Contents

INTRODUCTION

The following report is a combined study of various algorithms and techniques which include Support Vector Machines, Decision Trees, pruning, gradient boosting and k-fold cross validation.

We have two datasets:

1. Runtime performance (sgemm_product)
2. Diabetic Retinopathy

We have worked previously with Runtime data only and performed a descriptive analysis in the previous report. The description of the diabetic retinopathy dataset is following:

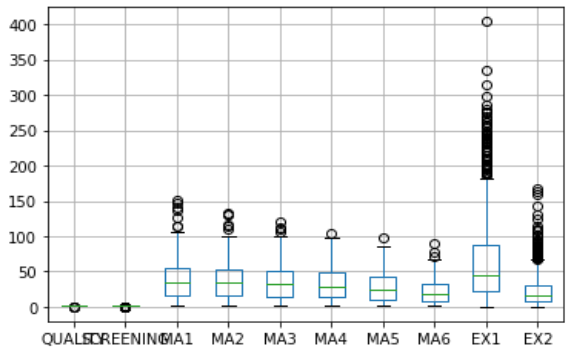
1. The data contains 1151 rows and 20 columns
2. The dependent variable is 'Class' which was UTF-8 encoded and was decoded as 0 (b'\x30') and 1 (b'\x31')
3. The data was scaled to handle the skewness and bring to common scale using [-1,1]
4. The dataset does not have any null values

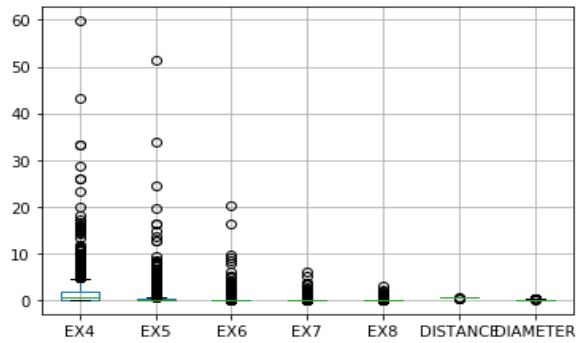
Reason for choosing data Set

People with diabetes can have an eye disease called diabetic retinopathy. This is when high blood sugar levels cause damage to blood vessels in the retina. These blood vessels can swell and leak. Or they can close, stopping blood from passing through. Sometimes abnormal new blood vessels grow on the retina. All these changes can steal your vision.

This can be used in the healthcare industry which will reduce the inaccuracies to test with human eye or could be validation to medical reports. This can affect the patient's life and would be a boon for healthcare artificial intelligence. Also, we have learnt in the class about how neural networks can help us in predictions and going forward it will be interesting to compare the variegated models. This will enhance the model and learning as it contains both numerical and categorical features.

Following is the box plot of features in the dataset:



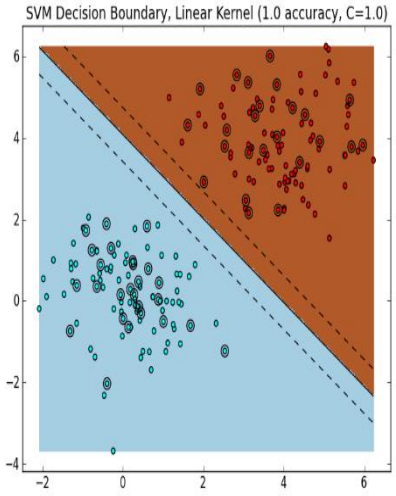


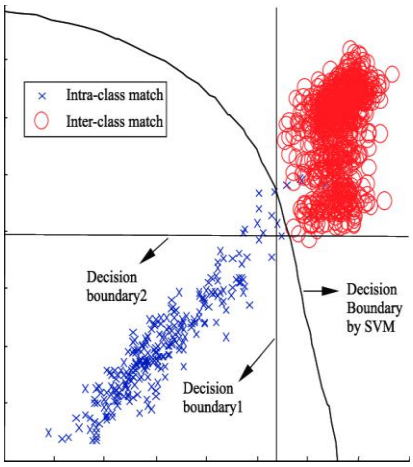
Outlier Treatment

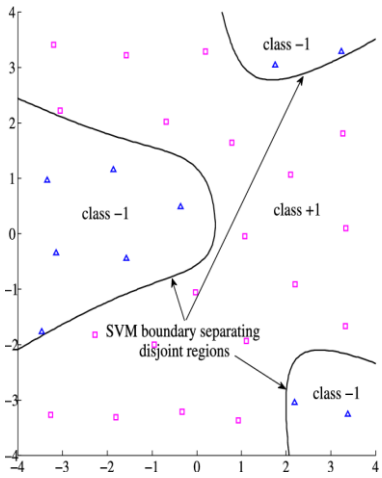
Since, we can see the outliers in MA (Microaneurysms) and EX (Exudates) the urge could be to remove them from the dataset. But, the range of MA and EX is very large and can from individual to individual. In healthcare industry, the patients who are fit would have similar levels of the test while the sick's level can vary depending on severity. Hence, we will not remove the outliers in this case.

SUPPORT VECTOR MACHINE

In machine learning, support-vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis (both linear and non-linear). There are different kernels that can be used depending on the dispersion of data. Each kernel has different boundary condition to cater to the needs of classification or regression problem which we can visualize as follows:







Kernel= 'Linear'

$$k(x_i, x_j) = (x_i \cdot x_j)^d$$

Kernel= 'Rbf or Gaussian'

$$k(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

Kernel= 'Polynomial'

$$K(x, y) = (1 + x \cdot y)^d$$

Accuracies for datasets

Data - Runtime		
Kernel	Train	Test
Linear	83.10%	83.37%
Rbf	96.13%	96.07%
Polynomial	90.89%	91.08%

- Looking at the results table on the left, gaussian or rbf kernel fits the dataset in the best possible way
- There's no linear decision boundary for this dataset, but an RBF kernel can automatically decide a non-linear one

Data - Diabetic Retinopathy		
Kernel	Train	Test
Linear	75.53%	69.36%
Rbf	75.16%	64.45%
Polynomial	69.81%	55.78%

- For this dataset on diabetic retinopathy, linear kernel fits the best
- There is a linear boundary that separates whether the patient suffers from the disease or not

DECISION TREES

Decision Trees are a class of very powerful Machine Learning model cable of achieving high accuracy in many tasks while being highly interpretable. We want to build a tree with a set of hierarchical decisions which eventually give us a result, i.e. our classification or regression prediction. The decisions will be selected such that the tree is as small as possible while aiming for high classification / regression accuracy.

Advantages	Disadvantages
1. With reference to SVM, there was a drastic decrease in the execution time 2. The reason for this is excluding the insignificant variables from the tree	1. The results are often biased depending on the levels of data 2. Overfitting of the data is a common problem if not pruned

Pruning: Pruning is the process of removing the unnecessary structure from a decision tree, effectively reducing the complexity to combat overfitting with the bonus of making it even easier to interpret. There are two methods of pruning:

Gini	Entropy
Computationally less extensive as does not include logarithmic transformation	Take more time to compute as it has logarithmic transformation
Recommended on large datasets	Recommended on smaller datasets
$I_G = 1 - \sum_{j=1}^{c} p_j^2$	$I_H = -\sum_{j=1}^{c} p_j log_2(p_j)$

Initially, both the methods were used initially to see the difference in accuracy, but it escalated the execution time for entropy. Hence, we will use entropy for diabetic retinopathy and gini for runtime data.

Accuracies

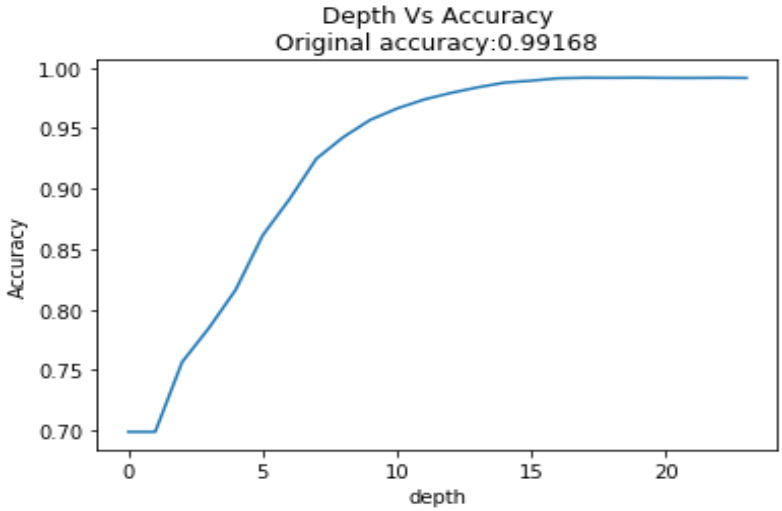
Dataset- Runtime

Data - Runtime		
Decision Trees	Train	Test
Without pruning	100.00%	99.168%
After pruning (depth=18)	100.00%	99.176%

- The decision tree resulted in 99.17% without pruning
- The reason of pruning is to yield maximum accuracy with the least information
- Hence, the classifier was looped with varying depths and the least depth which beats accuracy of the classifier was taken
- This process is called pruning

Let us see how the accuracy varies while changing for the depth in decision trees.

- The plot on right shows that accuracy is constant after certain depths in the decision trees
- The minimum depth which beats the accuracy of unpruned decision tree is 18
- There is always a trade off between accuracy and depth
- Also, pruning is very necessary to avoid overfitting of the data



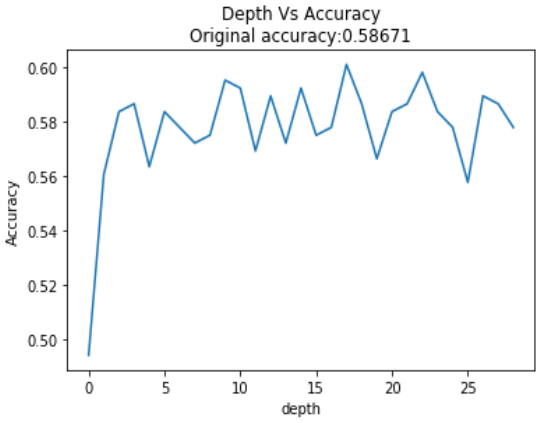
Dataset- Diabetic Retinopathy

Data - Diabetic Retinopathy		
Decision Trees	Train	Test
Without pruning	100.00%	58.67%
After pruning (depth=12)	100.00%	58.96%

- In this case, SVM tends to do a better job with higher accuracy than decision trees
- This often happens and that is why we use ensemble methods to get the best model or average of all models
- We found that pruning at depth=12 has a higher accuracy than unpruned decision tree

Let us see how the accuracy varies while changing for the depth in decision trees.

- We can infer that the accuracy at depth=12 is higher than the accuracy of unpruned decision trees
- The crest and trough in the graph on the right seems the accuracy is varying a lot due to the scale



BOOSTING

Unlike many ML models which focus on high quality prediction done by a single model, boosting algorithms seek to improve the prediction power by training a sequence of weak models, each compensating the weaknesses of its predecessors.

AdaBoost is a specific Boosting algorithm developed for classification problems (also called discrete AdaBoost). The weakness is identified by the weak estimator's error rate:

$$for\ a\ weak\ classifier\ C$$
$$X: n \times d, Y: n \times k\ with\ sample\ weights\ W: n \times 1$$
$$Error = \frac{\sum_{j=1}^{n} W_j I(C(X_j) \neq Y_j)}{\sum_{j=1}^{n} W_j}, I(x) = \begin{cases} 1, & if\ x\ is\ True \\ 0, & if\ x\ is\ False \end{cases}$$

In each iteration, AdaBoost identifies miss-classified data points, increasing their weights (and decrease the weights of correct points, in a sense) so that the next classifier will pay extra attention to get them. The default number of iterations is 50 in Adaboost which seems sufficient in this, but one can alter that.

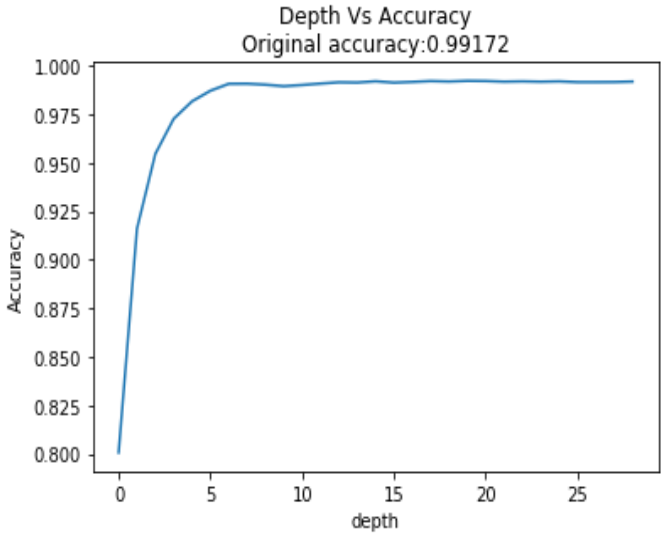
Accuracies

Dataset- Runtime

Data - Runtime		
Decision Trees	Train	Test
Without pruning	100.00%	99.172%
After pruning (depth=15)	100.00%	99.200%

- The accuracy without pruning is same as before without using Adaboost boosting
- But, depth at 15 beats this accuracy
- The earlier optimized depth without using boosting was 18
- This is because Adaboost makes the classifier learn the weak learners and assigns weight to each learner and enhances accuracy

Let us see how the accuracy varies while changing for the depth in decision trees while boosting with Adaboost.



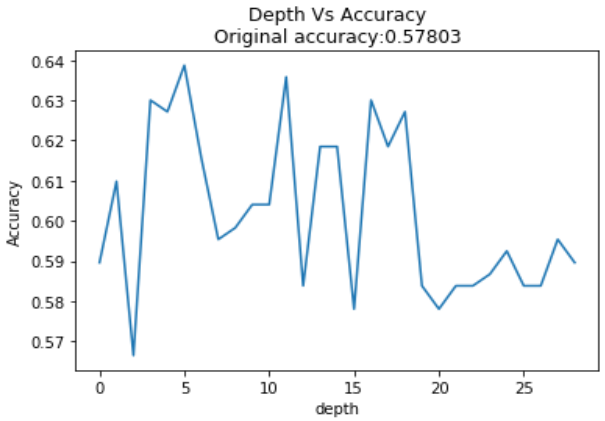
- From the exercise and the graph on the left, we can see the accuracy doesn't increase by a huge difference after depth=15
- The accuracy at depth=15 with boosting is higher than accuracy at depth=18 performed previously without boosting by 3 basis percentage points
- Although, the change is very minimal but with varying depths accuracy can get increase/decrease

Dataset- Diabetic Retinopathy

Data - Diabetic Retinopathy		
Decision Trees	Train	Test
Without pruning	100.00%	57.80%
After pruning (depth=2)	100.00%	62.43%

- The accuracy without pruning is lesser than the one using Adaboost boosting
- Also, depth=2 beats this accuracy
- The earlier optimized depth without using boosting was 12
- This is because Adaboost makes the classifier learn the weak learners in the dataset

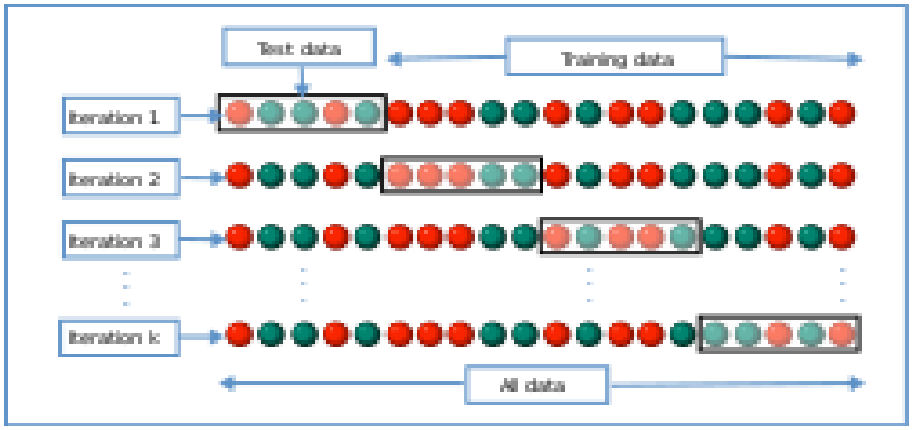
Let us see how the accuracy varies while changing for the depth in decision trees while boosting with Adaboost.



- The accuracy without pruning is lesser than the one using Adaboost boosting by 5 %
- Also, depth=2 beats this accuracy
- The earlier optimized depth without using boosting was 12
- This is because Adaboost makes the classifier learn the weak learners in the dataset

K-FOLD CROSS VALIDATION

One round of cross-validation involves partitioning a sample of data into complementary subsets, performing the analysis on one subset (called the *training set*), and validating the analysis on the other subset (called the *validation set* or *testing set*). To reduce variability, in most methods multiple rounds of cross-validation are performed using different partitions, and the validation results are combined (e.g. averaged) over the rounds to give an estimate of the model's predictive performance.



- The value of k can be varied from 2 to any number
- The value for our analysis is taken as 10

SUPPORT VECTOR MACHINE

Data - Runtime		
Kernel	Train	Test
Linear	82.59%	80.69%
Rbf	95.02%	89.43%
Polynomial	94.01%	81.71%

- The best model is Gaussian or rbf which was concluded before as well
- But the accuracy is lower by 7% as sampling created a bias before and k-fold is more reliable
- The value of k=10 here and the accuracy is the average of all the scores

Data - Diabetic Retinopathy		
Kernel	Train	Test
Linear	69.33%	70.14%
Rbf	64.47%	66.02%
Polynomial	61.86%	63.09%

- Looking at the results table on the left, linear kernel fits the dataset in the best possible way
- There's a linear decision boundary for this dataset, which separates the dependent variable with 70.14% accuracy

DECISION TREES

Data - Runtime		
Decision Trees	Train	Test
Without pruning	100.00%	77.685%
After pruning (depth=18)	99.85%	76.254%

- The decision tree resulted in 77.68% without pruning
- The depth=18 was used previously where we got the maximum accuracy, but cross validation did not beat that accuracy
- This is possible as sampling can create a bias in the results

Data - Diabetic Retinopathy		
Decision Trees	Train	Test
Without pruning	100.00%	61.773%
After pruning (depth=12)	87.55%	63.421%

- The decision tree resulted in 61.77% without pruning
- The depth=12 was used previously where we got the maximum accuracy of 58%, but cross validation resulted in higher accuracy
- The value of k was 10 and the accuracy is average of all the scores

BOOSTING

Data - Runtime		
Decision Trees	Train	Test
Without pruning	99.57%	77.693%
After pruning (depth=15)	100.00%	77.848%

- The decision tree with boosting does not have any effect on the accuracy for unpruned tree
- But, for optimum depth=15, the accuracy improved by 1% with cross validation

Data - Diabetic Retinopathy		
Decision Trees	Train	Test
Without pruning	100.00%	60.992%
After pruning (depth=2)	95.71%	63.990%

- The accuracy has improved by 2% with cross validation as compared to 58% previously
- This is due to sampling that could have created bias previously and resulted in low accuracy

CONCLUSION

Dataset- Runtime

1. The best model turned out to be decision trees when pruned at depth=15 with accuracy 99.20% on test dataset
2. The pruning method used was gini as it takes less computational time due to lack of logarithmic function unlike entropy
3. The K-cross validation accuracy for Gaussian kernel yielded 89.43% accuracy which was the second-best model as sampling can often create bias

Dataset- Diabetic Retinopathy

1. The best model is K-fold cross validation in SVM with Linear kernel with an accuracy of 70.14%
2. Therefore, there exists a linear boundary that separates whether the patient suffers diabetic retinopathy or not