HR Prediction: Predicting Whether an Employee is looking for a new job

Submitted By

Md.Ashraful Haque 160204101
160204101@aust.edu

Shafkat Bin Shihab 160204106
160204106@aust.edu

Sabeek Bin Sayeed 160204108
160204108@aust.edu

Introduction:

Human Resource Management is the process of recruiting, selecting,employees, appraising the performance of employees and providing benefits, welfare and healthy measures in compliance with orders of the concerned High Court and Supreme Court, if any. Human resource management is meant for proper utilisation of available skilled workforce and also to make efficient use of existing human resources in the organisation. Humans are crucial assets for any organisation, although today many tasks have been handed over to the artificial intelligence but they lack judgement skills which cannot be matched with the human mind.

Machine learning can be applied in different sectors of HRM. For example, machine learning can be applied in hiring employees. It is possible to discover which software discovers that a certain job ad website yields more successful hires. It is also seen that people who use a certain type of social media turn out to be better employees. A software that utilizes machine learning is the only kind of entity that can hope to analyze all this data and find the patterns. A human HR professional could never do something like this. Again in Employee Attrition Machine learning plays a significant role. It is impossible for a human HR professional to do comprehensive analysis of people's statements, questions, intentions and decisions that would lead to employee attrition. For a piece of software based on machine learning, certain patterns become identifiable. Using this pattern Machine learning plays an important role in Employee Attrition.

Our model will be able to predict whether an employee will look for a new job or stay in the current job. In our case some Data scientists are wanting to be hired by a company which is active in Big Data and Data Science . Many people sign up for training. The candidates that are serious to work for the company after the training are being identified by the company. From different information like demographics, education, experience it is possible to detect whether the employee is going to stay in the company or look for a new job. And by doing that the HRM job becomes really easy and it saves a lot of time.

The factors that lead a person to leave their current job for HR research are being designed in this dataset. The probability of a candidate to look for a new job or will work for the company, as well as interpreting affected factors on employee decision will be predicted by the model that uses the current credentials, demographics, experience data .

Related Works:

Some of the related task in this field has been done in kaggle which are described below-

In one of the tasks titled '**Predict who will move to a new job**' by Siti Khotijah[1] visualization of the features was done and comparison of the features was shown that might influence moving to a new job. Also the max, min and mean of the features was shown. Here in the preprocessing steps the nan values were replaced with the average of columns. Here correlation in data was measured using Correlation coefficients. Correlation coefficients are used to measure how strong a relationship is between two variables.The formulas return a value between -1 and 1, where 1 indicates a strong positive relationship. And -1 indicates a strong negative relationship. In this model **CatBoost** algorithm is used . It is an algorithm for **gradient boosting on decision trees**. Finally they got a Train AUC score of 0.8981 and Test AUC score of 0.7972.

In another task titled '**Who will leave a job?"** by Ankit Gupta[2] AUC curve was used to show the results. Transforming Categorical features into numerical features was done after reading the dataset. K nearest algorithm is used to fill the missing data. In the notebook they used **Bayesian optimization with a gaussian process**. According to them the best metric for this binary classification problem would be Area Under the ROC Curve (AUC). They used AUC since it combines both precision and recall. They also used Extreme Gradient Boosting (XGBoost) Classifier in this notebook. Finally they got a Train AUC score of 0.9702 and Test AUC score of 0.9213.

Another task titled **'predict who is job-seeking and who is not' by** Joshua Swords[3] where the result was found implementing **logistic regression using SMOTE**. At first the dataset was read and then dealt with missing values. It was shown that the dataset was not balanced properly .Exploratory visualization of the data was shown. Different analysis of the dataset was done and it was shown based on the characteristics of those who are looking for a job. Finally the accuracy was found after trying different algorithms . Highest accuracy of 78.8% was found by applying Random Forest technique. After that they also experimented implementing SMOTE Finally they found that logistic regression using SMOTE gave the highest recall value and a high accuracy overall.

Project Objectives:

Every modern organisation desires to have skilled and competent people to make their organisation competent and best. Among the five Ms of management, HRM deals about the first M, which is men. It is the most challenging to manage and the success of a company mainly depends on how it is managed. If Deep Learning can be used in this sector then it can save a lot of time and effort and also do the task more precisely. For example in our case It is important for the company to know which candidates are serious to work for the company or they are just looking for a new employment. It helps the company to reduce cost and time as well as to plan the courses and categorization of candidates. As a result the HRM can be more efficient and can do their work with minimum effort. So if we can detect the employees who are willing to work then it will help the company a lot. .A software that utilizes machine learning is the only kind of entity that can hope to analyze all this data and find the patterns. A human HR professional could never do something like this.

The flowchart of our process is given below-

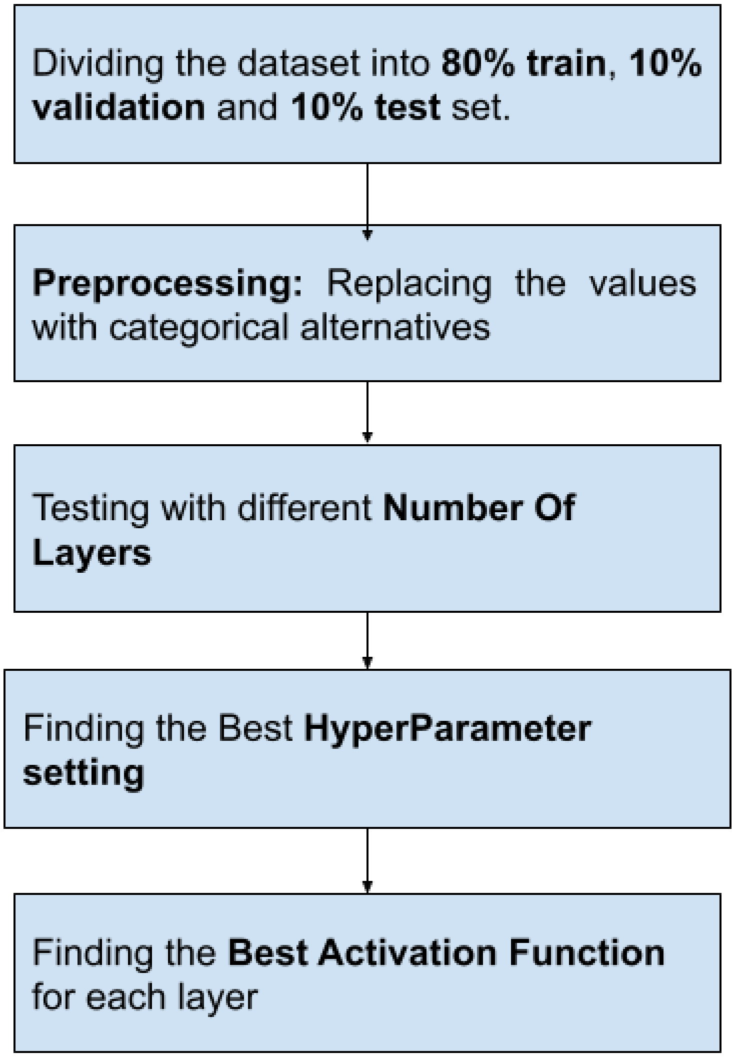

Dividing the dataset into 80% train, 10% validation and 10% test set.
Preprocessing: Replacing the values with categorical alternatives
Testing with different Number Of Layers
Finding the Best HyperParameter setting
Finding the Best Activation Function for each layer

Figure : Activity flowchart

Sample Input and Output:

The **input** of the Model are-

- **enrollee_id** : 33241

- **City**: city_115
- **city_development_index** : 0.789
- **gender**: Male
- **relevent_experience**: No relevant experience
- **enrolled_university**: Full time course
- **education_level**: Graduate
- **major_discipline** : Humanities
- **experience**: >20
- **company_size**: 50-99
- **company_type** : Funded Startup
- **last_new_job**: >4
- **training_hours**: 47

The **Output** of the Model are-

1.0 if the employee is predicted to be the one who is **looking for a new job**
0.0 if the employee is predicted to be the one who is **not looking for a new job**

Methodologies:

PreProcessing:

The preprocessing steps of the dataset are discussed below-

1. Downloaded the dataset from the site kaggle. Then import all the necessary libraries. After extracting the downloaded dataset, the dataset is read and stored on a dataframe.
2. The dataset had **three** types of data: **int64**, **float64**, **object**.
3. Then we convert the dataframe into a **dictionary** to keep track of the data.
4. The dataset has a total of **14** columns.
5. Then we found the **unique values** of each column of the dataframe/dataset. We used the **unique()** function to find the unique values.

   Then we replace those unique values if their type is string with either integer or float values for **mathematical convenience**. And that's the reason for finding the unique values. We also replace the **NaN** values with the **mean**(in most cases) values of unique values. The reason to replace or eliminate the NaN values is to aid mathematical convenience. We used the **replace()** function to to replace the values.

   Then save the values to a dictionary for tracking purposes. We counted the amount of values of each unique value using **value_counts()** function. Finally we checked if there is any NaN value left in the dataframe just in case using **isna().sum()**. The goal is to find the value 0.

6. **city_development_index:** For the city_development_index column the same procedure as before but we did not change the type of the values here as they are already in floating point number.
7. **gender:** For the gender column we replace the unique values: **'Male'**, **'Female'**, **'Other'**, **NaN** with **1, 2, 3, 1.5** respectively. We replaced **NaN** values with **1.5** between 1 and 2 which are male and female respectively as the majority lies between males and females.
8. **relevent_experience:** The column relevent_experience has two attributes. They are '**Has relevant experience**', '**No relevant experience**' replaced by **1, 0** respectively. There is no NaN value in this column as we found out about it using the unique() function.
9. **enrolled_university:** For the enrolled_university column we replace the unique values: '**no_enrollment**', '**Full time course**', '**Part time course**', **NaN** with **1, 2, 3, 4** respectively. We also change the data type using astype() function to integer.
10. **education_level:** In the education_level column we type cast the attribute from string to int and set the NaN value with the mean value of the unique values. Here **NaN** value is **3**.
11. **major_descipline:** In the major_descipline column we type cast the attribute from string to float and set the NaN value with the mean value of the unique values. Here **NaN** value is **3.5**.
12. **experience:** In the experience column there are a lot of unique values.
    Here we considered '**<1 as 0**' and '**>20 as 21**' and found the **mean** value of them which is **11** and replaced the **NaN** value with **11**. We change the attribute type from string to integer.
13. **company_size:** In the **company_size** column we type cast the attribute from string to float and set the NaN value with the mean value of the unique values. Here **NaN** value is 4.5.
14. **company_type:** In the company_type column we type cast the attribute from string to float and set the NaN value with the mean value of the unique values. Here NaN value is **3.5**.
15. **last_new_job:** Here in unique values we considered **'never'** as **0**. And find the **NaN** value calculating their **mean**. Here **NaN** value is **4**.
16. **training_hour:** There is no need to modify training_hour and target column of the dataframe/dataset.
17. Then we loaded the modified dataframe to a new csv file uploaded to drive so that we can use it any time we want.

Experiments:

Dataset:

This dataset is designed to understand the factors that lead a person to leave their current job for HR research. By model(s) that uses the current credentials,demographics,experience data you will predict the probability of a candidate to look for a new job or will work for the company, as well as interpreting affected factors on employee decision. The dataset is imbalanced. Most features are categorical, some with high cardinality. The features of the dataset are-

- enrollee_id : Unique ID for candidate
- city: City code
- city_ development _index : Development index of the city
- gender: Gender of candidate
- relevent_experience: Relevant experience of candidate
- enrolled_university: Type of University course enrolled if any
- education_level: Education level of candidate
- major_discipline :Education major discipline of candidate
- experience: Candidate total experience in years
- company_size: No of employees in current employer's company
- company_type : Type of current employer
- last_new_job: Difference in years between previous job and current job
- training_hours: training hours completed
- target: 0 – Not looking for job change, 1 – Looking for a job change

Features	Number of Unique Values	Number of NaN Value	Unique values
enrollee_id	19158	0	8949, 29725, 11561, ..., 24576, 5756, 23834
city	123	0	'city_103', 'city_40', ……,'city_31', 'city_171'
city_development_index	93	0	.92, .776,...,.807,.664
gender	4	4508	'Male', 'Female', 'Other'
relevent_experience	2	0	'Has relevant experience', 'No relevant experience'
enrolled_university	4	386	'no_enrollment', 'Full time course', 'Part time course'

education_level	5	52	'Graduate', 'Masters', 'High School', nan, 'Phd', 'Primary School'
major_discipline	7	2813	'STEM', 'Business Degree', 'Arts', 'Humanities', 'No Major', 'Other'
experience	22	729	'>20','15', '5', '<1','11', '13', '7', '17', '2', '16', '1', '4', '10', '14', '18', '19', '12', '3', '6', '9', '8', '20'
company_size	9	5938	'50-99', '<10', '10000+', '5000-9999', '1000-4999', '10/49', '100-500', '500-999'
company_type	7	6140	'Pvt Ltd', 'Funded Startup', 'Early Stage Startup', 'Other', 'Public Sector', 'NGO'
last_new_job	6	40	'1', '>4', 'never', '4', '3', ‘2’
trainig_hours	241	0	36,47,83,....270,286
target	2	0	1.0, 0.0

Evaluation Metric:

Evaluation metrics are used to measure the quality of the statistical or machine learning model. Evaluating machine learning models or algorithms is essential for any project. There are many different types of evaluation metrics available to test a model. These include classification accuracy, cross entropy loss, confusion matrix, and others.

In this case we used accuracy and cross entropy loss.

Accuracy:

Accuracy is the number of correctly predicted data points out of all the data points. More formally, it is defined as the number of true positives and true negatives divided by the number of true positives, true negatives, false positives, and false negatives. We can state the equation of accuracy as below-

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

Cross-entropy Loss:
Cross-entropy is commonly used in machine learning as a loss function.Also called logarithmic loss, log loss or logistic loss. Each predicted class probability is compared to the actual class desired output 0 or 1 and a score/loss is calculated that penalizes the probability based on how far it is from the actual expected value. The penalty is logarithmic in nature yielding a large score for large differences close to 1 and small score for small differences tending to 0. Cross-entropy loss is used when adjusting model weights during training. The aim is to minimize the loss, i.e, the smaller the loss the better the model. A perfect model has a cross-entropy loss of 0. Cross entropy defined as-

$$L_{\mathrm{CE}} = -\sum_{i=1}^{n} t_i \log(p_i), \quad \text{for n classes,}$$

where t_i is the truth label and p_i is the Softmax probability for the i^{th} class.

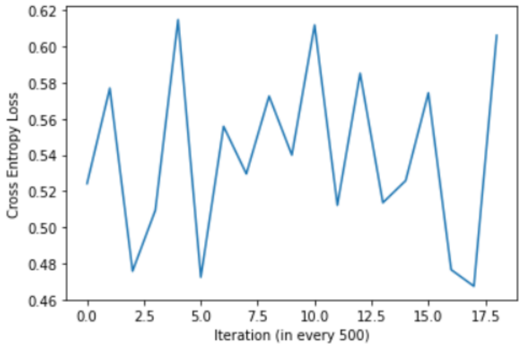
Results:(49 setups of different Hyperparameters and other variations)

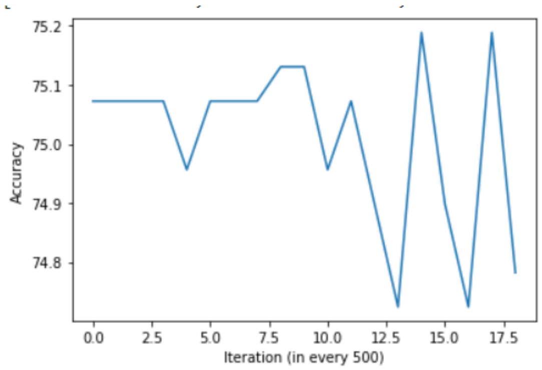
- **Setup 1 Neural Network with 24 nodes and 2 hidden layers with ReLU Activation**

Hyper Parameters	Values
Batch Size	100
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.001
Number of Nodes	24

Number of hidden layers	2

Visualization:





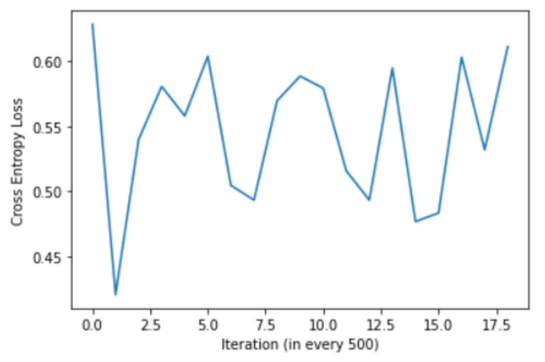
Here we can see the best accuracy we got is upto 75.17

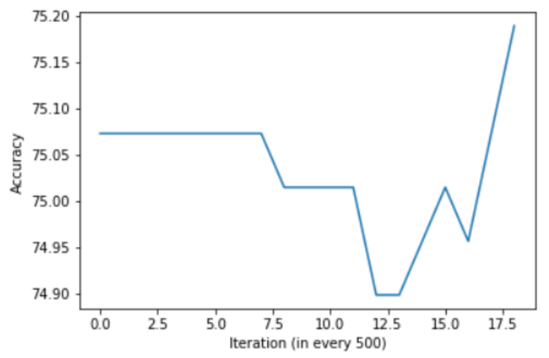
- ### Setup 2 Neural Network with 24 nodes and 3 hidden layers with ReLU Activation

Here we increase the number of hidden layers by 1. Here the number of hidden layers is 3.

Hyper Parameters	**Values**
Batch Size	100
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.001
Number of Nodes	24
Number of hidden layers	3

Visualization:





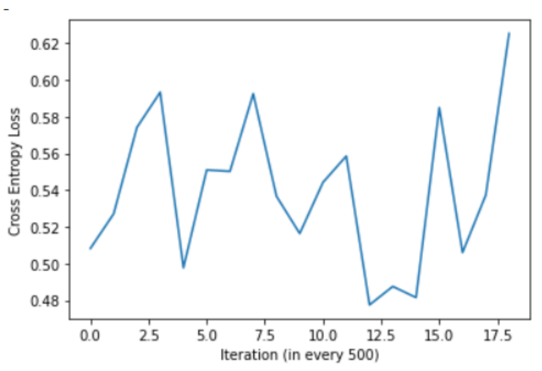
Here we can see the best accuracy we got is upto 75.07

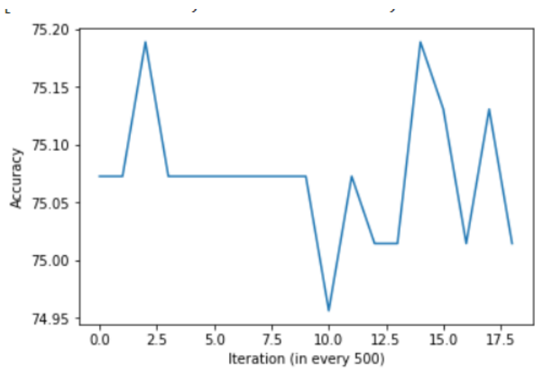
- **Setup 3 Neural Network with 100 nodes and 5 hidden layers with ReLU Activation**

Here the number of hidden layers is 5.

Hyper Parameters	**Values**
Batch Size	100
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.01
Number of Nodes	100
Number of hidden layers	5

Visualization:





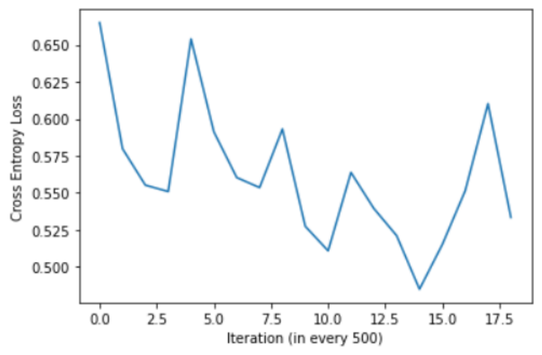
Here we can see the best accuracy we got is upto 75.18

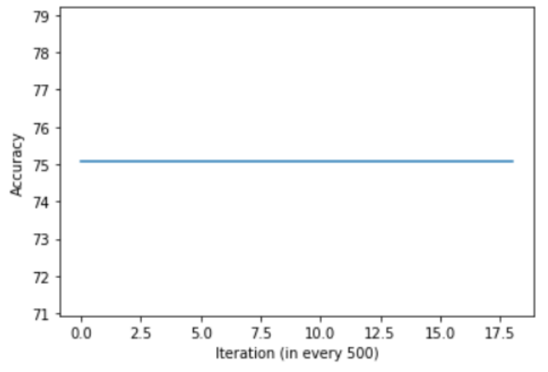
- **Setup 4 Neural Network with 24 nodes and 7 hidden layers with ReLU Activation**

Here the number of hidden layers is 7. Everything else remains unchanged.

Hyper Parameters	Values
Batch Size	100
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.001
Number of Nodes	24
Number of hidden layers	7

Visualization:





Here we can see the best accuracy we got is upto 75.07

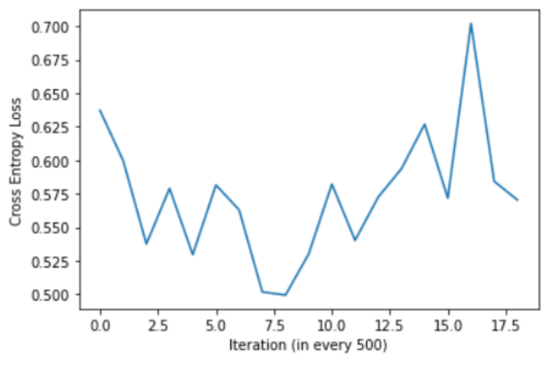
- **Setup 5 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

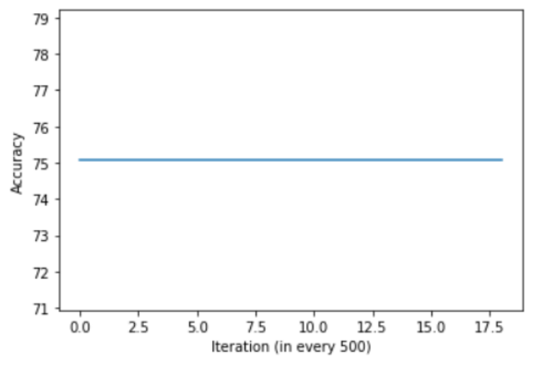
Here the number of hidden layers is 9. Everything else remains unchanged.

Hyper Parameters	Values

Batch Size	100
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.001
Number of Nodes	24
Number of hidden layers	9

Visualization:





Here we can see the best accuracy we got is upto 75.07

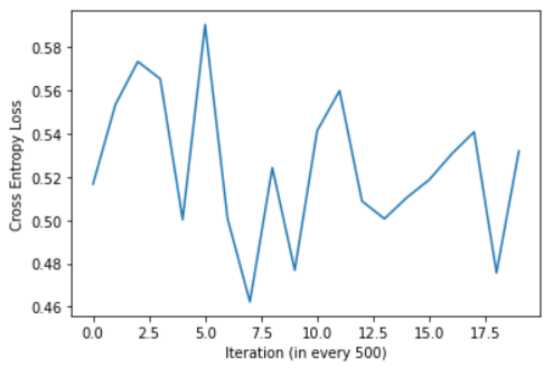
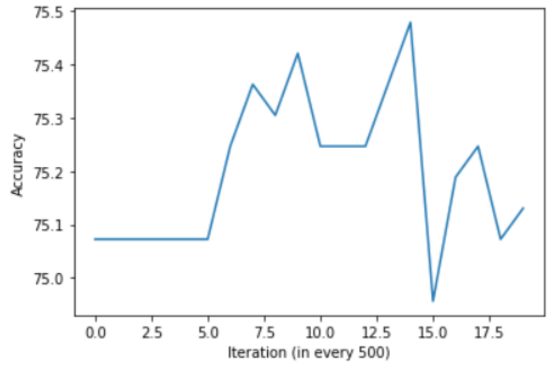
- **Setup 6 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here we increase the bath size by 100 more which means the current batch size is 200.
Also changed the learning rate from 0.001 to 0.03 and the number of hidden layers is 9.

Hyper Parameters	**Values**
Batch Size	200
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.03

Number of Nodes	24
Number of hidden layers	9

Visualization:

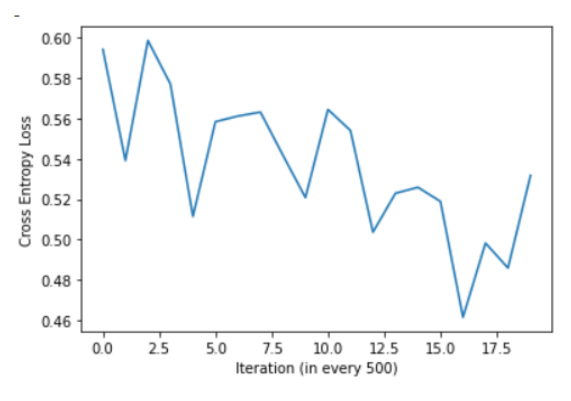
Here we can see the best accuracy we got is upto 75.47937245787332

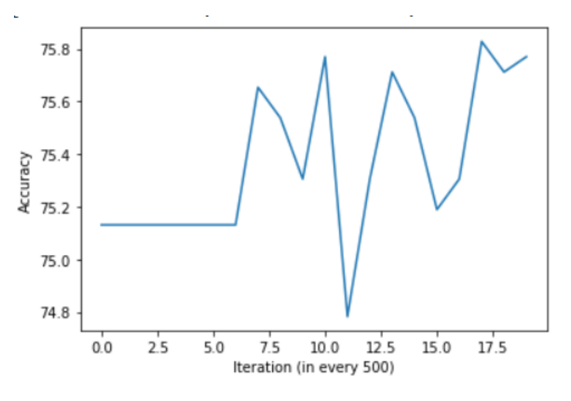
- **Setup 7 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the batch size is 300. Everything else is the same as setup 6.

Hyper Parameters	Values
Batch Size	300
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.03
Number of Nodes	24
Number of hidden layers	9

Visualization:





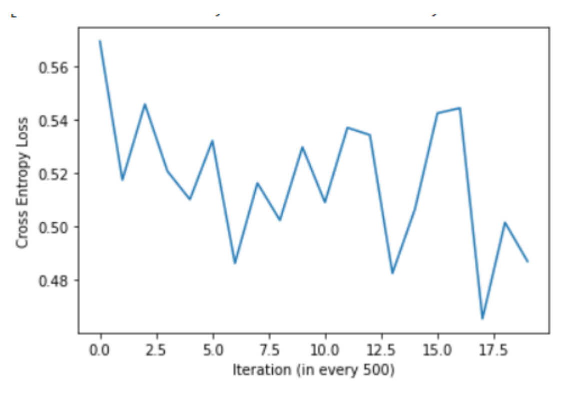
Here we can see the best accuracy we got is upto 75.7699012202208

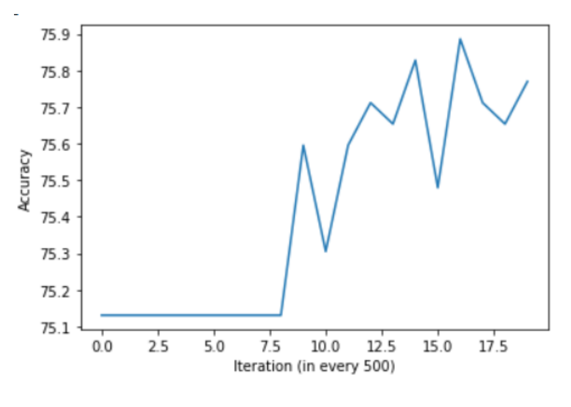
- **Setup 8 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the batch size is 400. Everything else is the same as setup 7.

Hyper Parameters	**Values**
Batch Size	400
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.03
Number of Nodes	24
Number of hidden layers	9

Visualization:





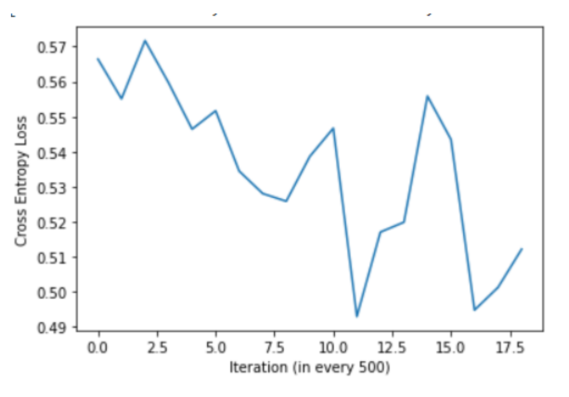
Here we can see the best accuracy we got is upto 75.82800697269029

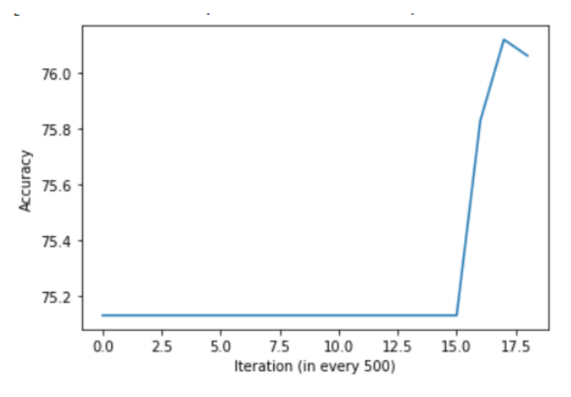
Setup 9 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation

Here the batch size is 500. Everything else is the same as setup 8.

Hyper Parameters	Values
Batch Size	500
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.03
Number of Nodes	24
Number of hidden layers	9

Visualization:





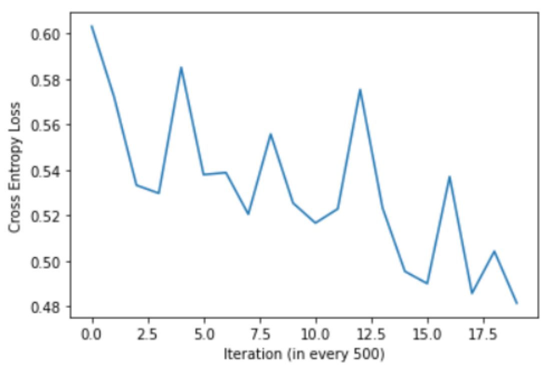
Here we can see the best accuracy we got is upto 76.11853573503777

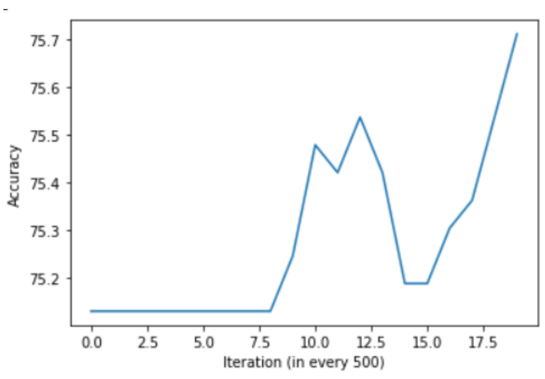
- **Setup 10 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the batch size is 600. Everything else is the same as setup 9.

Hyper Parameters	**Values**
Batch Size	600
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.03
Number of Nodes	24
Number of hidden layers	9

Visualization:





Here we can see the best accuracy we got is upto 75.7117954677513

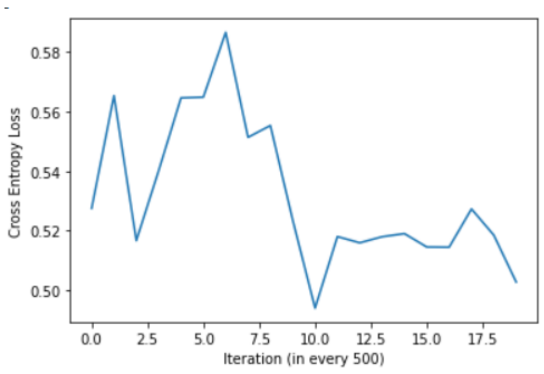
- **Setup 11 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

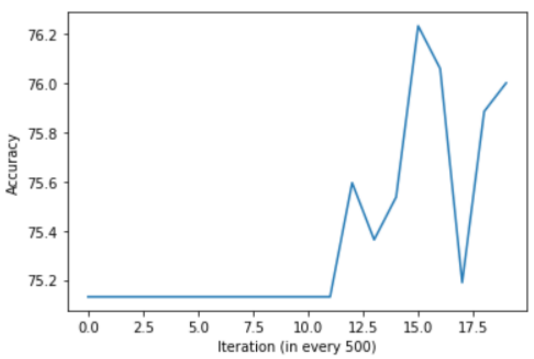
Here the batch size is 700. Everything else is the same as setup 10.

Hyper Parameters	**Values**
Batch Size	700
Number of Iteration	10000

Number of features	12
Output dim	2
Learning rate	0.03
Number of Nodes	24
Number of hidden layers	9

Visualization:





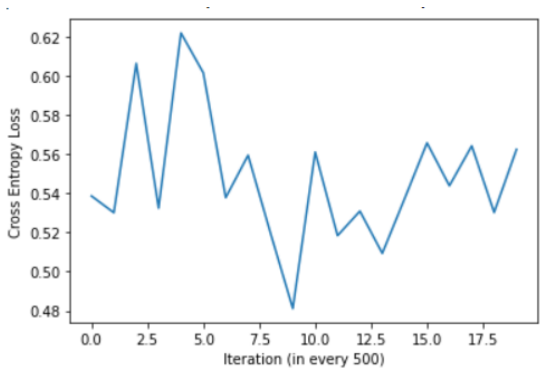
Here we can see the best accuracy we got is upto 76.23474723997676

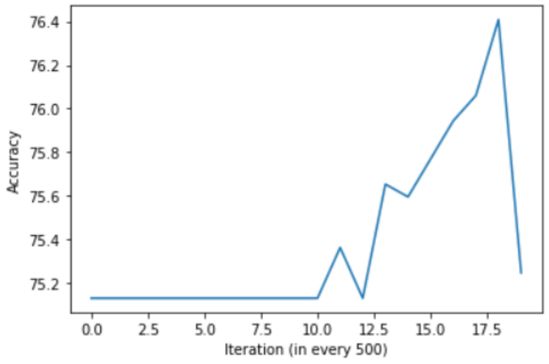
- **Setup 12 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the batch size is 800. Everything else is the same as setup 11.

Hyper Parameters	**Values**
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.03
Number of Nodes	24
Number of hidden layers	9

Visualization:





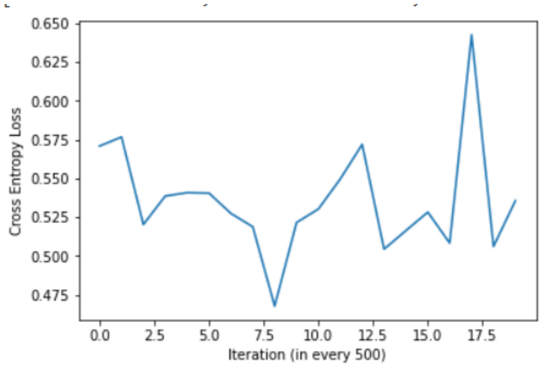
Here we can see the best accuracy we got is upto 76.40906449738524

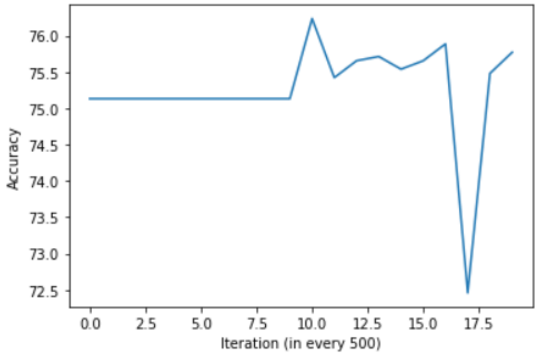
- **Setup 13 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the batch size is 900. Everything else is the same as setup 12.

Hyper Parameters	Values
Batch Size	900
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.03
Number of Nodes	24
Number of hidden layers	9

Visualization:





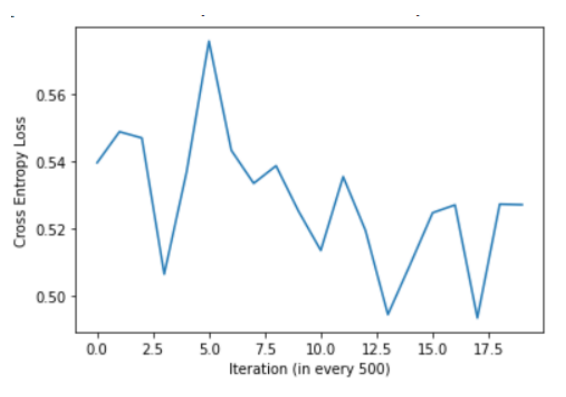
Here we can see the best accuracy we got is upto 76.23474723997676

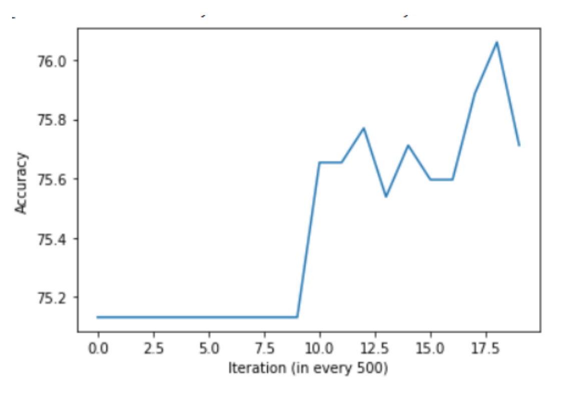
- **Setup 14 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the batch size is 1000. Everything else is the same as setup 13.

Hyper Parameters	Values
Batch Size	1000
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.03
Number of Nodes	24
Number of hidden layers	9

Visualization:





Here we can see the best accuracy we got is upto 76.06042998256828

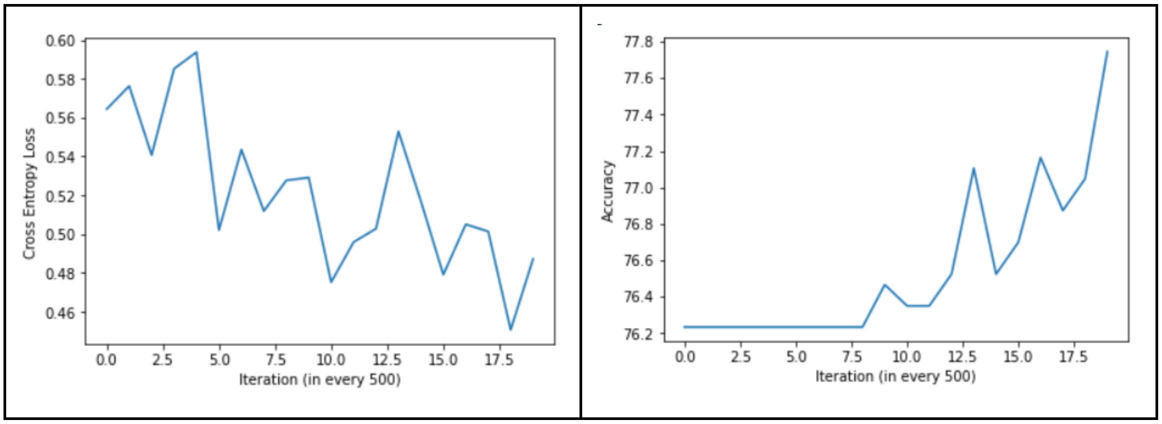
- **Setup 15 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here we use the batch size of 800 as it has given the best accuracy out of other scenarios. We are changing the learning rate to see if we can manage to find better results.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000

Number of features	12
Output dim	2
Learning rate	0.04
Number of Nodes	24
Number of hidden layers	9

Visualization:



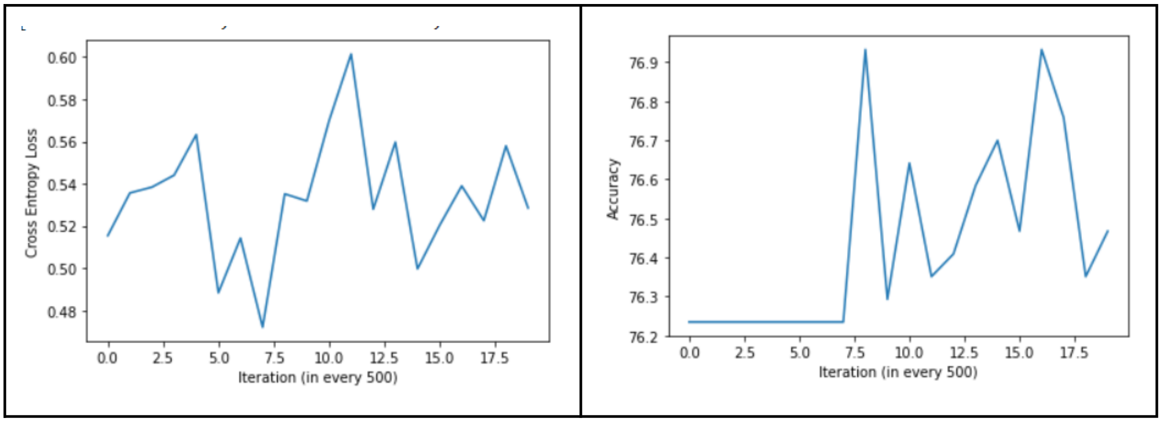
Here we can see the best accuracy we got is upto 76.06042998256828

- **Setup 16 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.05. Everything else is the same as setup 15.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.05
Number of Nodes	24
Number of hidden layers	9

Visualization:



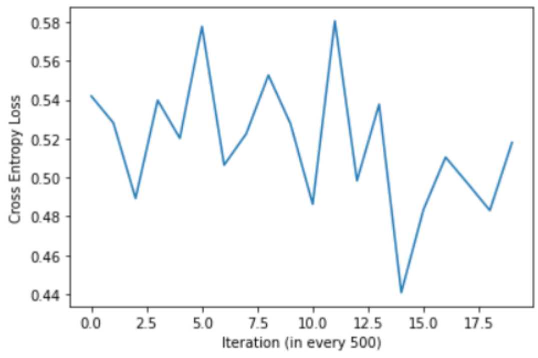
Here we can see the best accuracy we got is upto 76.93201626961068

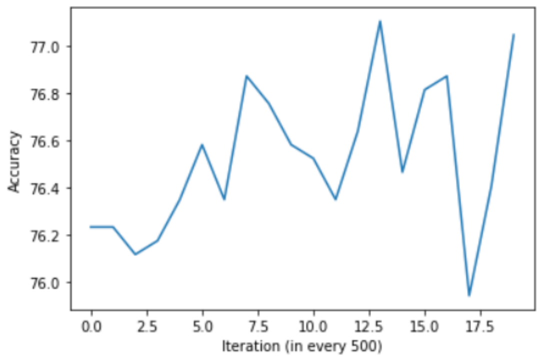
- **Setup 17 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.06. Everything else is the same as setup 16.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.06
Number of Nodes	24
Number of hidden layers	9

Visualization:





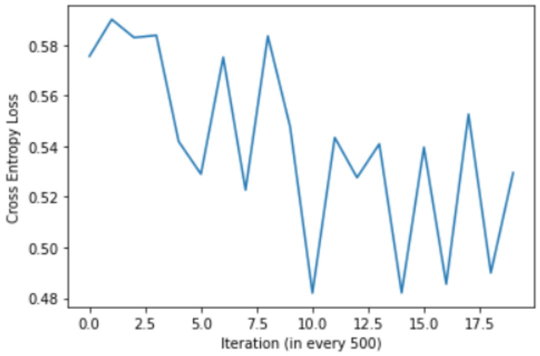
Here we can see the best accuracy we got is upto 77.10633352701917

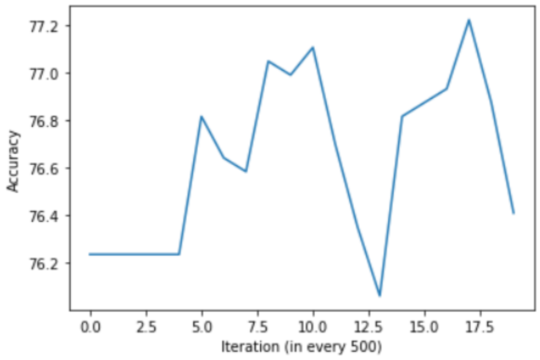
- **Setup 18 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.07. Everything else is the same as setup 17.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.07
Number of Nodes	24
Number of hidden layers	9

Visualization:





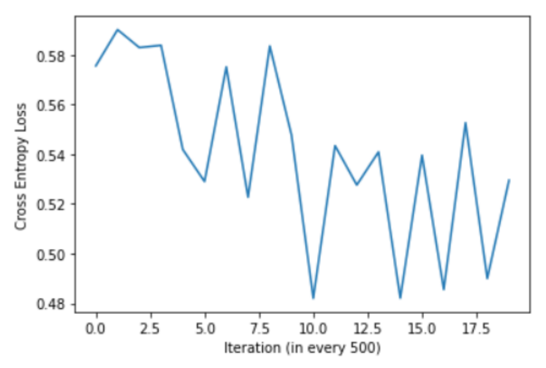
Here we can see the best accuracy we got is upto 77.22254503195816

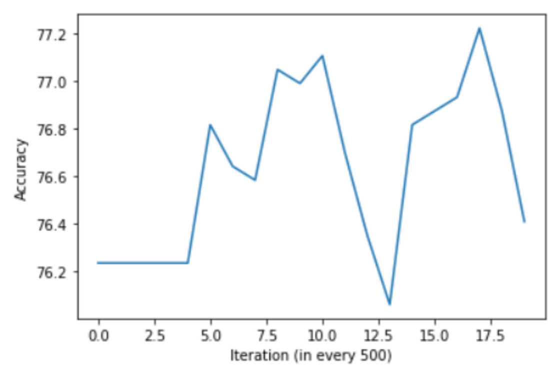
- **Setup 19 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.08. Everything else is the same as setup 18.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.08
Number of Nodes	24
Number of hidden layers	9

Visualization:





Here we can see the best accuracy we got is upto 77.22254503195816

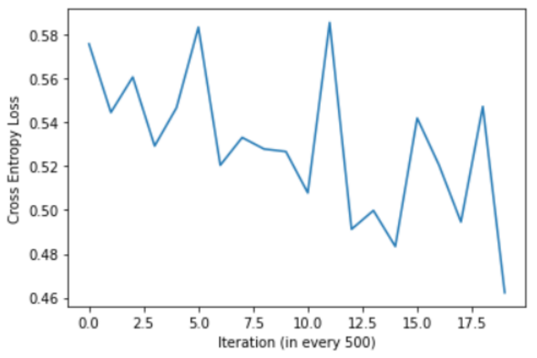
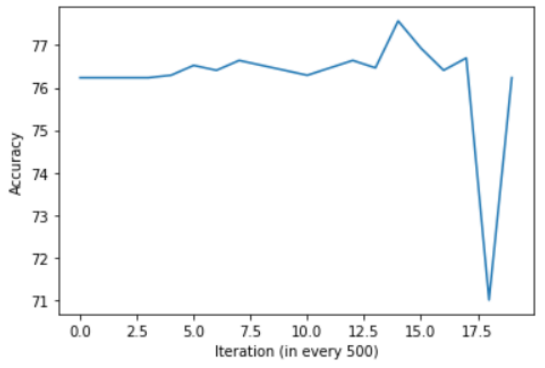
- **Setup 20 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.09. Everything else is the same as setup 15.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000

Number of features	12
Output dim	2
Learning rate	0.09
Number of Nodes	24
Number of hidden layers	9

Visualization:

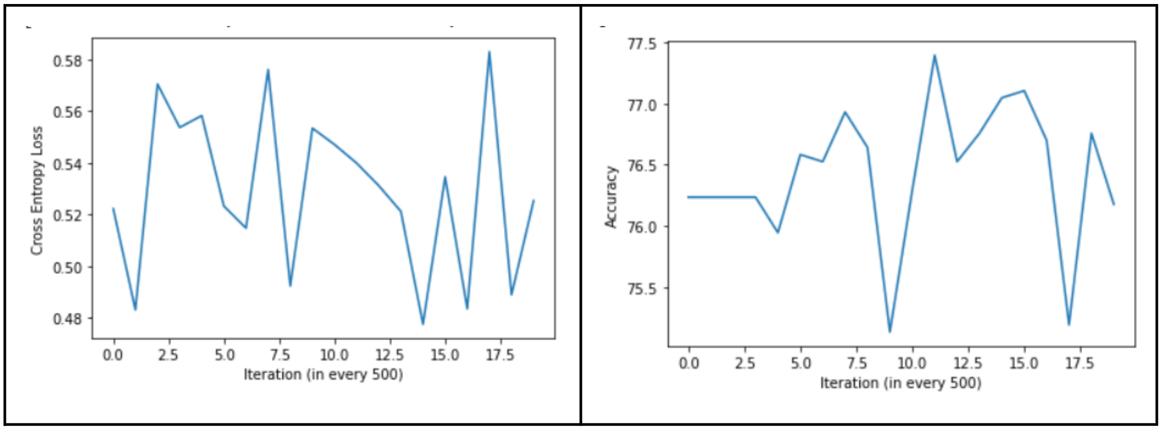
Here we can see the best accuracy we got is upto 77.57117954677513

- **Setup 21 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.11. Everything else is the same as setup 15.

Hyper Parameters	**Values**
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.11
Number of Nodes	24
Number of hidden layers	9

Visualization:



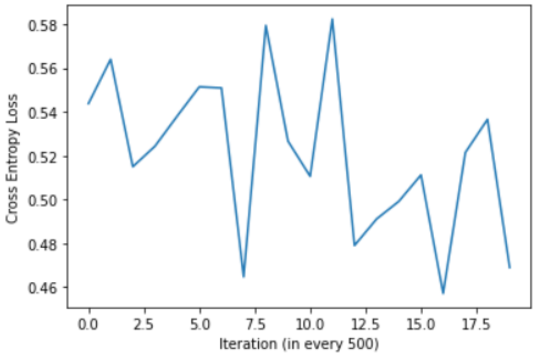
Here we can see the best accuracy we got is upto 77.39686228936665

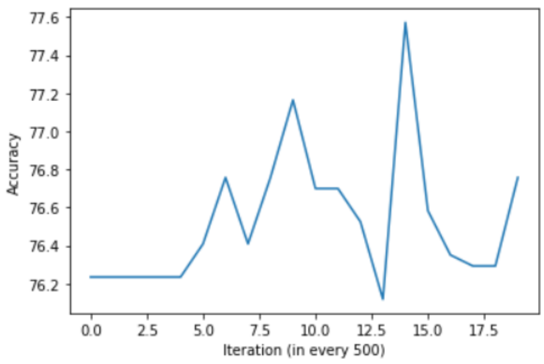
- **Setup 22 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.12. Everything else is the same as setup 15.

Hyper Parameters	**Values**
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.12
Number of Nodes	24
Number of hidden layers	9

Visualization:





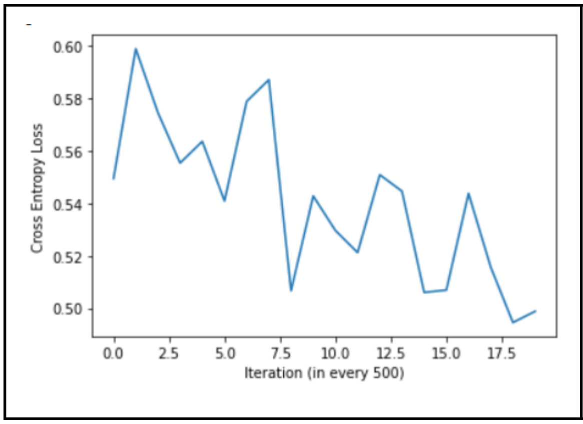
Here we can see the best accuracy we got is upto 77.57117954677513

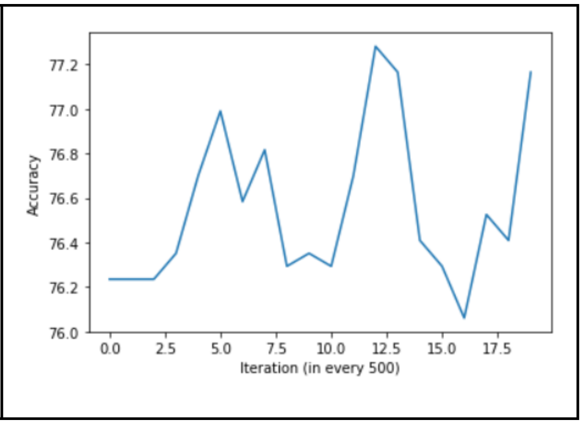
Setup 23 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation

Here the Learning rate is 0.15. Everything else is the same as setup 15.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.15
Number of Nodes	24
Number of hidden layers	9

Visualization:





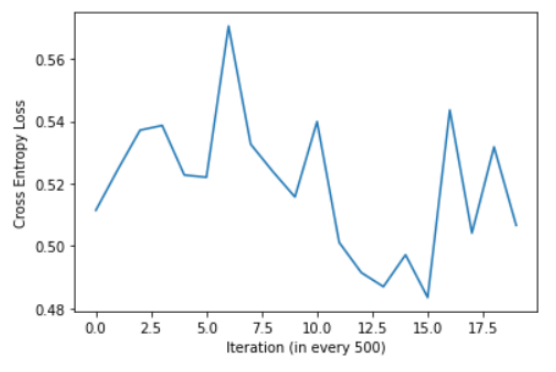
Here we can see the best accuracy we got is upto 77.28065078442766

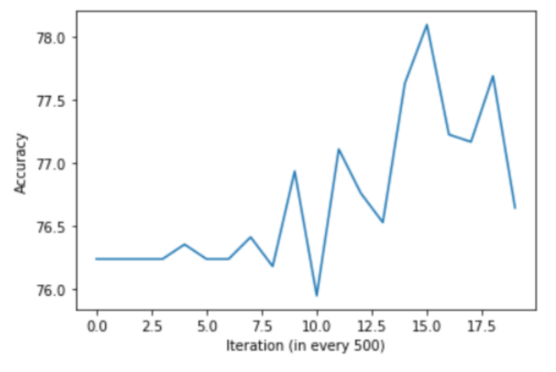
- ## Setup 24 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation

Here the Learning rate is 0.17. Everything else is the same as setup 15.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	24
Number of hidden layers	9

Visualization:





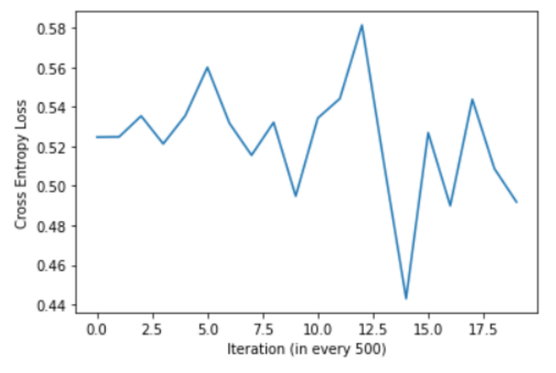
Here we can see the best accuracy we got is upto 78.09413131900058

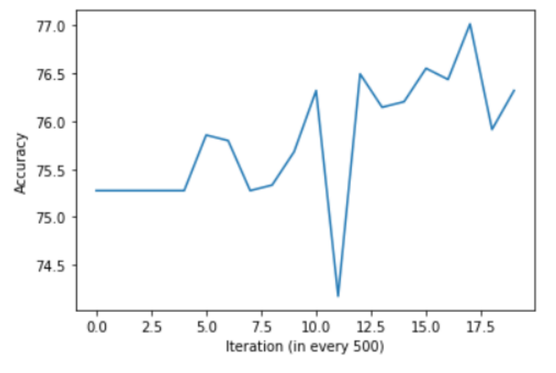
- **Setup 25 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.19. Everything else is the same as setup 15.

Hyper Parameters	**Values**
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.19
Number of Nodes	24
Number of hidden layers	9

Visualization:





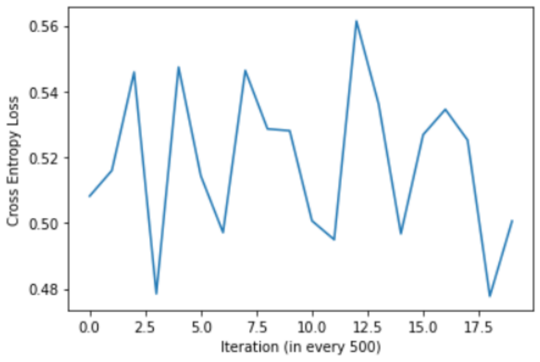
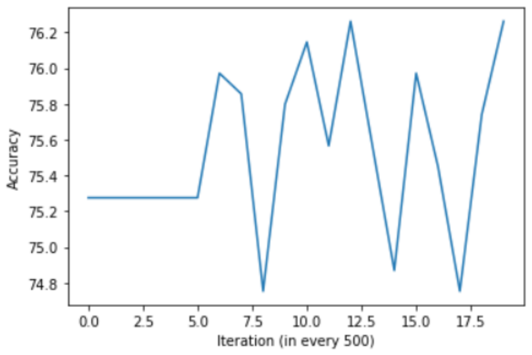
Here we can see the best accuracy we got is upto 77.01683110853163

- **Setup 26 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.21. Everything else is the same as setup 15.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.21
Number of Nodes	24
Number of hidden layers	9

Visualization:

Here we can see the best accuracy we got is upto 76.26233313987231

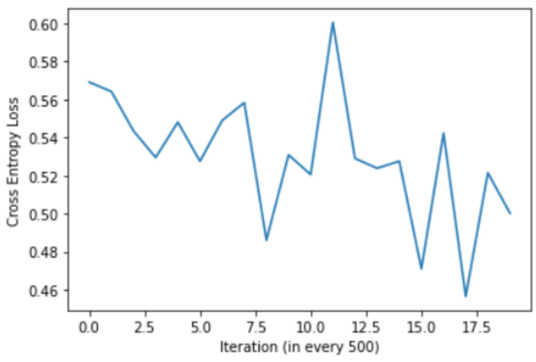
- **Setup 27 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

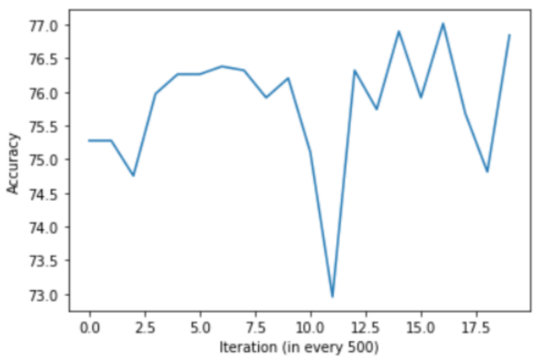
Here the Learning rate is 0.23. Everything else is the same as setup 15.

Hyper Parameters	Values
Batch Size	800

Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.23
Number of Nodes	24
Number of hidden layers	9

Visualization:





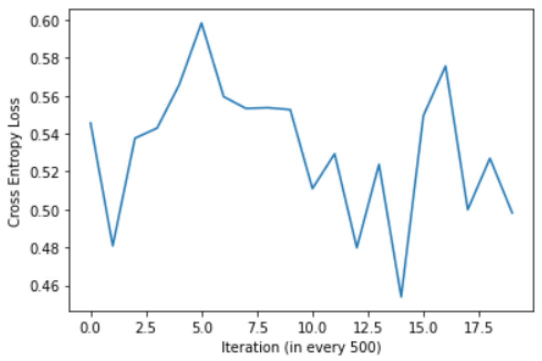
Here we can see the best accuracy we got is upto 76.84271619268718

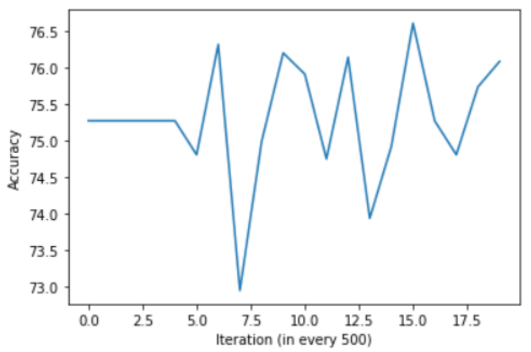
- **Setup 28 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Learning rate is 0.25. Everything else is the same as setup 15.

Hyper Parameters	**Values**
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.25
Number of Nodes	24
Number of hidden layers	9

Visualization:





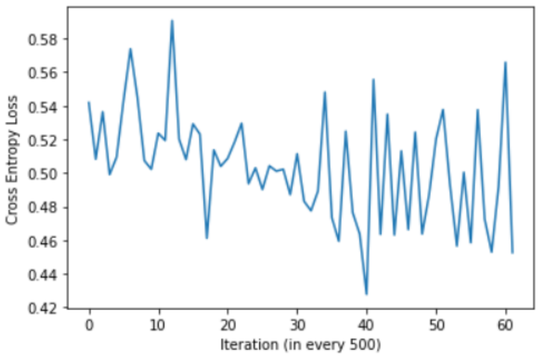
Here we can see the best accuracy we got is upto 76.61056297156124

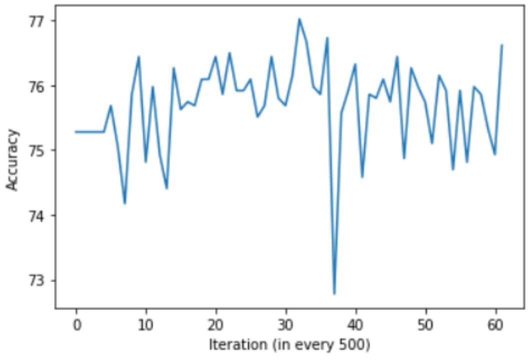
- ## Setup 29 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation

Here the Batch size is 850 and the Number of Iterations is 30000. Everything else is the same as setup 15.

Hyper Parameters	**Values**
Batch Size	850
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	24
Number of hidden layers	9

Visualization:





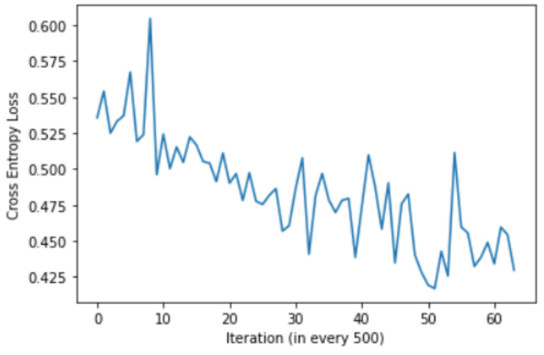
Here we can see the best accuracy we got is upto 76.61

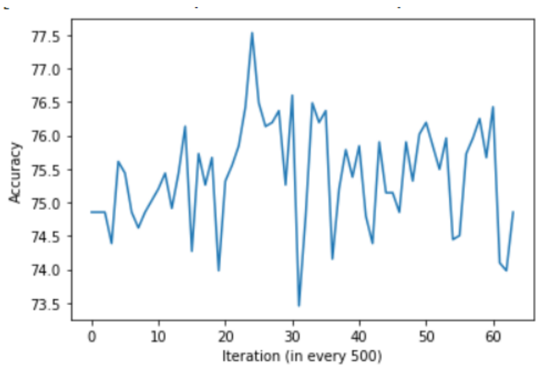
- **Setup 30 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Batch size is 1500. Everything else is the same as setup 29

Hyper Parameters	**Values**
Batch Size	1500
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	24
Number of hidden layers	9

Visualization:





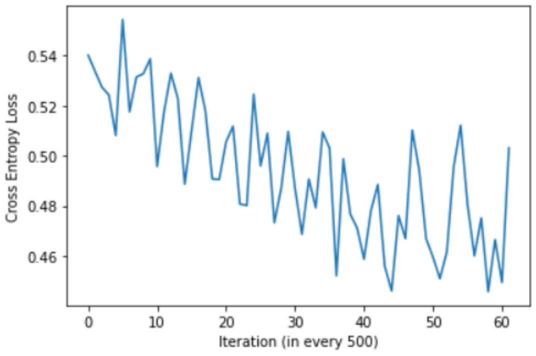
Here we can see the best accuracy we got is upto 77.53

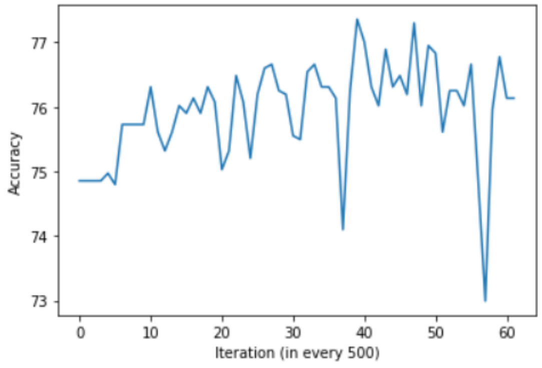
- **Setup 31 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation**

Here the Batch size is 2000. Everything else is the same as setup 29

Hyper Parameters	Values
Batch Size	2000
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	24
Number of hidden layers	9

Visualization:





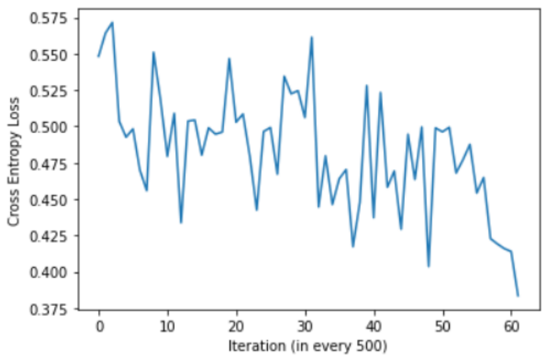
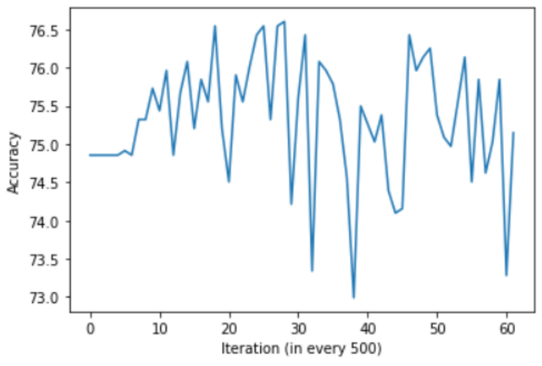
Here we can see the best accuracy we got is upto 77.36

- **Setup 32 Neural Network with 34 nodes and 9 hidden layers with ReLU Activation**

Here Batch Size is 800 and Number of Nodes are 34. Everything else is the same as setup 31.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	34
Number of hidden layers	9

Visualization:

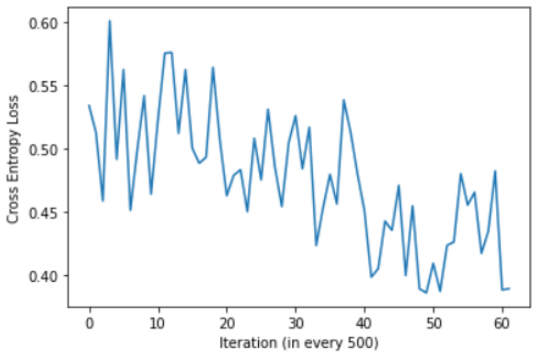
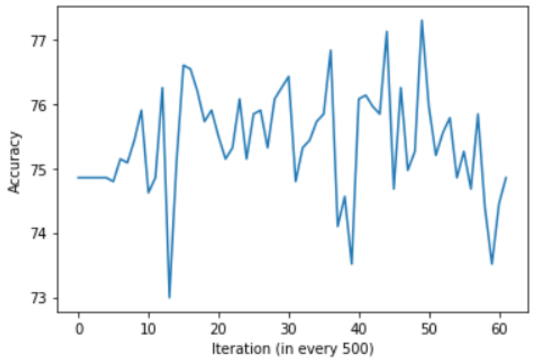
Here we can see the best accuracy we got is upto 76.54

- **Setup 33 Neural Network with 44 nodes and 9 hidden layers with ReLU Activation**

Here Number of Nodes are 44. Everything else is the same as setup 32.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	44
Number of hidden layers	9

Visualization:

Here we can see the best accuracy we got is upto 77.3

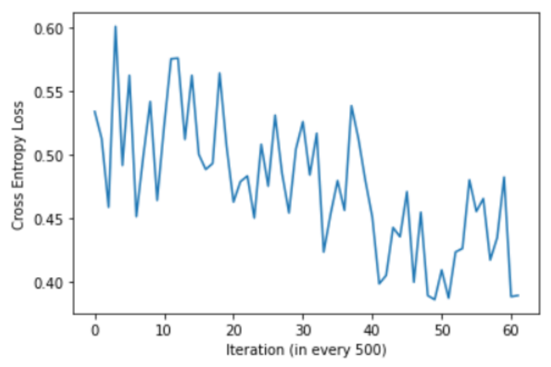
- **Setup 34 Neural Network with 54 nodes and 9 hidden layers with ReLU Activation**

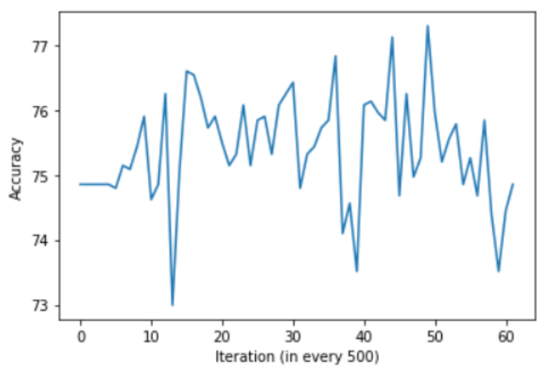
Here Number of Nodes are 54. Everything else is the same as setup 32.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2

Learning rate	0.17
Number of Nodes	54
Number of hidden layers	9

Visualization:





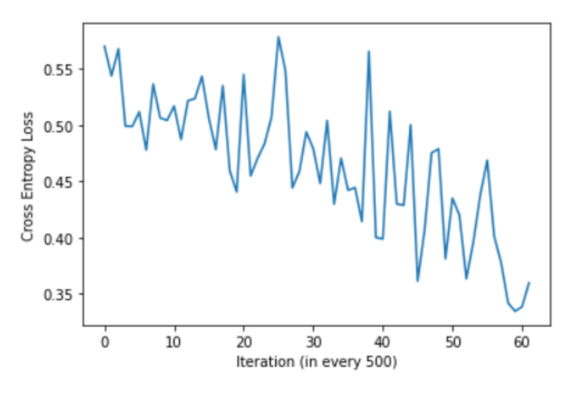
Here we can see the best accuracy we got is upto 76.89

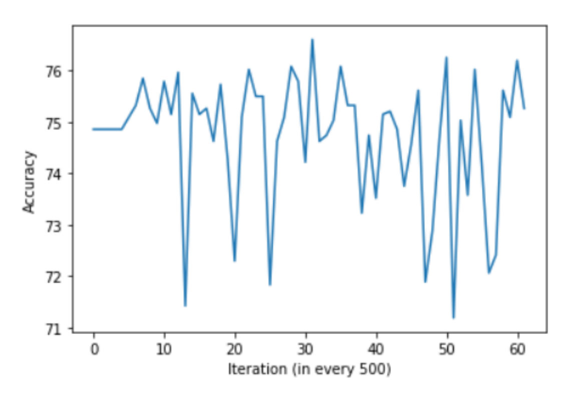
- **Setup 35 Neural Network with 64 nodes and 9 hidden layers with ReLU Activation**

Here Number of Nodes are 64. Everything else is the same as setup 32.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	64
Number of hidden layers	9

Visualization:





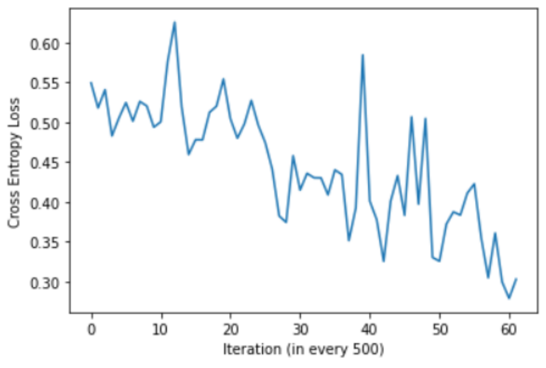
Here we can see the best accuracy we got is upto 76.25

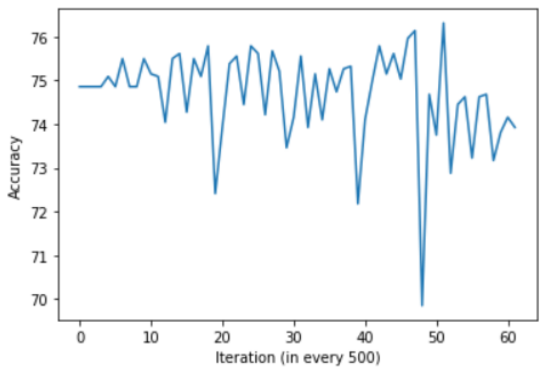
- **Setup 36 Neural Network with 74 nodes and 9 hidden layers with ReLU Activation**

Here Number of Nodes are 74. Everything else is the same as setup 32.

Hyper Parameters	**Values**
Batch Size	800
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	74
Number of hidden layers	9

Visualization:





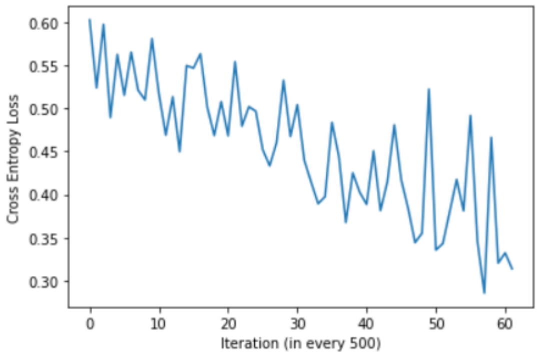
Here we can see the best accuracy we got is upto 75.66

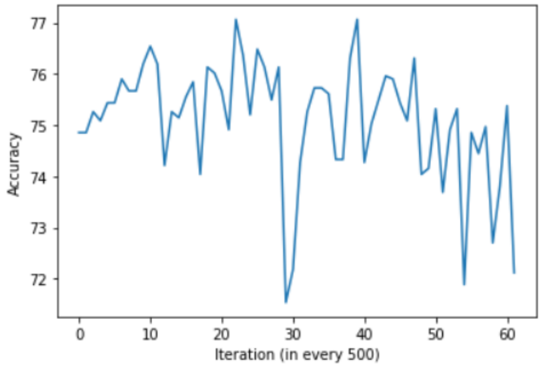
- ## Setup 37 Neural Network with 94 nodes and 9 hidden layers with ReLU Activation

Here Number of Nodes are 94. Everything else is the same as setup 32.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	94
Number of hidden layers	9

Visualization:





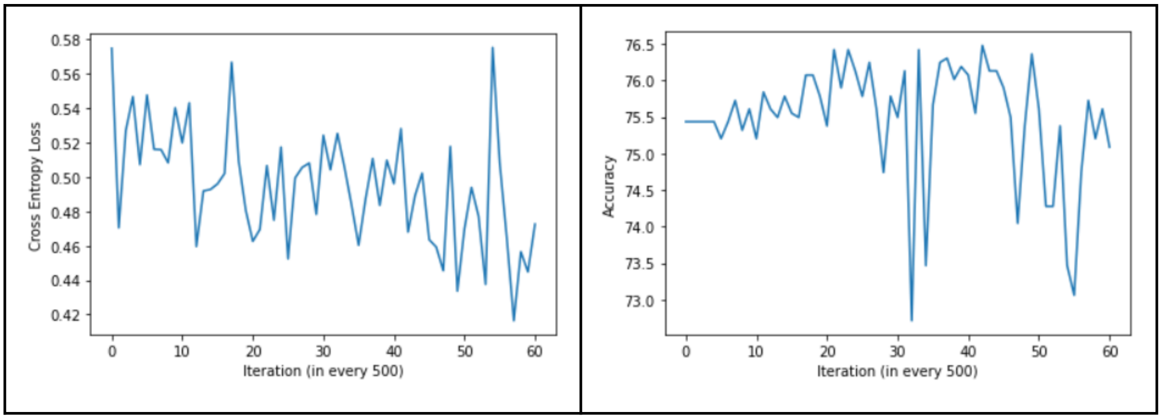
Here we can see the best accuracy we got is upto 77.07

- ## Setup 38 Neural Network with 26 nodes and 9 hidden layers with ReLU Activation

Here Number of Nodes are 26. Everything else is the same as setup 32.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	26
Number of hidden layers	9

Visualization:



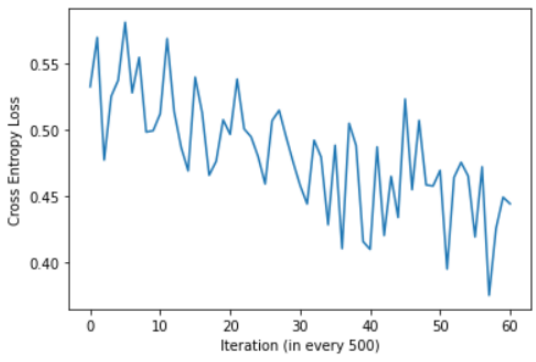
Here we can see the best accuracy we got is upto 76.47

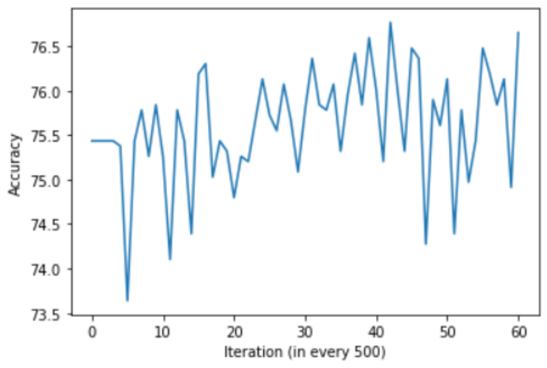
- **Setup 39 Neural Network with 29 nodes and 9 hidden layers with ReLU Activation**

Here Number of Nodes are 29. Everything else is the same as setup 32.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	29
Number of hidden layers	9

Visualization:





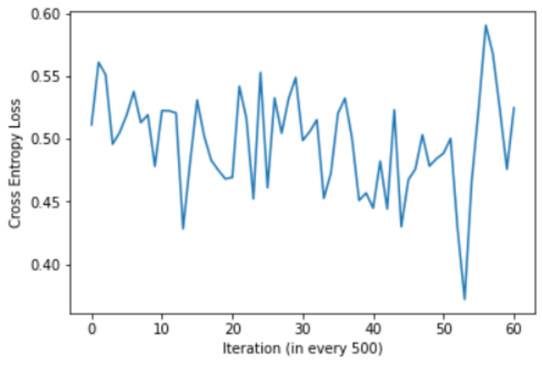
Here we can see the best accuracy we got is upto 76.47

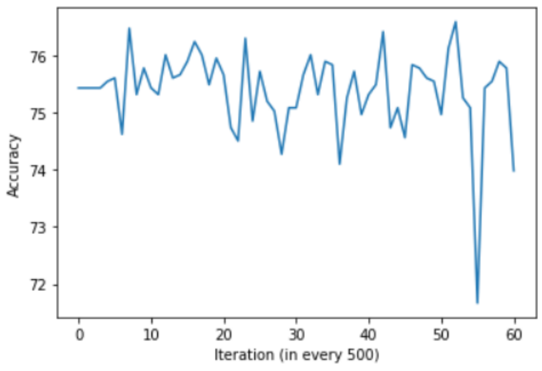
- ## Setup 40 Neural Network with 32 nodes and 9 hidden layers with ReLU Activation

Here Number of Nodes are 32. Everything else is the same as setup 32.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	32
Number of hidden layers	9

Visualization:





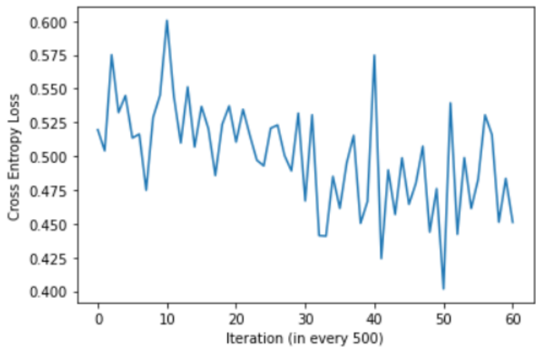
Here we can see the best accuracy we got is upto 76.41

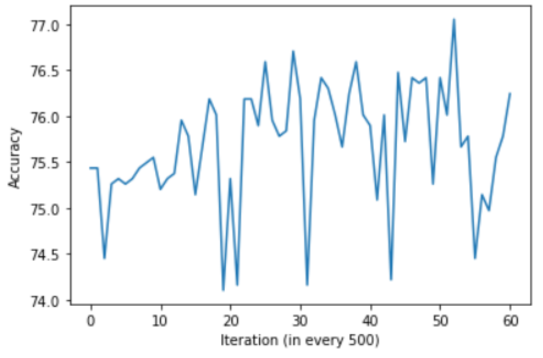
- **Setup 41 Neural Network with 19nodes and 9 hidden layers with ReLU Activation**

Here Number of Nodes are 19. Everything else is the same as setup 32.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	19
Number of hidden layers	9

Visualization:





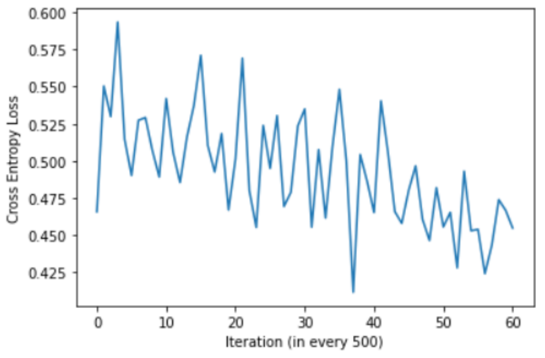
Here we can see the best accuracy we got is upto 76.59

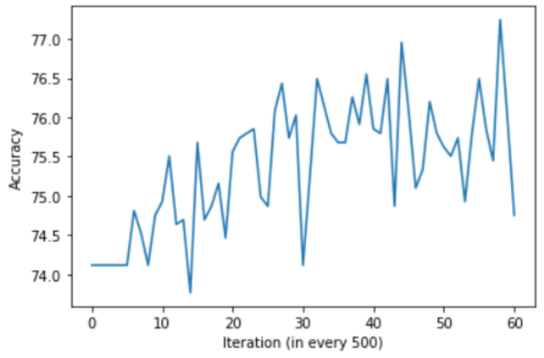
- **Setup 42 Neural Network with 19nodes and 9 hidden layers with ReLU Activation**

Here Number of Nodes are 22. Everything else is the same as setup 32.

Hyper Parameters	Values
Batch Size	800
Number of Iteration	30000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	22
Number of hidden layers	9

Visualization:



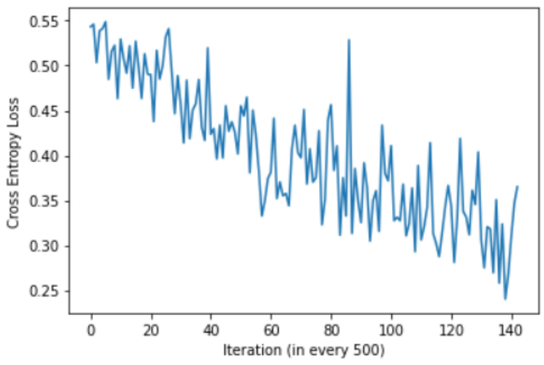


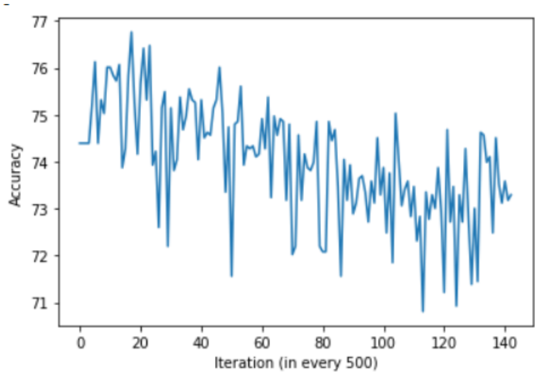
Here we can see the best accuracy we got is upto 77.24

- **Setup 43 Neural Network with 19nodes and 9 hidden layers with LeakyReLU Activation**

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	24
Number of hidden layers	9

Visualization:





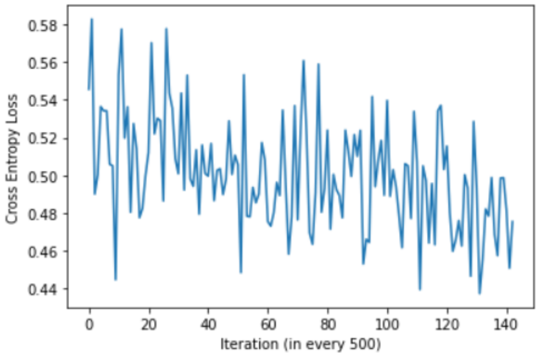
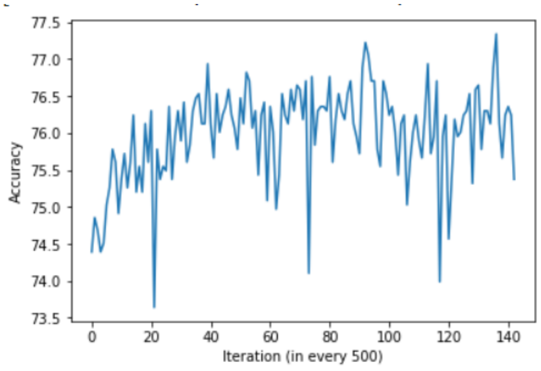
Here we can see the best accuracy we got is upto 76.42

- **Setup 44 Neural Network with 19nodes and 9 hidden layers with Tanh Activation**

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12

Output dim	2
Learning rate	0.17
Number of Nodes	24
Number of hidden layers	9

Visualization:

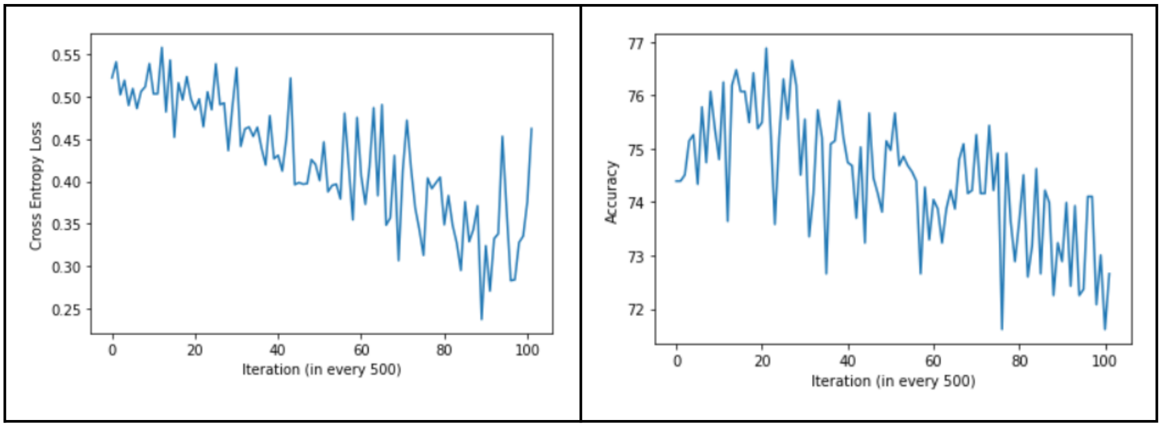
	

Here we can see the best accuracy we got is upto 77.34

- **Setup 45 Neural Network with 19nodes and 9 hidden layers with Tanh and LeakyReLUActivation**

Hyper Parameters	Values
Batch Size	800
Number of Iteration	10000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	24
Number of hidden layers	9

Visualization:



Here we can see the best accuracy we got is upto 76.88

Incremental Progress of the Model :

The table to understand the optimization process are given below-

Setup	Accuracy	Optimization

Setup-1	75.15	Batch Size=100;
		Iteration = 10000;
		Learning Rate = 0.001;
		Nodes = 24;
		Hidden Layers = 2

Setup-4	75.00	Batch Size=100;
		Iteration = 10000;
		Learning Rate = 0.001;
		Nodes = 24;
		Hidden Layers = 7

Setup-5	75.00	Batch Size=100;
		Iteration = 10000;
		Learning Rate = 0.001;
		Nodes = 24;
		Hidden Layers = 9

Setup-8	75.82	Batch Size=400;
		Iteration = 10000;
		Learning Rate = 0.03;
		Nodes = 24;
		Hidden Layers = 9

Setup-9	76.11	Batch Size=100;

		Iteration = 10000;
		Learning Rate = 0.001;
		Nodes = 24;
		Hidden Layers = 2

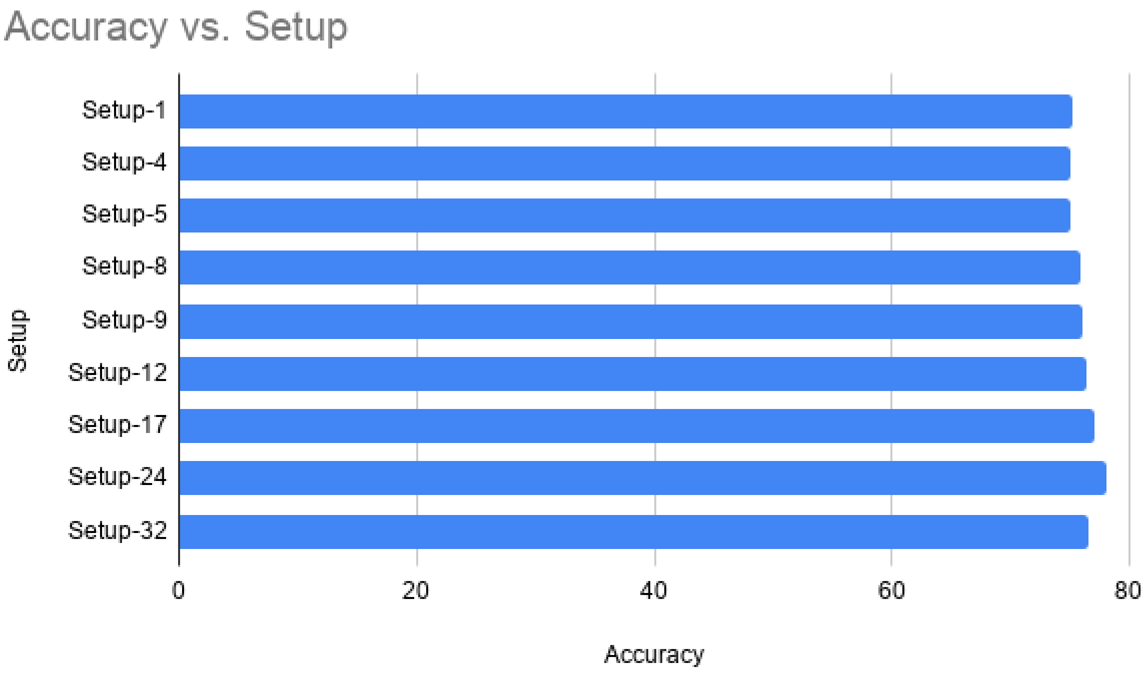
Setup-12	76.40	Batch Size=800;
		Iteration = 10000;
		Learning Rate = 0.03;
		Nodes = 24;
		Hidden Layers = 9

Setup-17	77.10	Batch Size=800;
		Iteration = 10000;
		Learning Rate = 0.06;
		Nodes = 24;
		Hidden Layers = 9

Setup-24**	78.09	Batch Size=800;
		Iteration = 10000;
		Learning Rate = 0.17;
		Nodes = 24;
		Hidden Layers = 9

Setup-32	76.54	Batch Size=800;
		Iteration = 30000;
		Learning Rate = 0.17;
		Nodes = 34;
		Hidden Layers = 9

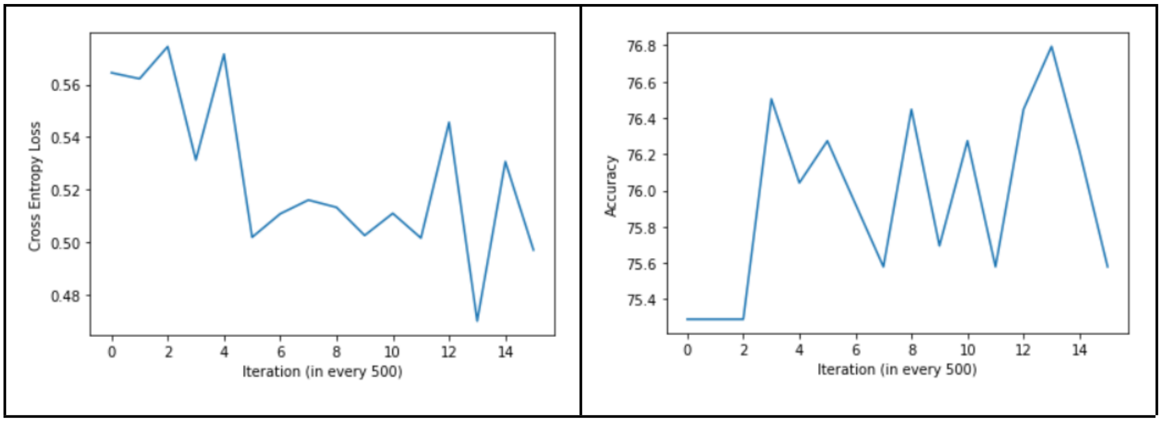
A bar chart is given below-



Final Setup from setup 24 Neural Network with 24 nodes and 9 hidden layers with ReLU Activation

Hyper Parameters	Values
Batch Size	800
Number of Iteration	8000
Number of features	12
Output dim	2
Learning rate	0.17
Number of Nodes	24
Number of hidden layers	9

Visualization:



Here we can see the best accuracy we got is upto 76.79

Result of the model on the Test Set(10 Percent):

Loss: 0.4867

Accuracy: 76.221

Conclusion:

In modern days most organisations desire to have skilled people to make their organisation competent and best. If Deep Learning can be used in the sector of HRM then it can save a lot of time and effort and also do the task more precisely.For our project it was important for the company to know which candidates are serious to work for the company or they are just looking for a new employment. It helps the company to reduce cost and time as well as to plan the courses and categorization of candidates. We experimented with different hyperparameter settings. We only tried simple neural network but experimenting with CNN and RNN may have given better results.

References:

1. (Predict who will move to a new job | Kaggle)
2.(Who will leave a job? | Kaggle)
3. (Awesome HR Data Visualization & Prediction | Kaggle)