Predicting the Survival of Titanic Passengers using backpropagation Algorithm

Name: Sukriti Mishra
SJSU Id: 014580696

In this report, I describe the whole process of creating a deep learning algorithm on the famous Titanic dataset, available on the Kaggle "Titanic: Machine Learning from Disaster competition". It provides dataset that contains information on the fate of passengers on the Titanic, summarized according to economic, status (class), sex, age and survival. I applied an Artificial Neural Network with 10 hidden layers was also used to compare the scores, accuracy and time taken for both training and test data. The format of output is binary i.e. 1: Survival; 0: Non-Survival

ANN mimics like the human brain, with neuron nodes interconnected like a web. The human brain consists of hundreds of billions of cells called neurons. Each neuron is made up of a cell body that is responsible for processing information by carrying information towards (inputs) and away (outputs) from the brain. ANN has hundreds or thousands of artificial neurons called processing units, which are interconnected by nodes. These processing units are made up of input and output units. The input units receive various forms and structures of information based on an internal weighting system, and the neural network attempts to learn about the information presented to produce one output report. Just like humans need rules and guidelines to come up with a result or output, ANNs also use a set of learning rules called backpropagation, to perfect their output results. Backpropagation consists of doing a feedforward operation, comparing the output of the model with the desired output, calculating the error, running the feedforward operation in backwards direction to spread the error to each of the weights. Use this update the weights and gets a better model. Continue till we have a model that is good. For this project, I have applied the backpropagation algorithm to predict the survival of titanic passengers. The pseudocode for backpropagation I used for this project is explained below:

function *BACK-PROP-LEARNING (examples, network)* ***returns*** *a neural network*
  inputs *examples, a set of examples, each with input vector* ***x*** *and output vector* ***y***
    network, a multilayer network with L layers, weights $w_{i,j}$*, activation function g*
  local variables*: Δ, a vector of errors, indexed by network node* ***repeat***
    for each *weight* $w_{i,j}$ *in network* ***execute***
      $w_{i,j} \leftarrow$ *a small random number*
    for each *example* (***x***, ***y***) ***in*** *examples* ***execute***
      / Propagate the inputs forward to compute the outputs */*
      for each *node i in the input layer* ***execute***
        $a_i \leftarrow x_i$
      for $l = 2$ ***to*** L ***do***
        for each *node j in layer l* ***execute***
          $in_j \leftarrow \Sigma_i w_{i,j} a_i$
          $a_j \leftarrow g(in_j)$
      / Propagate deltas backward from output layer to input layer */*
      for each *node j in the output layer* ***execute***
        $\Delta[j] \leftarrow g'(in_j) \times (y_i - a_j)$
      for $l = L - 1$ ***to*** 1 ***do***
        for each *node i in layer l* ***execute***
          $\Delta[i] \leftarrow g'(in_i) \Sigma_j w_{i,j} \Delta[j]$
      / Update every weight in network using deltas */*

for each *weight* $w_{i,j}$ *in network* ***execute***
$w_{i,j} \leftarrow w_{i,j} + \alpha \times a_i \times \Delta[j]$
until *some accuracy rate is satisfied*
return *network*

Dataset: The dataset is divided into two files: train.csv and test.csv. We use training data to train the models whereas the test data (hidden data) to test our models. The training-set has 891 examples and 11 features + the target variable (survived). 2 of the features are floats, 5 are integers and 5 are objects whereas test data has

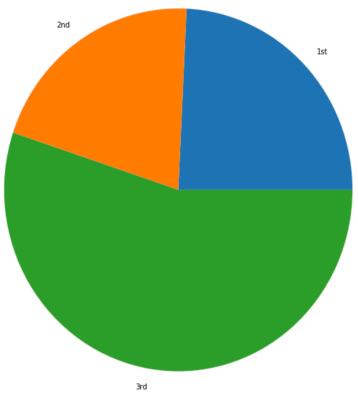
Data Exploration:
Exploratory data analysis involves identification of variables, univariate (bar plots and histograms) and bivariate data analysis (scatter plots and box plots), outliers' detection and feature engineering.

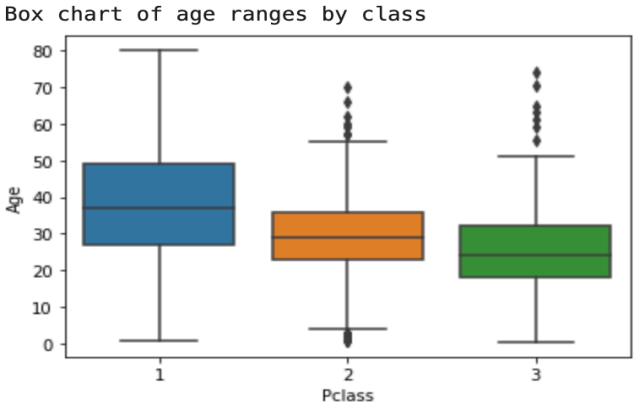
Data preprocessing steps includes data cleaning, data transformation, missing values imputation, data normalization, feature selection, and other steps depending on the nature of the dataset. There are three columns that have the most missing values or values set to 0. The columns include:

1. Age (Substituted missing value by median)
2. Fare (Substituted missing value by median)
3. Embarked (Substituted missing value by the most frequently occurring source which is 'S')

Data transformation is the process of converting data from one format into another format, according to the need of the project. Data transformation is interpretative to data integration and data management. Data transformation can include a range of tasks such as convert data types, clean data by removing nulls or duplicate data or perform aggregations to make it suitable with other data, depends on the requirements of our project.

Data Visualization: The raw data visualization includes the visualization by plotting the correlation matrix to find out the correlation of different features with each other, the Percentage of survivals vs sex by plotting the bar graph, by plotting pie chart to find out the of numbers passengers in each class and by calculating the target values count (1-342, 0-549) and so on. to get the clear picture of the dataset and make it better organized and drew some conclusion such as number of passengers was maximum in the third class, females survival rate was higher than the males, the survival rate was maximum for the first-class passengers.





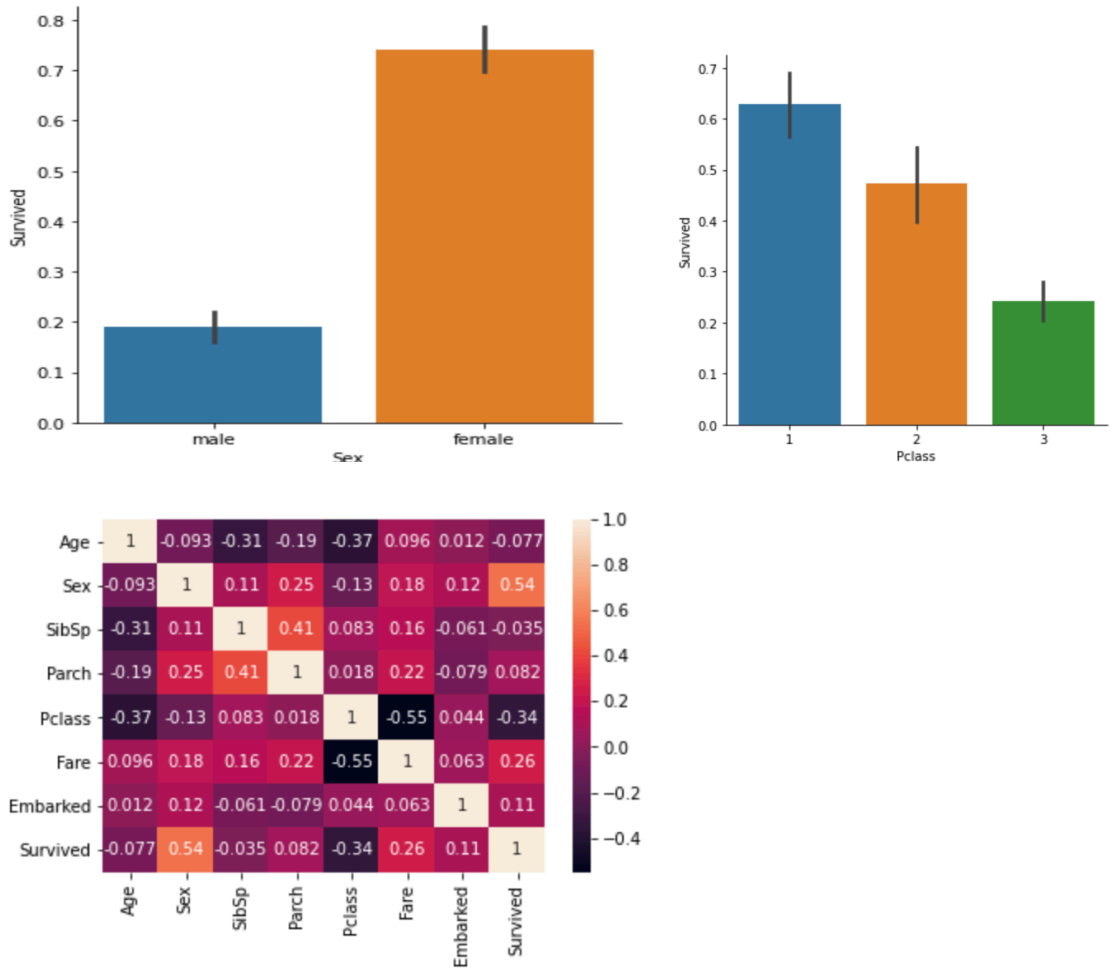


Figure 1-5 from top left to bottom left: Pie-chart(No. of passengers in each class), Box plot,(age range vs Passenger class, bargraphs(sex vs survived and survived vs passenger class) and correlation matrix

Feature Engineering:

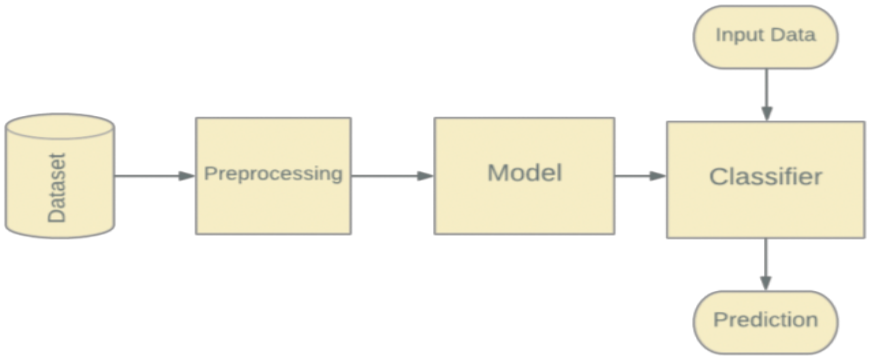
Feature selection is one of the most important tasks to get higher accuracy. Initially, I tried with all the features and got 73% accuracy that was not so great. Then I plotted the bar graph to find out the relationship strength between target and features i.e. (target vs features) such as Passenger class vs survived, Embarked vs survived, Age vs Survived, parch vs survived, and Fare vs Survived. Additionally, I plotted the heatmap to find out the relationship between the features. On the basis of graphs, I observed that that Sex, Pclass, Fare and Embarked are associated with Survived.

I performed the feature selection for the improvement and better performance of the model. In feature selection, we can include domain knowledge, interaction features such as average mean, average sum, etc. of the attributes, and by removing unused features. We can also replace redundant features by adding some domain-specific other features during feature selection. In this project, the titanic dataset has 12 features, but I preferred to use a subset of 10 of them with two of the features derived from other columns in the dataset. The features include:

1. Passenger Class

2. Sex (Converted to binary values where male = 0 and female = 1)
3. Age (Categorized into groups based on range values)
4. Sibling/Spouse
5. Parent/Children
6. Title (Extracted from Name column)
7. Fare (Categorized into groups based on range values)
8. Embarked (Categorized based upon the symbols)
9. Family Size (Sum of Sibling/Spouse + Parent/Children + 1)
10. Has Cabin (Derived from Cabin column whether a passenger has a cabin)

Data Modelling and Validation:



Simplified representation of model

The accuracy and performance of any model depends on the algorithms and the quality of the dataset. Based upon the backpropagation that is described above, we implemented the workflow to create a neural network. Initially, we had set up k-fold cross validation on the model with number of folds = 5, learning rate = 0.1, hidden layers = 5, and all features from the dataset excluding cabin, fare, name, ticket and passenger Id. From the cross validations, I got 5 neural networks out of which I selected the one with the highest accuracy that was 87% However, after running the model multiple times this way we could only get an accuracy score of around 75%.

As next steps, I performed some feature extraction based upon the exploratory data analysis:

1. Derived family size by summing up sibsp and parch columns
2. Derived whether a passenger had a cabin
3. Categorized fare based upon range value
4. Extracted title from name column

In addition to six features earlier, I used the above 4 features, increased hidden layers to 10, varied learning rate between .004 and .07, removed cross validation from the workflow. After running multiple times, I was able to get a model with learning .04 that could predict the survival rate with approximately 78.71% accuracy. So, I can say to increase the accuracy rate of any model, feature selection and feature extraction play a significant role.