CEREAL ANALYSIS BASED ON RATINGS BY USING MACHINE LEARNING TECHNIQUES BY IBM WATSON

Overview

A customer wants to buy some food items with high dietary benefits so that he wants to know which food item has high dietary benefits. It is so difficult to choose an item .Usually a customer expects to consume dietary cereals with high proteins, fiber and low sugars, fats. Predicting a brand with high dietary cereals became a big issue. The project objective is to find the high dietary food that is predicted on the basis of rating of the food.

1. To find which quantities are showing more impact on the rating of food.
2. To show the food which is impacting less on the rating of food?

We use machine learning algorithms to predict the food with a high beneficiary diet. The model can predict the rating of the food more accurately by giving the inputs which are the cereals and ingredients present in the food. Thus a customer can get high dietary food by the rating of the food given to it from the cereals and ingredients present. The rating is predicted using the neural networks model.

Purpose

The purpose of this analysis is to take a close look at the cereal industry. I order to do so it is necessary within this analysis to identify the market structure of the industry. It is also imperative to recognize the main competitors within the industry in order to calculate the market share of the main companies that dominate the current industry.

LITERATURE SURVEY

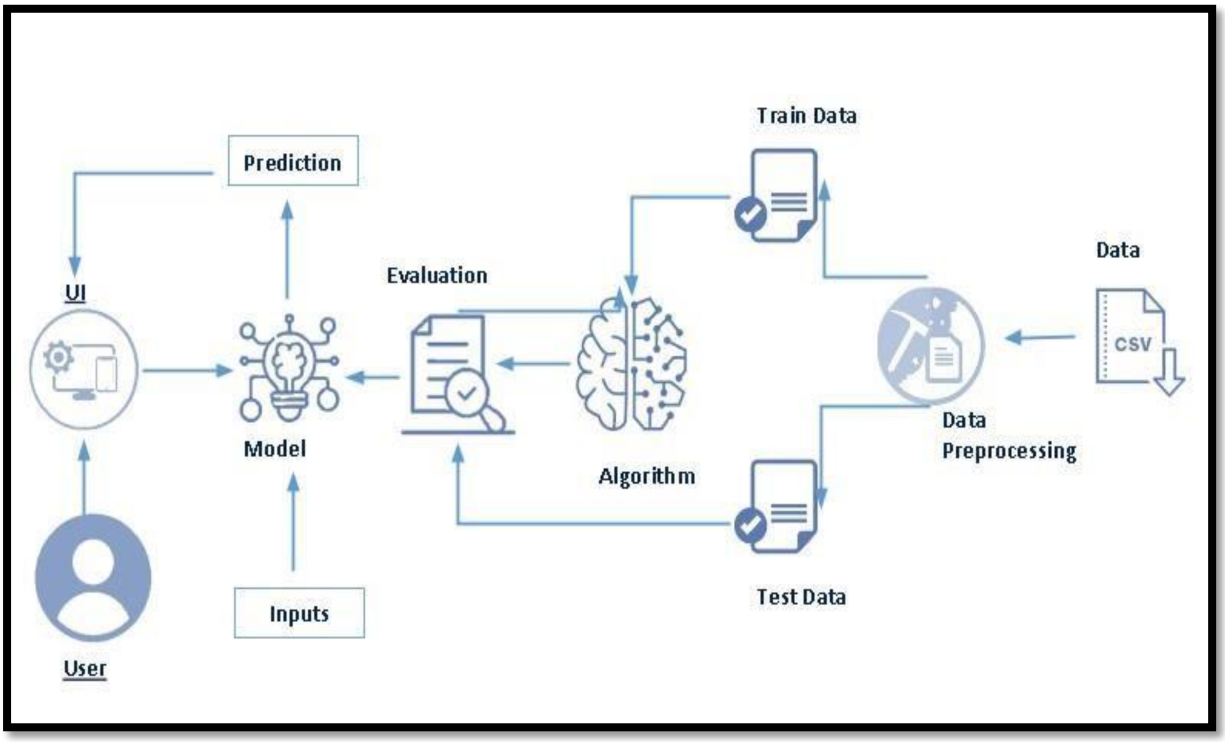
Existing Problem

Problem identification in major Cereal Analysis .The data was analysed through statistical package system. Index value was used to identify the major problem of the major cereal crops. The major problems before the cultivation of cereal crops were found to be unavailability of hybrid seeds, weeds and grass problems, irrigation problems, labours and mechanization problems fertilizers and manures problems etc.

Proposed Solution

- The strategy used by the companies within the industry in order to give a better understanding on how the main competitors diversify their products to meet consumers needs. This will give us further understanding on not only the market structure but also how this has been able to dominate the market. It is able to understand the problem to classify if it is a regression or a classification kind of problem. It is able to analyze or get insights of data through visualization.

Technical Architecture:



Hardware/Software designing-

Hardware Requirements:

- System with 2-core processor

Software Requirements:

- Spyder
- Anaconda
- Jupyter Notebook
- IBM-Watson Studio
- Any browser eg:Google chrome

Project Structure

Create a Project folder which contains files as show

Milestone 1: Data Collection

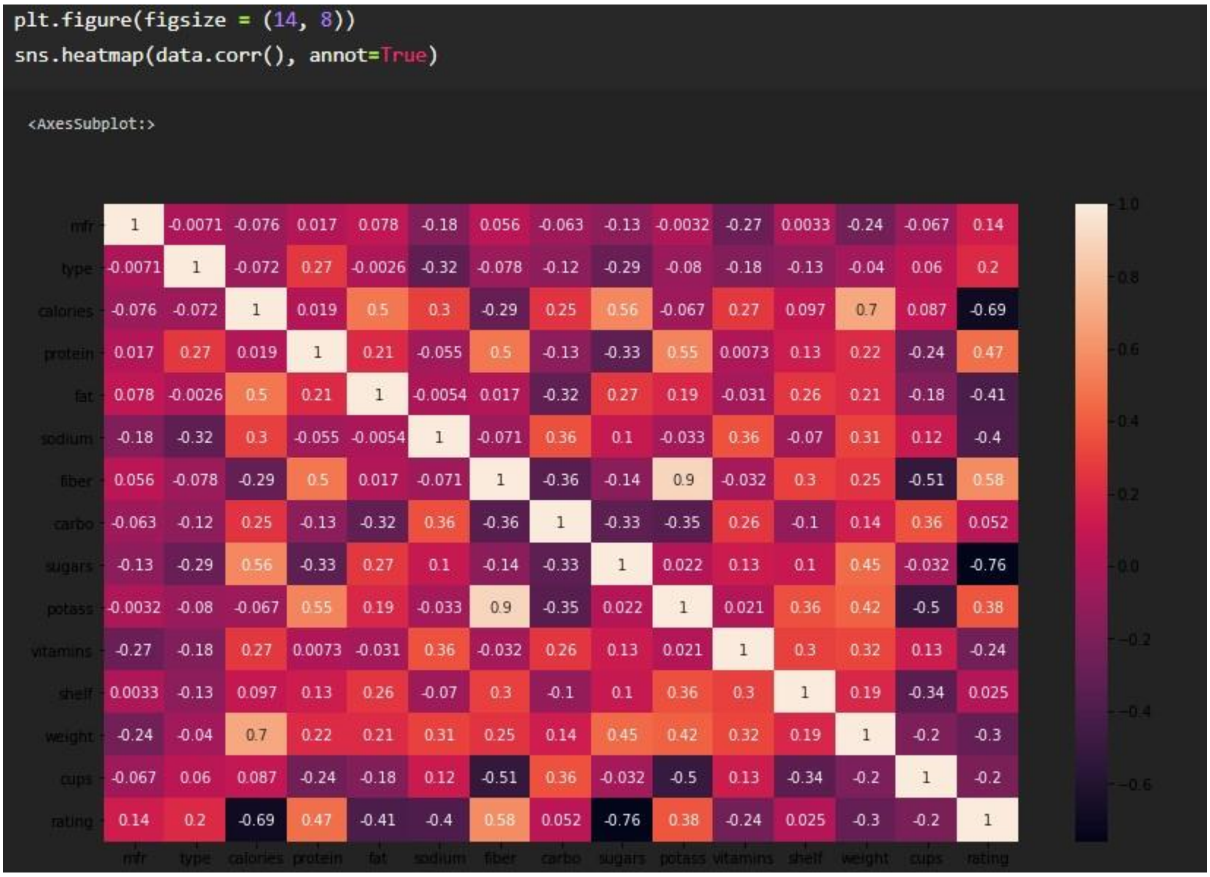
ML depends heavily on data, without data, it is impossible for an “AI” to learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions.

Activity 1: Download the dataset

You can collect datasets from different open sources like kaggle.com, data.gov, UCI machine learning repository etc.

The dataset used for this project was obtained from Kaggle . Please refer to the link given below to download the data set and to know about the dataset

Link: ~~https://www.kaggle.com/crawford/80-cereals~~



```
plt.figure(figsize = (14, 8))
sns.heatmap(data.corr(), annot=True)
```

```
<AxesSubplot:>
```

	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
mfr	1	-0.0071	-0.076	0.017	0.078	-0.18	0.056	-0.063	-0.13	-0.0032	-0.27	0.0033	-0.24	-0.067	0.14
type	-0.0071	1	-0.072	0.27	-0.0026	-0.32	-0.078	-0.12	-0.29	-0.08	-0.18	-0.13	-0.04	0.06	0.2
calories	-0.076	-0.072	1	0.019	0.5	0.3	-0.29	0.25	0.56	-0.067	0.27	0.097	0.7	0.087	-0.69
protein	0.017	0.27	0.019	1	0.21	-0.055	0.5	-0.13	-0.33	0.55	0.0073	0.13	0.22	-0.24	0.47
fat	0.078	-0.0026	0.5	0.21	1	-0.0054	0.017	-0.32	0.27	0.19	-0.031	0.26	0.21	-0.18	-0.41
sodium	-0.18	-0.32	0.3	-0.055	-0.0054	1	-0.071	0.36	0.1	-0.033	0.36	-0.07	0.31	0.12	-0.4
fiber	0.056	-0.078	-0.29	0.5	0.017	-0.071	1	-0.36	-0.14	0.9	-0.032	0.3	0.25	-0.51	0.58
carbo	-0.063	-0.12	0.25	-0.13	-0.32	0.36	-0.36	1	-0.33	-0.35	0.26	-0.1	0.14	0.36	0.052
sugars	-0.13	-0.29	0.56	-0.33	0.27	0.1	-0.14	-0.33	1	0.022	0.13	0.1	0.45	-0.032	-0.76
potass	-0.0032	-0.08	-0.067	0.55	0.19	-0.033	0.9	-0.35	0.022	1	0.021	0.36	0.42	-0.5	0.38
vitamins	-0.27	-0.18	0.27	0.0073	-0.031	0.36	-0.032	0.26	0.13	0.021	1	0.3	0.32	0.13	-0.24
shelf	0.0033	-0.13	0.097	0.13	0.26	-0.07	0.3	-0.1	0.1	0.36	0.3	1	0.19	-0.34	0.025
weight	-0.24	-0.04	0.7	0.22	0.21	0.31	0.25	0.14	0.45	0.42	0.32	0.19	1	-0.2	-0.3
cups	-0.067	0.06	0.087	-0.24	-0.18	0.12	-0.51	0.36	-0.032	-0.5	0.13	-0.34	-0.2	1	-0.2
rating	0.14	0.2	-0.69	0.47	-0.41	-0.4	0.58	0.052	-0.76	0.38	-0.24	0.025	-0.3	-0.2	1

From the heatmap, we see that there are no missing values in the dataset.

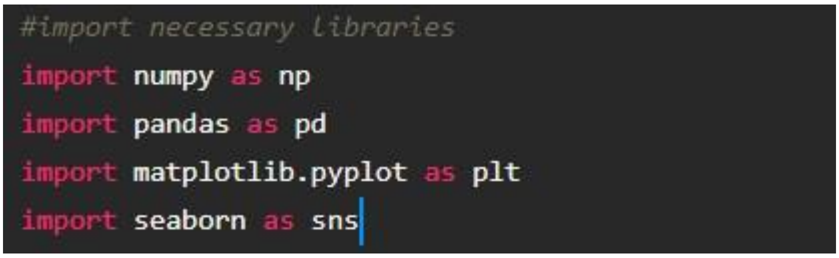
Milestone 2: Data Pre-processing

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analysing techniques.

Note: There is n number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

Activity 1: Importing the libraries

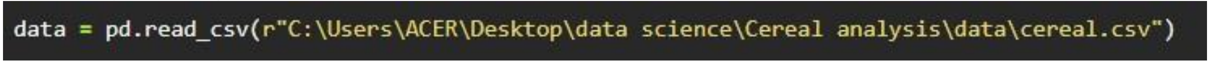
- It is important to import all the necessary libraries such as pandas, numpy, matplotlib.
- **Numpy-** It is an open-source numerical Python library. It contains a multi-dimensional array and matrix data structures. It can be used to perform mathematical operations on arrays such as trigonometric, statistical, and algebraic routines.
- **Pandas-** It is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.
- **Seaborn-** Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- **Matplotlib-** Visualisation with python. It is a comprehensive library for creating static, animated, and interactive visualizations in Python



```
#import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Activity 2: Importing The Dataset

- You might have your data in .csv files, .excel files
- Let’s load a .csv data file into pandas using read_csv() function.We will need to locate the directory of the CSV file at first (it’s more efficient to keep the dataset in the same directory as your program).



```
data = pd.read_csv(r"C:\Users\ACER\Desktop\data science\Cereal analysis\data\cereal.csv")
```

- If your dataset is in some other location ,Then

Data=pd.read_csv(r”File_location”)

Note:r stands for "raw" and will cause backslashes in the string to be interpreted as actual backslashes rather than special characters.

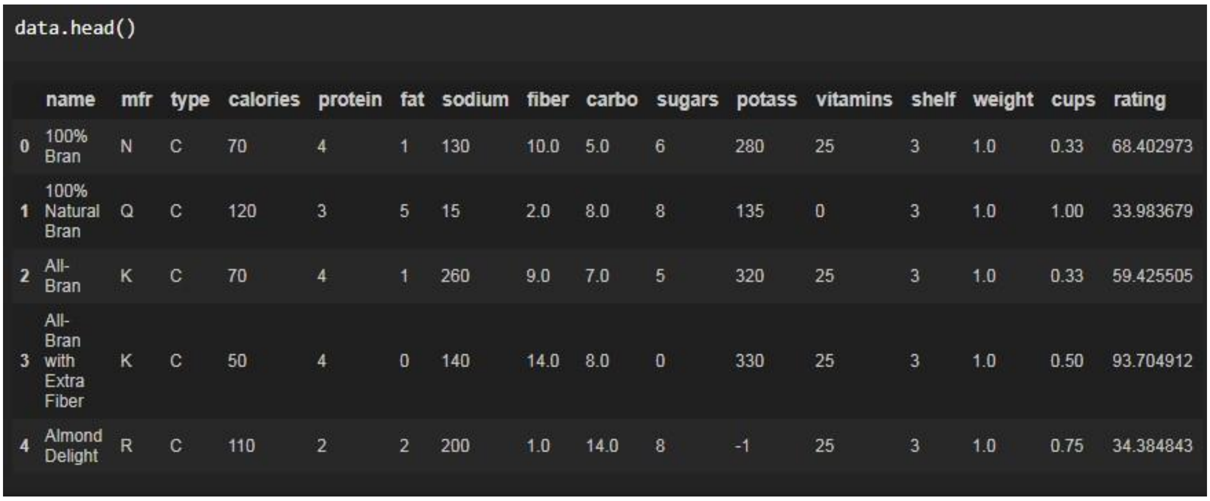
- If the dataset is in the same directory of your program, you can directly read it, without giving raw as r.
- Our Dataset cereal contains following Columns

1. name
2. mfr
3. type
4. calories
5. protein
6. fat
7. sodium
8. fiber
9. carbo
10. sugars
11. potass
12. vitamins
13. shelf
14. weight
15. cups
16. rating

The output column to be predicted is rating .Based on the input variables we predict the food with a high beneficiary diet.

Activity 3: Analyse The Data

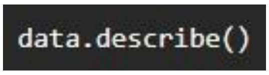
- head() method is used to return top n (5 by default) rows of a DataFrame or series.



```
data.head()
```

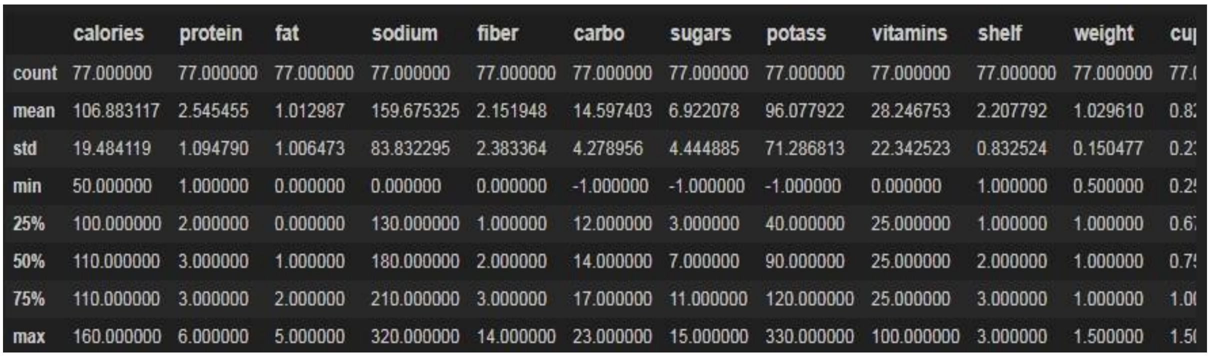
	name	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
0	100% Bran	N	C	70	4	1	130	10.0	5.0	6	280	25	3	1.0	0.33	68.402973
1	100% Natural Bran	Q	C	120	3	5	15	2.0	8.0	8	135	0	3	1.0	1.00	33.983679
2	All-Bran	K	C	70	4	1	260	9.0	7.0	5	320	25	3	1.0	0.33	59.425505
3	All-Bran with Extra Fiber	K	C	50	4	0	140	14.0	8.0	0	330	25	3	1.0	0.50	93.704912
4	Almond Delight	R	C	110	2	2	200	1.0	14.0	8	-1	25	3	1.0	0.75	34.384843

- describe() method computes a summary of statistics like count, mean, standard deviation, min, max and quartile values.



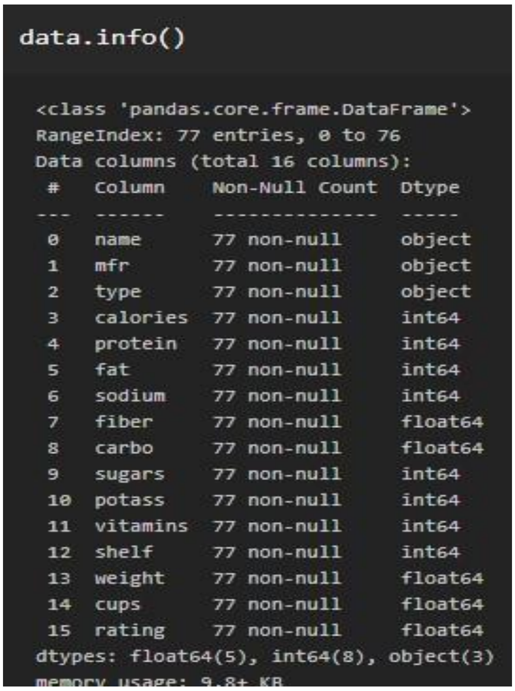
```
data.describe()
```

The output is as shown below



	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cu
count	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.0
mean	106.883117	2.545455	1.012987	159.675325	2.151948	14.597403	6.922078	96.077922	28.246753	2.207792	1.029610	0.8
std	19.484119	1.094790	1.006473	83.832295	2.383364	4.278956	4.444885	71.286813	22.342523	0.832524	0.150477	0.2
min	50.000000	1.000000	0.000000	0.000000	0.000000	-1.000000	-1.000000	-1.000000	0.000000	1.000000	0.500000	0.2
25%	100.000000	2.000000	0.000000	130.000000	1.000000	12.000000	3.000000	40.000000	25.000000	1.000000	1.000000	0.6
50%	110.000000	3.000000	1.000000	180.000000	2.000000	14.000000	7.000000	90.000000	25.000000	2.000000	1.000000	0.7
75%	110.000000	3.000000	2.000000	210.000000	3.000000	17.000000	11.000000	120.000000	25.000000	3.000000	1.000000	1.0
max	160.000000	6.000000	5.000000	320.000000	14.000000	23.000000	15.000000	330.000000	100.000000	3.000000	1.500000	1.5

- info() gives information about the data



```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 77 entries, 0 to 76
Data columns (total 16 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   name      77 non-null     object
 1   mfr       77 non-null     object
 2   type      77 non-null     object
 3   calories  77 non-null     int64
 4   protein   77 non-null     int64
 5   fat       77 non-null     int64
 6   sodium    77 non-null     int64
 7   fiber     77 non-null     float64
 8   carbo     77 non-null     float64
 9   sugars    77 non-null     int64
 10  potass    77 non-null     int64
 11  vitamins  77 non-null     int64
 12  shelf     77 non-null     int64
 13  weight    77 non-null     float64
 14  cups      77 non-null     float64
 15  rating    77 non-null     float64
dtypes: float64(5), int64(8), object(3)
memory usage: 9.8+ KB
```

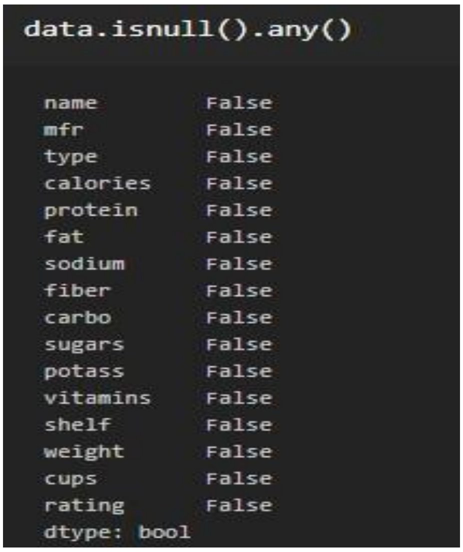
Activity 4: Handling Missing Values

1.After loading it is important to check the complete information of data as it can indication many of the hidden information such as null values in a column or a row

2.Check whether any null values are there or not. if it is present then following can be done,

a.Imputing data using Imputation method in sklearn

b.Filling NaN values with mean, median and mode using fillna() method.



```
data.isnull().any()

  name        False
  mfr         False
  type        False
  calories    False
  protein     False
  fat         False
  sodium      False
  fiber       False
  carbo       False
  sugars      False
  potass      False
  vitamins    False
  shelf       False
  weight      False
  cups        False
  rating      False
  dtype: bool
```

3. Heatmap**: It is way of representing the data in 2-D form. It gives coloured visual summary of the data.**

Activity 5: Data Visualisation

- Data visualization is where a given data set is presented in a graphical format. It helps the detection of patterns, trends and correlations that might go undetected in text-based data.
- Understanding your data and the relationship present within it is just as important as any algorithm used to train your machine learning model. In fact, even the most sophisticated machine learning models will perform poorly on data that wasn’t visualized and understood properly.
- To visualize the dataset we need libraries called Matplotlib and Seaborn.
- The Matplotlib library is a Python 2D plotting library which allows you to generate plots, scatter plots, histograms, bar charts etc.

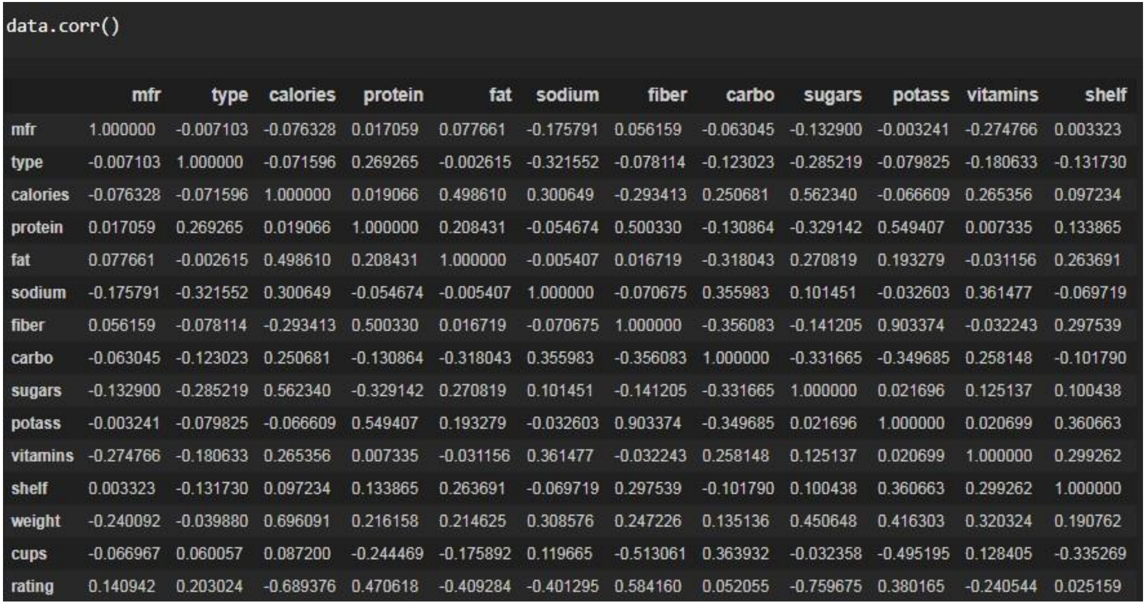
Let’s visualize our data using Matplotlib and seaborn library.

Before diving into the code, let's look at some of the basic properties we will be using when plotting.

xlabel: Set the label for the x-axis. **ylabel**: Set the label for the y-axis. title: Set a title for the axes.

Legend: Place a legend on the axes.

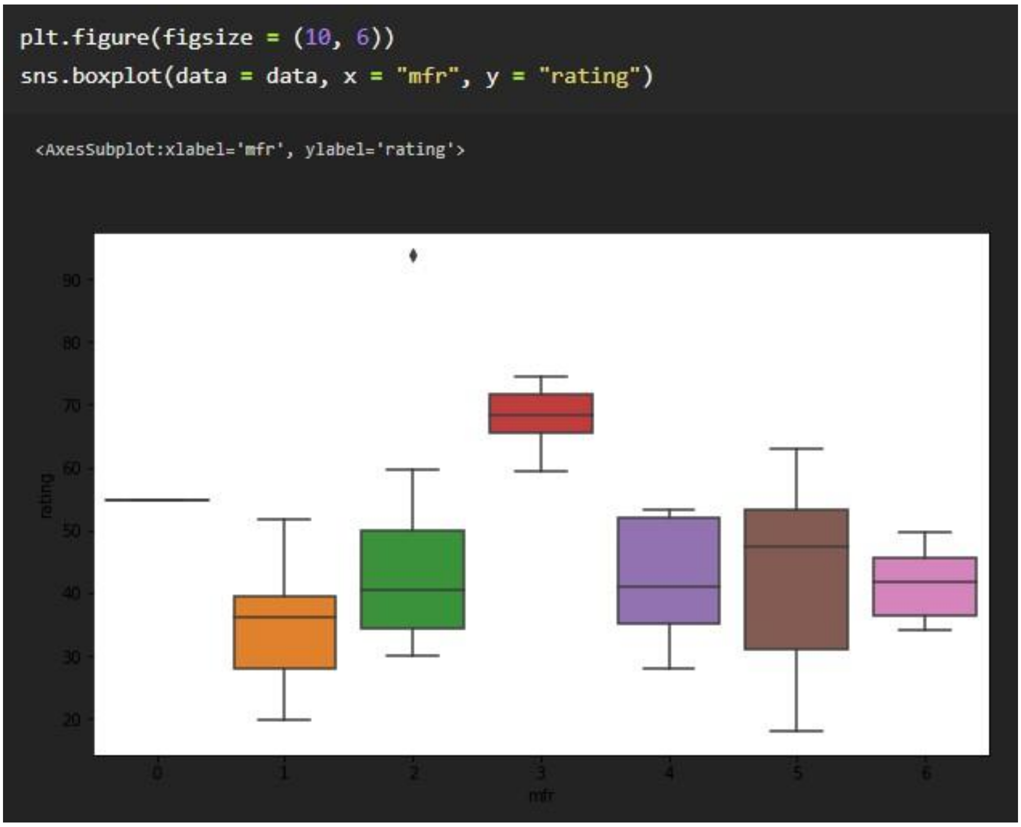
1. data.corr() gives the correlation between the columns



```
data.corr()
```

	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf
mfr	1.000000	-0.007103	-0.076328	0.017059	0.077661	-0.175791	0.056159	-0.063045	-0.132900	-0.003241	-0.274766	0.003323
type	-0.007103	1.000000	-0.071596	0.269265	-0.002615	-0.321552	-0.078114	-0.123023	-0.285219	-0.079825	-0.180633	-0.131730
calories	-0.076328	-0.071596	1.000000	0.019066	0.498610	0.300649	-0.293413	0.250681	0.562340	-0.066609	0.265356	0.097234
protein	0.017059	0.269265	0.019066	1.000000	0.208431	-0.054674	0.500330	-0.130864	-0.329142	0.549407	0.007335	0.133865
fat	0.077661	-0.002615	0.498610	0.208431	1.000000	-0.005407	0.016719	-0.318043	0.270819	0.193279	-0.031156	0.263691
sodium	-0.175791	-0.321552	0.300649	-0.054674	-0.005407	1.000000	-0.070675	0.355983	0.101451	-0.032603	0.361477	-0.069719
fiber	0.056159	-0.078114	-0.293413	0.500330	0.016719	-0.070675	1.000000	-0.356083	-0.141205	0.903374	-0.032243	0.297539
carbo	-0.063045	-0.123023	0.250681	-0.130864	-0.318043	0.355983	-0.356083	1.000000	-0.331665	-0.349685	0.258148	-0.101790
sugars	-0.132900	-0.285219	0.562340	-0.329142	0.270819	0.101451	-0.141205	-0.331665	1.000000	0.021696	0.125137	0.100438
potass	-0.003241	-0.079825	-0.066609	0.549407	0.193279	-0.032603	0.903374	-0.349685	0.021696	1.000000	0.020699	0.360663
vitamins	-0.274766	-0.180633	0.265356	0.007335	-0.031156	0.361477	-0.032243	0.258148	0.125137	0.020699	1.000000	0.299262
shelf	0.003323	-0.131730	0.097234	0.133865	0.263691	-0.069719	0.297539	-0.101790	0.100438	0.360663	0.299262	1.000000
weight	-0.240092	-0.039880	0.696091	0.216158	0.214625	0.308576	0.247226	0.135136	0.450648	0.416303	0.320324	0.190762
cups	-0.066967	0.060057	0.087200	-0.244469	-0.175892	0.119665	-0.513061	0.363932	-0.032358	-0.495195	0.128405	-0.335269
rating	0.140942	0.203024	-0.689376	0.470618	-0.409284	-0.401295	0.584160	0.052055	-0.759675	0.380165	-0.240544	0.025159

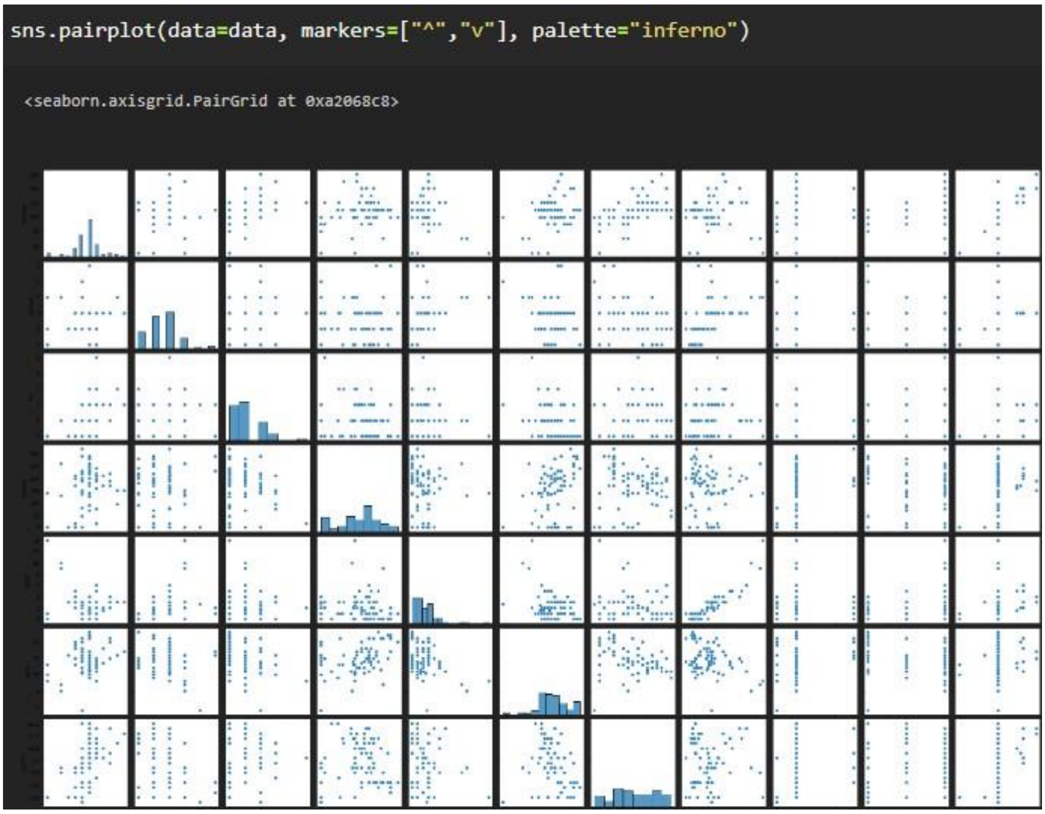
2. Box plot



```
plt.figure(figsize = (10, 6))
sns.boxplot(data = data, x = "mfr", y = "rating")
```

```
<AxesSubplot:xlabel='mfr', ylabel='rating'>
```

3. Pairplot



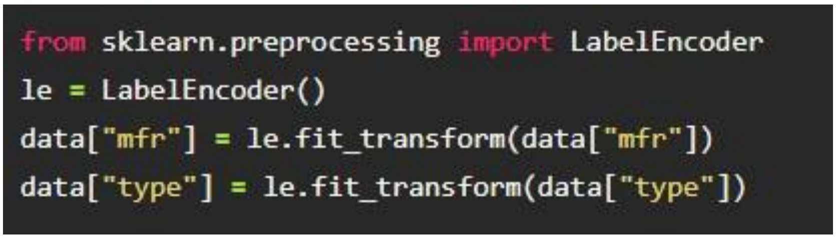
```
sns.pairplot(data=data, markers=["^","v"], palette="inferno")
```

```
<seaborn.axisgrid.PairGrid at 0xa2068c8>
```

Activity 6: Label Encoding

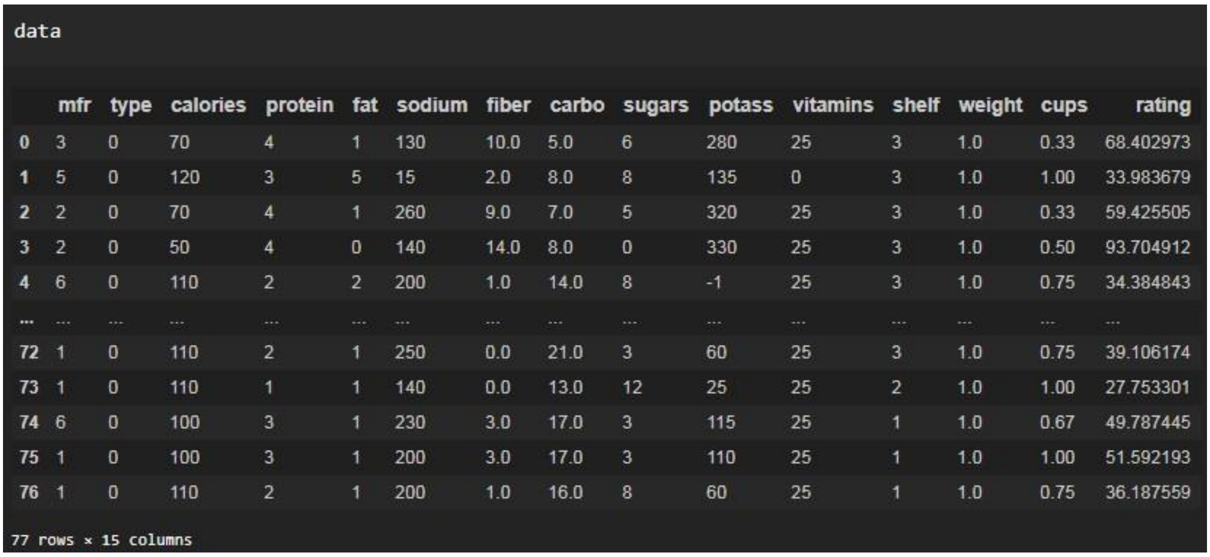
In machine learning, we usually deal with datasets which contain multiple labels in one or more than one column. These labels can be in the form of words or numbers. To make the data understandable or in human readable form, the training data is often labelled in words.

Label Encoding refers to converting the labels into numeric form so as to convert it into the machine-readable form. Machine learning algorithms can then decide in a better way on how those labels must be operated. It is an important preprocessing step for the structured dataset in supervised learning.



```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data["mfr"] = le.fit_transform(data["mfr"])
data["type"] = le.fit_transform(data["type"])
```

Data after label encoding



```
data
```

	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
0	3	0	70	4	1	130	10.0	5.0	6	280	25	3	1.0	0.33	68.402973
1	5	0	120	3	5	15	2.0	8.0	8	135	0	3	1.0	1.00	33.983679
2	2	0	70	4	1	260	9.0	7.0	5	320	25	3	1.0	0.33	59.425505
3	2	0	50	4	0	140	14.0	8.0	0	330	25	3	1.0	0.50	93.704912
4	6	0	110	2	2	200	1.0	14.0	8	-1	25	3	1.0	0.75	34.384843
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
72	1	0	110	2	1	250	0.0	21.0	3	60	25	3	1.0	0.75	39.106174
73	1	0	110	1	1	140	0.0	13.0	12	25	25	2	1.0	1.00	27.753301
74	6	0	100	3	1	230	3.0	17.0	3	115	25	1	1.0	0.67	49.787445
75	1	0	100	3	1	200	3.0	17.0	3	110	25	1	1.0	1.00	51.592193
76	1	0	110	2	1	200	1.0	16.0	8	60	25	1	1.0	0.75	36.187559

77 rows × 15 columns

All the data is converted into numerical values **.**

Activity 7: Splitting The Dataset Into Dependent And Independent Variable

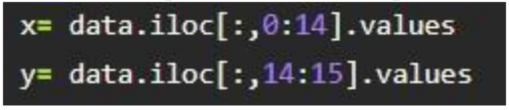
- In machine learning, the concept of dependent variable (y) and independent variables(x) is important to understand. Here, Dependent variable is nothing but output in the dataset and the independent variable is all inputs in the dataset.
- With this in mind, we need to split our dataset into the matrix of independent variables and the vector or dependent variable. Mathematically, Vector is defined as a matrix that has just one column.

To read the columns, we will use **iloc** of pandas (used to fix the indexes for selection) which takes two parameters — [row selection, column selection].

Let's split our dataset into independent and dependent variables.

1. The independent variable in the dataset would be considered as 'x' and name, mfr, type, calories , protein , fat, sodium , fiber, carbo, sugars , potass , vitamins , shelf , weight , cups columns would be considered as independent variable.
2. The dependent variable in the dataset would be considered as 'y' and the ' rating ' column is considered as dependent variable.

Now we will split the data of independent variables,

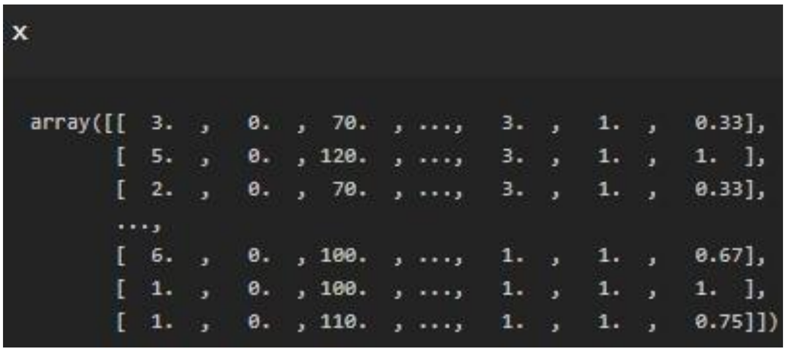


```
x= data.iloc[:,0:14].values
y= data.iloc[:,14:15].values
```

From the above code ":" indicates that you are considering all the rows in the dataset and "**0:18**" indicates that you are considering columns 0 to 8 such as sex, job and purpose as input values and assigning them to variable x. In the same way in second line ":" indicates you are considering all the rows and "**18:19**" indicates that you are considering only last column as output value and assigning them to variable y.

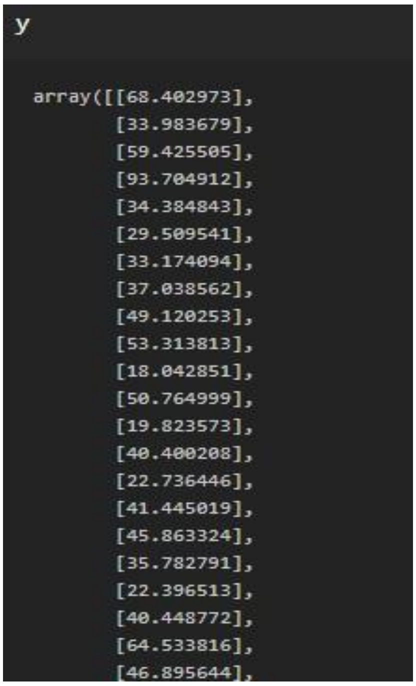
After splitting we see the data as below

x



```
x

array([[  3.  ,   0.  ,  70.  , ...,   3.  ,   1.  ,   0.33],
       [  5.  ,   0.  , 120.  , ...,   3.  ,   1.  ,   1.  ],
       [  2.  ,   0.  ,  70.  , ...,   3.  ,   1.  ,   0.33],
       ...,
       [  6.  ,   0.  , 100.  , ...,   1.  ,   1.  ,   0.67],
       [  1.  ,   0.  , 100.  , ...,   1.  ,   1.  ,   1.  ],
       [  1.  ,   0.  , 110.  , ...,   1.  ,   1.  ,   0.75]])
```

Y



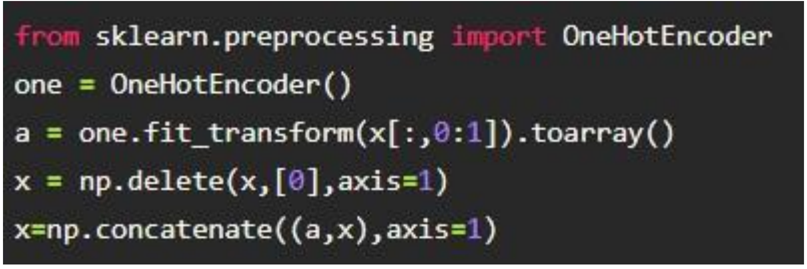
```
y

  array([[68.402973],
         [33.983679],
         [59.425505],
         [93.704912],
         [34.384843],
         [29.509541],
         [33.174094],
         [37.038562],
         [49.120253],
         [53.313813],
         [18.042851],
         [50.764999],
         [19.823573],
         [40.400208],
         [22.736446],
         [41.445019],
         [45.863324],
         [35.782791],
         [22.396513],
         [40.448772],
         [64.533816],
         [46.895644],
```

Activity 8: OneHot Encoding

Sometimes in datasets, we encounter columns that contain numbers of no specific order of preference. The data in the column usually denotes a category or value of the category and also when the data in the column is label encoded. This confuses the machine learning model, to avoid this, the data in the column should be One Hot encoded.

One Hot Encoding –

It refers to splitting the column which contains numerical categorical data to many columns depending on the number of categories present in that column. Each column contains “0” or “1” corresponding to which column it has been placed.

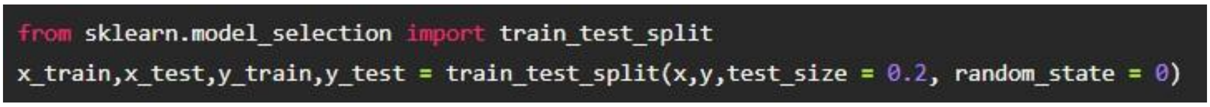


```
from sklearn.preprocessing import OneHotEncoder
one = OneHotEncoder()
a = one.fit_transform(x[:,0:1]).toarray()
x = np.delete(x,[0],axis=1)
x=np.concatenate((a,x),axis=1)
```

Activity 9: Splitting The Data Into Train And Test

- When you are working on a model and you want to train it, you obviously have a dataset. But after training, we have to test the model on some test dataset. For this, you will have a dataset which is different from the training set you used earlier. But it might not always be possible to have so much data during the development phase. In such cases, the solution is to split the dataset into two sets, one for training and the other for testing. □ But the question is, how do you split the data? You can’t possibly manually split the dataset into two sets. And you also have to make sure you split the data in a random manner. To help us with this task, the Scikit library provides a tool, called the Model Selection library. There is a class in the library which is,‘train_test_split.’ Using this we can easily split the dataset into the training and the testing datasets in various proportions.
- The train-test split is a technique for evaluating the performance of a machine learning algorithm.
- Train Dataset: Used to fit the machine learning model.
- Test Dataset: Used to evaluate the fit machine learning model.

- In general you can allocate 80% of the dataset to training set and the remaining 20% to test set.We will create 4 sets— X_train (training part of the matrix of features), X_test (test part of the matrix of features), Y_train (training part of the dependent variables associated with the X train sets, and therefore also the same indices), Y_test (test part of the dependent variables associated with the X test sets, and therefore also the same indices.
- There are a few other parameters that we need to understand before we use the class:
- **test_size** — this parameter decides the size of the data that has to be split as the test dataset. This is given as a fraction. For example, if you pass 0.5 as the value, the dataset will be split 50% as the test dataset
- **train_size** — you have to specify this parameter only if you're not specifying the test_size. This is the same as test_size, but instead you tell the class what percent of the dataset you want to split as the training set.
- **random_state** — here you pass an integer, which will act as the seed for the random number generator during the split. Or, you can also pass an instance of the Random_state class, which will become the number generator. If you don't pass anything, the Random_state instance used by np.random will be used instead.
- Now split our dataset into a train set and test using train_test_split class from scikit learn library.



```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2, random_state = 0)
```

Advantages

- Activities of the company better than competitors.
- Advantages of the organization
- Unique resources and low cost resources
- Unique selling preposition of the company

Disadvantage

- Improvement that could be done
- Activities that can be avoided for cereal case
- Activities can be determined as your weakness in the market.
- Factors that can reduce the sales

Milestone 4: Model Building

Model building includes the following main tasks

- Import the model building Libraries
- Initializing the model
- Training and testing the model
- Evaluation of Model
- Save the Model

Training And Testing The Model

- Once after splitting the data into train and test, the data should be fed to an algorithm to build a model.

- There are several Machine learning algorithms to be used depending on the data you are going to process such as images, sound, text, and numerical values. The algorithms that you can choose according to the objective that you might have may be Classification algorithms or Regression algorithms.

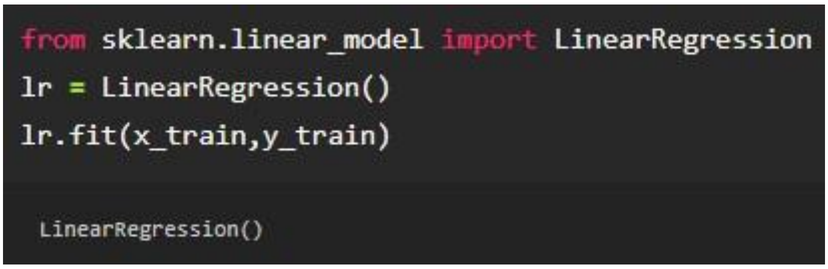
1.Linear Regression

2.Decision Tree Regressor

3.Random Forest Regressor

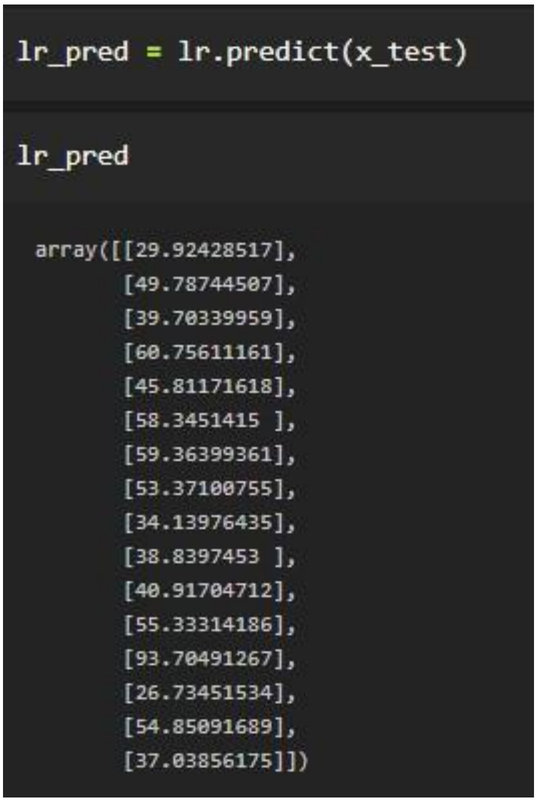
We’re going to use x_train and y_train obtained above in train_test_split section to train our decision tree regression model. We’re using the fit method and passing the parameters as shown below.

We are using the algorithm from Scikit learn library to build the model as shown below,



```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

```
LinearRegression()
```



```
lr_pred = lr.predict(x_test)
```

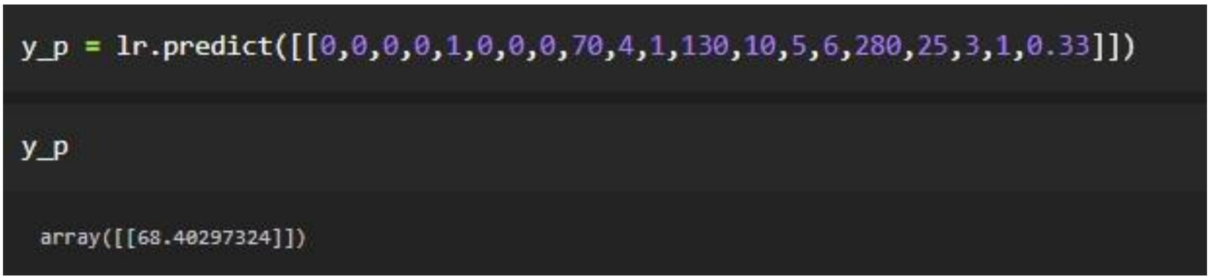
```
lr_pred
```

```
array([[29.92428517],
       [49.78744507],
       [39.70339959],
       [60.75611161],
       [45.81171618],
       [58.3451415 ],
       [59.36399361],
       [53.37100755],
       [34.13976435],
       [38.8397453 ],
       [40.91704712],
       [55.33314186],
       [93.70491267],
       [26.73451534],
       [54.85091689],
       [37.03856175]])
```

Once the model is trained, it’s ready to make predictions. We can use the **predict** method on the model and pass **x_test** as a parameter to get the output as **y_pred**.

Notice that the prediction output is an array of real numbers corresponding to the input array.

Testing the model with own values



```
y_p = lr.predict([[0,0,0,0,1,0,0,0,70,4,1,130,10,5,6,280,25,3,1,0.33]])
```

```
y_p
```

```
array([[68.40297324]])
```

Model Evaluation

After training the model, the model should be tested by using the test data which has been separated while splitting the data for checking the functionality of the model.

Regression Evaluation Metrics:

These model evaluation techniques are used to find out the accuracy of models built in regression type of machine learning models. We have three types of evaluation methods.

1. **R – Square (R2)**

Formula for calculating (R2) is given by:

R2= TSS-RSSTSS

- Total Sum of Squares (TSS) : TSS is a measure of total variance in the response/ dependent variable Y and can be thought of as the amount of variability inherent in the response before the regression is performed.
- Residual Sum of Squares (RSS) : RSS measures the amount of variability that is left unexplained after performing the regression.
- (TSS – RSS) measures the amount of variability in the response that is explained (or removed) by performing the regression

Where N is the number of observations used to fit the model, ?x is the standard deviation of x, and ?y is the standard deviation of y.

- R2 ranges from 0 to 1.
- R2 of 0 means that the dependent variable cannot be predicted from the independent variable
- R2 of 1 means the dependent variable can be predicted without error from the independent variable
- An R2 between 0 and 1 indicates the extent to which the dependent variable is predictable. An R2 of 0.20 means that 20 percent of the variance in Y is predictable from X; an R2 of 0.40 means that 40 percent is predictable; and so on.

2. Root Mean Square Error (RMSE)

RMSE tells the measure of dispersion of predicted values from actual values. The formula for calculating RMSE is

R2={1N*xi-meanx*yi-meany]/(x*y)}2

N : Total number of observations

Though RMSE is a good measure for errors, the issue with it is that it is susceptible to the range of your dependent variable. If your dependent variable has a thin range, your RMSE will be low and if the dependent variable has a wide range RMSE will be high. Hence, RMSE is a good metric to compare between different iterations of a model.

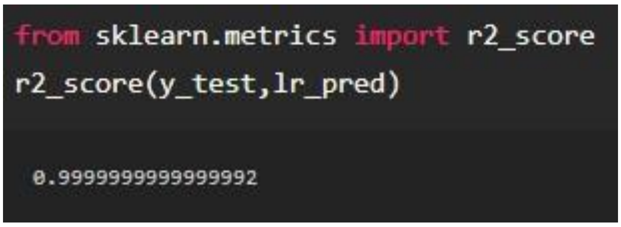
3. Mean Absolute Percentage Error (MAPE)?

To overcome the limitations of RMSE, analysts prefer MAPE over RMSE which gives error in terms of percentages and hence comparable across models. Formula for calculating MAPE can be written as:

RMSE= (YActual- YPredicted)2N

N : Total number of observations

For testing the model we use the below method,



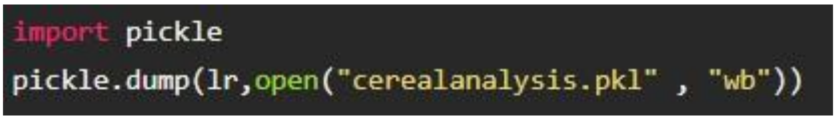
```
from sklearn.metrics import r2_score
r2_score(y_test,lr_pred)

 0.9999999999999992
```

Save The Model

After building the model we have to save the model.

Pickle in **Python** is primarily **used** in serializing and deserializing a **Python** object structure. In other words, it's the process of converting a **Python** object into a byte stream to store it in a file/database, maintain program state across sessions, or transport data over the network. wb indicates write method and rd indicates read method.

This is done by the below code



```
import pickle
pickle.dump(lr,open("cerealanalysis.pkl" , "wb"))
```

Conclusion

There is concern that agricultural production in developing countries will cause environmental threats in the future, as production will have to increase to satisfy the growing demand for food. Intensification leads to high inputs of nutrients in the form of mineral fertilizers and animal feed. Important parts of these inputs leak from the system in the form of nutrient leaching to groundwater and gaseous losses to the atmosphere.

There is a positive correlation between calories and sugars in cereal Most cereals do not have relatively high potassium values. Kellogg's offers the most cereals out of any manufacturer that are above the median calorie count (110). The more calories that a cereal has, the less likely it is to receive a high rating. Manufacturers that want to bring in high ratings should create cereals that are high in fiber, protein, and potassium and avoid creating cereals with high calorie counts or lots of sugar or fat. Cereals with high ratings are more likely to be placed on the first or third shelf, because that is generally where the consumers' eyes gravitate. Using a linear regression model can allow for accurate predictions of future cereal with less than ten percent error on average. For instance, a cereal that has thirteen grams of sugars, one-hundred and ten calories, and two grams of protein is projected to receive a rating of 32.03.

Future Scope

Cereals are of foods highly fortified with vitamins,minerals,and other nutritional additives.It is considered the most important meal of the day because it helps to revitalize the body,keeping it active and productive until the next meal. The demand for a full breakfast is increasing due to busy schedules and the need to improve consumers overtall nutritional needs.

Bibligraphy

https://www.marketdataforecast.com/market-reports/cereals-market

Breakfast Cereal Market Analysis Essay Example - PHDessay.com

The Cereal Industry - Law Essays (lawaspect.com)

Cereals Market | Size , Share, Growth, Value | 2022-2027 (marketdataforecast.com)

Milestone 5: Application Building

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.
This section has the following tasks

- Building HTML Pages
- Building server side script

Activity1: Building Html Pages:

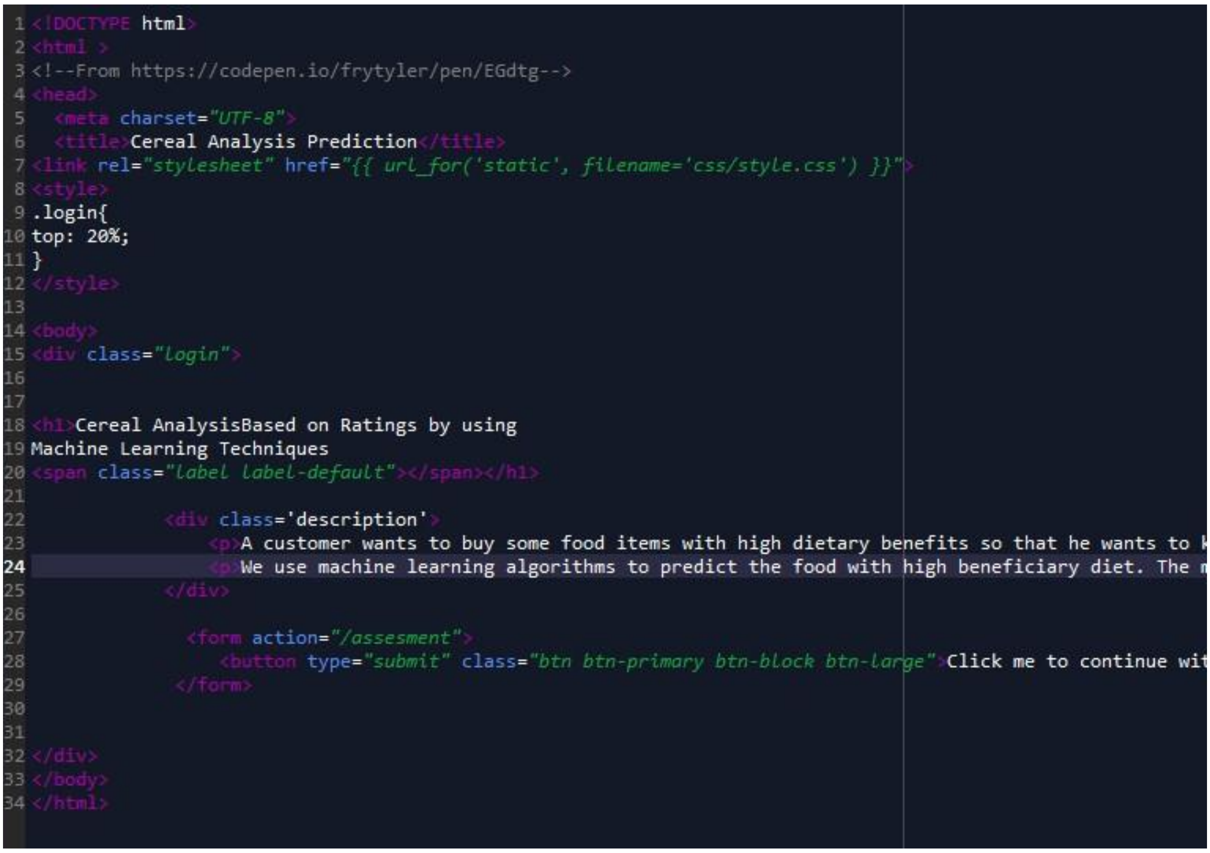
- In this HTML page, we will create the front end part of the web page. In this page we will accept input from the user and Predict the values.

  For more information regarding HTML

- In our project we have 3 HTML files ,they are

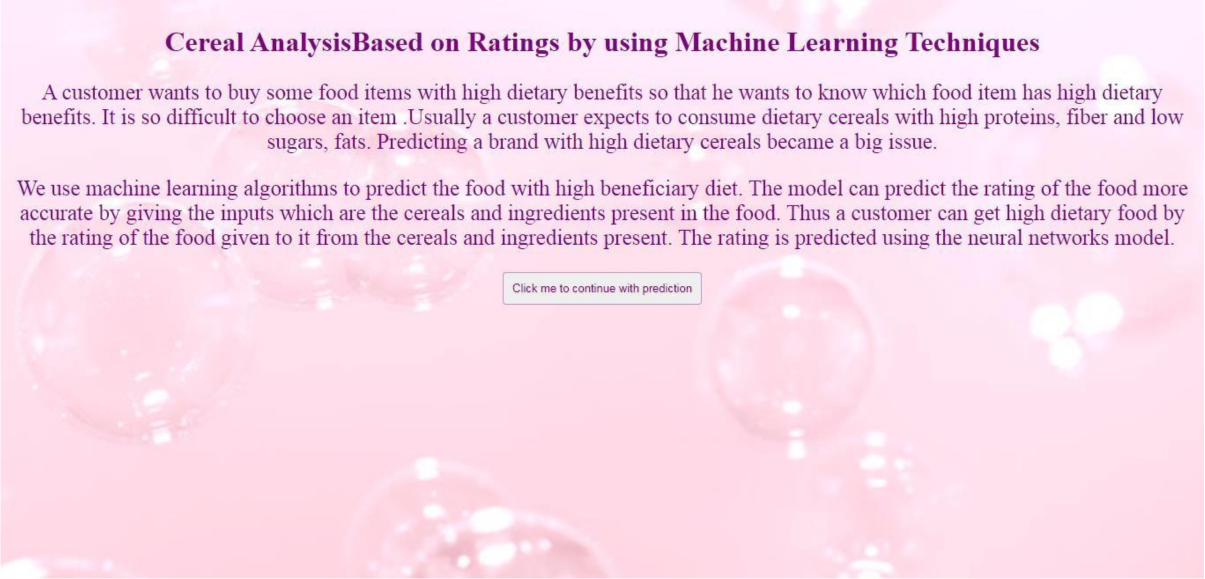
1. base.html
2. index.html 3. prediction.html

base.html



```html
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>Cereal Analysis Prediction</title>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
<style>
.login{
top: 20%;
}
</style>

<body>
<div class="login">


<h1>Cereal AnalysisBased on Ratings by using
Machine Learning Techniques
<span class="label label-default"></span></h1>

            <div class='description'>
                <p>A customer wants to buy some food items with high dietary benefits so that he wants to
                <p>We use machine learning algorithms to predict the food with high beneficiary diet. The
            </div>

              <form action="/assesment">
                 <button type="submit" class="btn btn-primary btn-block btn-large">Click me to continue wit
             </form>


</div>
</body>
</html>
```

The html page looks like

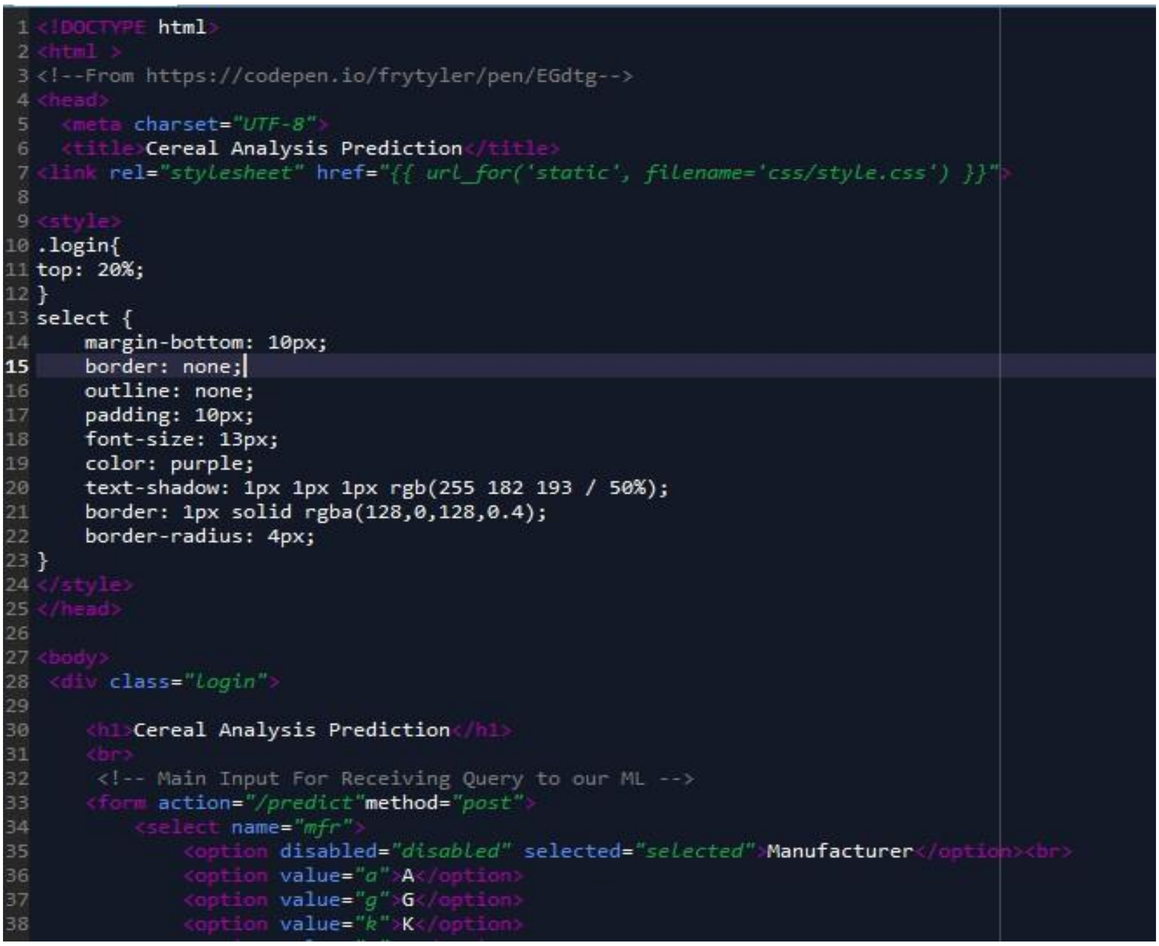


Cereal AnalysisBased on Ratings by using Machine Learning Techniques

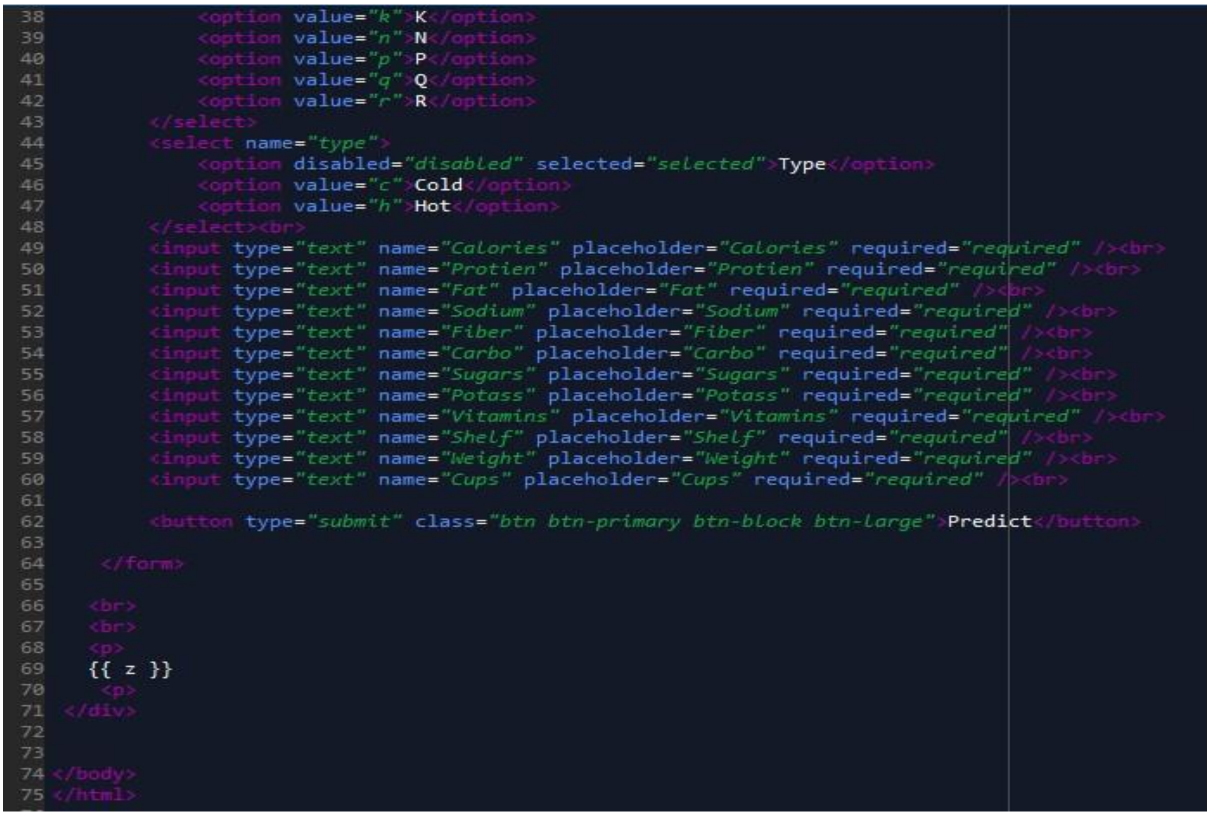
A customer wants to buy some food items with high dietary benefits so that he wants to know which food item has high dietary benefits. It is so difficult to choose an item .Usually a customer expects to consume dietary cereals with high proteins, fiber and low sugars, fats. Predicting a brand with high dietary cereals became a big issue.

We use machine learning algorithms to predict the food with high beneficiary diet. The model can predict the rating of the food more accurate by giving the inputs which are the cereals and ingredients present in the food. Thus a customer can get high dietary food by the rating of the food given to it from the cereals and ingredients present. The rating is predicted using the neural networks model.

index.html

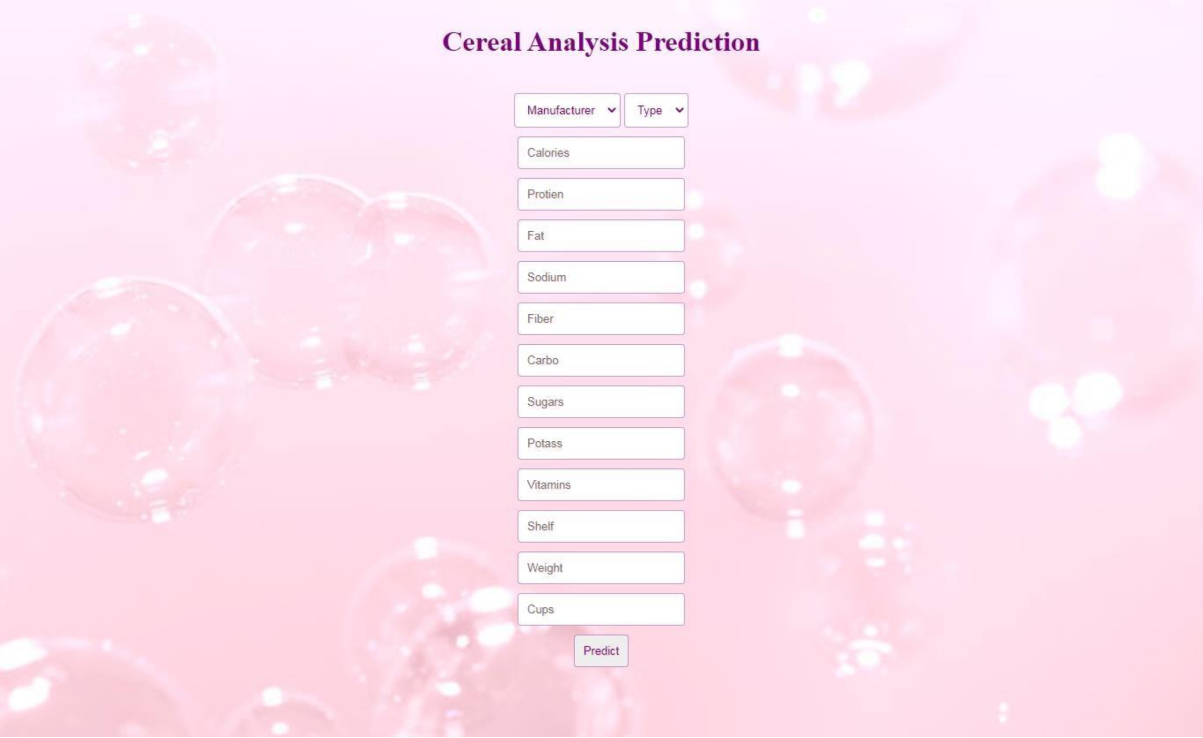


```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>Cereal Analysis Prediction</title>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">

<style>
.login{
top: 20%;
}
select {
    margin-bottom: 10px;
    border: none;
    outline: none;
    padding: 10px;
    font-size: 13px;
    color: purple;
    text-shadow: 1px 1px 1px rgb(255 182 193 / 50%);
    border: 1px solid rgba(128,0,128,0.4);
    border-radius: 4px;
}
</style>
</head>

<body>
 <div class="login">

    <h1>Cereal Analysis Prediction</h1>
    <br>
     <!-- Main Input For Receiving Query to our ML -->
    <form action="/predict"method="post">
        <select name="mfr">
            <option disabled="disabled" selected="selected">Manufacturer</option><br>
            <option value="a">A</option>
            <option value="g">G</option>
            <option value="k">K</option>
```

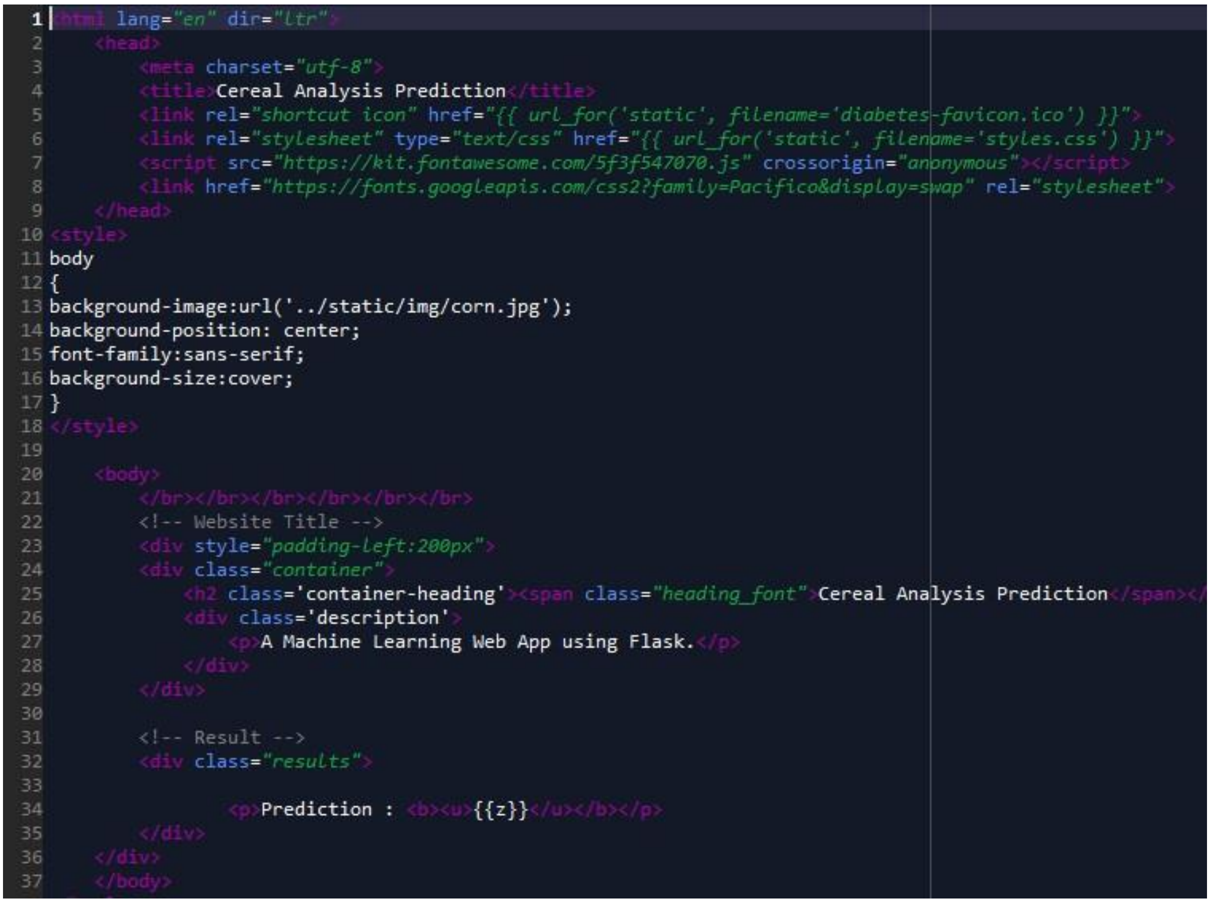


```
            <option value="k">K</option>
            <option value="n">N</option>
            <option value="p">P</option>
            <option value="q">Q</option>
            <option value="r">R</option>
        </select>
        <select name="type">
            <option disabled="disabled" selected="selected">Type</option>
            <option value="c">Cold</option>
            <option value="h">Hot</option>
        </select><br>
        <input type="text" name="Calories" placeholder="Calories" required="required" /><br>
        <input type="text" name="Protien" placeholder="Protien" required="required" /><br>
        <input type="text" name="Fat" placeholder="Fat" required="required" /><br>
        <input type="text" name="Sodium" placeholder="Sodium" required="required" /><br>
        <input type="text" name="Fiber" placeholder="Fiber" required="required" /><br>
        <input type="text" name="Carbo" placeholder="Carbo" required="required" /><br>
        <input type="text" name="Sugars" placeholder="Sugars" required="required" /><br>
        <input type="text" name="Potass" placeholder="Potass" required="required" /><br>
        <input type="text" name="Vitamins" placeholder="Vitamins" required="required" /><br>
        <input type="text" name="Shelf" placeholder="Shelf" required="required" /><br>
        <input type="text" name="Weight" placeholder="Weight" required="required" /><br>
        <input type="text" name="Cups" placeholder="Cups" required="required" /><br>

        <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>

    </form>

   <br>
   <br>
   <p>
   {{ z }}
    <p>
 </div>


</body>
</html>
```

The html page looks like

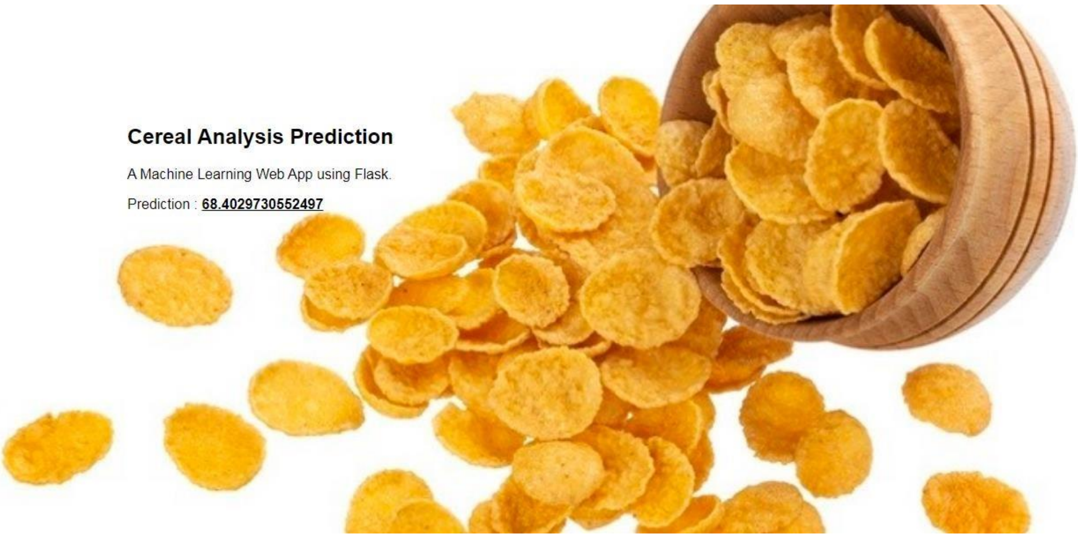


prediction.html



```html
<html lang="en" dir="ltr">
    <head>
        <meta charset="utf-8">
        <title>Cereal Analysis Prediction</title>
        <link rel="shortcut icon" href="{{ url_for('static', filename='diabetes-favicon.ico') }}">
        <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='styles.css') }}">
        <script src="https://kit.fontawesome.com/5f3f547070.js" crossorigin="anonymous"></script>
        <link href="https://fonts.googleapis.com/css2?family=Pacifico&display=swap" rel="stylesheet">
    </head>
<style>
body
{
background-image:url('../static/img/corn.jpg');
background-position: center;
font-family:sans-serif;
background-size:cover;
}
</style>

    <body>
        </br></br></br></br></br></br>
        <!-- Website Title -->
        <div style="padding-left:200px">
        <div class="container">
            <h2 class='container-heading'><span class="heading_font">Cereal Analysis Prediction</span></
            <div class='description'>
                <p>A Machine Learning Web App using Flask.</p>
            </div>
        </div>

        <!-- Result -->
        <div class="results">

                <p>Prediction : <b><u>{{z}}</u></b></p>
        </div>
    </div>
    </body>
```

The html page looks like



Cereal Analysis Prediction

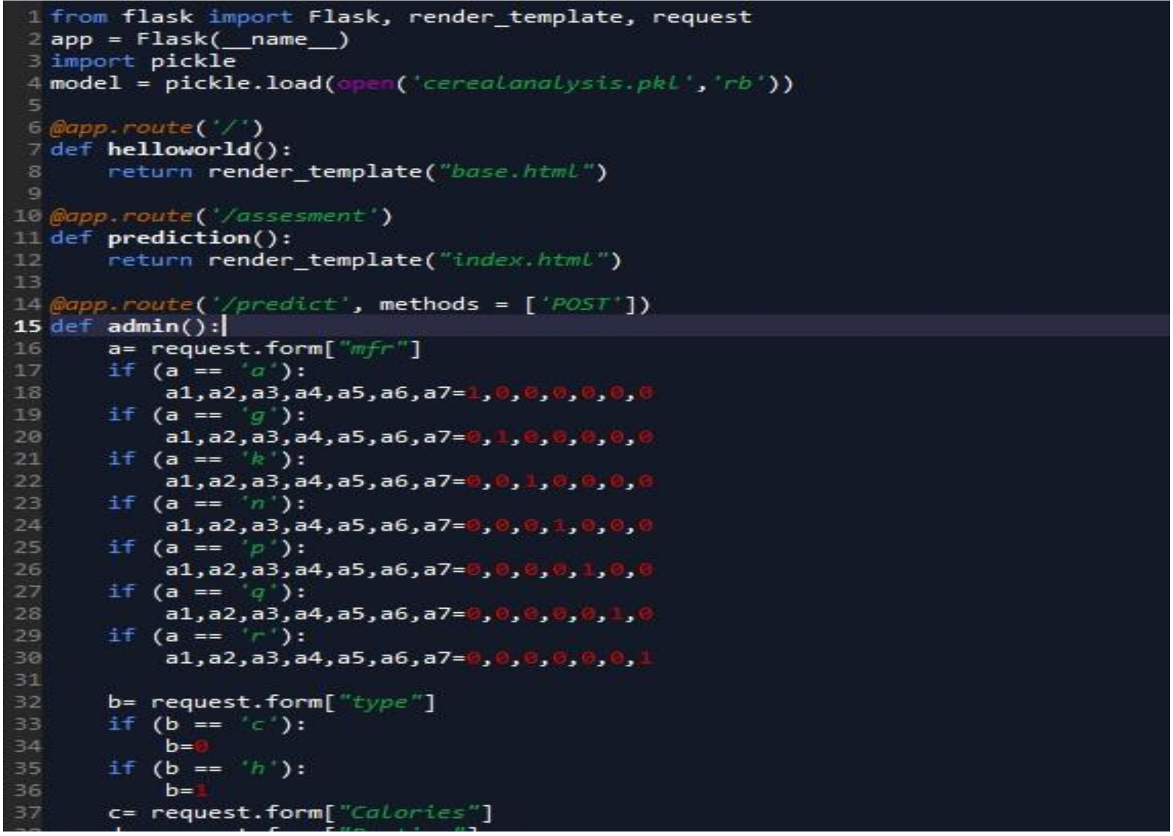
A Machine Learning Web App using Flask.

Prediction : **68.4029730552497**

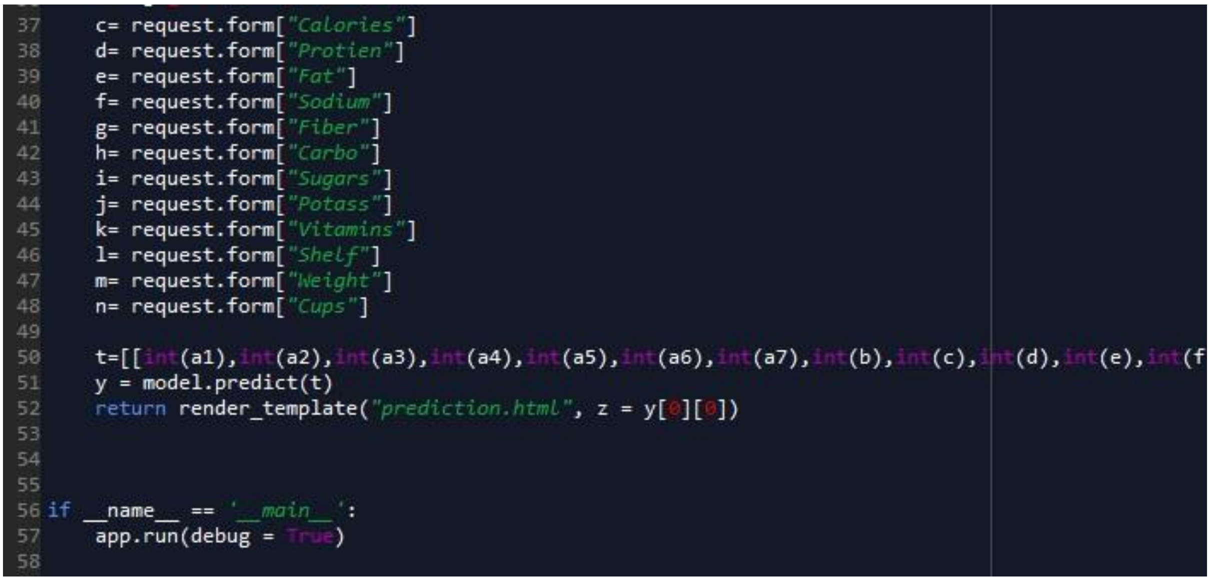
Activity 2: Main Python Script

Let us build an app.py flask file which is a web framework written in python for server-side scripting. Let's see the step by step procedure for building the backend application.

In order to develop web api with respect to our model, we basically use the Flask framework which is written in python.



```python
from flask import Flask, render_template, request
app = Flask(__name__)
import pickle
model = pickle.load(open('cerealanalysis.pkl','rb'))

@app.route('/')
def helloworld():
    return render_template("base.html")

@app.route('/assesment')
def prediction():
    return render_template("index.html")

@app.route('/predict', methods = ['POST'])
def admin():
    a= request.form["mfr"]
    if (a == 'a'):
        a1,a2,a3,a4,a5,a6,a7=1,0,0,0,0,0,0
    if (a == 'g'):
        a1,a2,a3,a4,a5,a6,a7=0,1,0,0,0,0,0
    if (a == 'k'):
        a1,a2,a3,a4,a5,a6,a7=0,0,1,0,0,0,0
    if (a == 'n'):
        a1,a2,a3,a4,a5,a6,a7=0,0,0,1,0,0,0
    if (a == 'p'):
        a1,a2,a3,a4,a5,a6,a7=0,0,0,0,1,0,0
    if (a == 'q'):
        a1,a2,a3,a4,a5,a6,a7=0,0,0,0,0,1,0
    if (a == 'r'):
        a1,a2,a3,a4,a5,a6,a7=0,0,0,0,0,0,1

    b= request.form["type"]
    if (b == 'c'):
        b=0
    if (b == 'h'):
        b=1
    c= request.form["Calories"]
```

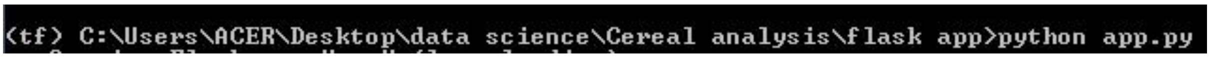


```python
    c= request.form["Calories"]
    d= request.form["Protien"]
    e= request.form["Fat"]
    f= request.form["Sodium"]
    g= request.form["Fiber"]
    h= request.form["Carbo"]
    i= request.form["Sugars"]
    j= request.form["Potass"]
    k= request.form["Vitamins"]
    l= request.form["Shelf"]
    m= request.form["Weight"]
    n= request.form["Cups"]

    t=[[int(a1),int(a2),int(a3),int(a4),int(a5),int(a6),int(a7),int(b),int(c),int(d),int(e),int(f
    y = model.predict(t)
    return render_template("prediction.html", z = y[0][0])



if __name__ == '__main__':
    app.run(debug = True)
```

Activity 3: Run the application

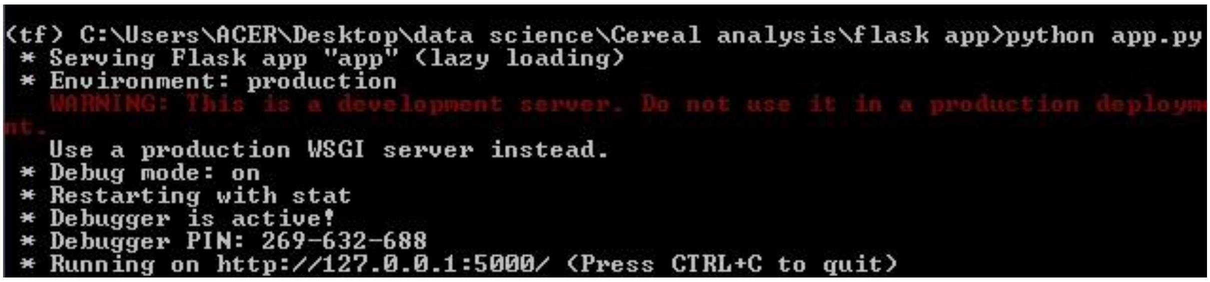
- Open anaconda prompt from the start menu
- Navigate to the folder where your python script is.
- Now type "python app.py" command

- Navigate to the localhost where you can view your web page



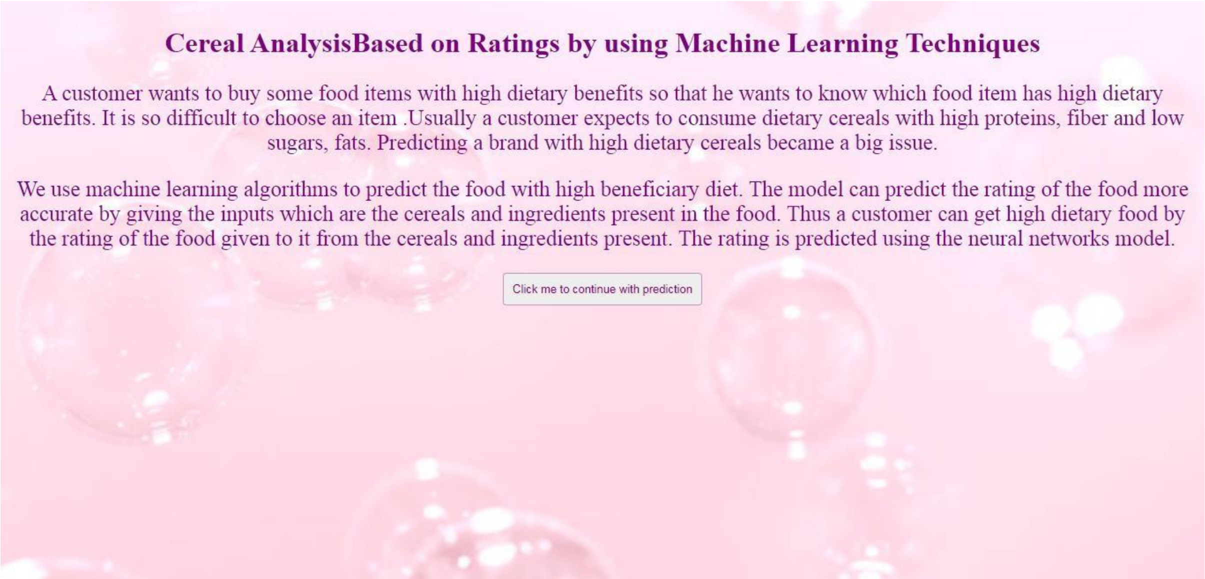
```
(tf) C:\Users\ACER\Desktop\data science\Cereal analysis\flask app>python app.py
```

Then it will run on **`localhost:5000`**



```
(tf) C:\Users\ACER\Desktop\data science\Cereal analysis\flask app>python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 269-632-688
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

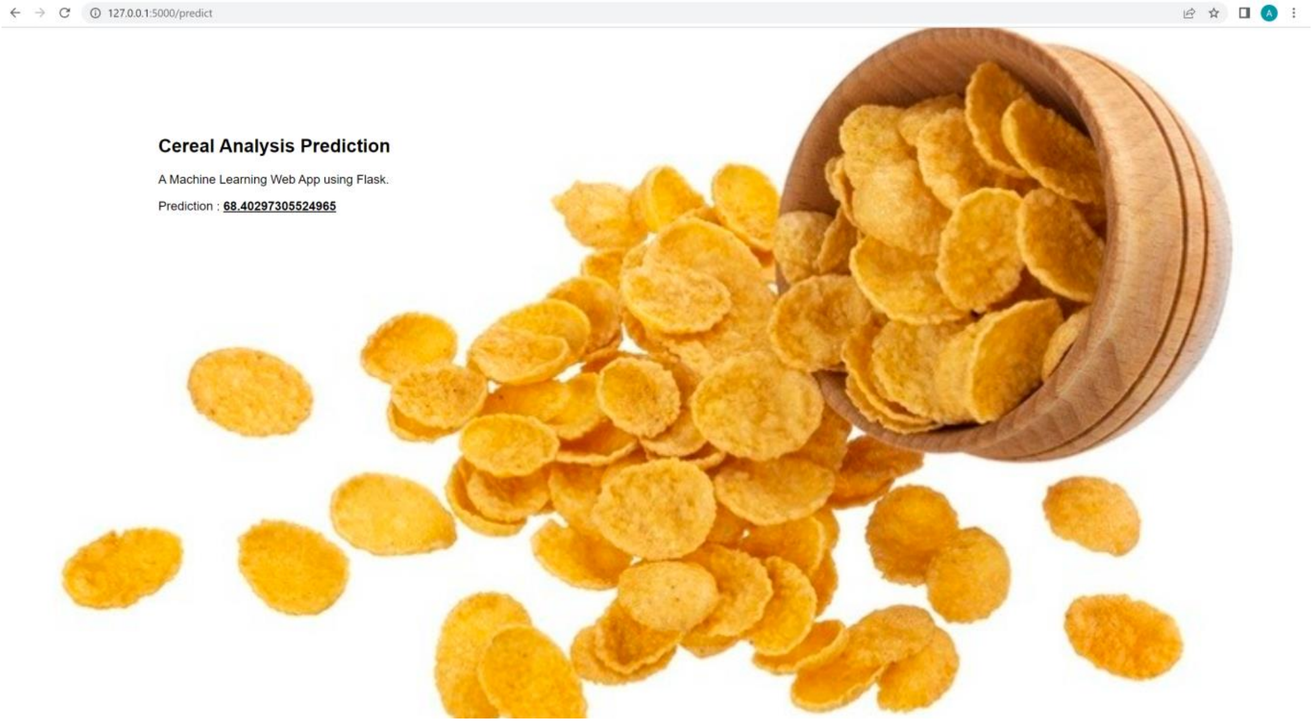
`Output Screenshots`:



Cereal AnalysisBased on Ratings by using Machine Learning Techniques

A customer wants to buy some food items with high dietary benefits so that he wants to know which food item has high dietary benefits. It is so difficult to choose an item .Usually a customer expects to consume dietary cereals with high proteins, fiber and low sugars, fats. Predicting a brand with high dietary cereals became a big issue.

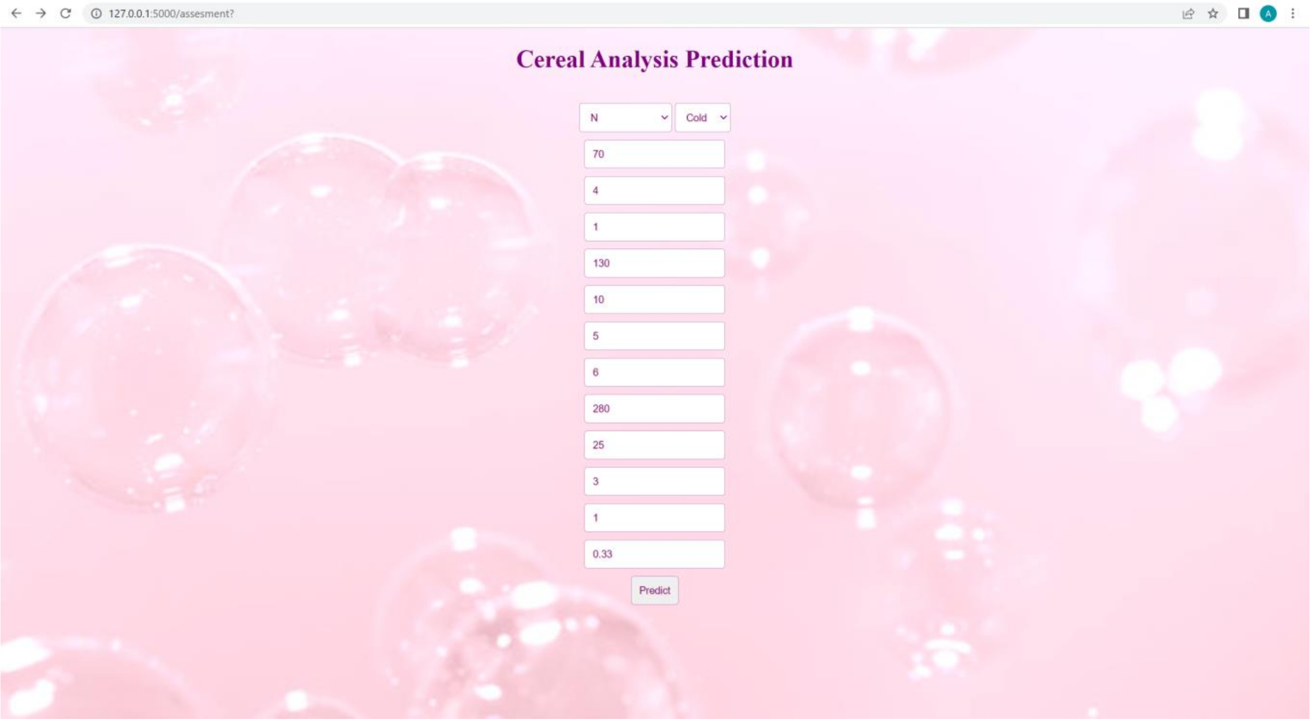
We use machine learning algorithms to predict the food with high beneficiary diet. The model can predict the rating of the food more accurate by giving the inputs which are the cereals and ingredients present in the food. Thus a customer can get high dietary food by the rating of the food given to it from the cereals and ingredients present. The rating is predicted using the neural networks model.



Cereal Analysis Prediction

A Machine Learning Web App using Flask.

Prediction : 68.40297305524965



127.0.0.1:5000/assesment?

Cereal Analysis Prediction

N Cold

70

4

1

130

10

5

6

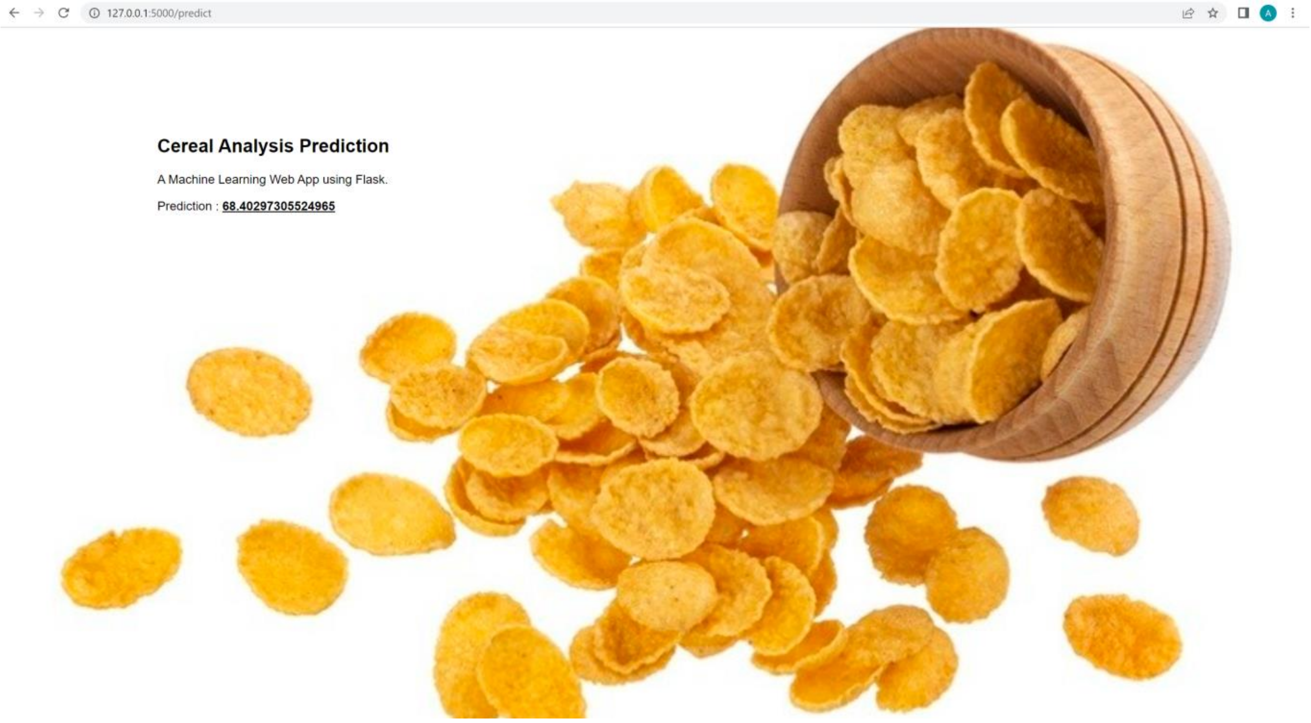
280

25

3

1

0.33



127.0.0.1:5000/predict

Cereal Analysis Prediction

A Machine Learning Web App using Flask.

Prediction : **68.40297305524965**