INTELLIGENT DIET CONSULTANT

Team Members:

16BCI0066 (Keshav Moorthy)
16BCE0783 (Daksh Bardia)

Report submitted for the
Final Project Review of

Course Code: CSE3013 – Artificial Intelligence

Slot: B1 + TB1

Professor: Dr. W.B.Vasantha

1. Introduction:

As people across the globe are becoming more interested in watching their weight, eating more healthy food and avoiding junk food, a system that can measure calories and nutrition in every day meals can be very useful for maintaining our health. Food calorie and nutrition measurement system is very beneficial for dietitians and patients to measure and manage the daily food intake. However, each person has a unique dietary pattern and have different health issues so a dietitian creates a meal plan depending on each case. A person in order to know its diet plan needs to give some information to the dietitian such as its body type, weight, height and its working hour details.

Smart phones and the Internet have revolutionized the communication and with it the lifestyle of people. An increasing number of smart phones and Personal Digital Assistants (PDA) allow people to access the Internet where ever they are and whenever they want. By using Internet they can obtain on one hand information on almost everything they wish to. Therefore, just by using smart phones user can get diet assistance anytime at free of cost. Artificial dietitian is an application with artificial intelligence about human diets. It acts as a diet consultant similar to a real dietician. This system acts in a similar way as that of a dietitian. A person in order to know his/her diet plans needs to give some information to the dietitian such as its body type, weight, height, and working hour details.

Now a days, human beings are suffering from many health problems such as fitness problem, maintaining proper diet problem, etc. Therefore, we are developing this website for providing special dietician information and proper exercise knowledge for people. The proposed system is an android application which contains the knowledge and data regarding the fitness of a person. It acts as a diet consultant similar to a real dietician. Dieticians are educated with nutrient value of foods. A dietician consults a person based on his schedule, body type, height and weight. The system too asks all this data from the user and processes it. It asks about how many hours the user works, his height, weight, age etc. The system stores and processes this data and then calculates the nutrient value needed to fill up user's needs. The effective personal dietary guidelines are very essential for managing our health, preventing chronic diseases and the interactive diet planning helps a user to adjust the plan in an easier way. The android app is to be produced on Artificial Intelligence base. This android application also suggests the user to what to do for example diet tips, Exercises, Online Training, etc.

2. Literature Review Summary Table

Authors and Year (Reference)	*Title (Study)*	*Concept / Theoretical model/ Framework*	*Methodology used/ Implementation*	*Relevant Finding*	*Limitations / Future Research/ Gaps identified*
Hitesh Pruthi,Hardik Parvadiya,Varun Rawool,Joel Philip, 2017	*Artificial Intelligence Dietician*	*Supervised Learning*	*Web Application*	*Allows the user to find customized diet plan*	*Better Workout Plan*
Talapanty Shwetha, Vangari Swetha , Singh Deepali, Gaonkar Vaishnavi, Prof Shrikant Sanas, 2017	*Artificial Intelligence Dietitian*	*Supervised Learning*	*Web Application*	*Better prediction for diet*	*Improved Application Interface*
Al-Thuhli, Balqees & Samir Al-Gadidi, Basma & Al-Alawi, Halima & Al-Busaidi, Kamla Ali. (2013).	*Developing a Nutrition and Diet Expert System Prototype*	*Classification and recommendation*	*Only Algorithm*	*Using dataset to predict the data.*	*More Efficient Algorithm*
GRACE J PETOT, CYNTHIA MARLING, LEON STERLING	*An Artificial Intelligence System for Computer-Assisted Menu Planning*	*Neural Network*	*Proposed Algorithm*	*BMI Calculation and recommendation*	*Better diet plan suggestions*

3. Objective of the project:

The objective of our project is to develop an intelligent diet consultant powered by Artificial Intelligence which uses forward propagation neural network. This intelligence diet consultant can be used by various people who want to manage their weight with the help of food they intake, whether it be increasing weight, maintain weight or decrease weight.

4. Innovation component in the project:

In the existing AI Diet Consultant system, you have to hire a dietitian in order to get advice. Hiring a nutrition doctor will not only waste your time and efforts for calling them, going to them and so on but also cost you very high as their charges per month are very high. With our developed system, a person can use the intelligent diet system anytime he or she wants and create a goal to achieve for the desired weight. We have used forward propagation neural network to create our program.

5. Work done and implementation

Methodology adapted:

- Using the following Kaggle dataset which provides a large variety of recipes we build the model for our neural network: https://www.kaggle.com/kaggle/recipe-ingredients-dataset. The test set containing recipes id, and list of ingredients

- Python3 along with tensorflow is used for building and training the classifier

- Tensorflow for android is used for transferring the model to an android environment. TensorFlow is a free and open-source software library for Machine Learning tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

- Forward Propagation Neural Network

  In neural networks, forward propagation is done to get the output and compare it with the real value to get the error. Now, to minimize the error, propagate backwards by finding the derivative of error with respect to each weight and then subtracting this value from the weight value.

  The basic learning that has to be done in neural networks is training neurons when to get activated. Each neuron should activate only for particular type of inputs and not all inputs. Therefore, by propagating forward one can see how well the neural network is behaving and find the error. After finding out that the network has error, back propagate and use a form of gradient descent to update new values of weights. Then, the network will again forward propagate to see how well those weights are performing and then will backward propagate to update the weights. This will go on until the iterations reach some minima for error value.

  The input X provides the initial information that then propagates to the hidden units at each layer and finally produce the output Y. The architecture of the network entails determining its depth, width, and activation functions used on each layer. Depth is the number of hidden layers. Width is the number of units (nodes) on each hidden layer since we don't control neither input layer nor output layer dimensions. There are quite a few set of activation functions such Rectified Linear Unit, Sigmoid, Hyperbolic tangent, etc. Research has proven that deeper networks outperform networks with more hidden units.

Dataset used :

a. Where from you are taking your dataset?
   https://www.kaggle.com/kaggle/recipe-ingredients-dataset
   The test set containing recipes id, and list of ingredients

b. Is your project based on any other reference project (Stanford Univ. or MIT)?
   No

c. How does your project differ from the reference project?
   No

Tools used

- Tensorflow
- Python Integrated Development Environment (IDE)
- Kaggle Dataset

Screenshot and Demo:

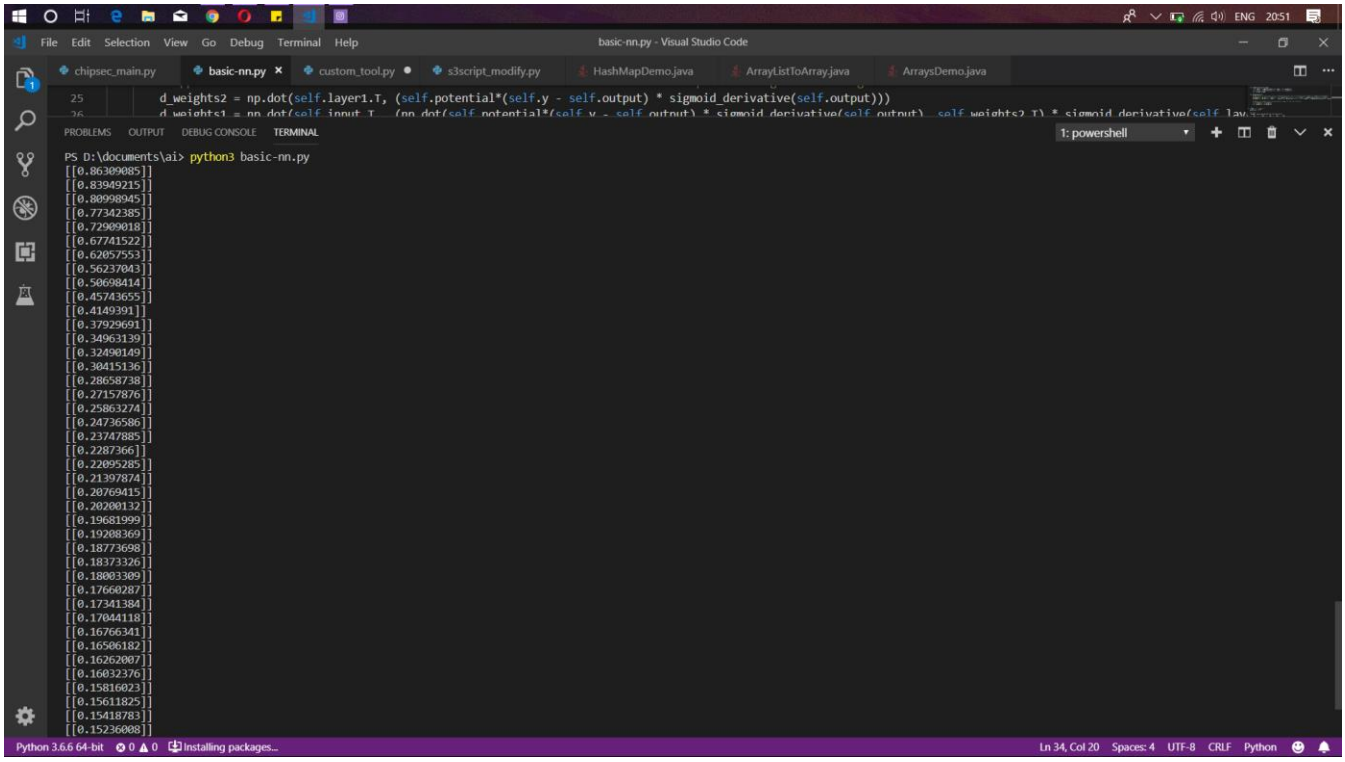
Code:

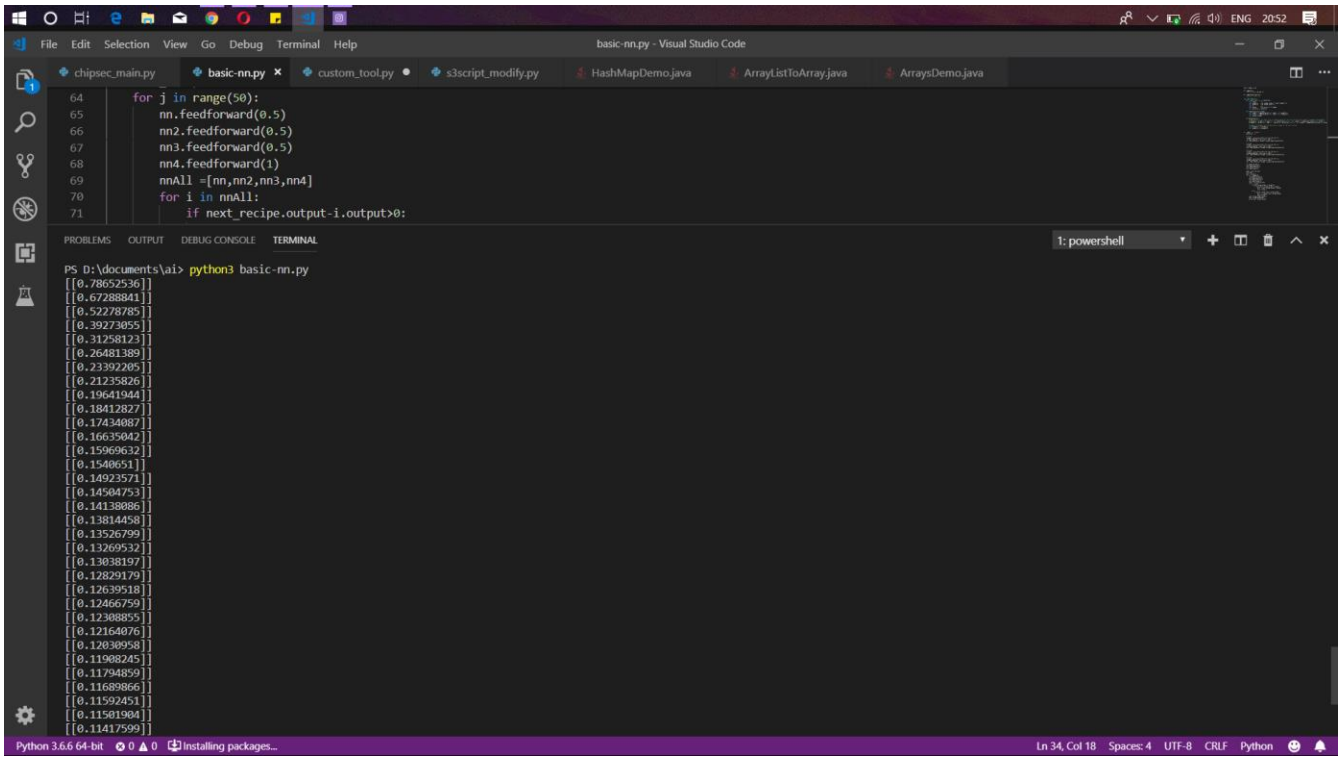
```
import numpy as np


def sigmoid(x):
    return 1.0/(1+ np.exp(-x))


def sigmoid_derivative(x):
    return x * (1.0 - x)


class NeuralNetwork:
    def __init__(self, x, y, potential):
        self.input      = x
        self.weights1   = np.random.rand(self.input.shape[1],4)
        self.weights2   = np.random.rand(4,1)
        self.y          = y
        self.output     = np.zeros(self.y.shape)
        self.potential = potential


    def feedforward(self,alpha):
        self.layer1 = sigmoid(np.dot(self.input, self.weights1))
        self.output = sigmoid(np.dot(self.layer1, self.weights2))
        self.output*=alpha


    def backprop(self):
        # application of the chain rule to find derivative of the loss function with respect to
weights2 and weights1
```

```
        d_weights2 = np.dot(self.layer1.T, (self.potential*(self.y - self.output) * sigmoid_derivative(self.output)))
        d_weights1 = np.dot(self.input.T,  (np.dot(self.potential*(self.y - self.output) * sigmoid_derivative(self.output), self.weights2.T) * sigmoid_derivative(self.layer1)))

        # update the weights with the derivative (slope) of the loss function
        self.weights1 += d_weights1
        self.weights2 += d_weights2


if __name__ == "__main__":
    potential = 0.15

    #recipe1:
    recipe1 = np.array([[0.8,0.4,0.1,0.3,0.9,0.7]])
    recipe1_expected = np.array([[.100]])
    nn = NeuralNetwork(recipe1,recipe1_expected,potential)

    #recipe2:
    recipe2 = np.array([[0.8,0.4,0.1,0.3,0.9,0.7]])
    recipe2_expected = np.array([[.100]])
    nn2 = NeuralNetwork(recipe2,recipe2_expected,potential)

    #recipe3:
    recipe3 = np.array([[0.8,0.4,0.1,0.3,0.9,0.7]])
    recipe3_expected = np.array([[.100]])
    nn3 = NeuralNetwork(recipe3,recipe3_expected,potential)
```

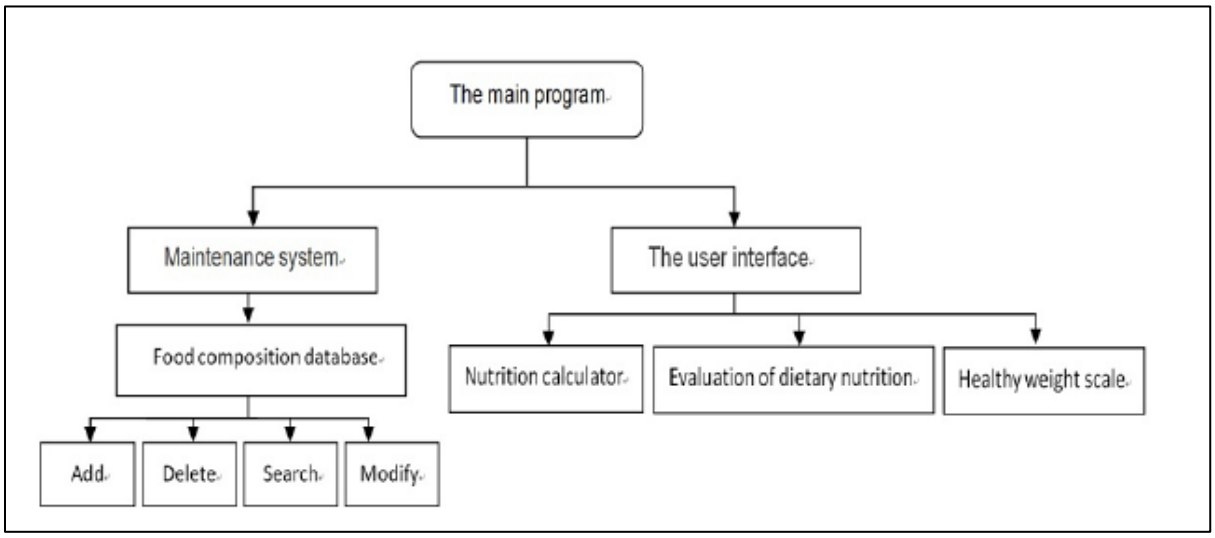
```
#recipe4:
recipe4 = np.array([[0.8,0.4,0.1,0.3,0.9,0.7]])
recipe4_expected = np.array([[.100]])
nn4 = NeuralNetwork(recipe4,recipe4_expected,potential)

nn.feedforward(0.5)
nn2.feedforward(0.5)
nn3.feedforward(0.5)
nn4.feedforward(1)

nnAll =[nn2,nn3,nn4]
diff=100
next_recipe=nn
for j in range(150):
    nn.feedforward(0.5)
    nn2.feedforward(0.5)
    nn3.feedforward(0.5)
    nn4.feedforward(1)
    nnAll =[nn,nn2,nn3,nn4]
    for i in nnAll:
        if next_recipe.output-i.output>0:
            if diff>next_recipe.output-i.output:
                diff = next_recipe.output-i.output
                next_recipe=i
        else:
            if diff>i.output-next_recipe.output:
                diff = i.output-next_recipe.output
```

```
        next_recipe=i
next_recipe.feedforward(1)
next_recipe.backprop()
print(next_recipe.output)
```





6. Results and discussion



As seen in the screenshot, the values are given output for the to different potential values of 0.75 and 2 respectively. It can be seen that the values are gradually declined to the desired value for given food parameters as input. The results obtained were in correspondence to the objective of the program.

7. References

1. Raman spectroscopy for determining nutritional facts By Moustakas, C. Dept. of Electr. & Comput.Eng., Univ. of Cyprus, Nicosia, Cyprus Pitris, C. and E-ISBN : 978-1-4244-5379-5; Print ISBN: 978-1- 4244-5379-5; INSPEC Accession Number: 11102584.
2. FOODS: A Food-Oriented Ontology-Driven System bySnae, C. Dept. of Comput. Sci. & Inf. Technol., Naresuan Univ., Phitsanulok Bruckner, M. and E-ISBN : 978-1-4244-1490-1;Print ISBN: 978-1-4244- 1489-5; INSPEC Accession Number: 10287294.
3. Al-Thuhli, Balqees & Samir Al-Gadidi, Basma & Al-Alawi, Halima & Al-Busaidi, Kamla Ali. (2013). Developing a Nutrition and Diet Expert System Prototype.
4. 'I had to lose weight for anorexia treatment', by Awful Spirale to lose, maintain weight, by Elizabeth Sommerfeld.
5. Jul 10, 2014 - Measuring Calorie and Nutrition from Food Image by Parisa Pouladzadeh, Shervin Shirmohammadi And Rana Almaghrabi and ISSN: 0018-9456; INSPEC Accession Number: 14432032.
6. Talapanty Shwetha, Vangari Swetha, Singh Deepali, Gaonkar Vaishnavi, Prof Shrikant Sanas, "Artificial Intelligence Dietitian Using Android", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 2, Issue 2, pp.713-715, March-April-2017.