CSE 590-04 Homework 2

Aliaa Elshamekh

February 22, 2021

1 Introduction

Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y). In this homework, we have movies reviews data set and the objective is to use some classification algorithms in predicting whether the review of the movie is *negative (0)* or *positive (1)*.i.e., it is a binary classification problem. From a modeling perspective, classification requires a training dataset with many examples of inputs and outputs from which to learn. A model will use the training dataset and will calculate how to best map examples of input data to specific class labels. As such, the training dataset must be sufficiently representative of the problem and have many examples of each class label Over-fitting is a common problem in machine learning which can occur in most models. k-fold cross-validation can be conducted to verify that the model is not over-fitted. In this method, the data-set is randomly partitioned into k mutually exclusive subsets, each approximately equal size and one is kept for testing while others are used for training. This process is iterated throughout the whole k folds.

Furthermore, Machine learning models are parameterized so that their behavior can be tuned for a given problem. These models can have many parameters and finding the best combination of parameters can be treated as a search problem.A model hyperparameter is a configuration that is external to the model and whose value cannot be estimated from data.They are often used in processes to help estimate model parameters. You cannot know the best value for a model hyperparameter on a given problem. You may use rules of thumb, copy values used on other issues, or search for the best value by trial and error. When a machine learning algorithm is tuned for a specific problem then essentially you are tuning the hyperparameters of the model to discover the parameters of the model that result in the most skillful predictions.

One simple strategy we are going to use in this homework is *GridSearching* which is an approach to hyperparameter tuning that will methodically build and evaluate a model for each combination of algorithm parameters specified in a grid.

In order to compare the performance of multiple machine learners, meaningful numeric measures of performance must be chosen. In this case, a performance measure is any way of assigning a number to measure how well a classifier's predicted classes for the test set.Another consideration for establishing the performance of a given learner on a given dataset is how to partition the set into training and testing sets. When preforming a single trial with one train/test partition, the learner's performance may be sensitive to the chosen partition match up with the ground-truth classes. A better way to handle this is the k-fold cross-validation

In this homework , we are asked to use 4 folds validation , so the training data will be portioned as shown in figure 1

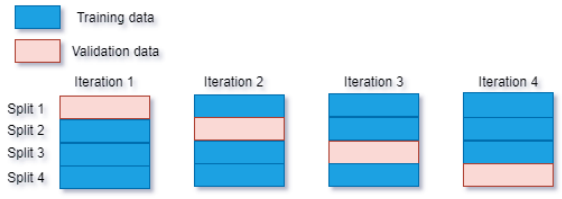

Figure 1: Example of a 4-fold cross-validation data split.

2 Logistic regression Classifier

Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable.

In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.

There are many assumptions for Logistic Regression Assumptions :

- it requires the dependent variable to be binary.
- the factor level 1 of the dependent variable should represent the desired outcome.
- The independent variables should be independent of each other. That is, the model should have little or no multicollinearity.
- The independent variables are linearly related to the log odds.
- Logistic regression requires quite large sample sizes.

After splitting the data using 4 folds technique , we train the logistic regression classifier , we can see that the model accuracy changes slightly across the 4 folds with initial value of $C = 1.$ as shown in figure 2

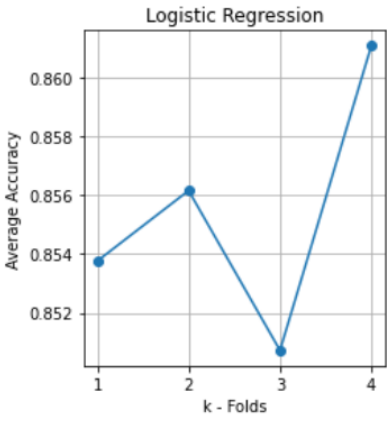

Figure 2: The accuracy of logistic regression model with 4-folds cross validation (C=1) .

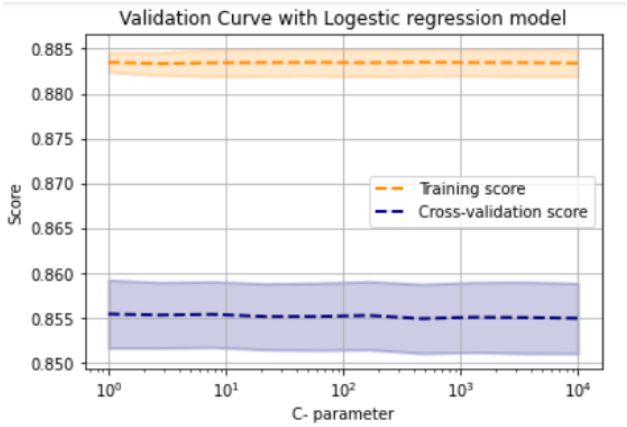

Figure 3: Logistic regression accuracy versus the different values of C .

The parameter C is the regularization parameter where $C = \frac{1}{\lambda}$. λ controls the trade-off between allowing the model to increase it's complexity as much as it wants with trying to keep it simple.For example, if λ is very low or 0, the model will have enough power to increase it's complexity *(over-fit)* by assigning big values to the weights for each parameter. If, in the other hand, we increase the value of λ , the model will tend to *(under-fit)* , as the model will become too simple.

Parameter C will work the other way around. For small values of C, we increase the regularization strength which will create simple models which under-fit the data. For big values of C, we low the power of regularization which implies the model is allowed to increase it's complexity, and therefore, over-fit the data.

The gridsearch technique will be used to find the best value of C.In figure 3 we can see an increase in the accuracy score, but there is a sufficient amount of growth in the execution time as well.The larger the grid, the more execution time. Moreover,in figure 3 , the cross validation score is almost the same with the change of the value of C.

The idea is to choose that C which offers the smallest difference between the training and testing accuracy. In table 1 ,we can see the performance and parameter of the best model in our experiment.

Train accuracy	Test accuracy	Best accuracy(Grid Search)	Best C
0.882	0.859	0.855	7.74

Table 1: Best Logistic regression model results

Figure 4 shows the confusion matrix of the model specified in table 1 , we can see how many mispredicted samples using our model , we have $3,528$ samples that are predicted wrong in the test data (never seen before).

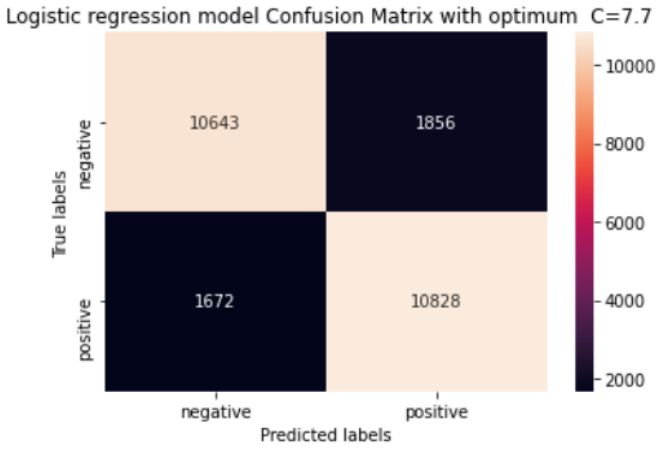

Figure 4: Logistic regression model Confusion Matrix $C = 7.7$

One of the advantages of using logestic regression classifier is that we can extract feature Importance. These values help us to feature selection process. We can drop or ignore some unimportant features to speed model training up. In figure 5 we can see the highest 8 importance for the features in the movies data set according to Logistic regression classifier.

To sum up, the strongest feature in movies data set is *0.0.599*. When this feature is one , it increases the odds of being positive class by a factor of 1.55 when all other features remain the same.

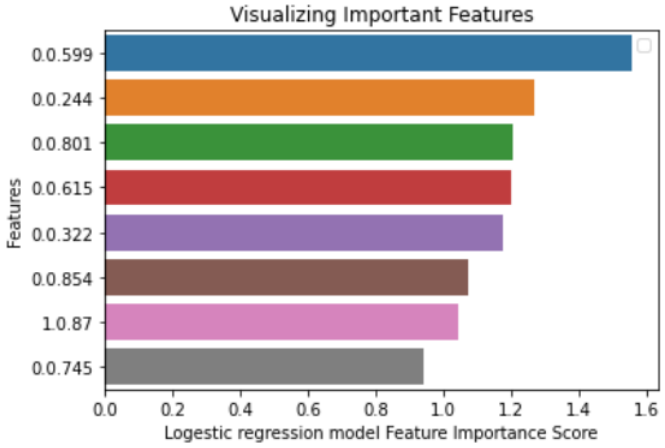

Figure 5: Logistic regression Important features for best model $C = 7.7$

3 Naive Bayes Classifier

Naive Bayes classifiers are built on Bayesian classification methods. These rely on Bayes's theorem, which is an equation describing the relationship of conditional probabilities of statistical quantities.

A Naive Bayes model assumes that each of the features it uses are conditionally independent of one another given some class, in other words as we say in the statistical inference class, they are *iid*. More formally, if I want to calculate the probability of observing features $x_1, \ldots x_n$, given some class c, under the Naive Bayes assumption the following equality holds for Bayes Rule :

$$p(x_1, ..., x_n|c) = \prod_{i=1}^{n} p(x_i|c)$$

Which means that if I am using a Naive Bayes model for classifying a new sample , the posterior probability will be

$$p(c|x_1, ..., x_n) \propto p(c)p(x_1|c)...p(x_n|c)$$

In this task, we are going to use *Bernoulli Naive Bayes* classifier since the data is discrete, in particular features are only in binary form.This classifier works for Bernoulli distribution . The main feature of Bernoulli Naive Bayes is that it accepts features only as binary values like true or false, yes or no, success or failure, 0 or 1 and so on. The pmf of Bernoulli distribution can be written as : $X \sim Ber(p)$

$$f(x; p) = px + (1 - p)(1 - x) \qquad \forall x \in \{0, 1\}$$

Consider a corpus of reviews (training data) whose class is given by $C = 1, 2, ..., K$. Using Naive Bayes, we classify a review R as the class which has the highest posterior probability $argmax_{k=1,2,...,K} p(C = k|R)$, which can be re-expressed using Bayes' Theorem:

$$p(C = k|R) = \frac{p(C = k)\, p(R|C = k)}{p(R)} \propto p(C = k)\, p(R|C = k)$$

Where: $p(C = k)$ represents the class k's prior probabilities, $p(R|C = k)$ is the likelihoods of the document given the class k and $p(R)$ is the normalizing factor which we don't have to compute since it does not depend on the class C . i.e. this factor will be the same across all class C , thus the numerator will be enough to determine which $p(C = k|R)$ is the largest.

Now Suppose we have a (features) X containing a set of n words and the t_{th} dimension of a review vector corresponds to word x_t in the features. Following the Naive Bayes assumption , that the probability of each word occurring in the review is independent of the occurrences of the other words, we can then rewrite the i_{th} review's likelihood $p(R_i|C)$ as

$$p(R_i \mid C) = \prod_{t=1}^{n} b_{it} p(x_t \mid C) + (1 - b_{it})(1 - p(x_t \mid C))$$

where $p(x_t|C)$ is the probability of word x_t occurring in a review of class C and b_{it} is either 0 or 1 representing the absence or presence of word x_t in the i_th review.

This product goes over all words in the features(vocabulary). If word x_t is present, then $b_{it} = 1$ and the associated probability is $p(x_t|C)$; If word x_t is not present, then $b_{it} = 0$ and the associated probability becomes $1 - p(x_t|C)$.

Then, we find the predicted class by maximizing posterior probability $p(y|\mathbf{x})$ Same as we did in the previous classifier , we use 4 fold cross validation for the training data . In figure 6 we can see how the accuracy changes between the folds slightly changes from one fold to another.

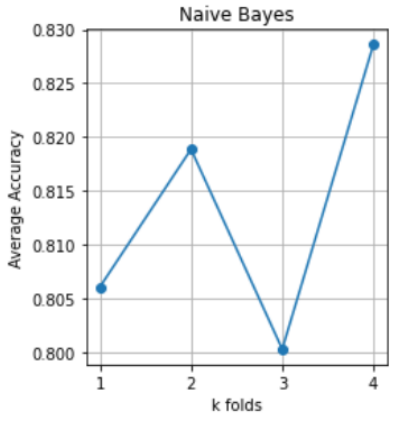

Figure 6: The accuracy of NB model with 4-folds cross validation ($\alpha = 1$) .

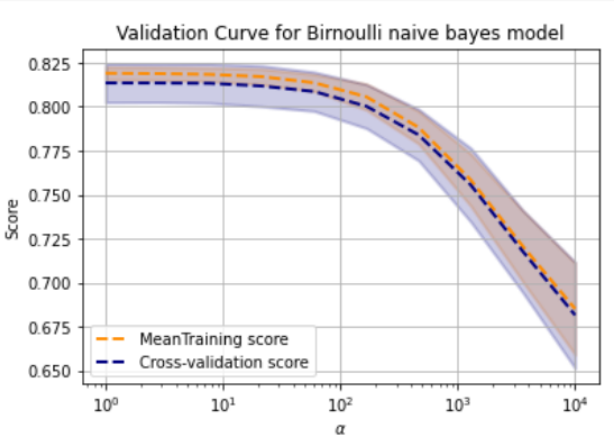

Figure 7: Naive Bayes model accuracy versus the different values of α .

Tuning of α parameter is very important,since it is the smoothing parameter.Smoothing allows Naive Bayes to better handle cases where evidence has never appeared for a particular category. The problem that we can see evidence values in the test data that were not seen in the training data.It can be solved by Laplace smoothing in Naive Bayes

classifier.In order to maintain a probability distribution,we make all events possible but reduce the probability to minimize the impact,such as adding a trivially small number.It comprises Laplace smoothing, add-k smoothing, good-turing, etc.It prevents probabilities from changing extremely due to some instances (imitate confidence),and stops us from discarding unseen events with utilitarian information, and limits plausible predictions as we don't see every possible attribute-value pair.Although,smoothing can estimate the probability of unseen events and create bias.

The idea is to choose that α which offers the smallest difference between the training and testing accuracy. In table 2 ,we can see the performance and parameter of the best model in our experiment.

Train accuracy	Test accuracy	Best accuracy(Grid Search)	Best α
0.818	0.815	0.813	0.01

Table 2: Best Bernoulli Naive Bayes model results

Figure 8 shows the confusion matrix of the model specified in table 2 , we can see how many mispredicted samples using our model , we have $4,622$ samples that are predicted wrong in the test data (never seen before).

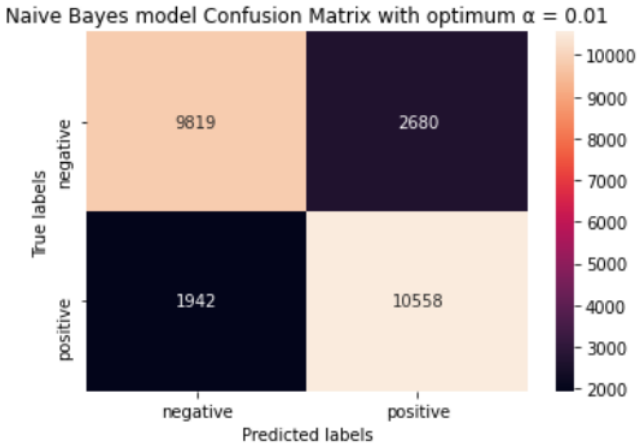

Figure 8: NB model Confusion Matrix $\alpha = 0.01$

Finally,it is worth to mention that this classifier is extremely fast for both training and prediction ,and have few or none parameters to tune.

4 Random Forest Classifier

Random forest is an ensemble tool which takes a subset of observations and a subset of variables to build a decision trees.It builds multiple such decision tree and amalgamate them together to get a more accurate and stable prediction.This is direct consequence of the fact that by maximum voting from a panel of independent judges, we get the final prediction better than the best judge.

We generally see a random forest as a black box which takes in input and gives out predictions,without worrying too much about what calculations are going on the back end.This black box itself have a few levers we can play with.Each of these levers have some effect on either the performance of the model or the resource –time balance. In this article we will talk more about these levers we can tune, while building a random forest model.At each node,the algorithm choose small subset of variables at random,and a variable (and a value for that variable) which optimizes the split.

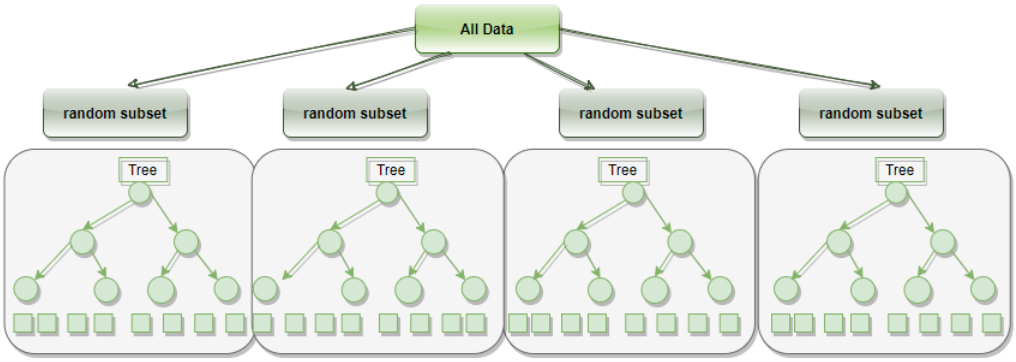

Figure 9: Example of Random forest

Random forest consists of a number of decision trees.Every node in the decision trees is a condition on a single feature, designed to split the dataset into two so that similar response values end up in the same set.The measure based on which the (locally) optimal condition is chosen is called impurity.For classification, it is typically either Gini impurity or information gain/entropy and for regression trees it is variance.Thus when training a tree, it can be computed how much each feature decreases the weighted impurity in a tree.For a forest,the impurity decrease from each feature can be averaged and the features are ranked according to this measure.

Here is how such a system is trained; for some number of trees T:

1. Sample N cases at random with replacement to create a subset of the data (see top layer of figure 9). The subset should be about 66% of the total set.
2. The predictor variable that provides the best split, according to some objective function, is used to do a binary split on that node.
3. At the next node, choose another m variables at random from all predictor variables and do the same.for random forests : m << number of predictor variables. There are some possible values for m as : $\frac{\sqrt{m}}{2}$,$\sqrt{m}$,$2\sqrt{m}$.

Parameters in random forest are either to increase the predictive power of the model or to make it easier to train the model.Initially,we have trained the model using 4- fold cross validation with $n\_estimators = 50$ and all other parameters set to the default values. The model scores in each fold was as shown in figure 10

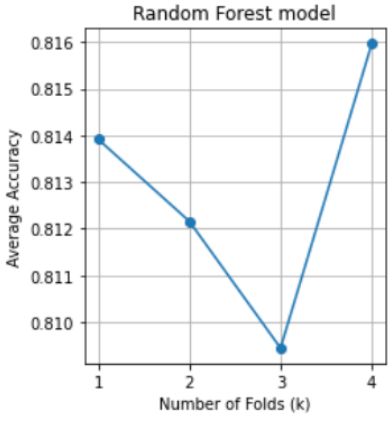

Figure 10: The accuracy of Random Forest model with 4-folds cross validation ($n\_estimators = 50$) .

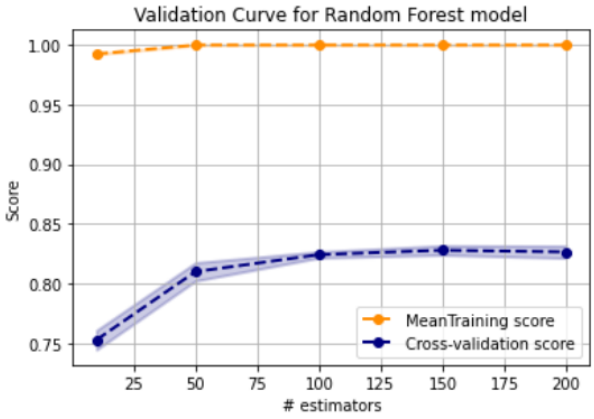

Figure 11: Random Forest model accuracy versus the different number of estimators.

As expected since working with Random forest which is an eager classifier/learner , the training accuracy is almost 1 , however, we can see that validation/testing accuracy in the cross validation is not good enough, since 50 estimator (tree) in the model is not good (make the model not general).

In the next step we varied the number of estimators in our model.This is the number of trees you want to build before taking the maximum voting or averages of predictions. Higher number of trees give you better performance but makes your code slower. You should choose as high value as your processor can handle because this makes your predictions stronger and more stable.Using Gridsearch algorithm,we can say that the model performance is better when *n_estimators* is 200.

The performance of the best random forest model in this experiment is shown in table 3

The feature importance derived from decision trees can explain non-linear models perfectly. Decision trees make splits that maximize the decrease in impurity. By calculating the mean decrease in impurity for each feature across all tress we can know the feature importance.

Train accuracy	Test accuracy	Best accuracy(Grid Search)	n_estimators
1.000	0.833	0.830	200

Table 3: Best Random Forest model results

Gradient boosting machines and random forest have several decision trees. We will calculate feature importance values for each tree in same way and find average to find the final feature importance values.

In figure 12, we can see the top 5 highest ranked feature when we are using Random forest classifier with 200 tree. The importance is low as I believe we need to expand the number of trees in the model more may be up to 500 .

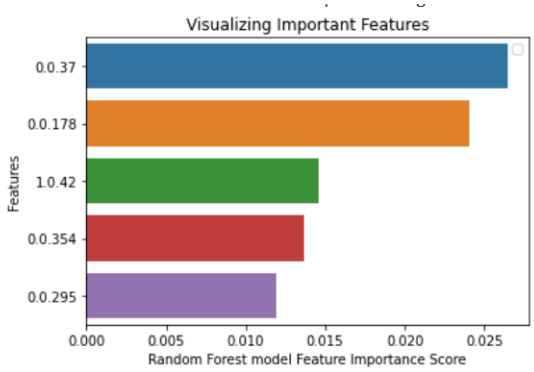

Figure 12: Random forest feature importance for n_estimatores = 200

5 Gradient Boosted Classifier

Boosting is a sequential technique which works on the principle of ensemble. It combines a set of weak learners and delivers improved prediction accuracy. At any instant t, the model outcomes are weighed based on the outcomes of previous instant $t-1$.

To find weak rule, we apply base learning (ML) algorithms with a different distribution. Each time base learning algorithm is applied, it generates a new weak prediction rule. This is an iterative process. After many iterations, the boosting algorithm combines these weak rules into a single strong prediction rule.For choosing the right distribution, here are the following steps:

1. The base learner takes all the distributions and assign equal weight or attention to each observation.
2. If there is any prediction error caused by first base learning algorithm, then we pay higher attention to observations having prediction error. Then, we apply the next base learning algorithm.
3. Iterate Step 2 till the limit of base learning algorithm is reached or higher accuracy is achieved

Finally, it combines the outputs from weak learner and creates a strong learner which eventually improves the prediction power of the model. Boosting pays higher focus on examples which are misclassified or have higher errors by preceding weak rules.

The outcomes predicted correctly are given a lower weight and the ones miss-classified are weighted higher. This technique is followed for a classification problem while a similar technique is used for regression.

First Run a baseline model without tuning , the baseline classifier using 4-fold cross validation has the performance on each fold shown in figure 13. This basline classifier has *learning_rate=0.1, n_estimators=100,max_depth=3* and *max_features='sqrt'*

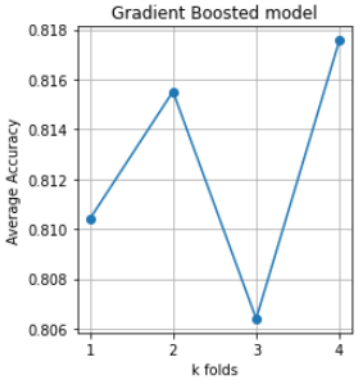

Figure 13: GB model for 4 fold cv case

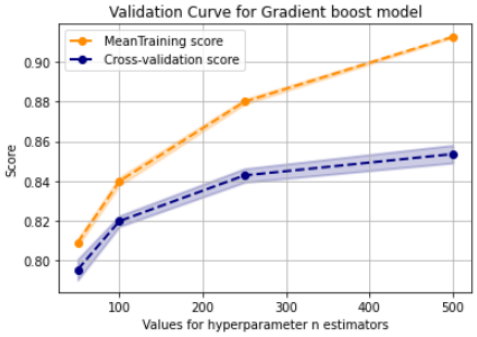

Figure 14: The accuracy of GB model : Tuning the hyperparameter n_estimators .

Tuning parameters is the next step as we did before.Many strategies exist on how to tune parameters. I think that *number of trees, tree depth and the learning rate* are most crucial parameters.

Hence, we will start off with these three and then move to other tree-specific parameters and the subsamples. I will use 4-fold cross validation and evaluate models based on accuracy. For tuning *n_estimators and Learning rate*,n_estimators captures the number of trees that we add to the model. A high number of trees can be computationally expensive. Generally, with a change in learning rate,n_estimators should also be adjusted (4-fold decrease in learning_rate should go in line with a approx. 4-fold increase in n_estimators.We can see the performance of the GB model in this case in table 4.

We can see the validation curves for no of estimators and learning rate in figure 14 and 15 respectively.

Train accuracy	Test accuracy	Best accuracy(Grid Search)	n_estimators	learning_rate
0.920	0.857	0.857	500	0.15

Table 4: GB best model using gridsearch on n estimators and learning rate

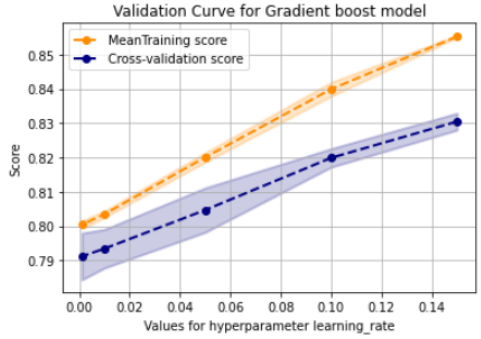

Figure 15: GB model : Tuning the hyperparameter learning_rate.

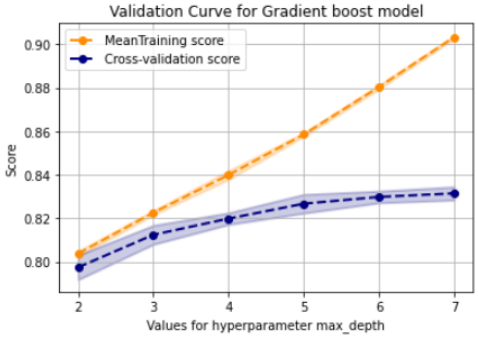

Figure 16: GB model : Tuning the hyperparameter max_depth.

We can see that the accuracy for training and validation accorss the 4 folds improves by increasing the number of trees in the model.However, the increase of these two parameters need much memory and time.

max_depth bounds the maximum depth of the tree. We can use the obtained results to tune the *max_depth* parameter, we can see the accuracy validation curve while tuning this parmater in figure 16

The updated results is shown in table 5

Train accuracy	Test accuracy	Best accuracy(Grid Search)	max_depth
0.981	0.857	0.857	5

Table 5: GB best model using gridsearch max_depth

Finally we combine all the best parameters together and build a new model with the optimized parameters, the results were training score = 0.98 and test score = 0.87.

The imporatnce of fetaures based on the new model are shown in figure 17

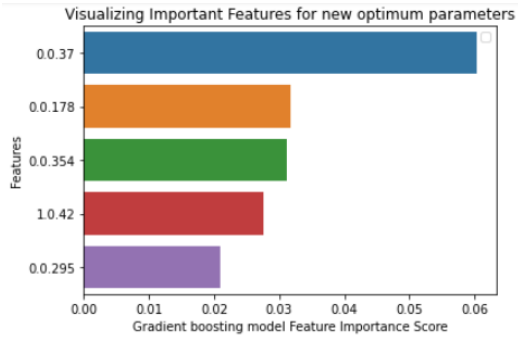

Figure 17: Top 5 highest features importance for new GB model.

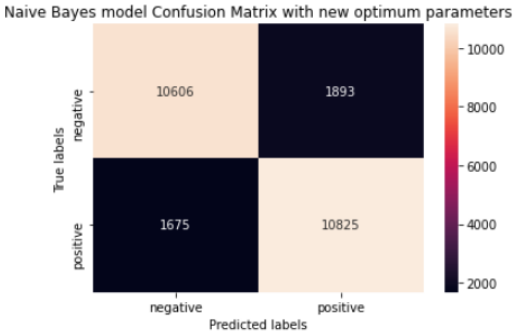

Figure 18: Confusion matrix of the new optimized GB model.

6 Conclusion

- In this homework , we use a binary classification problem to predict whether a review is negative or positive . We start by using 4 fold cross validation technique for model improvement .Then , we use a gridsearch for parameter tuning for each of the 4 classifiers.

	Logistic Regression	Bernoulli Naive Bayes	Random Forest	Gradient Boost
Train accuracy	0.882	0.818	1	0.981
Test accuracy	0.859	0.815	0.833	0.857

Table 6: Summary of the various model performance on testing and training data

- Based on table 6, we see that Gradient Boost gives the best overall accuracy,so for classification,this is the model we would want to use for this data set.However,all classifiers perform well, so one might consider other factors such as computational complexity when choosing a model.
- Random Forest and Gradient Boost Classifier take so much time and consume a lot of memory so that it can do the parameter tuning as whenever we increase no of trees in the model,it becomes more complex, more accurate but takes long execution times.
- Although Naive Bayes Classifier seems to be very simple,but it gives an impressive results and it is a simple model that has a reasonable execution time , which I believe is always a good classifier to start any experiment by it first.
- Random forest and Gradient boost have the ability to rank the features,which helps a lot in interpreting the model and find the more informative features in the dataset.We we know that we can interpret the model.