Data Mining
COSC 2111/2110
Assignment 2 Neural Networks

Data Mining - Neural Networks are just one of the tools used in data mining. ANNs are used to find patterns in the data and to infer rules from them. Neural networks are useful in providing information on associations, classifications, clusters, and forecasting. A neural network can be trained to produce outputs that are expected, given a particular input. If we have a network that fits well in modelling a known sequence of values, one can use it to predict future results. An obvious example is the Stock Market Prediction.

Part 1: Classification with Neural Networks:

Classification is one of the most active research and application area of neural networks. It is an essential feature to separate datasets into classes for the purpose of Rule generation, Decision Making, Pattern recognition, Dimensionality Reduction, Data Mining etc.

Answer 1:

Data Pre-processing - Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Major Tasks in Data Pre-processing are Data cleaning, fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies Therefore With the help of Weka software it is more convenient to perform all these necessary tasks for our dataset called hypothyroid. arff.

Pre-process has done by importing the dataset and I took an overview of dataset values. I got TBG to attribute in overall has Nan values which make data noisy, to deal with the same I chose the option to remove this attribute. Furthermore, in the class label, we have 4 negative class labels, compensated hypothyroid, primary hypothyroid, and secondary hypothyroid. I discovered that the secondary _ hypothyroid includes only two examples which illustrate no observation after training the model. Henceforth, the precision has been indeterminable for this label which leads to reassign this class into primary hypothyroid manually by using the most suited option from weka.
Moving forward, to deal with missing value again I used the most convenient option from weka "Replacemissingvalues" which automatically replace the missing values with some other value. This is also called imputing missing values.

Data transformation is again a major data mining technique that involves a transformation in the dataset and as per dataset, I preferred to move forward by choosing option "NominalToBinary" in weka which Converts all nominal attributes into binary numeric attributes.
After performing all the necessary tasks related to data pre-processing, I found that the dataset contains 51 input including all attributes also binary values and having 3 output by reassigning secondary hypothyroid into the primary.
Last but not the least, Data mining involves the Normalization task too. Normalization is the process of efficiently organizing data. There are two main objectives of the normalization process: eliminate redundant data and ensure data dependencies make sense. however, to deal with this task Weka has an approachable option "Normalize "through filter function.

Answer 2:

Separating dataset into training, testing, and Validation sets is an important part of evaluating data mining models Therefore, by using similar data for training, testing, and Validation, it can minimize the effects of data discrepancies and better understanding the characteristics of the model.

Training set: In weka software, it is very convenient to separate the training set among the whole dataset by pre-processing thus, it is attainable through resampling instances from filter function. A random subsample of a dataset using either sampling with the replacement or without replacement. It's recommendable to set "noReplacement" true to ignore any duplicate instances. Meanwhile, behind the scene, Weka will generate 60% as per my preference for the whole dataset and provided me 2263 instances as a training set.

Test set: To begin with all instances, invert selection must be True as we need to get the 40% of the dataset which makes a sum of 1509 instance therefore again with the same procedure but this invert selection must be false as it's a first operation of making half of the 40% dataset and rest keep that for validation set eventually I got 754 instances as a Test set which is used for performance evaluation.

Validation set: It is different from the test set. The validation set actually can be regarded as a part of the training set as we kept half of the 40% dataset because it is used to build the model of a neural network. It is usually used for parameter selection and to avoid overfitting. Whereas I got 755 instances as my Validation set.

In the nutshell, after dividing data into training and validation datasets we just need to train the neural network on the training dataset only, Therefore the network calculates the errors on the training data set and the validation data set. Stop training when the validation error is the minimum. This means the network can generalize to unseen data, If we stop training when the training error is minimum then will have overfitted and the network cannot generalize to unseen data. In the Cross-Validation process, the validation error is usually bigger than the training error in most cases. There are more complicated methods to do cross-validation. Cross-validation does not apply just to the neural network but is a way of selecting the best model that produces the best generalization to unseen data.

Answer 3:

Weka has a collection of machine learning algorithms for Neural Network and the best to use is Multilayer Perceptron to determine test data instance's class label Therefore I have used both training and testing datasets to find accuracy in between of both. The training set has given the accuracy of 96.1114%, on the other hand, I used the test data set followed by training which provides me an accuracy of 93.5099 %.
The parameter was chosen as the default setting which considers the average hidden layer with 500 number of epochs. Thus, the default parameter settings for both training hypothyroid_train. arff and Testing hypothyroid_test. arff dataset. In brief, the Train dataset has more accuracy than the test dataset henceforth, test data is data unseen by the model, and train data is the data were the model used to train itself.

Answer 4:

Training Set:

Run	**Parameters**	**Hidden Layer**	**No. of Epochs**	**Correctly Classifiec Accuracy**	**Incorrectly Classified Accuracy**
1	Lr = 0.3	0	500	95.1834 %	4.8166 %
2	Lr = 0.3	1	500	95.7578 %	4.2422 %
3	Lr = 0.3	3	500	94.5647 %	5.4353 %
4	Lr = 0.3	5	500	92.1343 %	7.8657 %
5	Lr = 0.3	7	500	92.1343 %	7.8657 %

- First, too many neurons in the hidden layers may result in overfitting. As experimenting with different hidden layers along with no hidden layer, I may find that model without a hidden layer has more accuracy than the other, however, Hidden layers allow for the function of a neural network to be broken down into specific transformations of the data. Each hidden layer function is specialized to produce a defined output, as I tried to increase the number of hidden layers with a minimum of 5 nodes has decreases its accuracy. Thus, to get a more accurate result parameter and the number of epochs has been set to default, while the hidden layer was on 5 number of trials. Furthermore, no hidden layer or single layer is much better and more accurate than others.

Test Set:

Run	Parameters	Hidden Layer	No. of Epochs	Correctly Classified Accuracy	Incorrectly Classified Accuracy
1	Lr = 0.3	0	500	94.1722 %	5.8278 %
2	Lr = 0.3	1	500	94.3046 %	5.6954 %
3	Lr = 0.3	3	500	92.3179 %	7.6821 %
4	Lr = 0.3	5	500	92.1854 %	7.8146 %
5	Lr = 0.3	7	500	92.1854 %	7.8146 %

- Followed by the training set, the default parameter and the same number of epochs have been performed on the test set as well, and again we got similar results, moreover, the model without hidden layer has got more accuracy and a single hidden layer has high accuracy too. Overall, more detectable points are that after the 4th number of trials we got the same result on the 5th trial as well. In summary, a model without a hidden layer or single layer has more accuracy than others, thus, it is more suitable for this set as well.

Answer 5:

Training and Test Set:

Run	Architecture	Parameters	Train MSE	Train Error (%)	Epochs	Test MSE	Test Error	Over fitting
1	51-5-3	Lr=0.2	0.38	4.8166 %	100	0.399	6.0927	0.019
2	51-5-3	Lr=0.2	0.376	4.684 %	200	0.403	6.0927	0.027
3	51-5-3	Lr=0.2	0.382	4.8608 %	400	0.411	6.2252	0.029
4	51-5-3	Lr=0.2	0.353	3.9328 %	600	0.398	5.8278	0.045
5	51-5-3	Lr=0.2	0.351	3.5793 %	800	0.384	4.9007	0.033

- During training, the network might learn too much, and this problem is referred to as overfitting. By looking at the above table, On the result, five random Epochs are selected and 100 Epochs have the lowest overfitting with a constant learning rate of 0.2 and 5 hidden nodes, therefore these were the best because it has least over fitting. As per the observation, overfitting is gradually increasing at some point when extreme 800 epochs performed it again get decreased, however, if we try without a hidden layer as per my experiment it has very less overfitting of 0.013.

Answer 6:

Training and Test Set:

Run	Architecture	Parameters	Train MSE	Train Error (%)	Epochs	Test MSE	Test Error %	Over fitting
1	51-3-3	Lr=0.2	0.381	4.7724	200	0.415	7.2848	0.034
2	51-6-3	Lr=0.2	0.376	4.7282	200	0.405	6.3576	0.029
3	51-9-3	Lr=0.2	0.375	4.8608	200	0.407	6.2252	0.032
4	51-12-3	Lr=0.2	0.366	4.0654	200	0.405	6.4901	0.039
5	51-15-3	Lr=0.2	0.363	4.1096	200	0.402	5.6954	0.039

- As per the observation when few hidden nodes are used, it may result in underfitting (neural network's output function is too simple and ignores important points in the data). However, if too many hidden nodes are used, it may result in overfitting. As shown on the result, the optimum number of hidden nodes is 6 because it has the least overfitting by setting constant parameters and the number of epochs.

Answer 7:

Training and Test Set:

Run	Architecture	Parameters	Train MSE	Train Error (%)	Epochs	Test MSE	Test Error %	Over fitting
1	51-5-3	Lr=0.1 Momentum = 0.5	0.352	3.5793	500	0.386	5.4305	0.034
2	51-5-3	Lr=0.2 Momentum = 0.5	0.385	5.2585	500	0.411	6.2252	0.026
3	51-5-3	Lr=0.2 Momentum = 0.7	0.376	4.5515	500	0.401	5.5629	0.025
4	51-5-3	Lr=0.2 Momentum = 0.3	0.390	5.7888	500	0.429	8.2119	0.039
5	51-5-3	Lr=0.1 Momentum = 0.8	0.372	4.4189	500	0.410	6.8874	0.038

- Backpropagation adjusts the weights to reach the minima of the error function. However, it may cause the network to be trapped in local minima. This can be solved by adding momentum to the training rule. Momentum can speed up calculations significantly. It forces the system to continue moving in the same direction on the error surface without trapping at local minima. Since we selected an average learning rate of 0.2, we need a high momentum of 0.7 to speed up the calculations. As shown in the result, 0.2 learning rate and 0.7 momentum are the most optimum since it has the lowest overfitting and it's the best thus, we can observe that when we change the parameters of learning rate and momentum the overfitting will gradually increase up to the certain point and again it starts from the beginning and starts increasing.

Answer 8:

J48 Classifier:

Run	Parameters	Train Error	Cross-Validation	Overfitting
1	C 0.25 M 2	0.3093 %	0.5303 %	0.221

➢ **Classification accuracy of Weka J48 algorithm: -**

```
Correctly Classified Instances          2256               99.6907 %
Incorrectly Classified Instances           7                0.3093 %
Kappa statistic                            0.9791
Mean absolute error                        0.0029
Root mean squared error                    0.0384
Relative absolute error                    3.9606 %
Root relative squared error               19.9727 %
Total Number of Instances               2263
```

- Overall, as a I observed J48 algorithm one the best classifier in weka for machine learning algorithm, however according to classification accuracy I found 99.6907% which depicts a best accuracy.

J48 Classifier Pros:

- High Accuracy to build a Model as compared to another classifier.
- J48 Classifier, like filters are organized in hierarchy.
- The algorithm is simple to understand, interpret and visualize as the idea is mostly used in our daily lives. Output of a J48 can be easily interpreted.
- J48 can be used for both classification and regression problems.
- J48 can handle both continuous and categorical variables.
- In Weka software has ability for no feature scaling (standardization and normalization) required in case of J48 as it uses rule-based approach instead of distance calculation.
- J48 can automatically handle missing values.
- J48 is usually robust to outliers and can handle them automatically.
- Training period is less as compared to another Classifier because it generates only one tree.

J48 Classifier Cons:

- This is the main problem of the J48 Classifier, it generally leads to overfitting of the data which ultimately leads to wrong predictions. In order to fit the data (even noisy data), it keeps generating new nodes and ultimately the tree becomes too complex to interpret. In this way, it loses its generalization capabilities.
- It performs very well on the trained data but starts making a lot of mistakes on the unseen data.
- Due to the overfitting, there are very high chances of high variance in the output which leads to many errors in the final estimation.
- Adding a new data point can lead to re-generation of the overall tree and all nodes need to be recalculated and recreated.
- Little bit of noise can make it unstable which leads to wrong predictions.
- If data size is large, then one single tree may grow complex and lead to overfitting. So, in this case, we should use another Classifier instead of a J48.

Multilayer Perceptron Classifier:

Run	Parameters	Train Error	Cross-Validation	Overfitting
1	L 0.3 M 0.2 N 500 V 0 S 0 E 20 H a	3.8886 %	6.2307 %	2.3421

- As per the above results of two different classifier, I can state clearly that J48 worked well for model as it has least overfitting as compare to Multilayer Perceptron.
- In above I tried to run with default setting of parameters for both J48 and Multilayer Perceptron. According to the results I got J48 is the best classifier to use.

- **Classification accuracy of Weka Multilayer Perceptron algorithm: -**

```
Correctly Classified Instances         2175                96.1114 %
Incorrectly Classified Instances        88                   3.8886 %
Kappa statistic                        0.6619
Mean absolute error                     0.0223
Root mean squared error                 0.1332
Relative absolute error                  29.9555 %
Root relative squared error              69.29    %
Total Number of Instances               2263
```

- In summary, I analysed that as compare to J48 I found less accuracy in multilayer perceptron along with that it took much time to build a model then J48 algorithm, henceforth, I might get high accuracy while playing with different parameter like learning rate, momentum, seeds and epochs but with the default parameter this results might leads to higher over fitting rate as compare to J48 Classifier.

Multi-Layer Perceptron Pros:

- Multi-layer is most of the neural networks expects deep learning. it uses one or two hidden layers. The main advantage is they can be used for difficult to complex problems. However, they need long training time sometimes.
- Multi-layer perceptron is easy to deal with missing values.
- It I also convenient in converting binary to numeric (0,1 or 1,0)
- It is more accessible in dividing dataset into three parts (Train, Test and Validation).

Multi-Layer Perceptron Cons:

- No direct connections between input and output layers
- The first is the well-known XOR problem were network is not able to separate non-linear distribution of data so that the hidden layer (middle layer) of the multilayer network just learns the transform data to make it linearly separable.
- Multi-layer perceptron in weka software does not support visualization to see error graph whether model is over-fitted or under-fitted.

Answer 9:

Javanns Pros: -

- Very nice and neat graphical user interface.
- In Javanns, we need to prepare data and encoding.
- Able to simulate multiple neural network architectures.
- Supports 25+ neural network architectures.
- Javanns has more flexibility to create own neural networks
- Parameter setting is very convenient. For example, learning rate, momentum etc.
- We can easily tune the parameter manually.
- Convenient to monitored behind the scenes.

Javanns Cons: -

- Multiple errors and poor error handling (program crashed multiple times).
- Operation is not much faster than other open software.
- Javanns reads only numerical data.
- Not all functionality is explained.
- Adds “temporary” files to the user’s machine without deleting or announcing it.
- Javanns is not much effective as well for machine learning algorithms.

Weka multilayer perceptron Pros: -

- As weka is fully implemented in java programming languages, it is platform independent & portable.
- weka can solve missing values and prepare data for neural network.
- Weka software contain very graphical user interface, so the system is very easy to access.
- There is very large collection of different data mining algorithms.
- Weka software is very quick to build own neural network.
- Weka software is a user-friendly tool for data mining filed.
- Weka s/w provides fast and efficient result.
- Easy to change various parameter setting.
- To build a neural network, weka does not need any kind of prior coding skills.
- It is very easy to create own neural network without any coding knowledge.

- **Weka multilayer perceptron Cons: -**

- Its accessible for only arff data file.
- Weka does not have facility to demonstrate how the graphs and learning rate works behind the scene.
- Weka GUI much not affective then other software.
- Weka s/w does not have any option to save graphical results.
- Not much popular as compare to other software for data mining.
- weka has most significant disadvantage that it can only handle small data set, whenever a set is bigger than a few megabytes an out of memory error occurs.
- Due to large data set weka get crash.

In brief , I find myself more comfortable to create model that Weka supports several standard data mining tasks, more specifically, data pre-processing, clustering, classification, regression, visualization, and selection. Weka is also a free and open source software same as Javanns. Since , it is fully implemented in the Java programming language and thus runs on almost any modern computing platform. A comprehensive collection of data pre-processing and modelling techniques Ease of use due to its graphical user interfaces.

Part 2: Numeric Prediction with Neural Networks:

Neural networks work better at predictive analytics because of the hidden layers. Linear regression models use only input and output nodes to make predictions. Neural network also uses the hidden layer to make predictions more accurate. Neural networks can learn complex patterns using layers of neurons which mathematically transform the data. The layers between the input and output are referred to as "hidden layers". A neural network can learn relationships between the features that other algorithms cannot easily discover.

Answer 1:

Data Pre-processing -To begin with; the clear understanding of data pre-processing in WEKA means removing the unwanted or rather the irrelevant details from data that is collected from the field which might lead to wrong analysis. For example, those unprocessed facts might have null fields or even columns that are irrelevant to the dataset called heart-v1. arff in hand.
Data cleaning - This is the removing of unnecessary raw facts from the given data. filling in missing values; sometimes might realise that of the important facts were omitted during the process of collecting the actual data which is worth to take time and replace the values, therefore in my observation data consist of missing values which I replaced or remove by using "Replacemissingvalues" function in weka.
Smooth noisy data: - this is simply the act of concentrating on the attention causing data. Weka has very convenient way to deal with this significant problem by choosing option called "Replacemissingvalues" other than that the required data scaling varies from minimum to maximum. Usually we use a given range. I observe that has 27 attributes including binary values.
Data transformation-This is the changing of data into required ways of presenting it before Constructing any model all nominal attributes are transformed into binary variables that are then treated as numeric in the form of 1,0 or 0,1 format, thus I processed the same to deal with it by using "nominalTobinary"function which has transformed all nominal values into numeric.
Data Normalization - Normalization takes the data one step further by reducing the records to just common event attributes. To normalize the data which has lot different value however the main function in dataset is that to put all numeric values in common scale, without distorting differences in the ranges of values with the help of "normalize" function. Moreover, in the pre-processing I have followed all the necessary steps to get 27 input (including binary values) and 1 output which is Chol attribute.

Answer 2:

It seems quite intuitive to split data into a training portion, test portion and validation portion, so the model can be trained on the first and then tested with the testing data followed by validation data. It may be a good idea to split the data in a way, so that the model can be trained on a larger portion in order to adapt to more possible data constellations.

Training set: In weka software, it is very convenient to separate training set among whole dataset by pre-processing thus, it is attainable through resample instances from filter function. Random subsample of a dataset using either sampling with replacement or without replacement. It is recommendable to set “noReplacement” true to ignore any duplicate instances. Meanwhile, behind the scene Weka will generate 60% as per my preference of whole dataset and provided me 363 instances among 606 instances as a training set.

Test set : To begin with all instances ,invert selection must be True as we need to get the 40% of the dataset which makes sum of 606 instance therefore again with the same procedure but this invert selection must be false as it’s a first operation of making half of the 40% dataset and rest keep that for validation set eventually I got 121 instance as a Test set which is used for performance evaluation.

Validation set: It is different from test set. Validation set can be regarded as a part of training set as we kept half of the 40% dataset because it is used to build a neural networks model. It is usually used for parameter selection and to avoid overfitting. Whereas I got 122 instances as my Validation set.

In the nutshell, after dividing data into training and validation dataset we must need to train the neural network on training dataset only Therefore the network calculates the errors on the training data set and the validation dataset.

MAE (mean-absolute error)- In weka software as it is based on Java inbuilt programming it has very convenient and faster way to calculate Mean Absolute error as its automatically calculates behind the scene however without software, I preferred to calculate mean-absolute error (MAE) by calculating the sum of absolute errors (SAE). The actual procedure is straightforward, however.

1. Absolute Values -Subtracted by true value (signified by xt) from the measured value (signified by xi), possibly generating a negative result depending on the data points followed by absolute value of the result to generate a positive number.
2. Repeat this process for each set of measurements and forecasts in data. The number of sets is signified by n in the formula, with the Σn i=1 indicating that the process starts at the first set
   (i = 1) and repeats a total of n times.
3. Add the absolute values together to generate SAE. Now we can calculate the mean absolute error. Once we calculate the SAE, we need to find the mean or average value of the absolute errors. We can use this formula MAE = SAE ÷ n to get result. We may find other formula too but there is no functional difference in them.

It is always convenient to scale the inputs to make all of them have a proper range. In the context of neural networks, the scaling function can be thought of as a layer connected to the neural network's inputs. The scaling layer contains some basic statistics on the inputs. They include the mean, standard deviation, minimum and maximum values.

Answer 3:

Training and Test Set:

Run	Hidden Layer	Parameters	Train MAE	Epochs	Test MAE	Over fitting
1	0	Lr=0.3	0.1083	100	0.1122	0.0039
2	1	Lr=0.3	0.0593	200	0.0627	0.0034
3	3	Lr=0.3	0.0372	500	0.0457	0.0085
4	5	Lr=0.3	0.112	700	0.1115	NO
5	7	Lr=0.3	0.112	1000	0.1115	NO

- As per the observation, to analyse this task I have chosen default parameter with random number of epochs which has been performed on different hidden layer. During training, the network might learn too much, and this problem is referred as overfitting. As per the above result 5 and 7 hidden layer with 700 and 1000 epochs has no overfitting, moreover It was the best when compared with others. However, the hidden layers perform nonlinear transformations of the inputs entered the network as we can see single number of hidden layers also has least overfitting by comparing with no hidden layer.

- In the nutshell, the two classes must be linearly separable for the perceptron network to function correctly. Indeed, this is the main limitation of a single-layer perceptron network. Therefore, since XOR outputs cannot be separated using a straight line, they cannot be classified using a single layer perceptron.

- Perceptron networks should be trained with adapt, which presents the input vectors to the network one at a time and makes corrections to the network based on the results of each presentation. Use of adapt in this way guarantees that any linearly separable problem is solved in a finite number of training, Therefore the output values of a perceptron can take on only one of two values (0 or 1) due to the hard-limit transfer function, however, to point out that networks with more than one perceptron can be used to solve more difficult problems.

Answer 4:

Training and Test Set:

Run	Hidden Layer	Parameters	Train MAE	Epochs	Test MAE	Over fitting
1	27-5-1	Lr=0.2	0.0463	300	0.052	0.0057
2	27-5-1	Lr=0.2	0.0425	500	0.0507	0.0082
3	27-5-1	Lr=0.2	0.0399	700	0.0498	0.0099
4	27-5-1	Lr=0.2	0.0364	1000	0.0492	0.0128
5	27-5-1	Lr=0.2	0.0345	1500	0.0497	0.0152

- During training, the network might learn too much, and this problem is referred as overfitting. Its recommendable to stop training when the error rate of validation data is minimum. Consequently, if we increase the number of epochs, we will have an over-fitted model.it means that your model does not learn the data. As per the above table it clearly stats the same that model has been performed with 5 constant hidden nodes and parameter of 0.2 learning rate with each trial. As a result, random Epochs are selected and 300 Epochs has the lowest overfitting which is most suitable among others, However we may get least or No overfitting with no hidden layer but layers allows for the function of a neural network to be broken down into specific transformations of the data that is reason it gets bit high overfitting as compare to no hidden layer.

Answer 5:

Training and Test Set:

Run	Hidden Layer	Parameters	Train MAE	Epochs	Test MAE	Over fitting
1	27-4-1	Lr=0.2	0.0417	100	0.0424	0.0007
2	27-8-1	Lr=0.2	0.0467	100	0.048	0.0013
3	27-12-1	Lr=0.2	0.0423	100	0.0433	0.001
4	27-15-1	Lr=0.2	0.0423	100	0.0423	NO
5	27-17-1	Lr=0.2	0.0427	100	0.0435	0.0008

- When too few hidden nodes are used, it may result in underfitting (neural network's output function is too simple and misses important details in the data). However, we tried random number of hidden nodes with constant parameter and number of epochs, therefore if too many hidden nodes are used, it may result in overfitting. As shown on the result, the optimum number of hidden nodes is 15 which has "NO OVERFITTING" it meant to be a best hidden node.

Answer 6:

Training and Test Set:

Run	Architecture	Parameters	Train MAE	Epochs	Test MAE	Over fitting
1	27-5-1	Lr=0.1 Momentum = 0.2	0.0423	100	0.0426	0.0003
2	27-5-1	Lr=0.2 Momentum = 0.6	0.0717	100	0.0715	NO
3	27-5-1	Lr=0.3 Momentum = 0.4	0.0608	100	0.0614	0.0006
4	27-5-1	Lr=0.7 Momentum = 0.3	0.0487	100	0.0499	0.0012
5	27-5-1	Lr=0.4 Momentum = 0.1	0.0522	100	0.0556	0.0034

- As we have constant number of epochs and 5 hidden nodes, back propagation adjusts the weights to reach the minima of the error function. However, it may cause the network to be trapped in local minima. This can be solved by adding momentum to the training rule. Momentum can speed up the calculations significantly. It forces the system to continue moving in the same direction on the error surface without trapping at local minima. Since we selected a low learning rate 0.2, we need a high momentum 0.6 to speed up the calculations. As shown in the result, 0.2 learning rate and 0.6 momentum is the most optimum since it has the No overfitting. Here we can clearly observe that when we change the high parameters of learning rate and low momentum the over fitting will increases.

Answer 7:

Training and Test Set:

Run	Architecture	Parameters	Train MSE	Epochs	Test MSE	Overfitting
1	27-5-1	Lr = 0.2	0.2278	500	0.2529	0.0251
2	27-5-1	Lr = 0.2	0.2130	1000	0.2487	0.0357
3	27-5-1	Lr = 0.2	0.2061	2000	0.2537	0.0476

- In the above table we can observed that 500 Epochs, it has least over fitting because the train MSE is lesser then the test MSE. Henceforth, the over fitting is 0.0357 however we tried multilayer perceptron with 5 hidden nodes with 0.2 learning rate but we might get high overfitting with a greater number of epochs, henceforth It produces quite a good result with a low overfitting.

Answer 8:

Relative-absolute error - It is very similar to the relative squared error in the sense that it is also relative to a simple predictor, which is just the average of the actual values. In this case, though, the error is just the total absolute error instead of the total squared error.

Thus, the relative absolute error takes the total absolute error and normalizes it by dividing by the total absolute error of the simple predictor. It is a way to measure the performance of a predictive model. It is primarily used in machine learning, data mining, and operations management.

Mean-absolute error -Mean -absolute error is a measure of how far 'off' a measurement is from a true value or an indication of the uncertainty in a measurement, therefore it is less sensitive to outliers in comparison to others since it doesn't punish huge errors. It is usually used when the performance is measured on continuous variable data. It gives a linear value, which averages the weighted individual differences equally. The lower the value, better is the model's performance. Concisely, I would prefer relative absolute error as mean absolute error does not penalize outliers very well. If we are required to look out for outliers, mean absolute error is not a good choice, whereas it also tells that whether the error of measured value is big or small, compared to the real value.

Answer 9:

Multi-layer Perceptron:

Run	Architecture	Parameters	Train MAE	Test MAE	Overfitting
1	27-5-1	L 0.3 M 0.2 N 500 V 0 S 0 E 20 H a	0.0362	0.048	0.0118

- **Multi-layer Perceptron Pros:**
- Connectionist: used as a metaphor for biological neural networks
- Computationally efficient. Can easily be parallelized
- Universal computing machines

- **Multi-layer Perceptron cons:**
- Convergence can be slow
- Local minima can affect the training process
- Hard to scale

M5P:

Run	Parameters	Train MAE	Test MAE	Overfitting
1	M 4.0	0.0369	0.0349	NO

- **After comparing both the classifier M5P produces good result with no overfitting, thus it is the best classifier for Numeric prediction as compare to Multilayer perceptron.**

M5P Pros:

- Compared to other algorithms decision trees requires less effort for data preparation during pre-processing.
- A decision tree does not require normalization of data.
- A decision tree does not require scaling of data as well.
- Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent.
- A Decision tree model is very intuitive and easy to explain to technical teams as well as stakeholders.

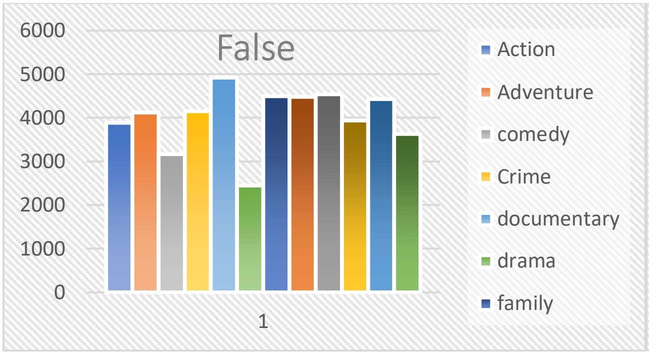
M5P Cons:

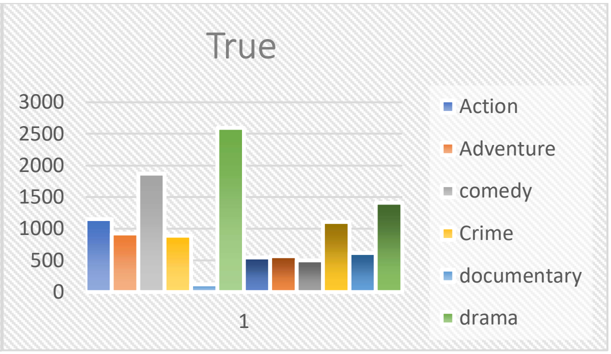
- A small change in the data can cause a large change in the structure of the decision tree causing instability.
- For a Decision tree sometimes, calculation can go far more complex compared to other algorithms.
- Decision tree often involves higher time to train the model.
- Decision tree training is relatively expensive as the complexity and time taken are more.
- Decision Tree algorithm is inadequate for applying regression and predicting continuous values. In short, I tried two different classifiers with their default parameters which is mentioned in above table and according to their performance both neural network and M5P produce almost similar output but M5P has better than multi-layer perceptron since it has No overfitting. thus, M5P performance is better than the other classifiers so this one is the best classifier to use.

Part 3: Data Mining:

Visualization:

- In current era, People used to watch colour visualize movies and in dataset, all the movies came in the colour but in 2015 and 2014 there was few black and white movies. The black and white movies which was taken until 2014 and 2015 were the last after that graphics changed to colour and movies picturized in colour graphics.





- As per the above histogram, it clearly stating that movies contain Action, Adventure, comedy, crime, documentary, drama, and family movies in which most admired movie are drama and comedy whereas documentary and mystery are less made.
- English is a widely used language and as per the dataset most of movies has been made in English which has been picturized in USA followed by UK and other countries. As compare to cast_total_facebook_likes, USA, also has more movie_facebook_likes among the other countries.

  Canada has highest imdb_score in contrary to other countries but overall movies which has been made in USA is more preferred and liked by the people because Hollywood movies (USA) has more Facebook rating and most of the directors belongs from USA.
- USA movies made huge collection of profit as their gross collection is highest then other countries Therefore, movie called "Avatar" made approx. 8 million whereas their budget was only for 2.37 million.

Classification:

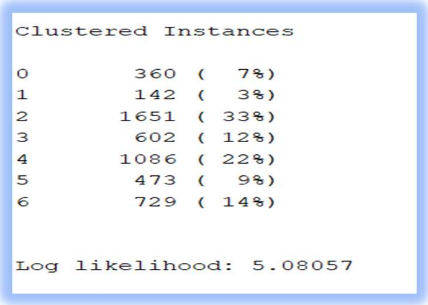
Due to high volume of instances and attributes I tried to reduce attributes with the help of "Select Attribute" to find best among them unfortunately it provided me only 6 best attributes which I found not much helpful because it contains many more relevant attributes.

Model (target – imdb_score)	Accuracy %
J48-C 0.25- M 2	99.9207 %
NaiveBayes	94.9038 %

- Predicting the exact score is quite difficult also any predictions would not be that accurate because the main contribution towards the score contains many other attributes of the movie like critical reviews and a rating or more. There seems very less relation between the stats and the rating since some actors may be good, but the movie's Facebook likes would not be that high because of less publicity of the movie.
- In brief, I would like to conclude that J48 is quite popular and good classifier among others and after using this classifier, the decision tree generated too complex leaf nodes, thus it is quite difficult to find any golden nuggets, however, I got best accuracy of 99.0207% on training set after losing 8 attributes among 37 as compared to other classifiers, moreover most of the classifier does not support like random forest, logistic regression etc. Due to limitation of size of the dataset.

Clustering: EM

A clustering algorithm finds groups of similar instances in the entire dataset. WEKA supports several clustering algorithms such as EM, FilteredClusterer, HierarchicalClusterer, SimpleKMeans and so on. As IMDB dataset has many instances as well as too many attributes, however it is quite difficult to perform any of the cluster algorithm. Hence, several attributes took long run and has limitation to perform numeric class attributes. Furthermore, cluster is used to form the grouped data. Here, I performed EM algorithm cluster to divided into 7 number of clusters with 9 among 37 attributes with respect to have different percentages mention in below table:



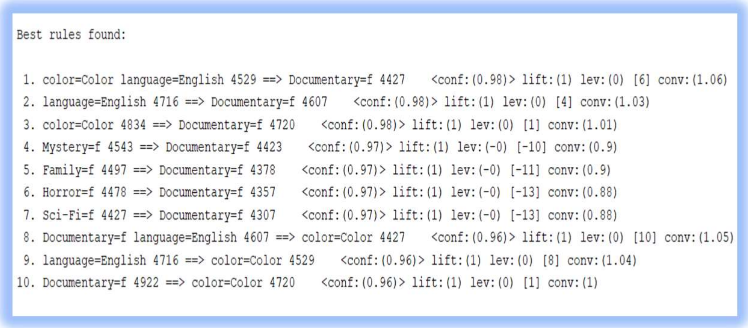
```
Clustered Instances

0        360 (  7%)
1        142 (  3%)
2       1651 ( 33%)
3        602 ( 12%)
4       1086 ( 22%)
5        473 (  9%)
6        729 ( 14%)


Log likelihood: 5.08057
```

As mentioned, the EM algorithm found 7 number of clusters however highest percentage (33) of cluster assigned to cluster 2.Cluster 2 consists of the highest number of voter users who are in the English language with color graphics .If a movie has R content ratings can be more interesting and popular if it will fall under the cluster 2 category. As it has more attributes and instances it is hard to find other relevant information that can make a movie great.

Association Finding: Apriori algorithm

The Apriori algorithm is one such algorithm in ML that finds out the probable associations and creates association rules. WEKA provides the implementation of the Apriori algorithm, which defines the minimum support and an acceptable confidence level while computing these rules. Here, I performed below parameters with selected attributes:

Parameters: -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Attributes: colour, Action, Adventure, Comedy, Crime, Documentary, Drama, Family, Horror, Mystery, Romance, Sci-Fi, Thriller etc.



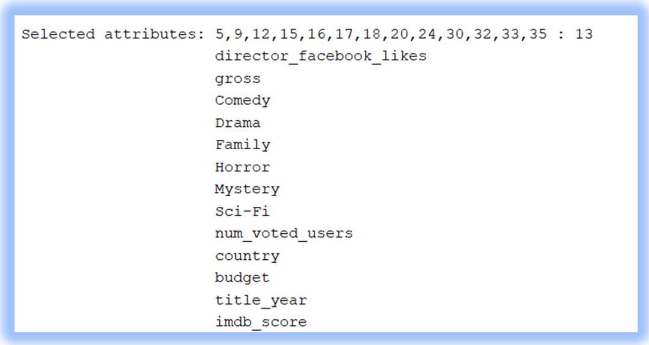
```
Best rules found:

 1. color=Color language=English 4529 ==> Documentary=f 4427    <conf:(0.98)> lift:(1) lev:(0) [6] conv:(1.06)
 2. language=English 4716 ==> Documentary=f 4607    <conf:(0.98)> lift:(1) lev:(0) [4] conv:(1.03)
 3. color=Color 4834 ==> Documentary=f 4720    <conf:(0.98)> lift:(1) lev:(0) [1] conv:(1.01)
 4. Mystery=f 4543 ==> Documentary=f 4423    <conf:(0.97)> lift:(1) lev:(-0) [-10] conv:(0.9)
 5. Family=f 4497 ==> Documentary=f 4378    <conf:(0.97)> lift:(1) lev:(-0) [-11] conv:(0.9)
 6. Horror=f 4478 ==> Documentary=f 4357    <conf:(0.97)> lift:(1) lev:(-0) [-13] conv:(0.88)
 7. Sci-Fi=f 4427 ==> Documentary=f 4307    <conf:(0.97)> lift:(1) lev:(-0) [-13] conv:(0.88)
 8. Documentary=f language=English 4607 ==> color=Color 4427    <conf:(0.96)> lift:(1) lev:(0) [10] conv:(1.05)
 9. language=English 4716 ==> color=Color 4529    <conf:(0.96)> lift:(1) lev:(0) [8] conv:(1.04)
10. Documentary=f 4922 ==> color=Color 4720    <conf:(0.96)> lift:(1) lev:(0) [1] conv:(1)
```

Best Rules:

1. All the English movies are picturized in colour, although they are not documentary as very few movies made on documentary, however we have confidence level of 98% which states given statement is True.
2. Most of the movie are in English but all are on based on documentary, however as per analysis it clearly states that statement is true with confidence level of 98%.
3. If the movies are based on colour graphics, then it might be another genre but not documentary.
4. Many of the movies are in English are not mystery and documentary as confidence level is 97%, thus this statement is correct.
5. Most of the family-oriented movies might not also be a documentary movie.
6. If a movie were not horror 97% it would not be a documentary-based movie.
7. As per mentioned confidence level of 97%, if movie was not Sci-Fi definitely it would not be a documentary buy might be based on action or comedy movies.
8. Given that 96% of documentary movies are neither in English nor in colour.
9. As introduced above 96% of all English movies picturized in colour graphics.
10. To recapitulate, most of the movies which are not based on documentary genre are visualize in colour.

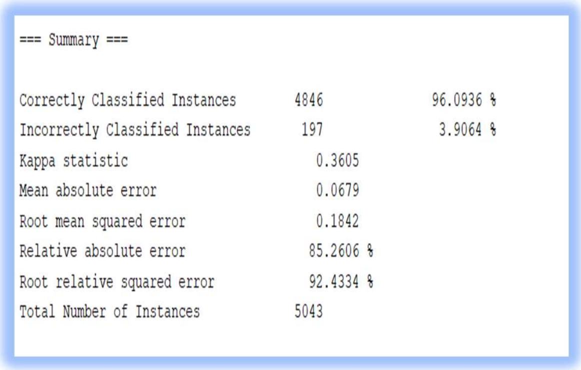
Attribute Selection:

In short, we may examine the entire dataset visually and decide on the irrelevant attributes. This could be a huge task for databases containing many attributes like IMBD dataset Fortunately, WEKA provides an automated tool for feature selection. In below, i just used the wrapper subset here as parameter in the Attribute Selection Mode, by using full training set option.

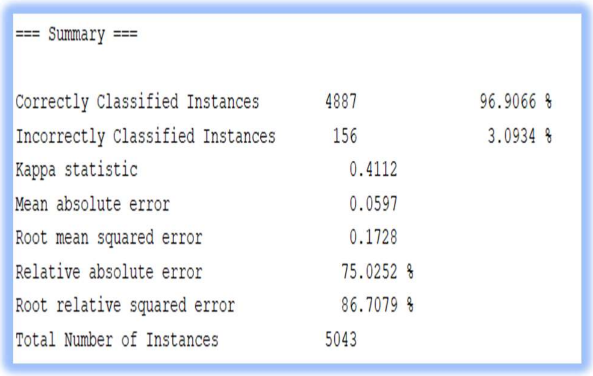


```
Selected attributes: 5,9,12,15,16,17,18,20,24,30,32,33,35 : 13
                     director_facebook_likes
                     gross
                     Comedy
                     Drama
                     Family
                     Horror
                     Mystery
                     Sci-Fi
                     num_voted_users
                     country
                     budget
                     title_year
                     imdb_score
```

In brief, parameter for attribute evaluate WrapperSubsetEval as it picks combination of attributes with J48 classifier with search attribute will be BestFirst parameters -D 1 -N 5 on training set however I got merit for almost 68% and accuracy of 96.0936% by experimenting wrapper again to run on best first attributes which is mentioned in above table so that we may get better results with backward search. However, when I did typically attribute selection process by using excellent classifier J48, hence my performance became slight accurate than before approx. 96.9066 %after losing most of the attributes, thus this dataset has so many relevant and important attributes which may lose accuracy after ignoring them too but it enhanced much better with below result with merit of 96%.



```
=== Summary ===

Correctly Classified Instances        4846              96.0936 %
Incorrectly Classified Instances       197               3.9064 %
Kappa statistic                          0.3605
Mean absolute error                      0.0679
Root mean squared error                  0.1842
Relative absolute error                 85.2606 %
Root relative squared error             92.4334 %
Total Number of Instances             5043
```



```
=== Summary ===

Correctly Classified Instances        4887              96.9066 %
Incorrectly Classified Instances       156               3.0934 %
Kappa statistic                          0.4112
Mean absolute error                      0.0597
Root mean squared error                  0.1728
Relative absolute error                 75.0252 %
Root relative squared error             86.7079 %
Total Number of Instances             5043
```

Conclusion:

In nutshell, WEKA is a powerful tool It provides implementation of several most widely used ML algorithms. Before these algorithms are applied to the dataset, it also allows to Pre-process of the data. The several types of techniques that support are classified under Classify, Cluster, Associate, and Select attributes. The result at various stages of processing can be visualized with a beautiful and powerful visual representation. This makes it easier to quickly apply the various machine learning techniques on this dataset from pre-processing to visualization.

Golden Nugget: As per observation we may not get a more accurate result and it is more difficult too by mentioning each and every algorithm however I found myself more comfortable with J48 and Multilayer Perceptron classifier which illustrates more accurate and best results. In order to understand data, I would like to conclude that majority of the movies are coloured and are in English which is created by USA directors and also I observed that we can’t predict the movie based on the budget, actors, directors because in some cases it was failing but in the majority of the cases it works and we can’t judge the movie by rating or Facebook likes because it was clearly mentioned that the movies which have low rating and likes have collected huge collections. By seeing the votes given by users we can know its success rate.

In brief I would like to conclude my choices for algorithms which I tried to experiment for getting better results moreover again it is quite difficult to mention each and every algorithms along with that dataset has so many attributes and instances which fails to support most of algorithms and weka software perhaps got crash too.i find myself comfortable with following algorithm mentioned above in every Data Mining techniques.