced world, where maintaining a healthy lifestyle is becoming increasingly essential, the need for accurate fitness tracking tools has never been greater. The recording of calories burned during physical activity is an essential component of fitness tracking. Gaining an understanding of calorie expenditure is essential to reaching fitness objectives, be they managing weight, enhancing sports performance, or just living a healthy lifestyle.

The creation of a Calorie Burnt Prediction App appears to be a workable response to this expanding requirement. This cutting-edge smartphone app seeks to give users instantaneous estimates of the number of calories they burn while engaging in different workouts and activities. The software provides individualized insights into calorie expenditure by utilizing the power of machine learning algorithms and user-specific data, such as biometric data (age, weight, height, and gender) and activity parameters (kind of exercise, intensity, and duration).

The Calorie Burnt Prediction App uses sophisticated prediction models to produce precise estimations that are customized for each user, going beyond simple calorie monitoring. The software gives users insightful feedback on their energy expenditure throughout cardio workouts, weight training, yoga sessions, and outdoor activities, enabling them to make well-informed decisions regarding their fitness regimens.

Additionally, the app functions as a feature-rich fitness partner, providing tools like goal-setting, activity tracking, progress tracking, and tailored recommendations. The software is suitable for users of all fitness levels, from novices to seasoned athletes, thanks to its simple layout and appealing design.

The Calorie Burnt Prediction App is a potent tool for encouraging physical exercise, improving general well-being, and developing a culture of fitness and self-improvement in this age of health consciousness and technological innovation. Let the Calorie Burnt Prediction App be your dependable walking partner as we set out on this path to healthier lifestyles and reach your fitness goals.

Market/Customer/Business Need Assessment:

Ways to generate revenue:

- **Subscription fees:** This application provides a paid subscription service that grants users access to extra features like customized training regimens, sophisticated analytics, and access to unique content.

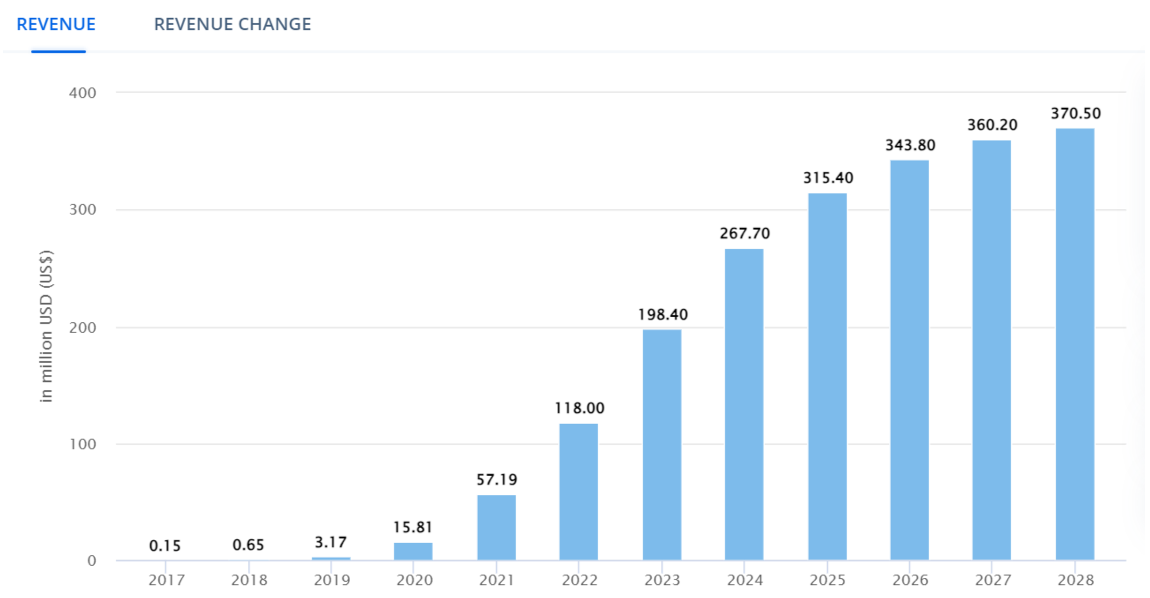
- **Selling data:** This app collects data about the users' fitness habits and sells it to third-party companies. Companies can use this data to develop new products and services or target users with personalized advertising.

- **Offering white-label solutions:** The app developers offer white-label solutions to other businesses. This allows businesses to create their own branded fitness apps without developing their technology.
- **Affiliate marketing:** This app can partner up with other businesses to promote their products and services. When a user clicks on an affiliate link and makes a purchase, the app developer earns a commission.
- **Advertising:** This app can generate revenue by displaying ads to users. The ads may be related to fitness and wellness products and services, or they may be more general.
- **In-app purchases:** This app offers in-app purchases that allow users to unlock new workouts, features, or equipment.

Customer need Assessment:

- **Tracking:** The App tracks the user‘s steps automatically. There is no need to start your session every time you go for a run or a walk. Users earn points (Fitcoins) based on the number of steps.
- **Public Challenges:** Users can participate in Public Challenges, and if they come up as winners, they get rewarded.
- **Private Challenges:** Via this option, users can throw fitness challenges to their friends/ colleagues/ family, and watch their progress while cheering and motivating each other.
- **Redeem Rewards**: Users can redeem these fit coins for different types of Exclusive Rewards by 25+ Reward Partners across different verticals. Rewards can be freebies or heavy discounts on various products and services.

Market Assessment:



The user base covers both paying and non-paying customers. Revenue figures in this market only include revenues generated from paid apps offering premium options and from in-app purchases. Revenues from app downloads and advertising are not included. The data provided specifically reflects business-to-consumer (B2C) revenues and does not include business-to-business (B2B) or consumer-to-consumer (C2C) revenues.

Target Specifications and Characteristics:

- To provide push notifications and reminders
- To provide progress tracking
- Help set the goal
- To provide a Customized diet plan
- Planning personalized exercise routine
- Comparing performances
- Online Consultation and Chat
- Multidevice Synchronization
- Social Sharing
- Water intake tracker
- Tracking number of steps
- Setting up user profile
- Heart monitoring

External Searches:

The number of calories burned is subjective and varies for each person based on their height, weight, and level of fitness. It is also dependent on internal and external factors. Calories are a measure of heat energy, but most people associate them with weight loss or eating less. From a human standpoint, the quantity of energy needed to complete a task is measured in calories. There are differences in the calorie values associated with different items. The human body produces heat energy when it engages in prolonged physical activity or workouts because it raises its body temperature and heart rate. which finally results in the burning of calories.

To demonstrate the same, we apply various regression algorithms, including linear regression, XG Boost regression, SVM, Decision tree regression, and Random Forest regression over the data to obtain the best and optimal results. We take some input parameters, like age, gender, height, and weight.

Factors affecting calories burnt during exercise are:

1. Age (in years) –As age increases the need for calories decreases in the body. This is often due to less physical activity, changes in metabolism, or age-related loss of bones and muscle mass.

2. Height- Taller people have greater lean mass, which is related to metabolic rate .which implies that people need more calories to function. Hence the calorie rate in taller people will be more than in shorter ones.

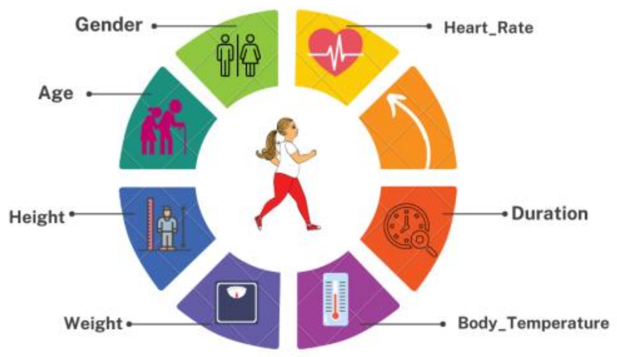
3. Weight- If you intake more calories than the calories you burn your weight will start rising. it means the more the weight of a person the more they need to burn calories to balance their intake calories.

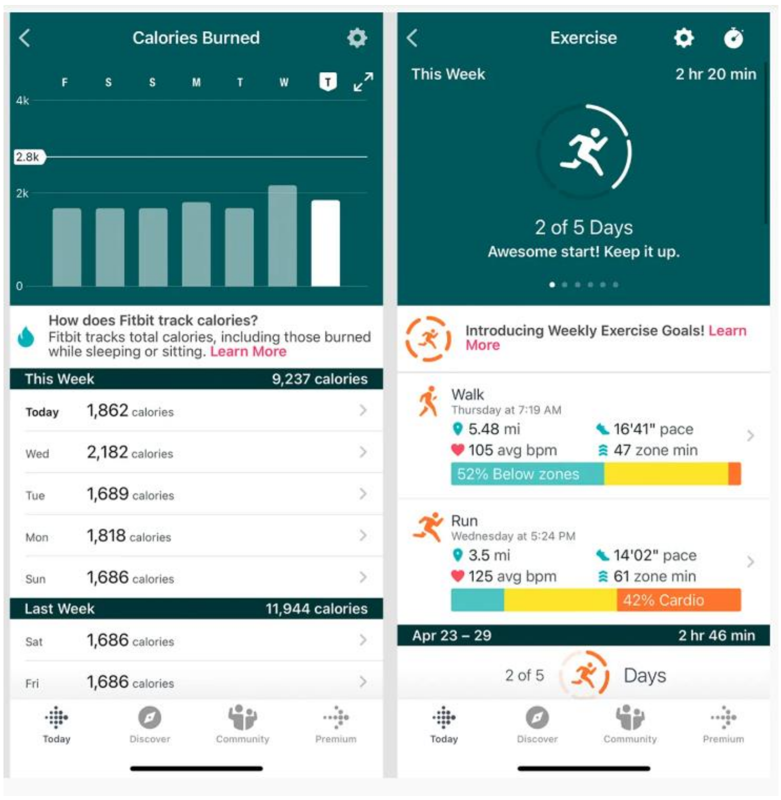
4. Gender- Human body's calorie needs vary according to gender, women need fewer calories than a man Because men have greater muscle mass.

5. Heart rate- While exercising heart rate increases and shows the exertion in doing an activity which determines the calories you burn. Also, a higher heart rate leads to body fat burn which is a composition of calories.

6. Body Temperature- As body temperature rises, more energy is used to cool it down which leads to calorie burn.

7. Duration- With the increase in the duration of exercise body temperature also increase, which leads to extra calorie burn hence the duration of exercise directly affects the calories burned in the body.





How does Fitbit track calories?
Fitbit tracks total calories, including those burned while sleeping or sitting. Learn More

This Week	9,237 calories
Today	1,862 calories
Wed	2,182 calories
Tue	1,689 calories
Mon	1,818 calories
Sun	1,686 calories
Last Week	**11,944 calories**
Sat	1,686 calories
Fri	1,686 calories

This Week 2 hr 20 min

2 of 5 Days
Awesome start! Keep it up.

Introducing Weekly Exercise Goals! Learn More

Walk
Thursday at 7:19 AM
5.48 mi 16'41" pace
105 avg bpm 47 zone min
52% Below zones

Run
Wednesday at 5:24 PM
3.5 mi 14'02" pace
125 avg bpm 61 zone min
42% Cardio

Apr 23 – 29 2 hr 46 min
2 of 5 Days

The World Health Organization (WHO)- There are many factors that affect the calories burned, but anyone can be modified their diet chart or activity level to get the desired results. There is a study in the literature that used ml and data mining to diagnose problems. When we compare from today's scenario some articles are published earlier with low accuracy of calories burned prediction problems.

Applicable Patents available for the app:

The patent application required: Provisional Patent Application

This application will require a provisional patent for the initial steps. A provisional patent preserves the confidentiality of this application without publication. Patents are normally granted based on priority. So, to claim first priority over any invention, a provisional patent application is useful. A provisional patent means our invention is still in the process of development. Besides, we are sure that it will be completed within a year. So, all applicants of a provisional patent get a year to finalize the invention, after applying for the registration.

Advantages:

- Cost effective: The Act provides for provisional patents to give investors and inventors time to obtain the necessary funds to file for a complete patent or retain the services of a patent agent. The expense of applying for a provisional patent is significantly less than that of a full patent.

- Tag: When attempting to determine the commercial potential of the new design or process, the inventors/investors may use the tag "Patent Pending" after receiving a provisional patent.
- Interim protection: A person filing for a provisional patent is granted protection for 12 months from the date of the initial filing, provided that the filing is accepted. This makes sure they can either continue to refine their idea in the interim or have enough time to finish any necessary steps before receiving a full patent.

Applicable Constraints:

Building a calories burnt prediction application involves various expenses, including software development, design, testing, and maintenance. To mitigate costs, developers may need to prioritize features and functionality based on budget constraints. Using cost-effective development tools and frameworks can also help reduce expenses.

The cost to develop a calories burnt prediction app varies on certain factors. These factors include the project's complexity, UI/UX design, tech stack, and more. If we are looking for any specific number – then that's not the case here in the mobile app development process.

Having an advanced version would increase the cost and vice versa. The estimated cost to develop a calorie burnt prediction app would be around $25,000 to $50,000. If we want advanced features and functionalities in our app, then the estimated cost would be $80,000 or above.

To get users, the app may need to be heavily promoted and marketed, which might be expensive. As they optimize the app for the app store and increase its visibility through organic channels, developers must allocate funds for user acquisition campaigns, advertising, and app store optimization.

This app would need $25,000 to go with a minimal viable product (MVP) to create a health and fitness app. When we have enough of our intended audience as our foundation in the market, we may approach investors for money to acquire completely developed health and fitness applications.

Storing user data, such as biometric information and activity logs, requires storage space. We must consider the volume of data generated by users and choose cost-effective storage solutions that can accommodate growth over time.

Compliance with data protection regulations and industry standards may require additional resources and expertise. There will be continuous expenses for hosting, bandwidth, and maintenance if the application needs server-side processing or user data storage. Selecting effective and scalable cloud services can assist in controlling these costs. Supporting a wide range of devices with varying screen sizes, resolutions, and operating systems can increase development and testing costs. Developers must prioritize compatibility with popular devices while considering the trade-offs between performance and cost.

Business Model:

Generating revenue from the app can offset development and maintenance costs. However, we must carefully consider the app's pricing model (e.g., freemium, subscription, in-app purchases) and strike a balance between maximizing revenue and providing value to users.

Monetization Strategies:

- Subscription Model: To grant customers access to premium services and content, a recurrent subscription charge is required. Monthly, quarterly, or annual subscriptions are available, giving consumers continuous access to exclusive features and updates.
- Affiliate marketing: Collaborate with producers of exercise gear, manufacturers of dietary supplements, or internet merchants to advertise their goods and services inside the app. Receive commissions for purchases made via referrals or affiliate links.
- White Labeling: By licensing the software, fitness centers, gyms, and corporate wellness initiatives can personalize and brand the app for their patrons or staff members. Based on the number of users or features offered, charge a licensing or membership fee.
- Sponsorships and Partnerships: Work together to co-create challenges, events, and content for the app with corporate sponsors, health brands, and fitness influencers. Make money via brand alliances, affiliate marketing contracts, and sponsorship agreements.
- Freemium Model: Provide the software with basic functions, like tracking calories burned from everyday activities, for free. Then, through in-app purchases or subscriptions, users can access premium services like customized training regimens, sophisticated analytics, or a wider selection of workouts.
- In-App Purchases: Offer users the option to purchase additional features, content, or virtual goods within the app. For example, users could buy workout plans created by fitness experts, premium recipes for healthy meals, or ad-free browsing.

Client Groups:

- Fitness Enthusiasts: People who are actively involved in fitness pursuits and are looking for precise performance evaluation and calorie tracking.
- Health-conscious People: To control their weight, users concentrate on leading a healthy lifestyle and tracking their caloric intake and expenditure.
- Professional sportsmen and sports teams need to manage calories precisely to optimize their training and improve their performance.

Concept Development:

- Data Description:
  User input- Users enter into the app their gender, age, height, and weight among other personal details. They could also elaborate on the kind and length of the physical exercise they intend to engage in. Data collection is an essential process in any machine learning project, as the quality of the data used has a significant impact on the performance of the resulting model.

- Data preprocessing:
  Pre-processing of the dataset is carried out to make data free from null values and keep the important attributes in the dataset. Because preprocess datasets are appropriate for applying into the algorithm for training and testing. We split the data into a training set (80% of the data) and a test set (20% of the data) for model training and evaluation.

- Evaluation of the Performance of Machine Learning Models:
  The app employs a predictive algorithm or model trained on a dataset of activities and corresponding calories burnt. This model takes into account user input and activity details to estimate the calories burnt.
  We evaluated the performance of four machine learning models: support vector machine (SVM), random forest, linear regression, XGBoost regression, and Decision Tree.

  Linear regression: In a linear regression model, the predictor and dependent variables are correlated linearly with one another. It is employed to determine how dependent two variables are on one another. It computes the temperature increase based on the quantity of exercise performed.

  XG boost regression: The acronym represents extreme gradient boosting. Through the combination of several weak models' predictions, this ensemble learning model generates strong predictions. It manages big datasets and offers effective missing value management.

  SVM(support vector machine): it tries to find a hyperplane that separates two classes and then classifies a new point depending on whether it lies on the positive or negative side of the hyperplane depending on the class to predict.

  Decision tree regression: It builds a regression model in a tree structure. To start constructing we start with a feature that will become the root node. the feature with the least impurity is considered as a node at any level sorts the data in ascending order and calculates the average of adjoining values. Then the impurity at each level is calculated.

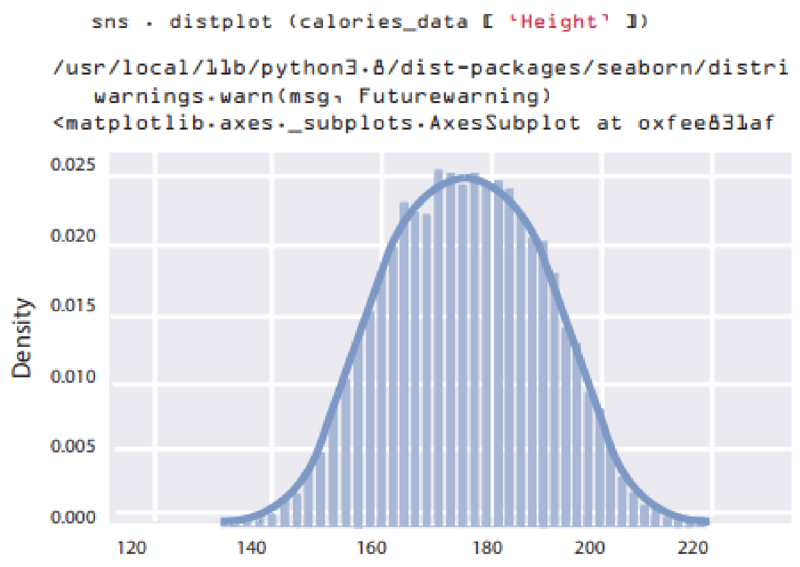
  Random forest regression: It is a supervised learning algorithm this model combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model. It is powerful and accurate

- Comparison of Feature Selection with Individual Evaluators:

The findings of several algorithms are compared, and the most accurate one is then used to predict how many calories will be burned during activity in addition to several other variables including height, weight, body temperature, and heart rate. We contrasted the performance of models that used all features to models that used feature selection methods such univariate feature selection and recursive feature elimination. To determine the relevance of the feature, we employed the correlation matrix.

- Deriving Key Features:
  By examining the model's feature importance scores, we were able to identify important characteristics. Heart rate, duration, and temperature were the main parameters. In conclusion, our work employed machine learning models to forecast the number of calories burned during physical exercise based on data. In addition to preprocessing the data and assessing the models' effectiveness, we also developed a forecasting system for any real-time data.

- Data Visualization:



```
   sns . distplot (calories_data [ 'Height' ])

/usr/local/lib/python3.8/dist-packages/seaborn/distri
   warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at oxfee831af
```

Fig: Plotting the gender column in the count plot

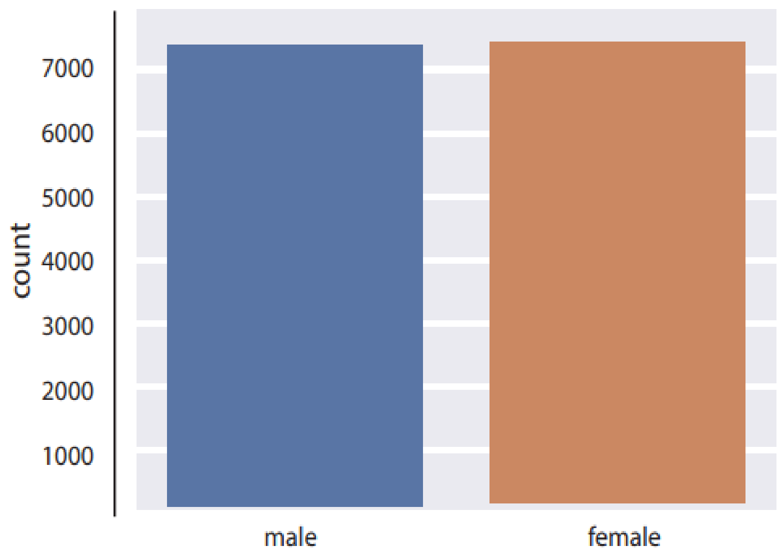


Fig: finding the distribution of Height column

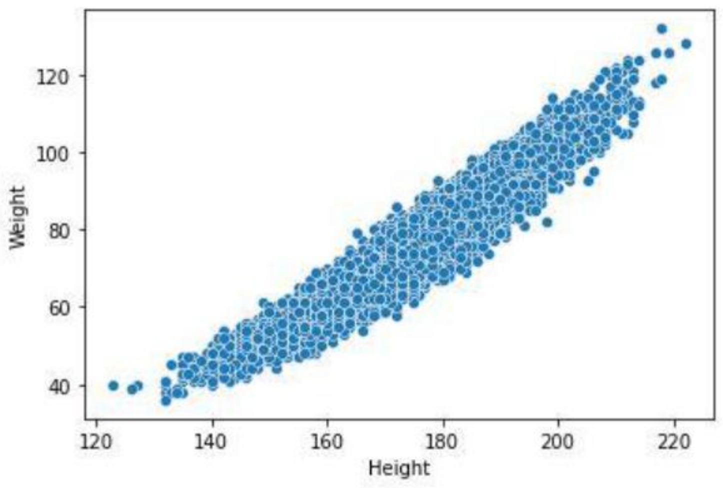


Fig: Scatterplot of height vs weight

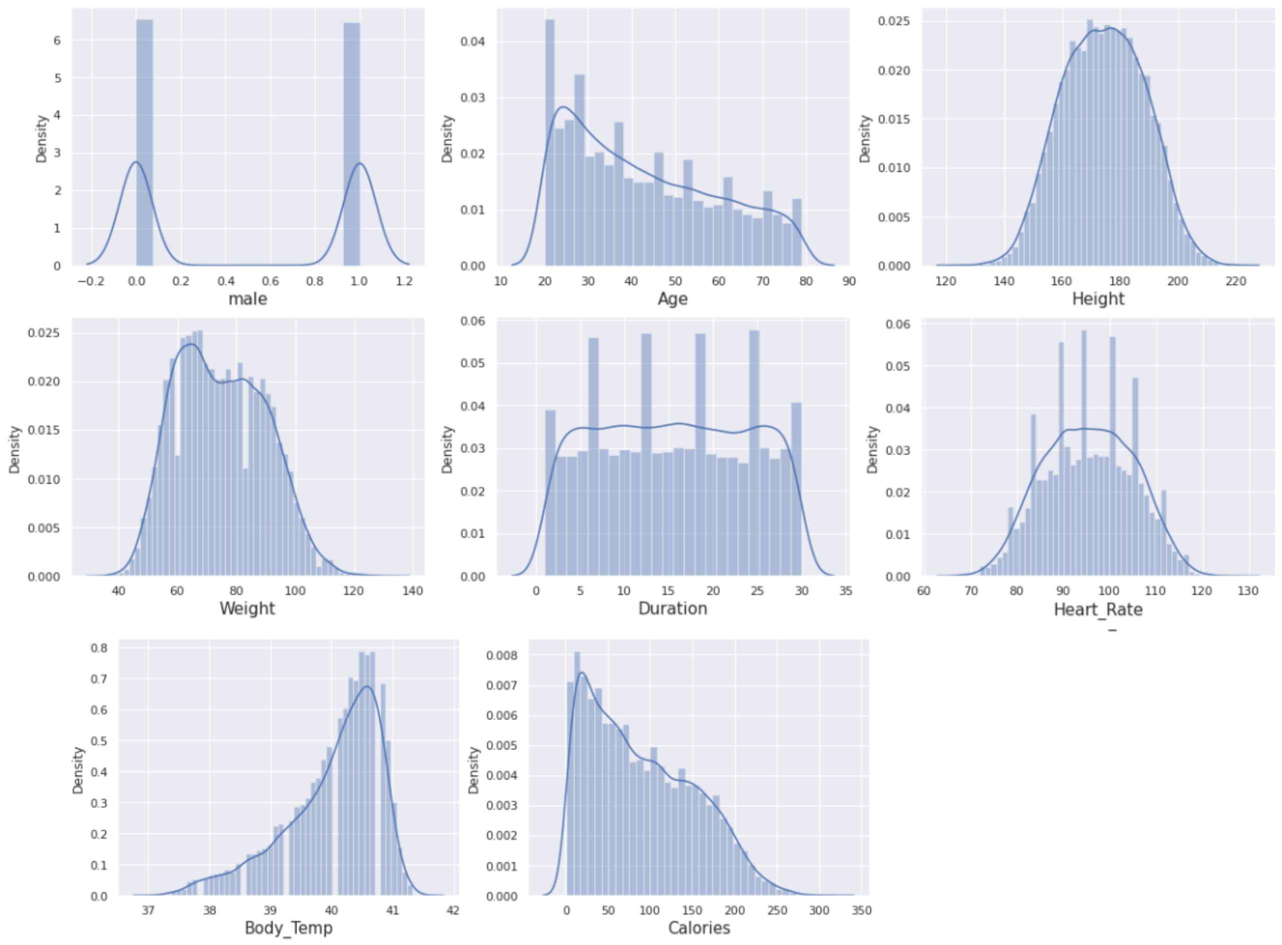


Fig: Distribution plot for continuous features

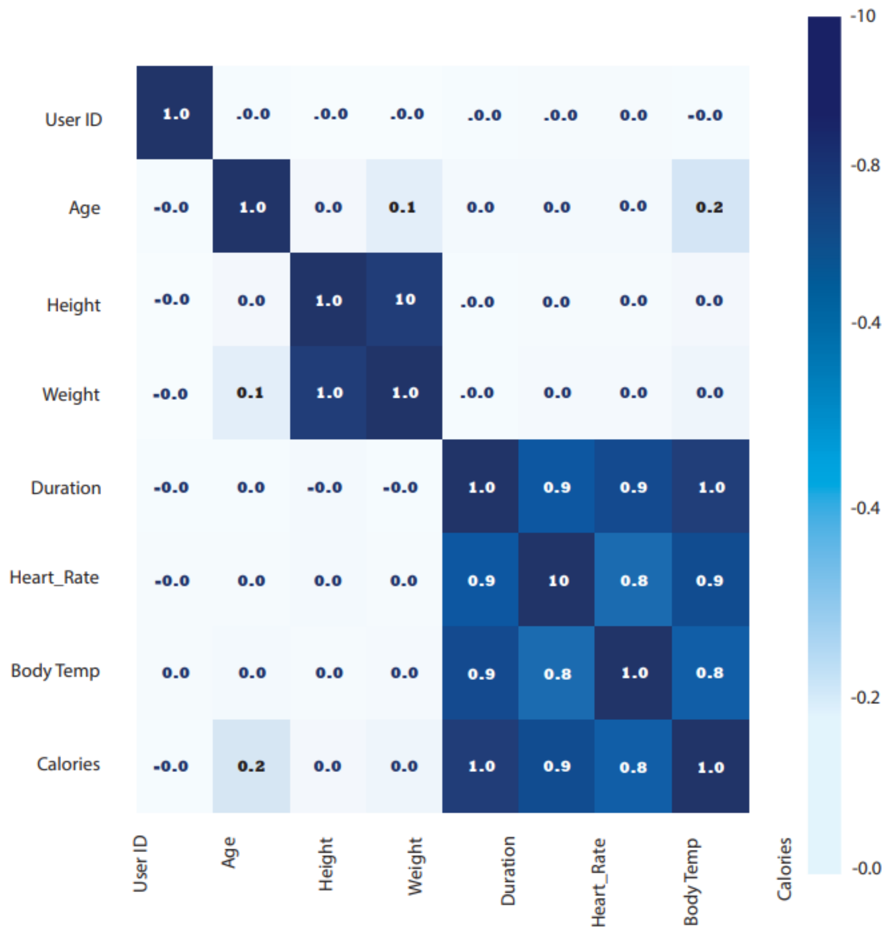


Fig: Heatmap to detect highly correlated features

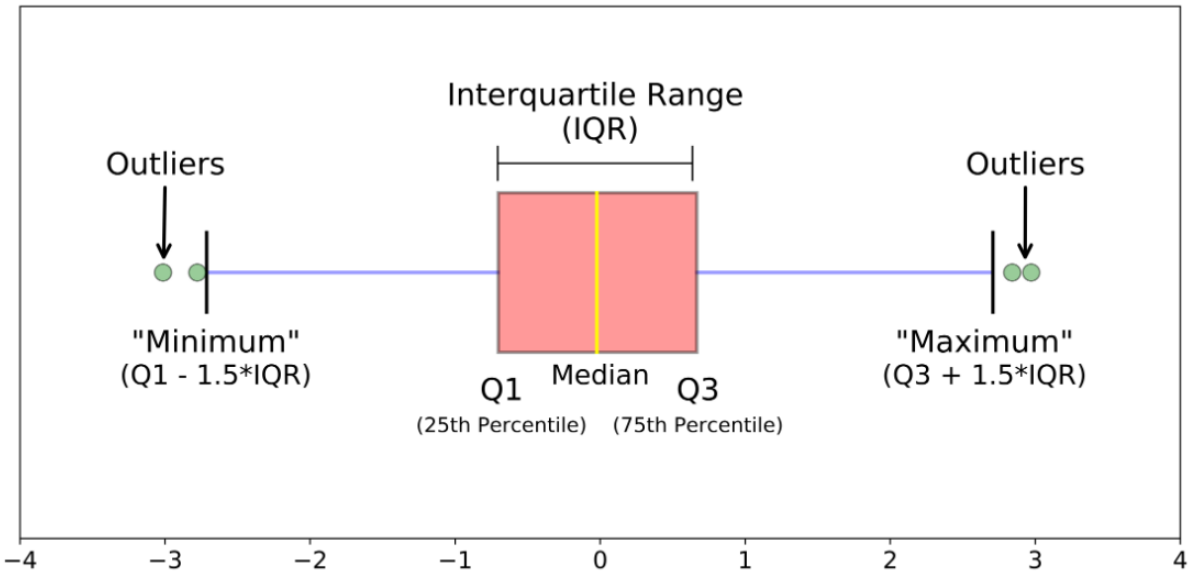


Fig: Box plot of median, Interquartile range and outliers

Final Product Prototype:

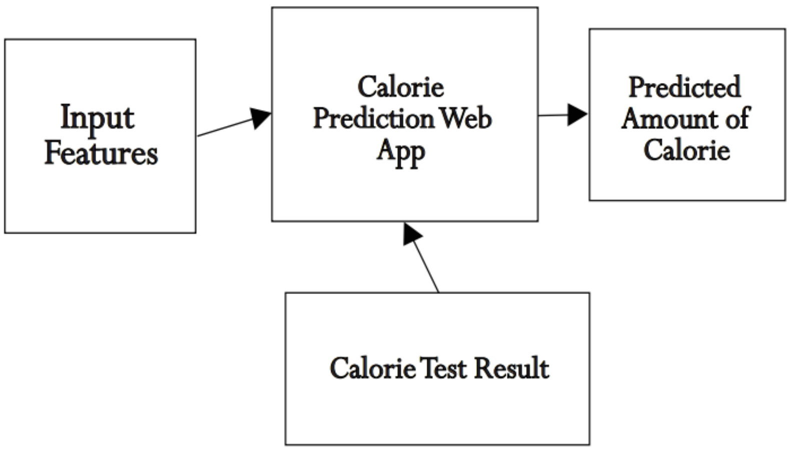


Fig: How to predict calorie burnt by web App

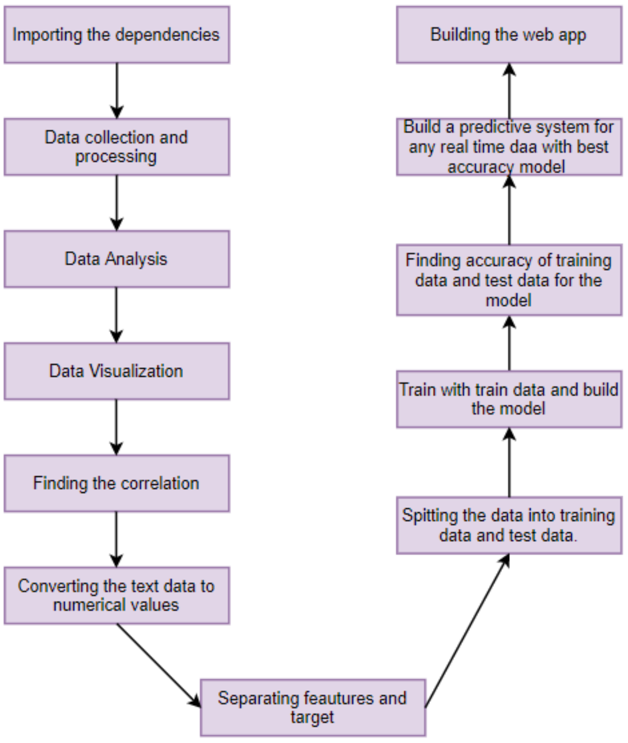

Importing the dependencies
Data collection and processing
Data Analysis
Data Visualization
Finding the correlation
Converting the text data to numerical values
Separating feautures and target
Spitting the data into training data and test data.
Train with train data and build the model
Finding accuracy of training data and test data for the model
Build a predictive system for any real time daa with best accuracy model
Building the web app

Fig: Workflow Diagram

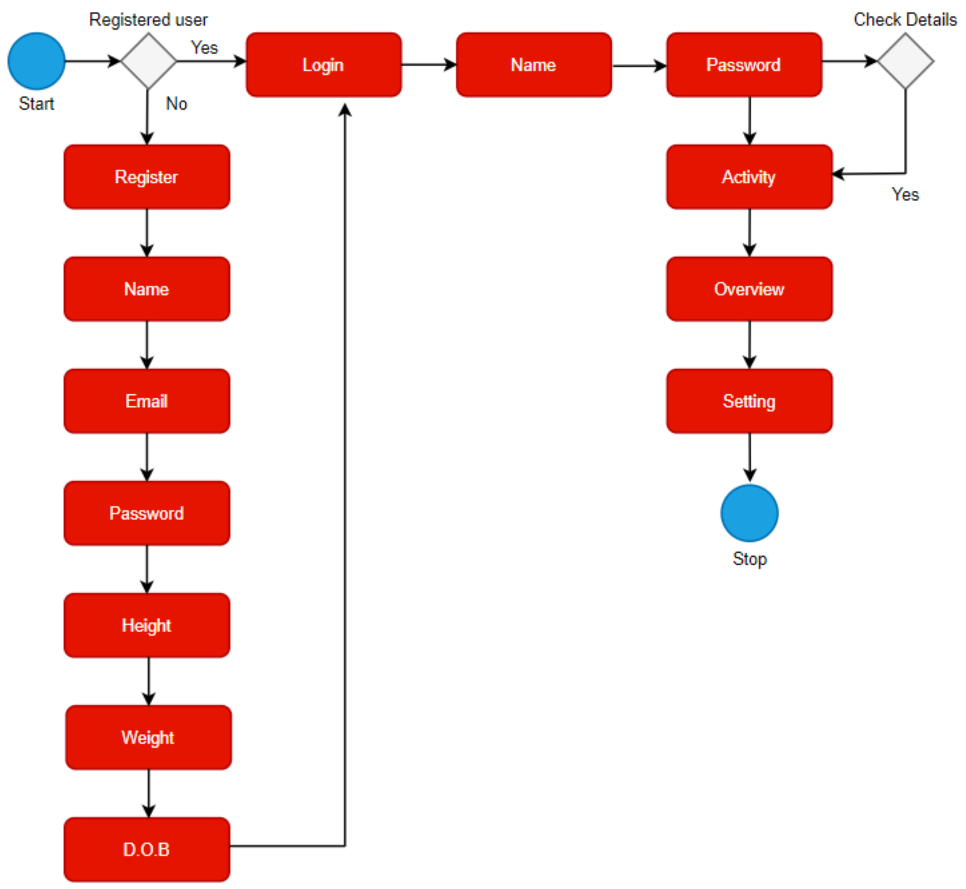


Fig: State diagram of the app

This prototype represents the culmination of extensive research, user feedback, and iterative design and development cycles aimed at creating a user-centric and feature-rich application.

Backend Technologies:

- Python: Because of its many libraries and adaptability, Python is a popular choice for creating the backend of a calorie-burn prediction program. NumPy and Pandas are two libraries that we can use for processing and manipulating data.

- Web Framework: Django or Flask: Popular Python web frameworks that speed up the development of backend applications include Django and Flask. While Flask offers more flexibility for smaller applications, Django offers a more extensive structure.

- Database Management System (DBMS): Relational database management systems (DBMS) like SQLite, PostgreSQL, or MySQL are frequently used with Flask or Django to store activity logs, user data, and other pertinent data.

- Machine Learning Libraries: Scikit-learn: scikit-learn is a potent Python machine-learning toolkit that offers tools for creating regression models that can be used to forecast calories. TensorFlow or PyTorch: TensorFlow or PyTorch can be used for deep learning and neural network-based predictions if you want to use more sophisticated machine learning models .Pandas – This library helps to load the data frame in a 2D array format and has multiple functions to perform analysis tasks in one go. Numpy – Numpy arrays are very fast and can perform large computations in a very short time. Matplotlib/Seaborn – This library is used to draw visualizations.

- API Development: Using the Django REST Framework or Flask-RESTful framework, you may build RESTful APIs to manage communication between our application's front-end and back-end parts.

Frontend Technologies:

- HTML, CSS, and JavaScript:
  HTML (HyperText Markup Language): Used for structuring the content of web pages.
  CSS (Cascading Style Sheets): Used for styling the appearance and layout of web pages.
  JavaScript: Used for adding interactivity and dynamic behaviour to web pages, such as form validation and user interface enhancements.

- Library/Framework for Front End:
  React, Angular, or Vue.js are frontend frameworks and libraries that offer components and capabilities for quickly and effectively creating dynamic user interfaces. React is a well-liked option because of its ecosystem of libraries and component-based architecture.

- Libraries for Data Visualization:
  Using the Chart.js or D3.js frameworks, you may make dynamic charts and visualizations that show trends and statistics related to calorie burn within your app.

- Requests over HTTP:
  Use the Fetch API or Axios to retrieve data for the frontend interface by sending asynchronous HTTP queries to the backend server.

Various user interfaces: The user needs to have a wide range of parameter alternatives to select from. Only after extensive testing and analysis of all the edge cases can this be optimized. Raw and unintelligible data: It will be returned by the interactive visualization of the data taken from the trained models. This needs to be written in an aesthetically pleasing and "easy to read" manner.
Feedback system: To identify the needs of the consumer that have not been satisfied, a useful feedback system needs to be created. This will support our ongoing model training.

Product Details:

- How does it work?
  -The calorie burn app estimates the number of calories expended during physical exercise by using user-provided data, including age, height, weight, gender, and type of activity.
  -Users provide information about their training sessions, such as the kind of exercise, length, and degree of difficulty.
  -Utilizing machine learning techniques, the app analyzes this data to produce customized calorie burn estimates depending on user attributes and activity parameters.
  -Users can monitor their fitness development and make educated choices regarding their exercise regimens and eating habits by using the real-time input on their calorie expenditure that they receive.

- Data Sources:
  - Activity Databases: Pre-existing databases that have been compiled from reliable sources, including fitness associations, academic research centers, and medical facilities, and that provide calorie burn estimates for a range of physical activities and exercises.
  - Data sources include user-provided information such as age, height, weight, gender, and activity details that are directly entered into the app by the user.

- Algorithms, Frameworks, Software, etc. Needed:
  - Machine Learning Algorithms: Regression algorithms, such as linear regression or polynomial regression, are employed to predict calorie expenditure based on user data and activity parameters.
  - Frameworks/Libraries: Python-based frameworks such as Django or Flask for backend development, scikit-learn for machine learning tasks, and React.js or Angular for frontend development.
  - Database Management System (DBMS): PostgreSQL, SQLite, or MySQL for storing user data and activity logs.
  - Development Tools: Integrated development environments (IDEs) such as PyCharm or Visual Studio Code, version control systems like Git, and project management tools for team collaboration and task tracking.

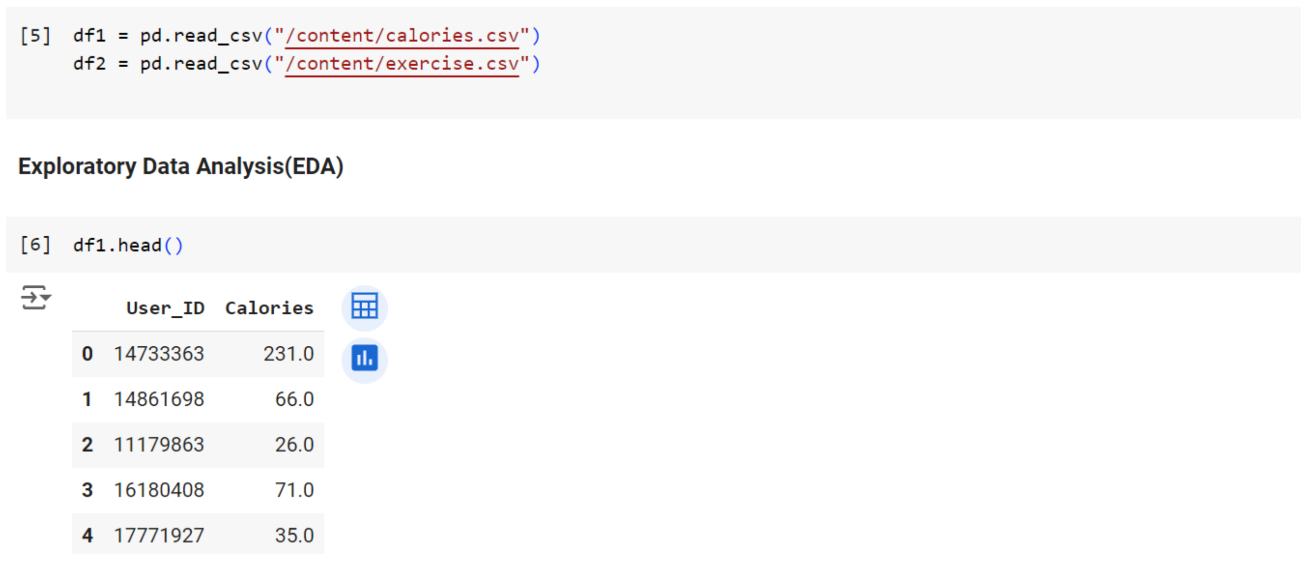
- Team required to build the app:
  - Backend Developers: Responsible for building the server-side logic, handling data processing, and implementing machine learning algorithms.
  - Frontend Developers: Design and develop the user interface, ensuring a seamless and intuitive user experience.

- Machine Learning Engineers/Data Scientists: Design and implement machine learning models for calorie prediction, fine-tuning algorithms for accuracy and efficiency.
- Database Administrators: Manage database systems, optimize data storage and retrieval, and ensure data integrity and security.
- Quality Assurance/Test Engineers: Conduct thorough testing of the app to identify and resolve any bugs or issues, ensuring reliability and performance.
- UI/UX Designers: Create visually appealing and user-friendly interfaces, focusing on usability, accessibility, and aesthetics.

• Cost:
  - Maintenance Costs: The total cost of ownership of the app includes ongoing maintenance, updates, and support services.
  - Development Costs: The price of creating a calorie-burning app varies depending on a number of variables, such as the team size, technology stack, complexity of features, and length of development time. A few thousand to tens of thousands of dollars could be involved.
  - Infrastructure Costs: The total cost is also affected by costs for database maintenance, cloud computing, and server hosting.

Code Implementation/ Validation on a small scale:

<u>Dataset 1 and 2</u>

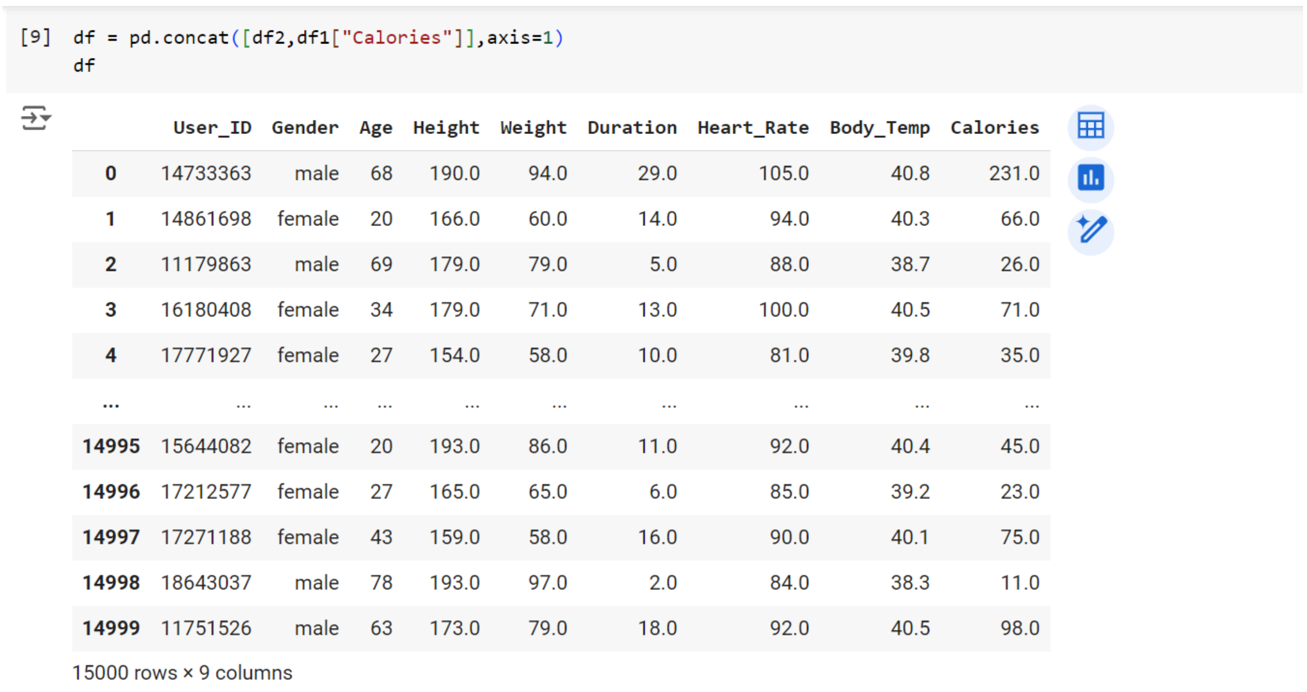


```
[5] df1 = pd.read_csv("/content/calories.csv")
    df2 = pd.read_csv("/content/exercise.csv")
```

Exploratory Data Analysis(EDA)

```
[6] df1.head()
```

	User_ID	Calories
0	14733363	231.0
1	14861698	66.0
2	11179863	26.0
3	16180408	71.0
4	17771927	35.0



```
[9]  df = pd.concat([df2,df1["Calories"]],axis=1)
     df
```

	User_ID	Gender	Age	Height	Weight	Duration	Heart_Rate	Body_Temp	Calories
0	14733363	male	68	190.0	94.0	29.0	105.0	40.8	231.0
1	14861698	female	20	166.0	60.0	14.0	94.0	40.3	66.0
2	11179863	male	69	179.0	79.0	5.0	88.0	38.7	26.0
3	16180408	female	34	179.0	71.0	13.0	100.0	40.5	71.0
4	17771927	female	27	154.0	58.0	10.0	81.0	39.8	35.0
...	...	...	...	...	...	...	...	...	...
14995	15644082	female	20	193.0	86.0	11.0	92.0	40.4	45.0
14996	17212577	female	27	165.0	65.0	6.0	85.0	39.2	23.0
14997	17271188	female	43	159.0	58.0	16.0	90.0	40.1	75.0
14998	18643037	male	78	193.0	97.0	2.0	84.0	38.3	11.0
14999	11751526	male	63	173.0	79.0	18.0	92.0	40.5	98.0

15000 rows × 9 columns

1.User_ID: The ID of the person who is unique.

2.Gender: Gender of the person.

3.Age: Age of the person.

4.Height: Height of the person in cm.

5.Weight: Weight of the person in kg.

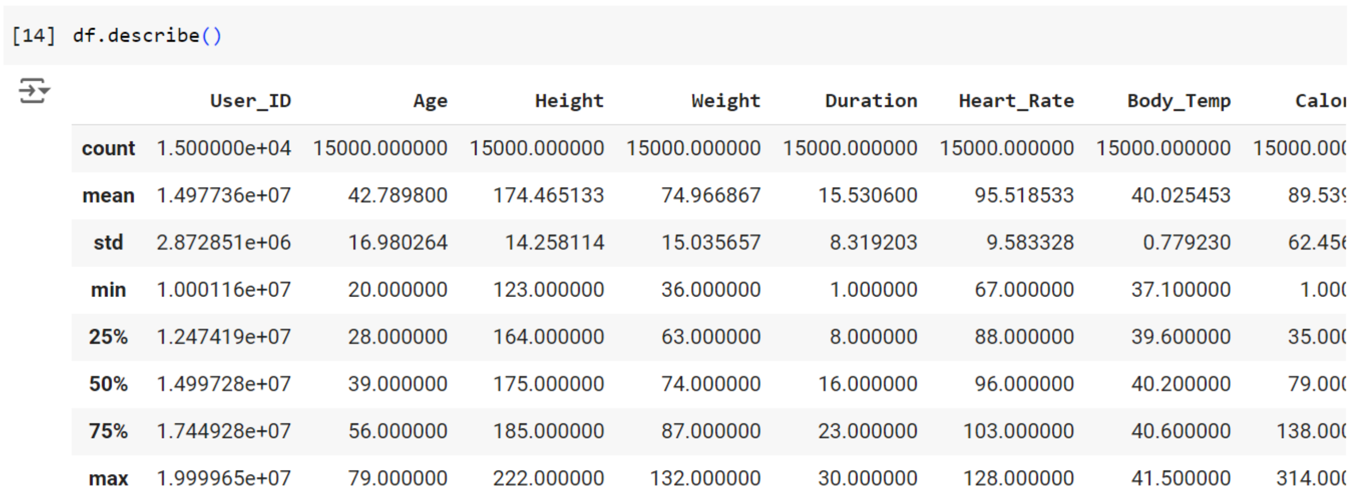
6.Duration: Duration of the person's exercise/activity.

7.Heart_Rate: Heart rate per minute of the person.

8.Body_Temp: Body temperature of the person in Celsius.

9.Calories: Calories burned in kilo calories.

Dataset's Overall Statistic

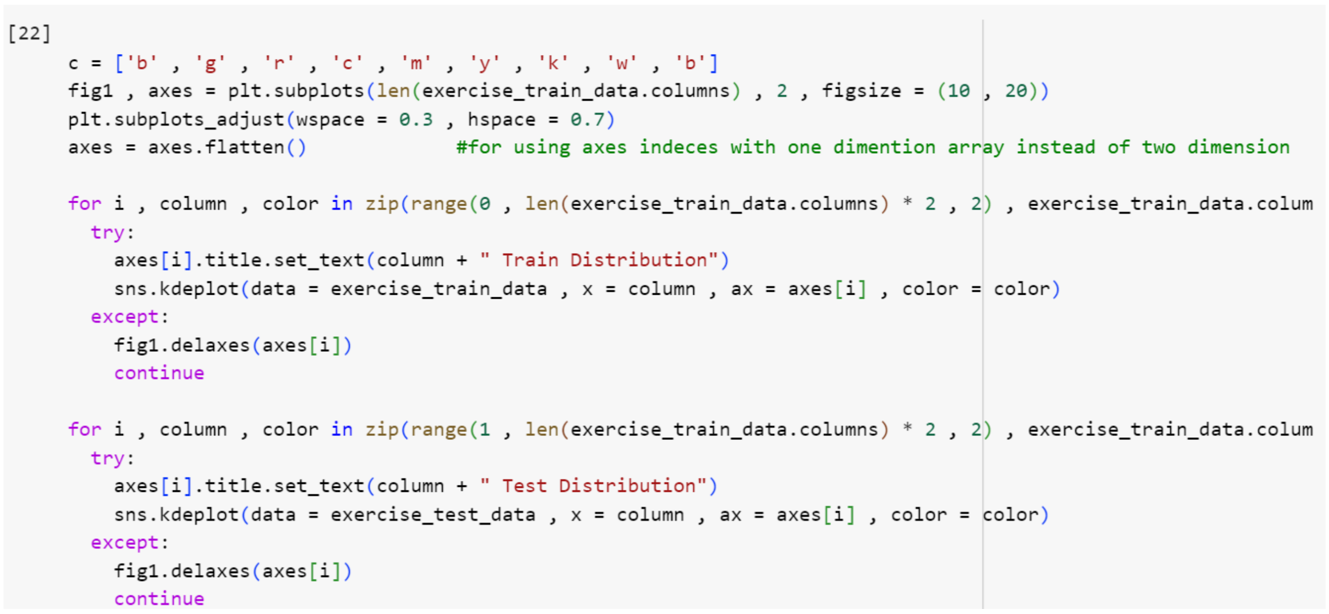


```
[14] df.describe()
```

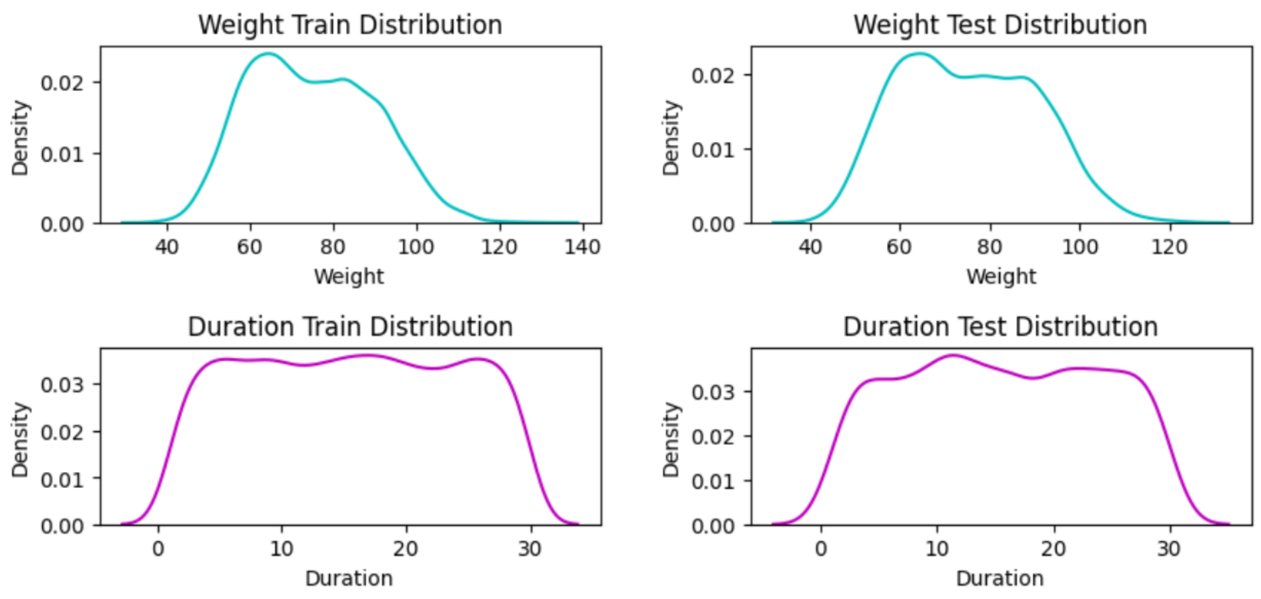
	User_ID	Age	Height	Weight	Duration	Heart_Rate	Body_Temp	Calo
count	1.500000e+04	15000.000000	15000.000000	15000.000000	15000.000000	15000.000000	15000.000000	15000.00
mean	1.497736e+07	42.789800	174.465133	74.966867	15.530600	95.518533	40.025453	89.53
std	2.872851e+06	16.980264	14.258114	15.035657	8.319203	9.583328	0.779230	62.45
min	1.000116e+07	20.000000	123.000000	36.000000	1.000000	67.000000	37.100000	1.00
25%	1.247419e+07	28.000000	164.000000	63.000000	8.000000	88.000000	39.600000	35.00
50%	1.499728e+07	39.000000	175.000000	74.000000	16.000000	96.000000	40.200000	79.00
75%	1.744928e+07	56.000000	185.000000	87.000000	23.000000	103.000000	40.600000	138.00
max	1.999965e+07	79.000000	222.000000	132.000000	30.000000	128.000000	41.500000	314.00

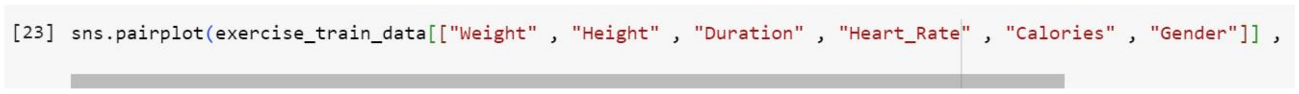
- As we can see, the table above shows each column or feature's Descriptive Statistics(for example, central tendency).
- For example for the Age column.%25 of the data lie between **20** and **28**, another %25 lie between **28** and **39**, and so on. The box plot shows the exact concept that I just mentioned.
- The outliers are shown with dots in box plots, which we will discuss about them in the next section.

Dataset's Distribution



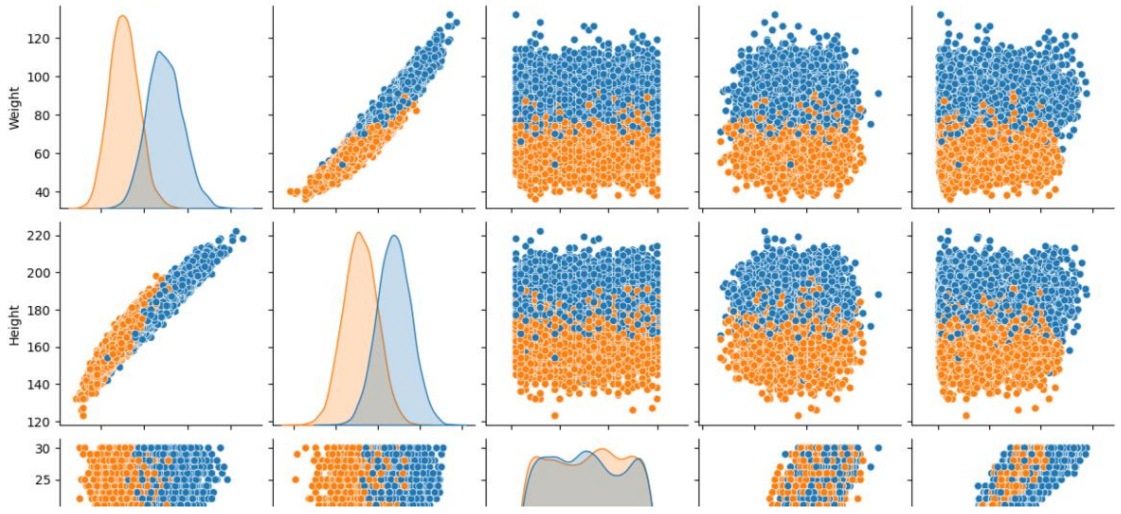
```
[22]
     c = ['b' , 'g' , 'r' , 'c' , 'm' , 'y' , 'k' , 'w' , 'b']
     fig1 , axes = plt.subplots(len(exercise_train_data.columns) , 2 , figsize = (10 , 20))
     plt.subplots_adjust(wspace = 0.3 , hspace = 0.7)
     axes = axes.flatten()            #for using axes indeces with one dimention array instead of two dimension

     for i , column , color in zip(range(0 , len(exercise_train_data.columns) * 2 , 2) , exercise_train_data.colum
       try:
         axes[i].title.set_text(column + " Train Distribution")
         sns.kdeplot(data = exercise_train_data , x = column , ax = axes[i] , color = color)
       except:
         fig1.delaxes(axes[i])
         continue

     for i , column , color in zip(range(1 , len(exercise_train_data.columns) * 2 , 2) , exercise_train_data.colum
       try:
         axes[i].title.set_text(column + " Test Distribution")
         sns.kdeplot(data = exercise_test_data , x = column , ax = axes[i] , color = color)
       except:
         fig1.delaxes(axes[i])
         continue
```





```
[23] sns.pairplot(exercise_train_data[["Weight" , "Height" , "Duration" , "Heart_Rate" , "Calories" , "Gender"]] ,
```

```
<seaborn.axisgrid.PairGrid at 0x7e8427b99420>
```

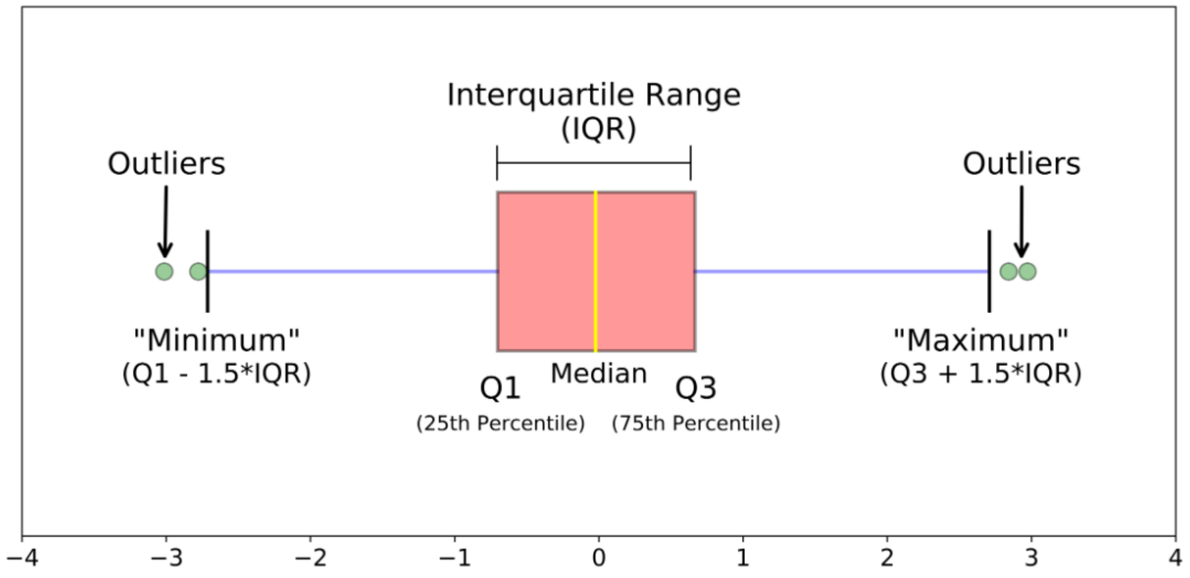


- As we can see from the graphs above, there is not a specific correlation or relationship between most of the features in the dataset. For example, there is not a specific relationship between Duration and Weight or between Duration and Height. This is because exercisers may have different exercise durations no matter their Weight and Height.

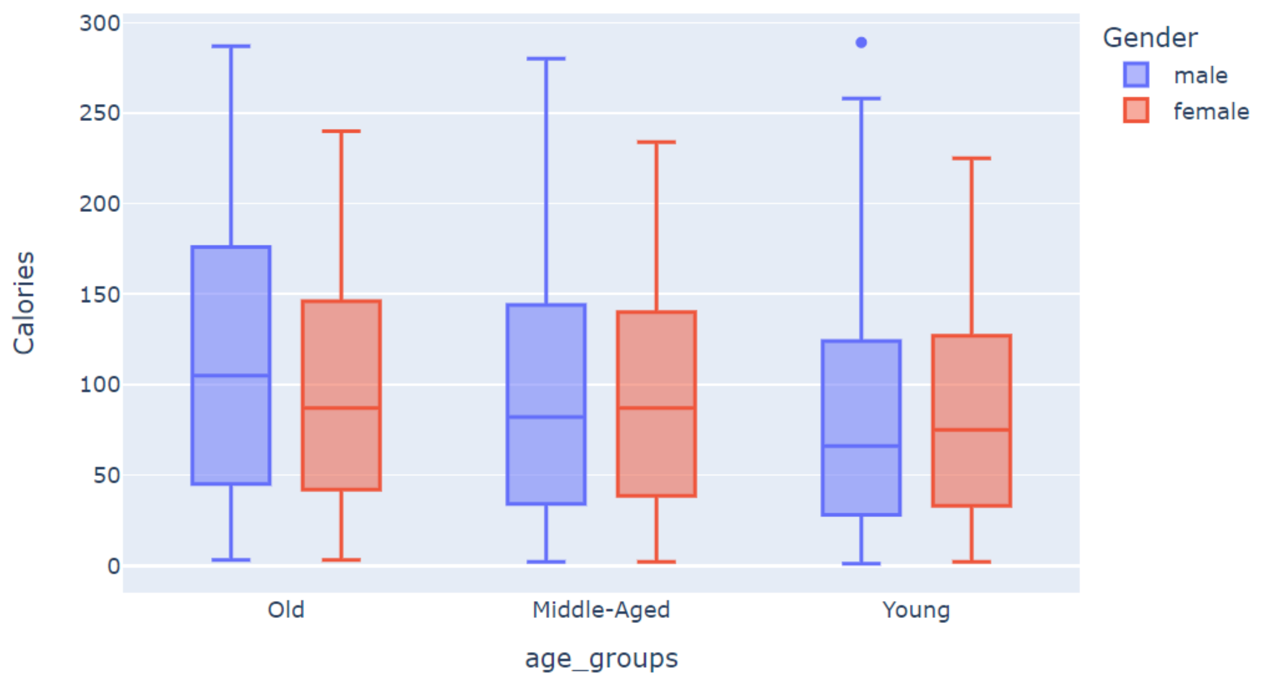
- In some cases, a feature has a low relationship with another feature, like Duration and Heart_Rate. Somehow (with low confidence) we can say that the more time somebody exercises the more 'Heart Rate' per minute he/she will have.

- In some cases, two features have a high relationship(in comparison to last two cases), like Height and Weight.

Lets analyze how many kilocalories each age groups burned.We will do this with `box plot`.Because `box plot` has a intuitive graph that we can extract **Median** , **Interquartile Range** , **Outliers** and etc.Just like the picture below shows.



```
[28] fig = px.box(exercise_train_data , x= "age_groups" , y = "Calories" , color = "Gender")

     fig.update_layout(
         width=700,
         height=450,
     )

     fig.show()
```



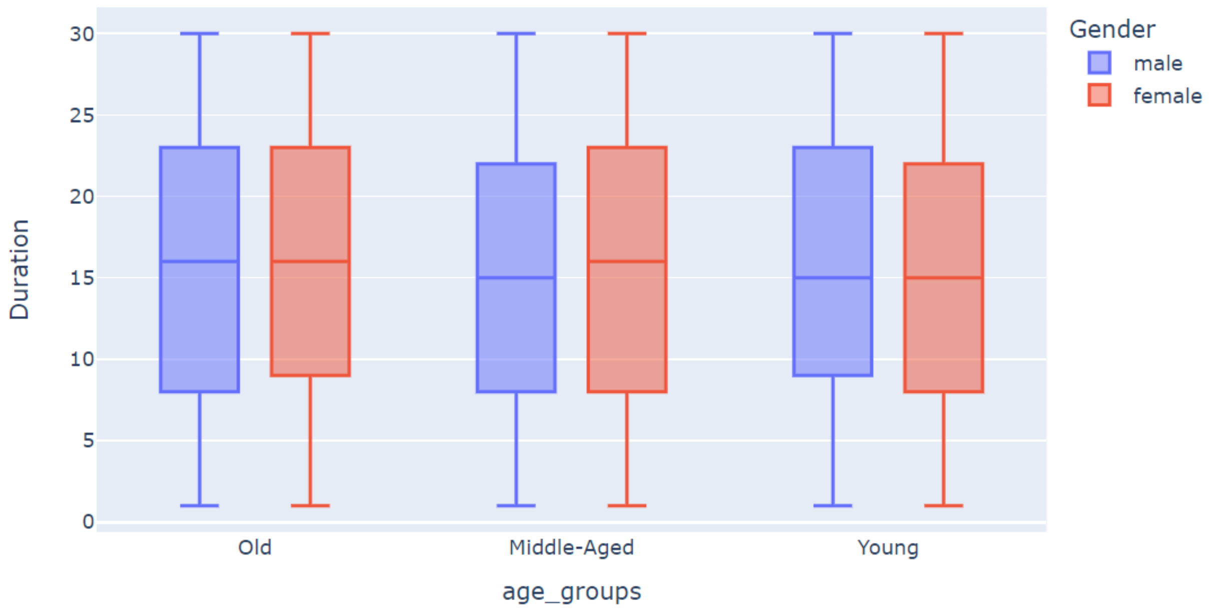
- As we can observe, old individuals have burned more kilocalories in comparison of the two other age groups. And the young persons are the least in burning kilocalories which is a surprise!

- Another interesting thing is, females in all age ranges performed very similar. In other words, they burned the same amount of kilocalories on average. But for males, the older group outperformed and the youth had the weakest performance.

- Also there is an outlier for young group which is shown by a point. This point has a value which is greater than the third quartile value(Q3) plus 1.5 times of interquartile range magnitude.

  Outlier > Q3+1.5*IQR
  OR
  Outlier < Q1 - 1.5 * IQR

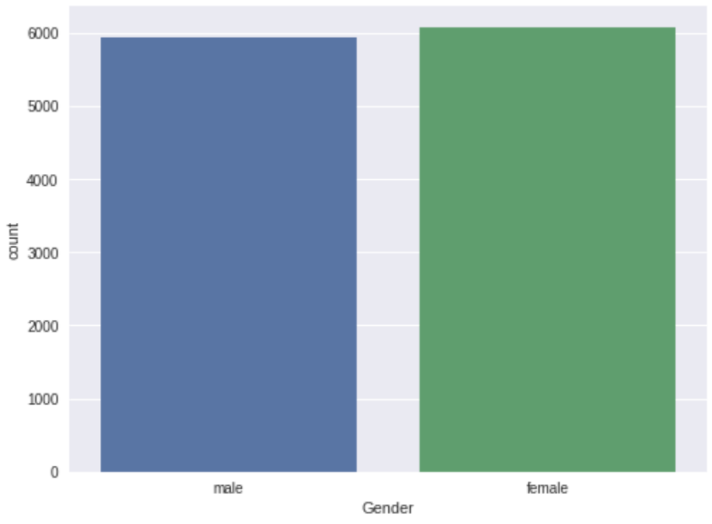
```
[29] fig = px.box(exercise_train_data , x= "age_groups" , y = "Duration" , color = "Gender")

     fig.update_layout(
         width=750,
         height=450,
     )

     fig.show()
```



- As we can see, the exercise duration of each group is pretty identical. Every group has the same interquartile range, median, and so on.
- In addition, the duration is very similar for males and females in old and middle-aged groups. But in youth, males outperformed.
- Another tip is, the median exercise duration of this dataset is about 15 minutes.

```
plt.rcParams["figure.figsize"] = 8 , 6
sns.countplot(data = exercise_train_data , x = "Gender")
```

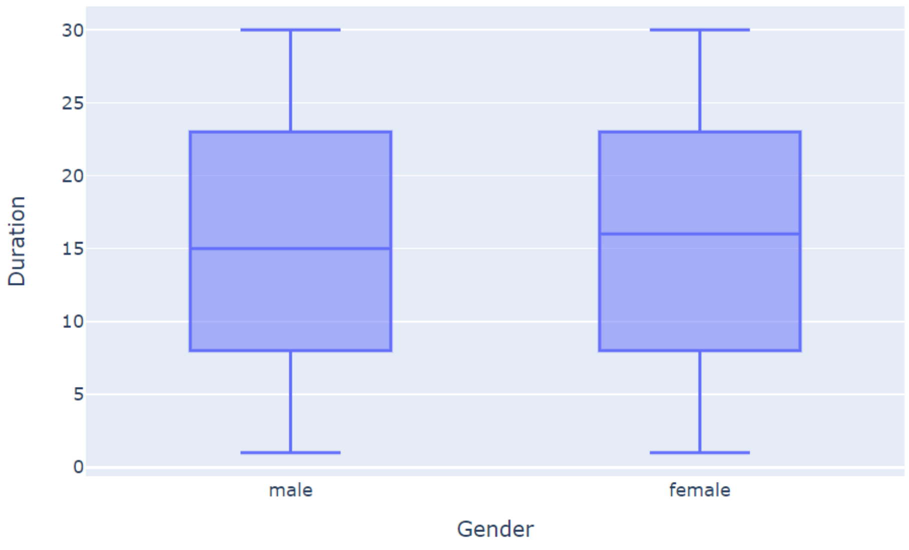
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f00d3b6aef0>
```



- As we can see, the number of females is slightly higher than men but this distinction is not significant. We can say, in general, they are equal.

In this section, let's compare the exercise duration between males and females.

```
[36] fig = px.box(exercise_train_data , x= "Gender" , y = "Duration")

     fig.update_layout(
         width=700,
         height=450,
     )

     fig.show()
```



- The BMI(Body Mass Index) formula:

$$BMI = \frac{Weight(kg)}{Height(m)^2}$$

OR

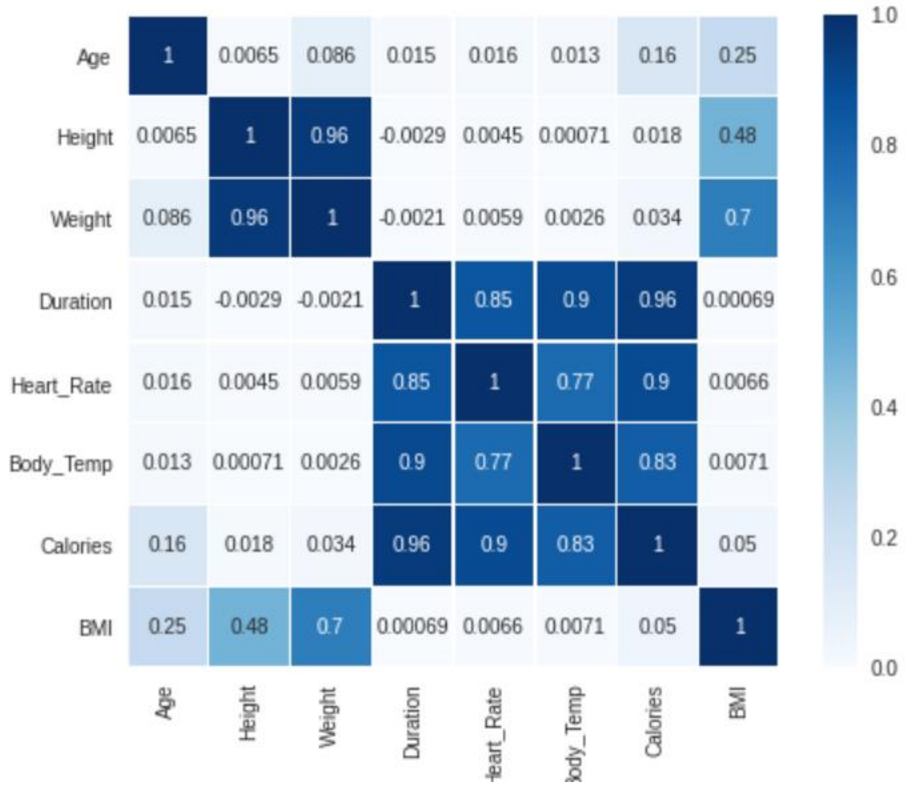
$$BMI = \frac{Weight(Ib)}{Height(in)^2}$$

- The first formula will be used because the units for `Weight` and `Height` of this dataset is `kg` and `meter` in respect.

Correlation:

```
plt.rcParams["figure.figsize"] = 8 , 6
corr = exercise_train_data.corr()
sns.heatmap(corr , annot = True , square = True , linewidth = .5 , vmin = 0 , vmax = 1 , cmap = 'Blues')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f00d38f75c0>
```



- This heatmap shows the correlation of both features in each cell. As we can see, many features have a high correlation with other features. One thing that has to be mentioned is that we have to drop useless features as many as possible. Because when we have many features the dimension of feature space will be very large and when our model runs on these features it will be very slow. Because of that, we have to drop some features.
- If two or more features have a high correlation with each other, we have to save one of them and drop the rest. In This way, we can improve the model's efficiency.

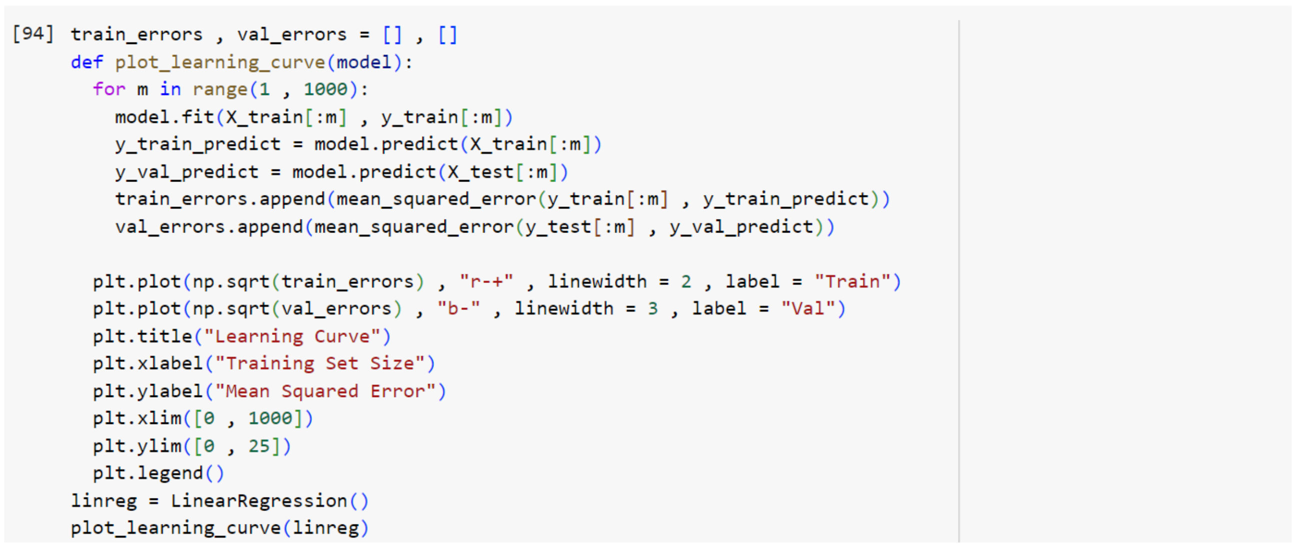
- According to the heatmap, Weight and Height have a high correlation but we combined them and put them into the BMI column. So we can drop the Weight and Height columns and save BMI.

Learning Curve:

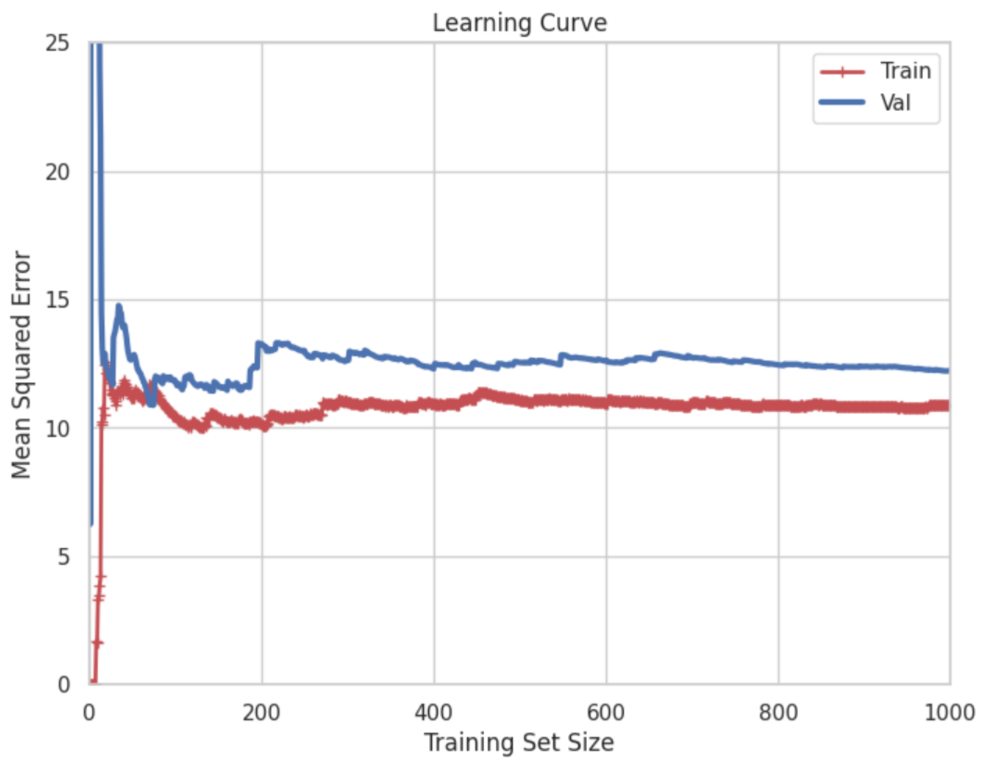
The Learning Curve is a plot of the model's performance on the training set and the validation set as a function of the training set size(or the training iteration). One of the concepts that we get is the appropriate number of examples in training set size. If we take a look at the plot down below we will see that both training set size and validation set reached a plateau at a certain training set size(for example 800 training set size). It means that, with only 800 training set size we will get similar results with our model in comparison to 1000, 2500, or 5000 training set size. In other word increasing the training set size more than 800 will not improve the model's performance significantly.

- As we can see , both curves plateaued at 800 in training set size axis. So it means if we increase the training set size more than 800 examples, model's performance will not be improved significantly. So we can reduce the training set size up to 800 examples without decreasing the model's performance.

- Another reason that the training set and validation set could not reach a lower MSE with more examples is that the dataset do not have enough informative features that our learning algorithm can leverage to build more performant model.



```
[94] train_errors , val_errors = [] , []
     def plot_learning_curve(model):
       for m in range(1 , 1000):
         model.fit(X_train[:m] , y_train[:m])
         y_train_predict = model.predict(X_train[:m])
         y_val_predict = model.predict(X_test[:m])
         train_errors.append(mean_squared_error(y_train[:m] , y_train_predict))
         val_errors.append(mean_squared_error(y_test[:m] , y_val_predict))

       plt.plot(np.sqrt(train_errors) , "r-+" , linewidth = 2 , label = "Train")
       plt.plot(np.sqrt(val_errors) , "b-" , linewidth = 3 , label = "Val")
       plt.title("Learning Curve")
       plt.xlabel("Training Set Size")
       plt.ylabel("Mean Squared Error")
       plt.xlim([0 , 1000])
       plt.ylim([0 , 25])
       plt.legend()
     linreg = LinearRegression()
     plot_learning_curve(linreg)
```



Concatenate Categorical and Numerical

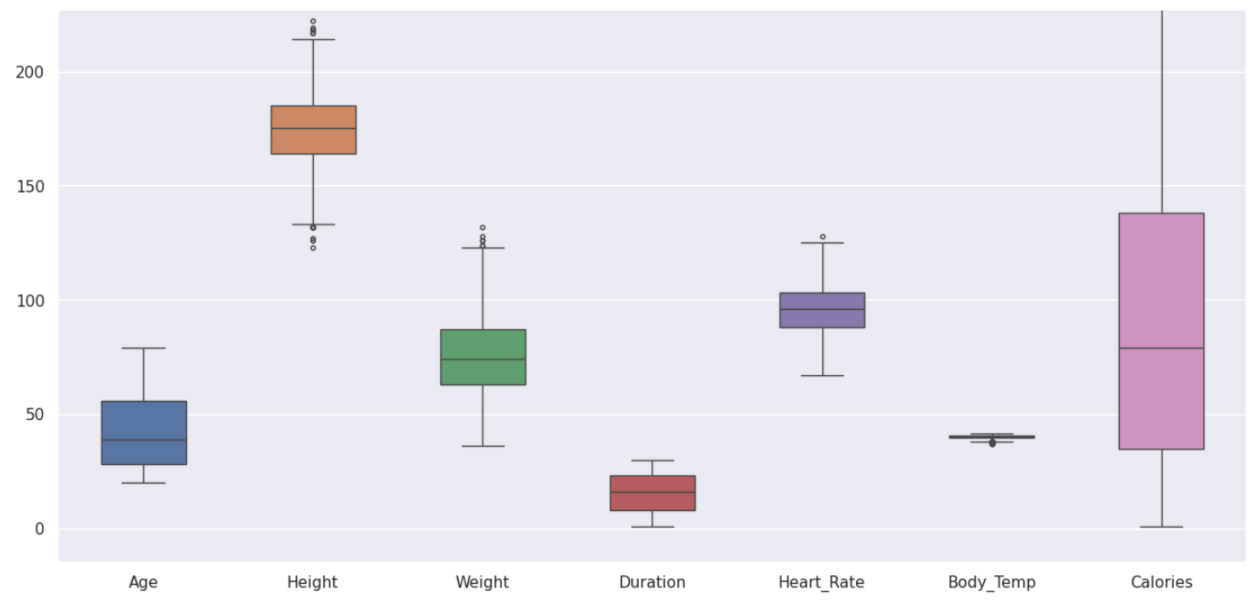
```
[50] data = pd.concat([categorical,Numerical],axis=1)
```

```
[51] data.head()
```

	Gender	Age	Height	Weight	Duration	Heart_Rate	Body_Temp	Calories
0	male	68	190.0	94.0	29.0	105.0	40.8	231.0
1	female	20	166.0	60.0	14.0	94.0	40.3	66.0
2	male	69	179.0	79.0	5.0	88.0	38.7	26.0
3	female	34	179.0	71.0	13.0	100.0	40.5	71.0
4	female	27	154.0	58.0	10.0	81.0	39.8	35.0

Next steps: Generate code with `data` View recommended plots

```
[52] fig,ax = plt.subplots(figsize = (15,10))
     sns.boxplot(data=data,width = 0.5,fliersize = 3,ax=ax)
```



Linear Regression

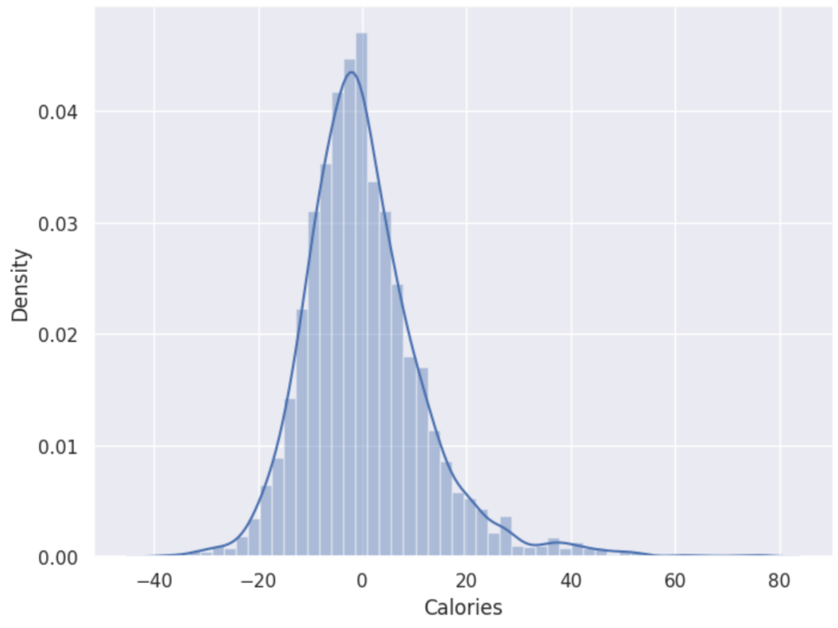
```
[61] linreg = LinearRegression()
     linreg.fit(X_train , y_train)
     linreg_prediction = linreg.predict(X_test)
```

```
[62] print("Linear Regression Mean Absolute Error(MAE) : " , round(metrics.mean_absolute_error(y_test , linreg_prediction)
     print("Linear Regression Mean Squared Error(MSE) : " , round(metrics.mean_squared_error(y_test , linreg_prediction) ,
     print("Linear Regression Root Mean Squared Error(RMSE) : " , round(np.sqrt(metrics.mean_squared_error(y_test , linreg
```

```
Linear Regression Mean Absolute Error(MAE) :  8.48
Linear Regression Mean Squared Error(MSE) :  138.12
Linear Regression Root Mean Squared Error(RMSE) :  11.75
```

```
[63] predict(LinearRegression(), X_train, X_test, y_train, y_test)
```

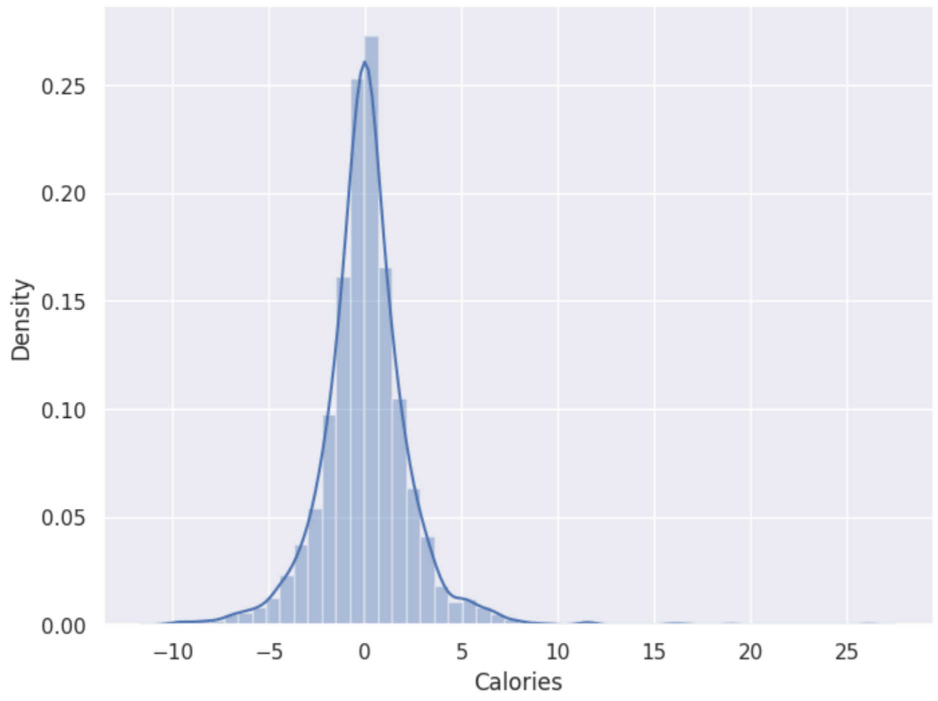
```
Score: 0.967592555473578
Predictions are:
 [198.81182363  80.43555305 194.40940033 ...  22.14745631 118.63504926
 -11.98134672]

R2 Score: 0.9655977245826504
MAE: 8.479071745987955
MSE: 138.12408611460899
RMSE: 11.752620393538157
```



XGB Regressor

```
[64] predict(XGBRegressor(), X_train, X_test, y_train, y_test)
```

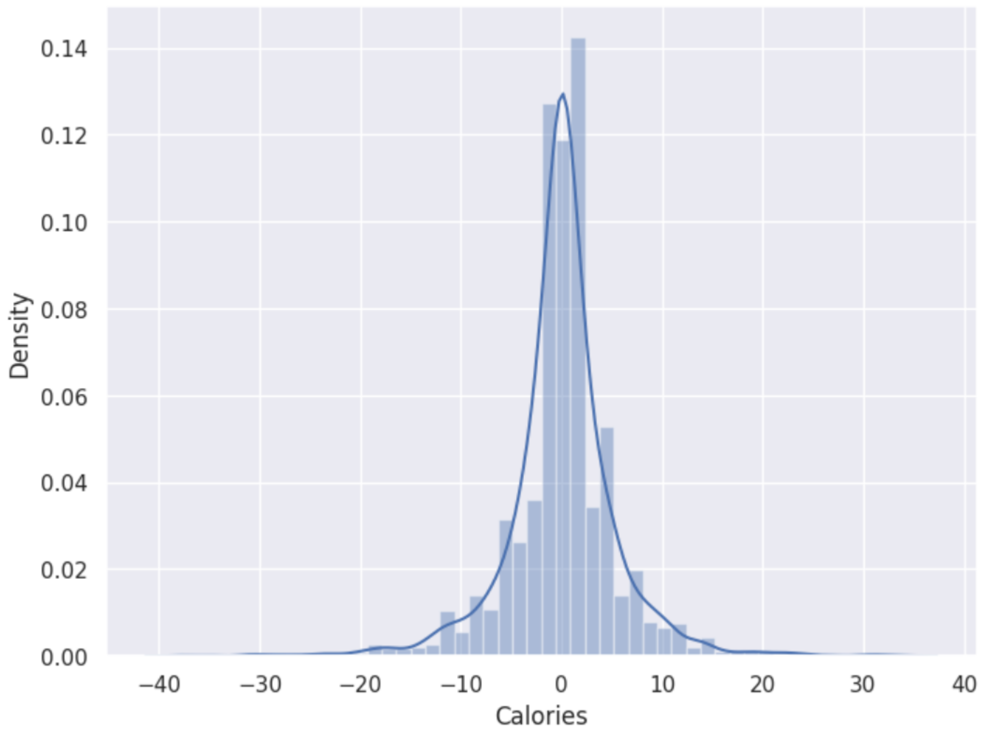
```
Score: 0.9995380557081355
Predictions are:
 [197.06581    70.867226 196.99498   ...  29.043041 104.09284   14.61472 ]

R2 Score: 0.9986863132331905
MAE: 1.5521575984954834
MSE: 5.2744122853837005
RMSE: 2.2966088664340956
```



Decision Tree Regression

```
predict(DecisionTreeRegressor(), X_train, X_test, y_train, y_test)
```

```
Score: 1.0
Predictions are:
 [194.  75. 206. ...  30. 109.  15.]

R2 Score: 0.9924265925079325
MAE: 3.5063333333333335
MSE: 30.407
RMSE: 5.514254256016855
```



RandomForest Regression

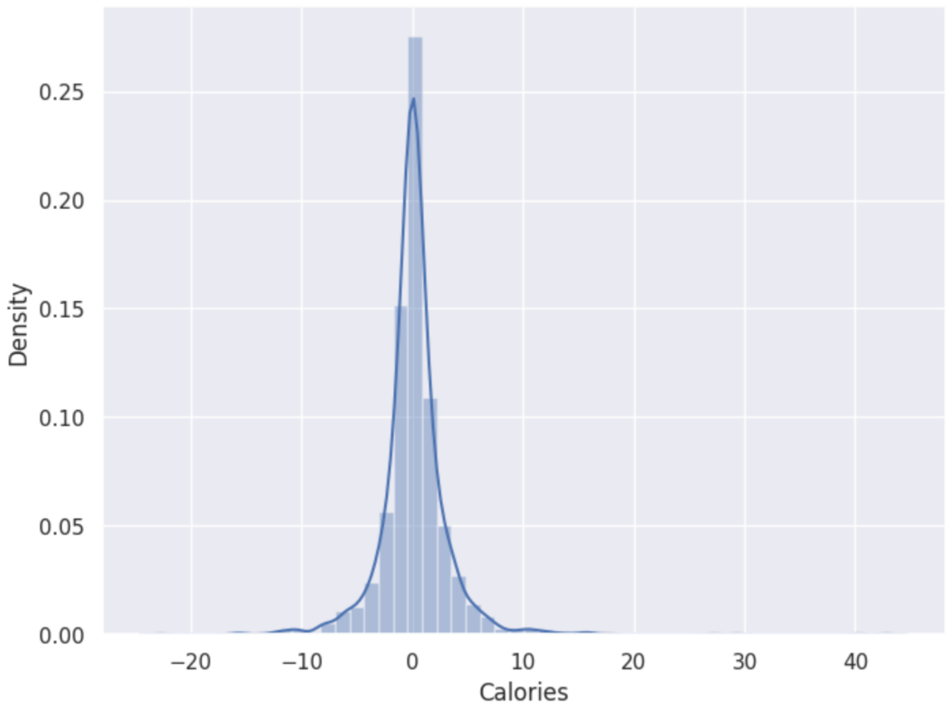
```
[66] random_reg = RandomForestRegressor(n_estimators = 1000 , max_features = 3 , max_depth = 6)
     random_reg.fit(X_train , y_train)
     random_reg_prediction = random_reg.predict(X_test)
```

```
[67] print("RandomForest Mean Absolute Error(MAE) : " , round(metrics.mean_absolute_error(y_test , random_reg_prediction)
     print("RandomForest Mean Squared Error(MSE) : " , round(metrics.mean_squared_error(y_test , random_reg_prediction) ,
     print("RandomForest Root Mean Squared Error(RMSE) : " , round(np.sqrt(metrics.mean_squared_error(y_test , random_reg_
```

```
RandomForest Mean Absolute Error(MAE) :  5.44
RandomForest Mean Squared Error(MSE) :  71.05
RandomForest Root Mean Squared Error(RMSE) :  8.43
```

```
[68] predict(RandomForestRegressor(), X_train, X_test, y_train, y_test)
```

```
Score: 0.9996767266090387
Predictions are:
 [197.04  66.55 196.26 ...  27.76 111.58  14.27]

R2 Score: 0.9975980724895278
MAE: 1.8348866666666666
MSE: 9.643665666666667
RMSE: 3.1054251990132795
```



Dataset 3: Fitness Trackers Market Analysis

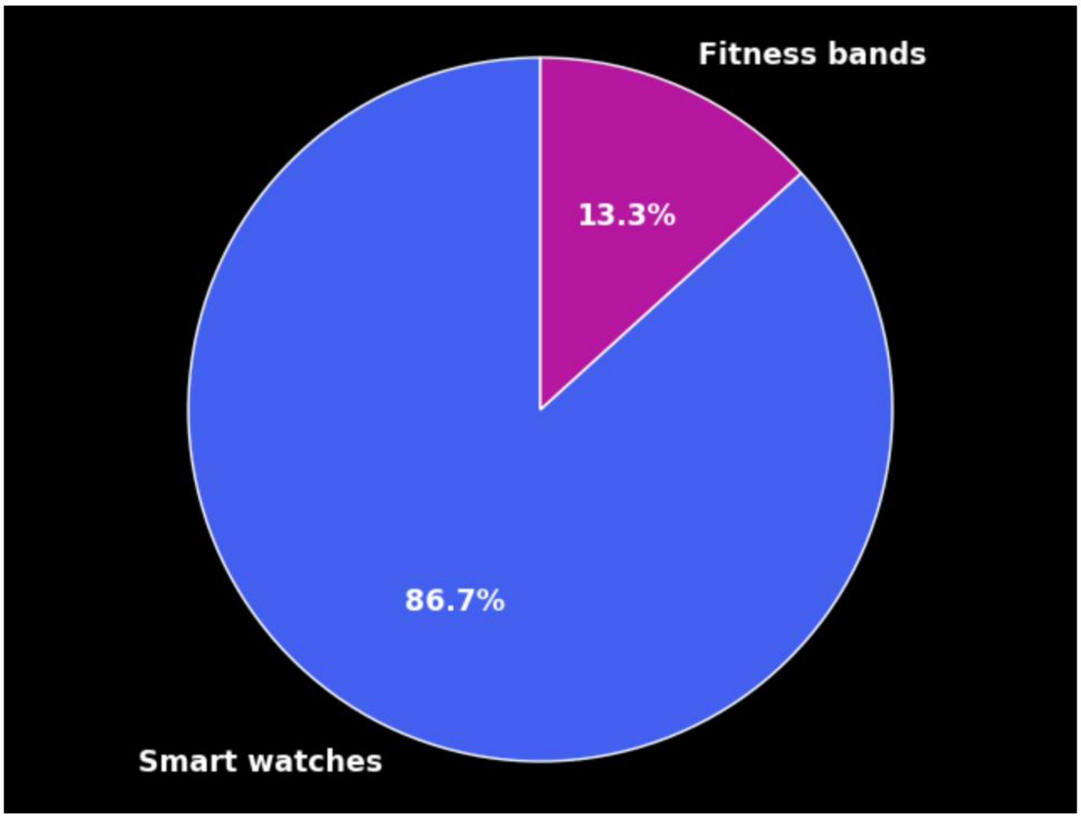
```
[72] colors = ["#F72585","#B5179E","#7209B7","#560BAD","#480CA8","#3A0CA3",\
     "#3F37C9","#4361EE","#4895EF","#4CC9F0","#a5a58d"]
     palette = sns.color_palette(palette = colors)

     sns.palplot(palette, size =1)
     plt.show()
```

Demand for fitness trackers

```
[77] df3['Brand Name'].groupby(df3['Device Type']).count().sort_values(ascending=False)
```

```
Device Type
Smartwatch     490
FitnessBand     75
Name: Brand Name, dtype: int64
```

```
[78] labels = 'Smart watches', 'Fitness bands'
     sizes = [490,75]
     fig1, ax1 = plt.subplots()
     fig1.set_facecolor('black')
     ax1.pie(sizes, labels=labels, colors=["#4361EE",'#b5179e'],autopct='%1.1f%%', startangle=90,textprops={'color':'w','w
     ax1.axis('equal')
     plt.show()
```



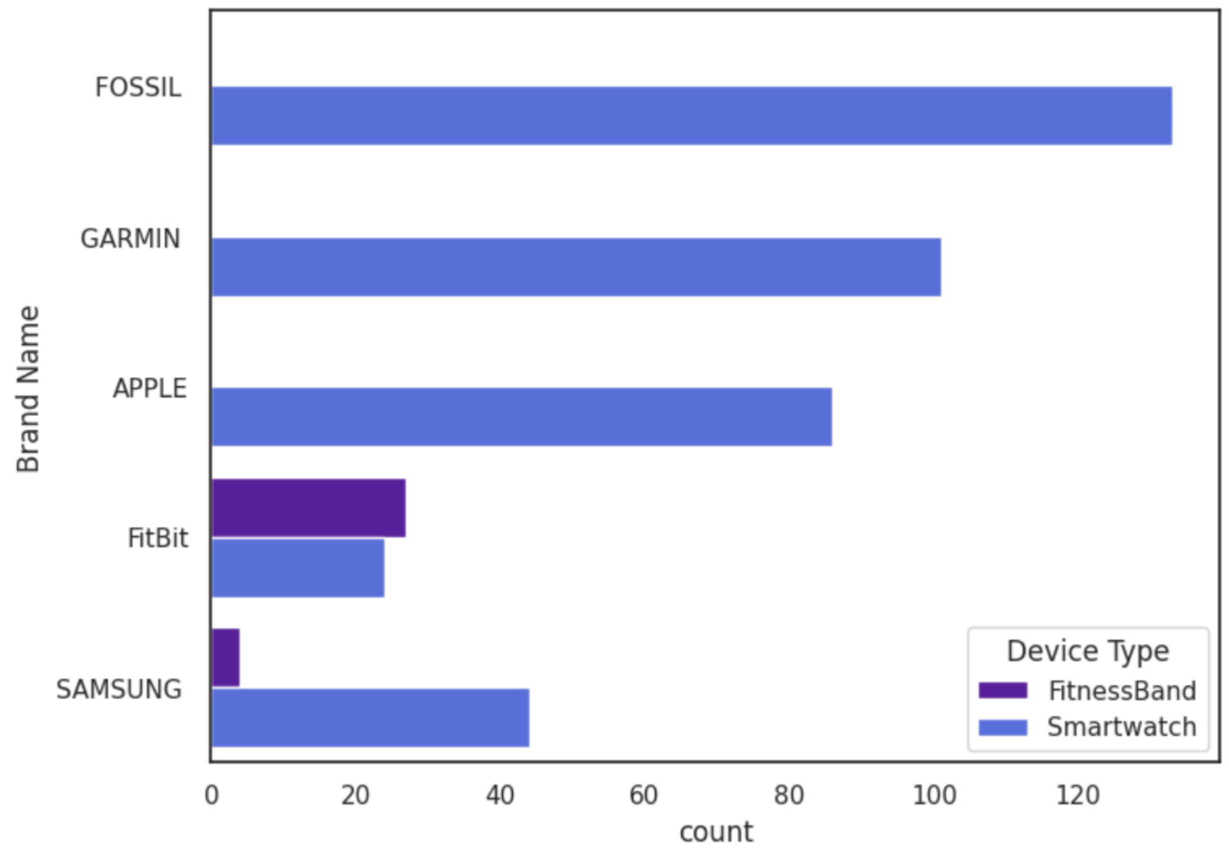
Number of Players in the market:

```
[79] df3['Brand Name'].groupby(df3['Brand Name']).count().sort_values(ascending=False).iloc[:5]
```

```
Brand Name
FOSSIL      133
GARMIN      101
APPLE        86
FitBit       51
SAMSUNG      48
Name: Brand Name, dtype: int64
```

533 smartwatches and 77 fitness bands from 25 different brands indicate a good demand for Fitness Trackers in the current scenario.

Which are the top 5 brands for fitness bands and smart watches?



The outliers are shown with dots in box plots, which we will discuss about them in the next section.

Fossil, Garmin, Apple, FitBit, and Samsung are the top 5 brands offering the most fitness trackers. Fossil, Garmin and Apple seem to offer only smartwatches while FitBit and Samsung offer both fitness bands and smartwatches.

Which brand has the highest number of products?

```
[81] df3['Device Type'].groupby(df3['Brand Name']).count().sort_values(ascending=False).iloc[:1]
```

```
Brand Name
FOSSIL     133
Name: Device Type, dtype: int64
```

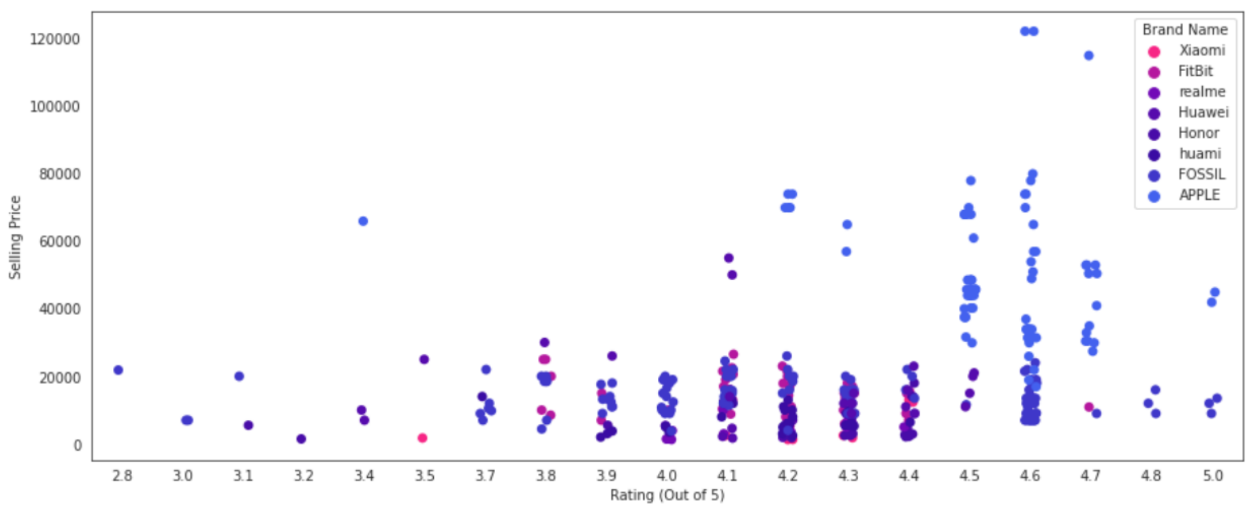
Fossil has the most number of products i.e. Smart watches.

Are fitness trackers with higher ratings more expensive?

```
[82] round(df3.groupby('Brand Name')['Rating (Out of 5)'].mean().sort_values(ascending=False).iloc[:10],1)
```

```
Brand Name
APPLE       4.5
OnePlus     4.3
FOSSIL      4.2
SAMSUNG     4.2
Honor       4.2
FitBit      4.2
Xiaomi      4.2
Huawei      4.2
huami       4.2
realme      4.1
Name: Rating (Out of 5), dtype: float64
```

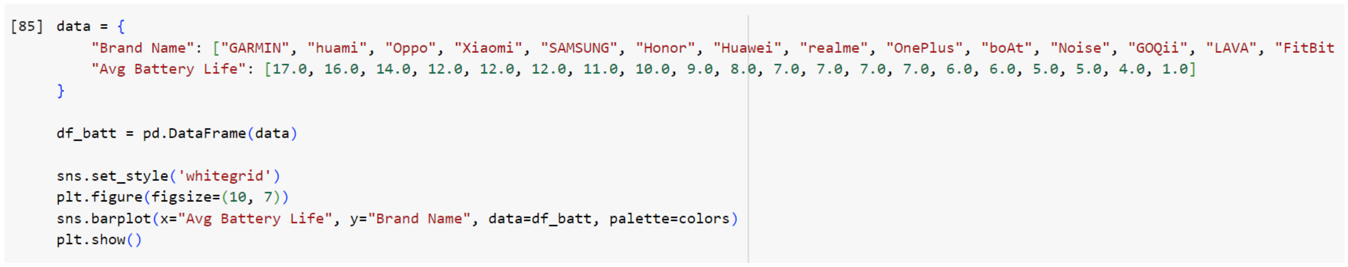
```
[83] list_of_brands = ["APPLE", "OnePlus", "FOSSIL", "SAMSUNG", "Honor", "FitBit", "Xiaomi", "Huawei", "huami", "realme"]

     df_f = df3[df3["Brand Name"].isin(list_of_brands)]

     fig, ax = plt.subplots(figsize=(15, 6))
     sns.stripplot(x="Rating (Out of 5)", y="Selling Price", data=df_f, hue="Brand Name", palette=colors, size=7, marker="
```



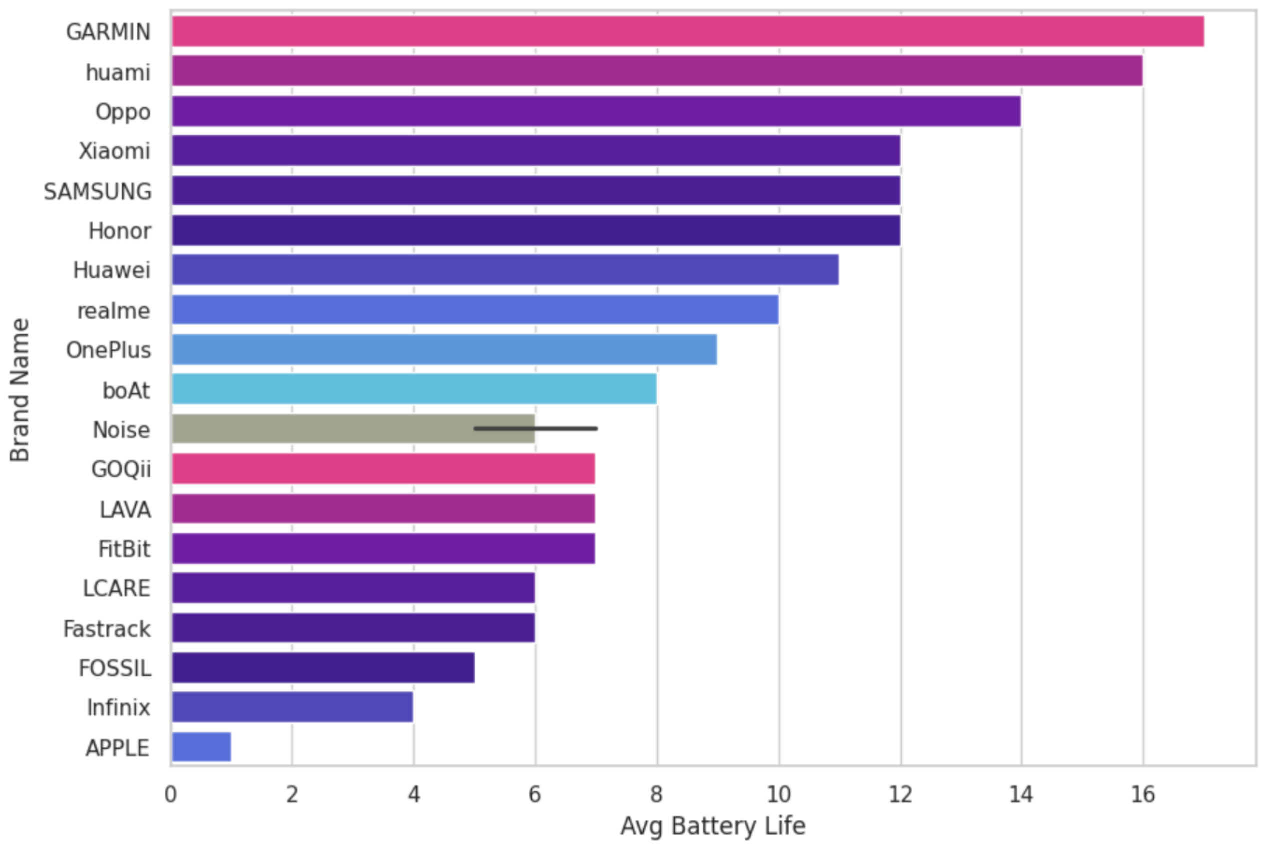
Do expensive fitness trackers have a better battery performance?

```
[84] round(df3.groupby('Brand Name')['Average Battery Life (in days)'].mean(),0).sort_values(ascending=False)
```

```
Brand Name
GARMIN      17.0
huami       16.0
Oppo        14.0
Honor       12.0
Xiaomi      12.0
SAMSUNG     12.0
Huawei      11.0
realme      10.0
OnePlus      9.0
boAt         8.0
FitBit       7.0
GOQii        7.0
LAVA         7.0
Noise        7.0
LCARE        6.0
Fastrack     6.0
Noise        5.0
FOSSIL       5.0
Infinix      4.0
APPLE        1.0
Name: Average Battery Life (in days), dtype: float64
```



```
[85] data = {
         "Brand Name": ["GARMIN", "huami", "Oppo", "Xiaomi", "SAMSUNG", "Honor", "Huawei", "realme", "OnePlus", "boAt", "Noise", "GOQii", "LAVA", "FitBit
         "Avg Battery Life": [17.0, 16.0, 14.0, 12.0, 12.0, 12.0, 11.0, 10.0, 9.0, 8.0, 7.0, 7.0, 7.0, 7.0, 6.0, 6.0, 5.0, 5.0, 4.0, 1.0]
     }

     df_batt = pd.DataFrame(data)

     sns.set_style('whitegrid')
     plt.figure(figsize=(10, 7))
     sns.barplot(x="Avg Battery Life", y="Brand Name", data=df_batt, palette=colors)
     plt.show()
```

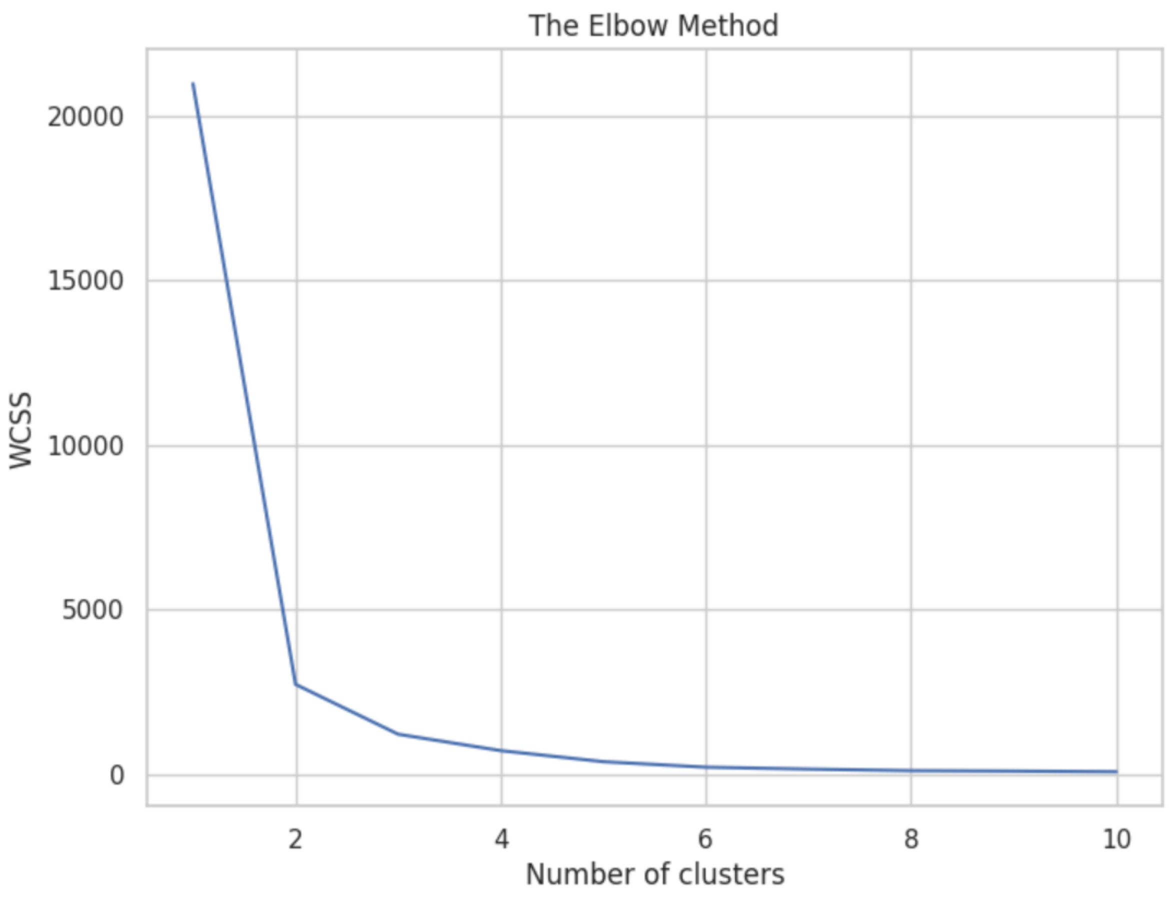


As an expensive brand, Garmin seems to provide the best battery life in days. Samsung on the other hand gives around 12 days of battery life, a week less than Garmin. Most Apple products provide only a 1-2 day battery life. Similarly, Fossil, although expensive, offers a very low battery life of less than a week.

Both Garmin and Fossil brands are expensive but do not have comparable battery life. Hence, expensive does not always equal to better battery life for the fitness tracker.

Elbow Method

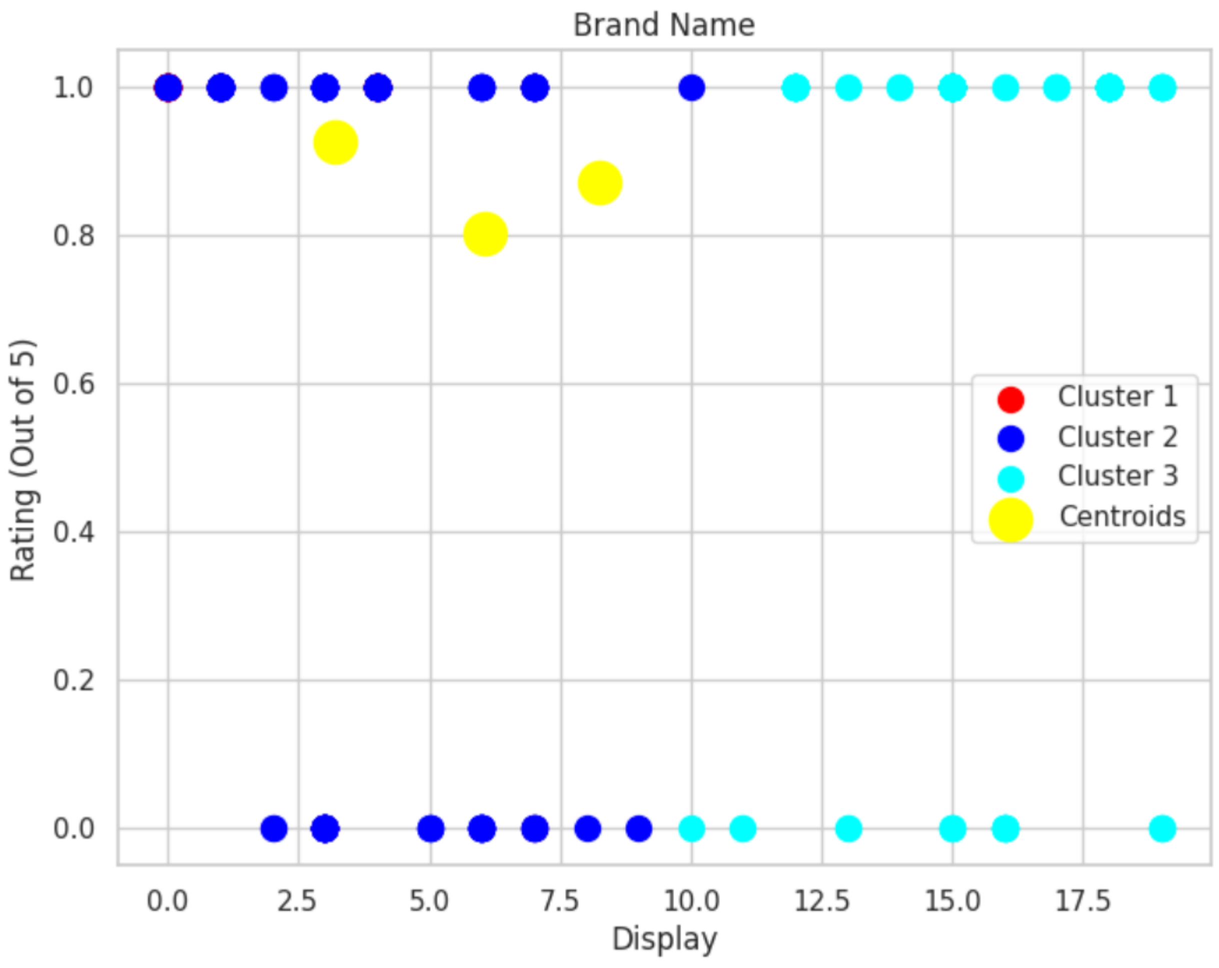
```
[88] X = df3.iloc[:, [0,1]].values

     # Using the elbow method to find the optimal number of clusters
     from sklearn.cluster import KMeans
     wcss = []
     for i in range(1, 11):
         kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
         kmeans.fit(X)
         wcss.append(kmeans.inertia_)
     plt.plot(range(1, 11), wcss)
     plt.title('The Elbow Method')
     plt.xlabel('Number of clusters')
     plt.ylabel('WCSS')
     plt.show()
```



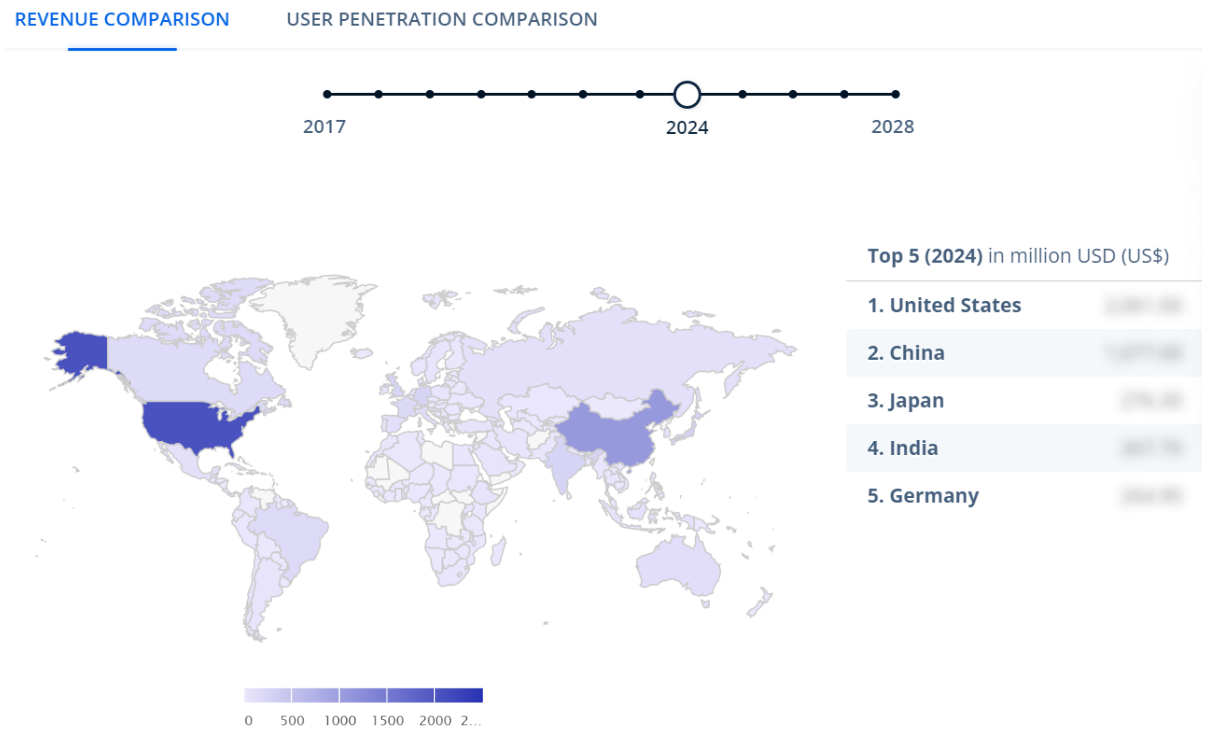
K-Means Clustering:

```
[89] kmeans = KMeans(n_clusters = 3, init = 'k-means++', random_state = 20)
     y_kmeans = kmeans.fit_predict(df3[['Brand Name','Display','Rating (Out of 5)']])
     kmeans.fit(df3)
     df3['Cluster'] = kmeans.labels_
```

```
[90] plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0, 1], s = 100, c = 'red', label = 'Cluster 1')
     plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1, 1], s = 100, c = 'blue', label = 'Cluster 2')
     plt.scatter(X[y_kmeans == 2, 0], X[y_kmeans == 2, 1], s = 100, c = 'cyan', label = 'Cluster 3')
     plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 300, c = 'yellow', label = 'Centroids')
     plt.title('Brand Name')
     plt.xlabel('Display')
     plt.ylabel('Rating (Out of 5)')
     plt.legend()
     plt.show()
```

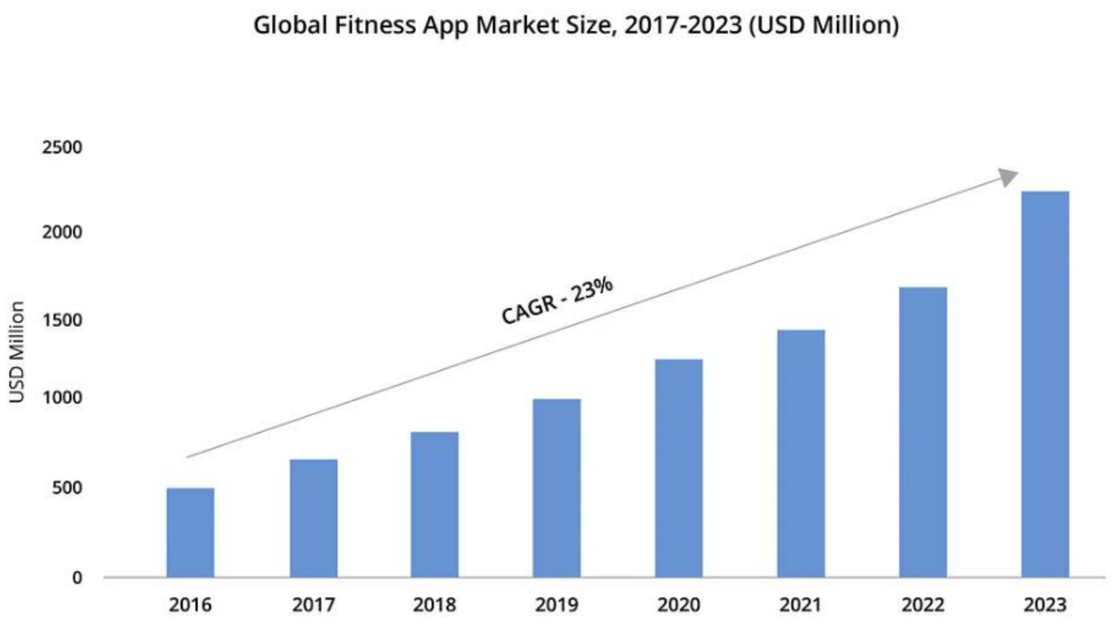


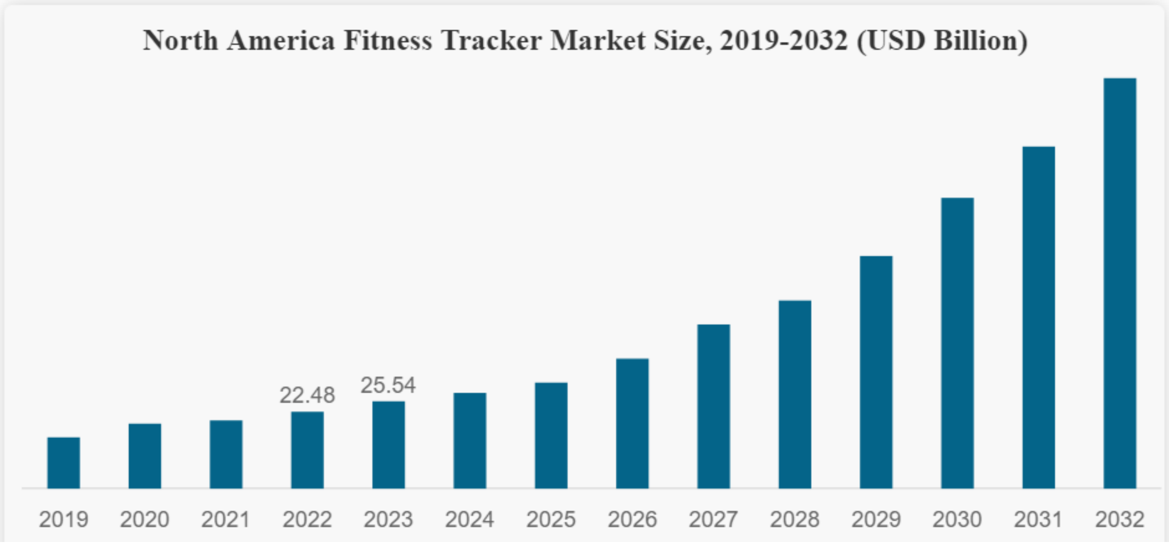
Financial Equation:



Top 5 (2024) in million USD (US$)

1. United States	[illegible]
2. China	[illegible]
3. Japan	[illegible]
4. India	[illegible]
5. Germany	[illegible]





Let:

SR = Subscription Revenue

IAP = In-App Purchases

AR = Advertising Revenue

FR = Affiliate Revenue

EF = Event Fees

DC = Development Costs

MC = Marketing Costs

OC = Operational Costs

CCC = Content Creation Costs

SW = Salaries and Wages

MNC = Maintenance Costs

MSC = Miscellaneous Costs

Revenue and Costs Equations:

R=SR+IAP+AR+FR+EF

C=DC+MC+OC+CCC+SW+MNC+MSC

P=(SR+IAP+AR+FR+EF)−(DC+MC+OC+CCC+SW+MNC+MSC)

In the form of y=mx+b, where:

y is the profit **P,**

x is the subscription revenue SR

m is the slope, and

b is the y-intercept.

For our equation:

Assumption:

y=P

x=SR

m=1 (since profit changes linearly with subscription revenue)

b=−43,000

P=1·SR−43,000

P=SR−43,000

This equation indicates that for every dollar increase in subscription revenue, the profit increases by one dollar, starting from a base loss of $43,000 when there is no subscription revenue.

Conclusion:

All things considered, the calories burnt prediction app project is a noteworthy undertaking in the field of fitness and well-being technology.
Throughout development, we have seen how data science, machine learning, and user-centric design concepts have come together to produce a useful tool for people looking to improve their health and fitness.

The software generates customized calorie burn estimates based on user-supplied data and machine learning algorithms, taking into account each user's distinct attributes and activity levels. Because of this degree of personalization, users are better equipped to plan their exercise regimens, burn calories as efficiently as possible, and monitor their progress over time.
In the future, there is a great deal of potential for the calories burned prediction software to improve people's lives all across the world. The app encourages users to prioritize their health and well-being, adopt a more active lifestyle, and eventually reach their fitness goals by offering practical insights, accountability, and inspiration.

References:

1. Khan, Abdul Wahid, et al. "Factors Affecting Fitness Motivation: An Exploratory Mixed Method Study." IUP Journal of Marketing Management 21.2
2. Tayade, Akshit Rajesh, and Hadi Safari Katesari. "A Statistical Analysis to Develop Machine Learning Models: Prediction of User Diet Type."
3. S. P. Vinoy and B. Joseph, “Calorie Burn Prediction Analysis Using XGBoost Regressor and Linear Regression Algorithms,” in Proceedings of the National Conference on Emerging Computer Applications (NCECA), Kottayam, 2022.
4. Important features for fitness tracking app development (peerbits.com)
5. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5496172
6. https://www.jetpac.com/
7. Tang, J., Abraham, C., Stamp, E., Greaves, C.: How can weight-loss app designers best engage and support users? A qualitative investigation. Br. J. Health. Psychol. 20(1), 151–171 (2015) 27. Clawson, J
8. Simpson, C.C., Mazzeo, S.E.: Calorie counting and fitness tracking technology: associations with eating disorder symptomatology. Eat. Behav. 26, 89–92 (2017)
9. A. Kadam, A. Shrivastava, S. K. Pawar, V. H. Patil, J. Michaelson and A. Singh, "Calories Burned Prediction Using Machine Learning," 2023 6th International Conference on Contemporary Computing and Informatics (IC3I), Gautam Buddha Nagar, India, 2023, pp. 1712-1717, doi: 10.1109/IC3I59117.2023.10397623. keywords: {Heart rate; Machine learning algorithms; Data preprocessing; Predictive models; Prediction algorithms; Data models; Random forests; Machine Learning; Random Forest Regressor; Stream lit; Calories Burned Prediction}.

10. M. Nipas, A. G. Acoba, J. N. Mindoro, M. A. F. Malbog, J. A. B. Susa and J. S. Gulmatico, "Burned Calories Prediction using Supervised Machine Learning: Regression Algorithm," 2022 Second International Conference on Power, Control and Computing Technologies (ICPC2T), Raipur, India, 2022, pp. 1-4, doi: 10.1109/ICPC2T53885.2022.9776710. keywords: {Training; Machine learning algorithms; Computational modelling; Data visualization; Predictive models; Prediction algorithms; Cleaning; calories; regression model; prediction; machine learning}.