February 27, 2018

**Lab Assignment 3**

Sanchit Singhal

INF 385T – Introduction to Machine Learning with Danna Gurari

Spring 2018

School of Information

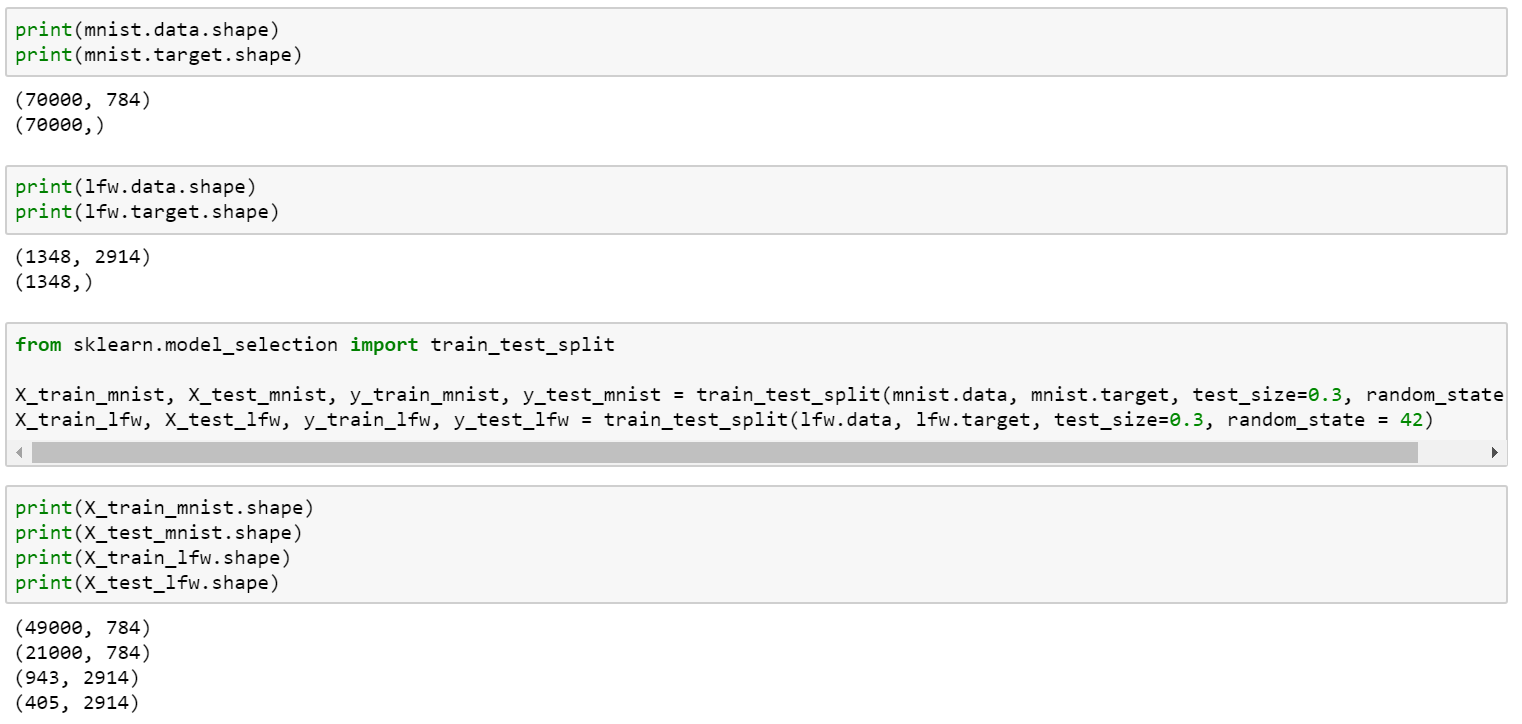
**Hyperlink to code:** [**https://introtoml-sanchit1276.notebooks.azure.com/nb/notebooks/IntroToML/LabAssignment3.ipynb**](https://introtoml-sanchit1276.notebooks.azure.com/nb/notebooks/IntroToML/LabAssignment3.ipynb)

1. **Image Classification with Dimensionality Reduction**
2. **Prepare two classification datasets**

Load data from external sources:

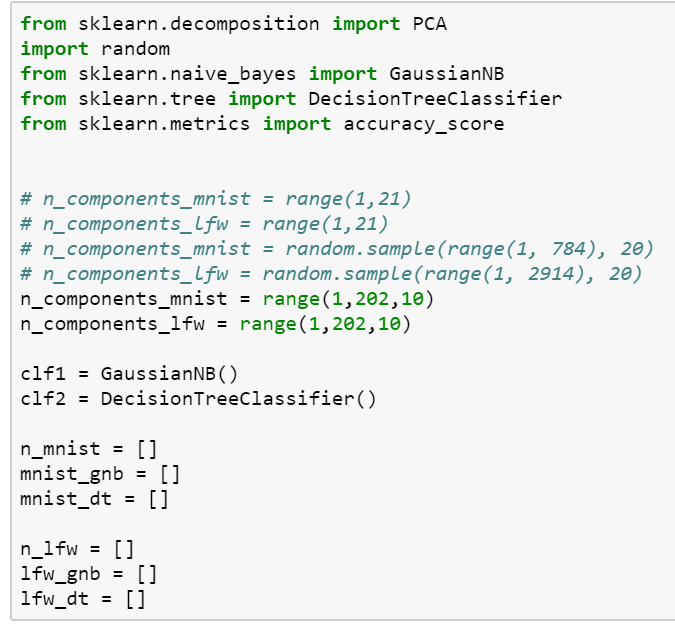


Split data using 70/30 split:

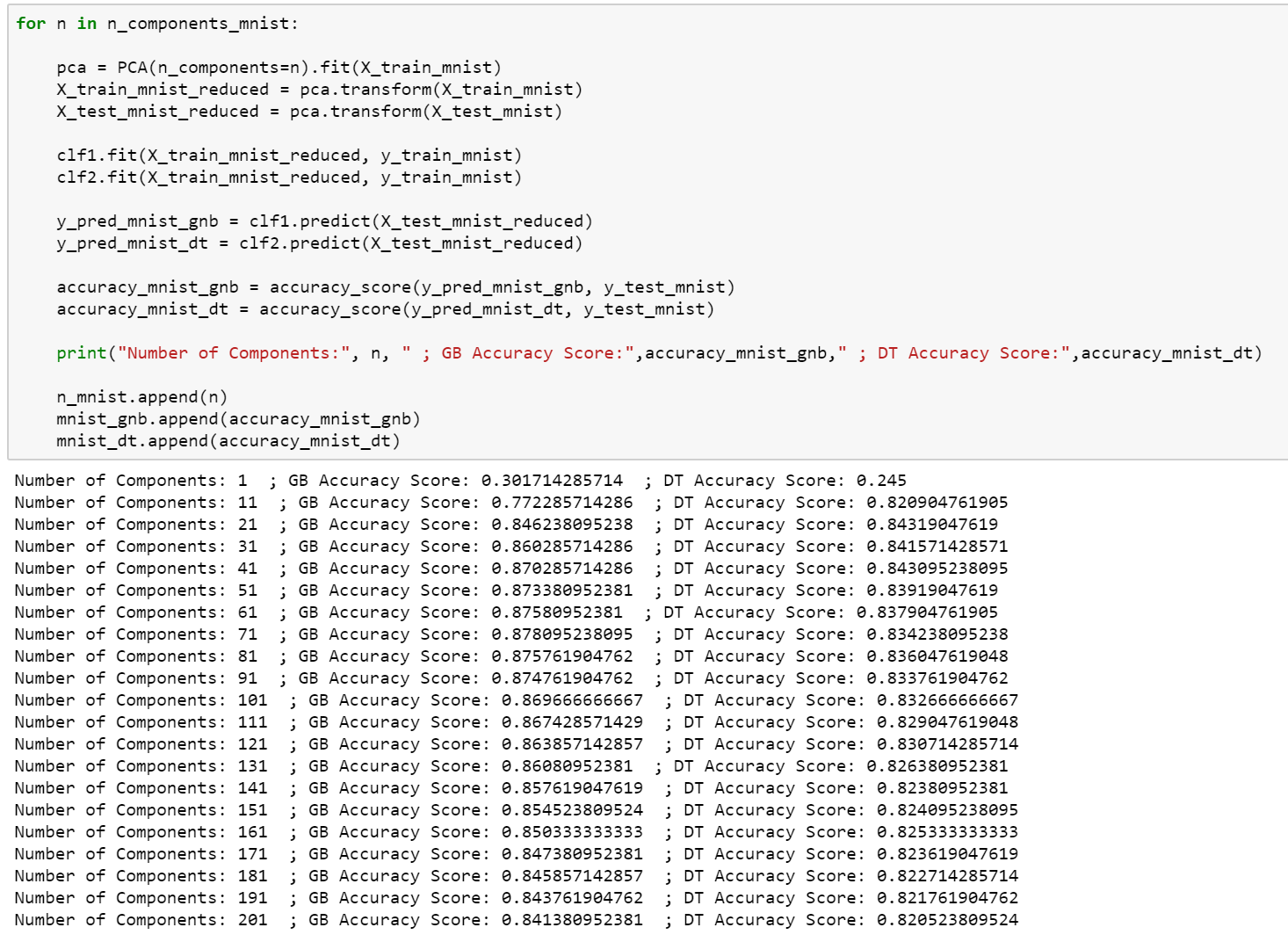


1. **Training and Evaluation of each dataset with 20 different feature dimension sizes on 2 classifiers**

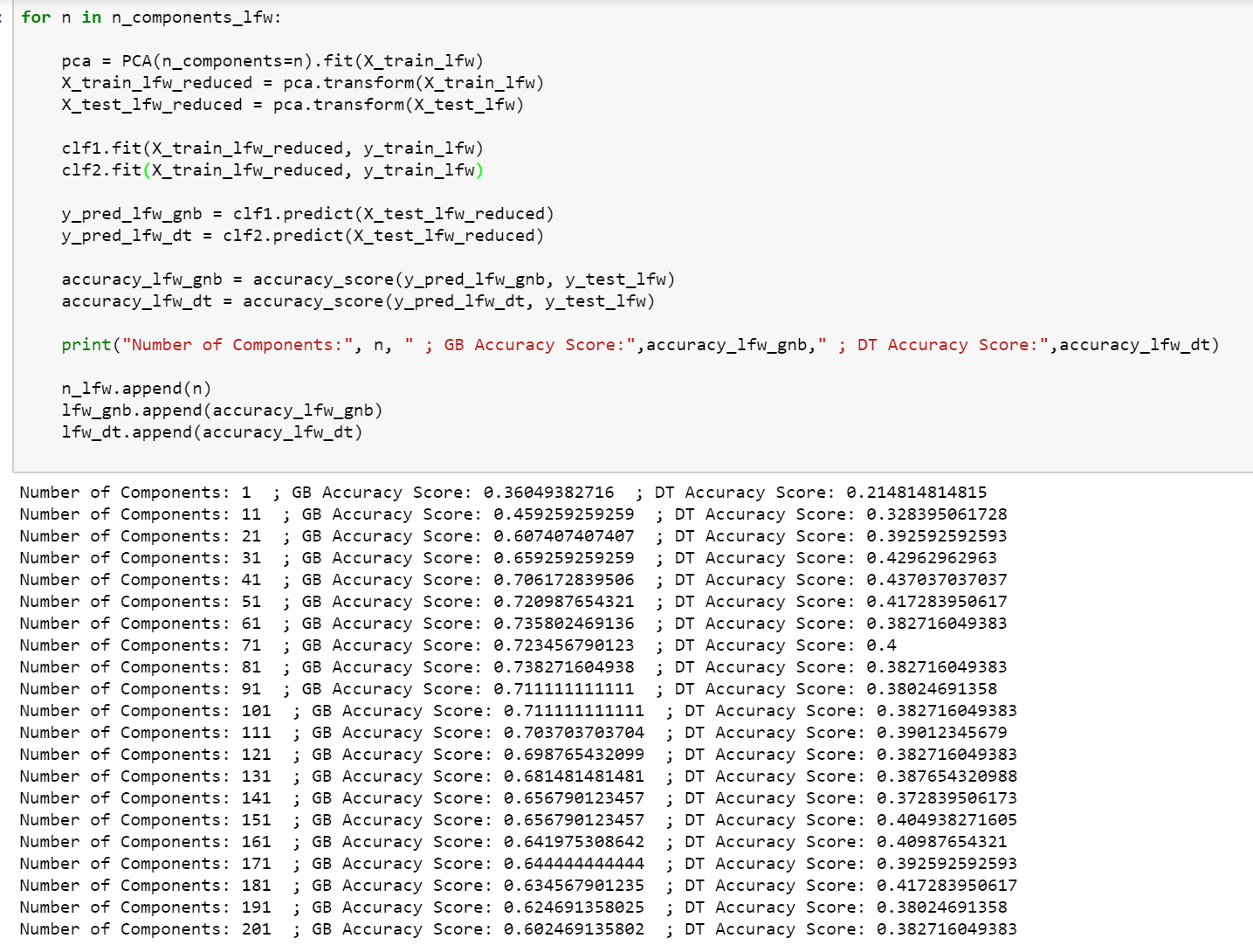
Preparing Naïve Bayes and Decision Tree classifiers, building array of component sizes:



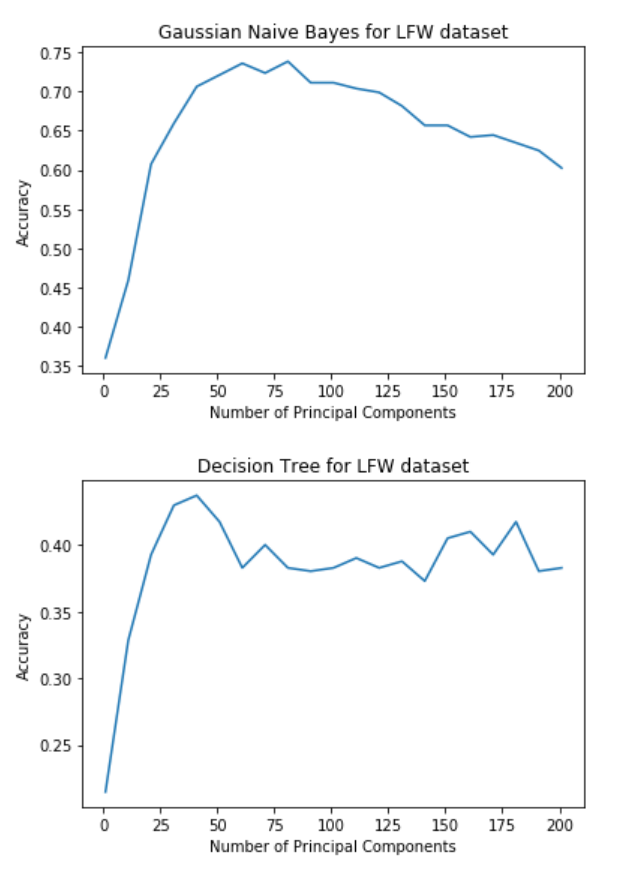
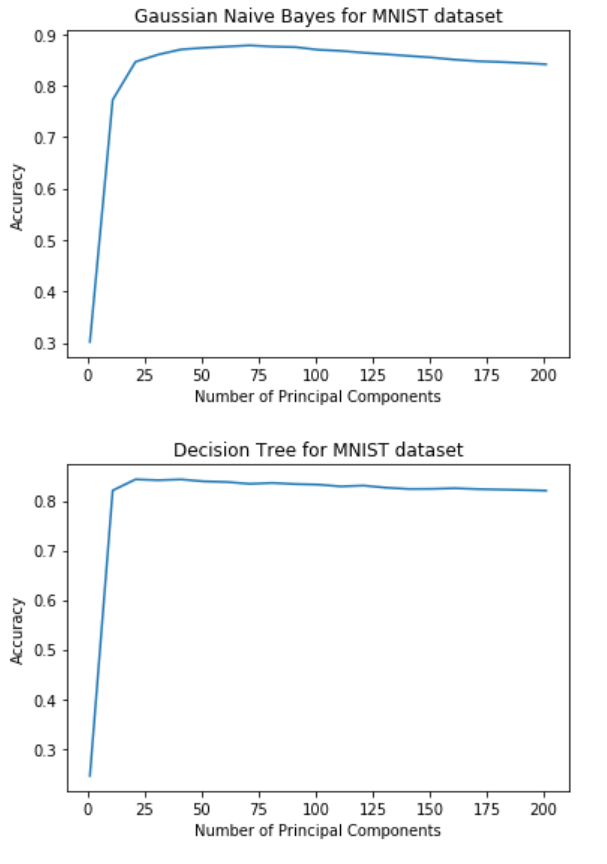
MNIST:



LFW:



1. **Predictive Performance**



The predictive performance for the MNIST dataset (around 70%-85%) was better for both classifiers as compared to the LFW dataset (around 35%-70%). This makes sense to me as facial recognition should be a harder problem to predict than number recognition. The LFW models were also trained with a smaller training set (due to computing resource issues) which perhaps led to lesser accuracy as well.

When comparing the predictive performance of the two classifiers, Gaussian Naïve Bayes models generally performed better than the Decision Trees for both datasets. They had a higher maximum accuracy in both cases. It must be noted that the Decision Tree for the LFW dataset did especially poorly – not even predicting with a 50% accuracy on any of the number of principal components.

1. **Influence of PCA on classification performance**

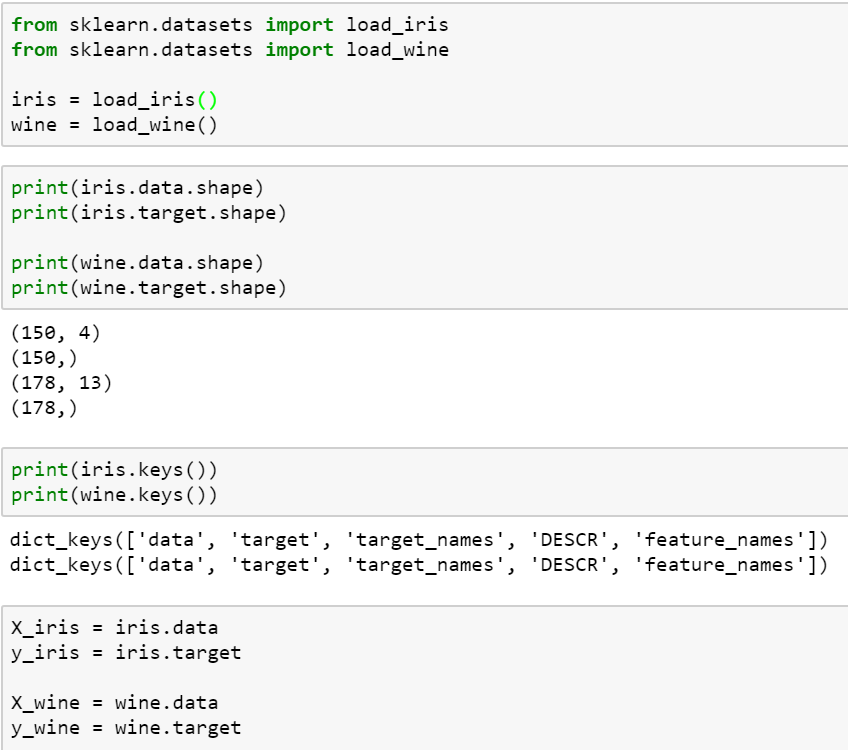
The trend of the accuracy among the four models when applying PCA with different dimension sizes is very similar. All four models have a sharp increase in classification performance initially as the number of principal dimensions increases from 1 to 25. After which point, accuracy continues to improve but the rate of improvement tapers off. For the MNIST dataset, the highest accuracy is around 25 components whereas for the LFW dataset, it is at around n=50. After hitting the peak accuracy value, all four models begin to lose accuracy as the number of principal components continues to increase. In the MNIST dataset, the rate of fall in accuracy is relatively slow, but in the LFW dataset, the classification performance falls significantly faster.

These results clearly indicate that reducing dimensionality (in this case through the unsupervised algorithm of PCA), is very important in obtaining accurate predictions. Intuitively it may seem that the more the features the better the model would be but as it can be seen, this is not always the case. In all four models, the accuracy fell after hitting a peak. This is most likely because of overfitting – the models begin learning the noise in the training data – which offer no assistance when predicting classes for new data. That being said, it is also fairly obvious that we do not want to reduce the dimensions too much such that we are not able to learn enough about the underlying structure – this is why the accuracy is sharply increases initially.

There must be an optimum number of components to reduce each problem and the challenge lies in finding it. In our case, the MNIST dataset was most accurately modelled when around 25 principal dimensions were used and the LFW dataset when approximately 50 were used. Since both classifiers, the Naïve Bayes and Decision Tree, had the highest accuracy at similar number of components for both the datasets it can be inferred that the optimum number of components is largely based on the problem and the dataset as opposed to the model being used to estimate. This makes sense to me – the number of predictors that defined a given class must be fixed in the real-world and therefore should not depend too much on which model we pick to describe that relationship. Again, the challenge for designing a machine learning system is figuring out the best number of components to use for that problem and an analysis similar to the one performed here is a good strategy to assist with that.

1. **Ensemble Learning**
2. **Load two classification datasets**

Load datasets:

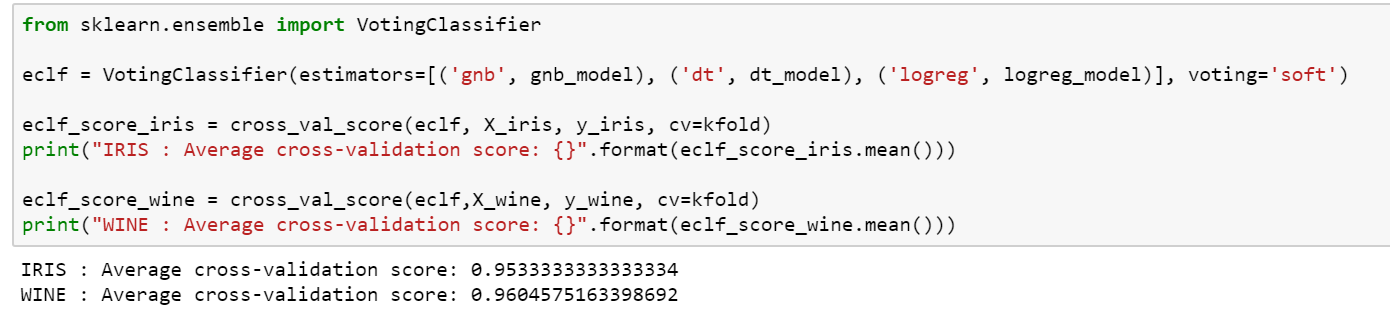


1. **Evaluate the following using 10-fold cross validation**
2. **Three different Classifiers**

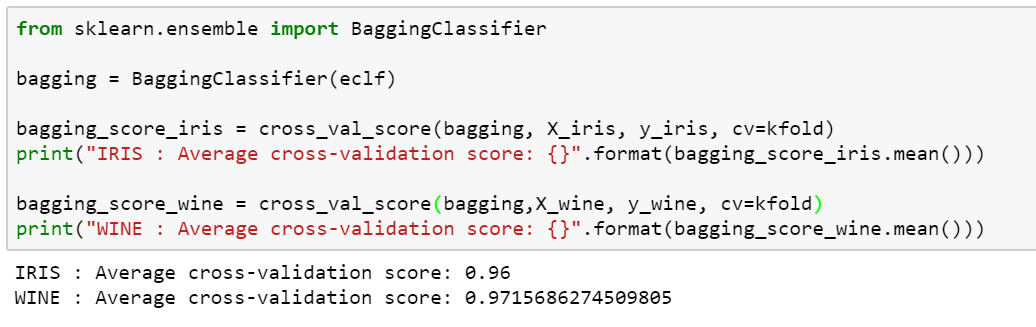
Chosen Classifiers: Naïve Bayes, Decision Tree, Logistic Regression



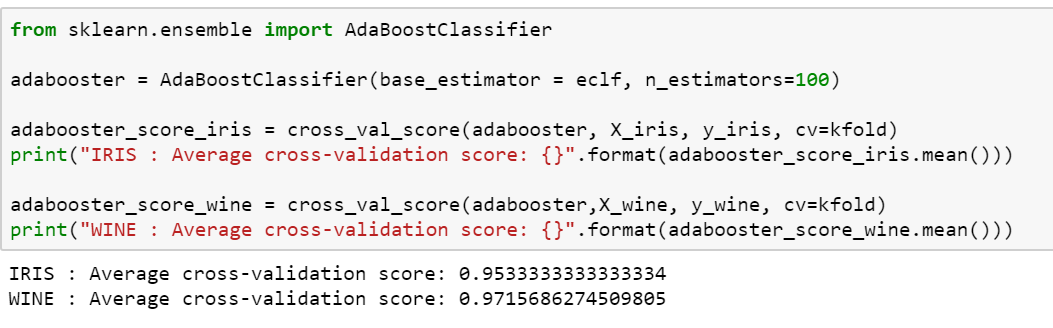
1. **Majority vote classifier**



1. **Bagging Method**



1. **Boosting Method**



1. **Mean Accuracy of each classifier for each dataset**

IRIS dataset:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Classifier | Naïve Bayes | Decision Tree | Log Reg | Majority Vote | Bagging | Boosting |
| Mean  Accuracy | 95.34% | 95.34% | 94.67% | 95.34% | 96% | 95.34% |

WINE dataset:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Classifier | Naïve Bayes | Decision Tree | Log Reg | Majority Vote | Bagging | Boosting |
| Mean  Accuracy | 97.16% | 89.97% | 94.67% | 96.05% | 97.16% | 97.16% |

For IRIS, Bagging of the ensemble provided the highest accuracy (96%) where as Bagging and Boosting performed the best for the WINE data (97.16%).

1. **Performance of the different ensemble methods**

On average it can be seen that the ensemble learning methods performed slightly better than the individual classifiers on both the data sets. This makes sense to me as the ensemble learning classifiers (in orange) draw from the individual learners and therefore should at least perform as well as the best base. When comparing the ensemble to the worst individual learning, Decision Tree on the Wine dataset, it can be clearly seen that an ensemble is able to mask the effects of a bad base classifier through voting and weights.

Since I have performed the ensemble methods of bagging and boosting on the eclf, which is the ensemble classifier, it is essentially using all three individual classifiers to consistently give more accurate classifications. I have decided to do this so that I am considering separate levels of ensembles where I am using majority voting as well as bagging/boosting. Just to try it out, my code also shows bagging/boosting on the individual classifier that performed the best, Naïve Bayes. It was interesting to see the results – Iris dataset did extremely well but the wine performed good, but not as good as the ensemble of ensembles.

This indicates to me that we have to be careful when building ensembles. If we choose a single classifier to boost/bag with then it may carry forward any systematic problems that it has. More importantly, I feel that ensembles are even more susceptible to overfitting as they may learn the noise which may lead to worse results on the test data. On the other hand, a good model together with bagging and boosting can amplify the results too because bias, and variance are minimized. As a modeler, I decided to use an ensemble to bag and boost because this would provide a diverse set of individual classifiers to the ensemble and more consistently give me better results. I believe it was also an important step to cross validate our models to reduce the possibility of overfitting which as mentioned can be fairly high in ensembles.