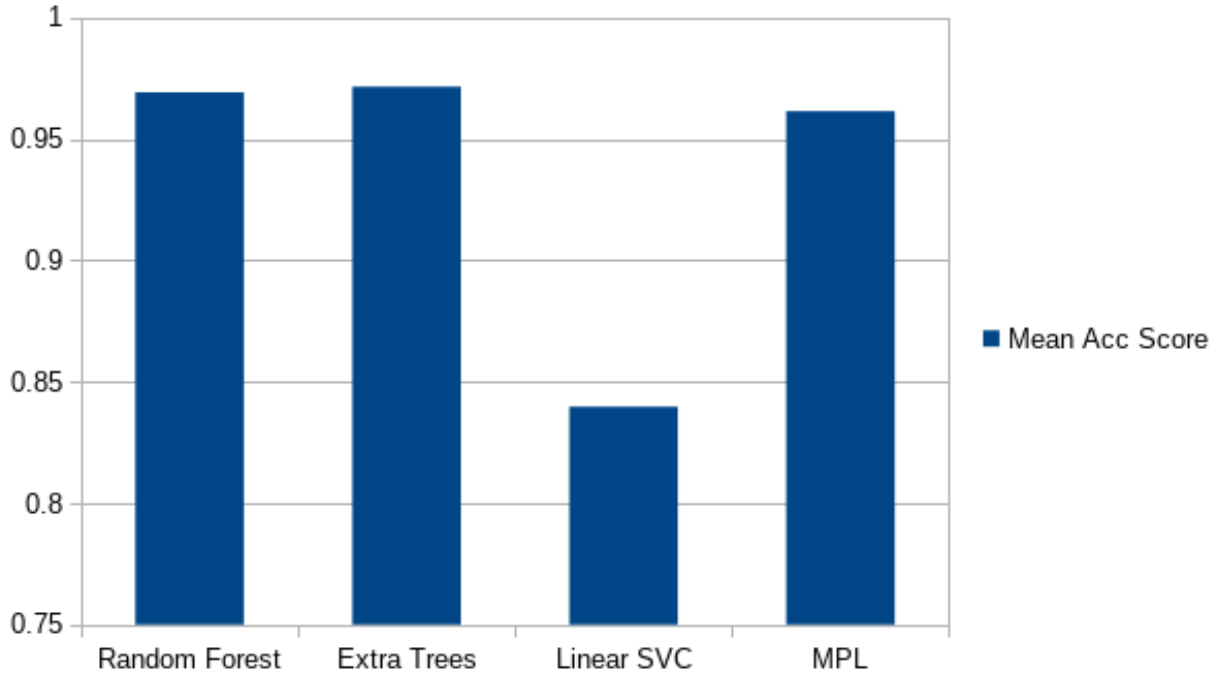
Ensemble Methods

In this exercise, we explored Ensemble methods by using the MNIST dataset. In particular, we ran a variety of classifiers (RandomForest, ExtraTrees, LinearSVC, and MLP) and looked at the results of those classifiers individually. Furthermore, we then combined those classifiers into an ensemble by using a Voting Classifier. The results of both the classifiers individually are shown below. Later on, we took a look at XGBoost to compare it to the others we've run.

Individual Models in Ensemble

Classifier Model	Mean Accuracy Score	Mean Accuracy Score on Test Set	Avg Recall	Avg Precision
Random Forest	0.9692	0.9645	.96	.96
Extra Trees	0.9715	0.9691	.97	.97
Linear SVC	0.8397	0.8449	.87	.85
MPL	0.9614	0.9607	.96	.96

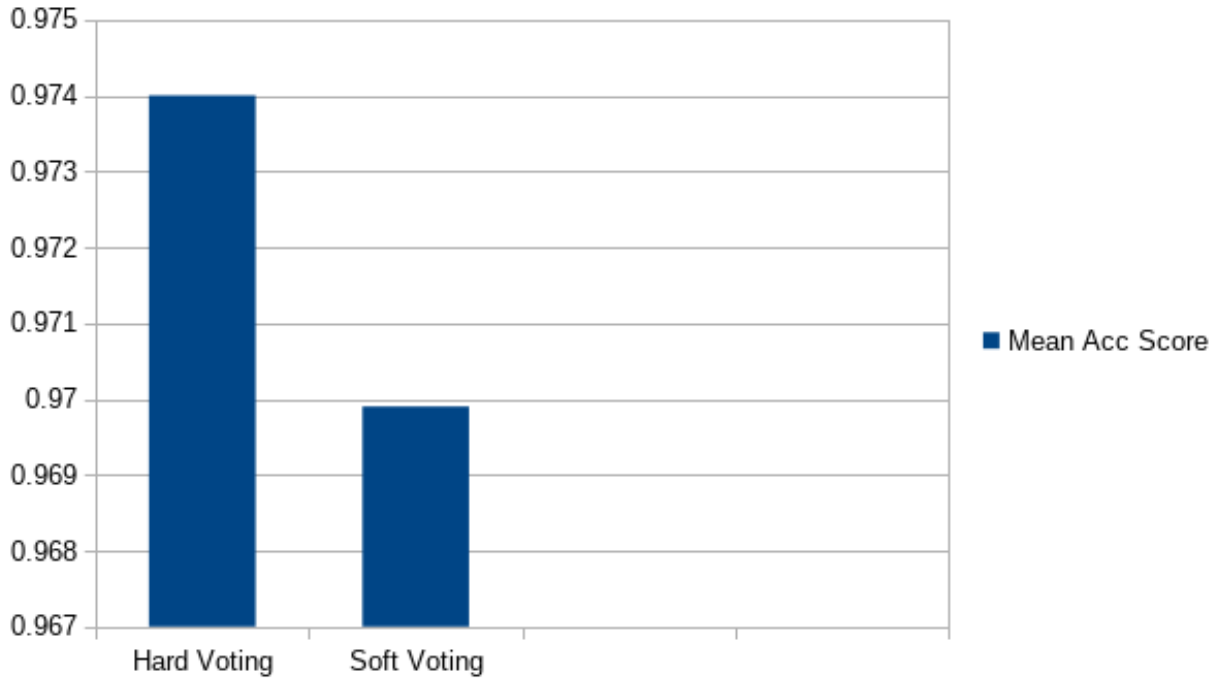


From looking at the individual scores, we can see Linear SVC clearly sticks out as faring much worse than the others. What is interesting is that if we look at the Linear SVC model's "classification_report" in scikit-learn, we can see that the precision and recall are both significantly lower than the other models, hovering around ~.85 for both. However, the other three models are all very high at around .96-.98 for both average precision and recall scores.

Voting Classifier Ensemble

Classifier Model	Mean Accuracy Score	Mean Accuracy Score on Test Set	Avg Recall	Avg Precision
Voting Classifier (hard voting)	0.9724	0.9691	.97	.97

Classifier Model	Mean Accuracy Score	Mean Accuracy Score on Test Set	Avg Recall	Avg Precision
Voting Classifier w/o Linear SVC (hard voting)	0.974	0.9703	.97	.97
Voting Classifier w/o Linear SVC (soft voting)	0.9699	0.968	.97	.97



From these scores, we can see that the Voting Classifier scored higher than all the other classifiers individually. Furthermore when the Linear SVC is removed from the list used in the Voting Classifier, it ends up with a slightly higher score at 0.974. Why is that? Well if we go to the scikit-learn documentation on the Voting Classifier, it says, "Such a classifier can be useful for a set of equally well performing models in order to balance out their individual weaknesses." Since the SVM had a significantly lower score, it appears that could be the reason it is hurting performance.

This Voting Classifier is trained using the list of other models we've inputted into it. It aggregates the findings of each of the individual classifiers and is able to predict the output based on the highest majority of the voting.

By default in scikit-learn, the Voting Classifier uses 'hard' voting. In hard voting, the predicted output is the class with a simple highest majority of votes. This differs from soft voting where the output class is based on the average of the probability that the class has. When we switch to soft voting for our Voting Classifier without Linear SVC, we get a mean accuracy score of 0.9699, which is worse than when we used hard voting.

XGBoost Classifier

Classifier Model	Mean Accuracy Score	Mean Accuracy Score on Test Set	Avg Recall	Avg Precision
XGBoost	0.9777	0.9739	.97	.97

XGBoost is an implementation of gradient boosted decision trees. Now boosting is based on weak learners, which means a combination of high bias and low variance. With weak learners in terms of decision trees, it means we have shallow trees, possibly even decision stumps. From our results, we can see that XGBoost ended up faring the best on this dataset that we used, even better than our Voting Classifier that was an ensemble of our other models, whether on hard or soft voting. However if, for fun, we train the Voting Classifier again without Linear SVC but with XGBoost, we get a mean accuracy score of 0.9775, which is still lower than XGBoost on its own.

In comparison, while both are ensemble methods, Random Forests uses fully grown decision trees, which means a combination of low bias and high variance. Both these models tackle error reduction in opposite ways. Whereas XGBoost is focusing on reducing bias, RandomForests is focused on reducing variance. It is noteworthy, however, that since XGBoost is adding one classifier at a time, being that the next one is trained to improve the already trained ensemble, it has to be trained sequentially. This deviates from other high scoring models we had, like Random Forests or Extra Trees.