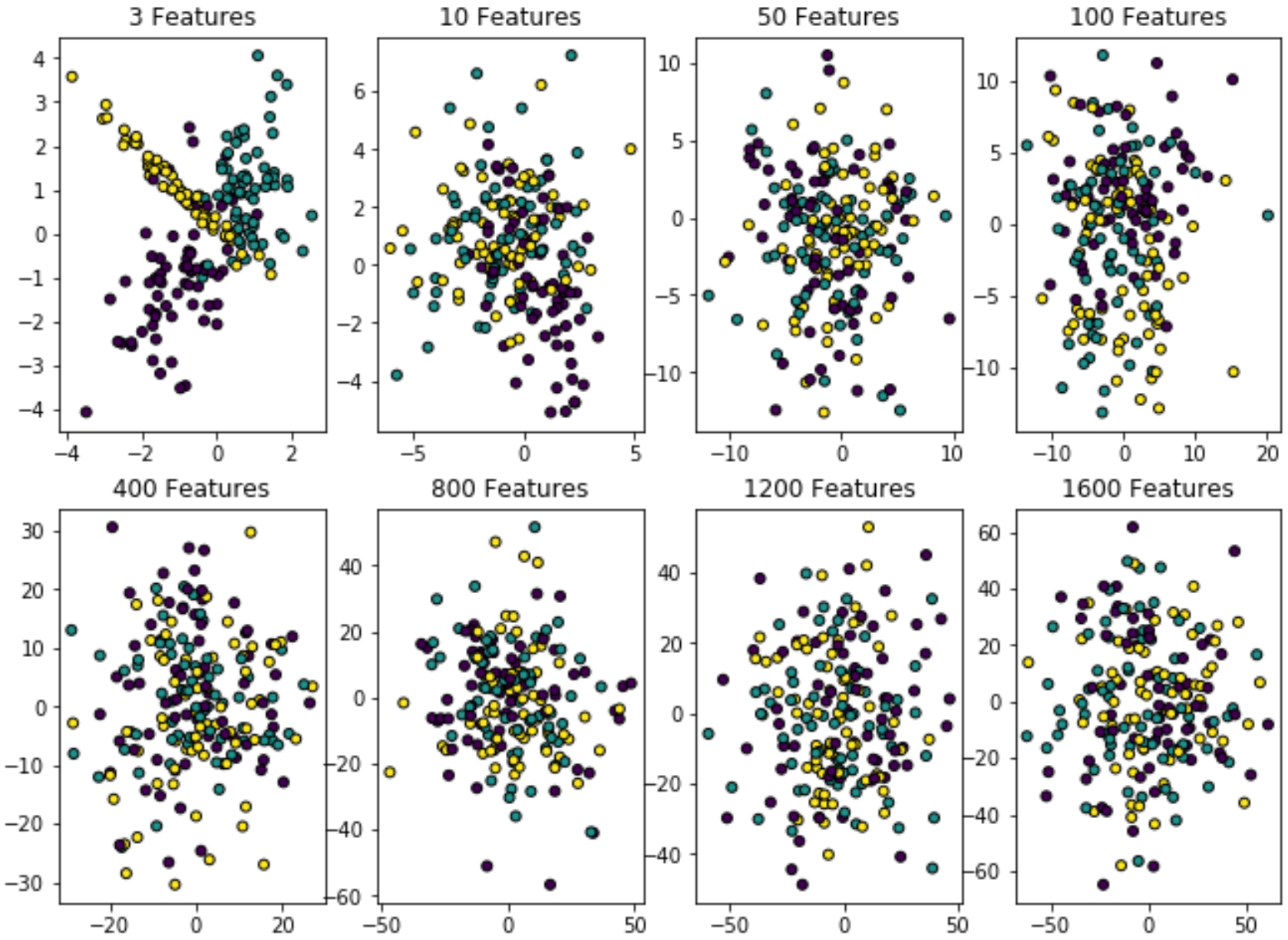
Classifiers in Increasing Dimensions

Camille Porter

Data Generation

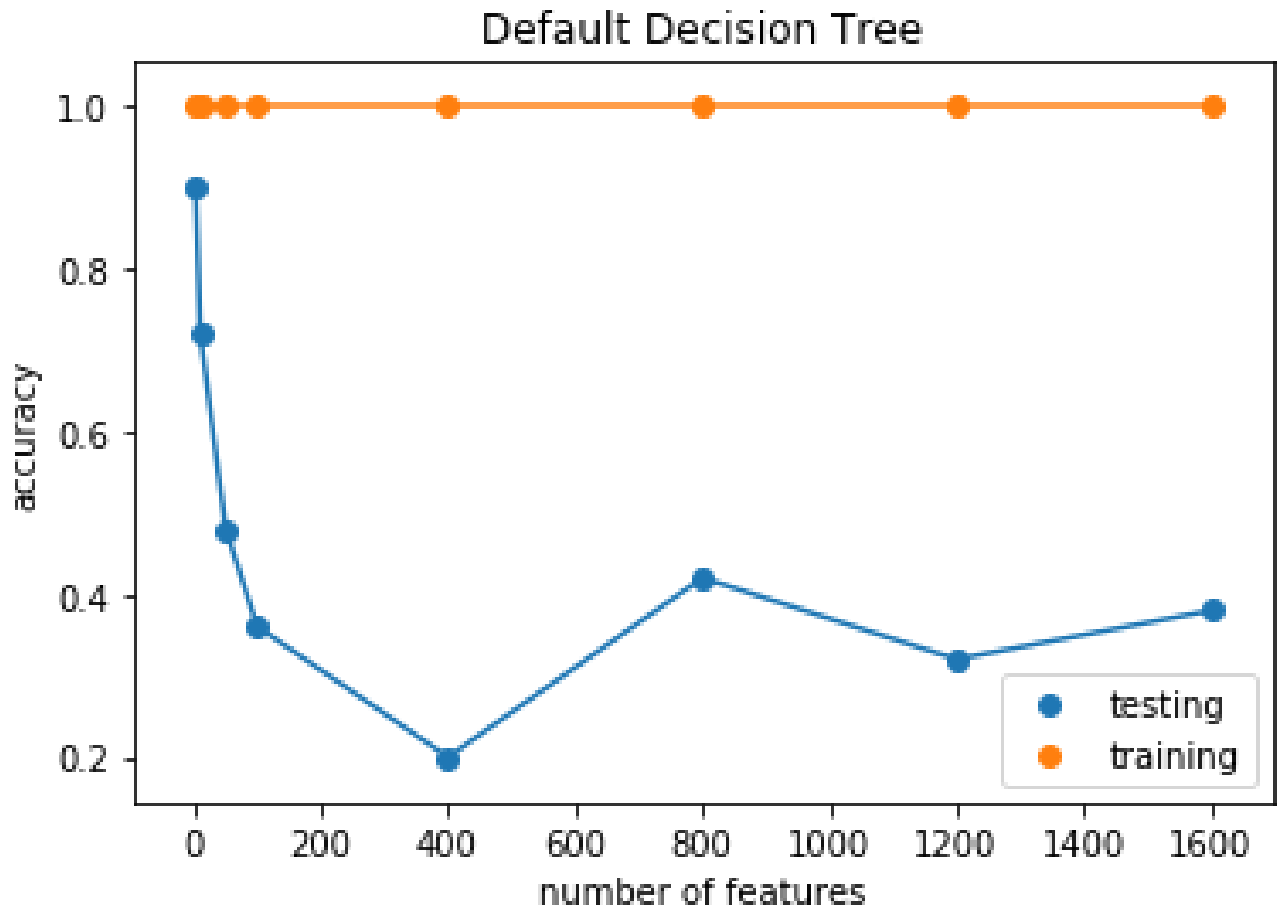
- For this project, I generated 8 different datasets using the make classification function in scipy
- I used 1 cluster per class, only informative features, non-repeated features, 3 different classes, and 200 samples
- The clusters are drawn independently from N(0,1) and placed on the vertices of a hypercube.
- The number of features increases from 3 to 1600.
- Training data is 75% of the dataset



- The axes for these plots are the first two features in the dataset. Colors are assigned by class.
- It seems possible to visually separate the classes when the number of features is small, but as the number of features increases, it is difficult visually separate the classes.
- As features are added, the dimensionality of the feature space grows and becomes more sparse

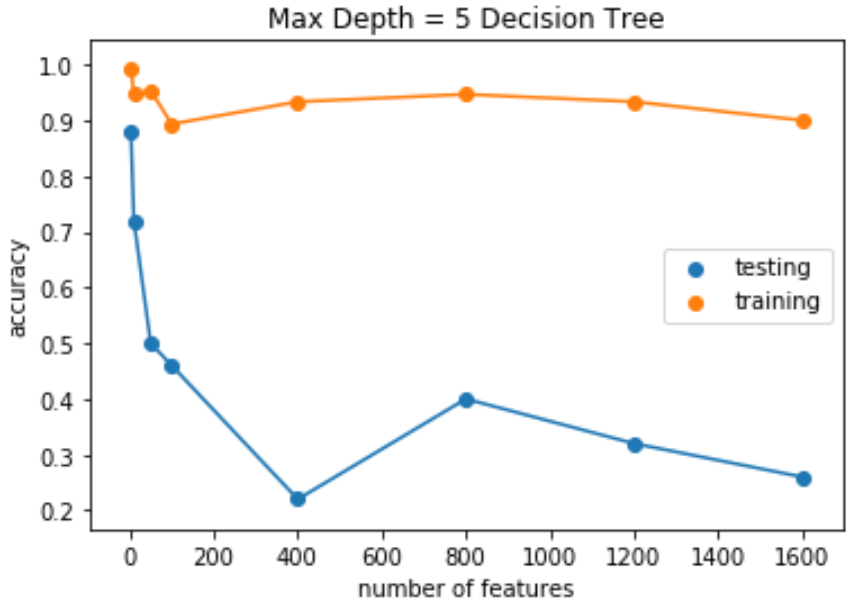
Decision Tree

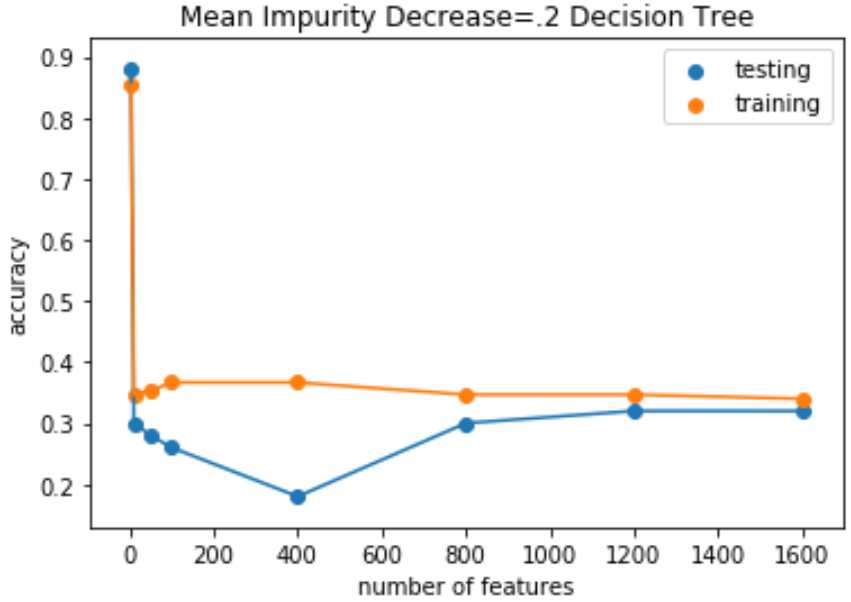
- The accuracy is good when the number of features is small
- When the number of features is greater than the number of samples (150), the accuracy is very low
- The models look very overfit. The training data has an almost perfect score, while the score of the testing data drops dramatically.
- Perhaps restricting complexity will decrease the amount of over training



Decision Tree

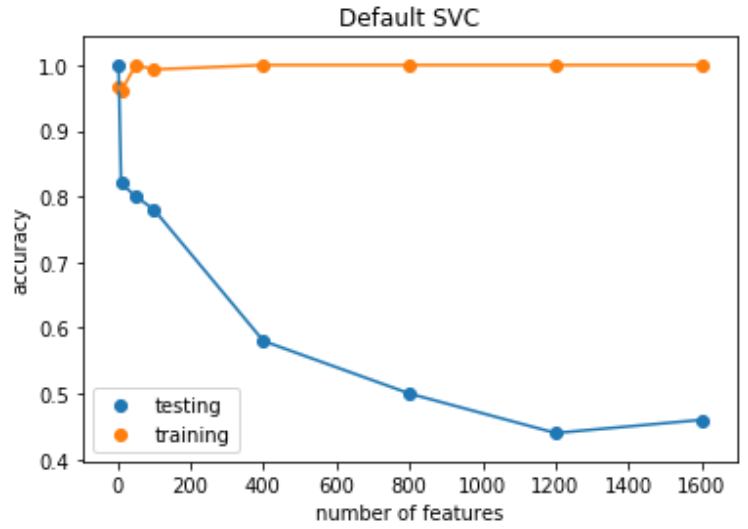
- Changing parameters that limit tree complexity (depth, mean impurity decrease) does not improve the accuracy.
- Decision Tree does not generalize well and is prone to over-fitting, so it is not a good method for many dimensions.





SVC

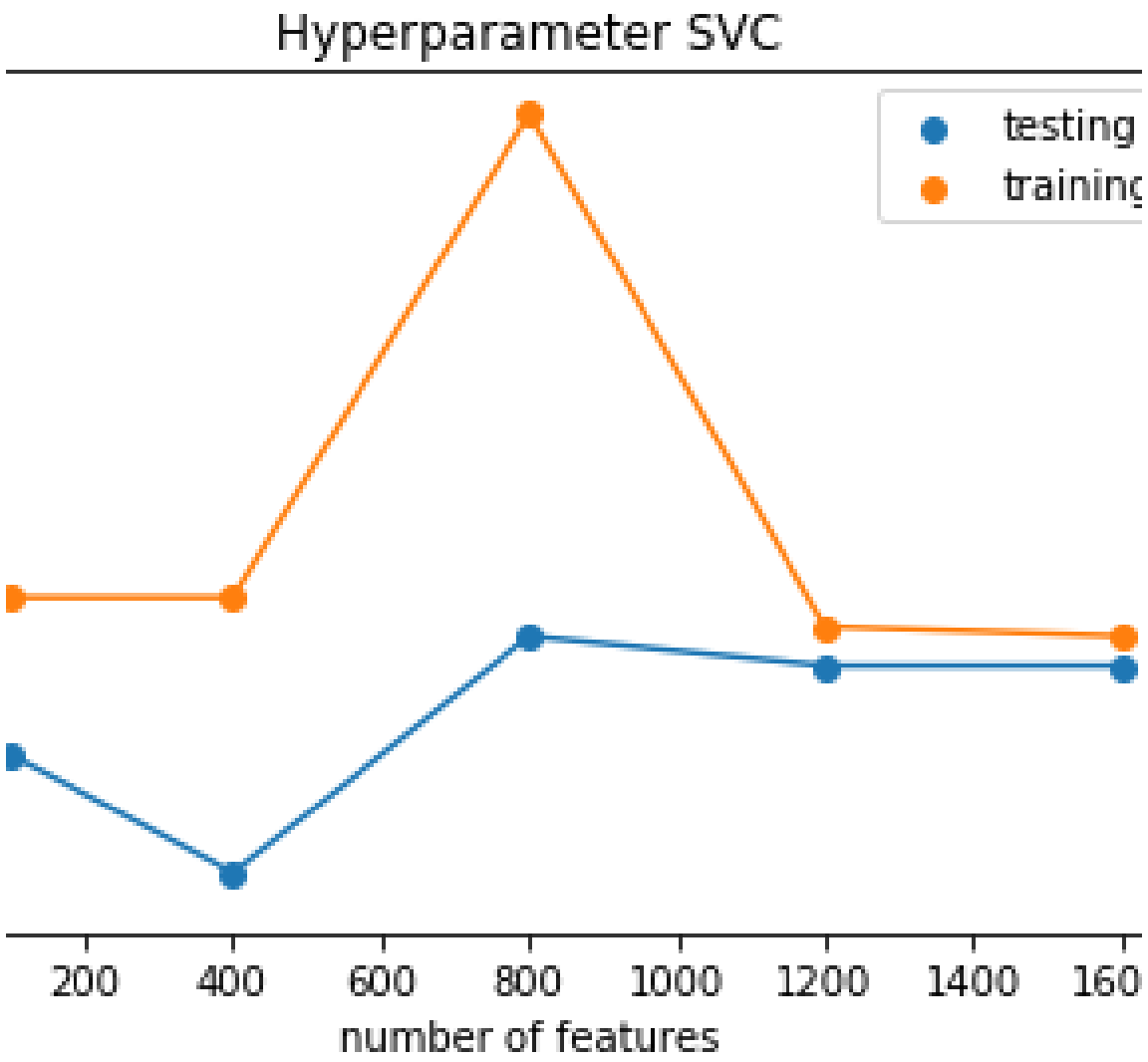
- SVC is supposed to be resistant to overtraining, but it needs hyperparameter tuning.
- The accuracy decreases as the number of features increases.
- The 1200 and 1600 feature models are becoming baseline predictors, i.e. only predicting 1 class
- This is the precision score for the 1600 feature model. The two worst performing models (1200 and 1600) are assigning all the samples to one class.
- Perhaps we can improve this by tuning the parameters.



	precision	recall	f1-score	support
0	0.32	1.00	0.48	16
1	0.00	0.00	0.00	18
2	0.00	0.00	0.00	16
accuracy			0.32	50
macro avg	0.11	0.33	0.16	50
weighted avg	0.10	0.32	0.16	50

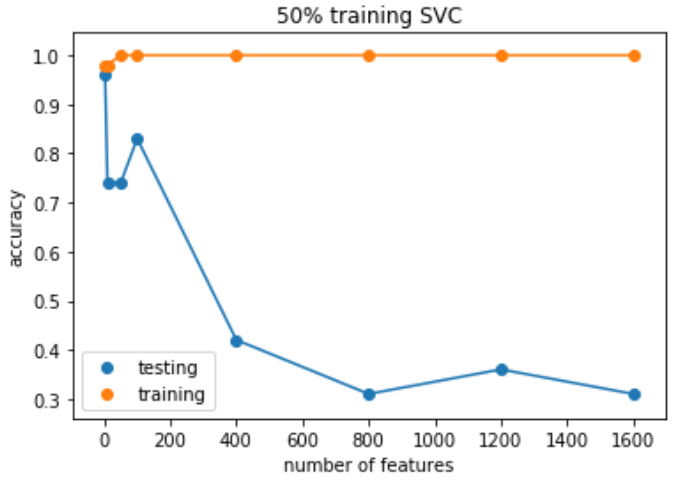
SVC

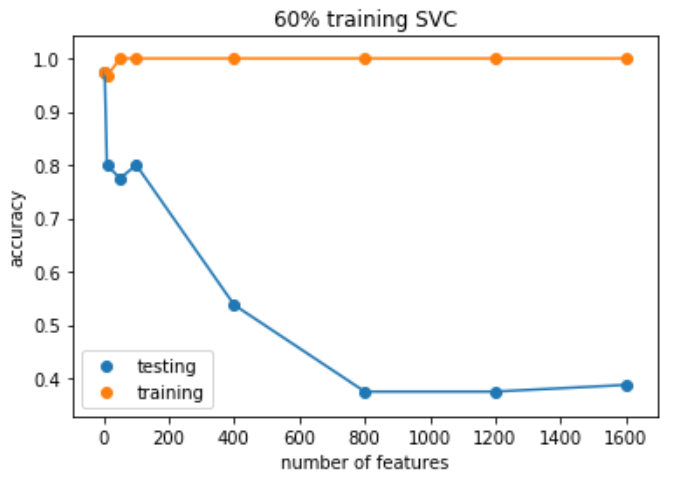
- Each model has different optimal hyperparamters.
- Even with optimizing hyperparameters, the two largest models only predict 1 class.
- This is supposed to optimize the 1600 feature model ('C': 0.1, 'gamma': 0.0001, 'kernel': 'rbf'), but the accuracy for the model does not increase

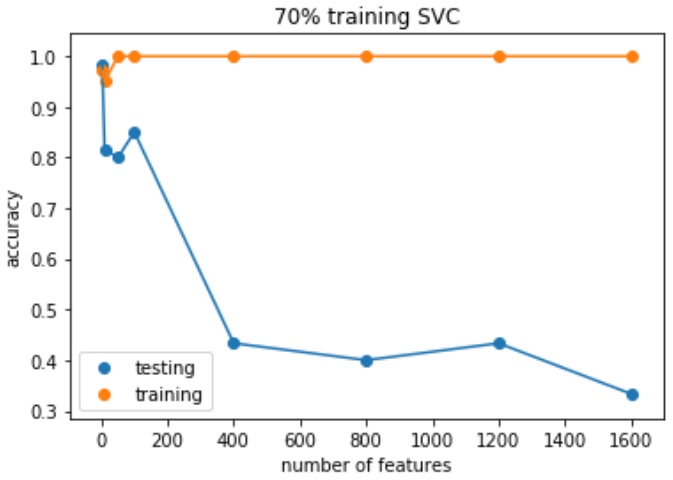


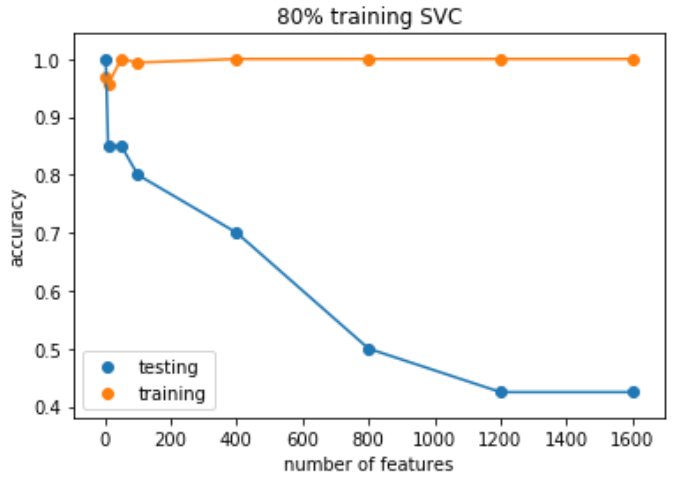
% Training Data

- The accuracy is improved by including a larger number of samples in the dataset.
- Even a small increase improves the accuracy.
- 50% - 100 samples
- 60% - 120 samples
- 70% - 140 samples
- 80% - 160 samples









Conclusions

- Classifier accuracy decreases as the number of features increases. The models become over trained very quickly. It is not easy to fix.
- The dataset becomes sparse as the number of features increases, and more training samples are needed
- Including as many samples as possible should help. Using cross validation instead of accuracy should also help increase the amount of training data.
- Genome studies often have millions of features and thousands of samples. They will have problems with this.