Evaluating the Impact of Dataset Sizes, Feature Dimensionality and Model Complexity in Supervised Learning

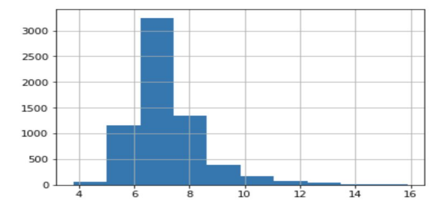
Overview

In this paper two different and disparate datasets, Wine Quality and Wine Reviews were used to train a variety of supervised learning models (decision trees, KNN, Gradient Boosting Machine, Neural Network, and Support Vector Machine) in order to understand how different training set sizes (small vs large) , variable feature types (continuous features vs binary categorical features), feature sizes/dimensions (low feature dimensionality vs high feature dimensionality), and model complexity (low hyper parameter value vs high hyperparameter value) could impact performance for different algorithms across different kinds of datasets. These experiments could then help identify what models were most suited for predictions across what kinds of datasets.

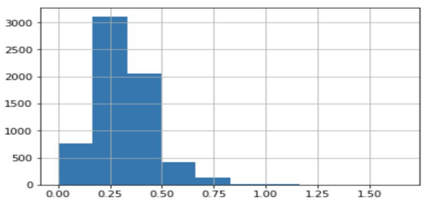
Datasets

The first dataset Wine Quality comes from the UCI Machine Learning Repository (both red and white wine data). The predictor features are all continuous numeric variables about a wine's chemical properties while the target feature was derived from the numeric 'quality' field and converted to a binary target which classifies a review as 'Excellent' (label = 1) if the rated quality in the dataset is greater than or equal to 7 and 'Not Excellent' if is below 7 (label = 0). Distributions of two features Fixed Acidity and Citirc Acid in the dataset are shown below to confirm their continuous numeric nature. A total of approximately 5K rows were part of this dataset.

Fixed Acidity Histogram



Citric Acid Histogram



The second dataset Wine Reviews was downloaded from Kaggle. The dataset has 130K reviews and text features about different types of Wine. For the purposes of this assignment only 3 data columns were used for feature derivation purposes: the wine review (description), wine variety, and the country (from which the wine came). The predicted feature in this case was whether a review about a wine was rated 'Excellent' and 'Not Excellent' using the 'Points' training label attributed to that review. If 'Points' was above a certain threshold (92 for this analysis) then it was classified as 'Excellent' (label = 1) and if not then it was classified as 'Not Excellent' (label = 0). Because the review 'description' column is all text data, some data preprocessing was done: Stop and punctuation words were removed, review words were stemmed. Then for all reviews only the top 150 occuring words were chosen as data features to be used for supervised learning. These words were converted into a bag of words 150 column array where each word was an individual feature such that if that word occurred in the review then that word had a value of 1 for that particular dataset row (wine review) and if not then that word had a value of 0. The other 2 categorical features 'variety' and 'country' were one hot encoded. Shown below are the first 5 rows of the raw wine reviews raw data. Note: only a sample of 5K reviews were considered for this analysis.

Reasons for Choosing These Datasets

It should be emphasized that even though both datasets are classification problems predicting binary 'wine quality' they are entirely different datasets and problems where the reviews don't necessarily correspond to any wine in the first dataset Wine Quality. The reason for choosing these datasets was primarily so a variety of questions could be answered around essentially "**how do the attributes of the data impact supervised learning performance?**". Both datasets had useful and varied properties which could be used to answer this. Specifically:

1.Does the number of features in the dataset impact model performance? The first data set Wine Quality is small dimensionally speaking (around 10 features). The Wine Reviews data however has almost 200 text based features. Choosing two datasets with vastly different feature dimensionality can help answer if certain algorithms perform better on low dimensional data vs higher feature dimensional data?

2. Does the type of features or the type of data used matter for learning? For example, do certain algorithms perform better when there is only continuous numeric data (Wine Quality dataset) and perform worse when there is only NLP/Text/categorical Binary Data (Wine Reviews). The idea is based on theoretical observations where non parametric methods like Decision Trees, KNN may perform better on binary data because they aren't constrained to "fitting a line" that might be best for continuous features.

3. Does training data size impact model performance in and out of sample? And is the impact of training size more pronounced when the feature set is high dimensional vs low dimensional for certain algorithms?

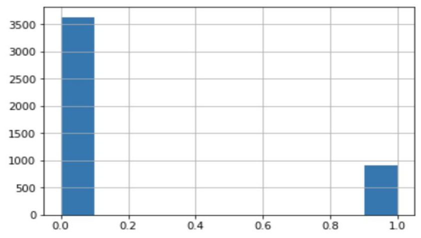
4. Does model complexity (hyper parameter values) usually impact model performance the same way given the type of data? For example, how is model complexity impacting performance when data is low dimensional vs high dimensional?

Model Analysis - Framework

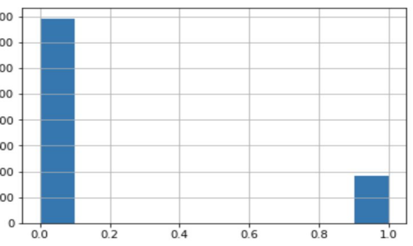
Train vs Test Splitting

The 5 models used in this analysis for comparison are Decision Trees, SVM, KNN, Neural Networks and Gradient Boosting Machines. Prior to training each dataset was split into a train and test set randomly such that the test set comprised 30% of the data and the train set comprised 70% of the data. The distribution of the target variable (Wine Quality in each case) was checked to see if it was similar in both the train and test set to rule out sampling bias. Plotted below is this distribution for the first dataset, where we can see that both train and test have the same distribution of positive and negative classes with approximately 78% classified as negative (Not Excellent). The models were all trained on train data and only in the last step to see which final model performed best on which dataset the test was used.

Train Labels Distribution - Wine Quality Data



Test Labels Distribution - Wine Quality Data



Evaluation Metric: For this analysis F1 Score was used as the primary evaluation metric for measuring model performance. The reason for choosing F1 Score is first this is a binary classification problem and second because in both datasets the classes are generally imbalanced (around 78-85% negative). A raw measure of accuracy would bias towards the negative class leading to very misleading high accuracies but very poor precision or recall.

Stratified K Fold Cross Validation: For actual tuning of the models and for all the experiments only train data was used. To measure model generalization performance on train data, Stratified 5 fold cross validation was used such that 4 equal random samples (folds) of train data were drawn from 80% of the train data for training while the remaining 20% of the train data was held out for validation with this being repeated randomly 5 times each. K-Fold Cross validation has the benefit of providing better estimates of out of sample model performance than a simple Train/Validation split as data is sampled repeatedly reducing some testing variance.

Feature Scaling: After train and test splitting for the Wine Quality dataset feature scaling was done so that all the continuous variables were scaled by removing the mean and scaling to unit variance. This is important because some models like neural networks and SVMs are sensitive to the magnitude of features and can penalize small valued features erroneously if not on the same scale. For Wine Reviews Dataset however no feature scaling was needed as all features are based on textual words and were binary features only.

Baseline Performance: Prior to analysis a baseline performance was gathered so we could see how well the model performs over random noise (so we can infer if any learning is even happening in the first place) and serve as a lower limit benchmark. The baseline here is such that we predict the positive class based on the probability of that class occuring. So for example since in Wine Quality 78% of data points are of the negative class we predict 0 with probability 78% of the time for all records, otherwise 1. This leads to a baseline performance of approximately 71% Accuracy, 19% Precision, 13% Recall and 15% F1 Score. For the Wine Reviews dataset, the same methodology was used and the baseline was 70% Accuracy, 17% Precision, 20% Recall, 18% F1 Score.

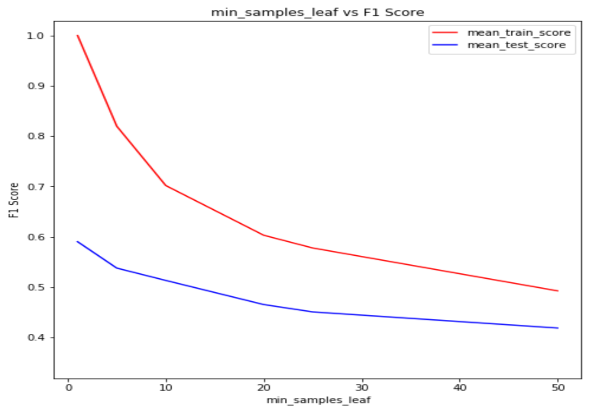
Experiment 1 - Model Complexity Analysis

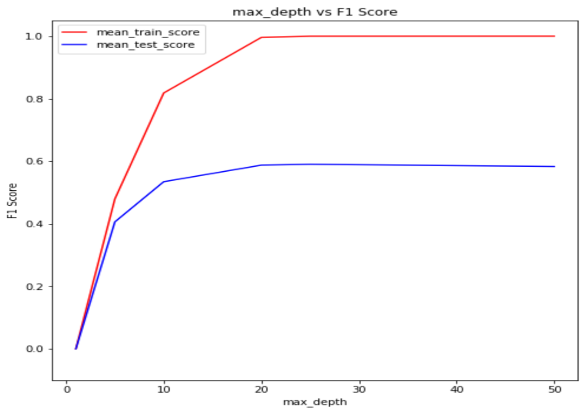
Before testing, each model was trained by increasing/decreasing hyperparameter values for each dataset to see 1) how changing the hyperparameter value or model complexity impacted a model's performance both on the training set as well as the cross validation set.

Decision Trees: For Decision Trees the following parameters were tweaked to ascertain the impact of pruning. Min Samples in Leaf Node which indicates the smallest number of samples in a node before it is considered a leaf node for decision voting/averaging purposes. The tree stops growing beyond that min sample. Similarly, max depth was also used to see the impact of the maximum number of levels a tree could have. A larger depth and/or smaller min leaf size indicates less pruning and greater model complexity and vice versa.

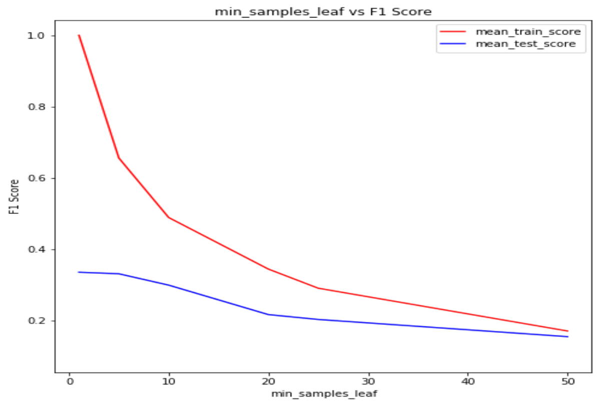
The charts show that reducing the minimum leaf size to 1 doesn't lead to overfitting as the CV score doesn't necessarily drop when going from a min sample leaf size of 5 to 1. However, as the min leaf size increases then the CV score and Training score both continue to decrease suggesting underfitting the less the model complexity. This makes sense since very large leafs are shallower by extension and construct simpler rules which might not capture the target hypothesis as well. The max depth chart shows that model performance improves at first for trees with increasing depth but then plateaus for the CV data while continuing to improve for a little more depth on the training set until eventually plateauing as well. After that increasing tree depth has little impact for either dataset. Again this points to the fact that for either of these datasets greater model complexity and less pruning leads to better performance and less underfitting without necessarily overfitting.

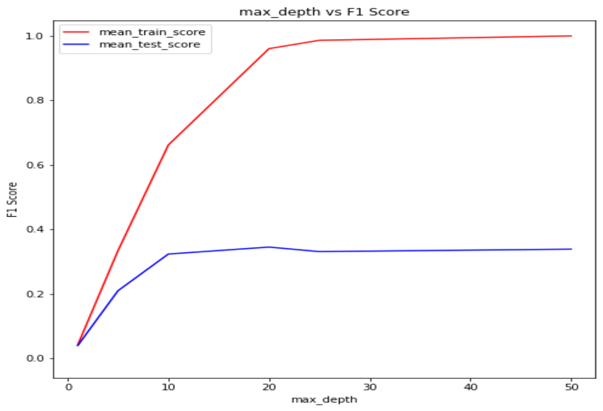
Wine Quality Data Decision Tree Model Complexity (Left: Min Samples vs F1 Score, Right: Max Depth vs F1 Score)





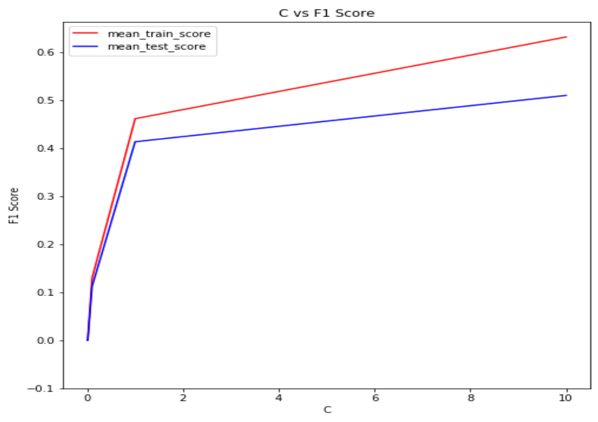
Wine Reviews Data Decision Tree Model Complexity (Left: Min Samples vs F1 Score, Right: Max Depth vs F1 Score)

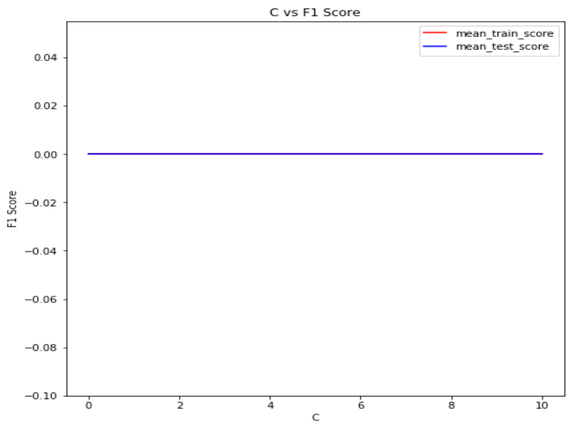




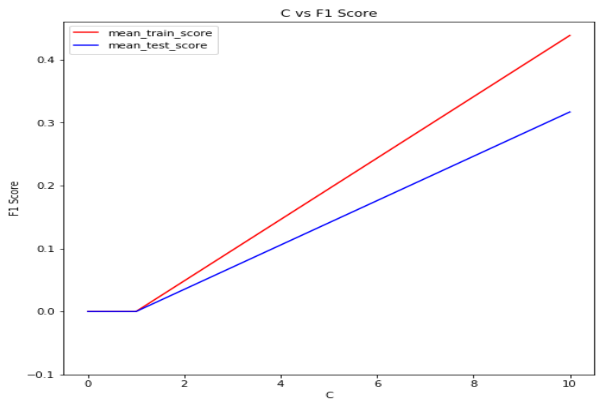
Support Vector Machines: For SVMs, two kernels were tried (linear and Radial basis function). In both datasets using the RBF kernel, larger values of C (the misclassification penalty) tend to improve the model F1 Score. But the Linear kernel fails on the first dataset which suggests this data is not appropriate for a linear kernel separator. Higher values of C imply that both our datasets are able to fit the data better as when there is a larger penalty for incorrectly misclassified points it encourages the SVM to make less mistakes.

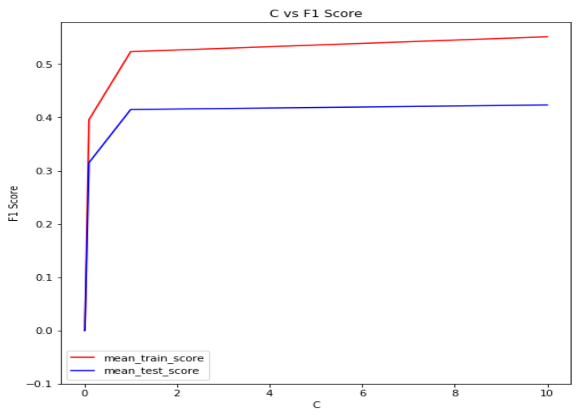
Wine Quality Data SVM Model Complexity for different values of C with different Kernels (Left: RBF Kernel, Right: Linear Kernel)





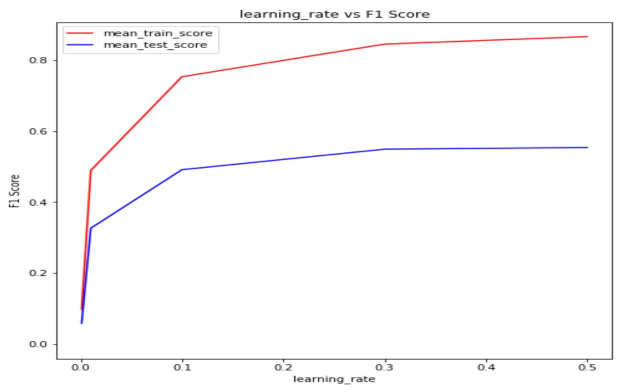
Wine Review Data SVM Model Complexity for different values of C with different Kernels (Left: RBF Kernel, Right: Linear Kernel)

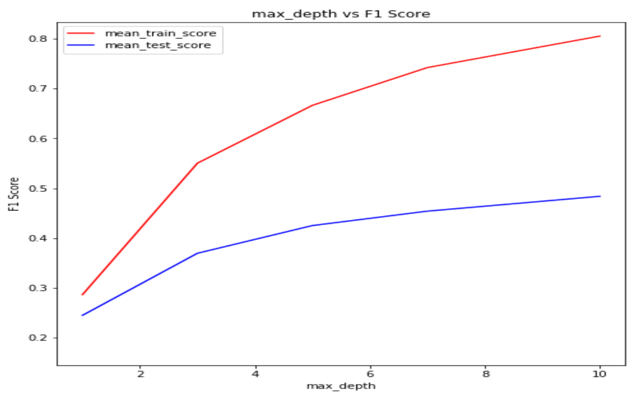




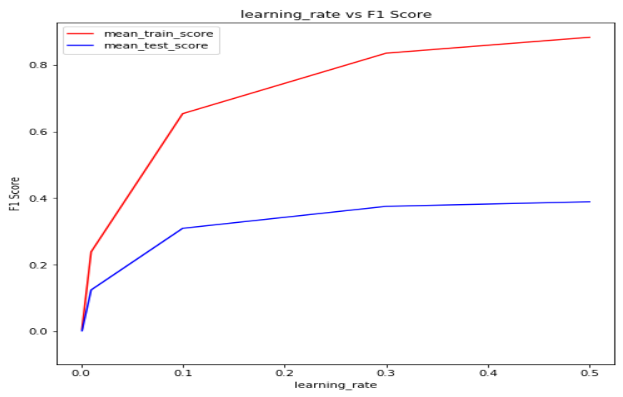
GBM: For GBM both shrinkage (learning rate) values and max depth tend to improve the cross validation score if their values are larger however the gains tend to plateau for values greater than 0.2 for learning rate and for tree max depth > 5. These parameters were chosen because the max depth tells us how complex each weak learner for the GBM would be and the learning rate impacts how much to boost the errors we make for subsequent trees. Larger values of max depth for each tree would result in a more complex learner.

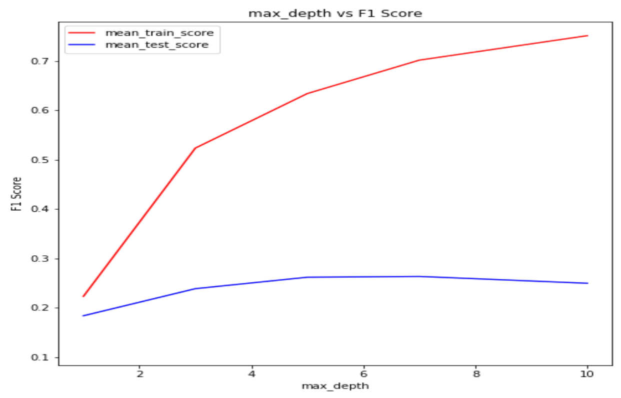
Wine Quality Data GBM Model Complexity (Left: Learning Rate (Shrinkage) , Right: Max Depth)





Wine Quality Data GBM Model Complexity (Left: Learning Rate (Shrinkage) , Right: Max Depth)



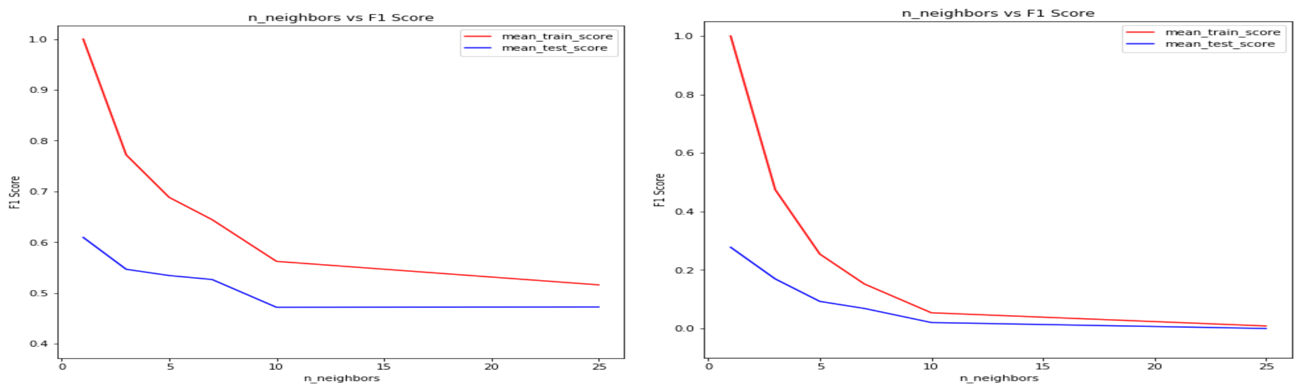


The charts above suggest that both parameters underfit for small values. For learning rate, it makes sense because we are updating the weights assigned to the misclassified and classified points at smaller rates at each tree at time t and so will need more iterations (trees) to compensate for the smaller weight updates (iterations are being held constant for these charts). For Max Depth, in the second dataset there is actually slight overfitting at max_depth >= 7. This makes sense because Boosting works best for weak learners with high bias as long as the weak learner performs better than chance. By building a very deep tree we are making the individual learners much stronger therefore defeating much of the purpose and benefit gained from boosting while introducing the risk of overfitting that comes from having deep trees, and so GBM learners with deep trees perform worse on the second data.

KNN: The KNN algorithm clearly does worse with larger values of K > 1 implying it does best when the model is more complex (K=1) than when it is not (e.g: K=10). It underfits dramatically for large value of K as both the training and CV error increase a lot (F1 Score drops precipitously). Although theoretically such small values of K could lead to overfitting in other datasets because by setting K=1 we can have a classifier very sensitive to noise and outliers.

So why doesn't K = 1 overfit in this data? One hypothesis is that the first dataset of Wine Quality is very small with only 10 features and more important these feature are natural chemical properties which are less susceptible to "noise" unlike let's say human behavior (like clicking an ad). What about the second dataset of Reviews? One hypothesis for the lack of overfitting observed when K=1 in Reviews can be explained by the fact that even though we have 200+ features, most of the word features will form distinct "clusters" of positive or negative words (most positive words will be correlated with each other and therefore clustered together in the same area of the hyperdimensional feature place) so the K-NN model does well finding the single closest data point as it's searching more or less in the same "neighborhood space" of positive or negative words when K =1.

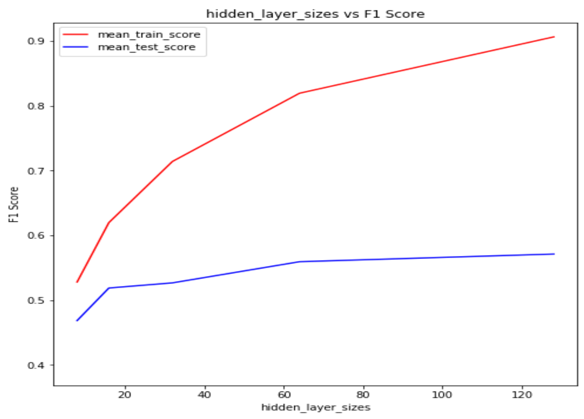
Left: Wine Quality Data KNN Model Performance vs Right: Wine Reviews Data KNN Model Performance with varying values of K

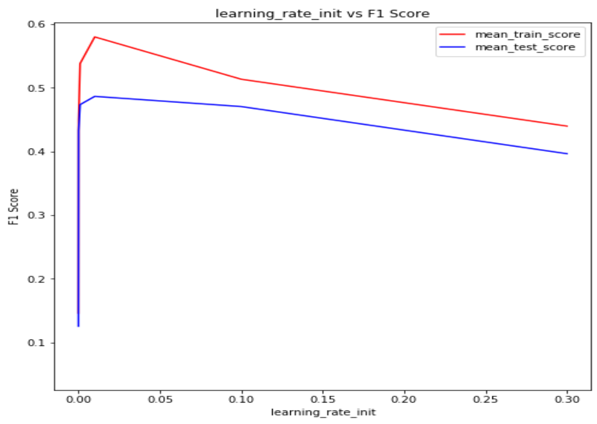


Neural Networks: For neural networks smaller values of learning rate (<0.001) seem to indicate some poorer performance for the Wine Quality dataset as the CV score drops in the beginning. This makes sense because the learning rate controls the rate of weight change and very small learning rates can result in networks getting stuck in suboptimal local optimums. However interestingly for the first dataset large values of learning rates lead to the CV and Train error to drop dramatically. This suggests that larger values instead of getting "stuck" might be entirely skipping the global optimum by incrementing weights too quickly. As a result the neural networks for these datasets are very sensitive to learning rates.

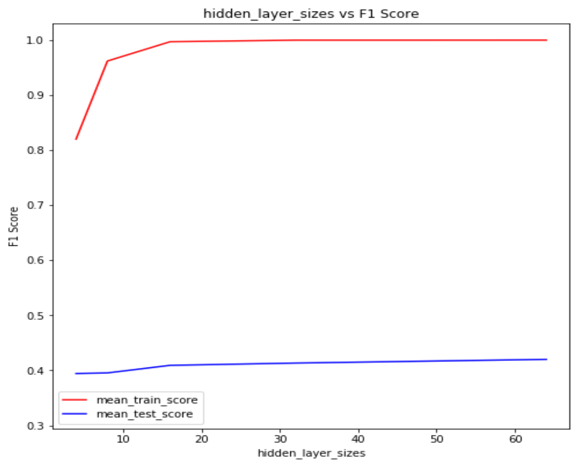
The second parameter which controls model complexity here is the hidden layer size. The results below suggest that the more the nodes in the hidden layer then the better the model performance although after 50 hidden nodes there isn't much improvement. This again makes intuitive sense because more nodes in the hidden layer allow the model to learn more complex decision boundaries so greater model complexity can lead to better performance and less underfitting as observed below. However, both charts for number of hidden nodes (especially the Wine Reviews data) suggest that if we were to continue to increase hidden nodes we will start overfitting and modeling the noise in the dataset because in both charts after 50+ hidden layers the training error is still decreasing (F1 score is increasing) But CV F1 score has stopped increasing.

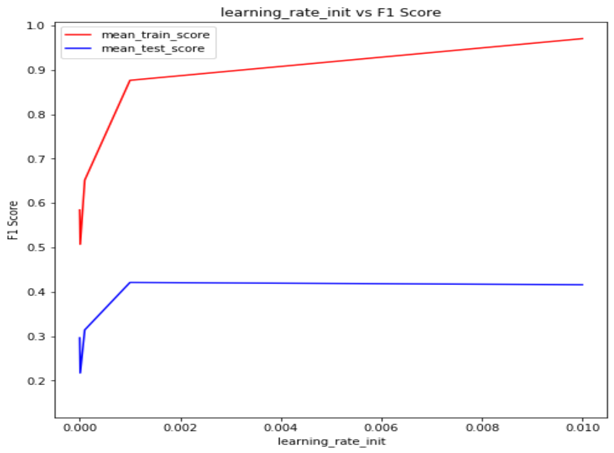
Wine Quality Data NN Model Complexity (Left: Number of Hidden Nodes, Right: Learning Rate)





Wine Reviews Data NN Model Complexity (Left: Number of Hidden Nodes, Right: Learning Rate)

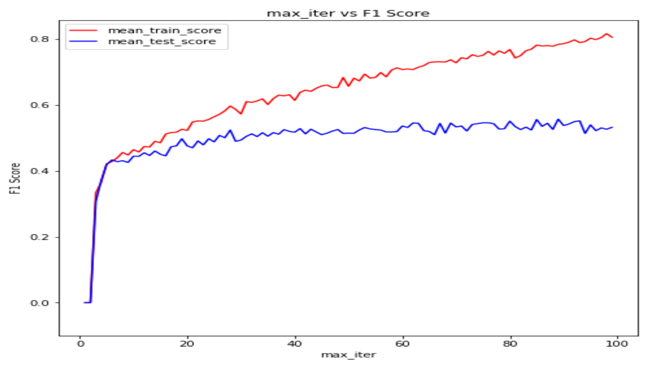


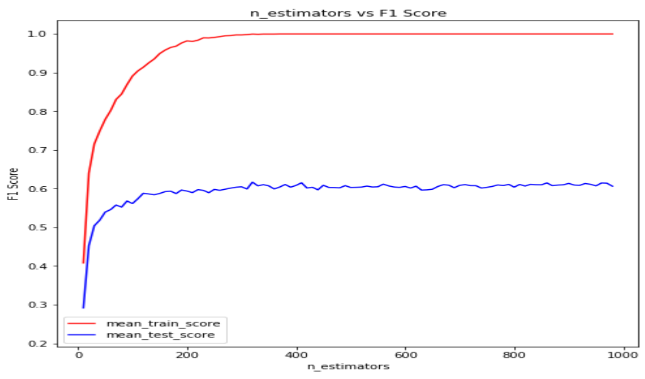


Impact of training iterations(time) on model performance

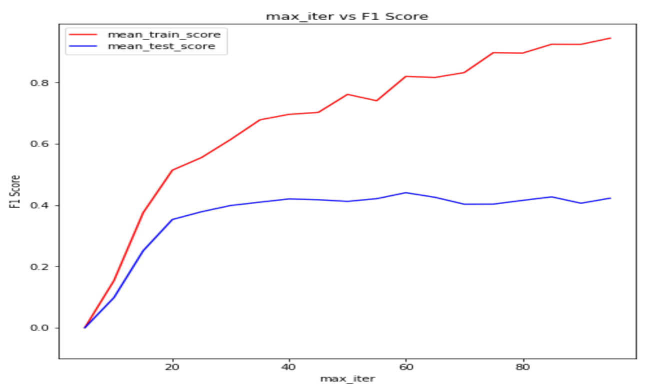
For iterative algorithms like Neural Networks and GBMs, the number of iterations (epochs in the case of NNs, and sequential trees in the case of GBMs) can also impact model performance. The following charts below show that the more the iterations that have elapsed during training then the better the performance of these iterative models but up till a certain point. After a certain number of iterations, it appears that the model has stopped learning and might be also be on its way to overfit as the CV score doesn't increase but the training score keeps increasing (for the first dataset this is 40 iterations for Neural networks and 200 iterations for the GBM).

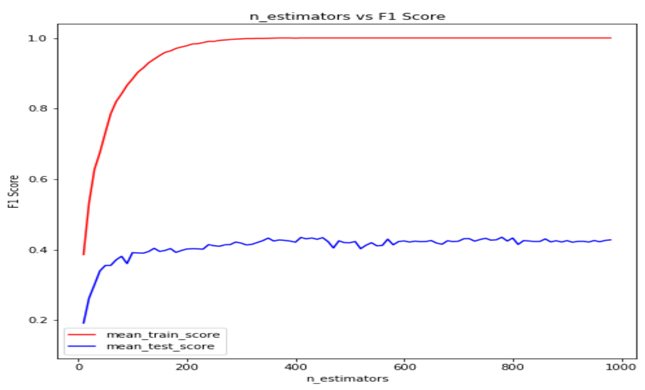
Wine Quality Data: Left Neural Network Performance vs Number of Iterations   Right: GBM Performance vs Number of Iterations





Wine Reviews Data: Left Neural Network Performance vs Number of Iterations   Right: GBM Performance vs Number of Iterations





A final point to note. From the charts above we can see that for both datasets it doesn't seem to be the case that the trends observed when changing hyperparameter values vary that much even if the exact values differ slightly. This would suggest that regardless of the dimensionality of a dataset and the types of features being used (continuous vs binary) the impact of a hyperparameter on a model's performance tends to generally be the same directionally speaking. However, the exact optimal value of the hyperparameter does indeed depend on the dataset. For example, the better kernel for the SVM model is RBF in the case of the Wine Quality data but Linear Kernel in the case of the Wine Reviews Data.

Experiment 1b - HyperParameter Grid Search for best Parameters

The analysis above show the impact of changing only one hyperparameter value at a time. But as seen with GBMs, changing one hyperparameter (learning rate) on its own may not be the most optimal approach unless another value is also changed along with it (number of iterations/trees). For this reason to find the optimal hyperparameters and come up with the best model, a grid search across hyperparameter spaces was done for a few parameters and their values for each model and the CV score was reported for each possible combination of the observed cross validated model on the train set. Summarized are the best models found after grid search for the first dataset. (Additional parameters used included gamma for SVM, Regularization, activation function for NN).

Hyperparameter grid search results for Wine Quality Dataset (Second dataset results now shown but grid search was also done for it)

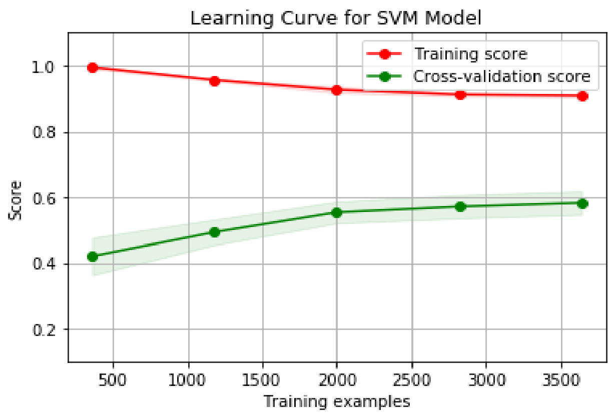
Model	Parameters Considered	Models Built	Best Parameter Values	Best CV F1 Score
Decision Tree	Max Depth, Min Leaf Size	125	Min Leaf Size = 1, Max Depth = 50	0.563
SVM	Gamma, C, Kernel	250	Kernel = rbf, C=10 , Gamma = 0.00001	0.567
GBM	Num Estimators, Max Depth, Learning Rate	625	Num Estimators = 250 , learning rate = 0.1 , depth =5	0.623
KNN	K	25	K = 1	0.609
Neural Network	Regularization Strength, Learning Rate, Activation, Optimizer, Hidden Layer Size,	2500	Hidden Layer Size = 128, regularization strength = 0.00001 , learning rate = 0.01, activation = relu	0.521

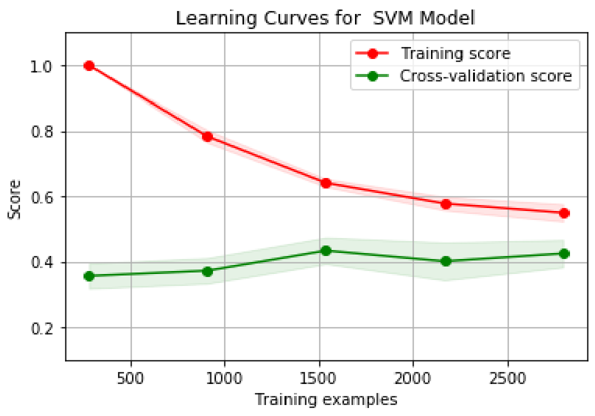
Experiment 2 - Learning Curves Analysis (Impact of Training Size)

Each of the 5 best models from grid search was then trained to better understand how training size affects model performance for both datasets and to see if either model was still suffering from bias or variance. The learning curves are plotted below. The charts on the left are all for the first dataset Wine Quality and the charts on the right are all for the second dataset Wine Reviews.

Wine Quality Data SVM Learning Curve

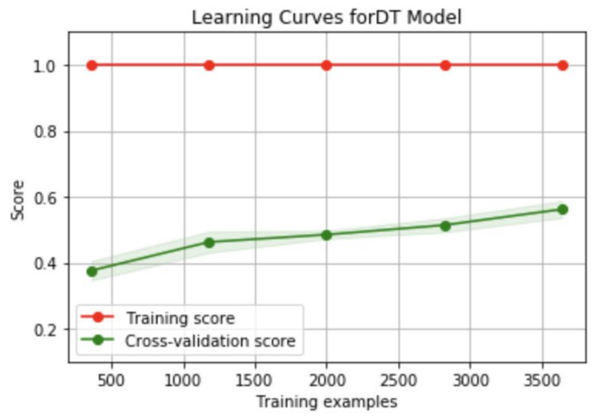
Wine Reviews Data SVM Learning Curve

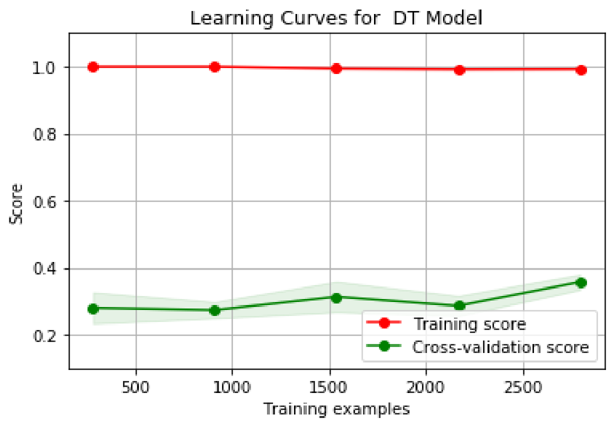




Wine Quality Data Decision Tree Learning Curve

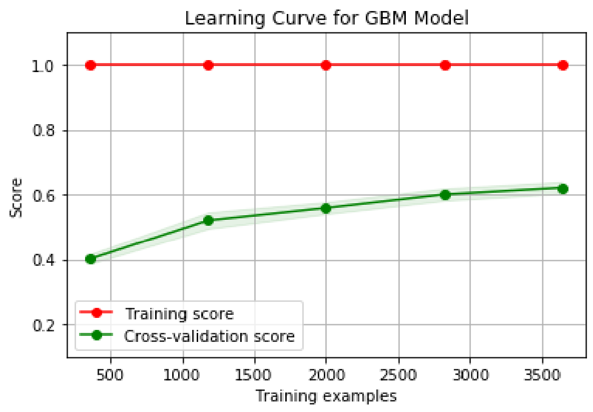
Wine Reviews Data Decision Tree Learning Curve

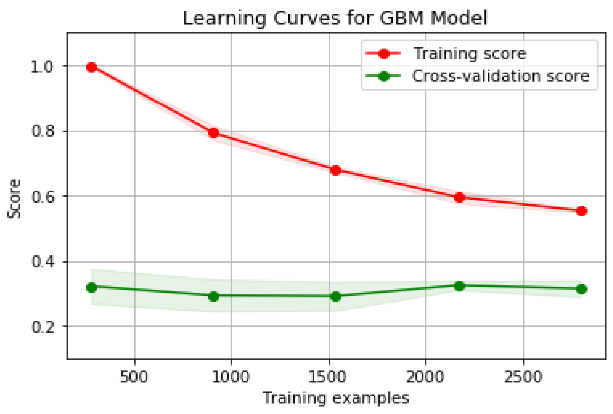




Wine Quality Data Gradient Boosting Learning Curve

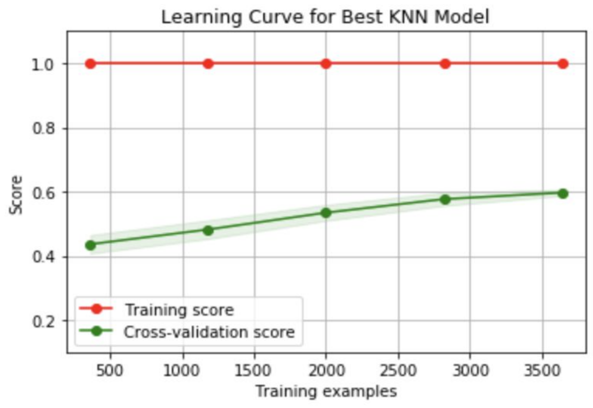
Wine Reviews Data Gradient Boosting Learning Curve

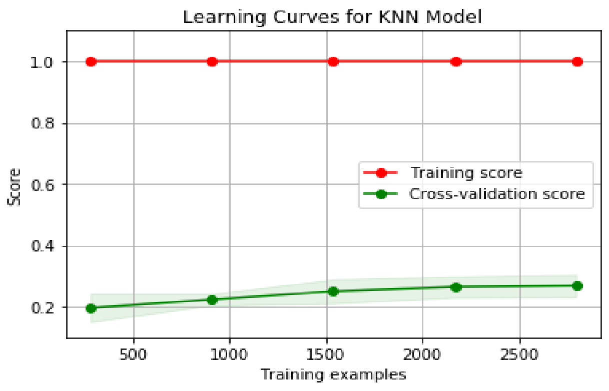




Wine Quality Data KNN Learning Curve

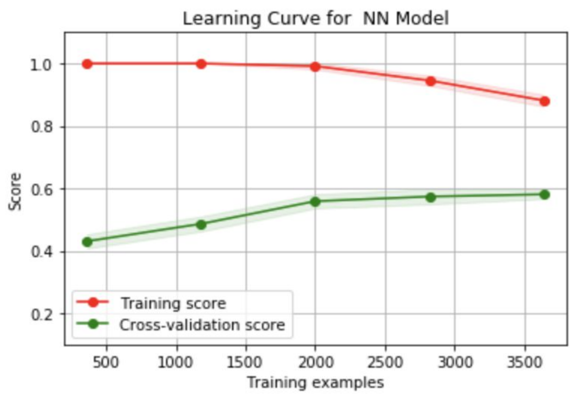
Wine Reviews Data KNN Learning Curve

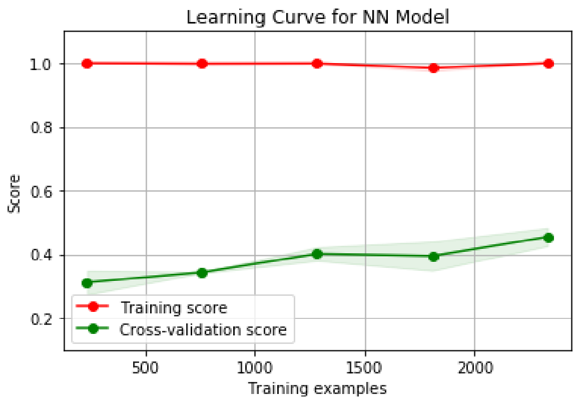




Wine Quality Data Neural Network Learning Curve

Wine Reviews Data NN Learning Curve





From the above curves we can see for the first Wine Quality dataset (left) all models improve their F1 Score with increased training examples on the Cross Validated validation data. The larger the training set therefore the better the model performance out of sample for each model. However, while all models have very low training bias they do have very high variance. For example, the Decision Tree, GBM, and KNN models all have perfect training scores but very low validation scores (example: Decision Tree Training Score = 1 while CV Score = 0.6 with 3.5K samples) The large variance is evident from the huge gap between the Training F1 Score and CV F1 Score. For the Wine Reviews Dataset (right), the conclusion is the same. There is large variance impacting the models as all models have perfect training scores but very low validation scores (except for SVM and GBM which seem to have higher bias this time around as train error decreases and gap narrows between the train and CV scores). This suggests these algorithms are overfitting and more data should be gathered to help improve their generalization. But SVM which has higher bias, a better kernel function could be found to better model the data.

A couple of other interesting observations. First, the models that seem to have the biggest variance (especially in the second data) seem to be Decision Trees and KNN. This makes sense as Decision Trees especially deep ones tend to have complex rules which in the presence of new training data would fall apart as the conditions might not be met due to being be too specific to a training data instance itself. For KNN, training was on K=1 so the only data point considered was the nearest neighbor itself. If this neighbor was indeed the incorrect neighbor on a different test point, then there would be no other data points in the dataset to “average” out and dampen this error.

Second, other than SVM all the other models in the second Wine Reviews dataset have higher variance relative to the first dataset as evident from the fact that the gap between the training and validation curves is much bigger in the second dataset on the right. This can shed some insight into the question earlier: “How does the feature dimensionality impact learning?” We see that in this particular case the Wine Reviews Dataset which has 200 features (vs the Wine Quality dataset which only has 10 features) appears to have higher variance for most models while equally low bias. While some of this difference is no doubt due to the fact that these are two different classification problems and so will have different errors, it also might be due to the fact that higher dimensional data (assuming the same amount of data samples) will have higher variance because of the “Curse of Dimensionality” which states that as our complexity in the terms of feature dimensionality increases then our variance also increases and we need more and more training data to compensate for more features.

Experiment 3 - Evaluation on Test Data

Once best hyperparameters for each model were established the best models for each algorithm above were then tested on the original hold out 30% of Test data for each dataset from the beginning. This data was completely hidden until now so performance on this dataset is the most accurate measure of generalization.

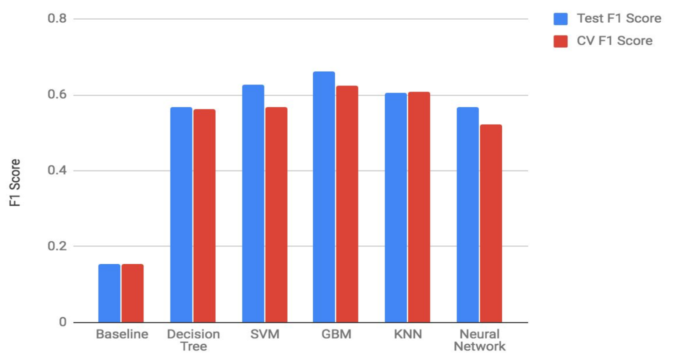
Wine Quality Test Set Performance of Best Model for each Algorithm

	Model	Test F1 Score	Test Accuracy	Test Precision	Test Recall	CV F1 Score	% Diff Test F1 and CV F1
0	Baseline	0.155	0.716	0.192	0.129	0.155	0.0
1	Decision Tree	0.568	0.832	0.548	0.589	0.563	0.9
2	SVM	0.628	0.870	0.677	0.586	0.567	10.8
3	GBM	0.663	0.887	0.753	0.592	0.623	6.4
4	KNN	0.606	0.849	0.591	0.622	0.609	-0.5
5	Neural Network	0.569	0.843	0.586	0.553	0.521	9.2

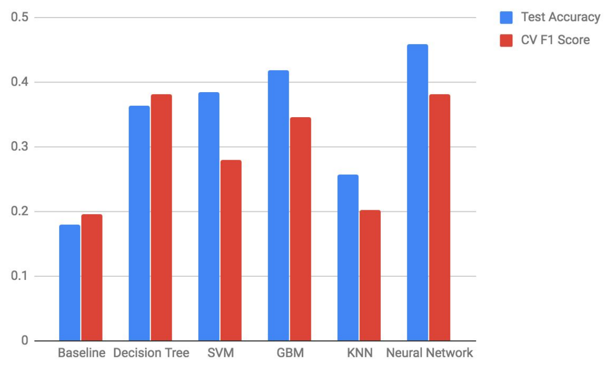
Wine Reviews Test Set Performance of Best Model for each Algorithm

	Model	Test F1 Score	Test Accuracy	Test Precision	Test Recall	CV F1 Score	% Diff Test F1 and CV F1
0	Baseline	0.180	0.705	0.166	0.196	0.180	0.0
1	Decision Tree	0.364	0.781	0.348	0.382	0.362	0.6
2	SVM	0.384	0.853	0.611	0.280	0.422	-9.0
3	GBM	0.418	0.842	0.528	0.346	0.436	-4.1
4	KNN	0.258	0.808	0.352	0.203	0.277	-6.9
5	Neural Network	0.459	0.852	0.573	0.382	0.477	-3.8

Wine Quality Data (Test vs CV F1 Score)



Wine Reviews Data (Test vs CV F1 Score)



From the charts above model performance can be compared. It can be seen that all models outperform the baseline. In the first dataset, Wine Quality (the small dataset) the best performing model is the GBM (0.66 F1 Score) model followed by SVM and the worst performing are both Decision Trees and Neural Networks both tied at 0.56 F1 Score. But in the second data, Neural Networks are the best. In addition all models outperform the baseline accuracy metric too despite being imbalanced in classes. Finally to double check that we aren't completely ignoring either Type 1 or Type 2 errors (False Positive or False Negative), Precision and Recall were also checked and we see all models have better precision and recall than random baseline. This tells us that all models are able to learn something. Finally in terms of run time all algorithms were run and the time taken to train was recorded. The results are as follows for the Wine Quality dataset. NN = 2.3s, GBM = 5.5s, DecisionTree = 0.03s, KNN = 0.007s, SVM = 0.14s. It makes sense for both NN and GBM to have higher run times as these are iterative algorithms training until some level of convergence. However a few things to note:

1.Some models do much better on the smaller dimensional data with continuous features than the larger dimensional dataset (Reviews) with binary features. For example, KNN is the 3rd ranked model in the Quality Dataset. But in the Reviews Data its F1 Score drops dramatically so that it is easily the worst. On the other hand, the Neural Network is tied for the worst performance in the first dataset with Decision Trees. But in the Reviews dataset it is by far the best performer even beating the GBM.

2. Second, most models in the Wine Quality dataset all have Test F1 Score within the same ballpark of around 60% +/- 5% points. This is a small spread so there isn't a huge variation in model performance regardless of algorithm for the first dataset. The difference between the best performing and worst performing model in this dataset is 0.66 - 0.56 = 0.1 (or the worst model is 15% worse compared to the best performer (the GBM)).

However, the second dataset, Wine Reviews (the large dimensional dataset with Number of features = 200) has much more variation in model performance. In this dataset it is the Neural Network which easily outperforms all other models with 0.46 F Score. The second best performing model is GBM with 0.42 F1 Score. However the worst performers this time perform worse by a much bigger factor. KNN has an F1 Score of only 0.25, SVM has an F1 Score of only 0.36. The difference between the best performing and worst performing model now in the second dataset is 0.46 - 0.25 or 0.21 or almost 46% worse compared to the best performer (the neural network).

3.Finally, when we compare the CV F1 Score with the Test F1 Score we can see that in the first dataset almost all models perform either the same on Test or actually even better (SVM jumps by 10% in performance on the test set). This tells us that there is little overfitting on test occuring when we train and cross validate the first small (Wine Quality) dataset. On the other hand, the second dataset (Reviews) which is large sees almost all of its models either flat or decline in out of sample Test performance compared to the CV score. For example, the SVM drops by 9% in performance this time and the KNN drops by 7% in performance on out of sample test data. This tells us that for the second problem of Wine Reviews our models do not generalize as well as the first dataset.

How do we explain these three observations? First, for 3) if we refer back to the learning curves we know that almost all models in the second Wine Reviews problem were struggling with higher variance possibly due to the "Curse of Dimensionality" so the drop in Test F1 Score for the second problem isn't as surprising as we would expect greater overfitting even on test data because our data size isn't any bigger (than the first dataset) but the feature size is a lot bigger in the second problem. But points 1) and 2) which show that Neural Networks perform best on the larger dimensional data and poorly on the first smaller dimensional data, and also that variation in performance between all models is much larger in the second than the first dataset, shed some insight into the initial question about "whether the dimensionality and type of features impacts learning". Neural Networks perform much better on very large dimensional data with hundreds of features often just binary or categorical in nature because by using hidden layers, and non-linear activation functions they can model very complex non linear decision boundaries and hypotheses. Their "restriction bias" in the set of hypotheses is smaller as they are not as constrained in the set of hypotheses compared to other simpler rule based models like KNN or Decision Tree or some linear models like Linear Regression or SVM. SVM can struggle with non-linear functions depending on the kernel used. By extension this might also explain why there is so much variance in both CV and out of sample Test performance. When the dimensionality of the dataset is small there is less feature complexity and the combined feature hyperplanes are in a smaller dimension. This makes the problems easier so that even simpler models can perform well. KNN for example does very well on a small featured dataset. As a result we have less variance in model performance regardless of the algorithm we choose but this variance in algorithm performance becomes more pronounced with higher dimensional data like Text or Image data.

Conclusion

In this paper, 5 models were trained on 2 datasets to predict wine quality. The first dataset included only continuous features and was small in dimensionality (10 features) while the second dataset included only non linear categorical features (words) and had big dimensionality (200 features). After training, hyperparam tuning and testing it was observed that first, on small dimensional datasets all models performed approximately similar in terms of F1 Score. But when looking at larger dimensional data with non linear binary features, there was greater variability in model performance with simpler algorithms like Decision Trees and KNN doing poorly compared to more complex Neural Networks. Second, most models showed greater error from Variance on higher dimensional data compared to lower dimensional data even if model performance generally improved with increasing training size on both datasets. Finally it was observed that hyperparameter tuning could greatly impact model performance but the impact of a hyperparameter (in terms of directionality) was similar regardless of the type of data used.