student id: c1954633 name: Rongjian Huang

Part1:
1.Theory
Question1: In machine learning program learn rules from input data, while in rule-based system rules are given by human and program follow rules to achieve a task.

Question2: Supervised learning need lables which means the correct answer to train a model, but unsupervised doesn't. Unsupervised learning tend to find out the data with similar pattern of features.

Question3: Overfitting means model have bad generalisation. It can fit training set well but bad at unseen test set.

2.Pratice
Task1:
TP=6 FP=3
FN=3 TN=8

precision = TP/(TP+FP) = 6/9 = 2/3
recall = TP/(TP+FN) = 6/9 = 2/3
F1-score = 2*precision*recall/(precision+recall) = 2/3
Accuracy = (TP+TN)/(TP+TN+FP+FN) = 14/20 = 7/10

Task2:
Linear regression using normal equation
Training to get theta

```
lin_reg = LinearRegression().fit(x_train, y_train)
```

Predicting labels

```
y_lin=lin_reg.predict(x_test)
```

RMSE: 0.7743778616729023

Lasso regression
Training model

```
ridge_reg = Ridge(alpha=1, random_state=42).fit(x_train, y_train)
```

Predicting labels

```
y_rid=ridge_reg.predict(x_test)
```

RMSE:0.779805972905897

result

	Precision	Recall	F1-score	Accuracy
Fold 1	0.750170733221581	0.742016857194338	0.7441936007796	0.751111111111111
Fold 2	0.753976415741122	0.745651486401012	0.748622618061309	0.759733036707453
Fold 3	0.757544528272536	0.738687969189262	0.744178439215353	0.761957730812013
Fold 4	0.735159236330397	0.72756445695331	0.72965575592004	0.737486095661847
Fold 5	0.742447098261052	0.734869088694946	0.737433819035672	0.747497219132369
Fold 6	0.756452157935309	0.744387251942197	0.747362059076018	0.756395995550612
Fold 7	0.736776605660058	0.725158125856095	0.727661980518646	0.737193763919822
Fold 8	0.743401339673774	0.742622248952589	0.742995997712979	0.75083426028921
Fold 9	0.746117762919194	0.744937067987207	0.745501903426167	0.756395995550612
Fold 10	0.728505984362032	0.716283192243687	0.720290321342088	0.738598442714127
Average	0.745055186237705	0.736217774541464	0.738789649508787	0.749720365144918

Task3:

What I have done is basically a bag-of-words model. I used the 1000 most frequent word to generate the dictionary, which is the bag of words. And my design is to mark the presence of a word as a boolean, with 0 being absent and 1 being present. So I have a vector for each speech in length of 1000. This features only focus on appearance but not the logic or order of the word. It's quite simple but effective method to analyze the dataset.
First I split dataset in 10 pairs of training set and testing set for cross-validation, then I defined the stop-words which need to be ignore in the program. Then in each fold, I generated a dictionary of 1000 most frequent words after lemmatization and transforming to lower case. After that, I extract the label of training set and transforming training set into vector. In this part, some tricks have been used, for example skip the first line which is the categories and using list[-1] to extract final element is list which are labels. Also, there is one line in dataset which is empty may lead to some error of list, I add a line of code to check and ignore this line(Since I don't know where it is). With the vectorized features, I trained a SVM model. Then I transform test set into vector just like what I have done above. After that I use SVM model to predict the label of test set and compare it with the real label. Then a result of Precision, Recall, F1-score, Accuracy of this fold can be generated with the help of sklearn package. After 10 folds, the result of each fold and the average of 10 folds were written in "result.csv" as an output which is the chart below.