**Homework #4 Instructions**

**Casey Bennett, PhD**

**DSC540, Winter 2019**

**DePaul University**

**Overview**

Same two datasets as previous homeworks (Diabetes and Wine Quality), along with the two Python scripts. We will explore ways of using SVMs and other discriminant analysis methods. We’re also going to investigate feature selection in a little more detail.

For classification, we will be creating an object we’ll name ‘clf’, and for regression we’ll name ‘rgr’. These are objects we can call methods on (such as fitting a model to some data), and access their internal variables (such as getting predicted class labels). Scikit API links are in the accompanying document. We will only use cross-validation in this homework.

*\*Follow the steps below, record answers to questions in a word document, and turn in both your completed code and the word doc.*

**Pima Diabetes**

Open up HW4\_Diabetes.py

1. First, let’s run an SVM classifier.
   1. First we need to import the functions, on line 11, replace the comment with a call to import SVC () from the sklearn “SVM” package.
   2. On line 278, create a SVC(). Using the API link in the accompanying document, call that function, and pass in the following parameters:
      1. Set kernel to ‘rbf’
      2. Set gamma = ‘scale’
      3. Set C = 1.0
      4. Set probability = True
      5. Set random\_state variable to rand\_st
   3. Add in a cross\_validate function on line 279 (use previous homework as an example) with 5 folds, and pass in the clf object.

*\*Question #1a: Run the code once, record the accuracy and AUC score. What do you notice about the scores?*

*\*Question #1b: In the Scikit API for SVC, it explains the probability parameter … why did we set it equal to ‘True’? What does that do?*

1. Let’s explore how changing some of the parameters for the SVM
   1. Change the kernel parameter of the SVC() to ‘sigmoid’
   2. Now set the kernel to ‘linear’

*\*Question #2: Run the code once for each setting of the kernel, record the accuracy and AUC scores. What do you notice about the scores compared to Question #1? What about run-times?*

1. Finally, let’s run feature selection again on the Diabetes dataset, but this time do it using SVMs. Since SVMs depending on the kernel type can function like a linear regression method, and produce coefficients that we can use as a measure of “feature importance” natively.
   1. Let’s leave the SVC() kernel set as ‘linear’
   2. To turn on feature selection, we need to first on line 38 change the feat\_select flag to equal 1 instead of 0
   3. Note that there is an option to change the feature selection type is already set to 2 (wrapper-based) on line 39
   4. You will need to add a SVC(), call to pass to the clf object on line 191, you can use something similar to the calls used elsewhere in the code. Don’t forget to set the parameters, particularly the kernel.
   5. Make sure you set the kernel to the ‘linear’ when using it for feature selection (so it produces coefficients for FS), or your code will give errors.

*\*Question #3a: Run the code once, record the accuracy and AUC scores. What do you notice about the scores? How do they compare to the performance Question 2 above for SVMs using a linear kernel with no feature selection?*

*\*Question #3b: What features were selected, and which were removed? Were there any differences from when you did feature selection with Boosting in HW3, or Random forests in HW2?*

**Wine Quality Dataset**

Open up HW4\_Wine.py … First, let’s repeat the steps we did above for Diabetes.

1. First, let’s run an SVM regressor.
   1. First we need to import the functions, on line 12, add a call to import SVR() from the sklearn “SVM” package.
   2. On line 368, create a SVR(). Using the API link in the accompanying document, call that function, and pass in the following parameters:
      1. Set kernel to ‘rbf’
      2. Set gamma = 0.1
      3. Set C = 1.0
   3. Add in a cross\_validate function on line 369 (use previous homework as an example) with 5 folds, and pass in the rgr object.

*\*Question #4a: Run the code once, record the RMSE and Explained Variance.*

*\*Question #4b: In the Scikit API for SVR, you will notice there is no probability parameter (averse to for the classifier version), why do you think that is?*

1. Let’s explore how changing some of the parameters for the SVM
   1. Change the kernel parameter of the SVR() to ‘sigmoid’
   2. Now set the kernel to ‘linear’

*\*Question #5: Run the code once for each setting of the kernel, record the RMSE and Explained Variance. What do you notice about the scores compared to Question #4? What about run-times?*

1. Discriminant Analysis methods can be prone to issues with data distributions, so let’s see if normalizing the features has any effect.
   1. Let’s leave the SVC() kernel set as ‘linear’
   2. On line 35, change the norm\_features flag to equal 1 instead of 0

*Question #6: Run the code once, record the RMSE and Explained Variance. What do you notice about the scores, or the run times? How do they compare to results in Question #5?*

1. Let’s run feature selection again on the Wine dataset, just like we did for Diabetes above.
   1. Let’s leave the norm\_features flag turned on (set to 1)
   2. To turn on feature selection, we need to first on line 38 change the feat\_select flag to equal 1 instead of 0
   3. Note that there is an option to change the feature selection type is already set to 2 (wrapper-based) on line 39
   4. You will need to add a SVR(), call to pass to the rgr object on line 254, you can use something similar to the calls used elsewhere in the code. Don’t forget to set the parameters, particularly the kernel.
   5. Make sure you set the kernel to the ‘linear’ when using it for feature selection (so it produces coefficients for FS), or your code will give errors.

*\*Question #7a: Run the code once Record the RMSE and Explained Variance Score. What do you notice about the scores? How do they compare to performance in Question 6 above for SVMs using a linear kernel with no feature selection?*

*Question #7b: What features were selected, and which were removed?* *How did this compare with features selected in previous homeworks (Random Forests, Gradient Boosting)?*

1. Let’s compare some other feature selection methods, such as using mutual information between each feature and the target (based on information theory and entropy).
   1. Set fs\_type line 39 to 3

*\*Question #8a: Run the code once Record the RMSE and Explained Variance Score. What do you notice about the scores? How do they compare to performance above using wrapper feature selection in Question 7?*

*Question #8b: What features were selected, and which were removed?*

1. Let’s run feature selection again on the Wine dataset, except using a full-blown wrapper. I’ve already written the helper function for you (line 55), you just need to turn it on. This is a very straightforward exhaustive search method, with no regularization, so it will probably pick more features than truly necessary. Warning: it can take a while to run, perhaps a couple minutes depending on your computer.
   1. Set fs\_type line 39 to 4
   2. You will need to add a SVR(), call to pass to the rgr object on line 304, you can use something similar to the calls used elsewhere in the code. Don’t forget to set the parameters, particularly the kernel.

*\*Question #9a: Run the code once Record the RMSE and Explained Variance Score. What do you notice about the scores? How do they compare to performance above for feature selection using the simple wrapper in Question 7 and the univariate mutual info in Question 8?*

*Question #9b: What features were selected, and which were removed?*

1. You might note the long run times for Question #9. You can imagine that would be even more exacerbated with different kernels or ML methods. Given that context, look at the comment on line 56. No changes are needed for this question, norcode to run here.

*\*Question #10a:* *Based on the comment on line 56, explain how we might search the feature set space in a more optimal manner.*

*\*Question #10b: If you uncomment the print statements on lines 87 and 96, and watch the code run, you may notice that there are actually several different feature sets that perform nearly the same as the optimal feature set, some of which have much fewer features than others. What is one way we could force the wrapper method to select smaller feature sets, even if they have slightly less performance? (HINT: line 56 also mentions something about this)*

**Summary Questions**

*\*Question #11: Line up the results from Homeworks 1,2,3,4 as a table for both Diabetes and Wine, with rows of performance metrics for each ML method (Decision Trees, Random Forests, Gradient Boosting, Ada Boost, Neural Networks, SVMs). Looking at this table of performance metrics, how would you explain the table to a boss or customer.*

*\*Question #12: If we had to explain to someone what really drives peoples’ perception of wine quality, what would you say based on your findings in this homework (e.g. Q8) and previous ones? Are there 2-3 features we can say are consistently most important? If so, can you hypothesize why those features might be important?*