Feature Selection Using LASSO, Recursive Feature Elimination, and SelectFromModel on Cancer nRNA Dataset

Ismael Clark, Antonela Radas, Hector Ramirez

CAP4612-Introduction to Machine Learning

Florida International University

Table of Contents

Abstract

This paper covers the use of several supervised learning algorithms in a feature selection task on a dataset. This dataset contains normalized long non-coding RNA (lncRNA) expression profiles of 12 cancer types. Each algorithm yields a set of features which are then used by three different classifiers. To check the performance of the classification (for each set of features), the results are evaluated using different performance metrics.

Introduction

Feature selection is the process of selecting a subset of relevant features (variables, predictors) for use in model construction. Among the most common applications of this technique are:

- simplification of models to make them easier to interpret by researchers/users,
- shorter training times,
- to avoid the curse of dimensionality,
- improve data's compatibility with a learning model class,
- encode inherent symmetries present in the input space.

Data contains some features that are either redundant or irrelevant and can thus be removed without incurring much loss of information, which is the central idea in feature selection. Redundant and irrelevant are two distinct notions, since one relevant feature may be redundant in the presence of another relevant feature with which it is strongly correlated.

Another technique that shares the general goal of reducing the number of features with feature selection, but that differs in its implementation is feature extraction. Feature extraction creates new features from functions of the original features, whereas feature selection returns a subset of the features.

Among the different feature selection techniques figure:

- Unsupervised Algorithms:
    - Multi-cluster feature selection (MCFS)
    - Unsupervised discriminative feature selection (UDFS)
    - Autoencoder
- Supervised Algorithms:
    - LASSO
    - SelectFromModel (python package)
    - Recursive Feature Elimination

The algorithms from the supervised learning category were used during this project.

Datasets and Methodology

LASSO

Least Absolute Shrinkage and Selection Operator (Lasso) LASSO is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model. LASSO was originally formulated for linear regression models. However, it is easily extended to other statistical models including generalized linear models, generalized estimating equations, proportional hazards models, and M-estimators.

For its implementation, LASSO’s cost function is defined as:

$$\frac{1}{2N_{training}} \sum_{i=1}^{N_{training}} (y_{real}^{i} - y_{predict}^{i}) = \alpha \sum_{j=1}^{n} |\alpha_j|$$

where α_j is the coefficient of the j-th feature. The final term is called l1 penalty and α is a hyperparameter that tunes the intensity of this penalty term. The higher the coefficient of a feature, the higher the value of the cost function. This model seeks to optimize the cost function reducing the value of the coefficient. The values must be scaled to achieve meaningful results.

While implementing the model, using the LassoCV library from the sklearn.linear_model package. Also, $a_i = i - 1 + 0.1 \mid 0 = 0.01 \; and \; i \in [0, 10)$. Each one of these values of alpha yielded a set of features which were stored for later analysis.

Recursive Feature Elimination (RFE)

Recursive feature elimination is a feature selection algorithm that selects features based on how they affect the performance of a particular model. It reduces model complexity by removing features one by one until the optimal number of features is left. The algorithm can wrap around any model, and it produces the best possible set of features that gives the highest performance.

For its implementation, the RFE class was used, taken from the sklearn.feature_selection package. Logistic Regression was used as the estimator, as required by the class. An estimator is the model whose performance will be evaluated for a set of features returned from the algorithm.

As specified in the project instructions, the number of feature to select (n) by the algorithm in each iteration is defined as

$n_i = n_{i-1} \mid n_0 = 20$ and $i \epsilon [0, 5)$

SelectFromModel (SFM)

SelectFromModel is a python class that acts as a meta converter that selects features based on importance weights. The post-fit evaluator must have feature_importances_ or coef_ attributes. [Sources: 3, 9]. Once set up, the method needs to be adapted for the training set and with the get_support attribute the importance of features can be controlled. Like the RFE, SelectFromModel from Scikit-Learn is based on a Machine Learning Model estimation for selecting the features. The differences are that SelectFromModel feature selection is based on the importance attribute (often is coef_ or feature_importances_ but it could be any callable) threshold. By default, the threshold is the mean. As specified in the project instructions, the number of feature to select (n) by the algorithm in each iteration is defined as

$n_i = n_{i-1} \mid n_0 = 20$ and $i \epsilon [0, 5)$

Classifiers

K-Nearest Neighbors (KNN)

A supervised learning and model classification algorithm that helps us find which class a new input belongs to when the k nearest neighbors are selected and the distance between them is calculated. KNN, also known as

K-Nearest Neighbor, is a supervised learning and model classification algorithm that helps us determine which class a new input belongs to (test value) when k nearest neighbors are selected and the distance between them. The k-nearest neighbors (KNN) algorithm is a data classification technique for estimating the likelihood that a data point will become a member of a given group, based on the group to which the nearest data points belong. The k-nearest neighbors (KNN) algorithm is a simple and easy-to-implement supervised machine learning algorithm that can be used to solve classification and regression problems.

Supervised machine learning (as opposed to unsupervised machine learning) relies on labeled inputs to learn a function that produces appropriate output when new untagged data is provided. The unsupervised machine learning algorithm uses untagged inputs, in other words, no teacher (shortcut) tells the child (computer) when he is right or when he made a mistake so that he can correct himself.

Support Vector Machine (SVM)

In its basic form, linear partitioning, the SVM tries to find a line that maximizes the separation between the dataset in two 2D spatial point classes. SVM classifies data by finding the best hyperplane that separates all data points of one class from those of another class. The best hyperplane for an SVM is the hyperplane with the largest gap between the two classes. In the SVM algorithm, we are trying to maximize the headroom between data points and hyperplane. The distance between the vector and the hyperplane is called the margin. The distance from the vector to the hyperplane is called the field, which is the distance between the line and the closest point in the class. The hyperplane (line) passes through the maximum field, which is the maximum distance between the data points of the two classes. Then the classification is completed by finding the hyperplane that best distinguishes the two classes.

Random Forest (RF)

While an estimate of the prediction accuracy is used for the importance of the permutation, in random forests this Gini admixture can be used to measure the informativeness of a function in a model. Swap importance is a measure that tracks the accuracy of a forecast in which variables are randomly swapped from over-the-counter samples. Tree-based algorithms typically use mean for continuous functions or mode for categorical functions when predicting training samples in the regions to which they belong.

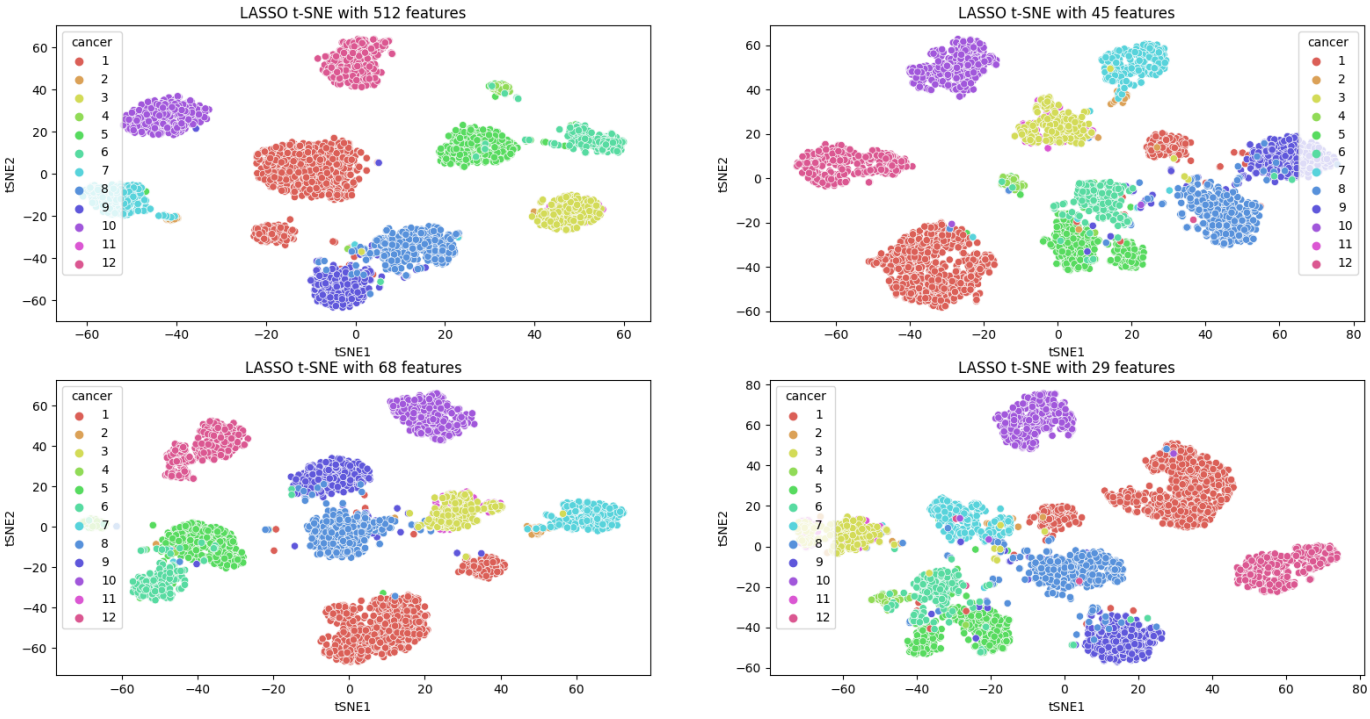
They can also make predictions with high accuracy, stability, and easy interpretation. Therefore, every data scientist should learn these algorithms and use them in their machine learning projects.

To facilitate the systematic and standardized refinement of somatic variants based on cancer sequencing data, random forest (RF) models and a deep learning (DL) approach have been used to demonstrate that these machine learning methods can provide classification efficiency. The classes are high and the same in all refinements of the variants (Ainscough et al., 2018). A machine learning approach called Cerebro improved the accuracy of invoking verified somatic mutations in tumor samples and outperformed many other methods for detecting somatic mutations (Wood et al., 2018). The potential for CNNs to use crude nucleotide sequences to classify cancer mutations was initially explored by using tag coding, hot coding, and embedding to preprocess DNA information.

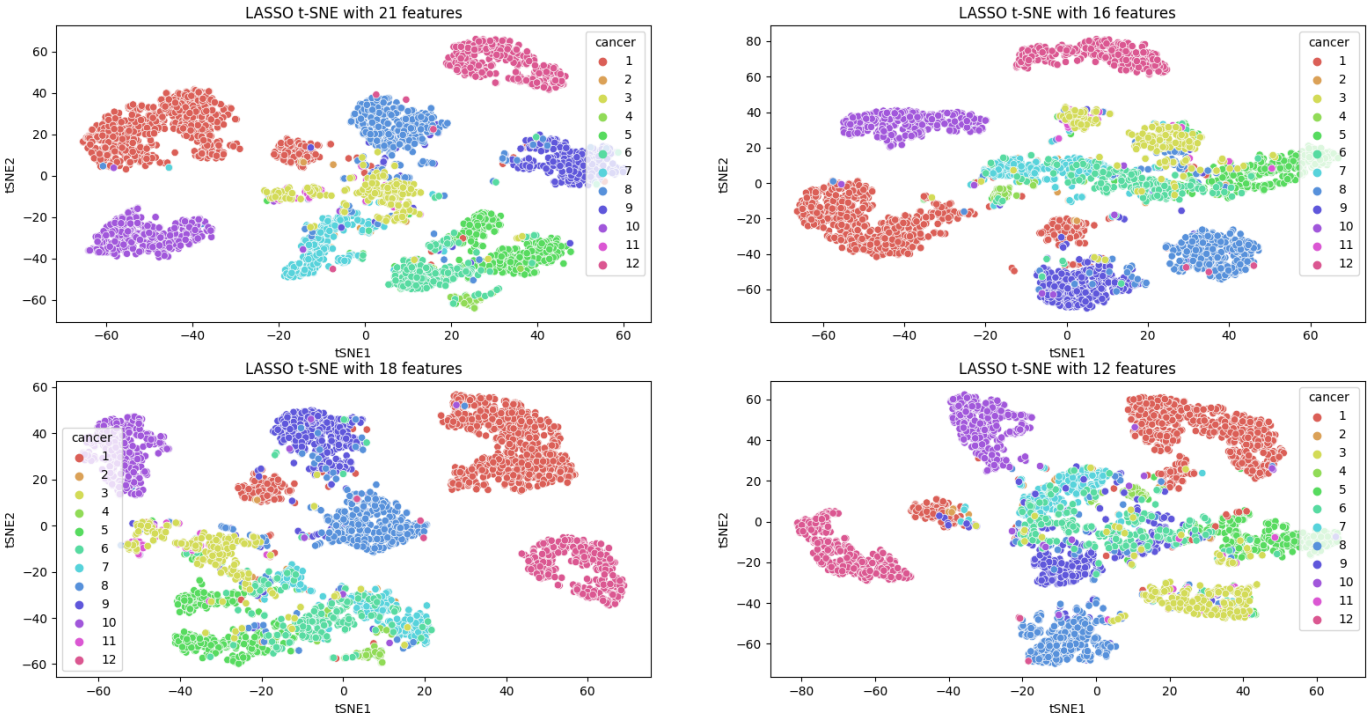
Results and Discussion

Lasso

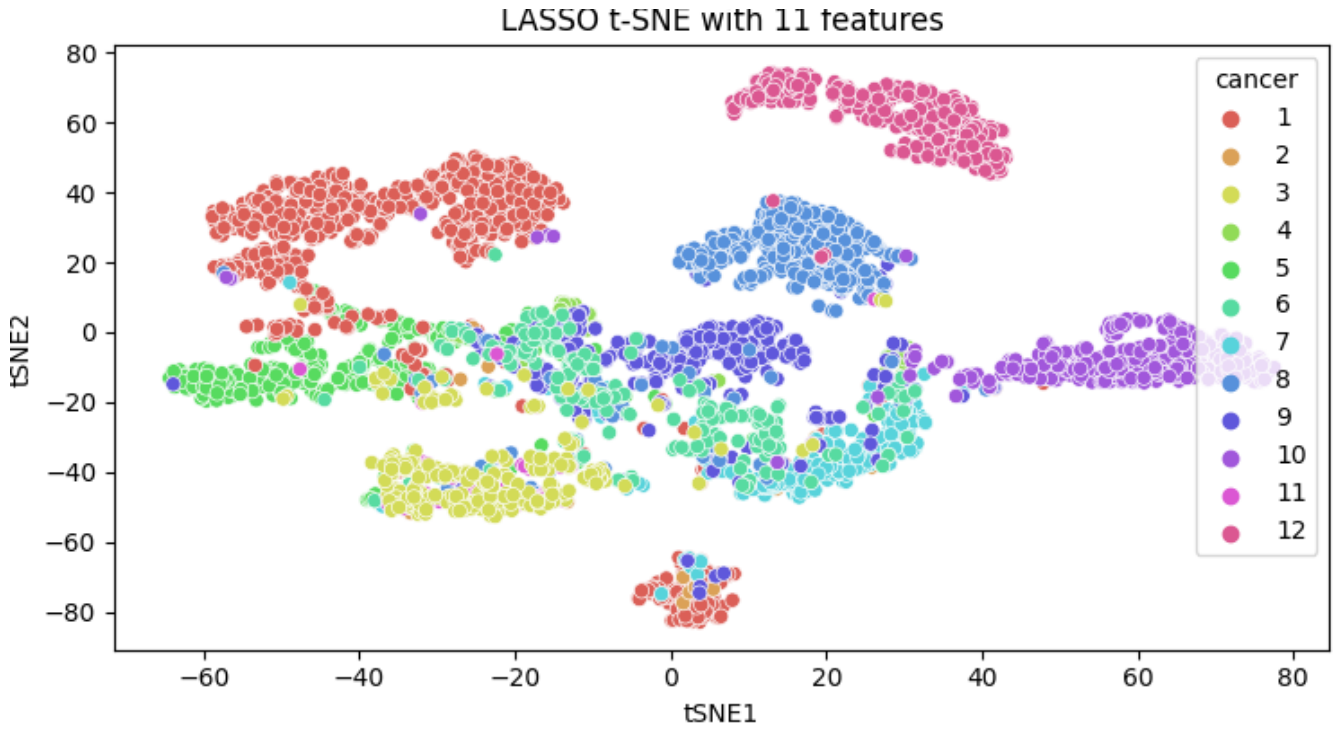
t-SNE

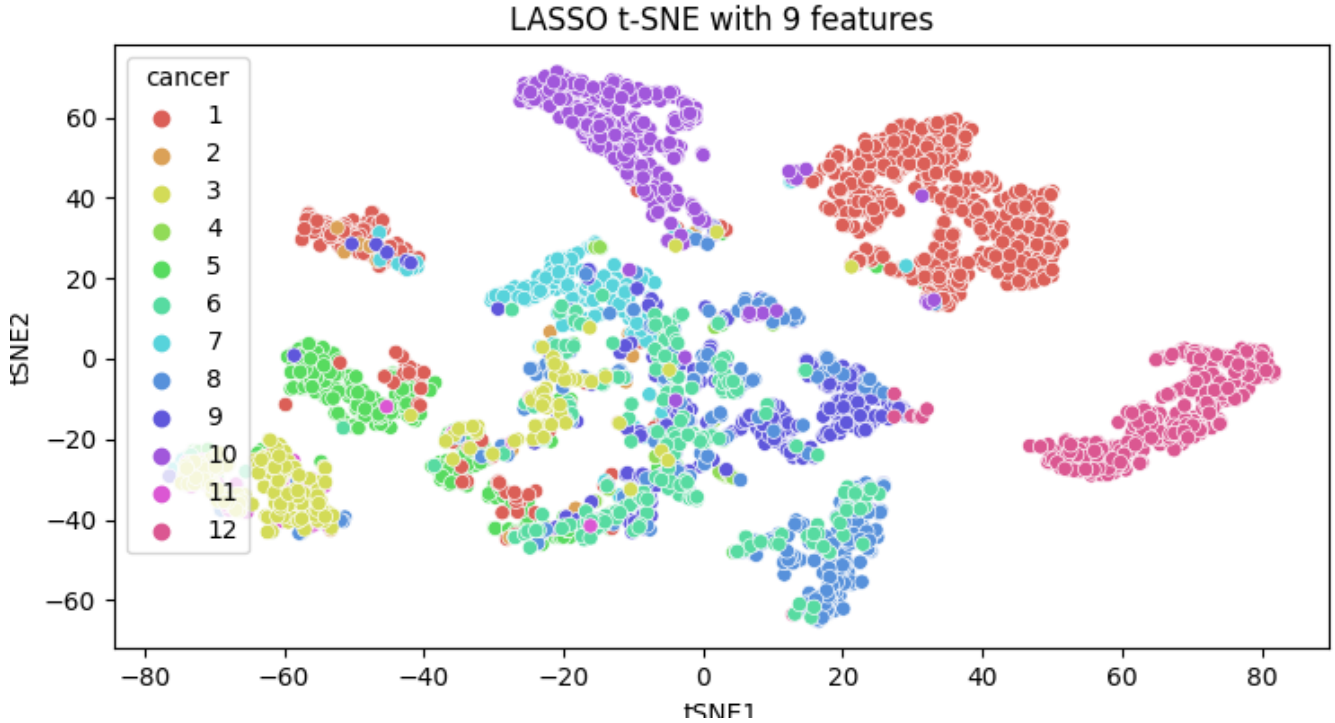


This image contains the tsne when lasso has selected 512 through 29 features. With 512 features we can see some real clear clustering and even with as low as 68 features classes seem to be very distinct.



This image shows selected features from 21 through 12. Compared the higher number of features there is a clear disorganization in the classes as the plotting becomes more chaotic.



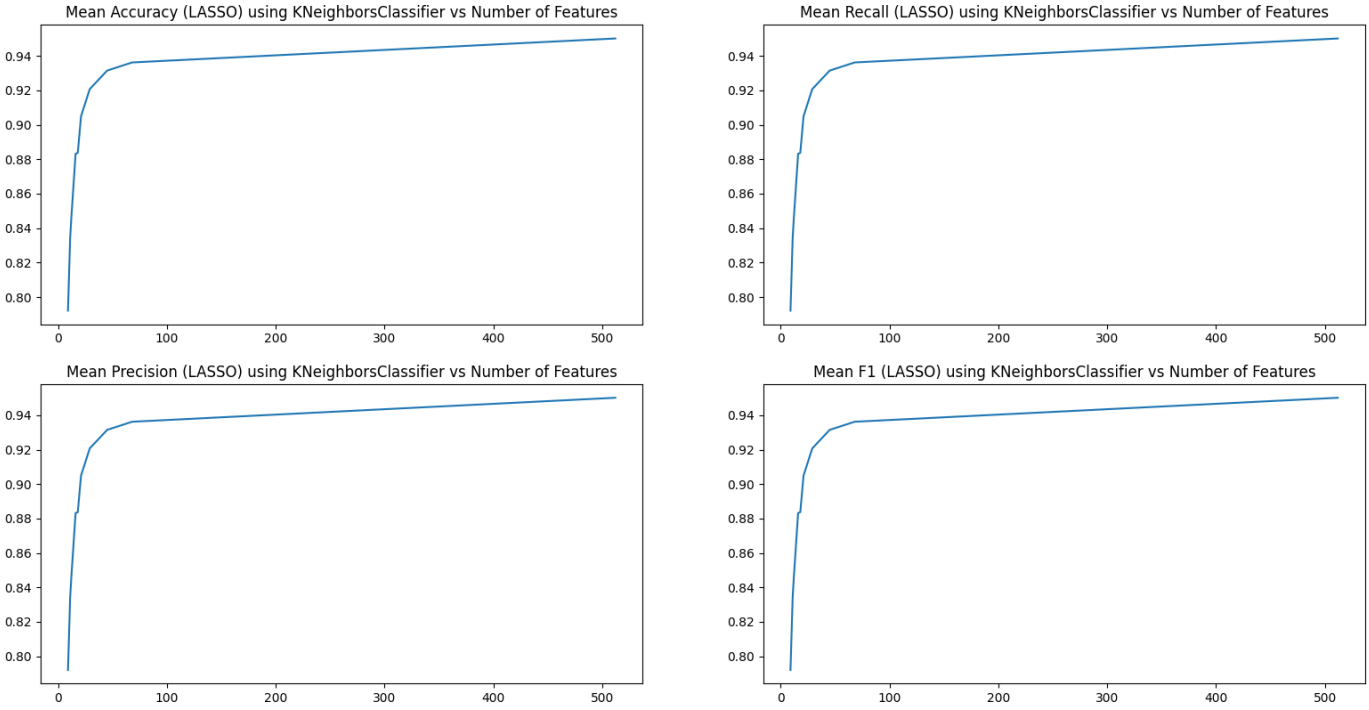


Lastly, this image shows lasso with 11 and 9 features selected. In this images the classes are even more chaotic with less clustering.

Performance Metrics

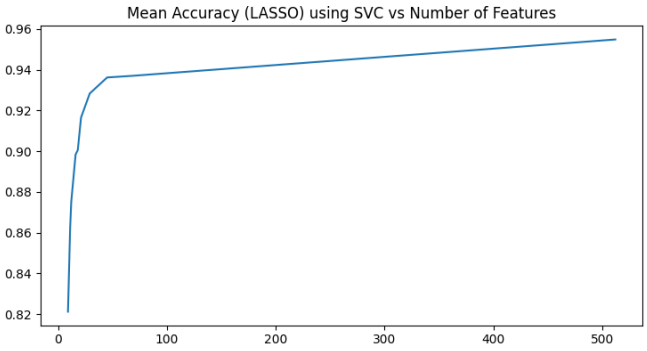
This section we will talk about how the performance metrics, average accurace, average recall, average precision, and average f1, change overtime with the number of features increase, using LASSO for Feature Selection and is verified using Stratified K fold, with 5 folds on each run.

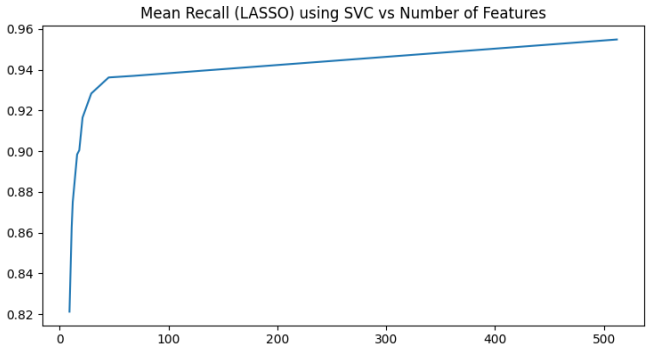
K-Nearest-Neighbor

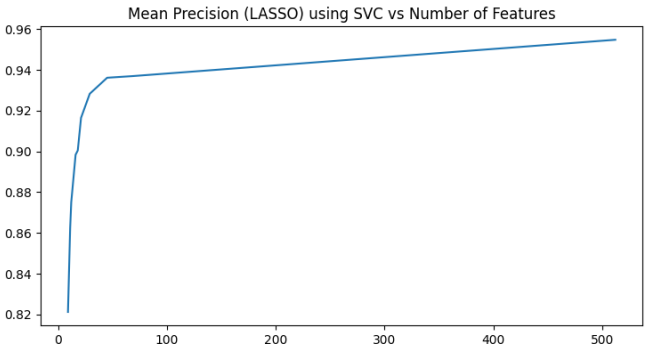


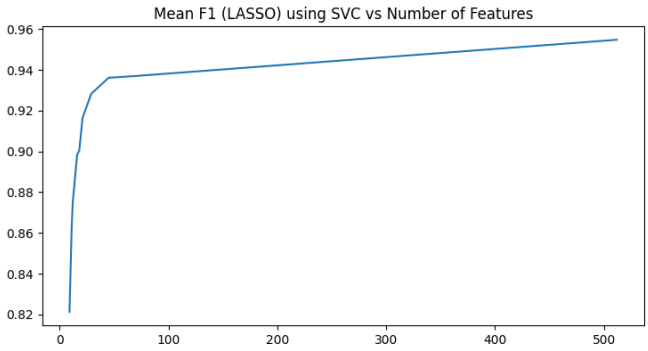
When running KNN on the data set with the LASSO selected feature, we can see that with high number of feature selection overall accuracy, precision, recall and f1 are around 94%, with a rapid decline when the number of features falls bellow 100.

Support Vector Machine



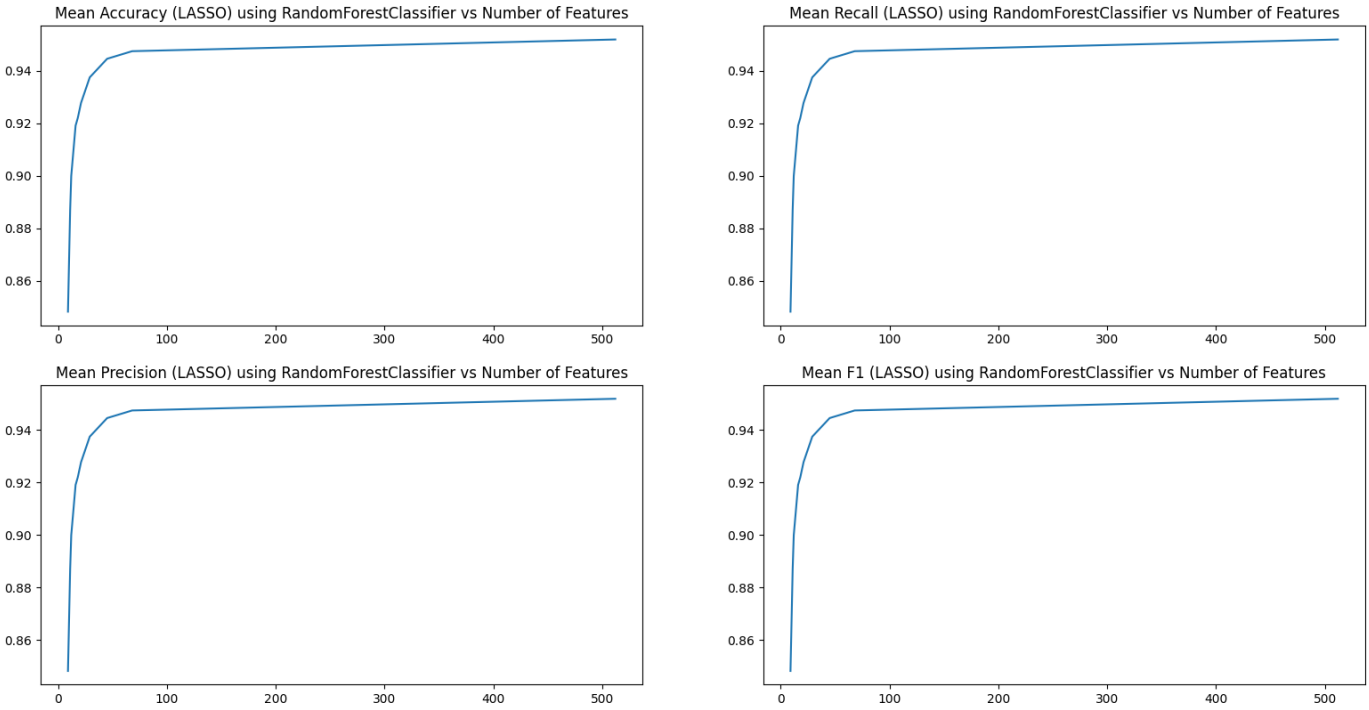






When running the same features through SVM we can see that just like KNN, SVM shows very high accuracy, precision, recall, and f1, with a sharap and rapid decline with the number of features fall bellow 100, this seems to indicate that optimal number of features is somewhere around the 100 features mark, with smaller gains as number of features increase.

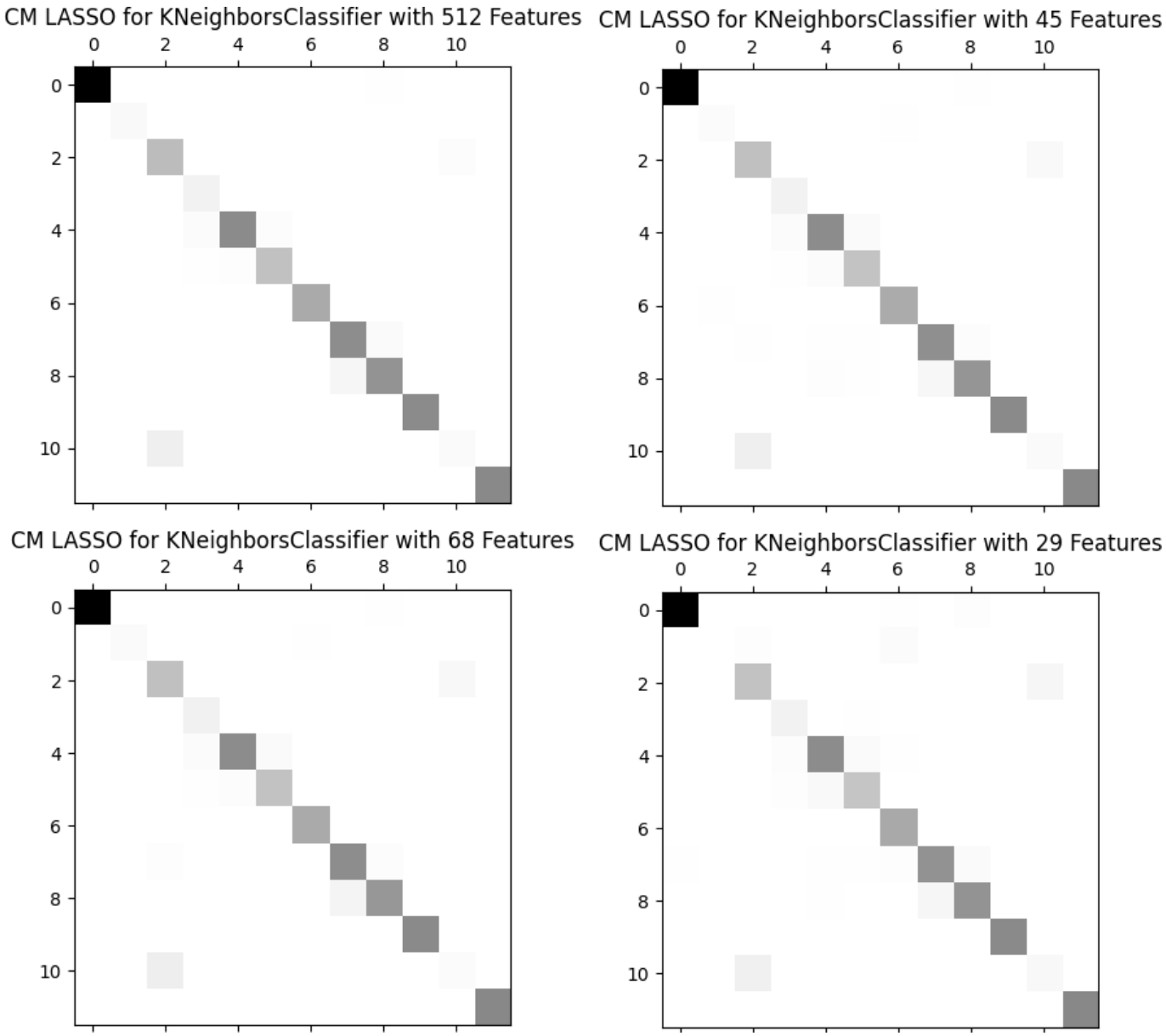
Random Forest



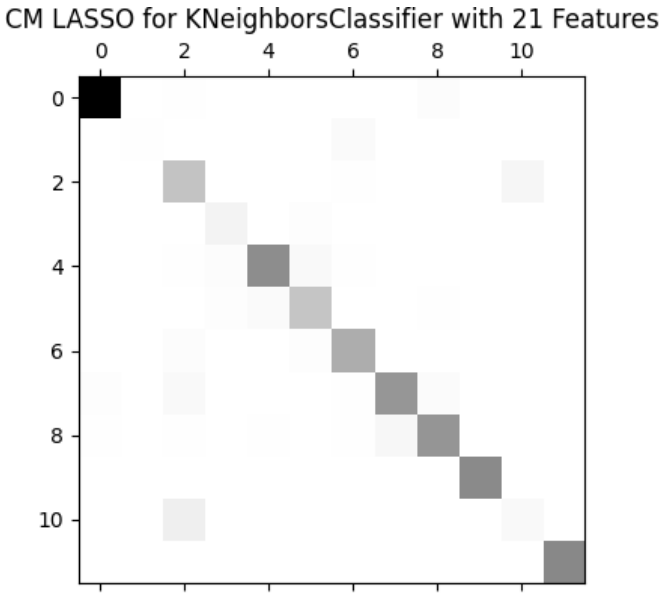
This image confirms the results from KNN and SVM, as the accuracy, recall, precision, and f1 scores are at 94% while number of features is above the 100, with a steep decline when they are bellow 100 features.

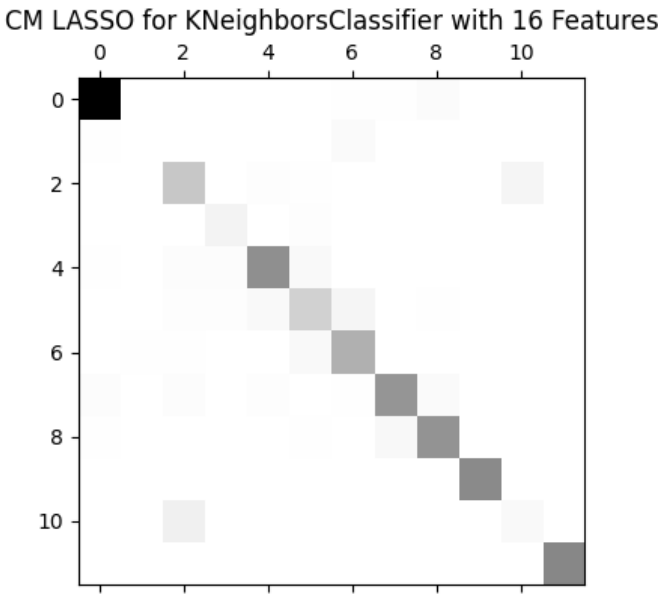
Confusion Matrix

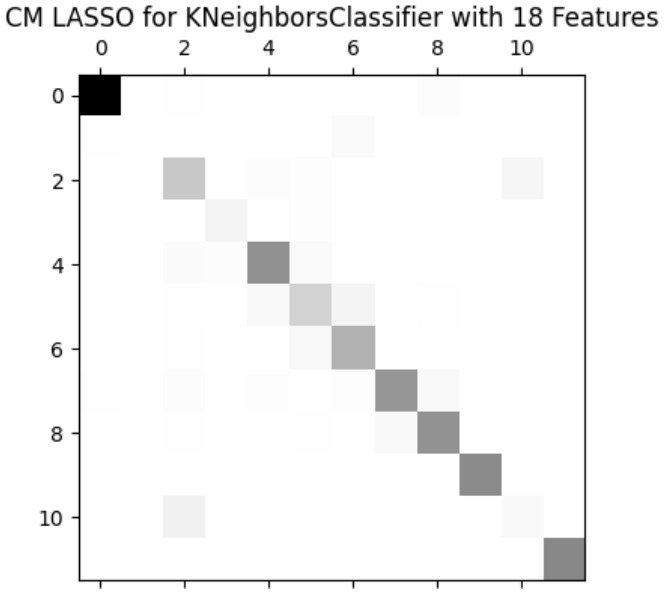
K-Nearest-Neighbor

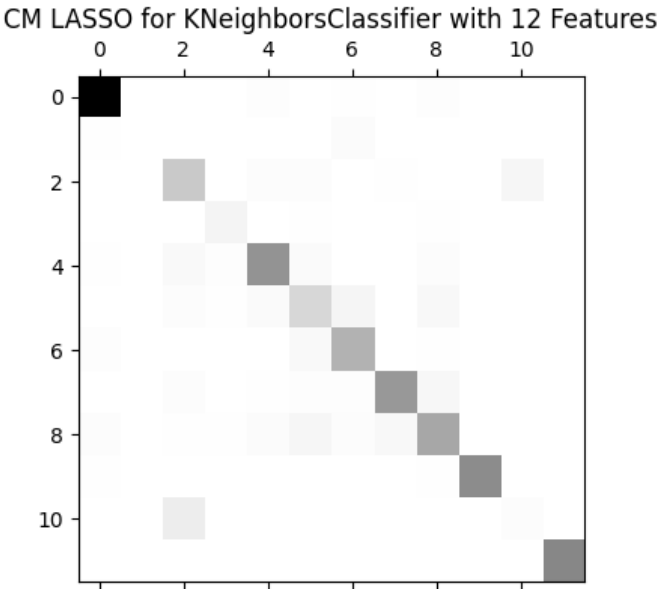


In this image we can see that there is a clear diagonal line with very litle diviation in the class prediction.

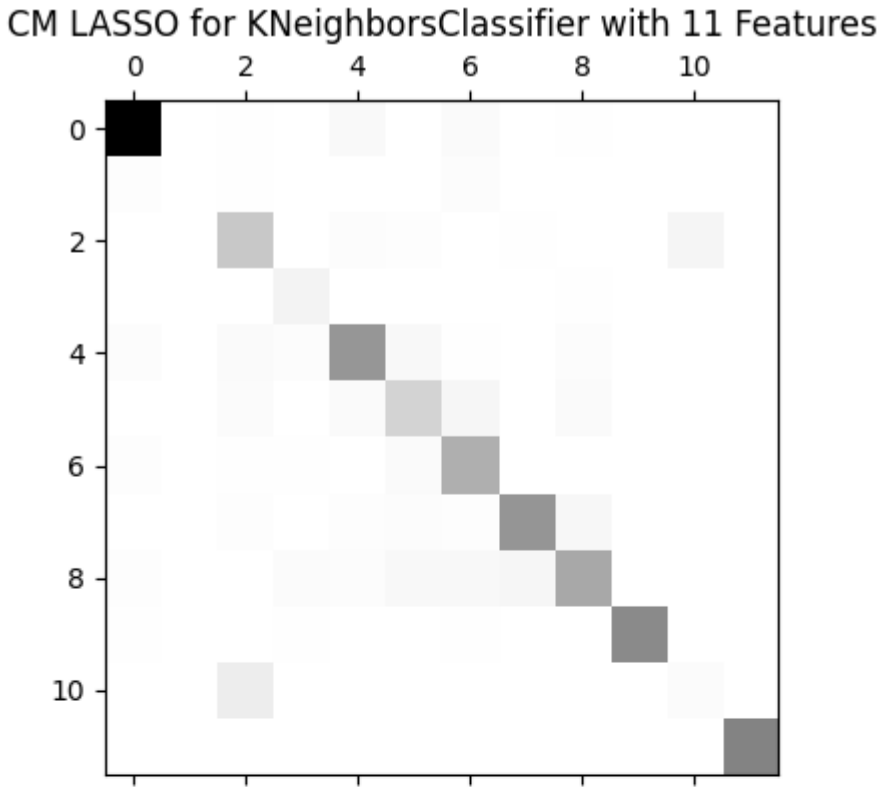


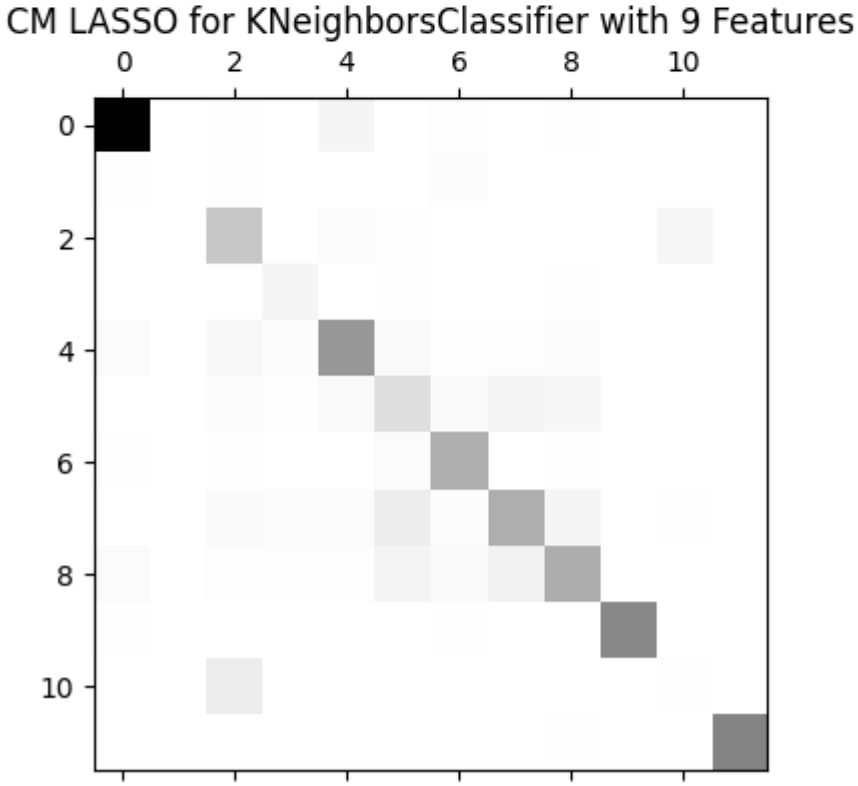






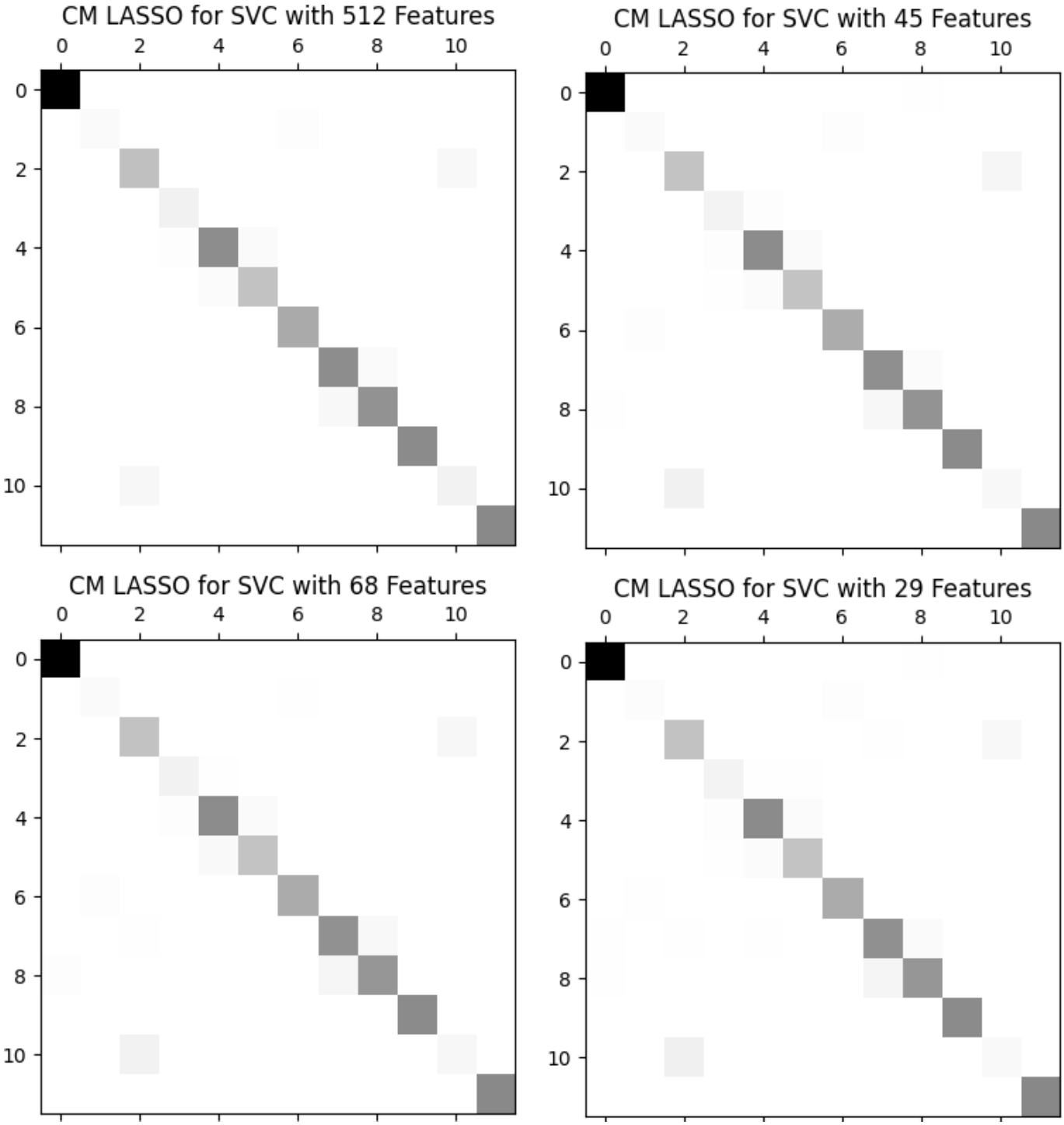
As the number of features start to fall bellow 100, we can start to see that there are more diviations from the diagonal, indicating higher instances of misclassification in the prediction.



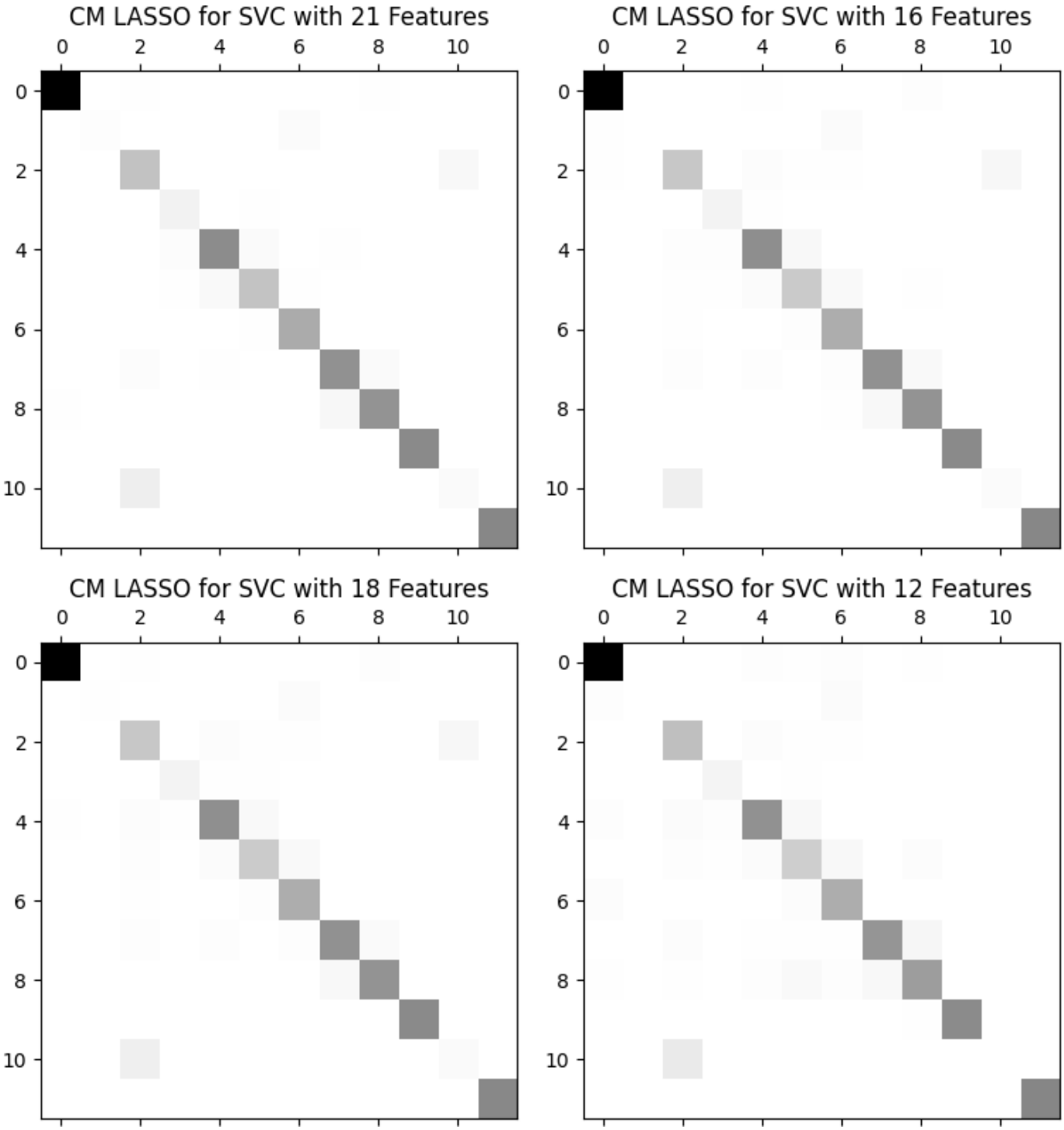


Lastly, with very few number of features, we can see that the confusion matrix has an increased instance of diviation, in particular with 9 features, which shows very high veriation and mislassification.

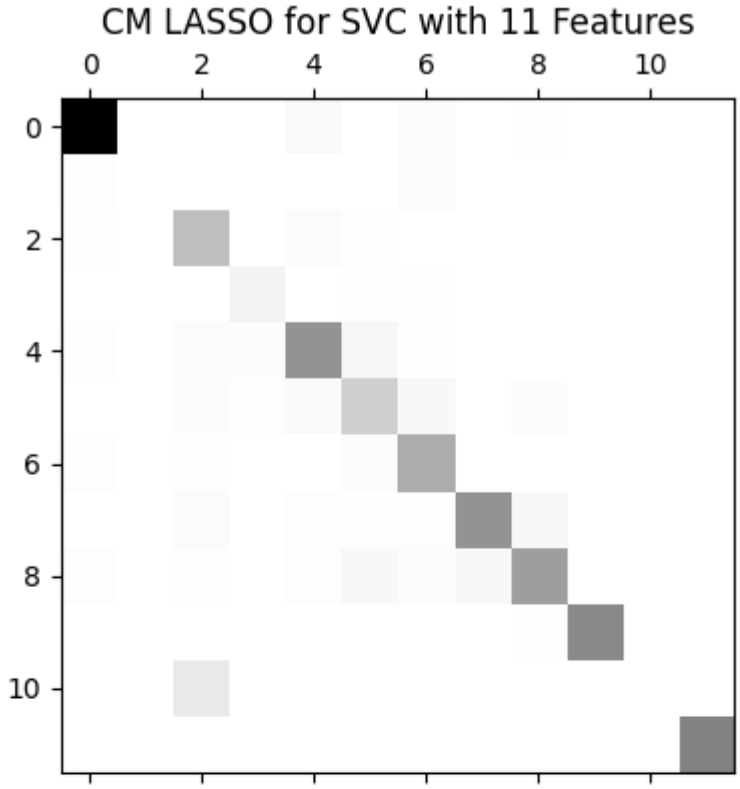
Support Vector Machine

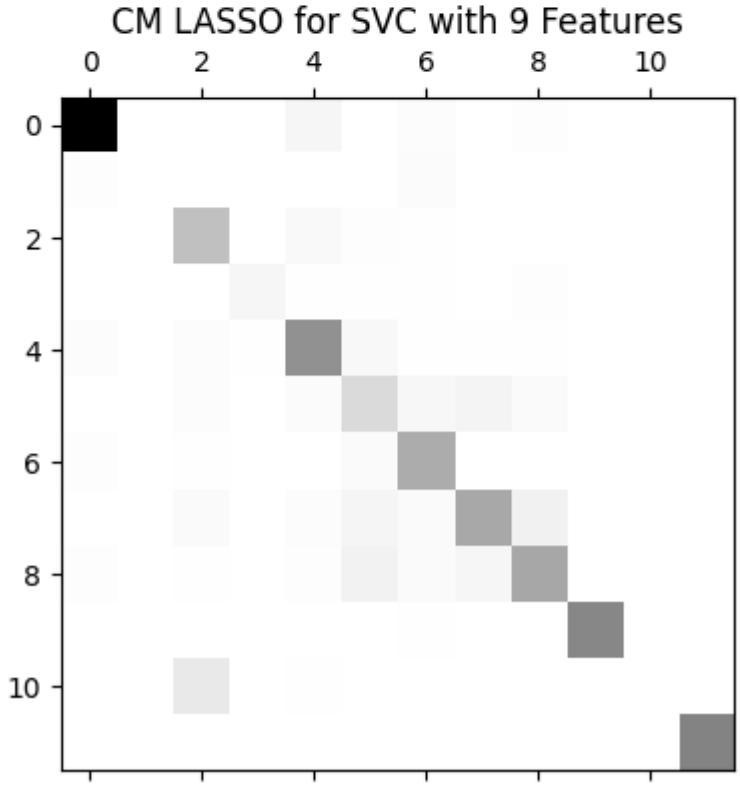


This image, just like in KNN, shows that the higher number of features have less divergance from the diagonal, this indicates that the accuracy and the TPR is higher overall, howver after 100 features, there are very similar results.



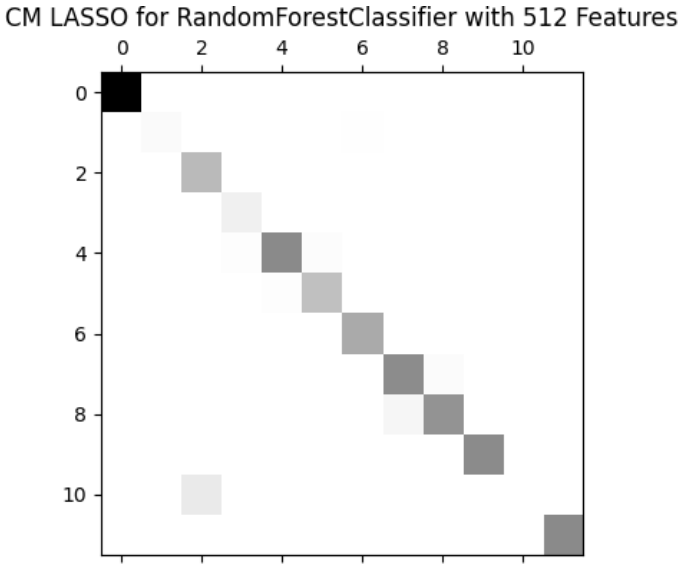
This image has features between 21 and 12, this also a higher degree of divergance from the diagonal which indicates higher rates of misclassification.

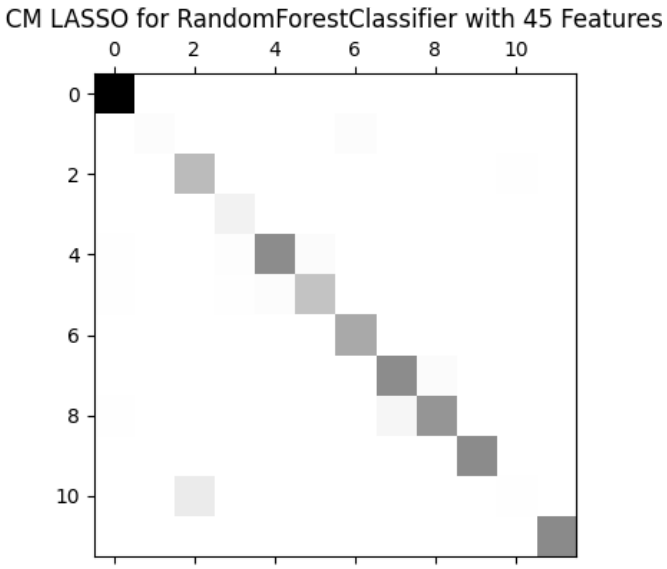


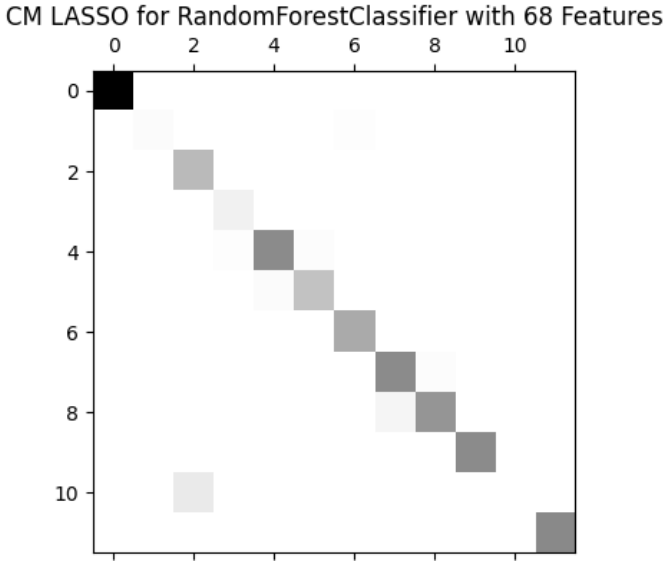


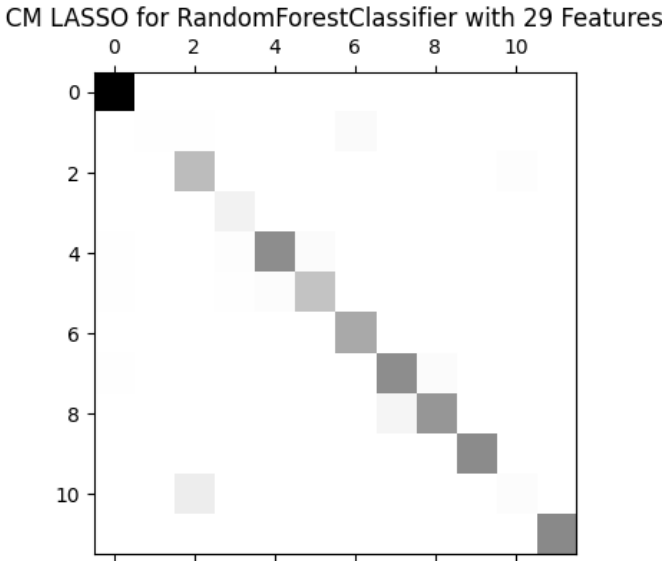
This image shows the confusion matrix with 11 and 9 features, both showed a higher degree of divergance from the diagonal and higher instances of misclassification.

Random Forest

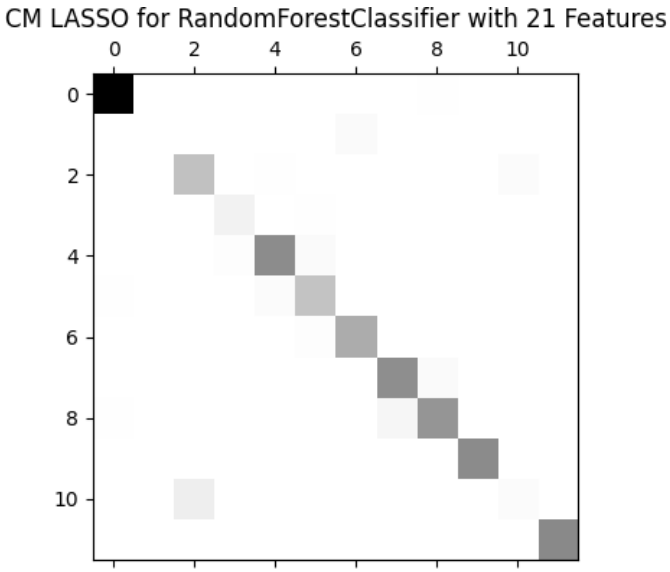


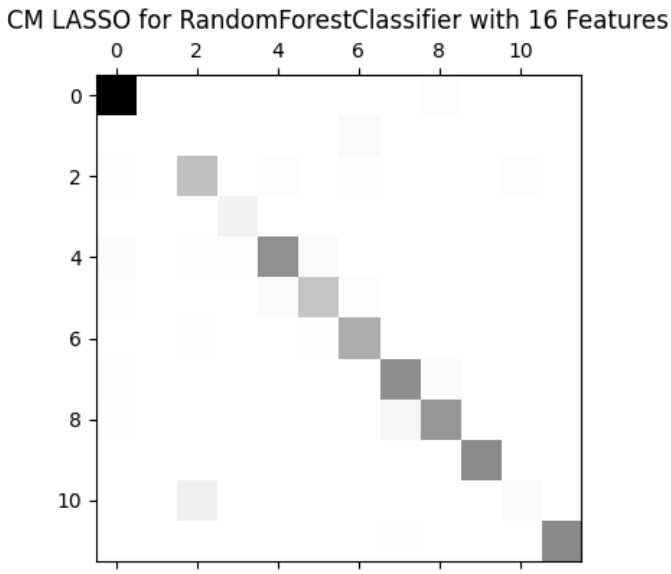


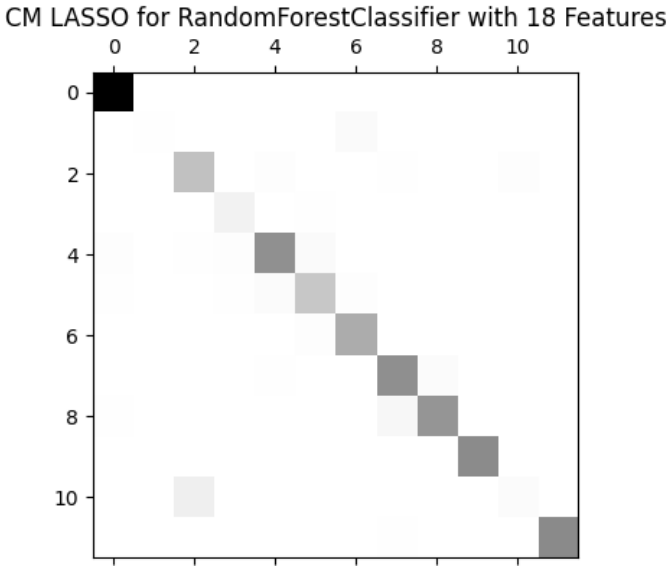


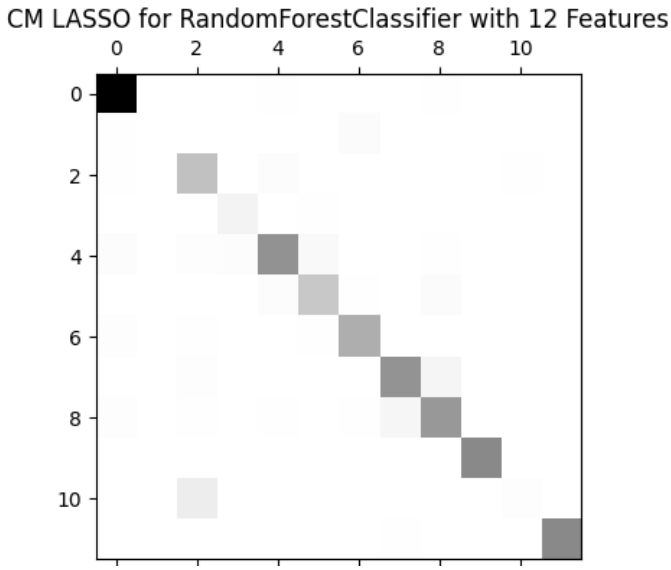


The random forest further confirms the findings from KNN and SVM, which indicate that with highe number of features there are less indices of divergance from the diagonal, and thus lower rates the misclassification.

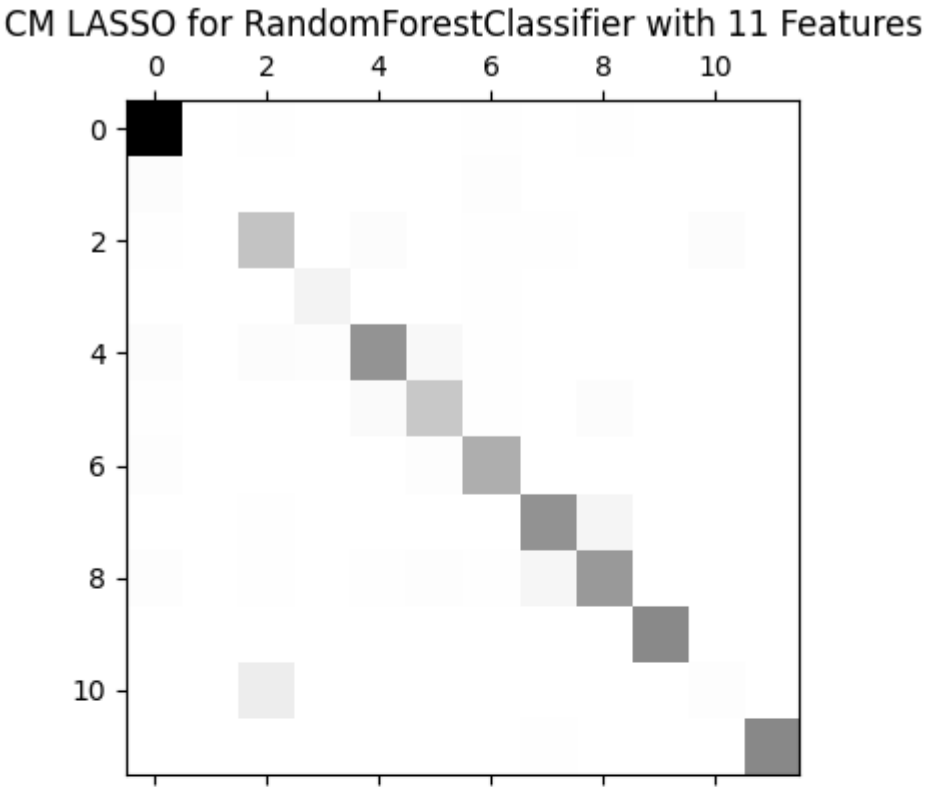


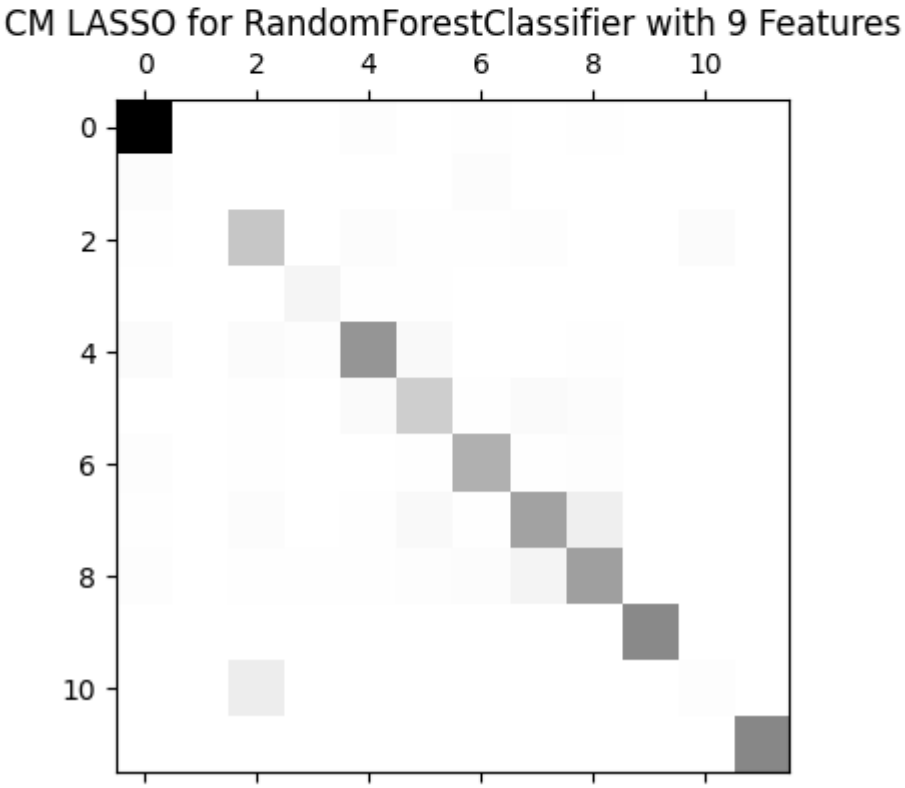






THis image just in like KNN and SVM shows Lasso with less the 100 features, and thus has much higher divergance from the diagonal and higher instances of missclassification.

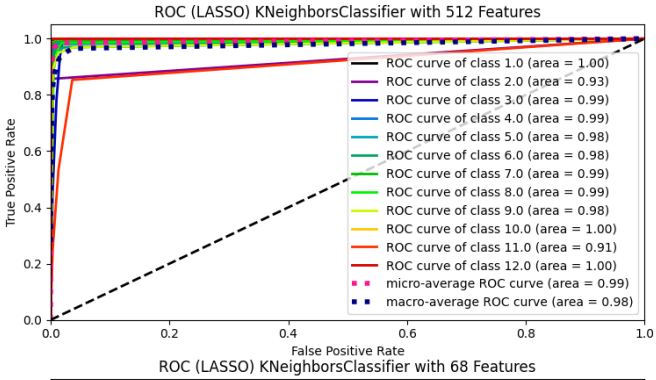


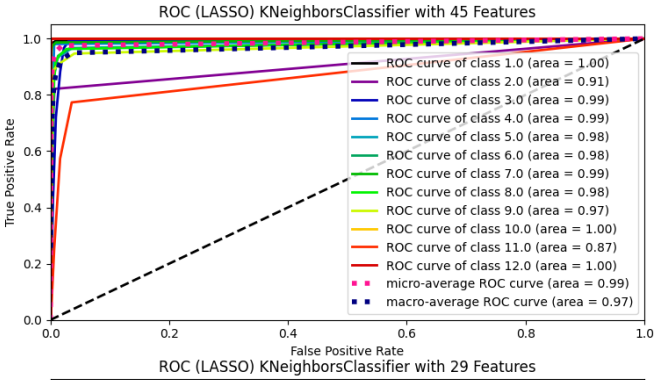


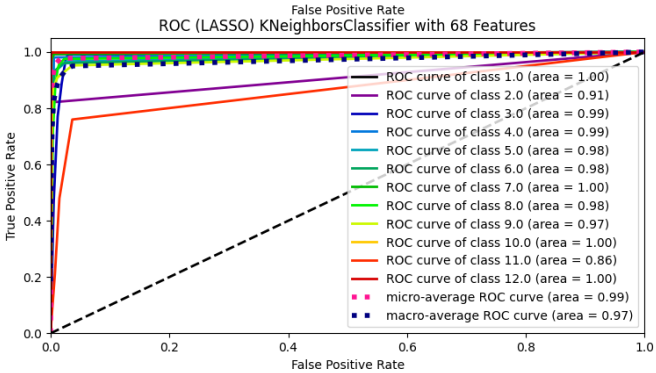
This image shows the extreme case, where even more divergence from the diagonal, indicating that there are higher instances of misclassification in the model. specially with classes 8 and 7, hiving a high decree of misclassification in the model.

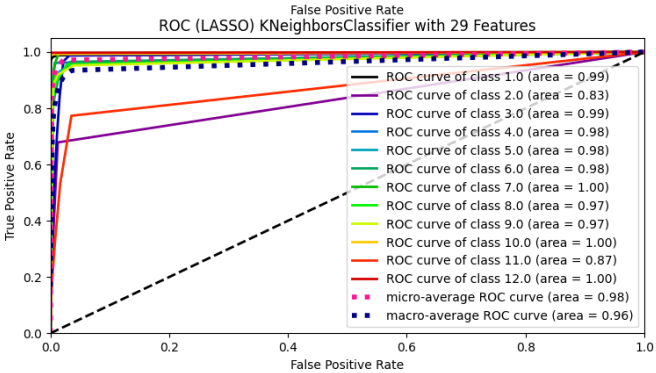
ROC Curves

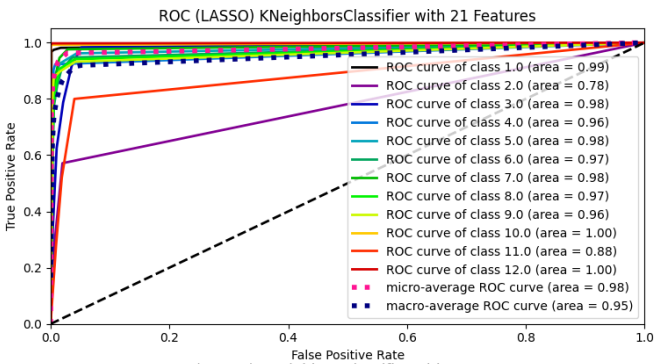
K-Nearest-Neighbor

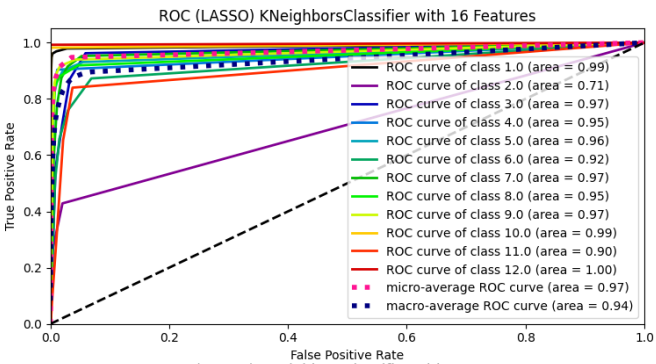


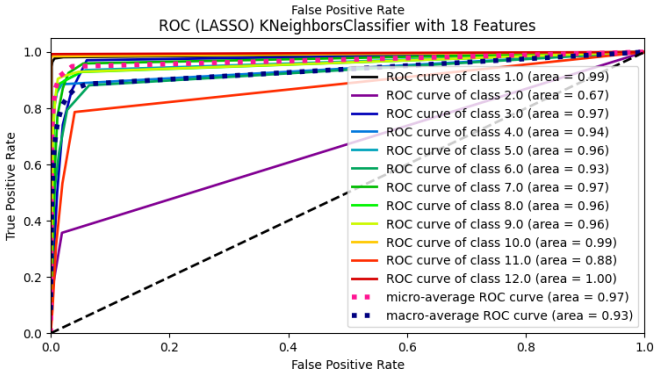


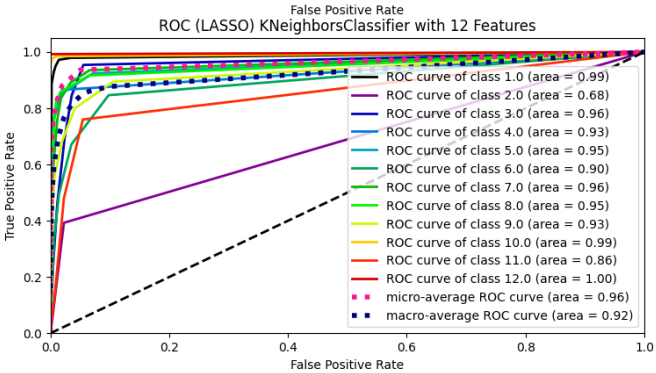


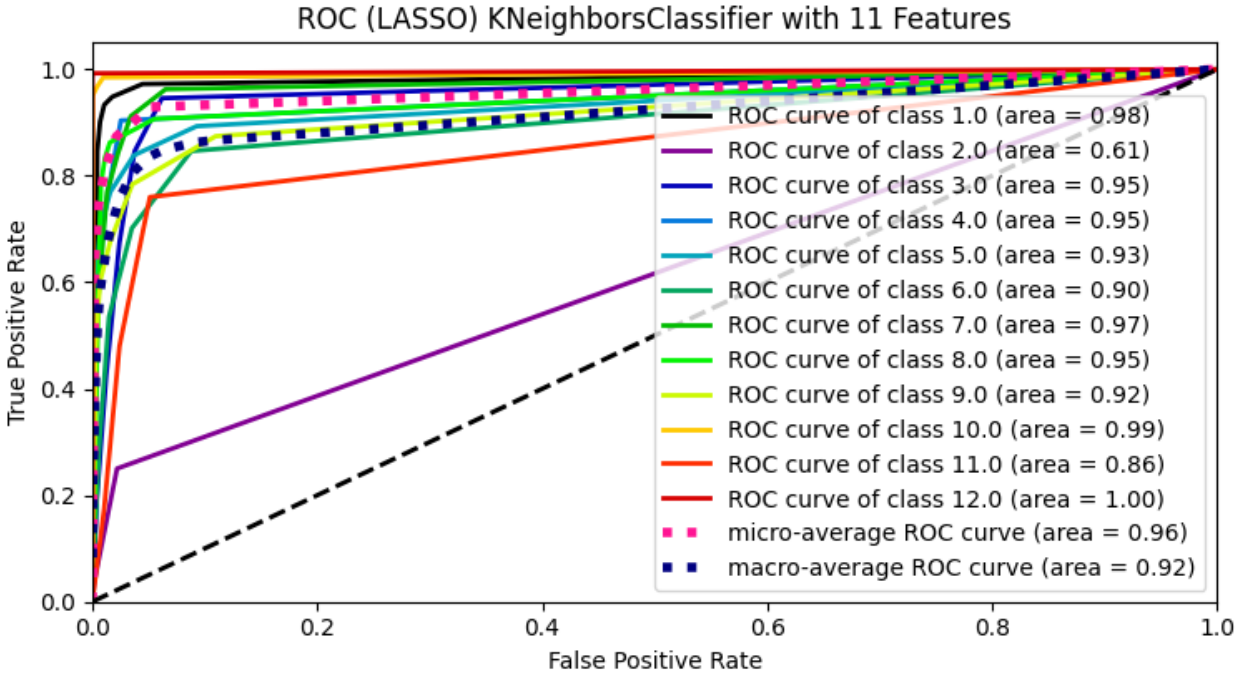


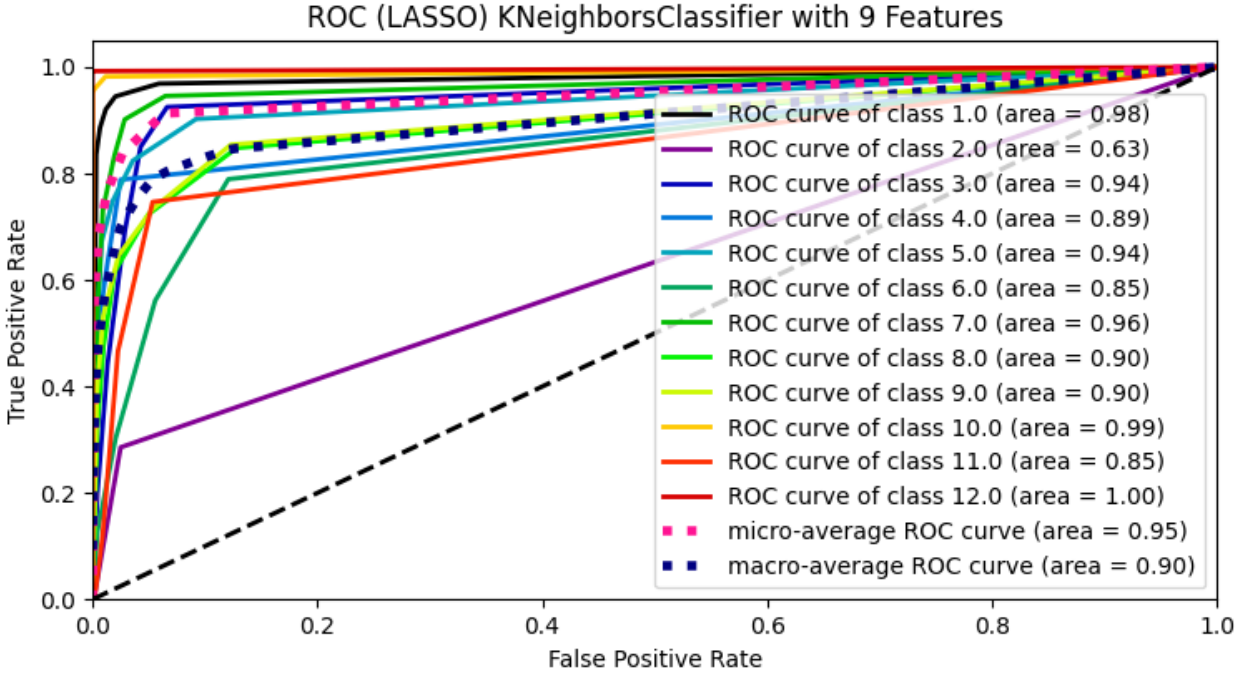






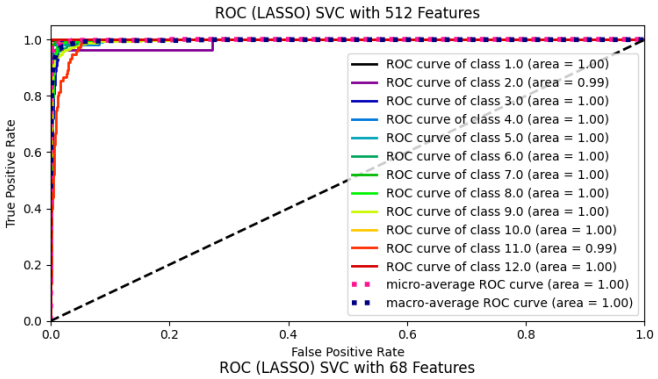


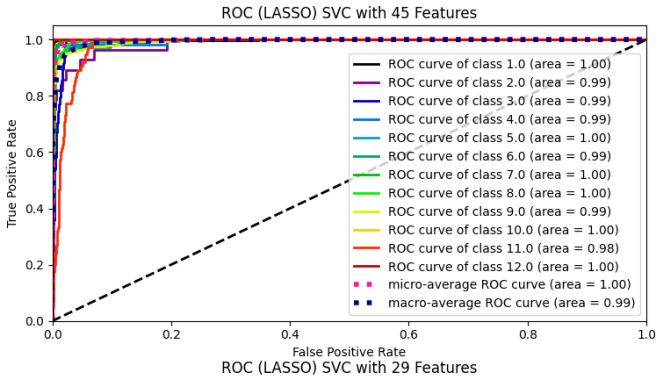


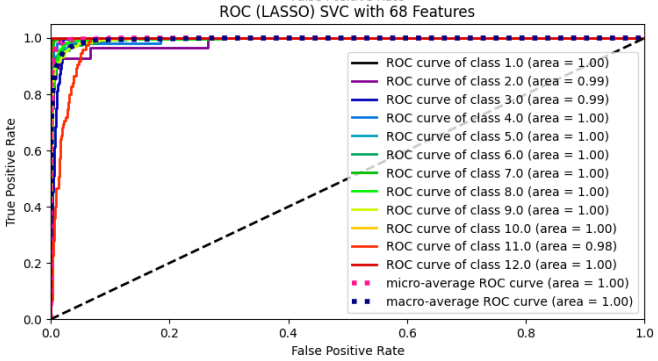


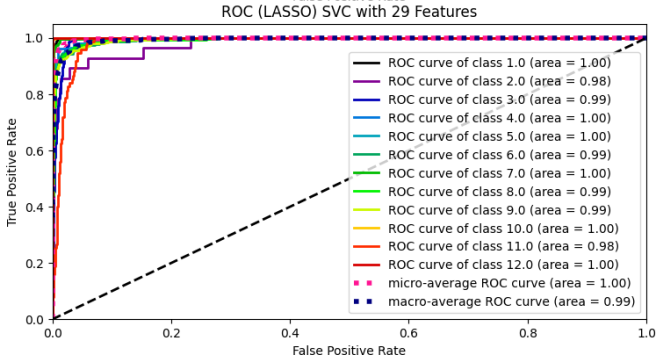
All this images show how the over TPR rapidly increases when the the number of features increases, but overall there was high rates of True positives with very low instances of False Positive Rate.

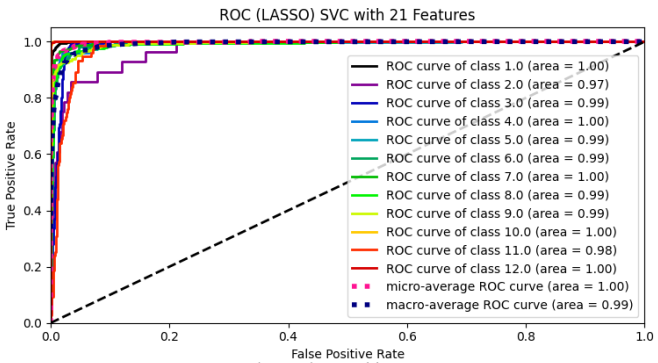
Support Vector Machine

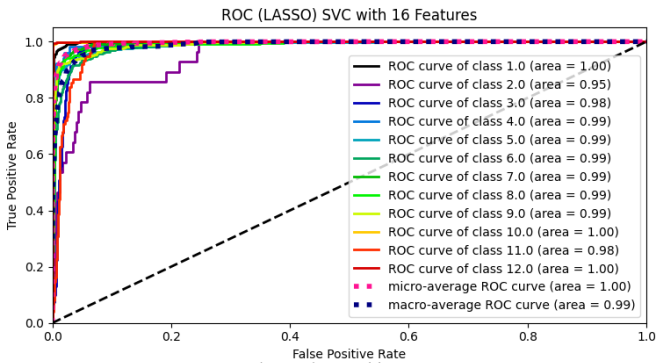


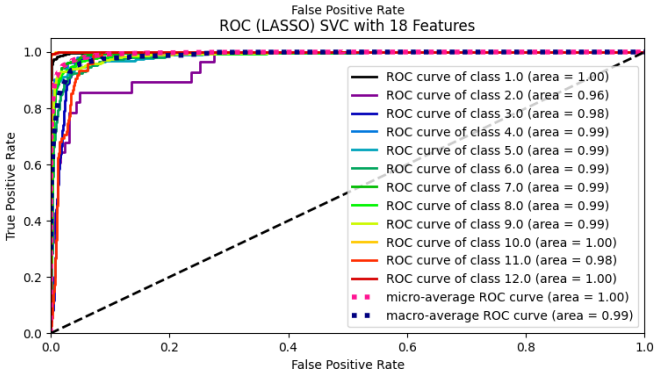


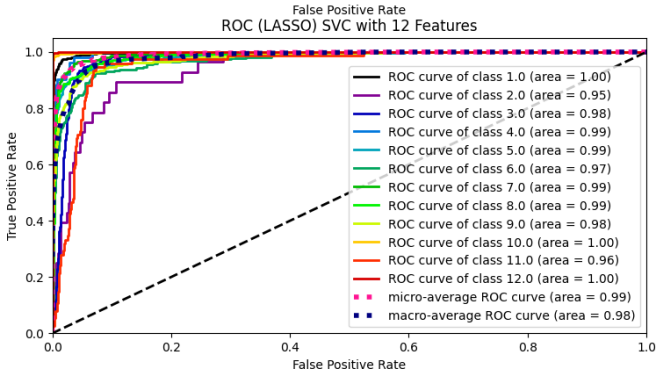


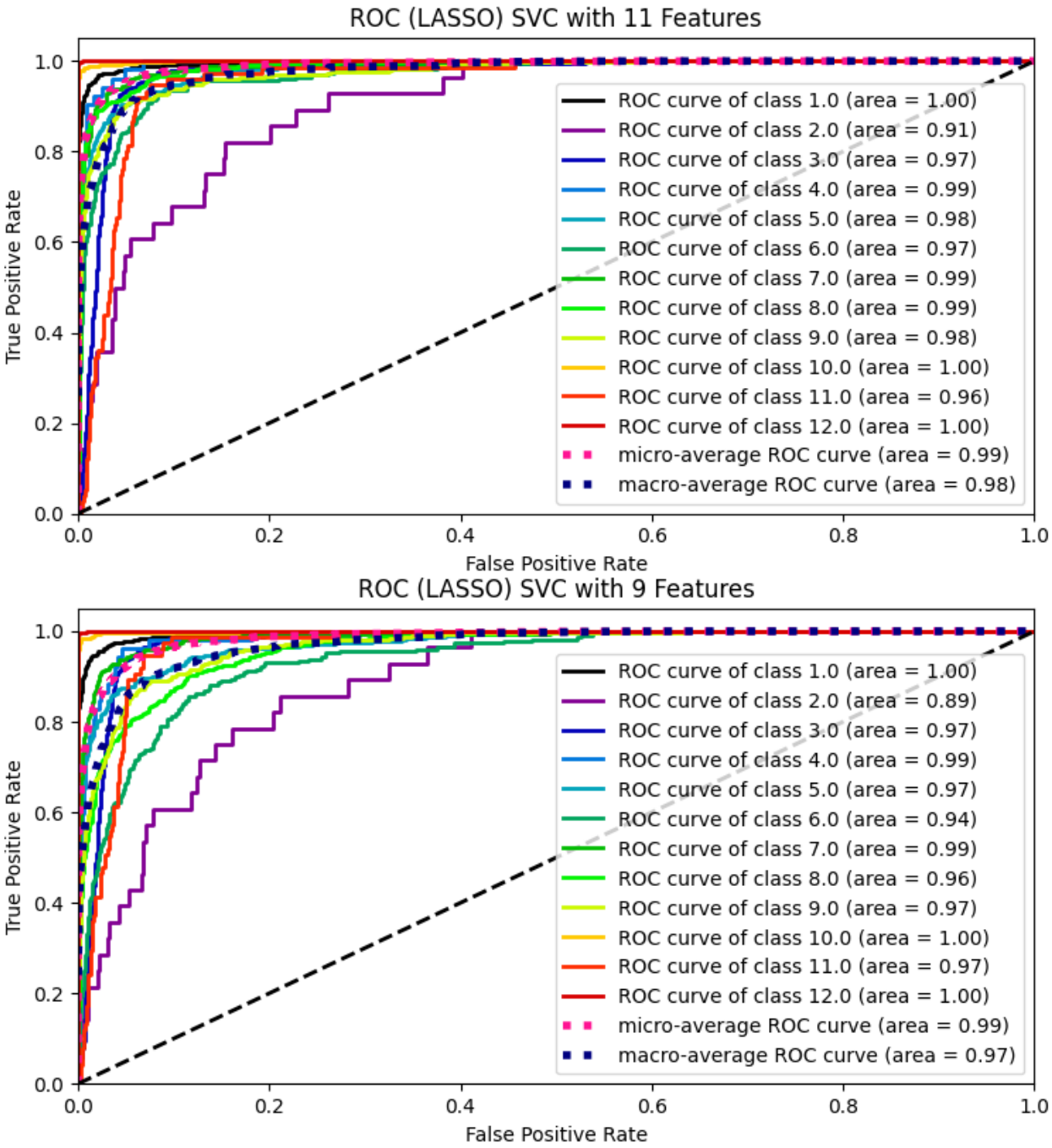






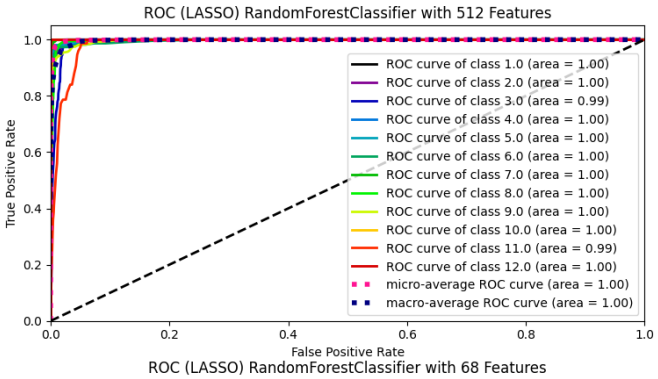


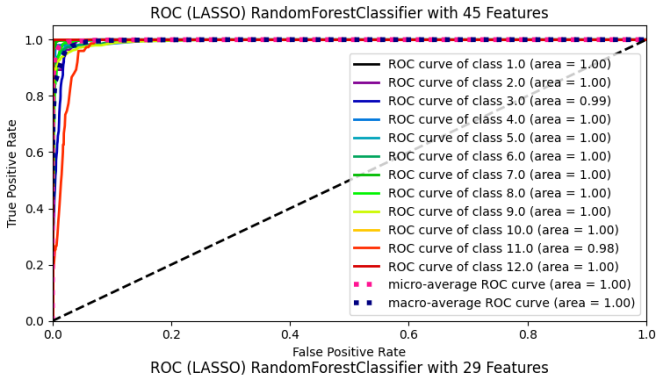


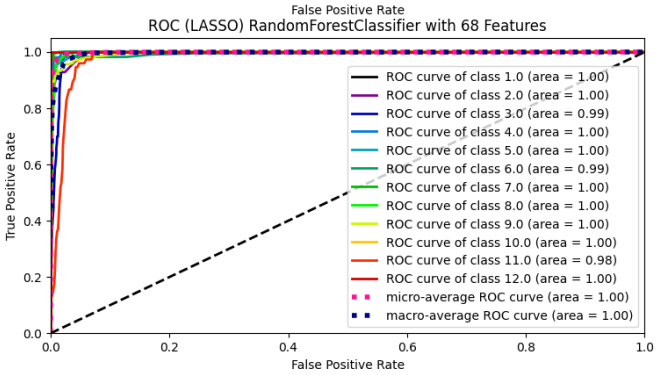


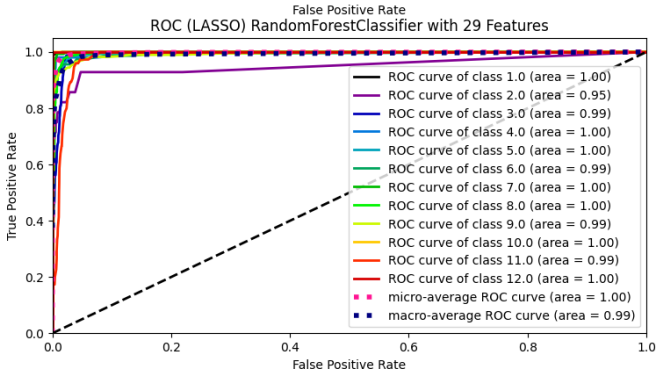
All this images show how the over TPR rapidly increases when the the number of features increases, but overall there was high rates of True positives with very low instances of False Positive Rate.

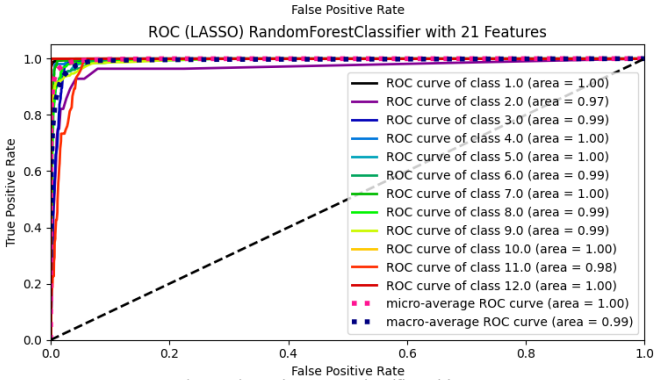
Random Forest

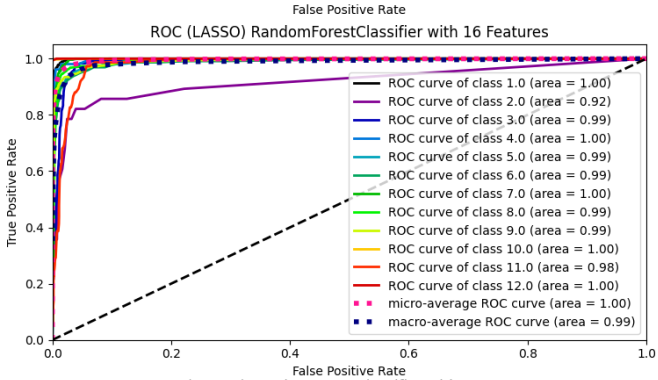


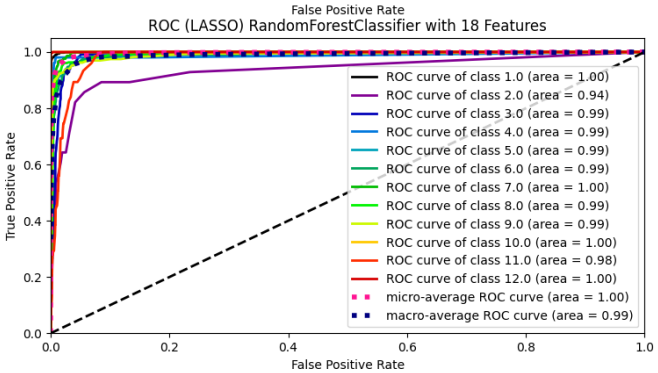


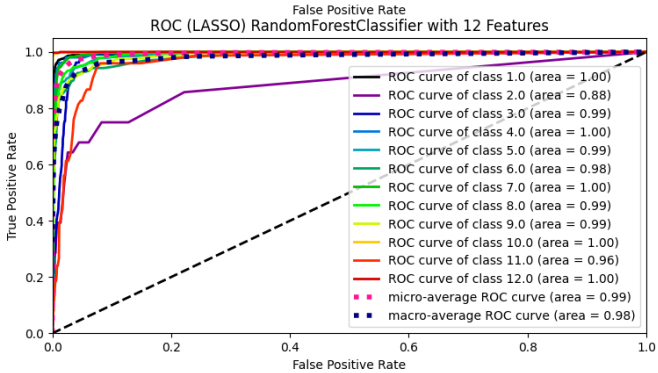


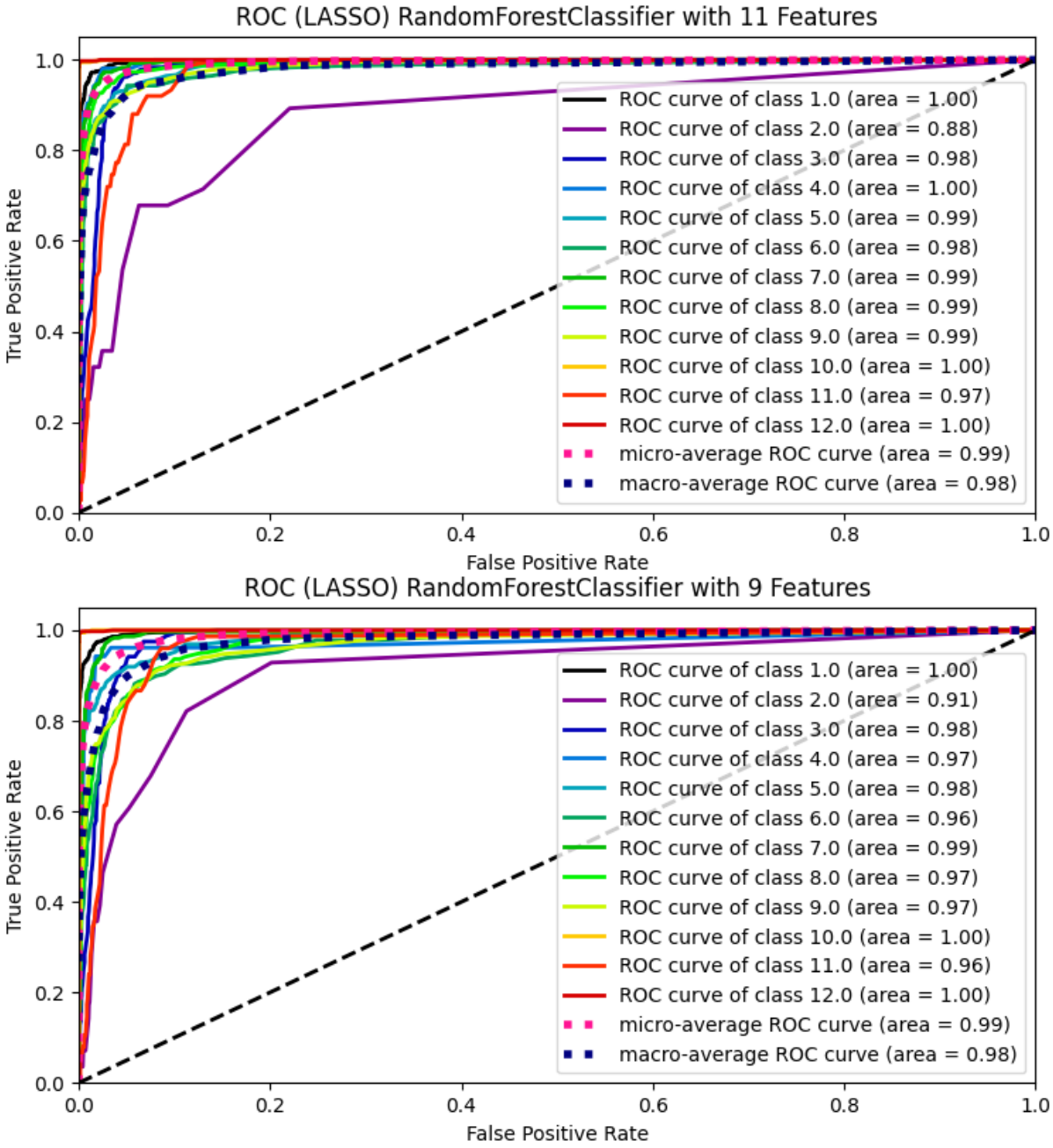








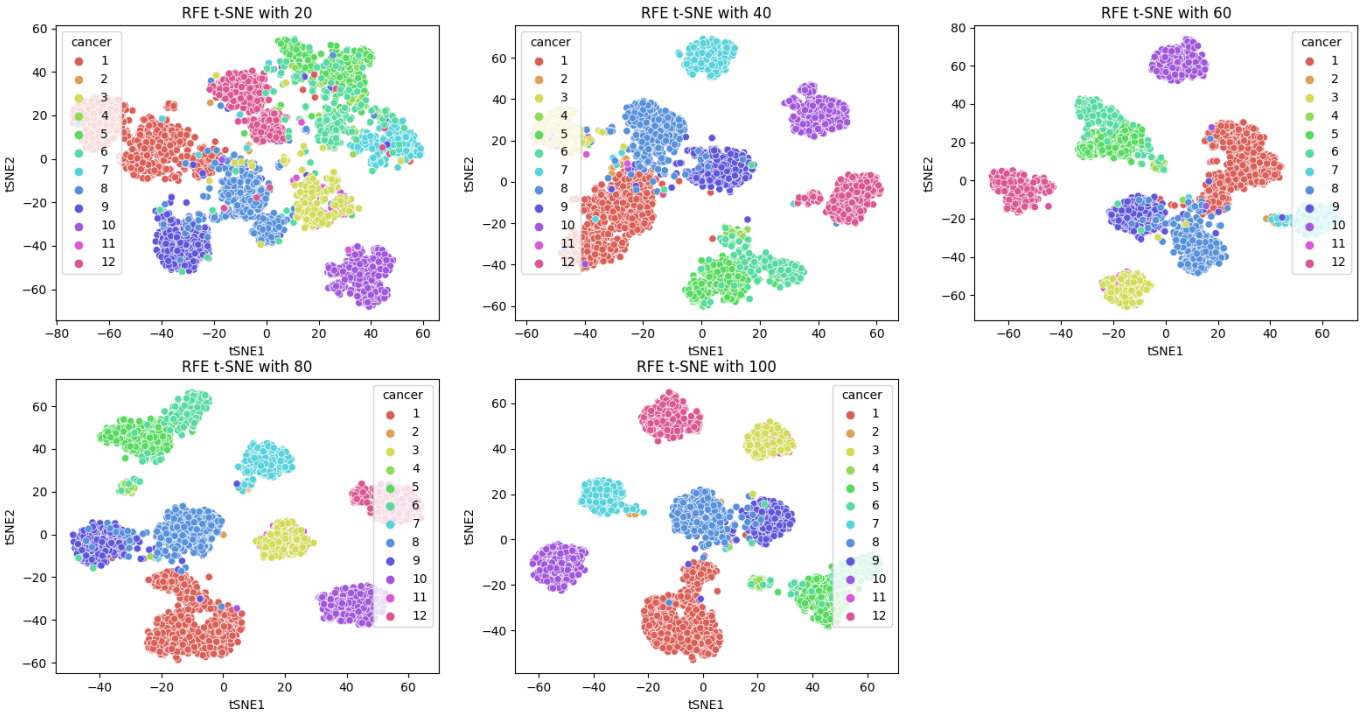




All this images show how the over TPR rapidly increases when the the number of features increases, but overall there was high rates of True positives with very low instances of False Positive Rate.

Recursive Feature Elimination (RFE)

t-SNE

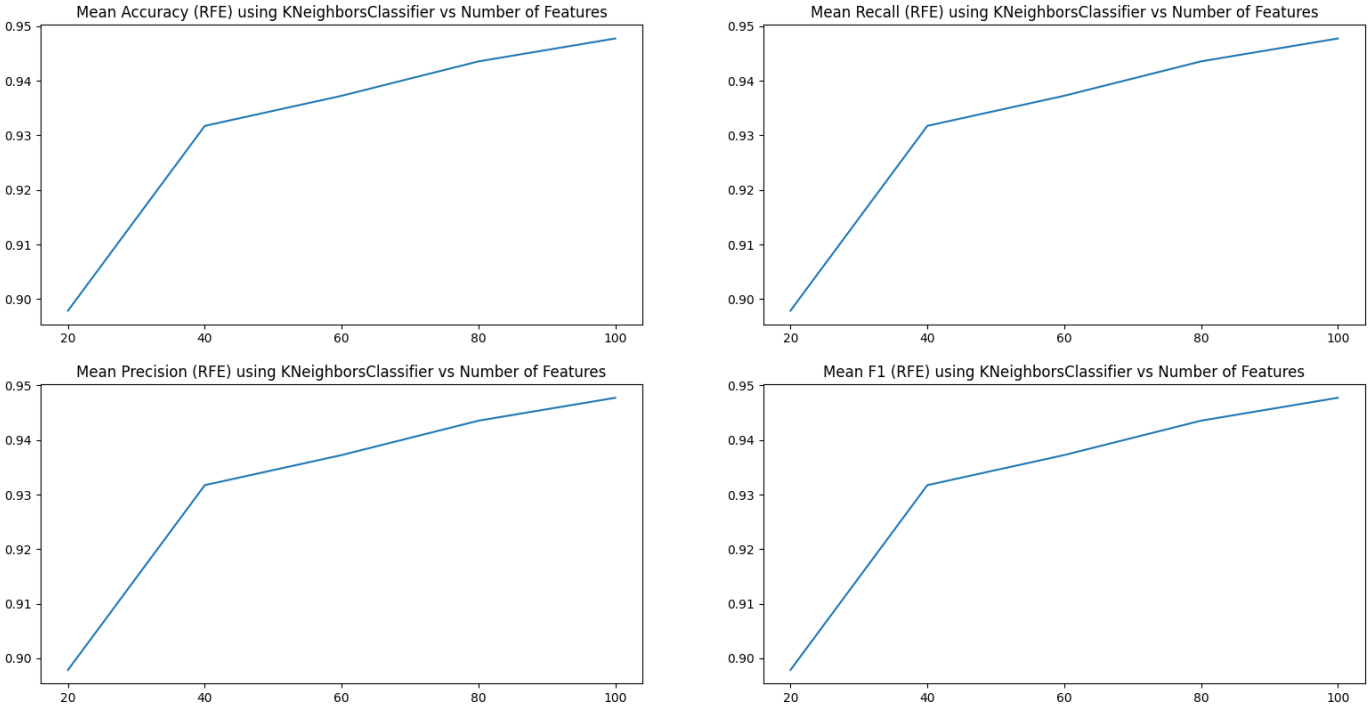


This figure is a really good example of how the clustering improves as the number of features increases. We can see a clear progression from iregular, choatic clustering to more organized grouped clusters.

Performance Metrics

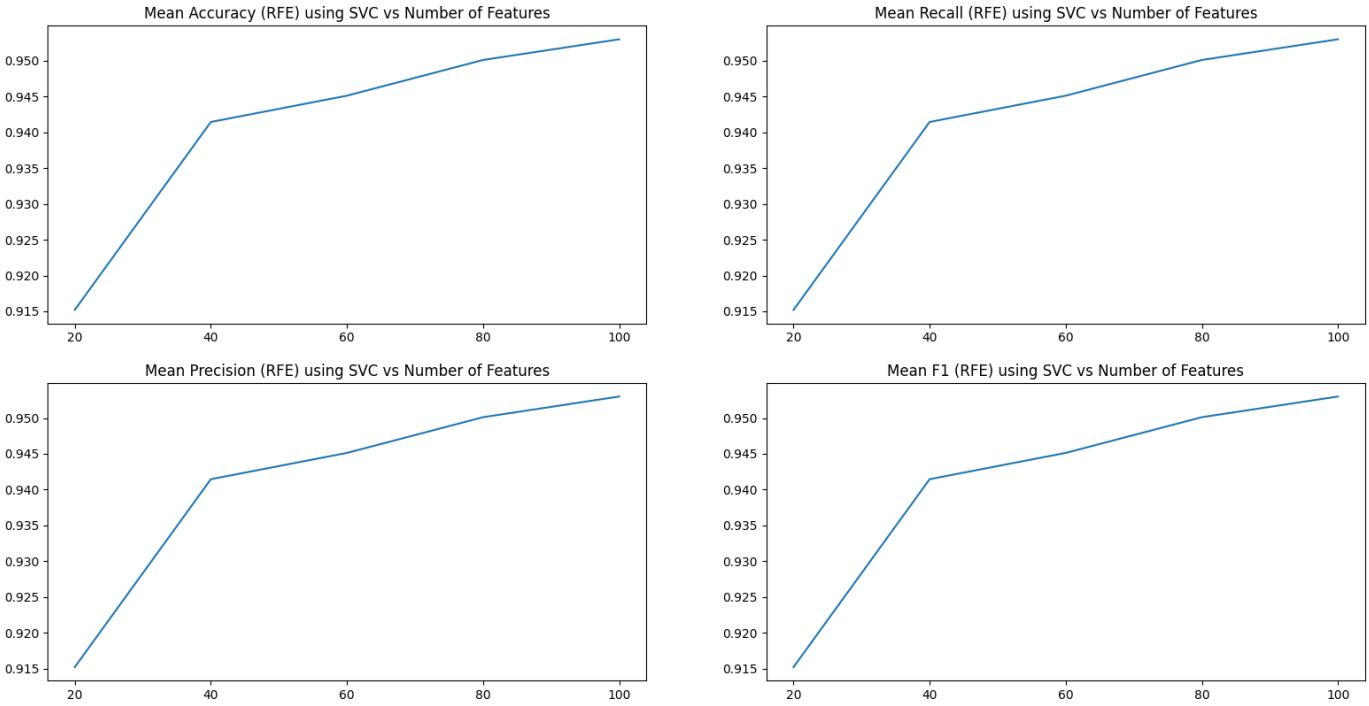
This section we will talk about how the performance metrics, average accurace, average recall, average precision, and average f1, change overtime with the number of features increase, using Recursive Feature Elimination for Feature Selection and is verified using Stratified K fold, with 5 folds on each run.

K-Nearest-Neighbor



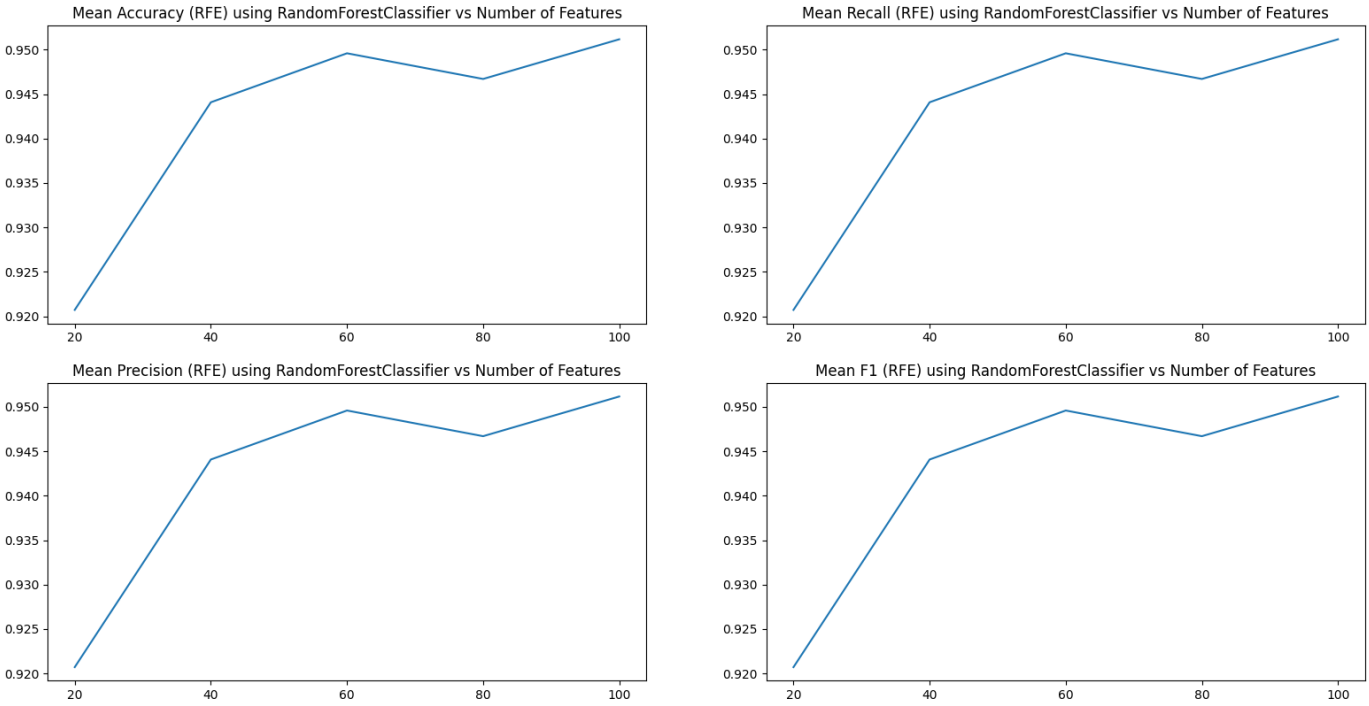
This figure shows that overall there is great increase in accuracy when the number of features increase, with maximum accuracy of 95% reached with 100 features, this result can be observed accross all metrics of accuracy, recall, precision, and f1.

Support Vector Machine



SVM results further validate the results obtained in KNN, as the mean accuracy, mean precision, mean recall, and mean f1 both increased in direct connection with the number of features selected from RFE.

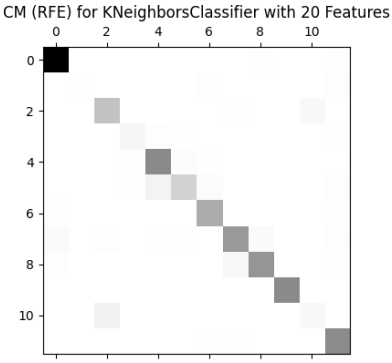
Random Forest

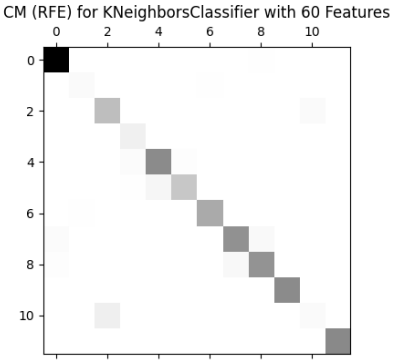


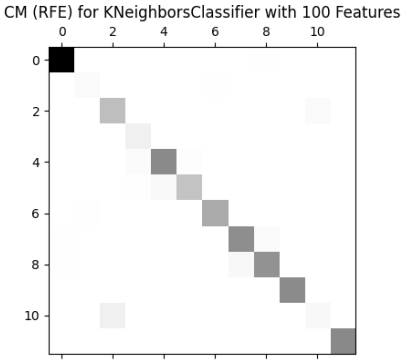
Random Forest as compared to KNN and SVM show a slightly diffrent image, with the accuracy, recall, precision and f1 both increasing with number of features, till around 60 features, where there is a slight decline, with another increase at 100 features.

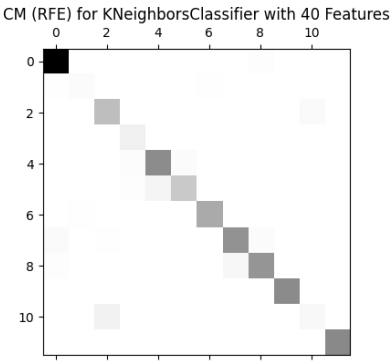
Confusion Matrix

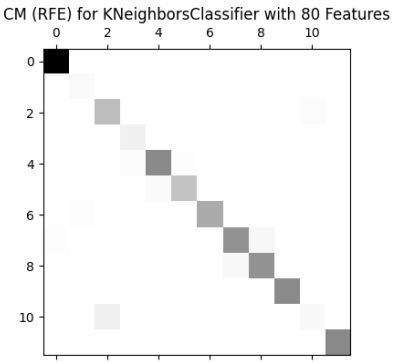
K-Nearest-Neighbor





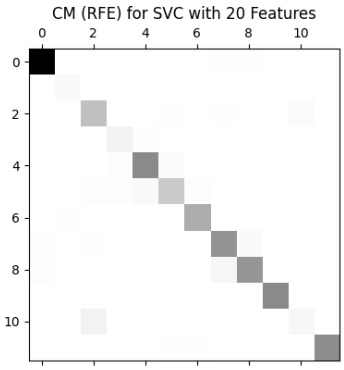


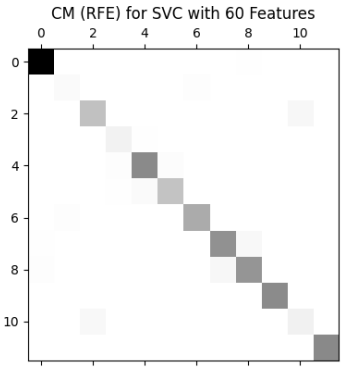


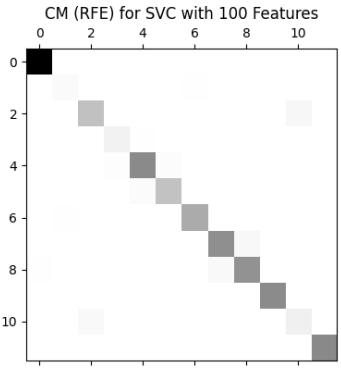


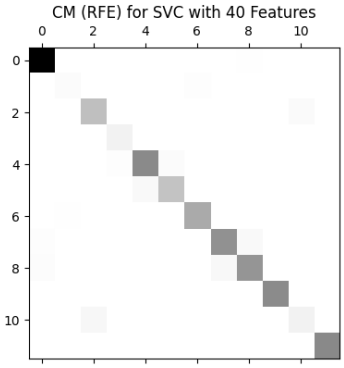
The confusion matrix with KNN shows as the number of features increase there is less divergance from the diagonal, indicating that the rates of missclassification are reduced as the number of features selected increase.

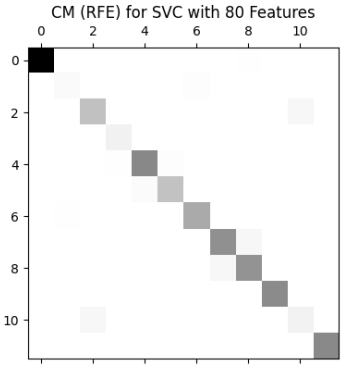
Support Vector Machine





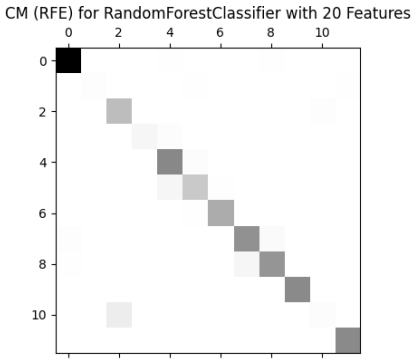


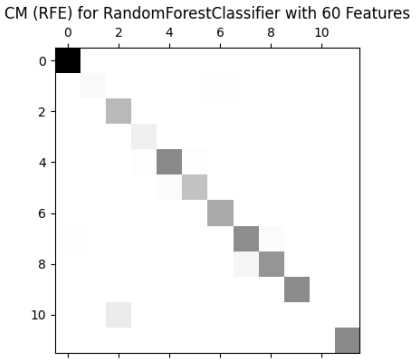


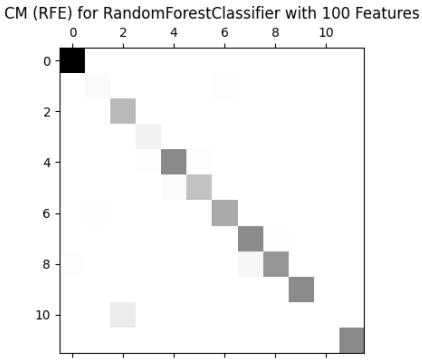


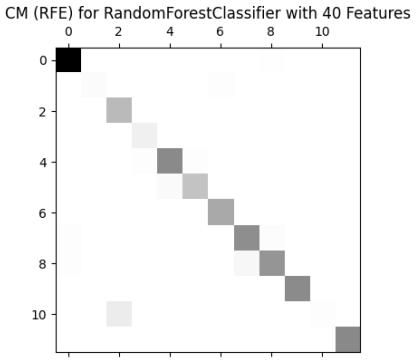
SVM validates the results from KNN, where as the number of features selected increase, the overall diversion decreses as the number of features goes up. with 100 features there are very minimal varience.

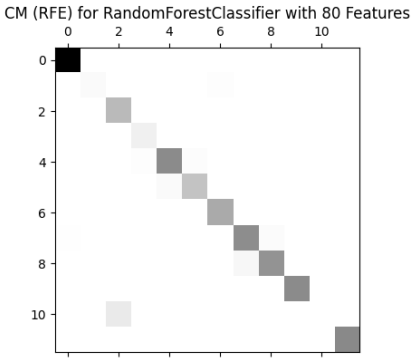
Random Forest







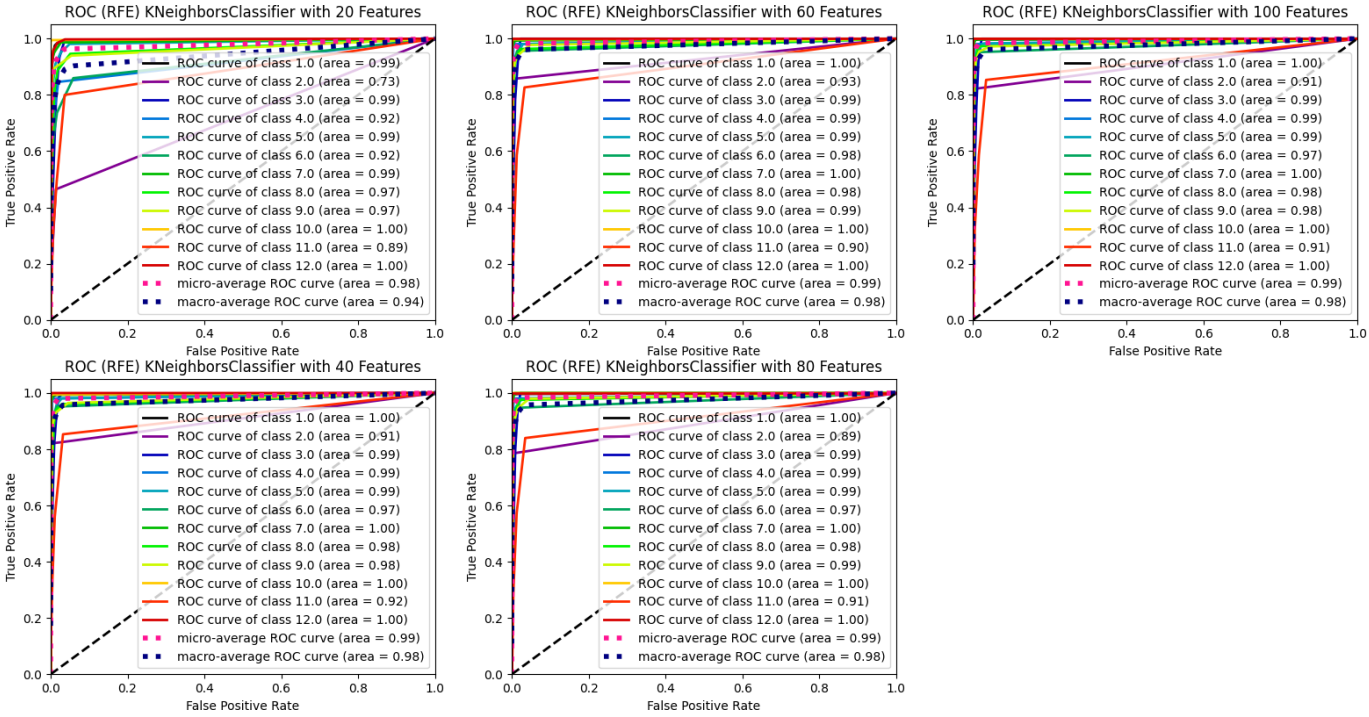




Random Forest, just like KNN, shows a progression from Confusion matriz with higher instances of divergance to a matrix with lower indices of misclassification.

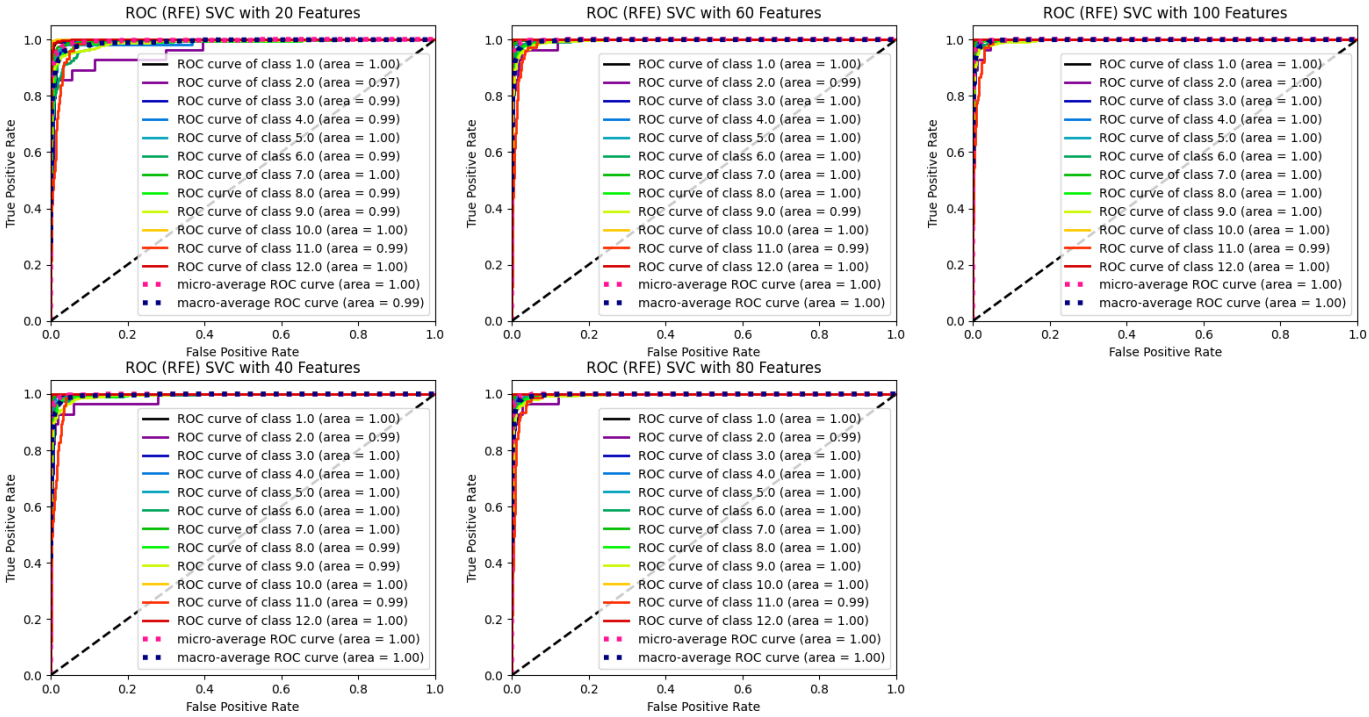
ROC Curves

K-Nearest-Neighbor



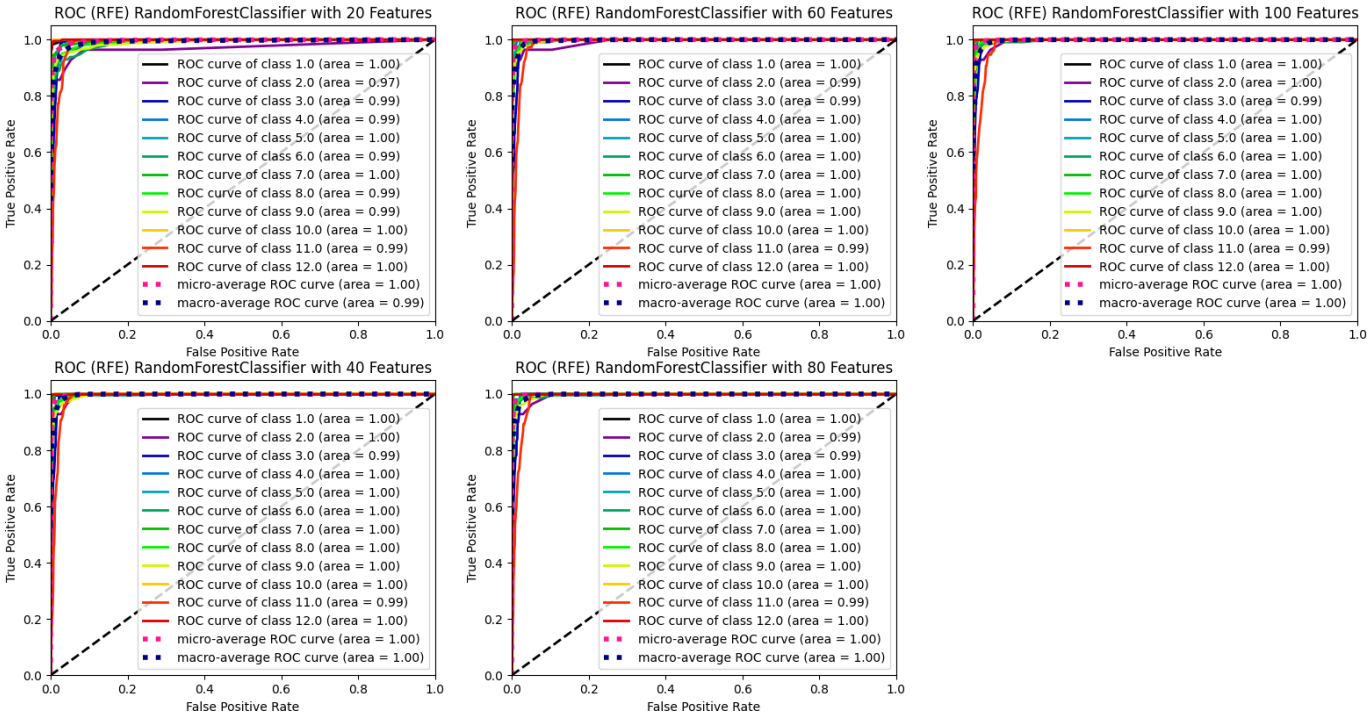
The ROC curve here shows that with lower number of features, some clases have higher indices of misclassification, in particular class 2 and 11, show a high degree of missclassification, while with higher number of features, they are indistinguishable from the other classes in therms of missclassification.

Support Vector Machine



When it comes to Support Vector machine, the missclassification is not as bad when compared to KNN, however, Class 2 still has a higher incidence rate with low number of features, but quicjly raises the the normal with more features.

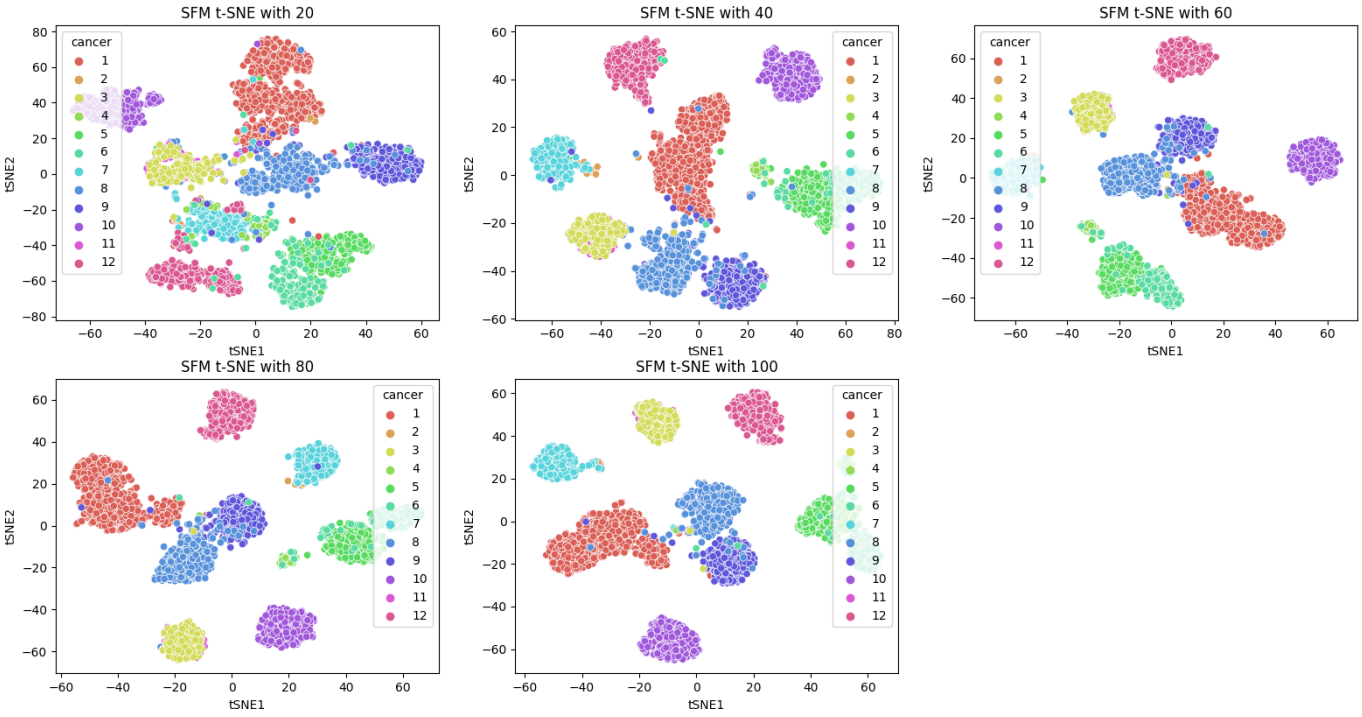
Random Forest



Random Forest in contrast showes very little diffrence in its ROC curves when the number of features changes, in general all classes are performing well in classification and True Positive Rates.

SelectFromModel

t-SNE

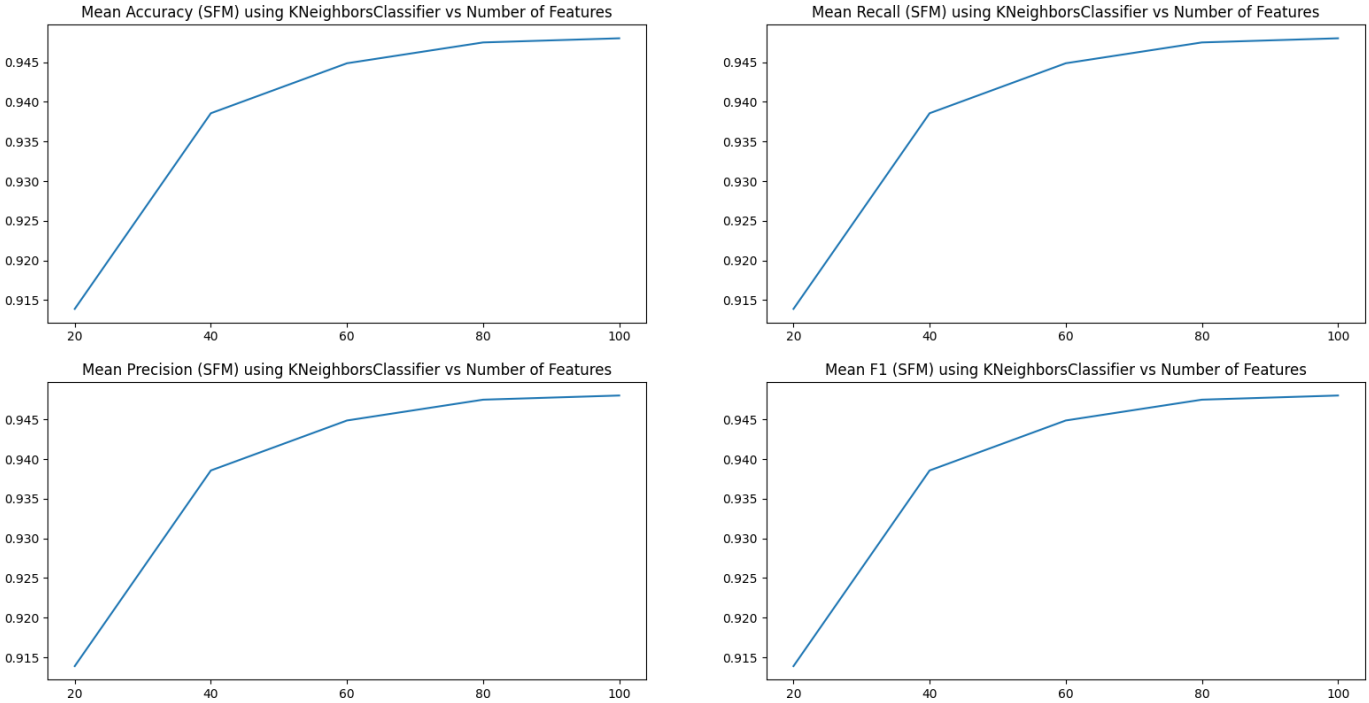


In this figure t-SNE is an ideal representation of how classes go from a very deeply coupled clustering to sparsely separated, tightly bound clusters of classes as the number of features increases, thus helping increase the overall accuracy of the algorithm.

Performance Metrics

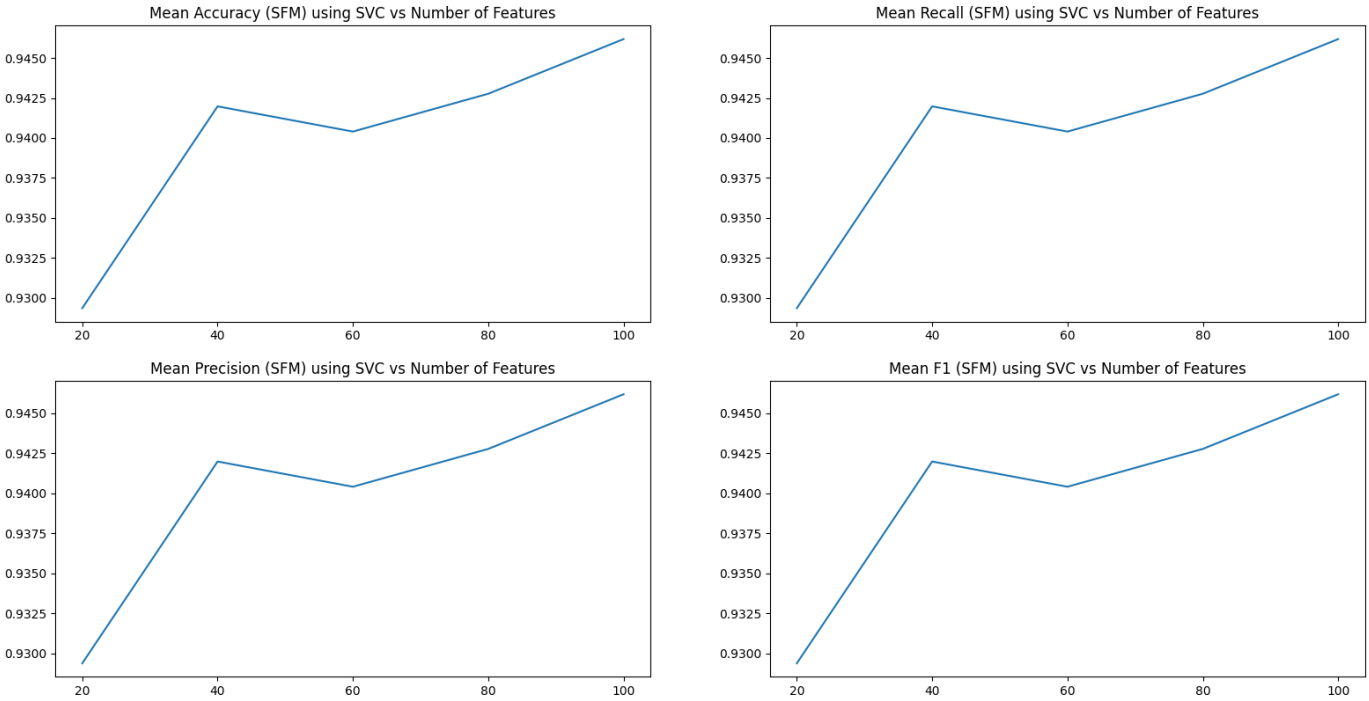
This section we will talk about how the performance metrics, average accurace, average recall, average precision, and average f1, change overtime with the number of features increase, using SelectFromModel for Feature Selection and is verified using Stratified K fold, with 5 folds on each run.

K-Nearest-Neighbor



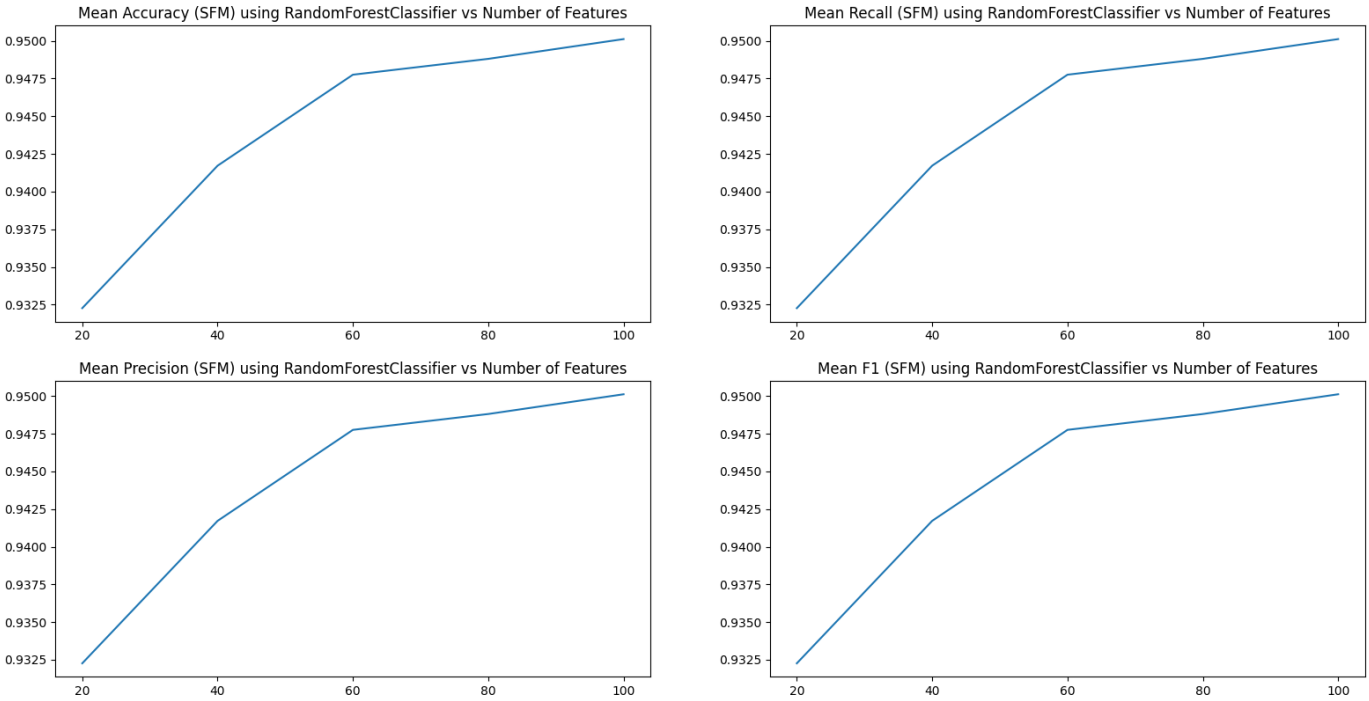
In this figure we can see how the accuracy quickly rices to 94% as the number of features selected using SelectFromModel increse, eventually hitting a platoe with only minimal improbements, for SVM with KNN, this appears to be around 40 Features.

Support Vector Machine



When it comes to Support Vector Machine, we see the standard increase in overall accuracy as the number of features increases, however after 4 features we actaully decrease the acuarcy slightly, with another improvement in average accuracy after 60 features.

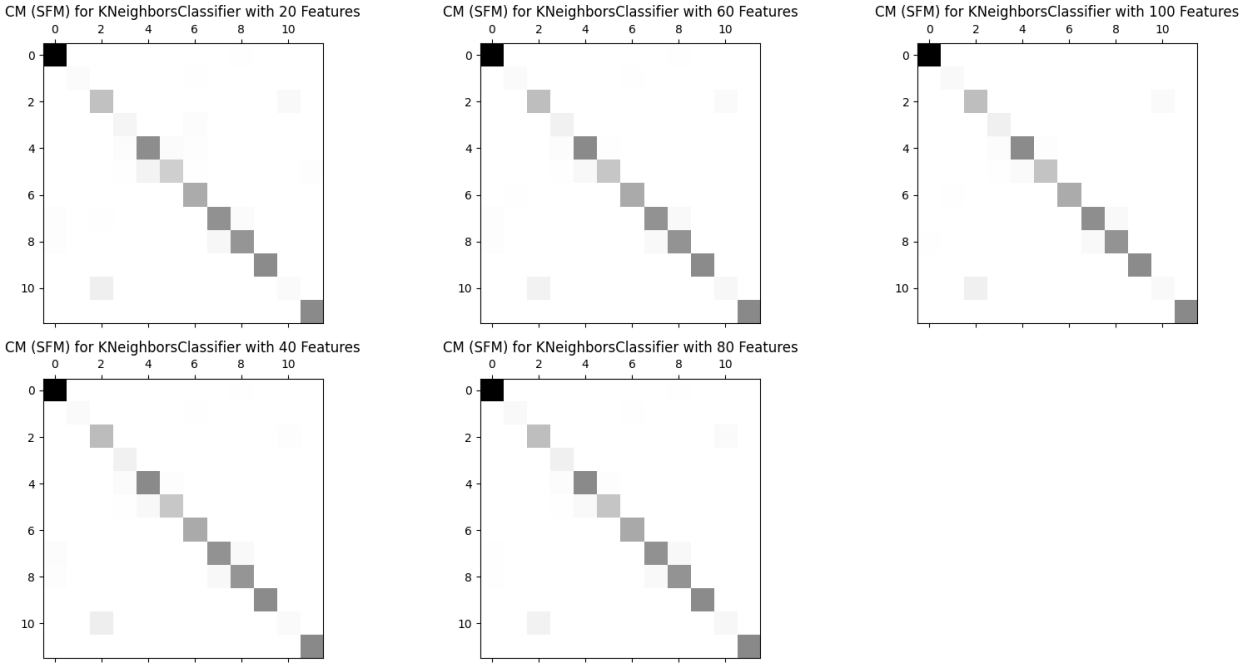
Random Forest



Random Forest, in contrast to Support Vector Machine, observes a constant increase in the overall metrics as the number of features increase. However diminishing returns are clearly observed after 80 features, where the persentage increase is very minimal.

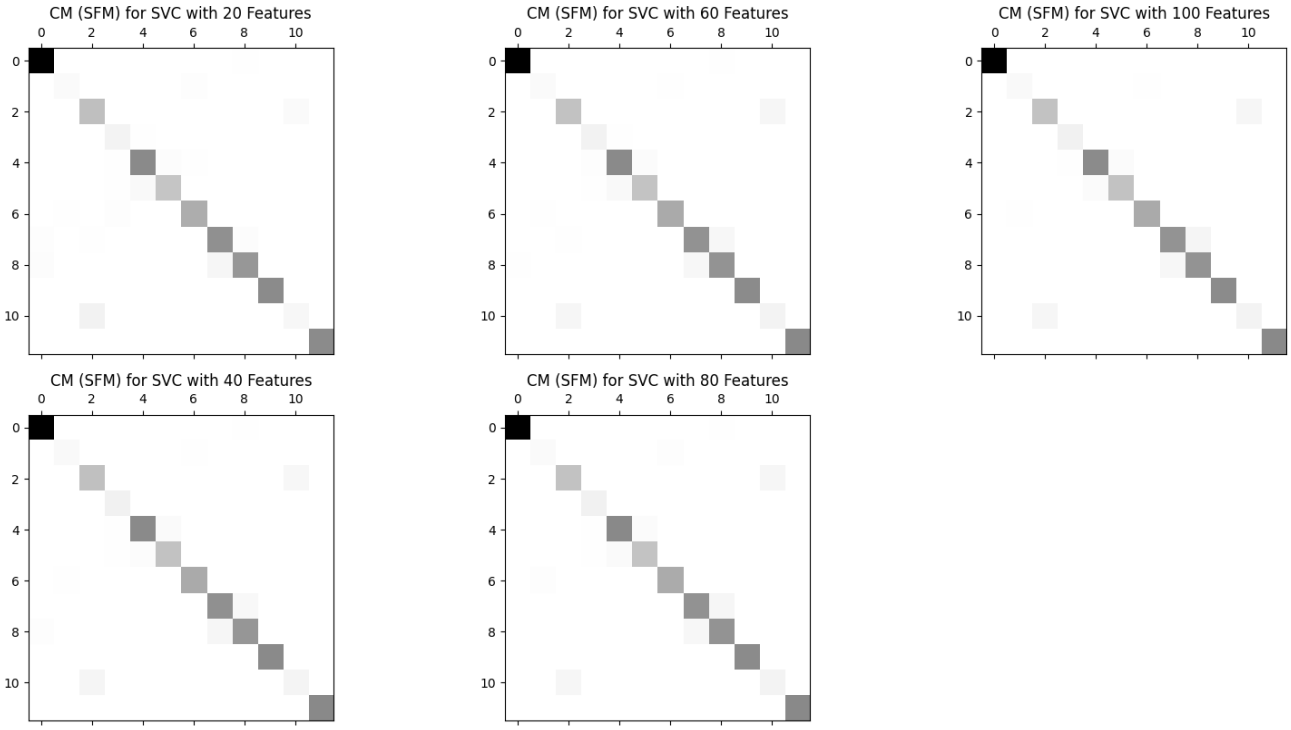
Confusion Matrix

K-Nearest Neighbor



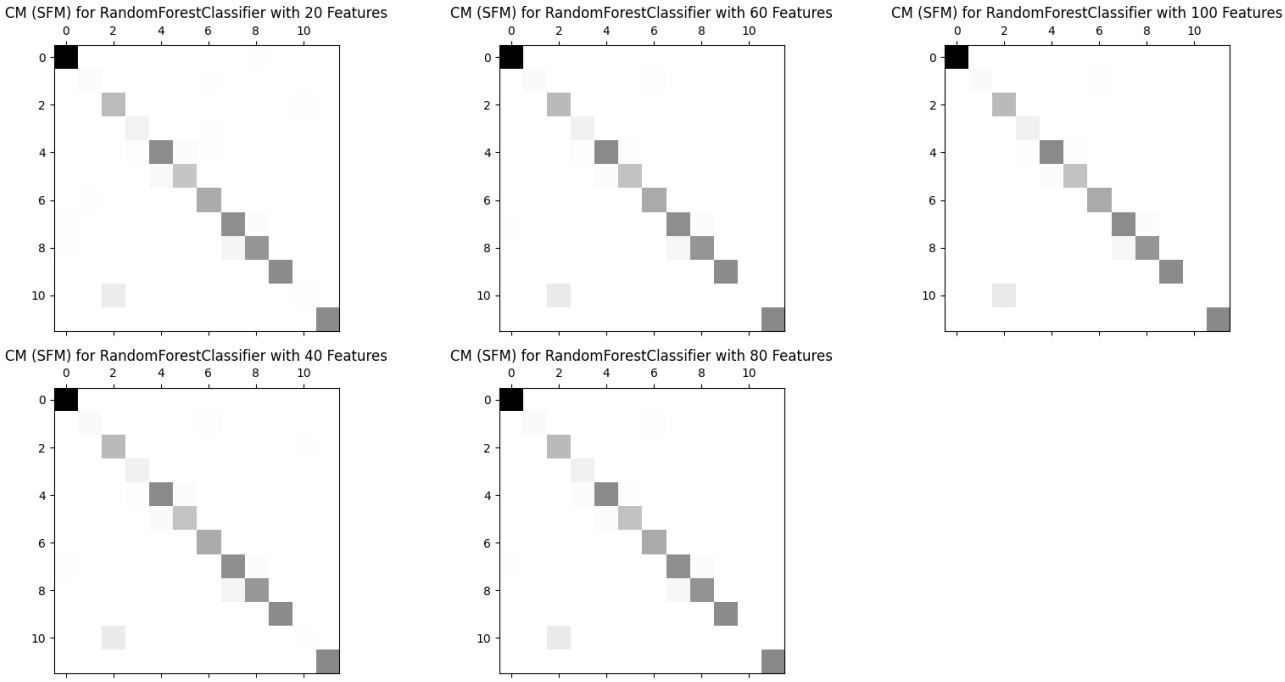
When it comes to KNN, we can see that overall there is a very well defined diagonal with very small diviations with all number of features, after all, even with 20 features, the accuracy is over 90%, howver we we increase the number of features, we do see some small reduction in the total number of diviations observed, specially when comparing class 2 being misscategorized as class 10 with 20 number of features, and the occurance of this decreasing as the number of features increase.

Support Vector Machine



In the Support Vector machine, just like KNN, there is a very well defined diagonal, indication good accuracy in the model, however there are still some diversions, speciall with class 2, sometimes being classified as class 11, as the numbers of features increases, this does improve, however it is not completelt eleminated from the model.

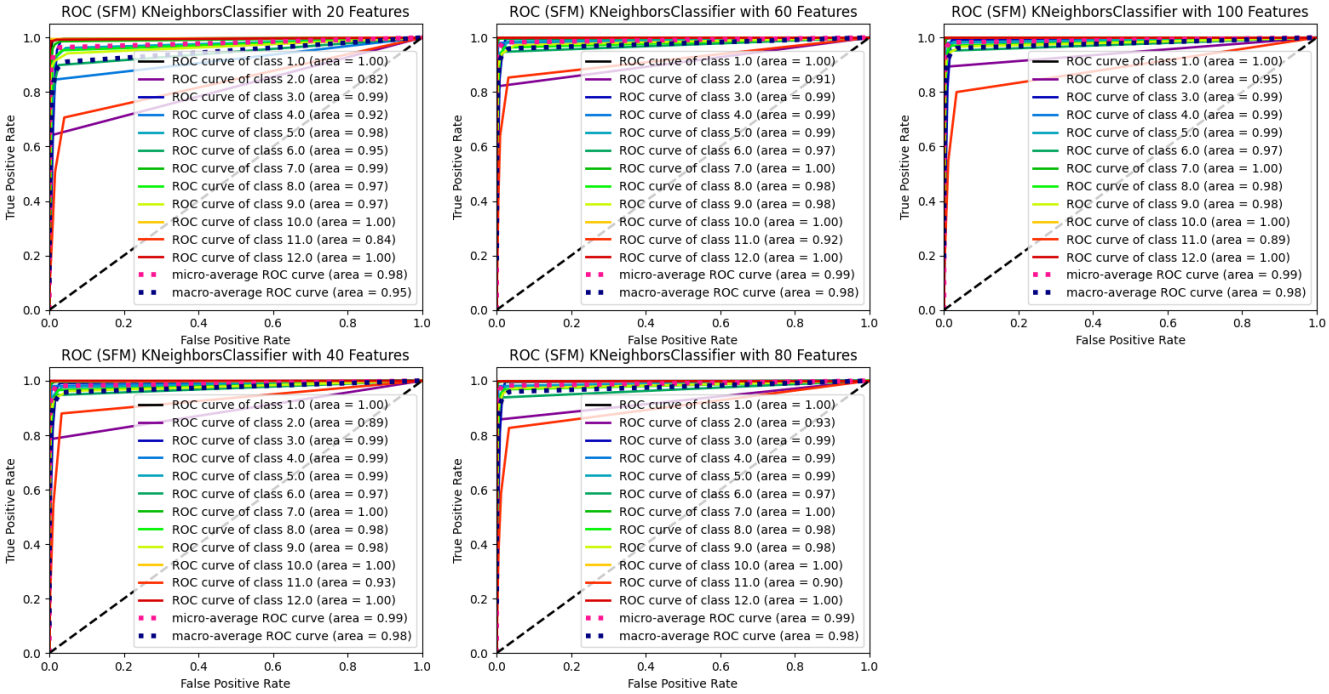
Random Forest



Random Forest Classifier, shows a similar response in the confusion matrix, a very well defined diagonal, with some slight missclassification, just like with Support Vector Machine, the model some times missclissifies class 2 as class 11, the occurance does decrease as the number of features increases, but the behavior is not completely eliminated from the model.

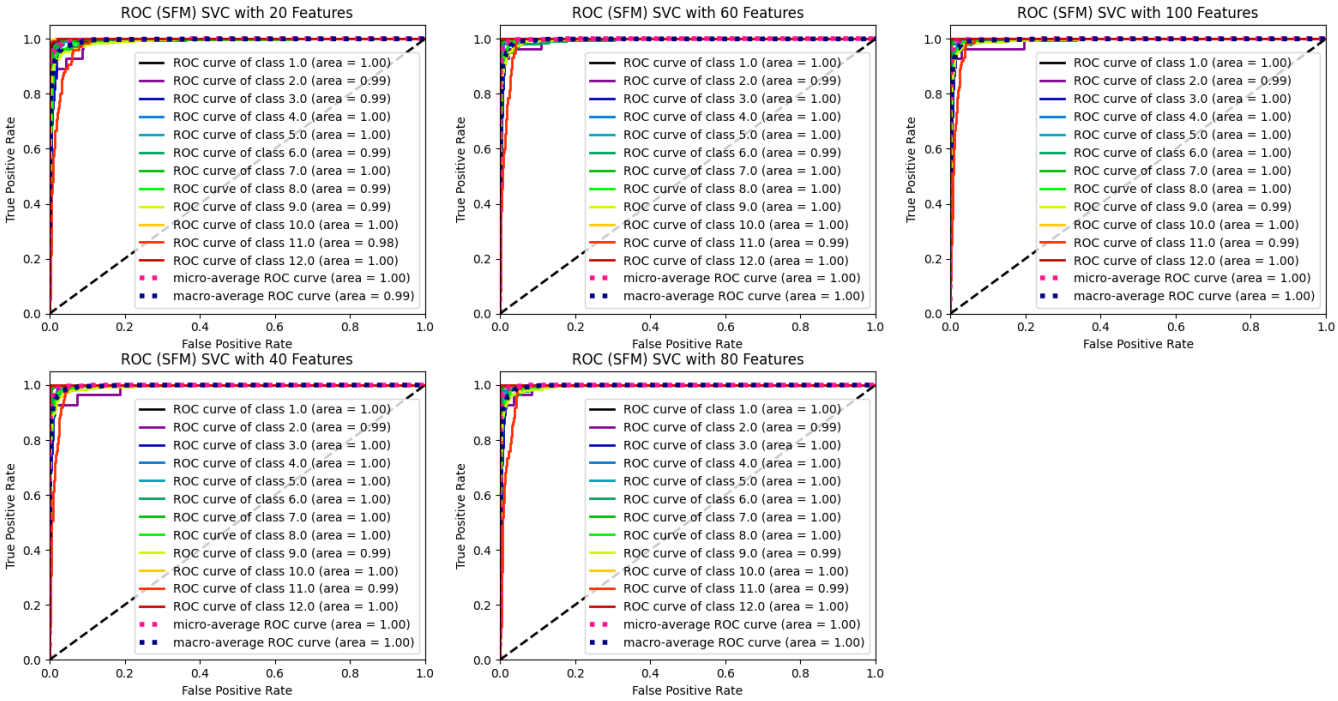
ROC Curves

K-Nearest Neighbor



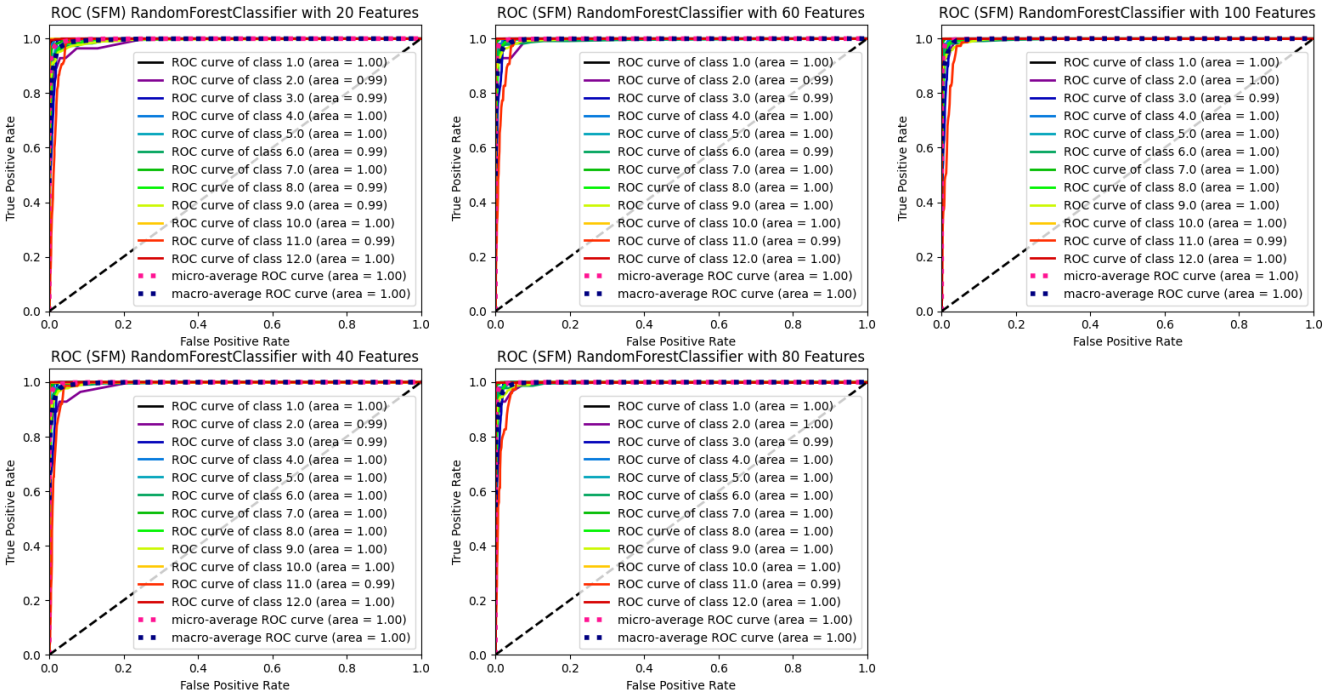
The ROC curves for KNN, shows the same behavior as seen in the confusion matrix, most classes have a very high True Positive Rate, however, classes 2 and 11 have a lower True Positive Rates (indicated with lower area under curve). As the number of features increases, we can see how the overall true positive rate increases for both classes, while not entirely reaching the same as the other classes, the overall area under the curve improves significantly.

Suppport Vector Machine



In Support Vector Machine, the ROC curves show a very good true positive rates for all classes, however class 2 and 11 still have the lowest area under the curve values, indicating that they still have issues with missclassication, although to a lower degree.

Random Forest



Just like Support Vector machine the ROC curve seems quite homogenious, with all classes performing well. Once more class 2 and 11 have some of the lowest scores in low feature counts, however class 2 reaches as high as 1 for area under the curve when the number of features increase.

Conclusion & Future Remarks

In general, two main conclusions can be drawn from the resuts of the project. First, the cancer types that have the least number of samples are the ones that showed worst clustering results during the tSNE phase. These were usually clustered on bellow other cancer types with more samples. Second, there is a clear relation between the number of features and the performance of the classifiers. More sparcity can be appreciated in the confusion matrix, as well as poorer clustering, when the number of features decreases, speacially obvious with the LASSO model. Here, the clustering for 512 shows clear distinctions between the cancer types, while with 9 features there is significant overlapping in the center.

Finally, we can conclude that the best pair of model/number of features is: LASSO with 512 features. However, it is the belieave of the team, that the number of features is the main reason for the better results. LASSO would have to be compared with the other feature selection models using the same number of features in order to more meaningfully asses their diffrenece in performance.

Between Recursive Feature Elimination and SelectFromModel there is no significant difference in performance (0.05 for mean accuracy, recall, F1 and precision in the classifers). Moreover, it is interesting to see how the mean accuracy decreases for 60 features in SVM (against 40 features) and 80 features in RF (against 60 features) for RFE and SFM respectively. However, the performace matrix and tSNE results are similar. Finally, 100 features yields the best results for both feature selection models.

References

- http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectFromModel.html
- https://www.analyticsvidhya.com/blog/2020/05/7-scikit-learn-hacks-tips-tricks/
- https://analyticsindiamag.com/guide-to-dimensionality-reduction-with-recursive-feature-elimination/
- https://www.titanwolf.org/Network/q/8f9189c7-fb76-4628-9707-822e8074ebfc/x
- https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761
- https://learn.g2.com/k-nearest-neighbor
- https://www.kdnuggets.com/2020/04/introduction-k-nearest-neighbour-algorithm-using-examples.html