HW4
Chaonan Shi
1901412

Pima Diabetes

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

Performance	*ACC*	*AUC*	*CV Runtime*
SVM	*0.76*	*0.82*	*0.30*

Analysis:
By running the code once, comparing with results of Gradient Boosting and Ada Boosting from HW3, the result from SVM almost same as those two. But running time is significant lower than the Gradient Boosting.

**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?*

Analysis:
From description, the probability parameter illustrate that: whether to enable probability estimates. This must be enabled prior to calling fit, and will slow down that method. Moreover, when the constructor option probability is set to True, class membership probability estimates (from the methods predict proba and predict log proba) are enabled. In the binary case, the probabilities are calibrated using Platt scaling: logistic regression on the SVM's scores, fit by an additional cross-validation on the training data. In the multiclass case, this is extended as per Wu et al.

2) **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?*

Performance	*ACC*	*AUC*	*CV Runtime*
sigmoid	*0.50*	*0.32*	*0.36*
linear	*0.77*	*0.83*	*96.24*

Analysis:

By running code once for each setting, I found that the accuracy score of 'sigmoid' is lower than the 'rbf' and 'linear', performed only 0.50 and 0.32; For 'linear', the accuracy score is similar with 'rbf', but running time is way longer than others with 96.24 sec.

In this case, the 'sigmoid' $K(X,Y)=\tanh(\gamma \cdot XTY+r)$ which is similar to the sigmoid in the logistic regression. For the 'linear' setting, on the other hand, the function more like linearly separable, that is, it can be separated using a single Line. From my point of view, since the dataset contains relatively large set of features, so that the linear function takes more longer than others.

3) **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?*

Performance	*ACC*	*AUC*	*CV Runtime*
SVM (linear)	*0.65*	*0.50*	*0.13*

Analysis:

By running once, I found that the performance of SVM (linear) with feature selection lower than the performance without feature selection for linear function. However, since number of features been reduced from feature selection. The running time also reduced significantly from 96 sec to 0.13.

**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?*

(From Random Forests in HW2)

Total Features (9)	*Class*	*Blood Glucose*	*BMI*	*'Family History'*	*'Age'*	*Times Pregnant*	*Blood Pressure*	*Skin Fold Thickness*	*2-Hour Insulin*
Selected (4)		√	√	√	√				

Performance	*ACC*	*AUC*	*CV Runtime*
Scores	*0.76*	*0.82*	*1.069*

(From Gradient Boosting in HW3)

Total Features (8)	*Class*	*Blood Glucose*	*BMI*	*'Family History'*	*'Age'*	*Times Pregnant*	*Blood Pressure*	*Skin Fold Thickness*	*2-Hour Insulin*
Selected (3)		√	√		√				

Performance	*ACC*	*AUC*	*CV Runtime*
Scores	*0.77*	*0.83*	*1.069*

(From SVM in HW4)

Total Features (8)	*Class*	*Blood Glucose*	*BMI*	*'Family History'*	*'Age'*	*Times Pregnant*	*Blood Pressure*	*Skin Fold Thickness*	*2-Hour Insulin*
Selected (1)				√					

Performance	*ACC*	*AUC*	*CV Runtime*
Scores	*0.65*	*0.50*	*0.13*

Analysis:
Comparing with *Gradient Boosting in HW3* *(Blood Glucose, BMI, AGE)* and *Random Forests in HW2* *(Blood Glucose, Family History, BMI, AGE)*. Only feature 'Family History' been selected at SVM with linear function.

Wine Quality Dataset

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

Algorithm /Performance	*RMSE*	Explained_Variance_Score	*CV Runtime*
SVM (rbf)	*0.77*	*0.08*	*1.12*

**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?*

Analysis:
From regression point of view, since we are running regression instead of the classification. In this case, the probability works more like logistic regression, in other works, the output has to be binary but continuous.

5) **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?*

Performance	*RMSE*	Explained_Variance_Score	*CV Runtime*
sigmoid	*0.91*	*0.00*	*0.56*
linear	*0.66*	*0.29*	*23.75*

Analysis:

By running code once, kernel with 'sigmoid' shows RMSE and Explained_Variance_Score with 0.91 and 0.00 respectively; and kernel with 'linear' shows RMSE and Explained_Variance_Score 0.66, 0.29 reselctively;
In this case, we can summarize that the performance of linear better than the performance of sigmoid. However, the running of linear is significant higher than the sigmoid.
Comparing with performance of 'rbf' at question 4, the linear better than the 'rbf' but sigmoid. For running time, the running of linear still much more slower than others.

6) *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?*

Performance	*RMSE*	Explained_Variance_Score	*CV Runtime*
linear	*0.66*	*0.29*	*1.16*

Analysis:

By running code once, kernel with 'linear' shows same level with question 5, but running time reduced significantly.

From my point of view, since we normalized the features by Discriminant Analysis, so that the noise should be reduced also scattered randomly after that. That makes linear function running more faster than before normalization.

7) **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?*

Performance	*RMSE*	Explained_Variance_Score	*CV Runtime*
Linear(with selection)	*0.68*	*0.27*	*0.63*

Analysis:

By running code once with feature selection, the performance stay at same level comparing with question 6, but running time reduced significantly from 1.16 to 0.63. It makes sense since less features been selected then number of running time should be reduced.

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

(From Random Forests in HW2)

Total Features (10)	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
Selected (3)		√								√	√

(From Gradient Boosting in HW3)

Total Features (10)	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
Selected (4)		√					√			√	√

(From SVM in HW4)

Total Features (10)	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
Selected (3)		√								√	√

Analysis:
Comparing with Gradient Boosting in HW3 (volatile acidity, total sulfur dioxide, sulphates, alcohol) and Random Forests in HW2 (volatile acidity, sulphates, alcohol). In this case, feature were selected from SVM exactly same as HW2 (volatile acidity, sulphates, alcohol).

8) **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?*

Performance	*RMSE*	Explained_Variance_Score	*CV Runtime*
Linear(with selection)	*0.68*	*0.27*	*0.71*

Analysis:
Comparing with wrapper feature selection, the level of accuracy stay on the same level, but running time increased a little bit from 0.63 to 0.71.
In this case, we could summarize that the wrapper feature selection and the Univariate Selection performed same.

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

Total Features (11)	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
Selected (5)		√					√	√		√	√

Analysis:
For selection of Univariate, features: volatile acidity, total sulfur dioxide, density, sulphates, and alcohol were selected.
Comparing with previous one, the feature 'density' were selected in this approach.

9) **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?*

Performance	*RMSE*	Explained_Variance_Score	*CV Runtime*
Linear(with selection)	*0.66*	*0.30*	*0.99*

Analysis:
Comparing with feature selection using the simple wrapper in Question 7 and the univariate mutual info in Question 8. The performance of this approach does not increase or decrease dramatically, still staying at same level.

However, the whole processing have taken more than 2 minute.

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

Total Features (11)	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
Selected (3)		√					√	√		√	√

Analysis:
For this approach, 9 features were selected, on the contrast, only features with 'fixed acidity' and 'citric acid' were not selected.

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

Analysis:
From comment, we can summarize that if we want to save more time for searching and more optimal manner, first we need to apply regularization to penalize the residuals and classify the records separately.
On the top of that, we can also apply various searching algorithms to reduce the running time of searching. Like comment says: greedy search, random search, genetic algorithms, etc. From my point of view, it would be more efficient if we apply binary search, since for binary search, the time complexity for this is log n, which is more faster than others.

**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)*

Analysis:

From my point of view, it could be more efficiency if using binary search. For instance, the processing behind this is that each time we separate the results to be half, and searching the optimal one from each part and make comparison.

Moreover, we can add grid search to find most adequate subset of variables or apply normalization. It also reduce running time significantly.

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.

For Diabetes:

Performance	*ACC*	*AUC*	*CV Runtime*
SVM	*0.76*	*0.82*	*0.30*
Gradient Boosting	*0.76*	*0.82*	*0.53*
Ada Boosting	*0.76*	*0.83*	*1.10*
Random Forest	*0.76*	*0.82*	*1.069*
MLPClassifier	*0.70*	*0.72*	*0.68*
Decision Tree	*0.68*	*0.5*	*0.66*

For Wine:

Performance	*RMSE*	Explained_Variance_Score	*CV Runtime*
SVM	*0.66*	*0.30*	*0.99*
Gradient Boosting	*0.65*	*0.34*	*0.44*
Ada Boosting	*0.66*	*0.31*	*1.39*
Random Forest	0.68	0.25	0.97
MLPClassifier	*0.64*	*0.33*	*2.89*
Decision Tree	0.96	-0.48	*0.66*

Analysis:

From my point of view, for the classification problem, basically most of algorithm performed at same level, but running time. By lining up all of algorithms for both Wine dataset and Diabetes dataset. In term of classification problem, I would like to choose SVM in this case, not only it performs same accuracy score comparing with others, but also running more efficiency which only cost 0.30 sec.

In term of regression problem, I would like to choose Gradient Boosting method as my best model. Since not only it explained highest variance comparing with others, but also running lowest time as well, only cost 0.44 sec.

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?*

Analysis:

Through all experiments. Features with volatile acidity, total sulfur dioxide, sulphates and alcohol should be considered as important features in this case. Not only these features explained most of variance, but also make our model more simpler.

From my point of view, for volatile acidity, Sweet wines made using botrytis cinerea (noble rot) are often quite high in volatile acidity levels, as the grapes tend to have high levels of acetic acid bacteria naturally present. The same can be said of wines made from dried grapes.

About total sulfur dioxide, Total Sulfur Dioxide (TSO2) is the portion of SO2 that is free in the wine plus the portion that is bound to other chemicals in the wine such as aldehydes, pigments, or sugars. The TSO2 level is also regulated by the U.S. Alcohol and Tobacco Tax and Trade Bureau (TTB): The maximum allowable concentration for a bottled wine is 350 ppm (mg/L) of TSO2. Theoretically, in a perfect world, the TSO2 in any wine should mirror the total of all the SO2 additions that have been made to it over the course of its history. However, the measured TSO2 rarely matches the expected amount perfectly, due to imprecise additions, SO2 removed with lees and sediments, and any volatile SO2 lost to the air.
[February 27, 2018, By Maureen Moroney]

Sulphates, there is lack of information for this Feature, but from my point of view, Sulphates is a chemical compound made up of sulfur and oxygen. It occurs naturally but can also be produced in a laboratory. It's used to preserve foods and beverages, which it does by acting as an antioxidant and antimicrobial.
From one article [By Nutrition Diva Monica Reinagel on July 15, 2017]. If drinking red wine gives you a headache, you've probably had someone tell you that sulfites are the likely culprit.

Last, alcohol is general concept, and alcohol content is an essential element of wine for its intoxicating effects. Without alcohol, wine would just be simple grape juice.

There is also 3 ways Alcohol Content Affects the Taste of Wine:

1) Flavor harmony
2) Body
3) Perceived taste