**INDEX**

**CHAPTER 1: INTRODUCTION**

1.1 OVERVIEW

1.2 PURPOSE

**CHAPTER 2: LITERATURE SURVEY**

2.1 EXISTING PROBLEM

2.2 PROPOSED SOLUTION

**CHAPTER 3: THEORITICAL ANALYSIS**

3.1 BLOCK DIAGRAM

3.2 HARDWARE/SOFTWARE DESIGNING

**CHAPTER 4: EXPERIMENTAL INVESTIGATIONS**

**CHAPTER 5: FLOWCHART**

**CHAPTER 6: RESULTS**

**CHAPTER 7: ADVANTAGES & DISADVANTAGES**

**CHAPTER 8: APPLICATIONS**

**CHAPTER 9: CONCLUSION**

**CHAPTER 10: FUTURE SCOPE**

**CHAPTER 11: BIBLIOGRAPHY**

**CHAPTER 12: REFERENCES**

12.1 SOURCE CODE

12.2 UI OUTPUT SCREENSHOT

**CHAPTER – 1**

**INTRODUCTION**

**1.1OVERVIEW**

The aim of the study was to investigate relationships between physical fitness and self-reported physical activity in adulthood and to what extent the level of physical fitness and leisure-time physical activity in adulthood can be explained by anthropometric demographic characteristics at the age of 16.A group of 157 men and 121 women was tested at the age of 16 and 34 by means of questionnaires and fitness tests.

**1.2 PURPOSE**

Assessing physical fitness in junior categories can be a useful resource to determine future karate success. Coaches in this sport should pay special attention to the levels of muscular power and agility shown by their athletes, as both fitness dimensions are the ones that could be indicators of future sportive success.

**CHAPTER – 2**

**LITERATURE SURVEY**

**2.1 EXISTING PROBLEM**

The internet hosts some incredibly useful health and fitness information – But it also has some woefully bad advice too. The problem is that it can be hard for people to discern what’s helpful and what’s not. The advisers give according to their research it may not help to know his/ her is fit or not. It may regulate the further component issues.

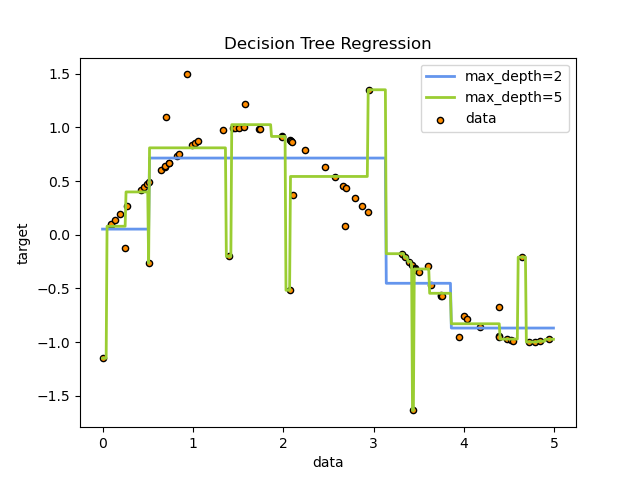
In social media there are many resources or finding like-minded people with similar interests. This is brilliant because no matter what your interests, you can find a community of people who support you and share your passion. But social platforms can also have the unintended side-effect of polarising people. In our model we use decision tree algorithm for predicting the fitness we use different fields like hours of sleep, steps count, weight etc. Which helps to his fit or not.

**2.2 PROPOSED SOLUTION**

**Methodology:**

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

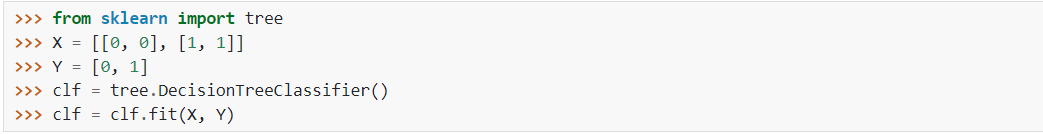
For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.



***Decision tree graph***

DecisionTreeClassifier is a class capable of performing multi-class classification on a dataset.

As with other classifiers, DecisionTreeClassifier takes as input two arrays: an array X, sparse or dense, of shape (n samples, n features) holding the training samples, and an array Y of integer values, shape (n samples,), holding the class labels for the training samples:



After being fitted, the model can then be used to predict the class of samples:



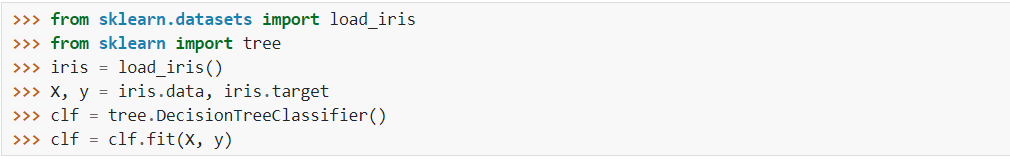
In case that there are multiple classes with the same and highest probability, the classifier will predict the class with the lowest index amongst those classes.

As an alternative to outputting a specific class, the probability of each class can be predicted, which is the fraction of training samples of the class in a leaf:



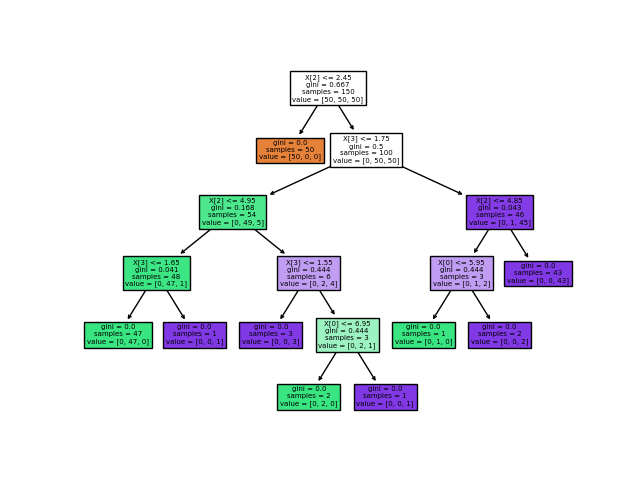
DecisionTreeClassifier is capable of both binary (where the labels are [-1, 1]) classification and multiclass (where the labels are [0, …, K-1]) classification.

Using the Iris dataset, we can construct a tree as follows:



Once trained, you can plot the tree with the plot\_tree function:



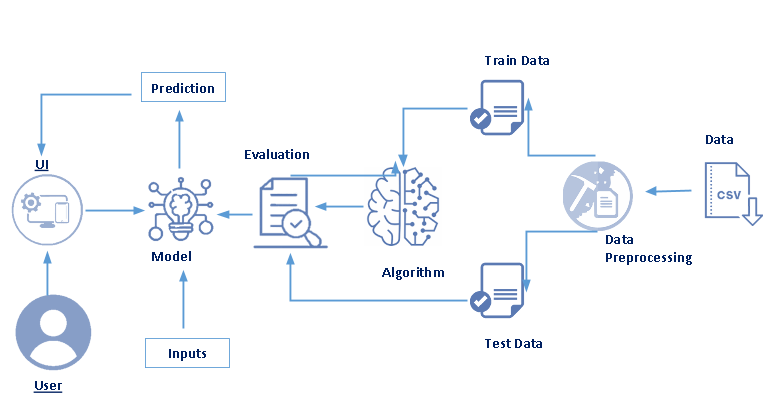


***Output of the tree plotted for above line of code***

**CHAPTER - 3**

**THEORITICAL ANALYSIS**

**3.1 Block diagram**



**3.2 Hardware/Software designing:**

The data collected consists of about 96 records and 7 attributes. To predict the body fitness, we need to build a machine learning model. For that we have used the Jupyter notebook and Flask Application for training the model and deploying it. Building a machine learning includes the following steps.

1. Data Pre processing
2. Feature Extraction
3. Model Training
4. Prediction
5. Deployment to Prediction

**Data Pre processing:**

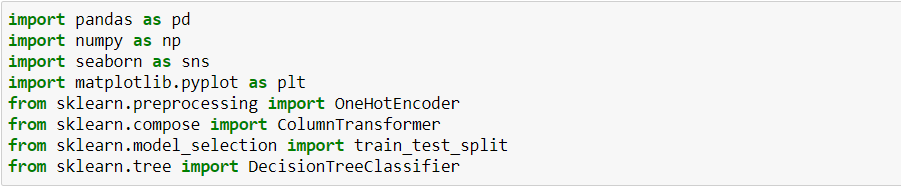
Data pre processing includes of 5 steps. They are:

1. Importing Libraries and Reading the Dataset
2. Check the missing values and considering the Correlational heatmap.
3. Separating the independent and dependent variables.
4. Converting the data into numpy array and perform Encoding on categorical variables.
5. Splitting the dataset for training and testing.

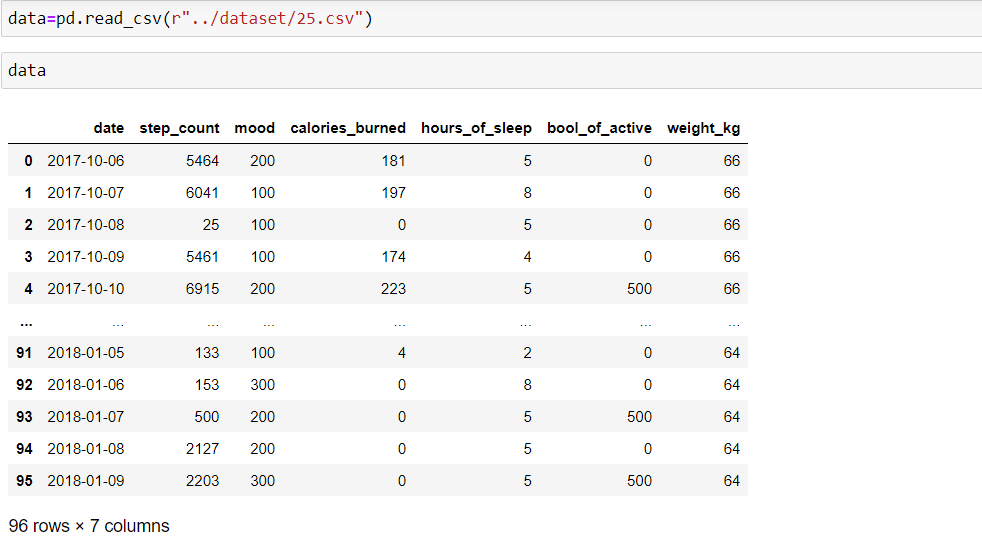
Firstly, we need to import the data to the operating environment. For loading the data sets and to preprocess them we need to import the libraries such as pandas, numpy and matplotlib for visualization.

**Step 1:**

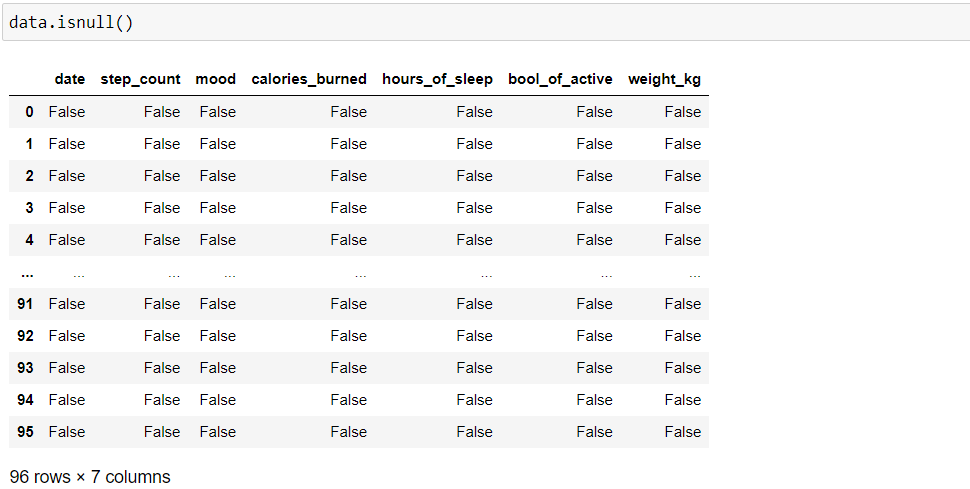
1. Importing the Libraries numpy, pandas, seaborn, matplotlib, sklearn.

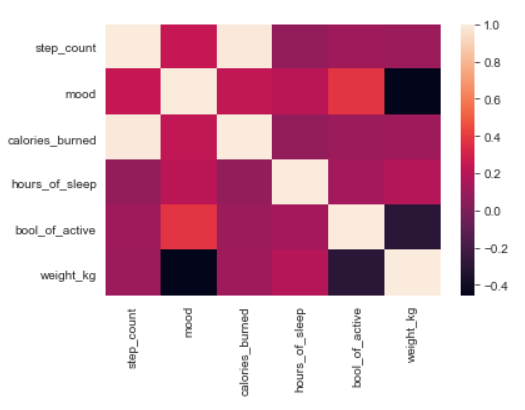


1. Read the dataset and displaying the dataset.

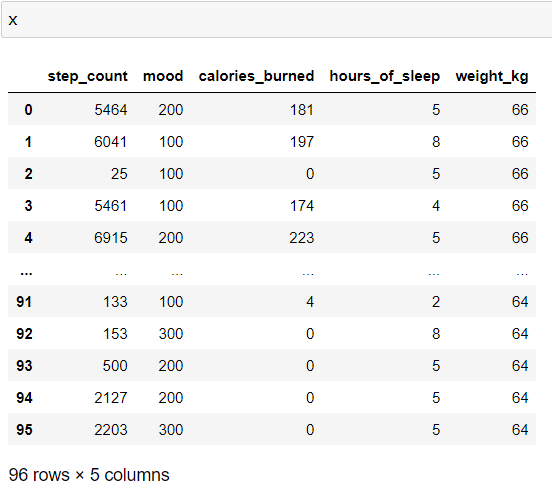


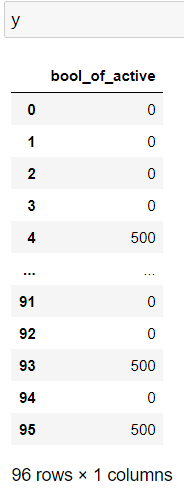
**Step 2:** Check if the dataset consists of any missing values and remove the poorly corelated independent variables by using the correlation heatmap.





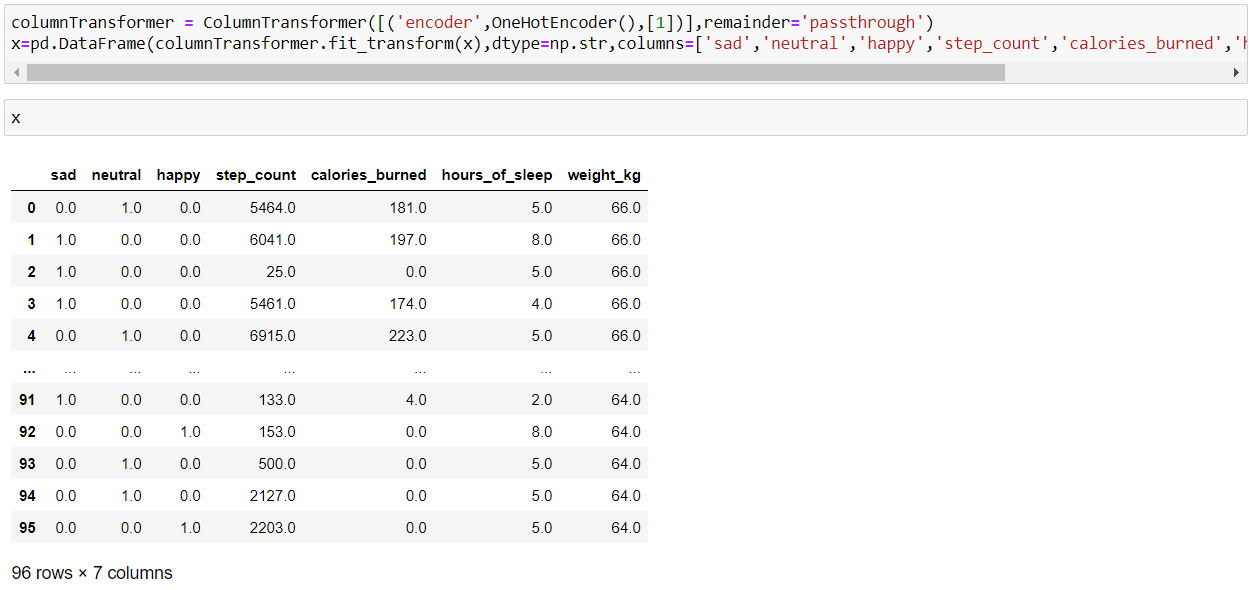
**Step 3:** Now separate the independent and dependent variables.





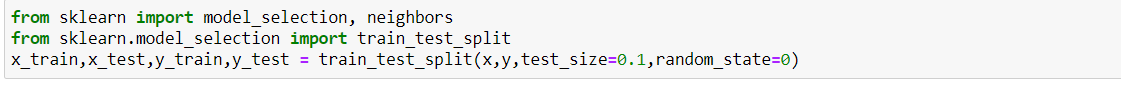
**Step 4:** Encoding the Categorical Variables

We cannot pass the text data to algorithm because these algorithms contain mathematical computations which cannot process strings so these must encode into numerical variables. And we encode using binary representation not directly assigning numbers because assigning numbers may be referred as priority to them. For encoding categorical text variables, we have package sklearn. This can be done after converting the data into numpy arrays. Label encoding and one hot encoding are not possible to perform on this dataset because there are no categorical   
data columns.

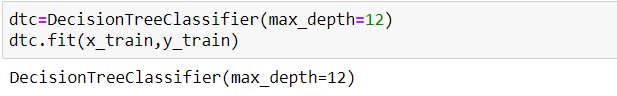


**Step 5:** Splitting the dataset

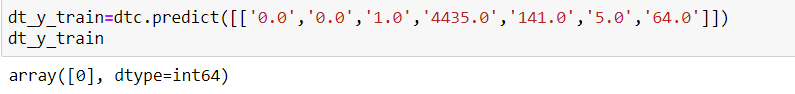
We have splinted the dataset as 80% of the data to train the model and 20% to test the model. For this we need to import the “train\_test\_split” from sklearn package. In this splitting we use class which have attributes like test\_size that specifies the percentage of test data and random\_state that can have values 0 or 1 which is used to set the test data from the dataset. In this the variables x\_train, y\_train refers to the training data and x\_test, y\_test refers to the test data. The train values are passed for training the model and the test values are passed for testing the model developed.



**Model Training:** To train the model we need to import the required model. As we are using the Decision Tree, we need to import the DecisionTreeClassifier class form sklearn.linear\_tree library. And the model is trained by passing the train data (x\_train and y\_train).



**Predicting:** Now we check the trained model by predicting the test data of dependent variable(dt\_y\_train) using the test data of independent variables(x\_test).



**Finding the accuracy score**: Now the accuracy score is predicted using trained data (x\_train and y\_train). On calculating we got accuracy score about 98% which is a good accuracy rate.

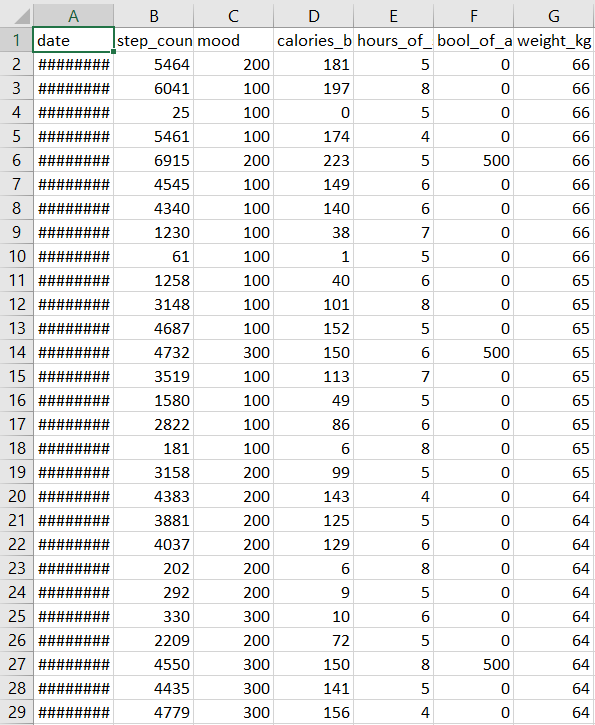


**Deploying the Prediction:**

The trained model is deployed in the Jupyter Notebook and UI is developed using the Flask Application using Spyder in Anaconda. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Poocco. Flask is based on the Werkzeg WSGI toolkit and the Jinja2 template engine. Both are Pocco projects.

Flask is a web framework, it’s a Python module that lets you develop web applications easily. It’s has a small and easy-to-extend core: it’s a microframework that doesn’t include an ORM (Object Relational Manager) or such features. It does have many cool features like URL (Uniform Resource Locator) routing, template engine. It is a WSGI web app framework.It is simply a Web Framework represents a collection of libraries and modules that enable web application developers to write applications without worrying about low-level details such as protocol, thread management.

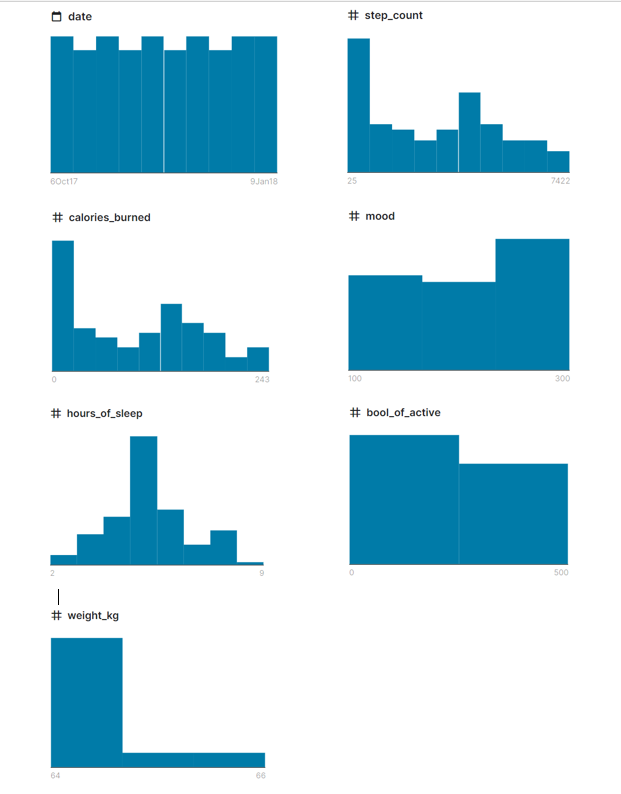
**2.3 Data Preparation:**

The data was sourced from Kaggle measure and save data's like mood, date, step\_count, calories\_burned etc. for every day. This file was taken from a Kaggle which contains data of 96 records and 7 attributes. The dataset contains 6 numerical attributes and a date providing various information. The data provided has no missing values.

A few observations are made based on information and histograms for numerical features

Date (For every day)  
Step Count: No. of steps did for a day  
Calories Burned(kcal): No. of calories burned for a day  
Mood: Mood of the person fro the given date  
Hours of Sleep(hrs): No. of hours did the person sleep for a day   
Feeling or Activeness or Inactiveness

Weight(kgs): Weight of the person for a given date



**CHAPTER - 4**

**EXPERIMENTAL INVESTIGATION**

1. **Choose a Project Idea:**

Body Fitness Prediction.

1. **Conduct Background Research**
2. **Compose a Hypothesis:**

Based on our Study and information gathered we can decide how fit a person is.

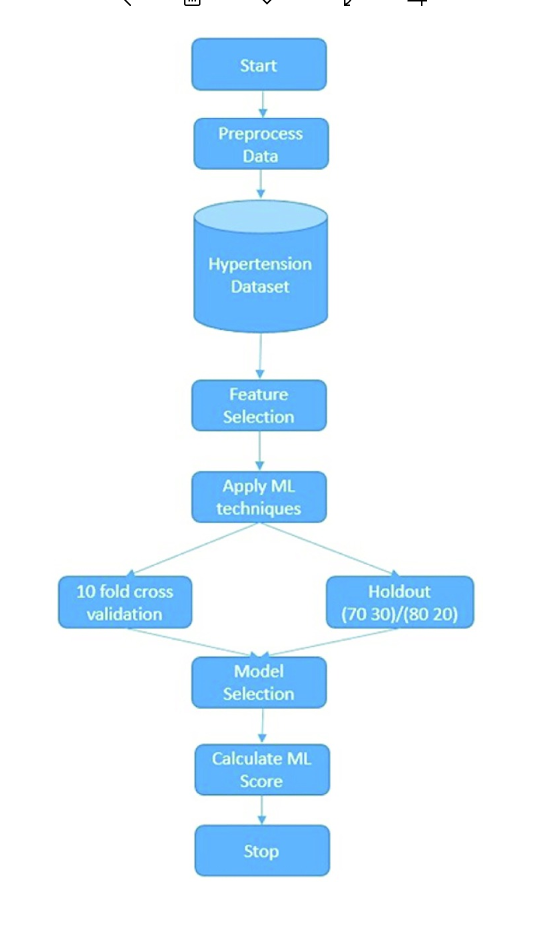
1. **Design your Experiment:**

First we need to collect the details like mood, step\_count, calories\_burned, hours\_of\_sleep, weight\_kg. Next, we give those values as input to the DecisionTreeClassifier which predicts the fitness of based on the given values.

1. **Draw Conclusions:**

After Building our model, we can able to know how fit a person is based on the values given to the model.

**CHAPTER – 5**

**FLOWCHART**

**CHAPTER – 6**

**RESULTS**

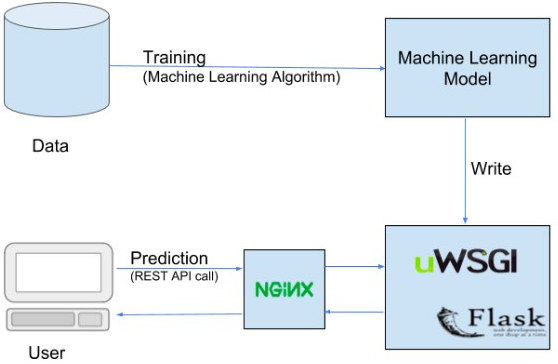
**Accuracy Score:**

The accuracy can be termed as closeness of measurements in statistical measures; however, it is also used in classifications. In classifications accuracy is the proportion of true results among the total number of cases. And to our model we got an accuracy of about 98% which indicates a good classification accuracy rate.



**Flask Application:**

Flask is a web framework, it’s a Python module that lets you develop web applications easily.The trained model is deployed in the Jupyter Notebook and UI is developed using the Flask Application using Spyder in Anaconda . It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Poocco. Flask is based on the Werkzeg WSGI toolkit and the Jinja2 template engine.Both are Pocco projects.



***Model Deployment using Flask Application***

**CHAPTER - 7**

**ADVANTAGES AND DISADVATAGES**

**Advantages:**

* Simple to understand and to interpret. Trees can be visualized.
* Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.
* The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
* Able to handle both numerical and categorical data. However, sci kit-learn implementation does not support categorical variables for now. Other techniques are usually specialized in analysing datasets that have only one type of variable.
* Able to handle multi-output problems.
* Uses a white box model. If a given situation is observable in a model, the explanation for the condition is easily explained by boolean logic. By contrast, in a black box model (e.g., in an artificial neural network), results may be more difficult to interpret.
* Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.
* Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.

**Disadvantages:**

* Decision-tree learners can create over-complex trees that do not generalize the data well. This is called over fitting. Mechanisms such as pruning, setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.
* Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.
* Predictions of decision trees are neither smooth nor continuous, but piece wise constant approximations as seen in the above figure. Therefore, they are not good at extrapolation.
* The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimally and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.
* There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.

**CHAPTER - 8**

**APPLICATIONS**

* This research indicates that two fitness components, explosive strength and agility , are of major importance for predicting sporting success in female karatekas .
* Therefore, karate coaches and trainers should pay special attention to those athletes who obtain high scores in field-based tests aimed at assessing both of these components , since they can indicate potentially successful elite competitors,
* In spite of this, studies that have tried to identify the fitness and anthropometric indicators that can help to predict international competitive success at a senior level in karate are almost non-existent.
* Consequently, it was not known which fitness and anthropometric attributes can help to discriminate those athletes with higher probability of success.
* The present research has its strengths and weaknesses.
* Its originality is a strong point, since, to the very best of the authors’ knowledge, no research focused on predictive models of sporting success in karatekas has been published so far.
* The considerable sample size, the very high standard of the karatekas included in the research , and the fact that the study focused specifically on female karatekas should also be highlighted.
* Another strong point worth mentioning is the external responsiveness shown by the simple fitness tests carried out, since they can be useful for a possible selection of talents as well as to discriminate among athletes of different competitive levels.

**CHAPTER -9**

**CONCLUSION**

Health is only beginning to see the potential of predictive analytics . The power of predictive algorithms will increase as more data and technologies present an enormous opportunity for new model development. The predictive power of a model developed on tens of millions of individuals across hundreds of input factors far exceeds traditional methods of training a regression model of ten inputs across a sample of a few thousand individuals . Big data technologies further empower data scientists to quickly build and test models within a few hours instead of days or weeks with traditional techniques.

**CHAPTER - 10**

**FUTURESCOPE**

This model is designed in such a way that the future modifications can be done easily. This application takes the manual details and the problems concern with it.

Well, we have worked hard to present an improved model better than the existing one’s the information about the various activities. Still, we found out that the entire project can be done in a better way.

Primarily, His/ Her giving the data manually. Instead of that we can use a camera which observes the body movement and heat releasing from the body which helps to know the person is fit or not but it helps only for indoor exercise. For outdoor we can use the sensors for knowing the person is fit or not. And we are also looking for better algorithm than decision tree.

Hence these are the enhancements we would like to suggest.

**CHAPTER – 11**

**BIBLIOGRAPHY**

* <https://www.youtube.com/watch?v=beCCPIH0-8c>
* <https://www.youtube.com/watch?v=vYreeoCoQPI>
* <https://www.youtube.com/watch?v=wUb--6FPBik>
* Smartinternz project description.

**CHAPTER - 12**

**APPENDIX**

**12.1 Source Code:**

**app.py:**

import numpy as np

import pandas as pd

from flask import Flask,render\_template,request

import requests

# NOTE: you must manually set API\_KEY below using information retrieved from your IBM Cloud account.

API\_KEY = "Uas2INotcbV0\_vnRrKwTvtEqu\_jw03g1YNDZmGsnh-Zj"

token\_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey": API\_KEY, "grant\_type": 'urn:ibm:params:oauth:grant-type:apikey'})

mltoken = token\_response.json()["access\_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

app = Flask(\_\_name\_\_)

@app.route('/')

def home():

return render\_template('index.html')

@app.route('/Prediction')

def prediction():

return render\_template('web.html')

@app.route('/Home')

def my\_home():

return render\_template('index.html')

@app.route('/predict',methods=['POST'])

def predict():

input\_features=[float(x) for x in request.form.values()]

features\_name=['sad','neutral','happy','step\_count','calories\_burned','hours\_of\_sleep','weight\_kg']

payload\_scoring = {"input\_data": [{"field": [features\_name], "values": [input\_features]}]}

response\_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/4022b48f-76ed-4773-8aa6-b34a070d10c3/predictions?version=2021-06-27', json=payload\_scoring, headers={'Authorization': 'Bearer ' + mltoken})

print("Scoring response")

predictions = response\_scoring.json()

print(predictions)

pred = predictions['predictions'][0]['values'][0][0]

if(pred == 0):

result = "You are not fit"

else:

result = "You are fit"

return render\_template('web.html',prediction\_text=result)

if \_\_name\_\_ =='\_\_main\_\_':

app.run(debug=False)

**index.html:**

<html>

<head>

<title>

Welcome user!

</title>

<link rel="stylesheet" href="../static/css/style1.css">

</head>

<body>

<div class="wholebody">

<img class="img" src="../static/images/bg.jpg" alt="BGImage"></img>

<div class="options">

<table cellpadding=10>

<tr>

<td><a href="http://localhost:5000/Home">Home</a></td>

<td><a href="http://localhost:5000/Prediction">Predict</a></td>

</tr>

</table>

</div>

<div class="content">

<h1>Body Fitness Prediction</h1>

<p><i>

Exercise is an important healt behaviour. Expressed reasons for participation are often

delayed outcomes i.e. health threats and benefits, but also enjoyment.

However, we donot know how people evaluate exercise as a reward. Delay discounting rates (k)

indicated that exercise was discounted like other consumable rewards at the same rate as food

and more rapidly than monetary rewards. Significant associations were detected of kex wiht

preferred speed and with extrinsic exercise motivation. Exercise training (n = 16) reduced kex

specifically, not affecting kfo. Our studies show, that participants perceived and discounted self-paced

walking/running like a consumable reward. Exercise discounting was quicker in individuals who preferred

lower speeds being less physically active and exercise training reduced the decay rate of exercise

specifically.

</i></p>

</div>

</div>

</body>

</html>

**web.html:**

<html>

<head>

<title>

Predicting from values

</title>

<link rel="stylesheet" href="../static/css/style2.css"></link>

</head>

<body>

<div class="head">

<h1>Body Fitness Prediction</h1>

<h3><i>A Machine Learning Web App, Built with Flask</i></h3>

</div><br/><br/>

<center>

<form action="http://localhost:5000/predict" method="POST" onsubmit="return validate()">

<table cellpadding=10>

<tr><td><select name="sad" id="s">

<option value=-1 selected>Is your mood is sad</option>

<option value=100>Yes</option>

<option value=0>No</option>

</select></td></tr>

<tr><td><select name="neutral" id="n">

<option value=-1 selected>Is your mood is neutral</option>

<option value=200>Yes</option>

<option value=0>No</option>

</select></td></tr>

<tr><td><select name="happy" id="h">

<option value=-1 selected>Is your mood is happy</option>

<option value=300>Yes</option>

<option value=0>No</option>

</select></td></tr>

<tr><td><input type="number" name="step\_counts" placeholder="Enter the step counts" required /></td></tr>

<tr><td><input type="number" name="calories\_burned" placeholder="Enter the calories\_burned" required /></td></tr>

<tr><td><input type="number" name="hrs\_of\_sleep" placeholder="Enter the hours of sleep" required /></td></tr>

<tr><td><input type="number" name="weight\_kg" placeholder="Enter your weight in kg" required /></td></tr>

<tr><td> <input class="btn" type="submit" value="Predict" /></td></tr>

</table>

</form><br/><br/><br/>

<h1>{{prediction\_text}}</h1>

</center>

<script>

function validate() {

var s,n,h;

s=parseInt(document.getElementById("s").value);

n=parseInt(document.getElementById("n").value);

h=parseInt(document.getElementById("h").value);

if(s==-1) {

alert("Please select whether your mood is sad or not")

return false;

}

if(n==-1) {

alert("Please select whether your mood is neutral or not")

return false;

}

if(h==-1) {

alert("Please select whether your mood is happy or not")

return false;

}

if((s>0 && n>0) || (s>0 && h>0) || (n>0 && h>0)) {

alert("Please select the mood properly.\nYou cannot have more than one mood at a time");

return false;

}

}

</script>

</body>

</html>

**style1.css:**

body,html{

margin:0;

}

.wholebody{

position: relative;

color: white;

}

.img{

width:100%;

height:100%;

}

.options{

position:absolute;

top:0;

right:0;

}

a{

text-decoration: none;

color:rgb(122, 121, 122);

font-size:22px;

}

.content{

position: absolute;

text-align: center;

top:170px;

}

h1{

font-size:40px;

}

p{

font-size:20px;

}

**style1.css**

.head{

text-align: center;

color:white;

margin-left: 50px;

margin-right: 50px;

border-radius:30px;

background-color: #f44336;

}

h1{

font-family: cursive;

font-size: 38px;

margin:0;

}

h3{

font-size:13px;

margin:7;

}

table{

text-align:center;

}

.btn{

font-size: 20px;

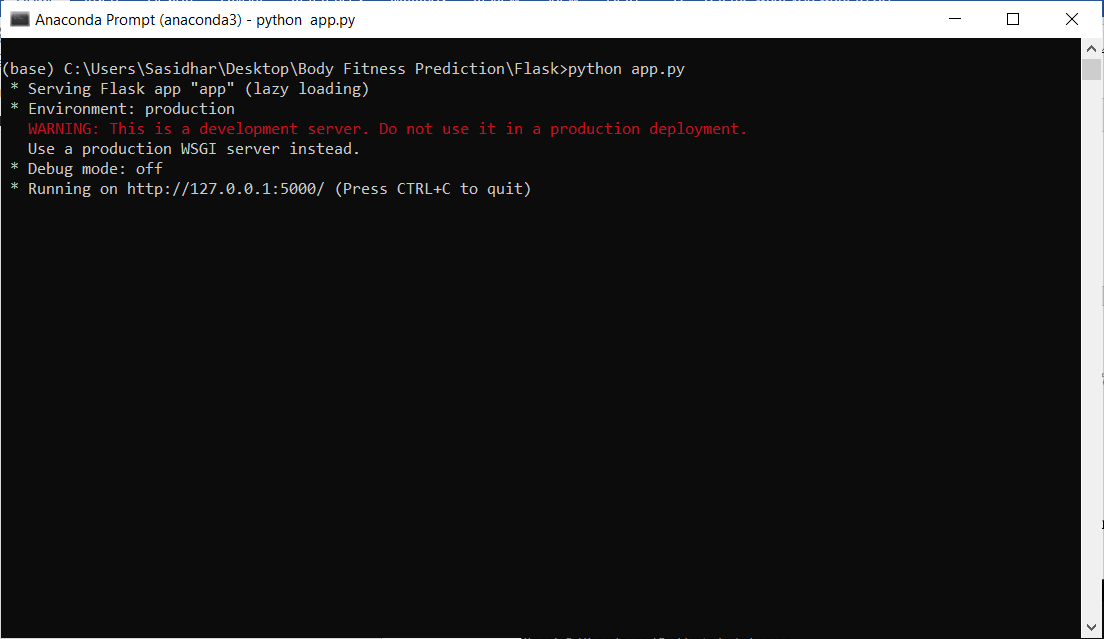
cursor:pointer;

border-radius: 15px;

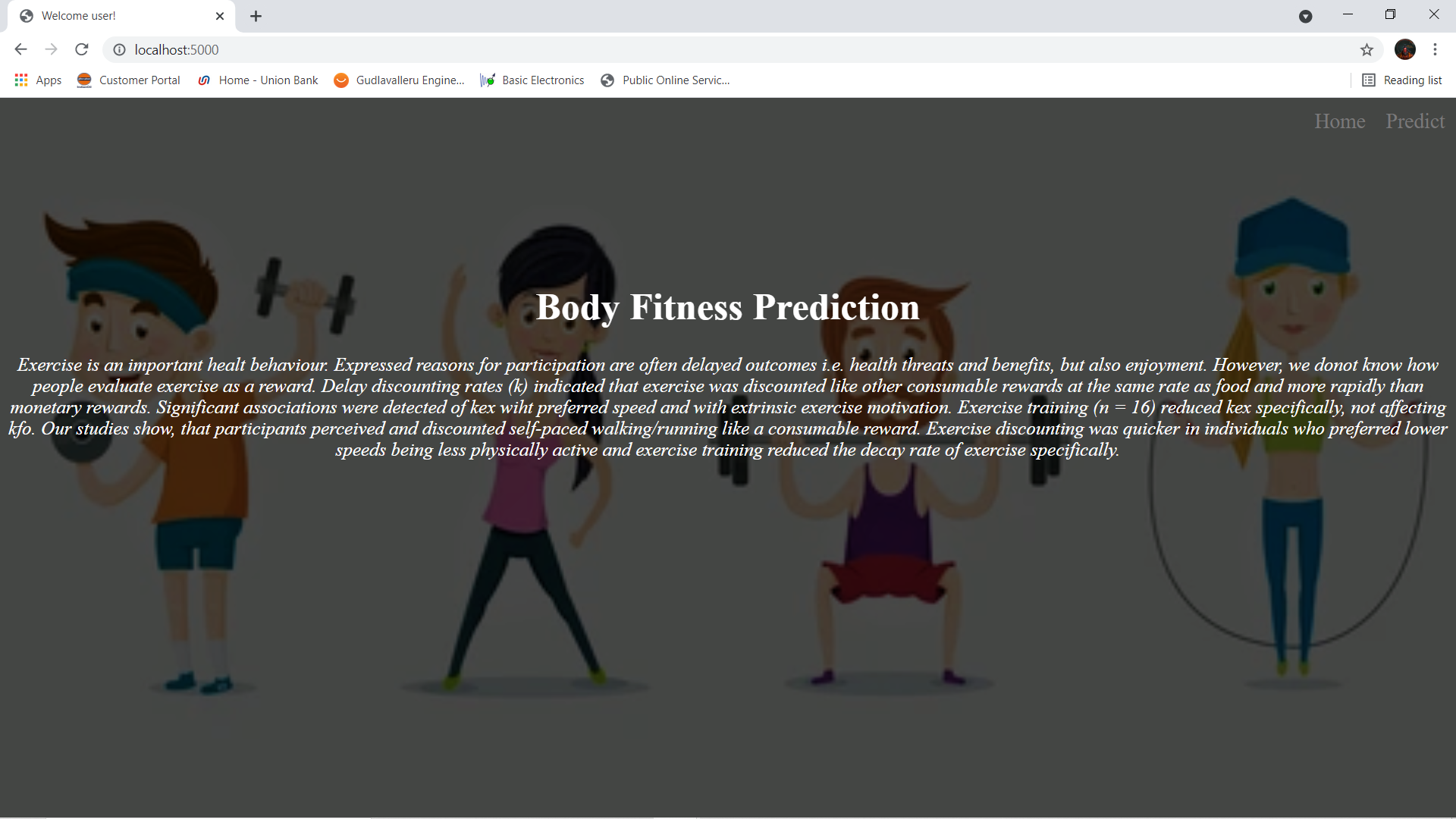
box-shadow: 2px 1px black;

}

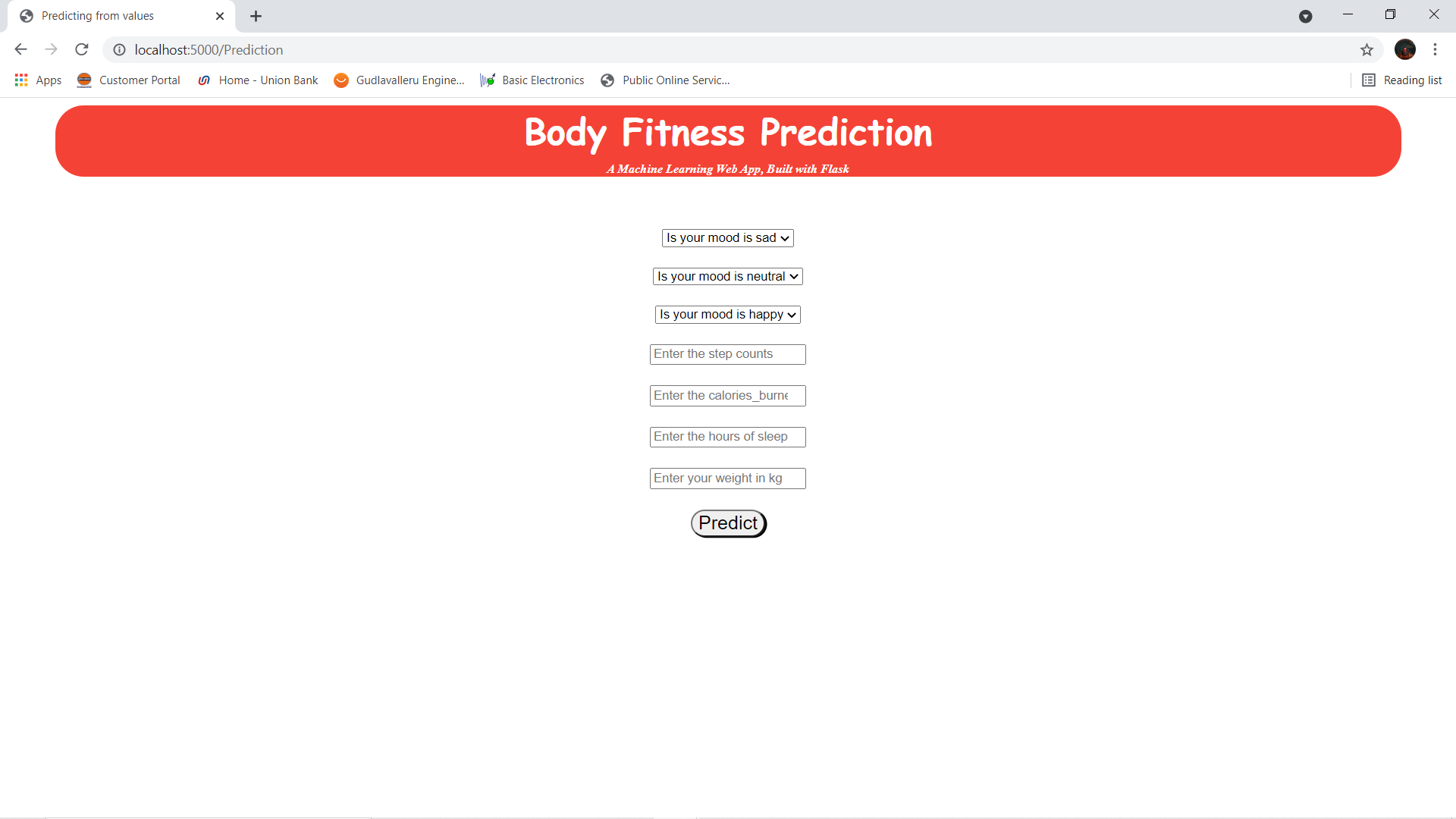
**12.2 Outputs:**

The output is displayed for running the Flask Application using Spyder in Anaconda command prompt.

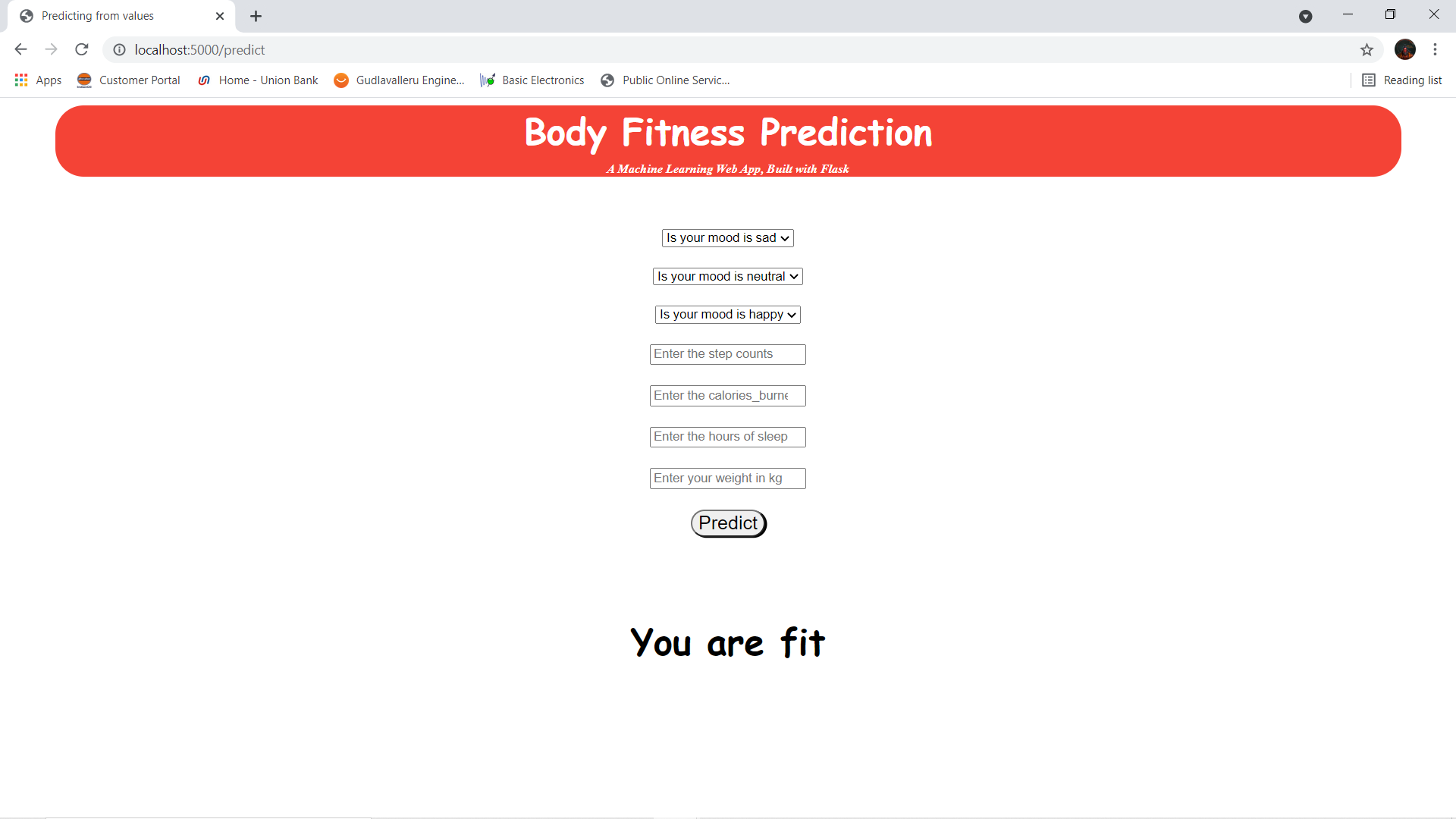
***Figure 4.1: Working on the results in prompt***



***Figure 4.2: User Interface of the project (Home Page)***



***Figure 4.3: User Interface of the project (Prediction Page)***

***Figure 4.4: Showing the resultant predicted output***