UNLV | UNIVERSITY LIBRARIES

UNLV Theses, Dissertations, Professional Papers, and Capstones

May 2016

A Web Based User Interface for Machine Learning Analysis of Health and Education Data

Chandani Shrestha
University of Nevada, Las Vegas

Follow this and additional works at: https://digitalscholarship.unlv.edu/thesesdissertations

 Part of the Computer Sciences Commons

Repository Citation

Shrestha, Chandani, "A Web Based User Interface for Machine Learning Analysis of Health and Education Data" (2016). *UNLV Theses, Dissertations, Professional Papers, and Capstones*. 2745.
http://dx.doi.org/10.34917/9112192

This Thesis is protected by copyright and/or related rights. It has been brought to you by Digital Scholarship@UNLV with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you need to obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself.

This Thesis has been accepted for inclusion in UNLV Theses, Dissertations, Professional Papers, and Capstones by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.

A WEB BASED USER INTERFACE FOR MACHINE LEARNING ANALYSIS OF HEALTH AND EDUCATION DATA

by

Chandani Shrestha

Bachelor in Computer Engineering (B.E.)
Kathmandu University, Nepal
2012

A thesis submitted in partial fulfillment of
the requirements for the

Master of Science in Computer Science

Department of Computer Science
Howard R. Hughes College of Engineering
The Graduate College

University of Nevada, Las Vegas
May 2016

© Chandani Shrestha, 2016
All Rights Reserved

UNLV | GRADUATE COLLEGE

Thesis Approval

The Graduate College
The University of Nevada, Las Vegas

April 15, 2016

This thesis prepared by

Chandani Shrestha

entitled

A Web Based User Interface for Machine Learning Analysis of Health and Education Data

is approved in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science
Department of Computer Science

Fatma Nasoz, Ph.D.
Examination Committee Chair

Kathryn Hausbeck Korgan, Ph.D.
Graduate College Interim Dean

Justin Zhan, Ph.D.
Examination Committee Member

Kazem Taghva, Ph.D.
Examination Committee Member

Qing Wu, Ph.D.
Graduate College Faculty Representative

Abstract

The objective of this thesis is to develop a user friendly web application that will be used to analyse data sets using various machine learning algorithms. The application design follows human computer interaction design guidelines and principles to make a user friendly interface [Shn03]. It uses Linear Regression, Logistic Regression, Backpropagation machine learning algorithms for prediction. This application is built using Java, Play framework, Bootstrap and IntelliJ IDE. Java is used in the backend to create a model that maps the input and output data based on any of the above given learning algorithms while Play Framework and Bootstrap are used to display content in frontend. Play framework is used because it is based on web-friendly architecture. As a result it uses predictable, minimal resources (CPU, memory, threads) for highly scalable applications. It is also developer friendly where changes can be made in the code and hitting the refresh button in browser will update the interface. Bootstrap is used to style the web application and it adds responsiveness to the interface with added feature of cross-browser compatible designs. As a result, the website is responsive and fits the screen size of computer.

Using this web application users can predict features, category of the entity in the data sets. User needs to submit data set where each row in the data set must represent attributes of the entity. Once data is submitted the application builds a model using user selected machine learning algorithm logistic regression, linear regression or backpropagation. After the model is developed in second stage of the application user can submit attributes of the entity whose category needs to predicted. The predicted category will be displayed on screen in third stage of the application. The interface of the application shows its current active stage. These models are built using 80% of submitted dataset and remaining 20% is used to test the accuracy of the application. In this thesis, prediction accuracy of each algorithm is tested using UCI breast cancer data sets. When tested on breast cancer data with 10 attributes both Logistic Regression and Backpropagation gave 98.5% accuracy. And when tested on breast cancer data with 31 attributes Logistic Regression gave 92.85% accuracy and Backpropagation gave 94.64%.

Acknowledgements

I am grateful to my advisor Dr. Fatma Nasoz for supporting in research and providing resources. She is always there for advice and working with her has made me feel, I can start a project from scratch in completely a new field and complete it through research. It is her constant support and faith on me to make this happen.

I would like to thank Dr. Ajoy Datta and Dr. Laxmi P. Gewali for their advice and care.

I am very much thankful to Dr. Evangelos Yfantis and Dr. Kazem Taghva whose classes have motivated me to do further research in data analysis and provided useful information to carry out the research.

I would also like to thank the thesis committee member Dr. Justin Zhan and Dr. Qing Wu for their support and understanding.

And above all I would like to deeply thank my parents Sarita and Nirmal Shrestha, sister Roshni Shrestha and Shreedhan Shrestha.

Chandani Shrestha

University of Nevada, Las Vegas
May 2016

Table of Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction and Motivation

The purpose of this thesis is to develop a user friendly web application that follows human computer interaction design guidelines and principles [Shn03]. This application is used to analyse patterns and predicts an attribute of an entity in a dataset. To use this application for prediction it must be trained with a dataset. The interface of an application facilitates user to upload datasets that will be used to train the application. This dataset should be about an entity which needs to be analysed or whose attribute needs to be predicted. Each column of an uploaded dataset should represent an attribute and each row describes an entity.

This application can only be used to analyse datasets have data stored in above given format. After the application is trained using the uploaded dataset user needs to submit a set of attributes representing an entity whose category or attribute needs to be predicted. The accuracy % and predicted value for a submitted entity is shown in last stage of the application. Each interface of the application informs user of current, previous and upcoming stage of the application.

Accuracy of the application is based on the how likely it makes correct prediction for unseen test datasets which have not been used for training. If accuracy of the model is high then it conveys that there is a high probability that the predicted value matches the target value.

This application can be used for classification problems e.g. to classify if the tumor is cancerous or not, predict how many bicycles will be rented, e.t.c. The only requirement of the application is each training data instance must have an output value. Although, the accuracy given by the applications is not always 100%. It can give accuracy range of $80\% - 100\%$ depending on the parameters used to train the model. As accuracy of the model increases likelihood of getting matching predicted and target value also increases.

The application is build using Supervised Machine Learning Algorithm [Ng], Java, Play Framework and Bootstrap. The list of supervised machine learning algorithms used to build a predictive model are:

1. Linear Regression [Ng]
2. Logistic Regression [Ng]
3. Backpropagation [Ng]

In supervised machine learning algorithm each data instance used for training has both input and output value e.g. a data which is being analysed for breast tumor has a set of features describing the tumor and an attribute stating if the tumor is cancerous or not.

The interface of the application follows Human Computer Interaction Design Rules [Shn03] to make it user-friendly. The interface is more focused towards novice and intermediate users. Following are the list of implemented design rules:

1. Consistency

   It is better to use upto three fonts, four sizes and four standard colors [Shn03]. Font size, color of text, button, input box size, alignment of content, background color, margin, organization of content is consistent. Similar sequence of actions are used to predict outcome in all three different algorithms. The flow of entering inputs and flow of moving from one page to next page consistent is also consistent in all the interfaces of the given algorithms. Headers and footers are consistent in all interfaces of the application.

2. Simple Layout

   Contents of the application are short and descriptive so that user don't skip or get annoyed with lengthy information. A simple design is more appealing and loads faster [RP04]. Web application should give access to the information not abundance of information [RP04]. Color blinking can sometime distract users [Shn03] so, it is avoided in this application. Simple well organized display is used where left section of the interface shows stage of the application and right side shows information related to current stage of the application. Cluttered display should be avoided [Shn03] so, it is avoided in this application.

3. Reduce short term memory load

   User don't have to memorize steps to get to Result interface that displays the predicted value. The progress towards prediction is displayed in left side of each interface in "Steps" section. Steps are listed in ascending order in which they will be executed. It conveys user the step they are currently in, previous step which they have completed and the upcoming step.

4. Error Prevention and Error Checking

   Each input box has a placeholder with an example. Also, error checking is done to prevent users from submitting missing or invalid information. Figure 1.3 shows its implementation.

5. Compatibility of data entry and data display

   The format of data entered as input to the application should be compatible with the format of data displayed as output [Shn03]. In this application the format of predicted value is similar to the format of value in the column representing dependent variable(output column) of data file.

Below given figure 1.1 is a home screen of the web application. To start analysing a data user needs to click on "Get Started" button.

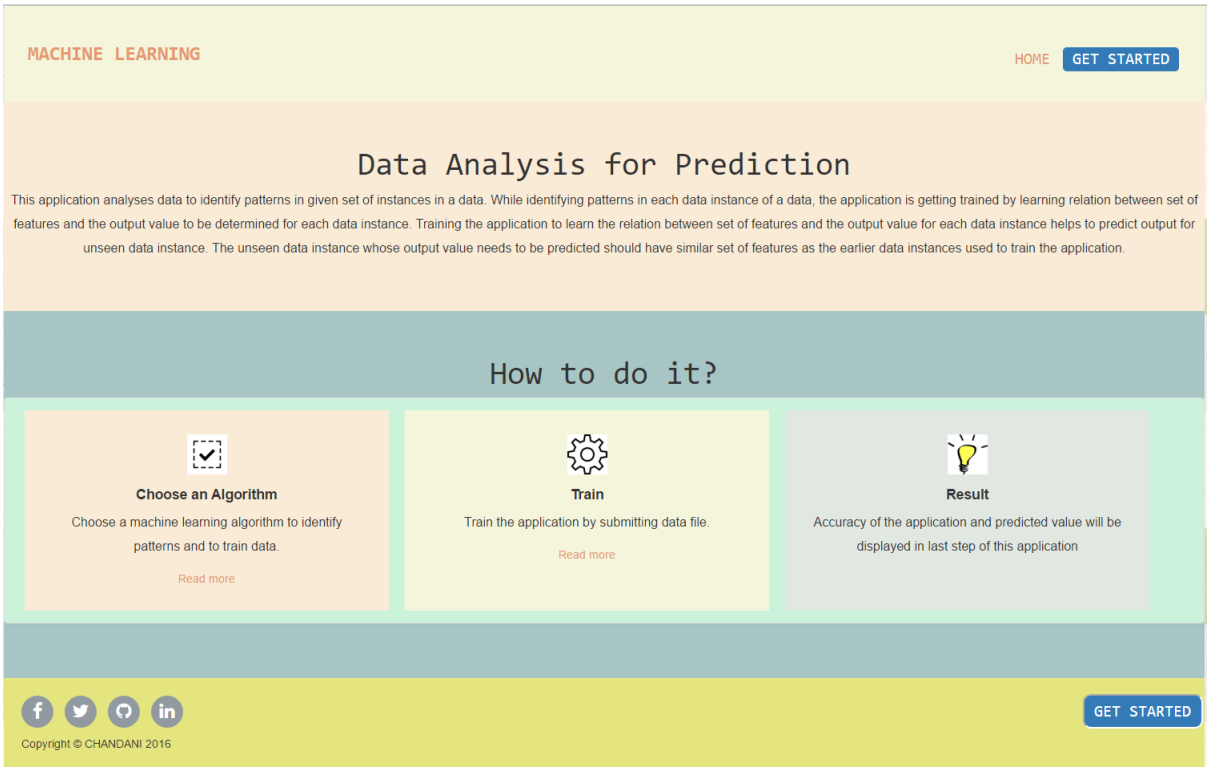

Figure 1.1: Cropped Figure of Home page

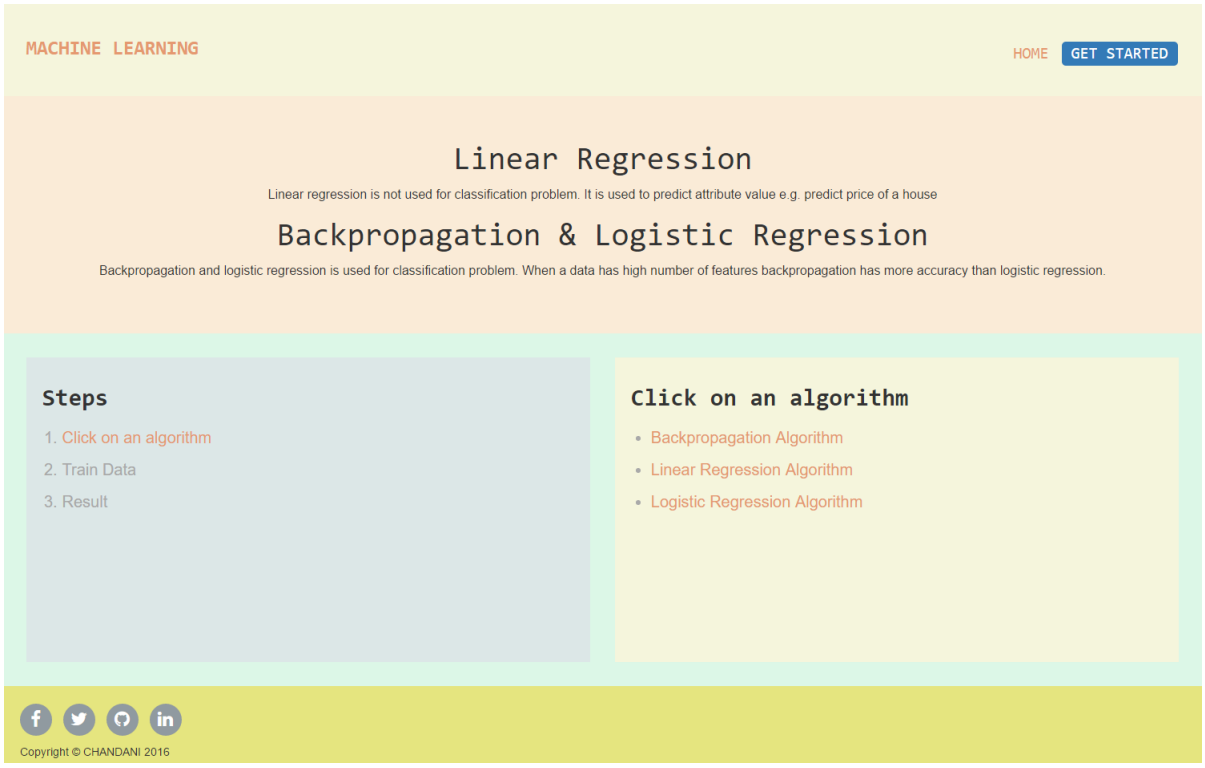

Figure 1.2: Content displayed after clicking Get Started button

Below given figure 1.3 is an interface of Backpropagation learning algorithm. The interfaces of other algorithms are similar to the interface shown in figure 1.3.

User can upload data which needs to be analysed in second stage of the application. Here, in this application both logistic regression and backpropagation are used for classification problems and linear regression is used to predict attribute value of an entity. The predicted value is shown in last stage of the application.

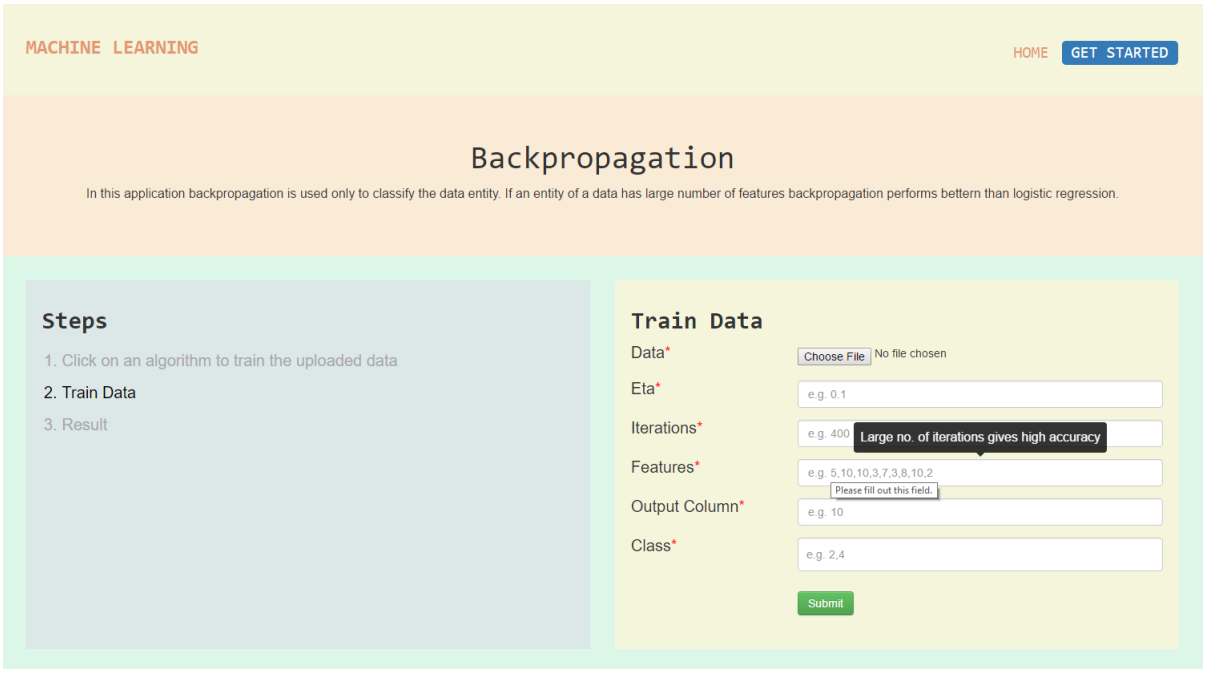

Figure 1.3: Content Displayed after clicking 'Backpropagation Algorithm' in figure 1.2

1.1 Motivation

An application can either make tasks easier or frustrate users [Gar10]. Each application delivers an experience. A functionality of a system might be complex but it should be easy to use [Gar10] e.g. the functions inside a telephone is complex but its interface makes it simple to use. A good user interface increases good experience which leads to good business [Gar10]. A good content of the interface retains its users [RP04]. Web users experience is more important [Gar10] because it might make them feel stupid about not knowing the technology [Gar10] but, its not the fault of the user if an application doesn't work [Gar10]. A simple design of the application appeals the user [RP04]. So, this thesis is focused in making a user-centred-design for machine learning application.

Machine learning and pattern recognition is used in health care and biomedical community to detect and diagonalise disease [Saj06]. As well as, it helps physicians and other health care professionals in decision making by giving narrow descriptive information about the data to be analysed [Saj06]. It is also used in analysing and detecting patterns in data [Alp14], interpret the patterns, deal with missing or imperfect data and make prediction [MP01].

Physicians can compare health data of a patient with past health datasets of other patients having same disease or health characteristics [Saj06]. The predicted outcome gives likelihood of the patient being affected by the disease [Saj06][RTB67]. Depending on the accuracy of the model physician can examine further which

can save both time and money. Physician don't have to start from the very beginning stage as machine learning tools can be used to study patterns in health of the patient [Saj06].

Machine learning is also used in education to understand students study patterns, scoring patterns, reasons for low score, financial condition of student, e.t.c. [Alp14]. Studying these patterns educational institutions can facilitate students accordingly. Similarly, a data analysis on education level and income source can help students decide which courses they should take to earn high income or increase their living standard.

In politics machine learning is used to analyse voting patterns depending on various factors like age, race, income, education e.t.c [Alp14]. In sales and marketing it is used to understand customers buying patterns, recommend products to the customer based on their buying patterns [Alp14][BM01].

Machine learning is used in banking to analyse risk associated with giving loan to customers [Alp14][BM01]. Banks does risk analysis by studying data which has list of customers, their income, saving and expenditure [BM01].

Data is being generated everywhere [Alp14]. When someone opens any page in the internet data is generated because most of the website stores customer activities [Alp14]. These huge complex data can be complicating and time consuming for human being to analyse alone. These complex tasks are made easier with machine learning as it narrows down the information and shows patterns that needs to be analysed [Alp14] [Roj13].

1.2 Goals and Objectives

There are massive collection of data related to health [Alp14] [MP01], business [BM01], education [Alp14] e.t.c. Each of these datasets has some patterns that can reveal cause of certain disease [Saj06], cause of business failure [BM01], relation between customers and products [BM01][WF05][Alp14], relationship between education and career opportunities [Alp14], e.t.c. In order to make human life easier there is an urgent need to extract information from these huge volume of data sets in digital world [KAP].

The goal of this thesis is to develop a web application that is trained to learn patterns in the data. And utilize these patterns to make prediction. Although, there are many research done in machine learning and advanced desktop applications e.g. MATHLAB and SASS has been developed to detect patterns and predict outcome. The developed web application design follows human computer interaction design guidelines and principles to help users easily analyse the data [Shn03].

In addition to predicting an outcome of the entity which is being studied it is also, focused in making the

interface easy to use and understand. Novice users don't need to learn how to use the application and don't have to worry about forgetting the steps to analyse the data. Each form used in the interface has an e.g. showing what kind of data needs to be entered this helps users in preventing errors and user don't have to remember the type of value entered in each input box. Both the error message and flow from one interface to next interface shouldn't make user anxious. Error messages are simple and descriptive telling user what they should do to fix the error. Similarly, each interface shows progress towards predicting the output by showing current step, previous completed steps and next upcoming steps. So, user doesn't get anxious about what will happen next.

Here are the golden rules considered while making the user-interface [Shn03]:

1. Let users control the application [Shn03] e.g. let users decide which learning algorithm to be used to train the data. As much as the application is under control of the users it's more likely to retain users [Shn03]. The application should provide meaningful paths for navigation [Shn03] and to make the system transparent it should display descriptive messages and feedbacks [Shn03].

2. The content of the interface should make users recall information [Shn03]. It should relieve users from remembering information [Shn03]. It should have visual clarification and should intuit users what will happen next [Shn03].

3. Consistency in a user-interface is a key of usable interfaces [Shn03]. The major benefit of consistency is user can use their knowledge [Shn03] while using new applications that have similar functionality.

The main motive of this thesis is to develop a user-centred-design that follows Human Computer Interaction design guidelines and principles [Shn03] in machine learning application.The task of this application is to build a model to map known input to known output i.e. for a given instance in the training data, the application must be capable of computing the exact value in the output column of training data.

Chapter 2

Machine Learning

Machine learning is a sub-discipline of artificial intelligence [Saj06] and uses theory of statistics [Alp14] to build a mathematical model [Alp14]. The developed model is capable of learning from complex large data or past experience [Saj06] [Alp14]. The model can be used for prediction or to visualize pattern in the data and understand information hidden in complex huge data.

Algorithms in machine learning focuses in optimizing the model [Alp14] i.e. adjusting the weight parameters of the model. The optimized model has more accuracy and has low time and space complexity [Alp14] [Saj06]. So, the task of machine learning is to

1. Develop a model using large data which can be related to past experiences in the field of education, health, e.t.c.
2. Optimize the developed model to give high accuracy i.e. runs faster and takes low memory space

2.1 Supervised Learning

Supervised learning means to build a model from a datasets that have both input and output. The training data has both cause and effect of input [Ng]. The goal of supervised learning is to determine function $f(x)$ that best fits the output value with its input set of features. In a optimized cost function the difference between output from $f(x)$ and target output is minimum which increases the accuracy of the system [RM98]. To derive this optimum model function $f(x)$ should be convex in nature. And the system must be flexible enough to use different functions [RM98].

Supervised learning is used both in continuous and classification problem. If the problem is continuous in nature it is called regression, if the problem is discrete in nature it is called classification problem [Ras06]. Supervised learning is used to analyse a dataset or it is used as a sub goal to analyse more complex data

[Ras06].

2.2 Training

Training is an iterative learning process guided by optimization algorithm [RM98]. One of the optimization algorithm used in training is gradient descent. Gradient descent determines weight parameter that when multiplied with a set of input features establishes the relation between input and target [RM98]. A training data consists of $x_1, x_2, x_3, ..., x_j, y$ where x_j is independent variable (input feature) [Men02] and y is dependent variable (output varaible) [Men02] to be determined [Wei05]. In linear regression y can be integer or floating point but in logistic regression it is a value representing a category to be determined [Ng]. Backpropagation is used to determine both categorical and numerical value (integer or float) [Ng].

The difference between output and the target is called error. The objective of training is to minimize this error for both known and unknown inputs [RM98] [Roj13]. Error can be decreased by using appropriate function that maps independent variables (x_j) to dependent variables (y) in training datasets.

Error function depends on structure of training datasets. There is no rule stating which function to use to analyse the dataset [Roj13]. Mostly, polylinear has low error [Roj13] and nonlinear functions are used in neural network [Roj13]. In addition to determining activation function in neural network and hypothesis function in linear and logistic regression we also need to determine number of degree of freedom, number of hidden layers in the network, number of iterations, and eta (η) to lower error [Ng].

Number of iteration means how many times the loop has been executed to determine an optimum weight with gradient descent algorithm [Ng]. If the value of η is low cost decreases with each iteration [Ng] and θ is updated to new optimum θ [Ng]. As a result, as the number of iteration increases cost decreases [Ng]. So, we need to iterate till we find θ that gives the lowest possible cost [Ng].

Number of degree means an integer power upto which the training data attributes are raised to e.g. if number of degree equals to two and there are two attributes x and y in a given data sets then new attribute x^2,y^2, $x * y$ will be added in the data set. Sometime increasing the number of degree decreases cost as it reduces bias (underfitting) [Ng].

2.3 Gradient Descent

The main objective of gradient descent is to improve our hypothesis $h_\theta(x)$ so that we get the minimum cost. In order to achieve the minimum cost we need to adjust weight parameter θ till it returns the minimum cost.

Objective function defined in equation 2.5 is used to calculate cost (error).

The general form of gradient descent algorithm is computed by below given formula[Ng]

$$\theta_j = \theta_j - \eta \frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1) \tag{2.1}$$

where η is learning rate and $J(\theta_0, \theta_1)$ is cost related to training algorithm which is being used. If the learning algorithm is linear regression use cost function defined in equation 2.5, if its logistic regression use cost function defined in equation 2.10, if its backpropagation use cost function defined in equation 2.25

Algorithm 1: Gradient Descent

The value of η should start with small value e.g. 0.1 so that it doesn't skip the lowest point of the convex curve computed by $\frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1)$. If the value of η is small cost decreases in each iteration [Ng]. If η is very small like $\eta = 0.001$ it takes many iteration to reach to the weight at which cost is minimum [Ng]. But, if η is large e.g. $\eta = 2$ then, cost keeps on fluctuating between large and small value of cost as a result, minimum value of cost is not achieved [Ng]. So, it is nice to start with small η but it shouldn't be too small that it has to loop through many iteration to get minimum value of cost as it is time consuming [Ng].

In case of sigmoidal networks like the cost function for logistic regression and backpropagation with classification problem, the initial weights are set to small random values [RM98]. It's main motive is to choose better initial values for weights to speed up learning process [RM98] which leads to reducing training time and decreasing probability of converging in local minima [RM98]. In addition to reducing training time it also provides a method to include domain dependent information in a network [RM98].

Random weights are chosen to break symmetry [RM98]. Breaking the symmetry is important to make nodes compute different functions [RM98]. If all the nodes have same weight and same error value then, backpropagating error doesn't give a chance to converge towards minimum cost because in each iteration it results in same cost [RM98]. If all the weights are symmetrical then, all the nodes in the layer are identical and acts as one node in the layer [RM98].

Small weights are chosen so that it doesn't reach saturation immediately [RM98]. Large weights amplify the inputs which results in high summation value of inputs to the next layer. This pushes the nodes into flat regions of non-linearities resulting in slow learning rate because of small derivative [RM98]. However, it shouldn't be too small as learning speed is dependent on it [RM98].

The initial weights are selected in the range

$$(\frac{-A}{\sqrt{N}}, \frac{A}{\sqrt{N}}) \tag{2.2}$$

where N is the number of inputs and $2 \leq A \leq 3$

2.4 Bias and Variance

The hypothesis function that matches input set of features to the target might be working good for training data as a result cost may be low [Ng]. But the function may not be working as expected for test data set which might lead to low accuracy and high error [Ng]. The cause for high error may be due to variance (over-fitting) or high bias (under-fitting) [Ng].

High bias means the model developed using training dataset doesn't fit the training data very well [Ng]. As a result, the weight estimated from the model doesn't work well with test data set which leads to high error [Ng]. So, high bias leads to under fitting [Ng]. On the other hand, high variance means the model developed using training data sets fits the training data very well and accuracy of prediction is high when tested with data set that was used to train the model[Ng]. A model with high variance is not of general form therefore, it doesn't work well with unseen data which has not been used to train the model. As a result, a model with high variance gives high error [Ng].

The data set is divided into three groups using k-fold method

1. Training data is 80% of total data
2. Cross validation is 10% of total data and is total data - train data
3. Test data is total data - training data - cross validation data

Cost associated with cross validation data set J_{CV} is compared with cost associated with training dataset $J_{train}(\theta)$ with respect to different parameters of the model like training set size, number of features, η value and number of hidden layer [Ng]. This comparison helps to know if we require more data or do we need to optimize parameters value like η, θ, number of iterations [Ng]. It also helps to know if it is suffering from underfitting (high bias) or overfitting (high variance) [Ng] .

1. J(θ) and Degree of polynomial

   As degree of polynomial (d) is increased $J_{CV}(\theta)$ decreases to a certain value of d then it starts to increase [Ng] . So we need to find d at which it is lowest [Ng] . $J_{train}(\theta)$ keeps on decreasing as d increases [Ng] . In high bias (underfitting) both $J_{train}(\theta)$ and $J_{cross}(\theta)$ will be high [Ng] .

2. J(θ) and η.

   In case of high variance (overfitting) $J_{train}(\theta)$ will be low and $J_{cross}(\theta)$ will be high [Ng] . Small η causes high variance and large η causes underfitting [Ng]. As value of η increases $J_{CV}(\theta)$ decreases for a while then, it reaches optimum value of η at which $J_{CV}(\theta)$ is lowest [Ng] . After this optimum value of η when η is increased $J_{CV}(\theta)$ increases [Ng]. And J_{train} also increases rapidly after the optimum value of η [Ng]. The optimum final value of η is at which $J_{CV}(\theta)$ is minimum [Ng].

3. J(θ) and Number of training data instances

   Training the model with large number of data instances fixes high variance problem but, it doesn't fix high bias problem [Ng]. In low training set size with high bias $J_{train}(\theta)$ is low and $J_{CV}(\theta)$ is high [Ng]. In large training set size with high bias $J_{train}(\theta) \approx J_{CV}(\theta)$ [Ng] . For a model suffering from high variance with less training set size $J_{train}(\theta)$ is low and $J_{CV}(\theta)$ is high [Ng] . For a model suffering from high variance with large training set size $J_{train}(\theta) < J_{CV}(\theta)$ [Ng].

Here are the list of common rules to fix high variance and high bias: [Ng]

1. Increasing the number of training instances fixes high variance
2. Decreasing number of features fixes high variance problem
3. Adding more feature fixes high bias
4. Increasing number of degree of polynomial fixes high bias
5. Decreasing η fixes high bias
6. Increasing η fixes high variance

2.5 Linear Regression

It is used in many every day life experience e.g. determine price (y) of the house depending on number of room (x_1), land area (x_2), location of building (x_3); determine how many bicycle will be rented (y) with respect to temperature (x_1), day (x_2), location (x_3), e.t.c. When evaluating a data for linear regression e.g. when analysing sales trend pattern a line is drawn that best fits the sales trend data. The equation of this line can be used to predict future sales informations. Similarly, it can be used to analyse behaviour of customer after the price of a product is changed; analyse risk in a investment e.g. commercial companies can use it to approximate profit.

In all above predictions we can notice a similarity i.e. linear regression is used to predict a numerical value; it is not used for classification problem [Men02]. It is because in linear regression the predicted value can extend to positive infinity as the value of independent variable x_j is increased or can extend to negative infinity as its value is decreased infinitesimally [Men02]. It is not used to give a fix integer value that represents a class to be determined [Ng].

The main objective of linear regression is to find the linear line that best fits all input independent variables to dependent output variable [Ng] [RM98] [Roj13]. This is done by finding continuous expected result function that takes x_j independent variables as input and gives output that is closer to the target

value [Ng]. The general form of hypothesis function used to map input to output is computed by below given formula [Ng]

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + +\theta_3 x_3 + ... + \theta_j x_j \tag{2.3}$$

It can be represented in vector form as below

$$h_\theta(x) = x^T * \theta \tag{2.4}$$

where x is a input vector of independent variables, θ is a weight vector to be determined and y is a output vector of dependent variables which are dependent on x [RM98].

We can notice that the above equation is of the form similar to equation of a line [Ng]. So, it is about finding the line that best fits all given data instances. In order to get that best line we need to try different values of θ in above equation.

Linear regression model provides linear relation between independent variable x_j and dependent variable y [Men02]. If there is no x_j and y then there is no use to build linear regression model as it won't have good prediction accuracy [Men02]. Linear Regression is similar to single layer neural networks [RM98]. Single layer neural networks don't have hidden layer and linear regression also don't have hidden layer. It is a useful first order approximations of non-linear system [RM98].

The cost of linear regression is computed by below given formula [Ng]

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2 \tag{2.5}$$

where $h_\theta(x_i)$ is the predicted value and y_i is the actual value for i^{th} row of data instances.

Above cost function is also called "Squared error function" or "Mean squared error" [Ng].

Cost signifies average sum of distances from the line $h_\theta(x_i)$ to data points (x_i, y_i) [Ng]. For a given datasets we plot a scattered points and $h_\theta(x_i)$ is a best possible line such that, the average distance from the line to data points (x_i, y_i) is less [Ng]. In a best possible case this line passes through all the data points in training datasets in this case cost = 0 [Ng]. In order to find vector θ that gives the minimum cost gradient descent method or newton method is used [Ng] [RM98].

Here are the steps for linear regression algorithm:

2.6 Logistic Regression

Logistic regression is used in classification problem [Ng]. There are many applications of logistic regression to determine if an event occurs or not, determine if a student will pass or fail the exam, determine if a family

1. Remove irrelevant attribute from the dataset which has no effect on training the model e.g. patient-id number in tumor datasets have been removed as it has not no effect on deciding if a tumor is cancerous or not
2. Feature normalize the remaining relevant attribute that will be used for training
3. Train the model and determine the optimal value of θ using gradient descent algorithm described in section 2.3
4. Predict output for a given set of features x using equation 2.4

Algorithm 2: Linear Regression

has low, medium or high income, e.t.c. [Men02].

In logistic regression the hypothesis is discrete is nature i.e. y $\in$ {0,1} [Ng]. In case of binary classification 0 and 1 is normally used to represent value for dependent variable (y) [Men02]. 0 represents negative class (e.g. tumor is benign) and 1 represents positive class (tumor is malignant) [Men02]. The maximum value of probability associated with output value (y) is 1 and minimum value is 0 [Men02].

Logistic regression is also used to predict class or classify given datasets into more than two classes [Ng] i.e. $y \in \{0, 1, 2, ...n\}$. This classification problem works similar to binary classification problem [Ng]. We set the current class to be identified as 1 and all other classes are set to 0 [Ng].

In our web application it doesn't require user to enter data in 0,1 format to represent dependent variable. User can upload data with dependent variable value other than 0, 1. The backend of the application makes the necessary changes of representing class in 0 and 1 format. So, user don't have to manually change the output class value (y) in 0 and 1 format. The predicted value is in the range [Ng]

$$0 \leq h_\theta(x) \leq 1 \tag{2.6}$$

Sigmoid function has been used to make prediction

$$h_\theta(x) = g(\theta^T x) \tag{2.7}$$

$$z = \theta^T x \tag{2.8}$$

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2.9}$$

If $h_\theta(x) \geq 0.5$ then, y = 1 else if $h_\theta(x)$<0.5 then, y = 0 [Ng]. Both hypothesis and cost function of logistic and linear regression are different [Ng]. Cost function of logistic regression is

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} -log(h_\theta(x)) - log(1 - h_\theta(x)) \tag{2.10}$$

Weights for logistic regression is calculated using Gradient Descent algorithm described in equation 2.1 [Ng]. We cannot use cost function of linear regression in logistic regression [Ng] because sigmoid function

used in linear regression will cause output to be wavy resulting in many local optima [Ng] i.e. it won't be convex in nature so, we cannot get the lowest value of cost [Ng]. As a result, logarithmic cost function which has the possibility to get lowest cost for given datasets.

Here are the steps for logistic regression algorithm:

1. Remove irrelevant attribute that will not be used to train the model e.g. remove patient-id number from tumor data because it doesn't effect on categorizing if a tumor is cancerous or not
2. Normalize the remaining relevant attributes which will be used for training
3. Train the model and determine optimal θ using gradient descent algorithm described in section 2.3
4. Predict the output for feature set x using equation 2.7

Algorithm 3: Logistic Regression

2.7 Neural network

Brain of any creature is highly complex comparing it to artificial neural network. Brain consists of large number of parallel processing units to process non-linear informations[HHHH09]. It is faster than any digital computer [HHHH09]. It can perform complex tasks at the same time e.g. it can study patterns of its surrounding, perceive emotions, motor control, e.t.c. all at the same time [HHHH09] but artificial neural network can perform limited tasks.

Neural network is a parallel distributed system consisting of many simple units that performs primitive actions, store information and makes it available to other layers in the network [HHHH09]. Neural network is an imitation of brain but it has limited functionality [HHHH09] [Ng].The main activities performed by neural network are: [HHHH09]

1. Acquires knowledge from its environment in learning process
2. Process acquired information using either primitive function or complex non-linear function known as activation function and store it for future use

Learning process means to change weight of the network in an orderly fashion so that it meets desired design objective [HHHH09]. It can be done using learning algorithm [HHHH09].

Neural networks are powerful because of [HHHH09]

1. parallel distributed architecture
2. abilitity to learn and generalize

Generalization refers to predicting reasonable output for unseen set of features (independent variables) that are not in the training datasets [HHHH09]. In practise, neural network are combined with system engineering approach [HHHH09] i.e. the complex large tasks are broken into simple tasks and neurodes are designed to solve these small tasks which are later combined to build actual design [HHHH09]. The way neural network calculates net input to a neurode and its output depends on the neurode being used in neural network [Ebe14].

Neural network consists of

1. Neurodes

   Neurodes are fundamental units of neural network. They are the inputs to next layer [HHHH09].

2. Weights

   Each inputs x_j in a layer is connected to other neurodes in next layer by w_{kj}

   where k is the layer number and j represents jth neurode in a layer [HHHH09].

3. Adder

   Adder has been used to sum together $x_j * w_{kj}$ of the previous layer to form a new neurode for the current layer. It consists of input signal (output signal of previous layer) and bias b_k to increase or decrease the net input of the activation function [HHHH09].

4. Activation Function

   It is required to limit amplitude of output given by a neurode [HHHH09] and activation function is used to limit this amplitude.

Mathematically, we can define neurode k as follows:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{2.11}$$

$$y_k = f(u_k + b_k) \tag{2.12}$$

where $x_1, x_2, x_3, ..., x_m$ are input signals; $w_{k1}, w_{k2}, w_{k3}, ..., w_{km}$ are weights; u_k is the adder; b_k is the bias unit; f(x) is the activation function and y_k is the output of neurode k

There are two basic types of activation function [HHHH09]. They are listed below:

1. Threshold Function

$$g(v) = 1 \quad if \quad v \geq 0 \tag{2.13}$$

$$g(v) = 0 \quad if \quad v<0 \tag{2.14}$$

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k \tag{2.15}$$

   where $g(v)$ is the threshold activation function; x_j is input; w_{kj} is weight; b_k is bias

2. Sigmoid Function

   Sigmoid function is the most common activation function used in neural network [HHHH09]. Logistic function is an example of sigmoid function [HHHH09][Roj13]. The graph of sigmoid function is "S" shaped [HHHH09][Roj13][Ng].

$$f(v) = \frac{1}{1 + exp(-av)} \tag{2.16}$$

   where a is slope parameter of sigmoid function.

   As slope parameter (a) approaches zero, sigmoid function becomes threshold function [HHHH09]. Sigmoid functions are differential but threshold functions are not differential [HHHH09]. Differentiability is an important feature of neural network [HHHH09]. The most common range of values desirable is -1 to +1 [HHHH09].

The benefits that can be obtained from neural networks are:

1. Nonlinearity

   Non-linearity is an important property to perform complex tasks like speech recognition, face recognition where inputs to the network are non-linear [HHHH09]. A computing unit neurode in neural network can be linear or non-linear [HHHH09]. A neural network consists of interconnection of non-linear neurodes[HHHH09]. Neurode is itself non-linear and is distributed throughout the network [HHHH09].

2. Input-Output Mapping

   In supervised learning where each input has a corresponding pre-specified output in a training dataset [HHHH09]. Each input data is multiplied with weights to predict an output. So, hypothesis function maps input and output variable of an entity. Weights of a neural network are modified so that the difference between predicted value and the desired value, cost is low [HHHH09]. In a training (learning) process weights are repeatedly modified till it reaches the steady stage at which cost is lowest. At this time when cost is lowest there is no significant change in weight [HHHH09].

3. Adaptivity

   A neural network built to work in a particular environment can be retrained to adapt new environment by changing its weights [HHHH09]. If a neural network is working in nonstationary environment it can be designed in a way in which its weights get updated with real time [HHHH09]. When a system is more adaptive to its surrounding most of the time it becomes more robust but not always [HHHH09].

Neural network is a hierarchical interconnected structure to process information. It is used to process both parallel and distributed information.[HN89]. In neural network both neural architecture and learning algorithms are important [Roj13]. Any continuous smooth function can be generated with neural network by

just changing the parameters of the network provided that the enough number of hidden units are available [Rip07].

A neural network has a input layer, hidden layer and output layer [Rip07]. All these layers have nodes or processing unit [Spe91]. Input layer has a node for each dataset. All these layers are interconnected by unidirectional parameter called weights which are adjusted to get the desired target value. And within each input node it stores a set features. Hidden layer has row of activation units which is the sum of input nodes times weights connecting the input layer with output layer. Nodes from hidden layer is used to compute output value [HN89].

Linear regression is not used to analyse complex datasets with many features [Ng]. If a given dataset has 100 features and needs to compute new feature with quadratic representation of given inputs then it is hard to find quadratic line that best fits all these features with low cost [Ng]. In such cases neural network can be used [Ng]. As, the degree of polynomial increases to compute new features computation time also increases.

Neural network must be supplied with enough dataset so, that it can use separate dataset for training and testing purpose [Ebe14]. The amount of data required to train the network depends on architecture of the neural network, problem to be addressed and training method[Ebe14].

Kolmogorov's theorem states that any function can be generated by a finite neural network with computing units provided that necessary primitive function is given for each node [Roj13]. We can also use bounded approximation error. In this case we take the best possible approximated value for the given function [Roj13]. This is the same approximation method that we follow in backpropagation and other neural network mappings [Roj13].

2.8 Feedforward

A normalized input values of range 0-1 is supplied to the network [Ebe14]. A neural network consists of input layer, hidden layer and output layer [Ng]. Each neurode in input layer is connected to all neurodes in hidden layer and each neurode in hidden layer is connected to all neurodes in output layer [Ng] [Ebe14] [RM98]. A weight is associated with each connection of neurode to neurode in hidden layer [Ebe14] [Ng] [Roj13] [RM98]. And the flow of data is from left to right [Ebe14] [Ng] [Roj13] [RM98]. There is no feedback back loop even to itself in feedforward network [Ebe14] [Ng] [Roj13] [RM98]. All backpropagation neural network implements feedforward network to propagate inputs [Ng] [Roj13].

Feedforward network is a non-linear regression model which determines the best function [Roj13] and weight parameter that fits input-output relation. The two main function of feedforward is:

1. Map known inputs to known outputs

2. The developed network parameters i.e. functions and weight should be of general form so that, it is capable to produce output with minimum error for unseen inputs [Rip07].

The approximated function to map input-output relation should have low training error for both seen training inputs and unseen test inputs. Training error can be reduced by increasing degree of freedom [Ng] but it can increase error related to test datasets [Rip07]. The degree of freedom is related with degree of polynomial of functional approximation [Rip07].

A input signal to a neurode in hidden layer is a product of previous layer input and weight of the connection signal [Ebe14] [Ng] [Roj13] [RM98]. The net input to a neurode is sum of all the input signals coming to that neurode [Ebe14] [Ng] [Roj13] [RM98]. It might also include bias, input from the neurode itself.

$$netinput_j = i_j = \sum_i w_{ji} o_i \tag{2.17}$$

where i_j is jth input signal; w_{ji} is weight of the signal that connects neurode from previous layer with the neurode in current layer; o_i is output of the signal.

The output of hidden layer neurode is given by sigmoid function applied to the net-input of that neurode [Ebe14]

$$output_j = o_j = \frac{1}{1 + exp(-i_j)} \tag{2.18}$$

Sigmoid function also called squashing function [Ebe14]. It is of "S" shape and its value is in range of 0 and 1 [Ebe14]. For a neurode with net-input of 0 its output given by sigmoid function is 0.5, for a large negative inputs its output is nearly 0 and for a large positive net-inputs its output is nearly 1. A non-linearity property of sigmoid transfer function plays an important role in robustness of neural network [Ebe14]. Other continuous functions which posses derivative at all points can also be used [Ebe14] as transfer function. Some of the other functions used as transfer function are trigonometric sine and hyperbolic tangent function [Ebe14].

The number of neurodes in a hidden layer depends on the application and its statistically significant factors that exists in the input data [Ebe14]. In many cases if we knew this number we can develop an algorithm and don't require to create neural network [Ebe14]. However, it is reasonable to start with is square root of a number of neurodes in the input layer plus number of neurodes in the output layer and few more neurodes [Ebe14].

If we have only few neurodes in a hidden layer then it will have high bias (underfitting) [Ebe14] [Ng] as a result, it cannot train a neural network and will have high training error [Ebe14] [Ng]. If we use barely

enough number of neurodes then it won't be robust enough to detect noise or it won't recognize patterns in new unseen data which were not used to train the neural network [Ebe14] [Ng]. On the other hand, if we use too many neurodes in hidden layer then it will suffer from high variance (over fitting). High variance leads to high error when tested on unseen data because the neural network may not be general enough to analyse unseen data [Ebe14] [Ng].

Once output for all the neurodes in the hidden layer are calculated it is passed as input to output layer. The input to each neurode in output layer is similar to how we calculated input to each hidden layer [Ebe14] [Ng] [Roj13] [RM98].

$$netinput_i = i_1 = \sum_i w_{1j} o_j \tag{2.19}$$

Similarly, output of each neurode in output layer is calculated as we calculate output of a neurode in a hidden layer [Ebe14] [Ng] [Roj13] [RM98].

$$output_1 = o_1 = \frac{1}{1 + exp(-i_1)} \tag{2.20}$$

Testing new input means calculating output in one forward pass using the test input using the optimized weights [Ebe14].

The number of output neurodes needed is equal to number of classes used to classify given datasets [Ebe14] [Ng]. In case of two mutaully exclusive classes e.g. if it needs to predict if a tumor is cancerous or not only one output node can be used to classify the data [Ebe14] [Ng]. Sometimes, it might be supposed that classes are mutually exclusive but it may not be the case so, it is better to use as many neurodes as the number of classes [Ebe14]. The computation time added by adding more output neurodes in output layer to make it equal to number of classes don't increase the computation time so much [Ebe14].

In feedforward each neurode performs two operations: [Ebe14]

1. Calculate sum of previous layer neurode output times weight of the connecting signal
2. Apply transfer function (sigmoid function most commonly used)

2.9 Backpropagation

Backpropagation is most commonly used [HN89][RM98] in real applications. It is a numerical method for statistical approximation [Rip07]. The popularity of backpropagation is mainly because of its ability to learn complicated multidimensional mappings[HN89]. It refers to two main points:

1. Calculate derivative of error in neural network with derivative chain rule[RM98]
2. Use above calculated derivative in gradient descent algorithm to optimize weight[RM98]

Backpropagation Neural Network is a supervised learning algorithm [RM98] with hierarchical structure of fully interconnected layers or rows of processing units with each unit itself consisting of several individual processing units [HN89]. For every input there is a pre-specified output which is the target value. The purpose of backpropagation is to adjust network weights so that the error which is a difference between the target value and network predicted value is low [RM98].

Input to a neural network can be a set of raw data, parameters, features that we have chosen to represent a pattern [Ebe14] e.g. if we are building a neural network for XOR a single value is provided as input to each node in input layer but if we are building neural network for video processing we present averaged value of several pixel to a node in input layer [Ebe14].

The equation used in calculation of backpropagation are divided into two categories [Ebe14]:

1. Feedforward calculation
2. Backpropagation calculation

The given datasets is divided into two sets:

1. Training set which is 90% of given dataset
2. Test set which is a remaining dataset i.e. 10% of given dataset

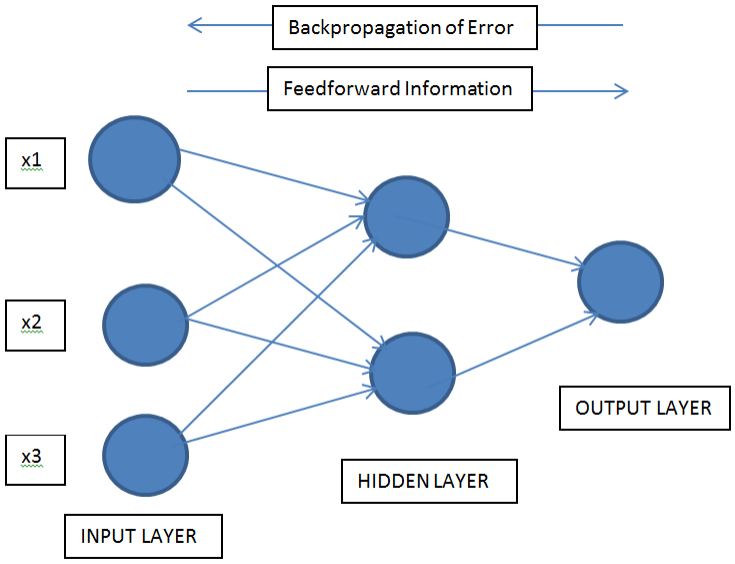

Figure 2.1: Backpropagation

The steps for training with backpropagation are listed below:

1. Normalize the training data.

   In backpropagation neural network each input node takes value between 0 and 1 [Ebe14]. This adds flexibility to a neural network as it processes continuous and normalized value between 0 and 1 [Ebe14]. The normalization value between 0 and 1 doesn't constrain the use of backpropagation as digital computers are limited by the size of inputs it can process [Ebe14]. In some neural networks the way we choose to normalize the data affects the performance of neural network [Ebe14].

2. Submit training data and propagate it through the network using feedforward propagation.

   In forward pass each node computes an output based on current inputs i.e. node i computes a weighted sum a_i of its inputs and pass it through a nonlinear sigmoid function to obtain output node y_i [RM98]. Sigmoid function changes the functional approximation to non-linear form and hence generates a continuous smooth step function [Rip07]. It is used in most of the backpropagation algorithm to calculate output of the neurode [Ebe14].

$$a_i = \sum_{j<i} w_{ij} y_j \tag{2.21}$$

   where w_{ij} is weight; y_j is current input

$$y_i = f(a_i) \tag{2.22}$$

   where y_i is output; a_i is activation node

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.23}$$

   where $f(x)$ is sigmoid function

3. Compute error E by calculating difference between target value and network output value [RM98].

   Error function or cost function measures the difference between target value and network output. [RM98]. The most commonly used cost function is mean sum square [RM98] because graph generated with this function is convex in nature which helps to obtain minimum cost.

$$E = \frac{1}{2} \sum_{p} \sum_{t} (d_{pi} - y_{pi})^2 \tag{2.24}$$

   where d_{pi} is desired output; y_{pi} is output of neural network
   For classification problem sometime logarithmic cost function is used [RM98]:

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log((h_\theta(x^{(i)}))_k) + (1-y_k^{(i)}) \log(1-(h_\theta(x^{(i)}))_k)] + \frac{\lambda}{2m} \sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}} (\theta_{j,i}^{(l)})^2 \tag{2.25}$$

4. Calculate derivative of above error E with respect to weight W using chain rule [RM98]

$$\frac{\delta E}{\delta W} \tag{2.26}$$

5. Adjust weights w to minimize error [RM98]

$$w(t+1) = w(t) - \eta \frac{\delta E}{\delta w} \tag{2.27}$$

   where η is learning rate.

6. Repeat until error is acceptably small or time is exhausted [RM98]

Algorithm 4: Backpropagation

Value of η affects the training process [FL89] i.e. it affects how fast the optimized weight is achieved and how much low is the training error [RM98]. The typical range of η values are $0.05 \leq \eta \leq 0.75$ [RM98]. The most common value of η is 0.1 [RM98] it is suitable for most of the problems[RM98]. If η is low it takes much longer time to train a network to converge [RM98] because change in weight is very small [Rip07]. But if learning rate η is high step size will be large which might then lead to skipping the point of convergence [Rip07] resulting to increase in cost. So, the proper learning rate (η) gives decreasing cost in each iteration of stochastic gradient descent [Ng].

In training phase, weights are adjusted depending on output from feedforward pass and backpropagated error [Ebe14]. This backpropagation of error helps in adjusting weight in a learning process which is the main reason for its effectiveness [Ebe14].

There are two ways to backpropagate error in neural network: [Ebe14]

1. Online Single pattern training

   In this training process, we backpropagate error and adjust weight of the network after each training pattern [Ebe14].

2. Batch Epoch training

   In this training process δ is calculated for each neurode in the entire training set, add them and then, backpropagate error based on grand total δ obtained [Ebe14].

$$\delta_l = f^1(i_1)(t_1 - o_1) \tag{2.28}$$

In a training process of a network we adjust its weight so, weight should be initialized before we start training [Ebe14] [Ng]. For any neurode weight is not initialized to zero because a network won't be trainable [Ebe14] if its weight is initialized to zero [Ng]. For this reason, we train a network with initial values of weights as random values between 0.3 and -0.3. The bound between 0.3 and -0.3 are found to work well for most of the applications [Ebe14]. For some neural network it works faster given these bounds compared to 1 and -1 [Ebe14].

Some other recommended way to speed up training is initializing weights of a signal connecting to output neurode in the range between 0.3 to -0.3 and initialize weights of a signal from input neurode to hidden layer is 0 (provided that the weights from the hidden layer to next layer is random non-zero value) [Ebe14].

In addition to adjusting weights it is also important to determine error which helps to know how effective is the network [Ebe14]. For each neurode in output layer we calculate difference between predicted value o_i calculated with feedforward pass and the actual value of the neurode supposed to have, t_i, sum this difference

for all the neurodes in the output layer [Ebe14] [Ng]. The summed value is the error associated with a neural network [Ebe14] [Ng].

Mathematically, it is denoted as below

$$E = 0.5 * \sum_{l=1}^{n} (t_l - o_l)^2 \tag{2.29}$$

where n is number of classes to classify submitted datasets; t_l is target value that a neurode supposed to have; o_l is predicted value obtained by using feedforward pass for a neurode.

The main objective of training process in backpropagation is to minimize average sum squared error over all training examples [Ebe14] [Ng][RM98]. Minimizing error with respect to neurodes in the hidden layer is key for the wide spread application of backpropagation [Ebe14].

2.9.1 Gradient Checking

Gradient Checking helps to check if backpropagation model is correct or not.

$$\frac{\delta}{\delta\theta} J(\theta) \approx \frac{J(\theta + \epsilon) - J(\theta - \epsilon)}{2\epsilon} \tag{2.30}$$

where $\epsilon = 10^{-4}$ We now check if $\frac{\delta J}{\delta \theta}$ obtained from above equation (2.30) is approximately equal to $\frac{\delta J}{\delta \theta}$ obtained from backpropation equation (2.25) [RM98].The difference between these two values should be less than 1^{-10} [Ng]

2.9.2 Limitation of Backpropagation

The main limitation of backpropagation learning algorithm is it is too slow [FL89] [Rip07] [RM98]. Two main cause for its slowness are:

1. Step Size Problem (η) [FL89] [Rip07] [RM98]
2. Moving Target Problem [FL89]

   Each unit in a model is acting as a feature detector that affects the overall efficiency of a model [FL89]. Each of these units receives input and backpropagates error signal [FL89] [Rip07] [RM98]. Error signal signifies the problem that a model is trying to solve [FL89]. And this error signal affects activation units and feature detectors [FL89]. As a result, it takes long time to settle down with a definite error. [FL89]

2.9.3 Complexity of Backpropagation

Backpropagation algorithm has a disadvantage that it takes long time to find optimal value of gradient at which cost is lowest [Roj13]. We can make this learning process faster to some extent by increasing η but as gradient approaches the flat regions of error function it is too small as a result, it takes a while to get minimum cost [Roj13]. And finding a weight for one input-output instance is hard, its NP-complete problem [Roj13].

Chapter 3

Applications of Machine Learning

According to Alpaydin [Alp14], data is generated and consumed everywhere. Data are generated when we buy goods, visit websites, rent books, e.t.c [Alp14]. Nowadays, customers want their needs and interest to be predicted [BM01]. Supermarkets sell large quantity of products to millions of customers, each of these transaction store customers details, product details, locations e.t.c. [Alp14]. Retailers want to increase their sells by knowing what customers want to buy and customers want to buy things that match their interests and needs [Alp14]. What customers exactly need cannot be known but their interest can be predicted [Alp14] e.g. kids like cartoons, ice-creams are mostly preferred in summer, e.t.c. So, different kinds of relations and patterns can be determined from a data [Alp14]. These patterns can be studied in datasets stored by education, finance, cosmetics, medicine e.t.c. organizations [Alp14].

Machine learning helps to detect patterns, regularities and make good approximation or prediction. It is used in data analysis of retail, finance, credit applications, fraud detection, stock market, medical diagnosis [Alp14]. It is also used to process large databases information where it is not obvious how information are interrelated [Che93]. Similarly, neural network is mostly used to analyse data of following categories: imprecise, fuzzy, imperfect knowledge and lack of mathematical algorithm to analyse the data [Ebe14].

Machine learning is not only about data analysis it also deals with intelligence as it is a sub-discipline of artificial intelligence [Alp14]. In order to make the application intelligent enough it must understand and learn patterns in the changing situation so that, system designers don't have to program the system to work in all possible situations. [Alp14]

Machine learning is used in genetic control and selective adaptation to study genetics [RTB67]. It uses self generating numeric patterns or single bits of patterns in the data sets. [RTB67]

It is used to study relation between customers and products in retails [Alp14][BM01] e.g. $P(chips|beer) = 0.7$. It means 70% of people who buy beer also buy chips. Similarly, it can be used to study relation between readers, authors, e.t.c.

In finance, machine learning is used to predict risk [Alp14][BM01] e.g. predict how likely is a customer going to return interest and bank loans. Banks have past history data about customers with their profession, income, bank transaction, e.t.c. and using these data machine learning applications make predictions.

So, machine learning is about developing a model that uses previous data sets to make future predictions [Alp14]. If future data is similar to training data using which the application model was developed then the accuracy of prediction is high [Alp14].

Machine learning is also used in pattern recognition e.g optical character recognition, face recognition, handwriting recognition, e.t.c. [Alp14]. In case of pattern recognition there are huge datasets of images and there are multiple classes for each image which needs to be recognized [Alp14].

Prediction problems where output is a number not a class is called regression [Alp14] e.g. predicting price of a house, car, e.t.c. Machine learning can be used in regression problems. To predict value of a dependent variable(e.g. price of a house) it uses independent variables like number of room, area of land, location, e.t.c. Problems where dependent variable is a class e.g. malignant or benign it is called classification. Machine learning is also used in classification problems.

Image compression, customer relationship management, document clustering, can be achieved using clustering technique of machine learning [Alp14].

Machine learning is also used in analysing policy. Output of a system is determined by sequence of previous actions. In a intermediate stage of a system it is hard to determine if an action leads to a goal or not. In these cases machine learning is used to study goodness of policy by studying past good actions [Alp14].

3.1 Application of Machine Learning in Health Data

There is an increasing interest in pattern recognitions and machine learning in biomedical field. Machine learning has shown to improve sensitivity in disease recognition and decision making [WF05] [Saj06]. There are huge collections of data from past decades for different kinds of disease. These data can be used to study patterns [Saj06] which can help to predict cause of a disease and ultimately develop tools to cure it.

According to Turgay, machine learning is being used to classify patient data in two category "patient suffering from a disease" and "patient not suffering from a disease". This classification helps physician to identify patients by just submitting the patient features in a neural network application and the application predicts the likelihood of person suffering from a disease [ACA+10].

Logistic regression and an artificial neural network are the most commonly used learning algorithms in clinical risk estimations[ACA+10]. It is used in analysis of medicines, therapy results, progression of disease, manage patients data, e.t.c. [MP01]. It also helps in efficient monitoring of patient activity and helps to deal with imperfect data [MP01].

Machine learning and pattern recognition are used in brain imaging and computational neurosciences, as they are instrumental for mining vast amounts of neural data of ever increasing measurement precision and detecting minuscule signals from an overwhelming noise floor [Alp14]. They provide the means to decode and characterize tasks relevant to brain states and distinguish them from a non-informative brain signals .

Machine learning have helped medical practitioners to analyze complex datasets and reveal relation between attributes in the dataset [Kon01].

Complementary medicine are becoming more and more important. Machine Learning can be used in analysis of these complementary medicine because of its transparency feature [Kon01]. Machine learning tools can be used by physicians as any other tools used to diagonalize disease [Kon01]. Depending on the accuracy of the machine learning model physician can accept or reject the predicted outcome [Kon01].

In this modern technical era machine learning is used in health informatics [CNCC15]

1. To build wearable sensors to collect health information of a patient
2. Biomedical signal processing
3. Analyse clinical data

3.2 Previous Studies and Analysis of the Data

Rudy and Huan [SL97] has used same UCI breast cancer datasets to study NeuroLinear. And their study has given a predictive accuracy of 99.91% which is higher than the accuracy given by our application using Logistic and Backpropagation algorithm when trained with a 1200 epoch.

NeuroLinear is a system for extracting oblique decision rules from a neural network which has been trained for classification of pattern [SL97]. It is effective in extracting compact and comprehensible rules

with high predictive accuracy for a given neural network [SL97].

A list of steps to implement neurolinear algorithm is given below: [SL97]

1. Train a neural network using efficient algorithm to meet the pre-specified accuracy.
2. Remove redundant connections in the network using pruning.
3. Discretize the hidden units activation values using Chi2 algorithm
4. Generate rules for small datasets with discrete values using X2R algorithm

Algorithm 5: Rule extraction using NeuroLinear

In a study by Hussein on same data the study finds that Memetic Pareto Artificial Neural Network (MPANN) generalizes better and has low computational cost compared to standard back-propagation algorithm [Abb02]. But using 20% of datasets for testing our application gave 98.5% accuracy in 400 epoch which is lower than the computational accuracy given by MPANN which is 98.1% [Abb02] achieved after 5100 epoch [Abb02]. The high accuracy in our web application might be because of the higher number of neurodes used in hidden layer compared to MPANN network. In our application the number of hidden units used is half the number of neurodes in input layer but, in neural network developed by Hussein the number of hidden units is 4.125+1.360 and 4.125-1.360 which is lower [Abb02].

The objective of Memetic Pareto Artificial Neural Network (MPANN) algorithm is

1. Decrease computational cost
2. Decrease number of hidden units

But the objective of our application is different, it's objective is to decrease error.

In a study by Endre and Peter on "Implementation of Logical Analysis of Data" the accuracy obtained on classifying the same dataset is 97.2% using 80% of data for training purpose [BHI$^+$00]. But in our application the accuracy given by both logistic and backpropagation algorithm is 98.5%. In our application all the features except the patient-ID has been used for training but in the neural network developed by Endre and Peter only a limited number of features [BHI$^+$00] were used to train the network this might have affected the accuracy %. The objective of their study was to determine minimal set of features to classify positive outcomes from negative outcomes [BHI$^+$00].

In a study "Extracting M-of-N Rules from Trained Neural Networks" done by Rudy the accuracy obtained on testing 10 fold dataset is 94.04% [Set00]. The accuracy percentage is low compared to the accuracy percentage of our application because it focuses only on some of the features to classify the data. The main objective of this study is to built a simple rule to classify the dataset [Set00].

In a similar study by Ismail and Joydeep which was to extract rules from trained network [TG96]. The accuracy given by different algorithms based on different extracting rules are:

1. BIO-RE 97.77% [TG96]
2. Partial-RE 97.22% [TG96]
3. Full-RE 97.77% [TG96]
4. NeuroRule 97.21% [TG96]
5. C4.5rules 94.72% [TG96]

The study performed by Chotirat and Dimitrios the same dataset gave an accuracy of 97.38% using Selective Bayesian Classifier [RG02]. The objective of their study was to show that Selective Bayesian Classifier performs better than Naive Bayesian Classifier and outperforms C4.5 on datasets in which C4.5 performs better than Naive Bayesian Classifier [RG02]

Chapter 4

Machine Learning Analysis of Data

4.1 Breast Cancer Data Instance with 10 attributes

The web application is tested using publicly available Wisconsin Breast Cancer Data found in UCI Machine Learning Repository http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29. The properties of data is listed below:

1. Number of Instances: 699
2. Number of Attributes: 10
3. Missing values: 16
4. Class distribution: 458 Benign and 241 Malignant

Attribute	Range of Value
Clump Thickness	1 - 10
Uniformity of Cell Size	1 - 10
Uniformity of Cell Shape	1 - 10
Marginal Adhesion	1 - 10
Single Epthelial Cell Size	1 - 10
Bare Nuclei	1 - 10
Bland Chromatin	1 - 10
Normal Nucleoli	1 - 10
Mitosis	1 - 10
Class:	1 - 10

Table 4.1: Tumor With 10 attributes

Table 4.1 lists attributes used to train neural network. From the dataset available in UCI *sample code* attribute has been removed from both training and testing datasets.

The data set is divided into three groups using k-fold method

1. Training data is 80% of total data
2. Cross validation is 10% of total data and is total data - train data
3. Test data is total data - training data - cross validation data

Accuracy is calculated by testing on this last 60 instances (10% of datasets) that have not been used to train the neural network. Testing means using the input features in column 1-8 application predicts the targeted value in 9^{th} column of the data file.

Class	Correctly Predicted Number of Instances	Actual Number of Instances
Benign	55	56
Malignant	14	14

Table 4.2: Number of correctly predicted instances using *Logistic Regression* on breast cancer data with 9 attributes

Table 4.9 shows result of neural network trained with 549 instances using *logistic regression* algorithm. Each training instance has 9 features all listed in table 4.1. Table 4.9 shows that only 1 benign instance is not predicted correctly.

Class	Correctly Predicted Number of Instances	Actual Number of Instances
Benign	55	56
Malignant	14	14

Table 4.3: Number of correctly predicted instances using *Backpropagation* on breast cancer data with 9 attributes

Again, we see in table 4.10 that only 1 benign instance is counter predicted when trained with same data and parameters using *backpropagation* learning algorithm.

Table 4.8 shows accuracy % and cost associated with each learning algorithm. In the given table accuracy % of linear regression is not calculated because linear regression is not used for classification problem therefore, it's accuracy is not calculated. But, cost associated with linear regression is calculated using same output column data with same value of η and number of iterations as in backpropagation and logistic regression. All the given algorithms use same training and test dataset as mentioned earlier.

Algorithm	η	Number of iterations	Cost	Accuracy %
Linear Regression	0.1	400	0.08054	Not Applicable
Logistic Regression	0.1	400	0.101	98.55%
Backpropagation	0.1	400	0.31	98.55%

Table 4.4: Classification accuracy on classifying if the tumor is cancerous or not given a breast cancer data with 9 attributes

The given screen shot in figure 4.1 shows parameters to train the model using linear regression. It has an input box to enter features and depending on the submitted feature it predicts other missing attribute value.

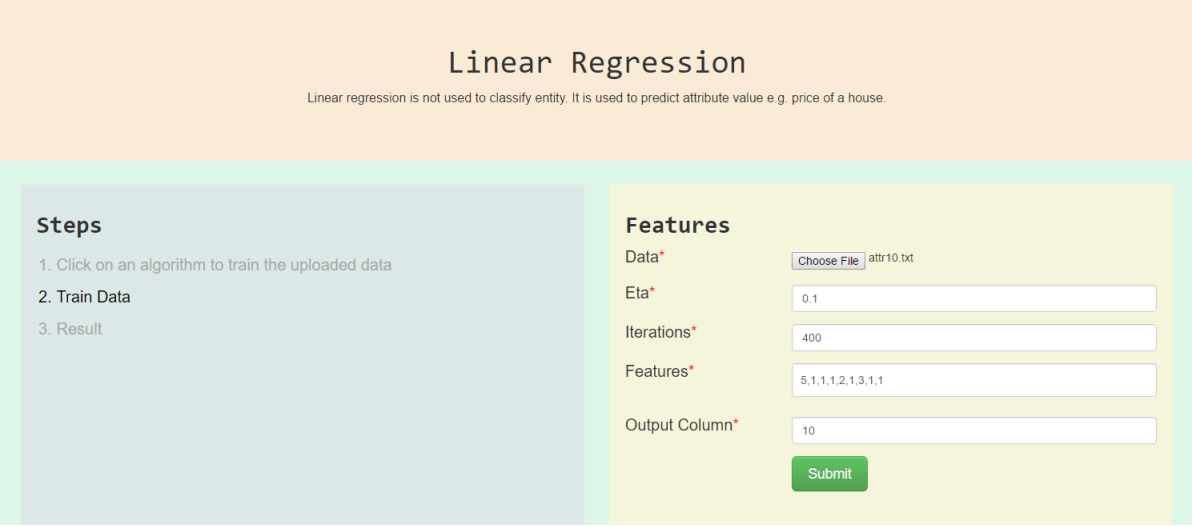

Figure 4.1: A tumor having attributes as entered in third input box is submitted for prediction.

Figure 4.2: Linear Regression result page showing cost and predicted value for above submitted parameters.

Below given figure 4.3 shows content of a file used to train and test the model. It's 90% of data has been used to train and 10% has been used to test the model. The content highlighted in blue is entered in input box, "*Features*" of figure 4.4. The targeted value to be predicted is the value in last column of highlighted data row which is "2".

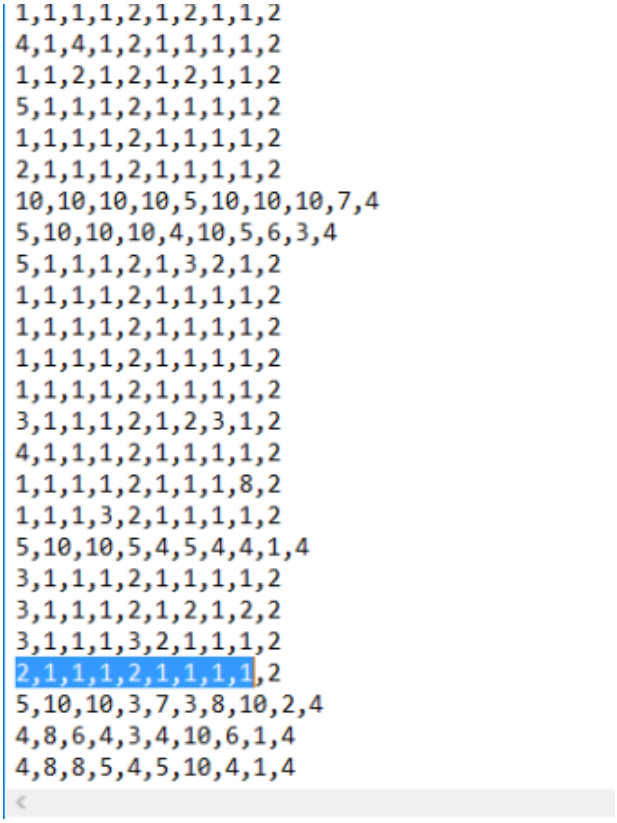

Figure 4.3: The highlighted feature represents attributes of a tumor whose category needs to be predicted. This set of attributes is passed as input in input box "Features" shown in figure 4.4.

Each input box in figure 4.4 has a value to train a network with logistic regression. Also, it has a input box to enter breast cancer attribute values to predict if a tumor with those set of attributes is cancerous or not.

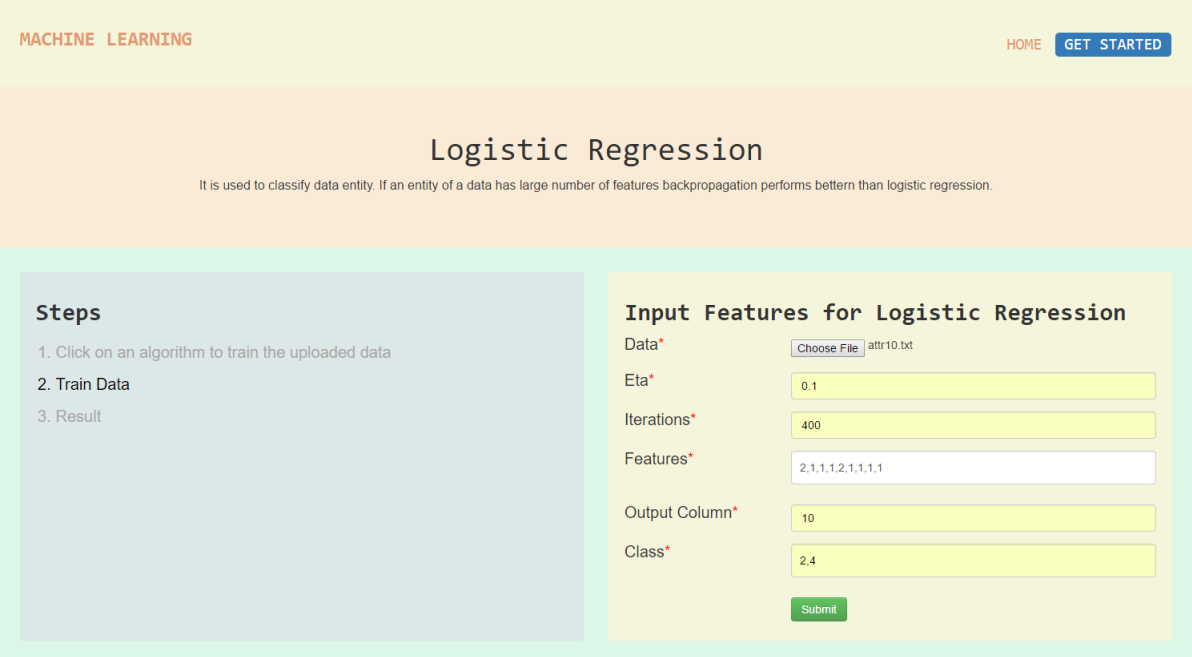

Figure 4.4: The application predicts if an above given set of attributes of a tumor is cancerous or not. The set of attributes is same as the highlighted value in figure 4.3

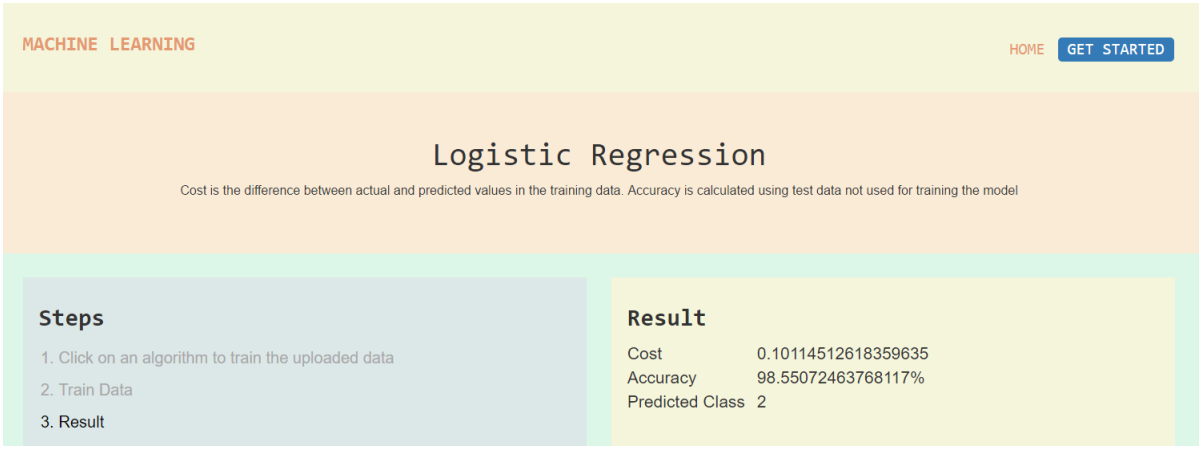

Figure 4.5: It shows the predicted category of a tumor having features as in figure 4.4. The predicted value matches the target value.

The application doesn't display "Benign" or "Malignant" as the predicted value but, displays "2". This is because the class predicted by the application is in same format as the format of classes used to train the neural network. The data instances used for training has 2 and 4 as classes to be detected so, the application

predicts either 2 or 4. We can say the predicted value is "benign" because data description describes 2 as benign and 4 as malignant.

Below give figure 4.6 motive is to show console section of backend of the web application when it is trained with logistic regression and breast cancer data having 9 attributes. It shows decreasing value of cost in each iteration of gradient descent, accuracy, number of correctly predicted benign and malignant data instances.

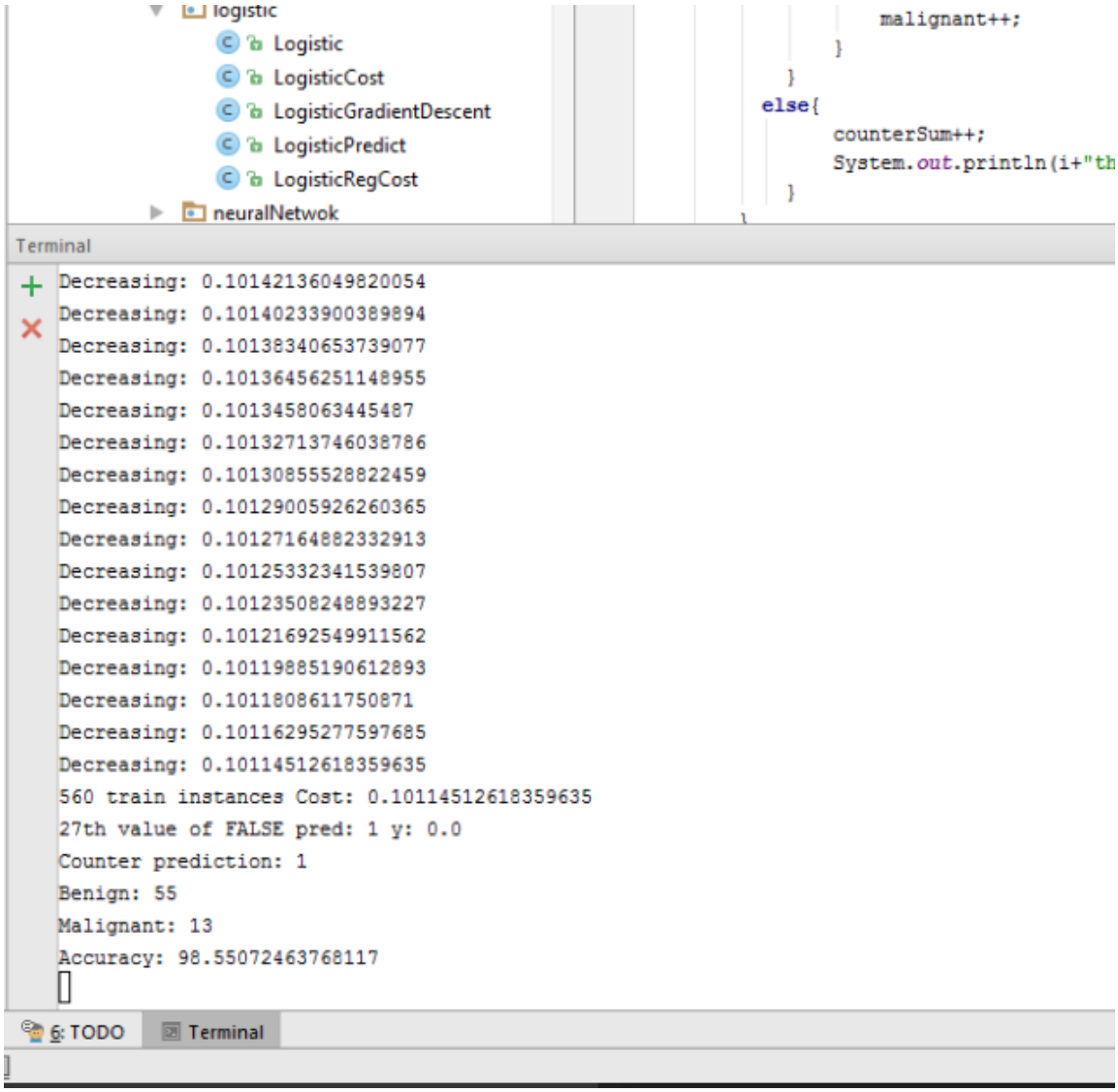

Figure 4.6: It shows decreasing value of cost in each iteration of gradient descent while training data with 9 attributes using logistic regression.

Below given figure is a cropped content of data having 9 attributes and was used to train a model using backpropagation. The highlighted content of the data are attributes of a tumor whose category needs to be determined i.e. if a tumor with a highlighted attributes is cancerous or not.

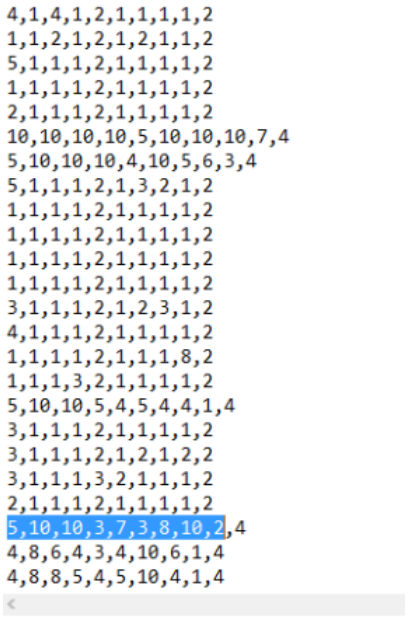

Figure 4.7: It shows cropped content of training dataset and the highlighted attributes of a tumor is passed as input to the application and its category is predicted using backpropagation learning algorithm.

Below given figure 4.8 shows the parameters used to train backpropagation neural network. In input box "*Features*" user enters attributes of the tumor which needs to be predicted if its cancerous or not. In that input box the set of attributes entered is the highlighted attributes shown in earlier figure 4.7

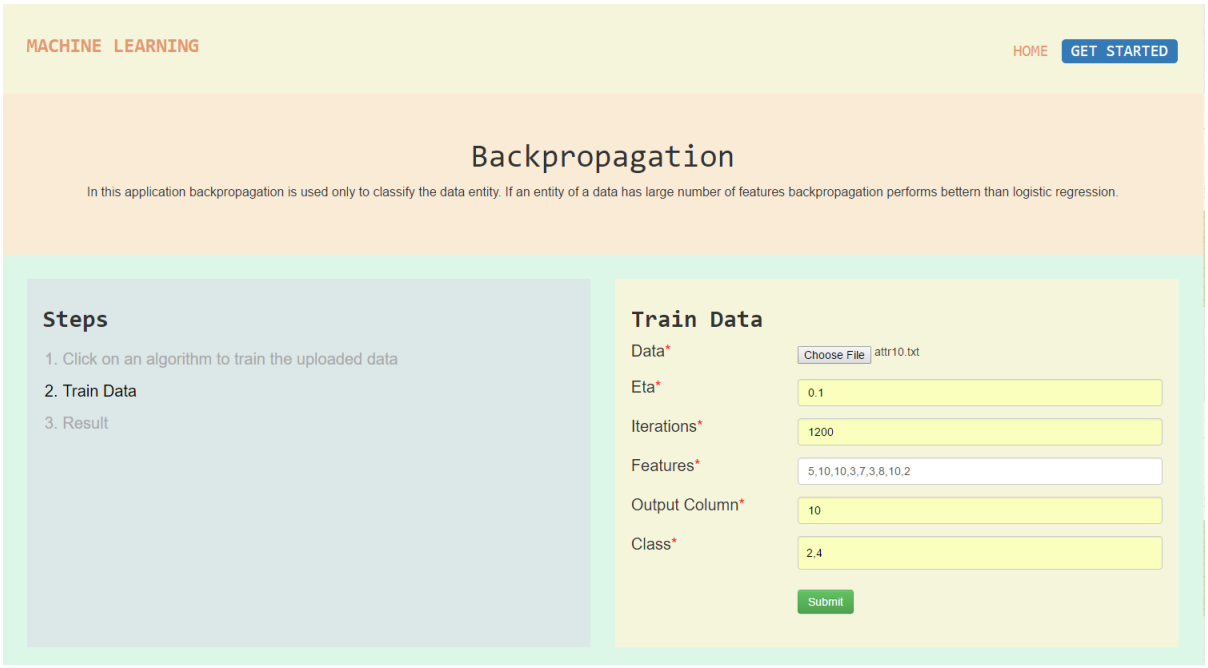

Figure 4.8: The form shows the parameters used to train the application. The third input box has same set of features as highlighted in figure 4.7. This figure captures the stage of the application before predicting category of a tumor with attributes as highlighted in figure 4.7.

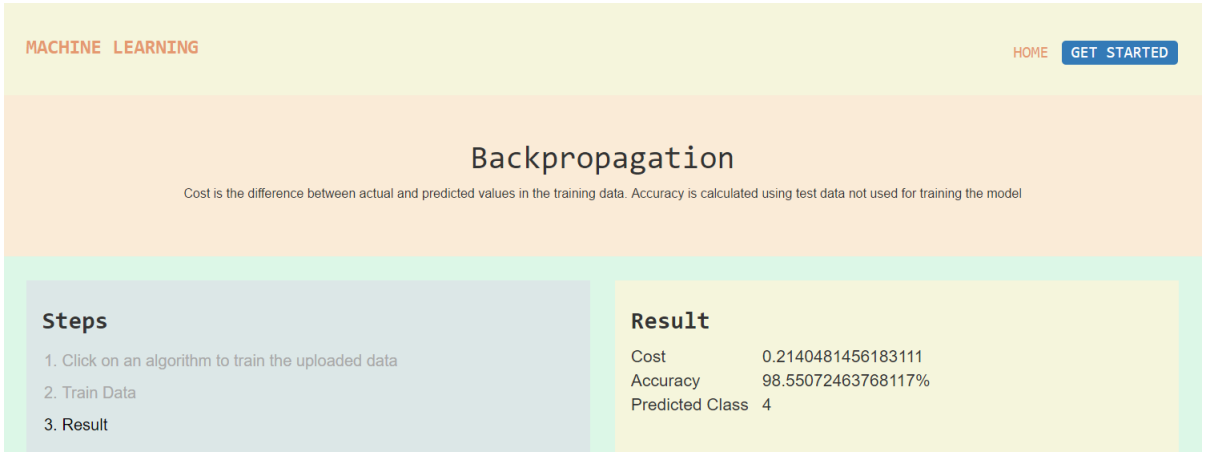

Figure 4.9: It shows predicted category of a tumor having attributes as highlighted in figure 4.7. The predicted output matches the actual targeted value.

Below given figure 4.10 motive is to show console section of backend of the web application when the model is trained with backpropagation and breast cancer data having 10 attributes. It shows decreasing value of cost in each iteration of gradient descent, accuracy, number of correctly predicted benign and malignant data instances.

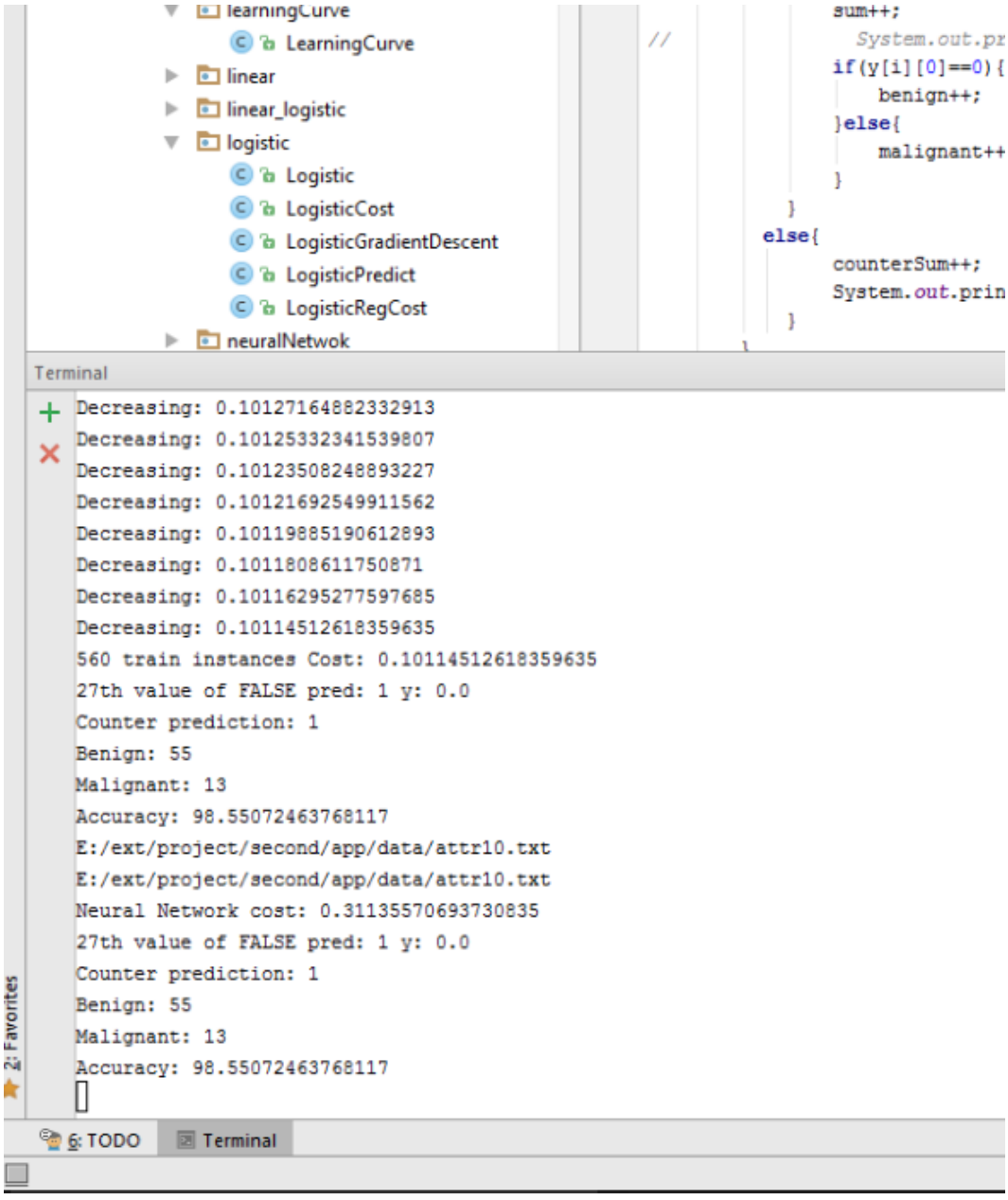

Figure 4.10: It shows decreasing value of cost in each iteration of gradient descent algorithm using back-propagation learning algorithm when trained with datasets having 9 attributes.

4.2 Breast Cancer Data Instance with 32 attributes

The other dataset used to test the web application is also a Breast Cancer dataset from UCI Machine Learning Repository http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29. This dataset has 32 attributes which is greater than the earlier dataset which had only 10 attributes.
The details of dataset is given below:

1. Number of instances: 569
2. Number of attributes: 32
3. Class value: 2 (Malignant), 4 (Benign)
4. 30 real value-valued input features: radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, fractal dimension, e.t.c.
5. missing attributes: none
6. Class distribution: 357 benign, 212 malignant

The given dataset has patient-id which has no effect on the learning process of the application so, it is removed from bot training and test data. In the table given below we see that 5 benign instances were not correctly classified when trained with 455 instances having 31 attributes using *logistic regression*. But, this prediction can be made better by decreasing η and increasing the number of iterations to train the network.

Class	Correctly Predicted Number of Instances	Actual Number of Instances
Benign	39	44
Malignant	13	13

Table 4.5: Number of correctly predicted instances using *Logistic Regression* on breast cancer data with 31 attributes

Similary, in table 4.6 we see that 4 benign instances were not correctly interpreted when trained with 455 instances having 31 attributes using *backpropagation*. This prediction can be improved by trying different η and increasing the number of iterations to train the network.

Class	Correctly Predicted Number of Instances	Actual Number of Instances
Benign	30	44
Malignant	13	13

Table 4.6: Number of correctly predicted instances using *Backpropagation* on breast cancer data with 31 attributes

The table given below shows accuracy % and cost related to each learning algorithm when tested on 70 data instances out of 699 total data instances. All the model in each of the algorithm is trained with 559 training instances.

Algorithm	η	Number of iterations	Cost	Accuracy %
Linear Regression	0.1	1200	0.1054	Not Applicable
Logistic Regression	0.1	1200	0.060	92.85%
Backpropagation	0.1	1200	0.121	94.64%

Table 4.7: Classification accuracy on classifying if the tumor is cancerous or not given a breast cancer data with 31 attributes

Below given figure 4.11 shows cropped content of a data with 31 attributes which has been used to train the model with logistic regression. The highlighted content is a set of features which belongs to category class "0" (column 1 value of highlighted row). This class "0" is a targeted value which needs to predicted by the trained model. This highlighted content is submitted as input in figure 4.12 in input box "*Features*". The predicted output for the highlighted set of features is shown in figure 4.13

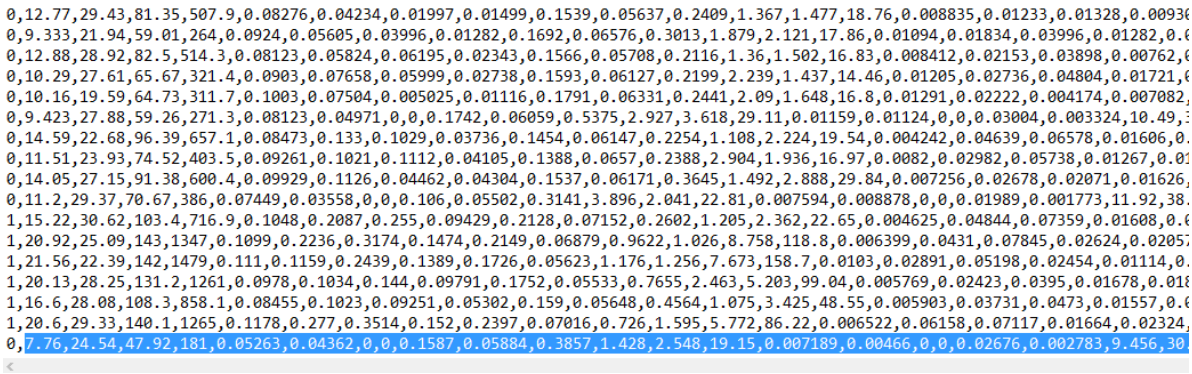

Figure 4.11: The highlighted features are passed as input to logistic regression to predict category of a tumor with that set of attributes.

In figure 4.12 the input box "*Features*" has the highlighted value shown in figure 4.11 so, its target is to predict that the tumor belong to category 0.

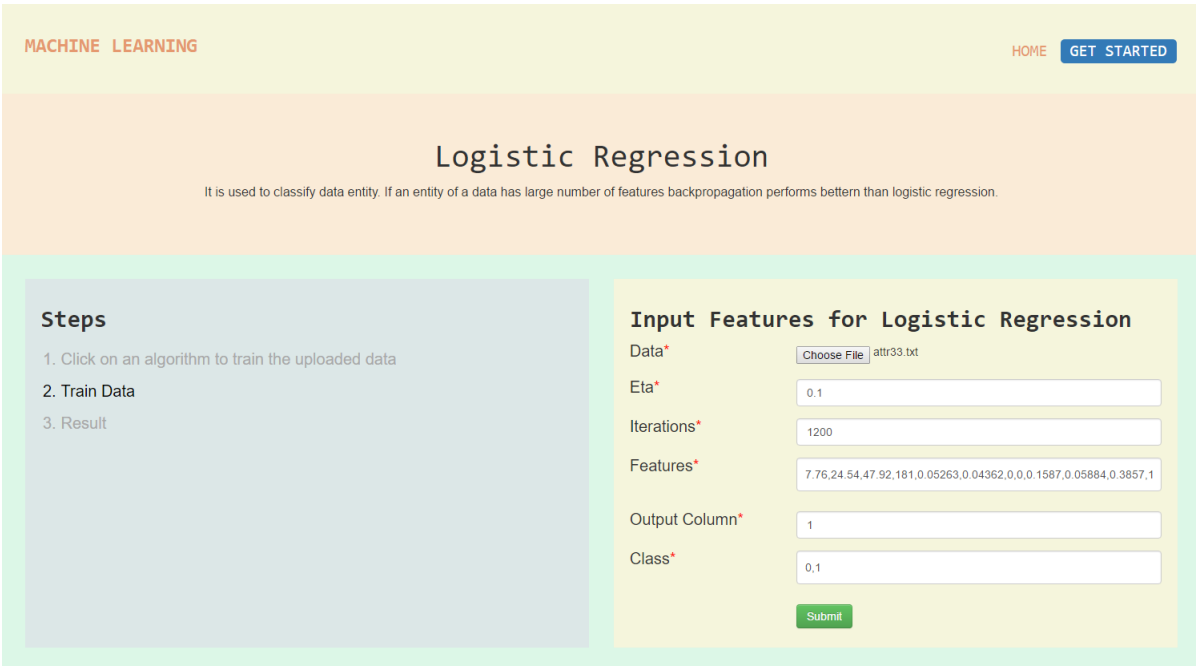

Figure 4.12: The highlighted features of a tumor in 4.11 is submitted as input in third input box. After information is submitted the application predicts if a tumor with that set of features is cancerours or not.

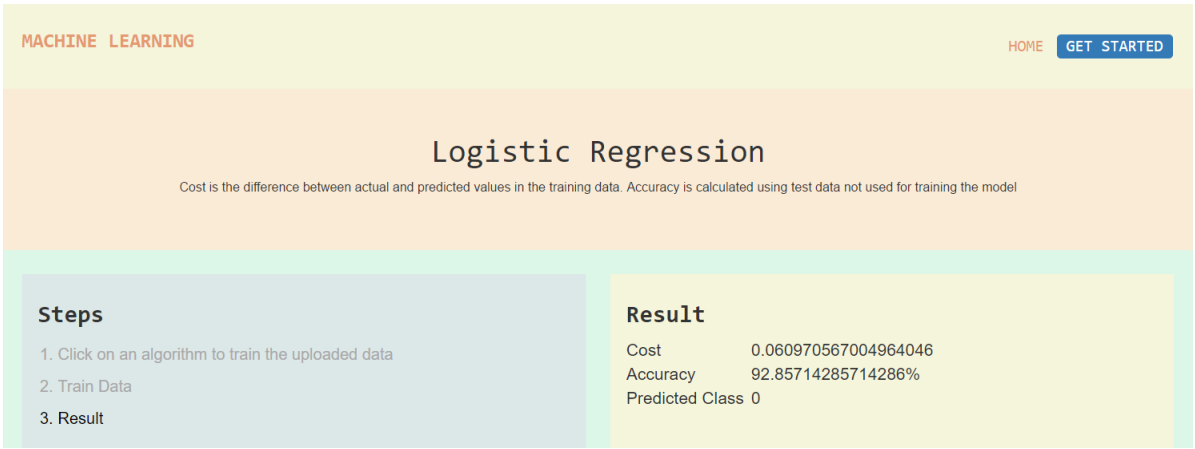

Figure 4.13: It shows predicted category of a tumor having a set of attributes as in figure 4.12. The predicted category matches the actual targeted value.

Below given figure 4.14 motive is to show console section of backend of the web application when the model is trained with logistic regression and breast cancer data having 31 attributes. The figure shows decreasing value of cost in each iteration of gradient descent, accuracy, number of correctly predicted benign and malignant data instances.

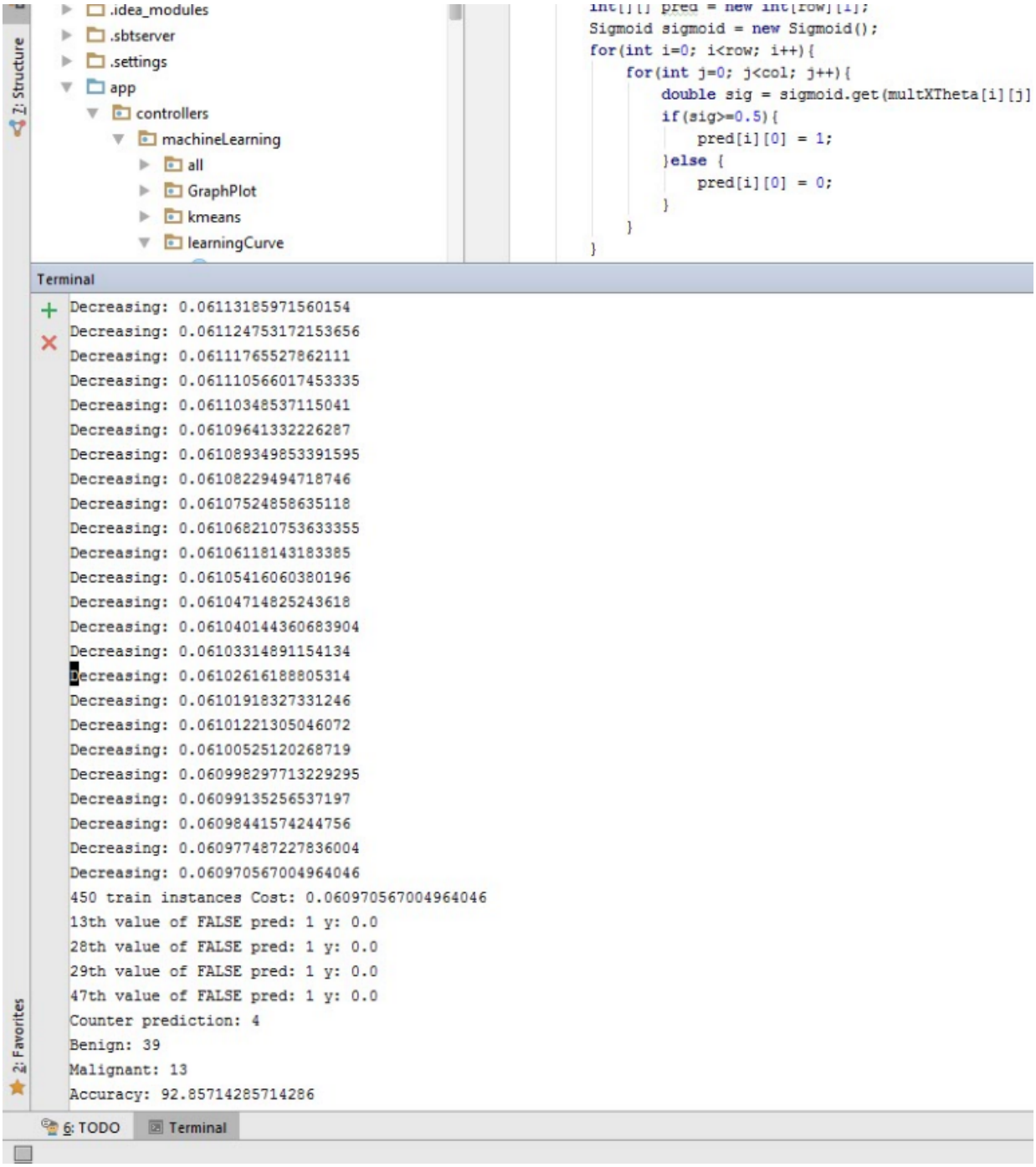

Figure 4.14: It shows decreasing value of cost in each iteration of gradient descent algorithm when data with 31 attributes is trained with logistic regression.

Below given figure 4.15 is a cropped content of data with 31 attributes. 90% of this data is used to train the model using backpropagation. Each row of the data represents a set of features of a breast tumor. The front-most value in a row represents the category of the tumor where "0" represents "benign" and "1" represents "malignant" tumor. We want to predict the category of a tumor having the feature highlighted in blue. So, the highlighted set of features in 4.15 is submitted in input box "*Features*" in figure 4.16.

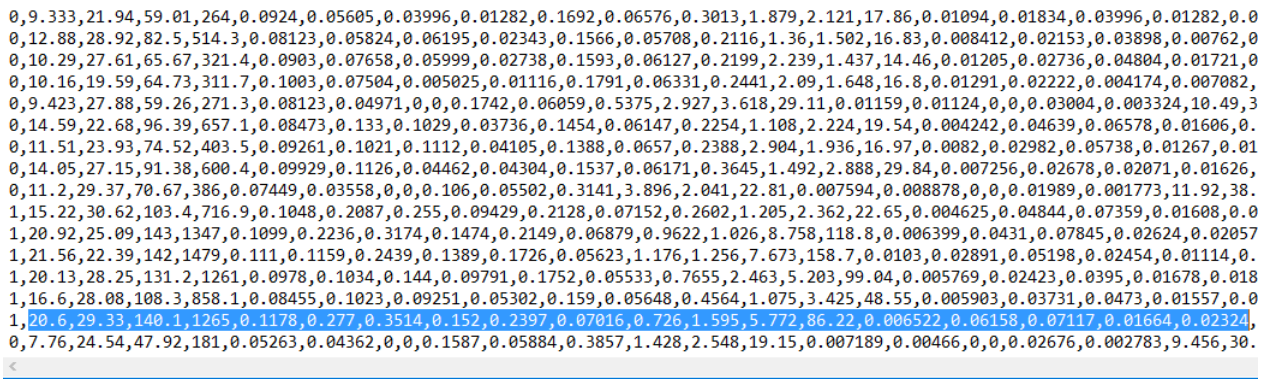

Figure 4.15: The figure shows cropped content of data used to train application using backpropagation algorithm. The highlighted content represents features of a tumor whose category needs to be predicted.

In figure 4.16 the input box "*Features*" has the highlighted value of figure 4.15 whose category needs to be predicted. The ultimate predicted category is shown in figure 4.16

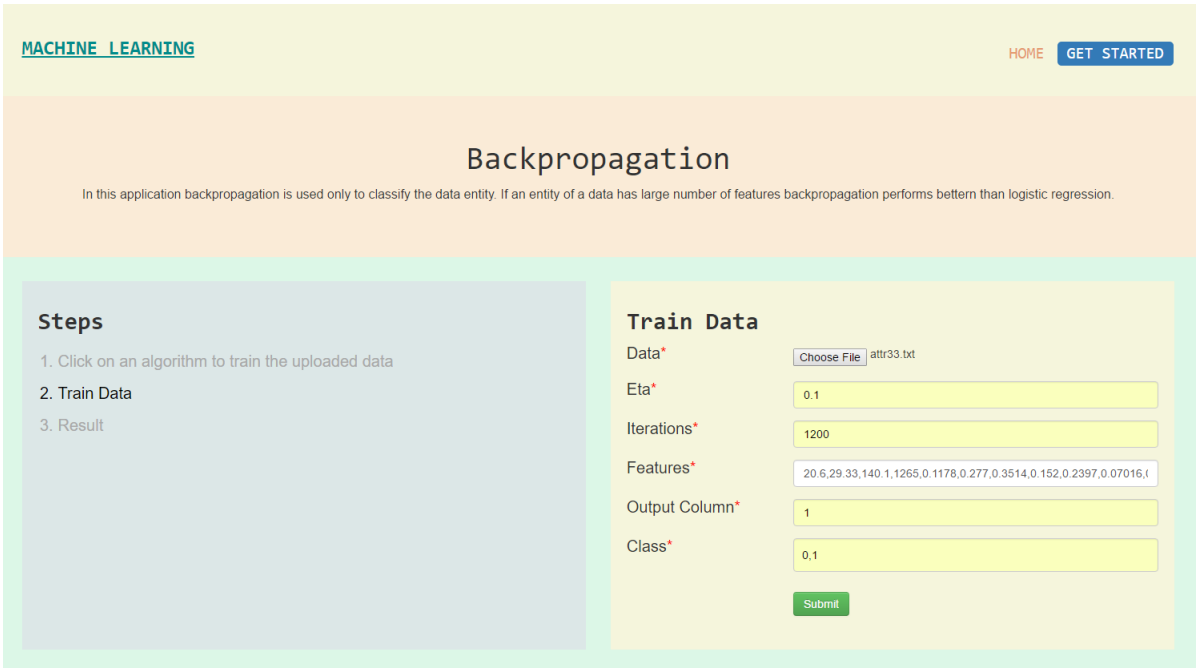

Figure 4.16: In this stage the highlighted feature of a tumor shown in 4.15 is passed as input to the application. Now, the application should predict category of a tumor with that set of attributes after information is submitted.

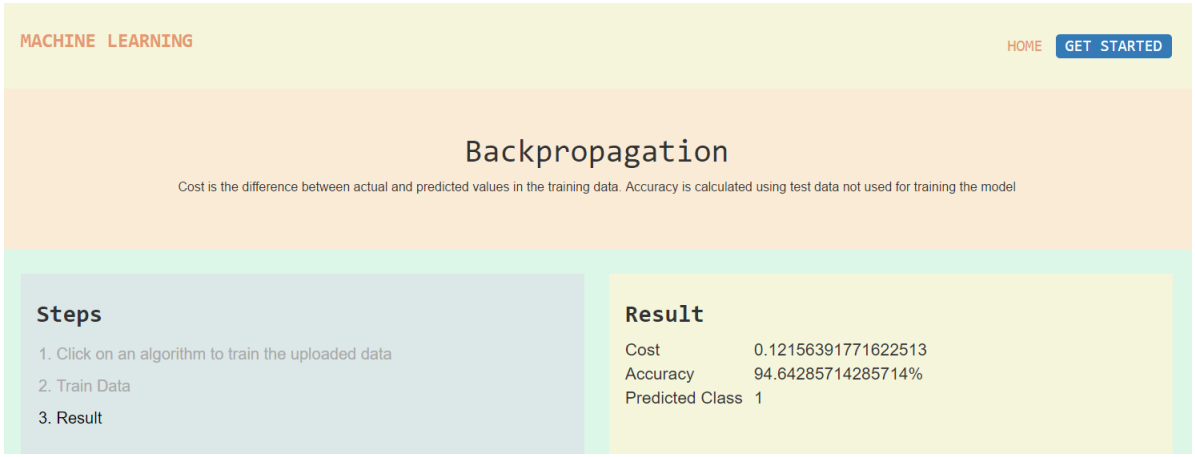

Figure 4.17: Predicted output for highlighted input in 4.15. The predicted value matches the target value.

Here figure 4.18 is a console section, backend of the web application. When the model is trained with backpropagation and breast cancer data having 31 attributes it shows decreasing value of cost in each iteration of gradient descent. The figure also shows accuracy, number of correctly predicted benign and malignant data instances.

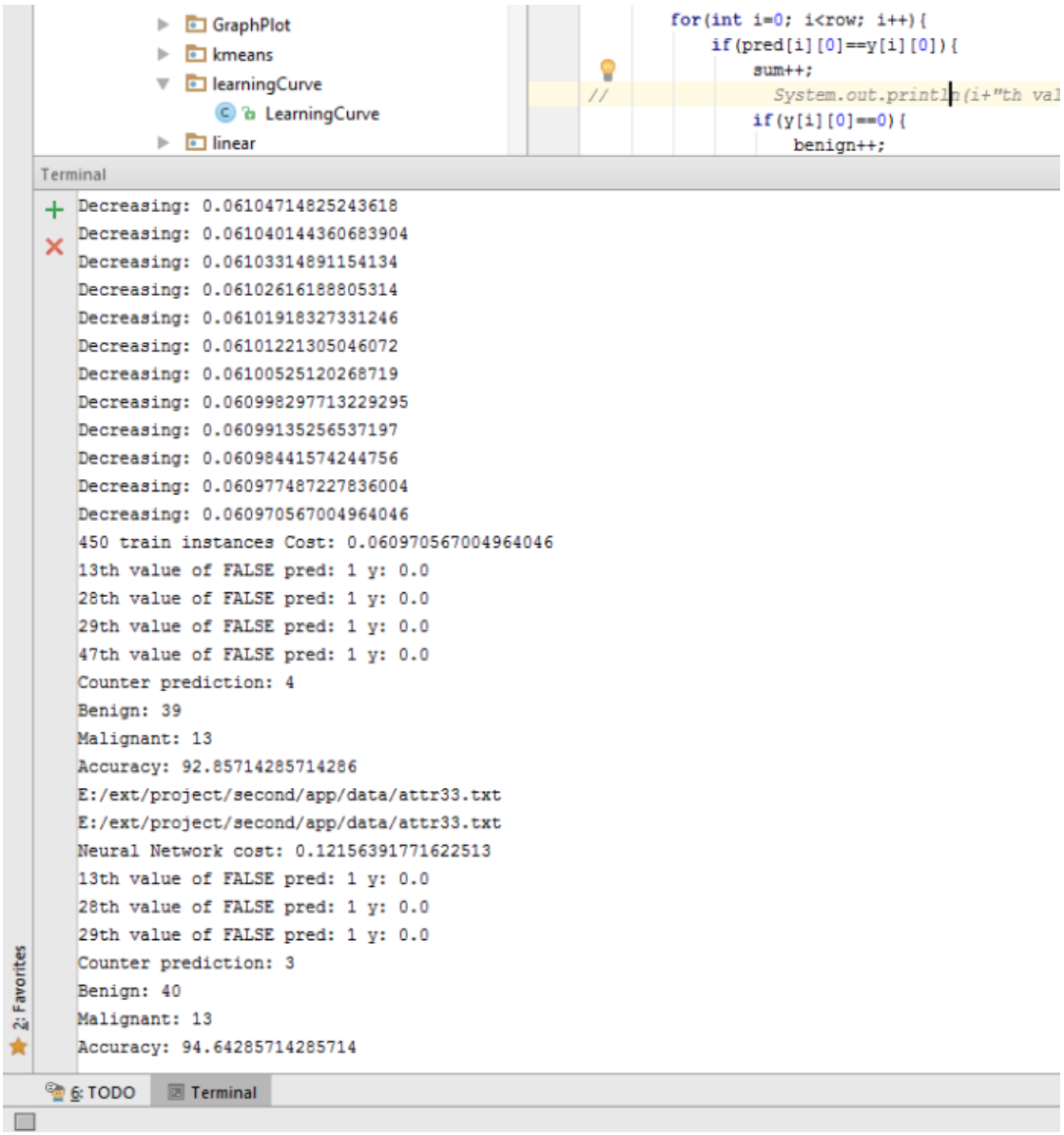

Figure 4.18: It shows decreasing value of cost in each iteration of gradient descent of algorithm while training a data with 31 attributes using backpropagation learning algorithm.

4.3 Student Performance Data

The application is tested on student performance data which is available publicly in UCI Machine Learning Repository http://archive.ics.uci.edu/ml/datasets/Student+Performance. The dataset has 30 attributes. The details of dataset is given below:

1. Number of instances: 649
2. Number of attributes: 30
3. missing attributes: none
4. Attributes: school, sex, age, address, family size, parent's living together or separate, mother's education, father's education, mother's job, father's job, reason to choose school, student's guardian, travel time from home to school, weekly study time, number of past classes failures, extra educational support, family educational support, extra paid classes within the course subject, extra-curricular activities, attended nursery school or not, wants to take higher education, internet access at home, in a romantic relationship, quality of family relationships, free time after school, going out with friends, workday alcohol consumption, weekend alcohol consumption, current health status and number of school absences.

The data set is divided into three groups using k-fold method

1. Training data which is 80% of total data
2. Cross validation which is 10% of total data and is total data - train data
3. Test data which is total data - training data - cross validation data

Student performance data was used to analyse and predict if students want to take higher education or not. Since, predicting if the students want to take higher education or not is related to categorization problem logistic regression and backpropagation are used for classification. Linear regression is used to predict continuous range value e.g. age of student. All the above listed 30 attributes were used to train the model created by linear regression, logistic regression and backpropagation.

In below given table 4.8 shows accuracy % obtained from linear regression, logistic regression and backpropagation.

Algorithm	η	Number of iterations	Cost	Accuracy %
Logistic Regression	0.2	500	0.18	92.18%
Backpropagation	0.2	500	0.37	90.625%

Table 4.8: Classification accuracy in categorizing students who want to take higher education

The below given table shows accuracy in classifying students who want to take higher education and students who don't want to take higher education. Both logistic and backpropagation correctly predicted all the instances that were related to students who want to take higher education. But, 5 data instances related to students who don't want to take higher education was misclassified when tested with a model that was trained with logistic regression and 6 data instances were misclassified when tested using a model that was trained with backpropagation.

Student wants to take higher education	Correctly Predicted Number of Instances	Actual Number of Instances
Yes	52	52
No	7	2

Table 4.9: Number of correctly predicted instances using *Logistic Regression* in students performance data with 30 attributes

Student wants to take higher education	Correctly Predicted Number of Instances	Actual Number of Instances
Yes	52	52
No	7	1

Table 4.10: Number of correctly predicted instances using *Backpropagation* in students performance data with 10 attributes

Chapter 5

Future Work

In this application as the size of data increases linear regression, logistic regression and backpropagation takes a bit long time to determine optimal weight(θ) [Ng]. The new technique to speed up computer is to keep the clock cycle fixed and double the number of cores which increases the speed of computer by double [CKL$^+$07]. Nowadays most of the computers are multi-core because increasing the clock rates to speed up computer has hit the power limit [CKL$^+$07]. Currently, the application is slow because linear regression, logistic regression and backpropagation takes time to converge to lowest cost and it doesn't utilize multi-core benefits which can make it faster.

So, in future I would like to utilize multi-core structure of the computer and make prediction faster. It has been said that parallelization technique increases the speed of learning by linear scale as the number of cores increases [CKL$^+$07]. And it has been approved that algorithms that fit Statistical Query model like logistic regression, backpropagation and kmeans can be changed to *summation form* which can easily be parallelized [CKL$^+$07]. The *summation form* doesn't change the architecture of the neural network but it is the exact implementation of above given algorithms.

Currently, neural network for backpropagation has only one hidden layer. I would like to add robustness in the system by automatically building the neural network with the optimum number of hidden layers as obtained from learning curve.

I would also like to add some unsupervised learning algorithm to cluster similar similar entities in a dataset and would like to add an interface to visualize these clusters. It is easy to visualize data with two features (x,y) as, they can be represented by scatter points in two dimensional plane with one feature being represented in x-axis and other feature being represented in y-axis. But, I would like to build an interface to visualize datasets with more than two features.

Appendix A

Resource of Data

The predictive accuracy of the developed machine learning application is checked using publicly available breast cancer and student performance data in UCI Machine Learning Repository

http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29

http://archive.ics.uci.edu/ml/datasets/Student+Performance

Bibliography

[Abb02] Hussein A Abbass. An evolutionary artificial neural networks approach for breast cancer diagnosis. *Artificial intelligence in Medicine*, 25(3):265–281, 2002.

[ACA+10] Turgay Ayer, Jagpreet Chhatwal, Oguzhan Alagoz, Charles E Kahn Jr, Ryan W Woods, and Elizabeth S Burnside. Comparison of logistic regression and artificial neural network models in breast cancer risk estimation 1. *Radiographics*, 30(1):13–22, 2010.

[Alp14] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2014.

[BHI+00] Endre Boros, Peter L Hammer, Toshihide Ibaraki, Alexander Kogan, Eddy Mayoraz, and Ilya Muchnik. An implementation of logical analysis of data. *Knowledge and Data Engineering, IEEE Transactions on*, 12(2):292–306, 2000.

[BM01] Indranil Bose and Radha K Mahapatra. Business data mininga machine learning perspective. *Information & management*, 39(3):211–225, 2001.

[Che93] Michael Chester. *Neural networks: a tutorial*. Prentice-Hall, Inc., 1993.

[CKL+07] Cheng Chu, Sang Kyun Kim, Yi-An Lin, YuanYuan Yu, Gary Bradski, Andrew Y Ng, and Kunle Olukotun. Map-reduce for machine learning on multicore. *Advances in neural information processing systems*, 19:281, 2007.

[CNCC15] DA Clifton, KE Niehaus, P Charlton, and GW Colopy. Health informatics via machine learning for the clinical management of patients. *Yearbook of medical informatics*, 10(1):38, 2015.

[Ebe14] Russell C Eberhart. *Neural network PC tools: a practical guide*. Academic Press, 2014.

[FL89] Scott E Fahlman and Christian Lebiere. The cascade-correlation learning architecture. 1989.

[Gar10] Jesse James Garrett. *Elements of user experience, the: user-centered design for the web and beyond*. Pearson Education, 2010.

[HHHH09] Simon S Haykin, Simon S Haykin, Simon S Haykin, and Simon S Haykin. *Neural networks and learning machines*, volume 3. Pearson Education Upper Saddle River, 2009.

[HN89] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural Networks, 1989. IJCNN., International Joint Conference on*, pages 593–605. IEEE, 1989.

[KAP] Chetna Kaushal, Aashima Arya, and Shikha Pathania. Integration of data mining in cloud computing.

[Kon01] Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, 23(1):89–109, 2001.

[Men02] Scott Menard. *Applied logistic regression analysis*, volume 106. Sage, 2002.

[MP01] George D Magoulas and Andriana Prentza. Machine learning in medical applications. In *Machine Learning and its applications*, pages 300–307. Springer, 2001.

[Ng] Andrew Ng. Coursera machine learning.

[Ras06] Carl Edward Rasmussen. Gaussian processes for machine learning. 2006.

[RG02] Chotirat Ann Ratanamahatana and Dimitrios Gunopulos. Scaling up the naive bayesian classifier: Using decision trees for feature selection. 2002.

[Rip07] Brian D Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.

[RM98] Russell D Reed and Robert J Marks. *Neural smithing: supervised learning in feedforward artificial neural networks*. Mit Press, 1998.

[Roj13] Raúl Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

[RP04] Deborah E Rosen and Elizabeth Purinton. Website design: Viewing the web as a cognitive landscape. *Journal of Business Research*, 57(7):787–794, 2004.

[RTB67] Jon Reed, Robert Toombs, and Nils Aall Barricelli. Simulation of biological evolution and machine learning: I. selection of self-reproducing numeric patterns by data processing machines, effects of hereditary control, mutation type and crossing. *Journal of theoretical biology*, 17(3):319–342, 1967.

[Saj06] Paul Sajda. Machine learning for detection and diagnosis of disease. *Annu. Rev. Biomed. Eng.*, 8:537–565, 2006.

[Set00] Rudy Setiono. Extracting m-of-n rules from trained neural networks. *Neural Networks, IEEE Transactions on*, 11(2):512–519, 2000.

[Shn03] Ben Shneiderman. *Designing the user interface*. Pearson Education India, 2003.

[SL97] Rudy Setiono and Huan Liu. Neurolinear: From neural networks to oblique decision rules. *Neurocomputing*, 17(1):1–24, 1997.

[Spe91] Donald F Specht. A general regression neural network. *Neural Networks, IEEE Transactions on*, 2(6):568–576, 1991.

[TG96] Ismail Taha and Joydeep Ghosh. Characterization of the wisconsin breast cancer database using a hybrid symbolic-connectionist system. In *Proceedings of the Intelligent Engineering Systems through Artificial Neural Networks Conference (ANNIE96)*. Citeseer, 1996.

[Wei05] Sanford Weisberg. *Applied linear regression*, volume 528. John Wiley & Sons, 2005.

[WF05] Ian H Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

Curriculum Vitae

Graduate College
University of Nevada, Las Vegas

Chandani Shrestha

Degree:

Bachelor of Engineering in Computer Science 2012
Kathmandu University

Thesis Title: A Web Based User Interface For Machine Learning Analysis Of Heath And Education Data

Thesis Examination Committee:

Chairperson, Dr. Fatma Nasoz, Ph.D.
Committee Member, Dr. Kazem Taghva, Ph.D.
Committee Member, Dr. Justin Zhan, Ph.D.
Graduate Faculty Representative, Dr. Qing Wu, Ph.D.