Project 1 Analysis Report

Yu Feng

GTID Name: yfeng40

1. Introduction

This report is to use five basic Machine Learning algorithms to analyze two classification use cases to find optimized classifier for each use case. The five chosen Machine Learnings are Decision Tree, Neural Networks, Boosting, Support Vector Machines and K Nearest Neighbors. The two use cases I chose are Mexican COVID Death classification and Credit Score classification.

2. Use case description

2.1 Mexican COVID Death classification

The Mexican COVID Death dataset is from Kaggle shared dataset from Mexico government (https://www.kaggle.com/datasets/meirnizri/covid19-dataset?resource=download). It's an important problem because during the last 3 years, there are over 677 million infections and 6.78 million deaths across the world. I also have friends and colleagues told me the death of their friends and families across US, China and India. It's both of social importance and personal interest to build a classifier to find out the main causal of death. The dataset has 1 million unique patients and over 76k death cases.

2.2 Credit Score classification

During the COVID time, I always hear of people rely on credit cards to pay for their food. It would be interesting to find out what features causes Bank to evaluate the credit score. And credit score is important for our lives to get loan and mortgage. It's a personal interest to investigate into this problem. The dataset used in this analysis is 100k person credit score classification dataset from Kaggle (https://www.kaggle.com/datasets/parisrohan/credit-score-classification).

3. Mexican COVID Death classification analysis

3.1 Data processing and feature engineering

There are 1 million records and 19 useful columns in the dataset. The columns used are MEDICAL_UNIT, SEX, PATIENT_TYPE, DATE_DIED, INTUBED, PNEUMONIA, AGE, PREGNANT, DIABETES, COPD, ASTHMA, INMSUPR, HIPERTENSION, OTHER_DISEASE, CARDIOVASCULAR, OBESITY, RENAL_CHRONIC, TOBACCO, CLASIFFICATION_FINAL, ICU.
Most of the columns are categorical data. Therefore, one hot encoding is used for most of the columns to generate features except AGE and DATE_DIED column. AGE is numerical feature itself, normalization might be needed for non-tree-based Machine Learning algorithms. DATE_DIED column is converted to target column as of death or not. After one hot encoding, the final features number increase from 19 to 66.

3.2 Train test data split

The Train test data split ratio is 80:20. After splitting, train data has 838,860 records with 61,539 deaths (7.336%). While test dat a has 209,715 records with 15,403 deaths (7.345%).

3.3 Train model

3.3.1 Decision Tree

Decision Tree is a classic Machine Learning algorithm. The first tuning parameter is max_depth of Decision Tree. After testing max depth from 3 to 20 with 5 folds cross validation, result shows max_depth of 12 is the best. As shown in Figure 1. The surprising part is, the best max_depth is 12. For lots of other use cases, the normal tree depth I experienced before is only in range of 6-8. This shows, since there are so many features and there so many root-cause, more tree depth is needed to learn this use case.
Then best criterion of gini, entropy and log_loss are also tested. Eventually find out gini criterion is the best for this use case. As shown in Figure 2.
After that, best alpha for Minimal Cost-Complexity Pruning from 0 to 0.035 step by 0.0025 is tested to find out best pruning alpha. Surprisingly, the best alpha is 0.000. As shown in Figure 3. The reason of it might be the causal of death are combination of various causal, more co-existing other diseases with various combination will lead to higher chance of death. Or say, the fewer other disease there are the higher survive rate it will be. Therefore, the complicated disease requires more levels of tree to have a better representation. This makes pruning leading to lower model accuracy. The best Cost-Complexity Pruning alpha is 0 shows any pruning will lead to worse accuracy in test dataset.
Eventually, the best Decision Tree model is trained with max_depth of 12, gini criterion and 0 pruning, which leads to 94.83% accuracy on test dataset.

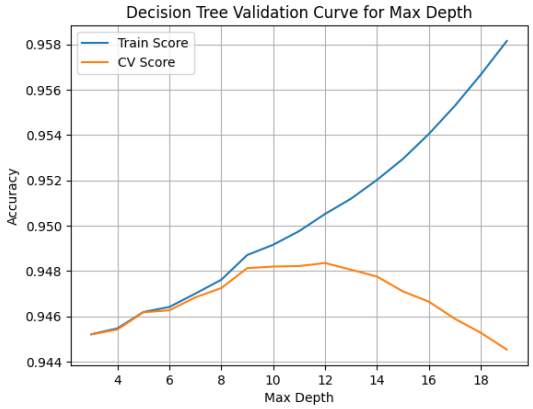

Figure 1. Covid death Decision Tree Validation Curve for Max Depth

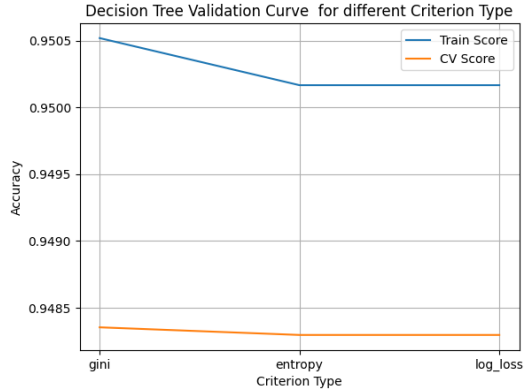

Figure 2. Covid death Decision Tree Validation Curve for different Criterion Type

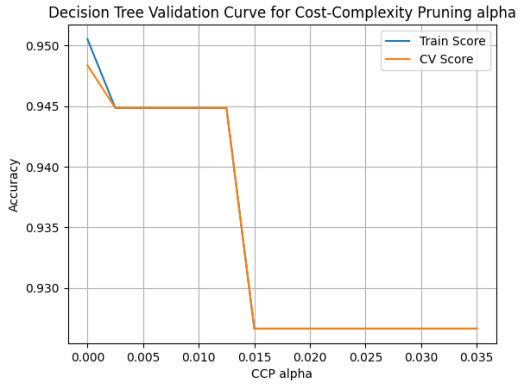

Figure 3. Covid death Decision Tree Validation Curve for Cost-Complexity Pruning alpha

3.3.2 Neural Network

Deep Learning of Neural Network is the most popular Machine Learning technique in the last 9 years. The Neural network used in this analysis is only shallow Neural Network learning with only two layers to show case the power of it.
First finding the neuron numbers in each layer. There are lots of combinations we can try, but choose 5 to 50 with step 5 for both layers to show case. Result shows 25 neurons on both layers works the best. As shown in Figure 4.

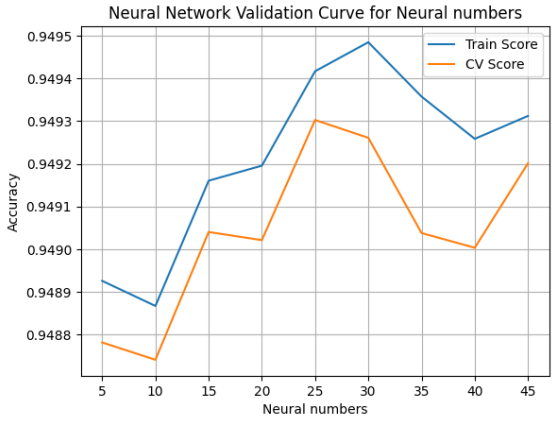

Figure 4. Covid death Neural Network Validation Curve for neuron numbers

Then activation functions of identity, logistic (sigmoid), tanh and relu are tested to find out the best activation functions. As shown in Figure 5. I was expecting sigmoid would perform better for low number of layers, but turns out relu still works better in low number of layers. The lower loss decay of relu during loss back propagation definitely helps on training better model.

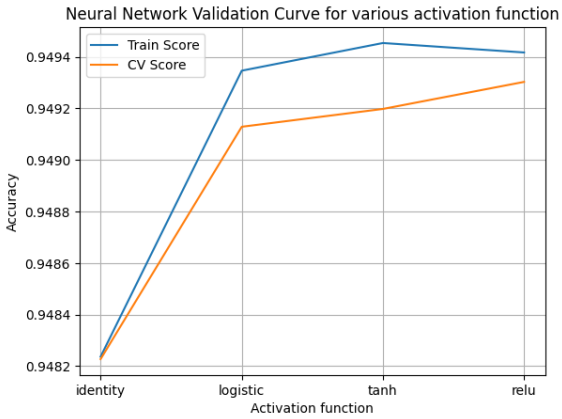

Figure 5. Covid death Neural Network Validation Curve for various activation function

Then learning function of lbfgs, sgd (stochastic gradient descent) and adam are compared. Result shows adam works best in this use case. As shown in Figure 6. There is no surprise here, adam is the well-known default first choice for normal Neural Network training.

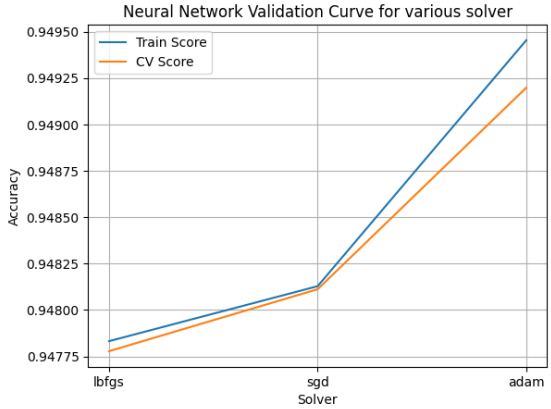

Figure 6. Covid death Neural Network Validation Curve for various solver

Eventually, the best Neural Network model trained by hidden layer size (25, 25), relu activation function and adam solver achieved 94.86% accuracy on test set.

3.3.3 Boosting

Boosting is dominant in general regression and classification problems. Common popular ones are xgboost and catboost. To investigate the foundations of boosting, AdaBoost is chosen here. I use basic Decision Tree as base estimator for AdaBoost. Since in the previous analysis, we find out best Decision Tree settings for this case, max_depth of 12, gini criterion and 0 pruning are used here.

First test number of estimators. Result shows 5 estimators works the best. As shown in Figure 7. Surprisingly, lower estimator number leads to higher validation accuracy. I think this is because I chose high base estimator settings that a few of estimators already working very well. If chose lower base estimator settings, like max_depth=5, result will be different.

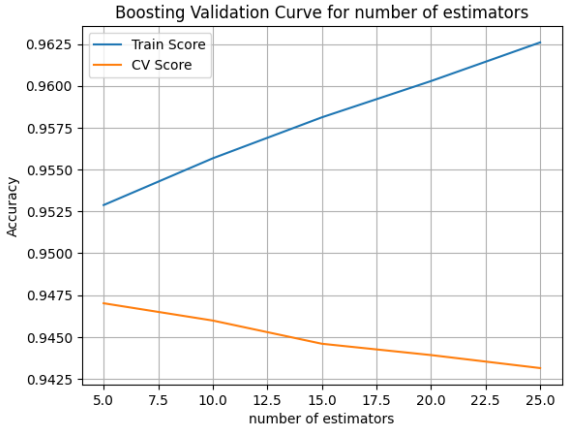

Figure 7. Covid death Boosting Validation Curve for number of estimators in AdaBoost

Then best learning rate from $1e^{-4}$ to $1e^{2}$ is tested. Result shows $1e^{-4}$, $1e^{-3}$, $1e^{-2}$ all works well. As shown in Figure 8. For common cases, $1e^{-3}$ is a safe learning rate. Therefore $1e^{-3}$ is chosen here. Notably, any learning rate more than 1 will lead to dramatically accuracy drop.

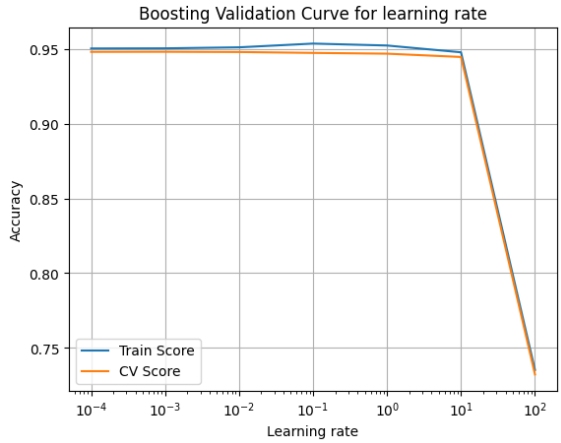

Figure 8. Covid death Boosting Validation Curve for learning rate in AdaBoost

Eventually, the best Boosting model with Decision Tree of 12 max_depth, gini criterion and 0 pruning, and number of estimators as 5, learning rate of 0.001 is trained to achieve 94.83% accuracy on test set.

3.3.4 SVM

SVM was very popular in first decade of 2000.

Due to there are 1 million records in dataset, while SVM is usually extremely slow in large dataset. Due to the complexity is O(number_samples^2 * number_features). Therefore, I used sampled data to find best training parameters. Otherwise, even after using sklearnex, which is 100 times faster than normal sklearn, it still take over 12 hours to do training with cross-validation. Therefore, I chose to sample 1% of training data (8k out of 839k) to find best parameter for SVM.

First, best C from 1e-3 to 1e2 is tested. Result shows C=100 performs the best. As shown in Figure 9.

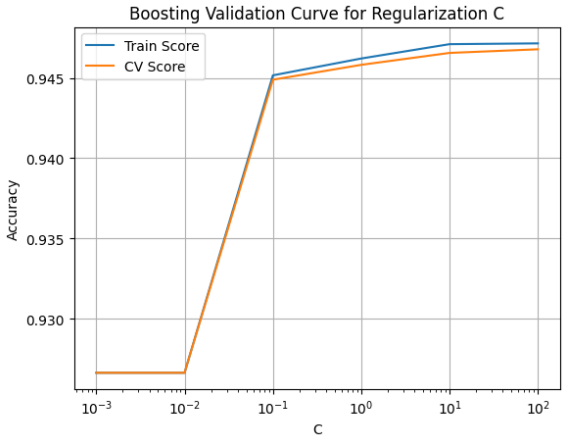

Figure 9. Covid death Boosting Validation Curve for C in SVM

Then test best kernel of linear, polynomial, rbf, sigmoid. The result shows rbf works best. As shown in Figure 10.

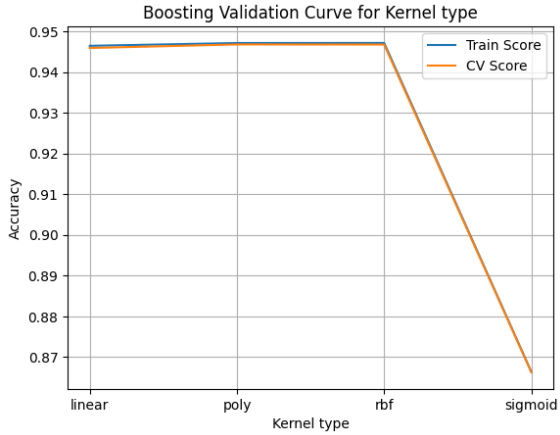

Figure 10. Covid death Boosting Validation Curve for kernels in SVM

Eventually the best SVM model using C=100 and rbf kernel achieved 94.70% accuracy on test set.

3.3.5 KNN

KNN is a basic Machine Learning algorithm. Due to KNN is also very slow in large data.
I choose to use sampled data to find best parameters. However, it's still way too slow.
Firstly, I tested KNN with neighbor setting range in 5-25, results show neighbors=20 got best accuracy. As shown in Figure 11.
This probably means, due to people all have various disease and symptom, and similar symptom with various background might have different result. Therefore, picking more neighbors would be better for better prediction result.

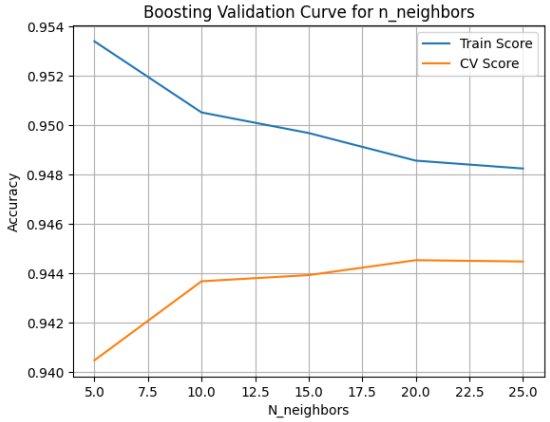

Figure 11. Covid death Boosting Validation Curve for neighbors setting in KNN

Secondly, Best algorithms of 'ball_tree', 'kd_tree' and 'brute' are tested. Result shows both 'ball_tree' and 'brute' works well. As shown in Figure 12.

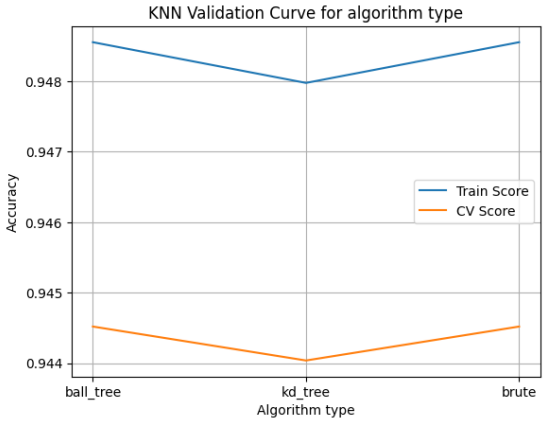

Figure 12. Covid death Boosting Validation Curve for best algorithms in KNN

Eventually, I choose number of neighbors=20 and 'ball_tree' algorithm for KNN, which achieved 94.46% accuracy on test set.

3.4 Model accuracy and run time comparison

The accuracy and training time complexity comparison is shown in Table 1. As shown, Neural Network gets the highest accuracy of 94.86%. But Decision Tree and Adaboost is only slightly lower, as of 94.83%. The difference is insignificant in this COVID Death case. Comparing the training time complexity, for this Covid Death classification use case, Decision Tree is already good enough for training an accurate model. While SVM and KNN both got lower accuracy and training time is extremely slow for this 800k training data. They are not recommended in this Covid Death case.

Table 1. Model Accuracy and Time Complexity comparison

Model Name	**Accuracy**	**Training Time Complexity**
Decision Tree	94.83%	O(number_samples * log(number_samples) * number_features)
Neural Network	94.86%	O(number_layers * number_neurons)
Adaboost	94.83%	O(number_estimator * number_samples * log(number_samples) * number_features)
SVM	94.70%	O(number_samples^2 * number_features)
KNN	94.46%	O(number_samples * number_features + k * number_samples)

3.5 Feature importance and conclusion

The top 10 most important feature generated by Decision Tree is shown in Figure 13. As it shows, "patient_type_hospitalized" impact death rate the most. People being intubated is also shown the serious of condition. Age is the third important, same as what we learned that death rate of elders is much higher. Pneumonia cause lung issue is also significantly impact death. Surprisingly, the test result as of Classification_Final and Medical facilities as of Medical_Unit that caused different treatment/in different facility significantly leads to different death rate. This shows with better treatment and better facility, it leads to much fewer death. It's shocking that lots of people are dead due to bad treatment. We heard a lot of it, but when we see the number it's still difficult to receive it.

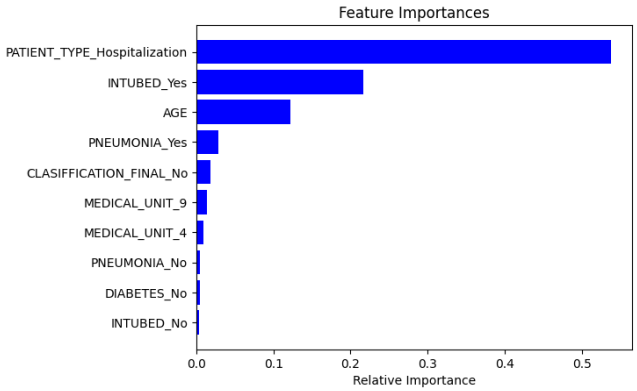

Figure 13. Covid death top 10 important features

4. Credit score classification analysis

4.1 Data processing and feature engineering

There are 100k records and 24 useful columns in the dataset. The columns used are ID, Customer_ID, Month, Name, Age, SSN, Occupation, Annual_Income, Monthly_Inhand_Salary, Num_Bank_Accounts, Num_Credit_Card, Interest_Rate, Num_of_Loan, Type_of_Loan, Delay_from_due_date, Num_of_Delayed_Payment, Changed_Credit_Limit, Num_Credit_Inquiries, Credit_Mix, Outstanding_Debt, Credit_Utilization_Ratio, Credit_History_Age, Payment_of_Min_Amount, Total_EMI_per_month, Amount_invested_monthly, Payment_Behaviour, Monthly_Balance, Credit_Score.

Most of the columns are numerical data. For numerical features, normalization might be needed for non-tree-based Machine Learning algorithms. Credit_Score column is converted to target column with label encoder.

After feature engineering, the final features number increase from 24 to 61.

4.2 Train test data split

The Train test data split ratio is 80:20. After splitting, train data has 80,000 records, while test data has 20,000 records.

4.3 Train model

4.3.1 Decision Tree

The first tuning parameter is max_depth of Decision Tree. After testing max depth from 2 to 20 with 2 folds cross validation, result shows max_depth of 16 is the best. As shown in Figure 14. The surprising part is, the best max_depth is 16. For lots of other use cases, the normal tree depth I experienced before is only with 6-8. This shows, since there are so many features and there so many root-cause and so many complicated situations, more tree depth is needed to learn this use case better and getter better decision.

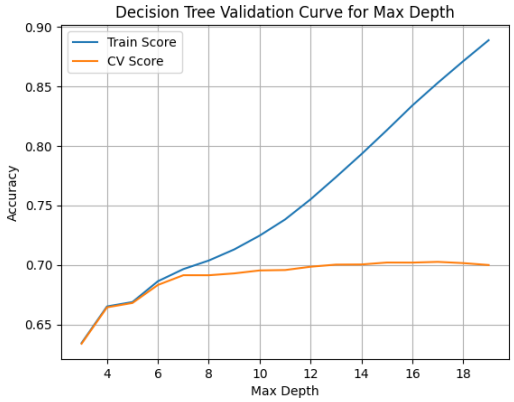

Figure 14. Credit Score Decision Tree Validation Curve for Max Depth

Then criterion of gini, entropy and log_loss are tested. Eventually find out gini criterion is the best for this use case. As shown in Figure 15.

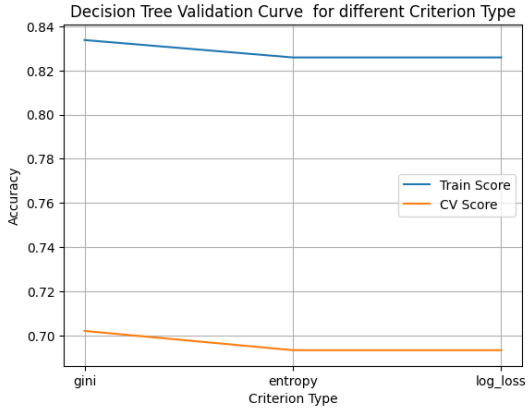

Figure 15. Credit Score Decision Tree Validation Curve for different Criterion Type

After that, best alpha for Minimal Cost-Complexity Pruning from 0 to 0.035 step by 0.0025 is tested to find out best pruning alpha. Surprisingly as well, the best alpha is 0.000. As shown in Figure 16. May be the reason is there are lots of reason for credit level

and each feature count a lot. Therefore, more levels of branching is needed to represent the complicated cases. It makes pruning leads to lower model accuracy.

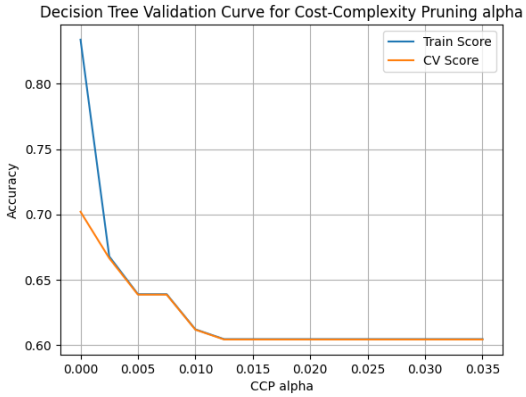

Figure 16. Credit Score Decision Tree Validation Curve for Cost-Complexity Pruning alpha

The best Decision Tree model with max_depth of 16, gini criterion and 0 pruning leads to 71.21% accuracy on test dataset

4.3.2 Neural Network

First finding the neuron numbers in each layer. There are lots of combinations we can try, but choose 10 to 90 with step 10 for both layers to show case. Result shows 50 neurons on both layers works the best. As shown in Figure 17.

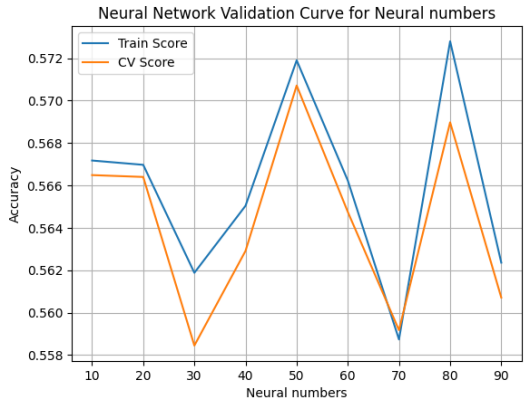

Figure 17. Credit Score Neural Network Validation Curve for neuron numbers

Then activation functions of identity, logistic (sigmoid), tanh and relu are tested to find out the best activation functions. As shown in Figure 18. I was expecting logistic (sigmoid) would perform better for low number of layers, and it is. This result is contradiction to COVID Death Classification. Seems relu might not be as good as logistic (sigmoid) activation function in low layer case. Or it could be because no normalization is done for numerical features.

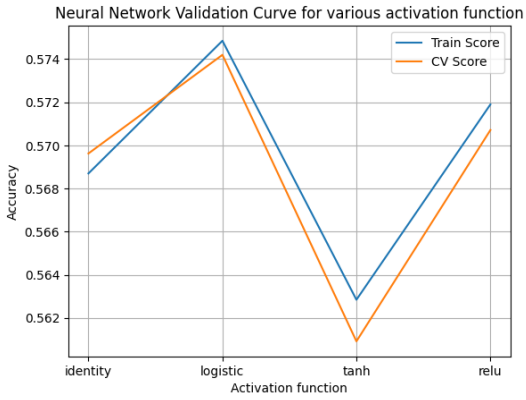

Figure 18. Credit Score Neural Network Validation Curve for various activation function

Then best learning function of lbfgs, sgd (stochastic gradient descent) and adam are compared. Result shows adam works best in this use case. As shown in Figure 19. There is no surprising here, adam is the well-known default first choice for normal Neural Network training.

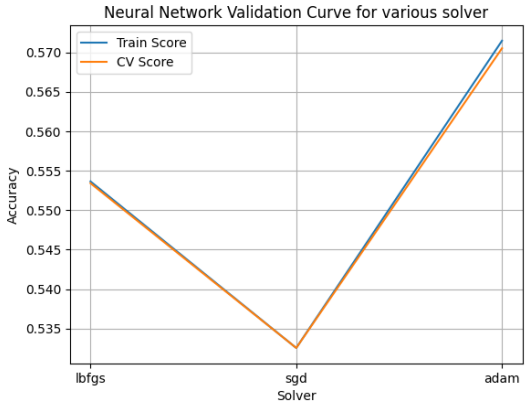

Figure 19. Credit Score Neural Network Validation Curve for various solver

The best Neural Network model trained by hidden layer size (50, 50), logistic (sigmoid) activation function and adam solver achieved 57.60% accuracy on test set.

4.3.3 Boosting

Boosting is dominant in general regression and classification problems. Common popular ones are xgboost and catboost. To investigate the foundations, AdaBoost is chosen here.

The AdaBoost is using basic Decision Tree as base estimator. Since in the previous analysis, we find out best Decision Tree settings for this case, max_depth of 16, gini criterion and 0 pruning are used here.

First test number of estimators range from 20 to 200 with step 10. Result shows 150 estimators works the best. As shown in Figure 20. I think this is because I chose high base estimator settings that few estimators already working very well. If chose lower base estimator settings, like max_depth=6, result will be different.

Then best learning rate from 1e-4 to 1e2 is tested. Result shows 1e-1 works best. As shown in Figure 21.

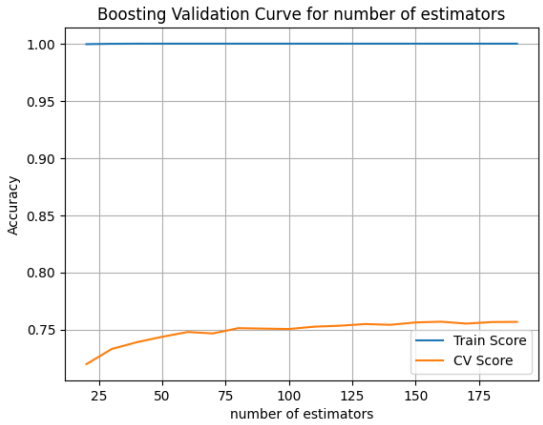

Figure 20. Credit Score Boosting Validation Curve for number of estimators

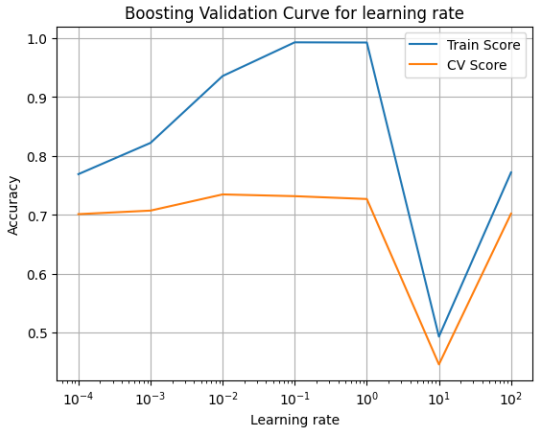

Figure 21. Credit Score Boosting Validation Curve for learning rate

Eventually, the best Boosting model with Decision Tree of 16 max_depth, gini criterion and 0 pruning, and number of estimators as 150, learning rate of 0.1 is trained to achieve 78.375% accuracy on test set.

4.3.4 SVM

Due to there are 100,000 records in dataset, while SVM is usually extremely slow in large dataset. Due to the complexity is O(number_samples^2 * number_features). Therefore, I used sample data to find best training parameters. Otherwise, even after using sklearnex, which is 100 times faster than normal sklearn, it still takes over 12 hours to do cross validation training. Therefore, I chose to sample 10% of training data (8k) to find best parameter for SVM.
First, test best C from $1e^{-3}$ to $1e^{2}$. Result shows C=0.1 performs the best. As shown in Figure 22.

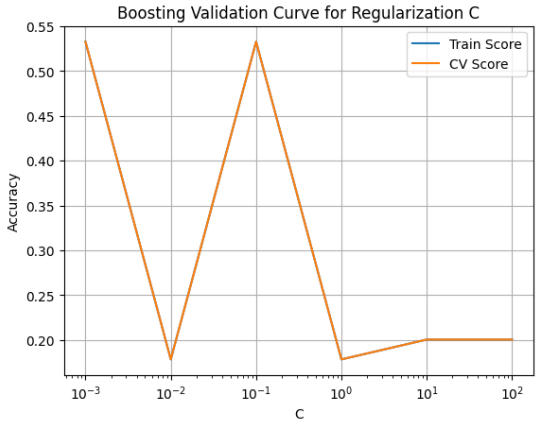

Figure 22. Credit Score Boosting Validation Curve for learning rate in SVM

Then test best kernel of linear, polynomial, rbf, sigmoid. The result shows rbf works best. As shown in Figure 23.

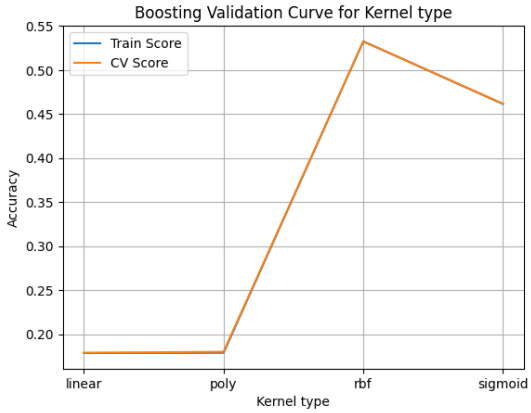

Figure 23. Credit Score Boosting Validation Curve for algorithm in SVM

Eventually the best SVM model using C=0.1 and rbf kernel achieved 52.86% accuracy on test set.

4.3.5 KNN

KNN is a basic Machine Learning algorithm. Due to KNN is also very slow in large data.
I choose to use sampled data of 10% to find best parameters.
First choose number of neighbors from 2 to 30 with step 2. Result shows 4 is the best. As shown in Figure 24.

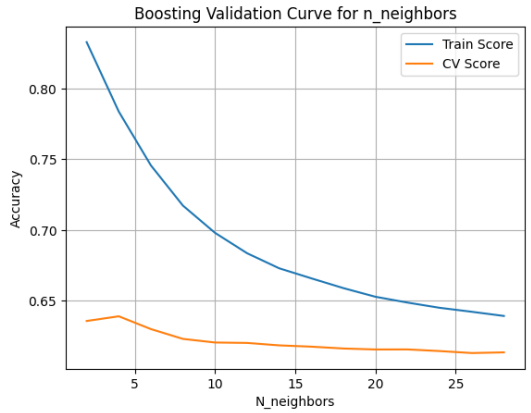

Figure 24. Credit Score Boosting Validation Curve for learning rate

Then Choose best algorithm from ball_tree, kd_tree, brute. Result shows all works the same. As shown in Figure 25. As common choice, kd_tree is selected here.

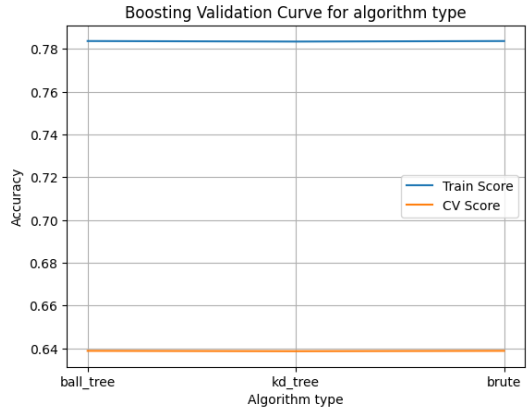

Figure 25. Credit Score Boosting Validation Curve for learning rate

Eventually the best KNN model using number of neighbors=4 and ball_tree algorithm achieved 66.52% accuracy on test set.

4.4 Model accuracy and run time comparison

The accuracy and training time complexity comparison for Credit Score case is shown in Table 2. As shown, Adaboost get the highest accuracy of 78.38%. Following it, Decision Tree got accuracy of 71.21%. Neural Network, SVM and KNN are not working well in this case. This might be due to numerical features are not normalized at the feature engineering phase. This shows Tree base algorithms works good on non-normalized data. Besides, KNN is still better than Neural Network and SVM, I think this is due to the data is not balanced, both Neural Network and SVM are more impacted by inbalanced data. Compare about the training time complexity, for this Credit Score prediction case, Decision Tree is good to train a baseline or fast model and use Adaboost or other Boosting model like XGBoost to train best model.

Table 2. Credit Score Model Accuracy

Model Name	**Accuracy**
Decision Tree	71.21%
Neural Network	57.60%
Adaboost	78.38%
SVM	52.86%
KNN	66.52%

4.5 Feature importance and conclusion

The top 15 most important feature generated by Decision Tree is shown in Figure 26. As it shows, Outstanding Debt impact credit score the most. Interest Rate is shown as second importance, I this it's due to high interest rate will cool down the economy, which leads to more loans for life expenses. Payment of minimal amount is the third important, shows if people is capable of paying back, indicating healthier financial statues. Surprisingly Changed credit limit is fourth important, could be due to credit limit get higher mean financially getting better, vice versa. Or maybe people requested for higher credit line might means they starts to have financial problem that needs to rely on credit card more.
Delay from due date, Number of Credit Card, Outstanding Debt, Annual income, Age, Monthly in hand Salary and Number of Inquiries are all obvious. The best features model learned are aligned with our common sense.

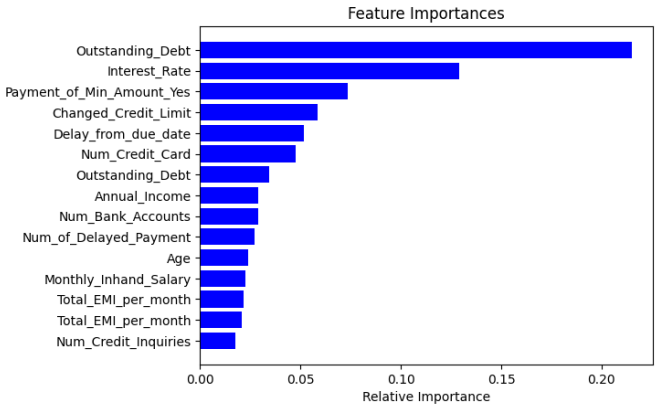

Figure 26. Credit Score top 15 important features

5. Summary

In analysis report, 5 basic machine learning algorithms are applied and optimized to find best model for two use cases, COVID Death Classification and Credit Score classification. It's fun to build the model and try different optimization. However, writing report is not.