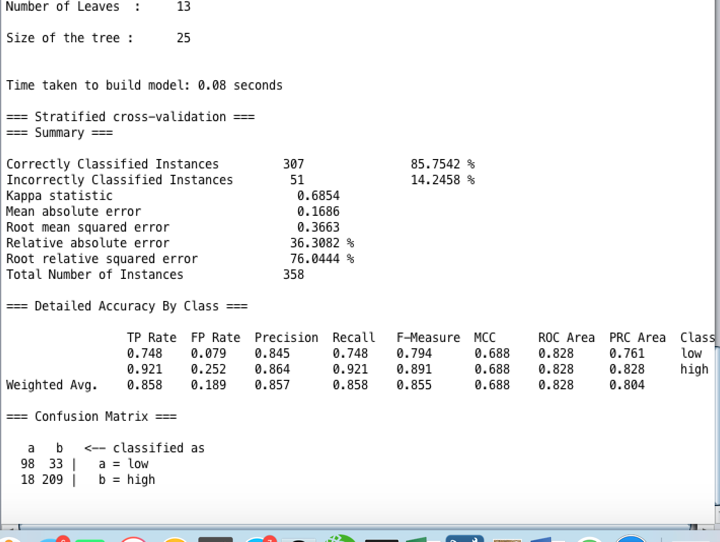
COMP47490 – Assignment 2

15464068 – Viacheslav Borodin

Question 1)-

(a)- Evaluate the performance of two basic classifiers on the dataset:

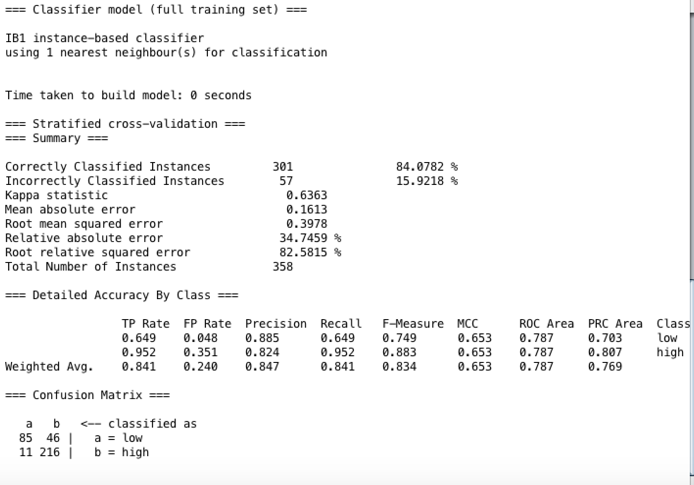
Decision Tress(J48):



Discussion of J48:

As seen from the performance of the J48 Decision tree algorithm the percentage of correctly classified instances was 85.7542% and the number of incorrectly classified instances was 14.2458%. As we can see from the output from above there are 13 leaves in the tree and the total size of the tree is 25. The actual numbers of the correctly and incorrectly classified instances are represented within the confusion matrix at the bottom of the page. The percentage of correctly classified instances is called the sample accuracy. Then in the detailed accuracy by class there are many values such as TP rate which is the rate of true positives, FP rate which is the rate of the false positives, precision the number of instances that are truly of a class divided by the total instances classified of that class, recall number of instances classified divided by the total of that class.

1-NN-

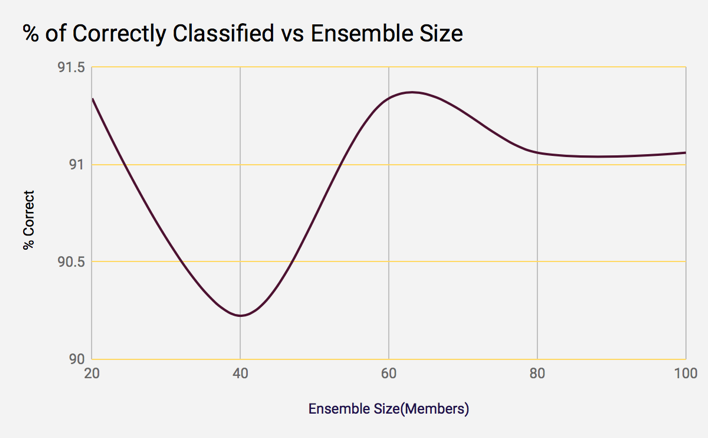


Discussion of 1-NN: As seen from the diagram above the number of correctly classified instances is 301 equivalents to 84.0782% and the number of incorrectly classified instances is 57 equivalents to 15.9218%. The confusion matrix at the bottom represents the actual number of incorrectly classified and correctly classified numbers. If all these numbers are added together, they will represent the total amount of numbers within sample size. 46 is the number of false positives and 11 is the number of false negatives. After analysing both of the performances of the J48 and the 1-NN classifiers the classifier with the best performance is the J48 classifier as it has a higher accuracy when compared to the 1-NN.

(b) Apply ensembles with bagging using both classifiers from a. Investigate performance of both classifiers as ensemble size increasing from 20 to 100 members.

**J-48:**

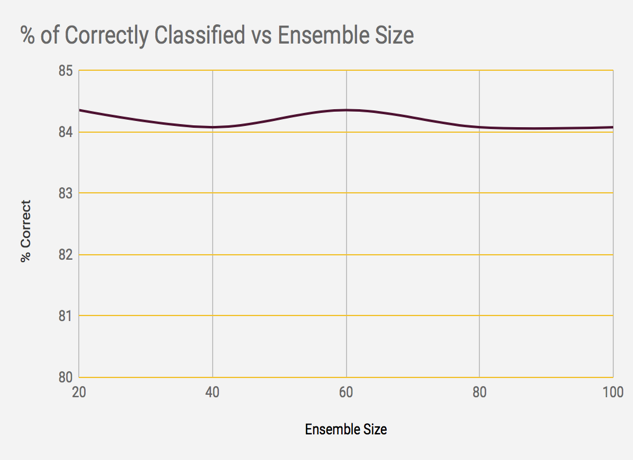
|  |  |  |
| --- | --- | --- |
| **Ensemble Size** | **% of correctly classified Instances** | **% of Incorrectly classified Instances** |
| **20** | **91.3408 %** | 8.6592 % |
| 40 | 90.2235 % | 9.7765 % |
| 60 | 91.3408 % | 8.6592 % |
| 80 | 91.0615 % | 8.9385 % |
| 100 | 91.0615 % | 8.9385 % |



As seen from the table above when bagging is applied it increases the percentage of correctly classified instances. As we can see from the table above with the incorrectly classified and correctly classified instances some of the values are the same even though the ensemble size is bigger. This is because the voting patterns become collinear and the vote stays the same. As we can see from the graph above once the ensemble size hits 80 members the graph begins to plateau this is because no new diversity is added so the ensemble accuracy begins to plateau. If I was to add more ensemble members I would expect little to no change in the %of correctly classified instances.

|  |  |  |
| --- | --- | --- |
| **Ensemble Size** | **% of correctly classified Instances** | **% of Incorrectly classified Instances** |
| **20** | **84.3575 %** | 15.6425 % |
| 40 | 84.0782 % | 15.9218 % |
| 60 | 84.3575 % | 15.6425 % |
| 80 | 84.0782 % | 15.9218 % |
| 100 | 84.0782 % | 15.9218 % |

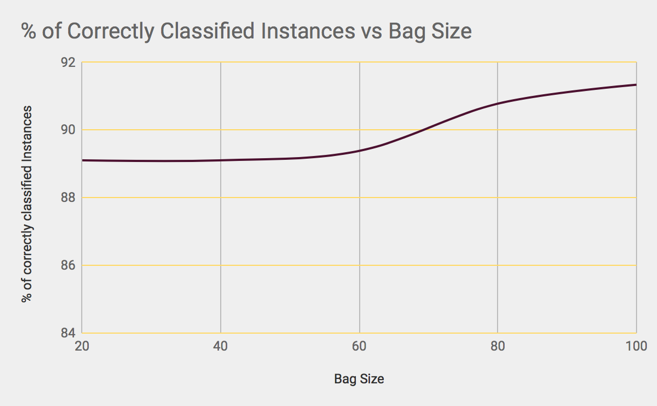
**1-NN:**



As we can see from the table above I decided to choose the bag sizes in 20’s. So I started with 20 and went up in twenties all the way to 100. The table above represents the % of correctly classified and % of incorrectly classified instances. As we can see the % of classified instances is much lower when compared to the J-48 classification. This is because bagging is much more effective when applied with the J-48 classifier. As seen from both the table and the graph representation the graph is very much stable and stays within the same ranges. As seen from the table the % of correctly classified instances is in the range of 84.0782% to 84.3575%. Once again once the ensemble size hits 80 instances the graph begins to hit a plateau as there is no more diversity. Another reason as to why the graph becomes to level off is because the new ensemble members begin to have prediction patterns collinear with the existing members. There is no new diversity added so this graph begins to plateau after 80 ensembles.

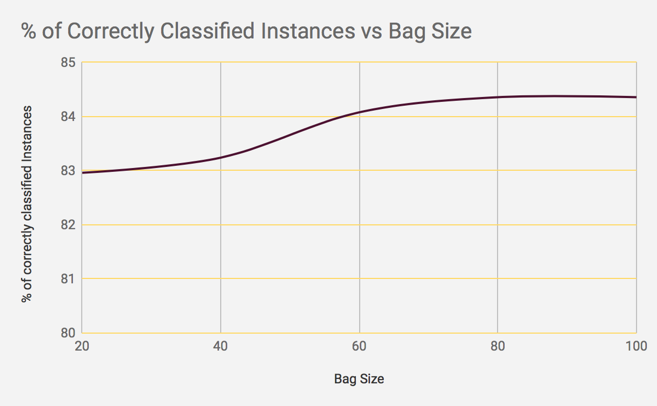
J-48 – Best performing ensemble size = 20

|  |  |  |
| --- | --- | --- |
| **Bag Size** | **% of correctly classified Instances** | **% of Incorrectly classified Instances** |
| 20 | 89.1061 % | 10.8939 % |
| 40 | 89.1061 % | 10.8939 % |
| 60 | 89.3855 % | 10.6145 % |
| 80 | 90.7821 % | 9.2179 % |
| 100 | 91.3408 % | 8.6592 % |



|  |  |  |
| --- | --- | --- |
| **Bag Size** | **% of correctly classified Instances** | **% of Incorrectly classified Instances** |
| 20 | 82.9609 % | 17.0391 % |
| 40 | 83.2402 % | 16.7598 % |
| 60 | 84.0782 % | 15.9218 % |
| 80 | 84.3575 % | 15.6425 % |
| 100 | 84.3575 % | 15.6425 % |

1-NN- Bag size – 20 best ensemble size.

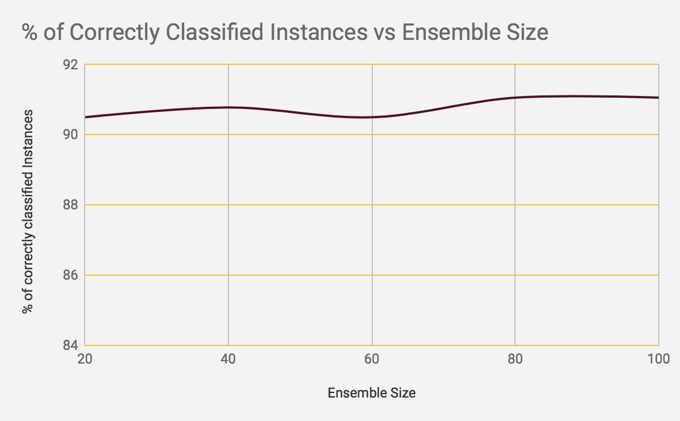


As seen from the two diagrams and tables above the best ensemble sizes were 20 for both the J-48 and the 1-NN classifier. As we can see from the results above bagging is particularly much more effective when applied with the J-48 classification as it is a technique that is well used with models that have a low bias and a high variance. As seen from the tables above the bagging is more effective with j48 as the more ensemble members that are added the higher the % of classified instances there are in comparison to 1-NN where the graph begins to plateau as there is no more diversity after 75 ensembles.

|  |  |  |
| --- | --- | --- |
| **Ensemble Size** | **% of correctly classified Instances** | **% of Incorrectly classified Instances** |
| 20 | 90.5028 % | 9.4972 % |
| 40 | 90.7821 % | 9.2179 % |
| 60 | 90.5028 % | 9.4972 % |
| **80** | **91.0615 %** | **8.9385 %** |
| 100 | 91.0615 % | 8.9385 % |

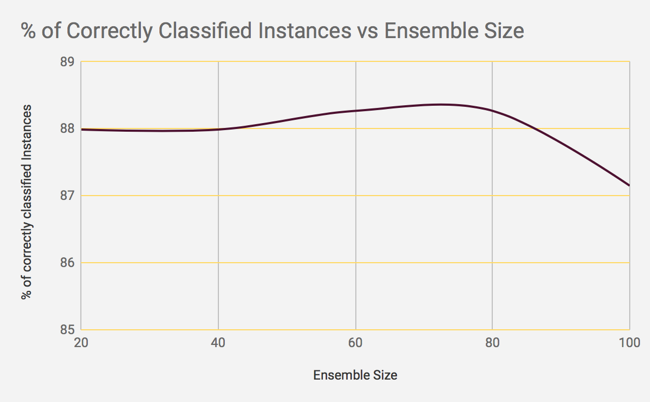
(c)- Random Subspace

**J-48:**



|  |  |  |
| --- | --- | --- |
| **Ensemble Size** | **% of correctly classified Instances** | **% of Incorrectly classified Instances** |
| 20 | 87.9888 % | 12.0112 % |
| 40 | 87.9888 % | 12.0112 % |
| **60** | **88.2682 %** | **11.7318 %** |
| 80 | 88.2682 % | 11.7318 % |
| 100 | 87.1508 % | 12.8492 % |

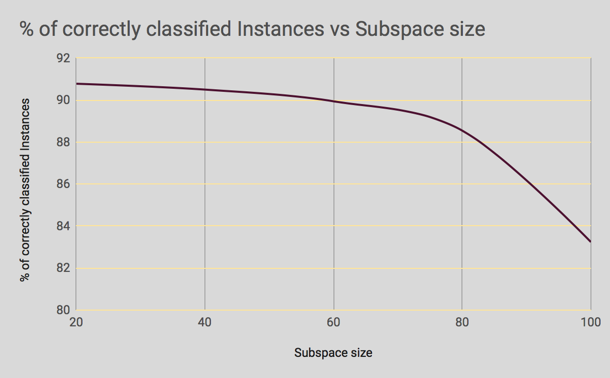
**1-NN:**



As seen from the two diagrams and the two tables representing J-48 and 1-NN classification Both tables represent the correctly classified and incorrectly classified instances when random subspace is applied. As seen above random subspace provides a higher correctly classification when applied alongside the J-48 algorithm. By adding random subspacing this adds more diversity into the ensemble because it uses different features when calculating the distances. The two graphs have more or less the same shape as the ensemble size increases the accuracy tends to drop as the diversity is added and increased. If there were more ensembles added it can be predicted that the accuracy will continue to drop especially with the J48 as there is a sudden drop as seen.

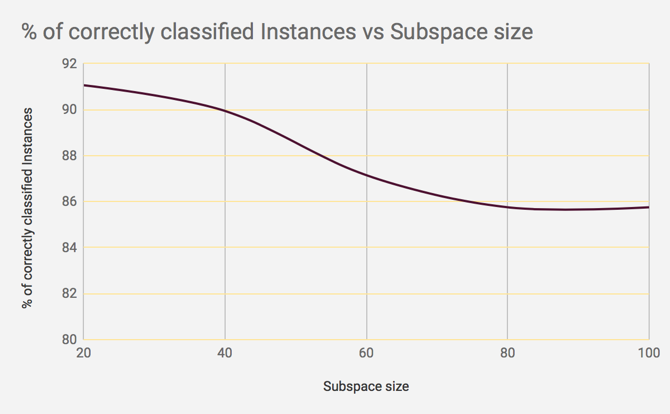
|  |  |  |
| --- | --- | --- |
| **Subspace size** | **% of correctly classified Instances** | **% of Incorrectly classified Instances** |
| 20 | 90.7821 % | 9.2179 % |
| 40 | 90.5028 % | 9.4972 % |
| 60 | 89.9441 % | 10.0559 % |
| 80 | 88.5475 % | 11.4525 % |
| 100 | 83.2402 % | 16.7598 % |

Subspace Size: J-48: 80 iterations is the best



|  |  |  |
| --- | --- | --- |
| **Subspace size** | **% of correctly classified Instances** | **% of Incorrectly classified Instances** |
| 20 | 91.0615 % | 8.9385 % |
| 40 | 89.9441 % | 10.0559 % |
| 60 | 87.1508 % | 12.8492 % |
| 80 | 85.7542 % | 14.2458 % |
| 100 | 85.7542 % | 14.2458 % |

1-NN: 60 iterations Is the best



As seen from the two tables above that for the J-48 algorithm the ensemble that performs the best is 80 and for the 1-NN the ensemble size that performs best is 60. The two graphs are very different as we can see from the J48 algorithm the % of correctly classified instances starts off really high however once the subspace size begin to increase the graphs begin to plummet and as the subspace gets higher the accuracy drops drastically and if the subspace size was to be increased it would continue to drop even further. In comparison the 1-NN performs best with 60 iterations again it starts off with a high accuracy but at about 75 subspace sizes the graph begins to plateau and we can see from 80 to 100 subspace sizes there is very little change as the graph just becomes constant which is not the case compared to the J48 algorithm. The accuracy with random subspacing is higher with the 1-NN in comparison to when bagging is applied to the 1-NN. This is because

Question 2)-

(a)- Bagging is a technique that can be used in order to improve the accuracy of Classification. The reason behind applying sampling to the training set when using bagging is because the algorithm randomly samples the training set with replacement. Bootstrap aggregation is used within bagging where there is random sampling from the training set with replacement, after each sample gets a classifier applied to it. The algorithm trains different subsets of the training data and uses replacement after the training. After the sampling a classifier is applied to each sample. The reason as to why this is done is because it encourages diversity within the ensemble and then it works better for unstable classifiers.

(b)- Stochastic gradient descent: Stochastic gradient descent is used to speed up the learning of the neural networks. Stochastic gradient descent is used to estimate the gradient for a small batch of training algorithms in order to make things work a bit faster. In gradient descent a batch is the total number of examples that are used in order to calculate the gradient in a single iteration. Batch gradient descent assumes that it will be the entire data set which in large datasets may take a very long time in order to run a single iteration. Large datasets may contain redundant data and as the batch size gets bigger there is a higher chance for redundant data. The idea of Stochastic gradient is to use a batch size of 1 per iteration. The only issue with Stochastic gradient is that it is very noisy.

Mini-Batch gradient descent: is a mix of Stochastic gradient descent and full batch iteration. A mini batch gradient descent is usually anywhere from 10 to 1,000 batch examples which are chosen at random. A mini batch gradient reduces the noise in comparison to Stochastic gradient and is more efficient when compared to a full batch. Mini batch gradient would be better used when the batch examples are quite low and smaller datasets.

(c)- A bias term within neural networks is an extra neuron that is added to the network with a value of 1. The bias term is used in conjunction within the neuron in order to send out a signal. It is used to help and shift the activation function to either the left or right which may be beneficial and influential in order to have successful learning. The reason why bias terms are so important is because it provides every node with a constant value on top of the normal inputs that are going to the nodes. The output of the network is computed by multiplying the input by the weight and then passing the result through an activation function.

(d)- When adding hidden nodes in a neural network it increases the set of functions that can be learned. A hidden layer in neural networks is one that is connected to inputs of other neurons and is not visible as an output. The hidden layer is a layer which is hidden in between the input and output layers since the output of one layer is linked to the input of another layer. Hidden layers perform more complex queries. The more hidden layers there are within the neural network the better as it increases the number of complex learning that can be done. Adding a hidden layer within the neural network will also allow you to separate from your training data. A hidden layer will decrease the amount of training errors produced.

(e)- Key Differences between bagging and boosting in ensemble classification:

**Bagging:** Bagging is related to bootstrap aggregation. Bagging is used when you need to reduce the variance but still keeping the bias. Bagging involves sampling the data with replacement in order to generate multiple sets of input data. The main idea of bagging is to reduce variance of the decision tree. The idea is to chose different subsets from the training sample randomly with the idea of replacement. Each subset is used to train the model and at the end the result will be many different trained models. The result is the average of all the training models which is much better than one single sample. Bagging is more useful on models that have low bias.

**Boosting:**  Boosting is an algorithm which is used in order to reduce bias and also variance. Boosting learns weak classifiers and then these weak classifier learners are converted to create a strong classifier. This is usually done by training a sequence of classifiers within a set of training data. By training the sequence of classifiers this helps later classifiers to be trained to better predict class labels in which the earlier ones performed worse in the earlier cases. It is important to identify the patterns and classifiers that were poorly classified where most of the errors occurred. Once the poorly classified classifiers have been identified these are the ones with the highest amount of errors, then these classifiers can be re trained in order to reduce the amount of errors and correctly classify these instances.

(f)-

The role of the activation function in neural networks is quite important and they are significant in every single neural network. The activation function is used to decide which nodes to activate or not. The activation function is applied to hidden and output layers of a neural network. The activation functions take the weighted sum of the inputs plus the bias as the input and apply it to the function which is (w(i)\*x(i)+b) and perform computations in order to decide which nodes to get rid of in the layer. Where w is the weight and x is the input. The activation function plays a significant role in deciding whether the information the neuron is receiving is relevant or should it just be ignored. The activation function applies non-linear transformation to the inputs which then makes it learn and perform more complex tasks.