REPORT OF SKLEARN

Maurizio Franchi

June 2015

The goal of the assignment is to compare the performance of three classification algorithms: *Naive Bayes*, *SVM (Support Vector Machines)*, and *Random Forest*.
To do that, we use *Python* with the *Scikit-Learn library*.

1 Methodology

In the first step it is required to create a classification dataset, with at least 1000 samples and at least 10 features, with binary values. (For this assignment I used 5000 samples and 50 features) The code to do that is the following:

```
dataset_x, dataset_y = datasets.make_classification(n_samples=5000, n_features=50)
```

In which:

- `dataset_x` is array of shape [n_samples, n_features] and then the generated samples
- `dataset_y` is array of shape [n_samples] and then the integer labels for class membership of each sample.

To train the algorithms, the dataset is splitted using 10-fold cross validation which the following code:

```
kf = cross_validation.KFold(n, n_folds=10, shuffle=True, random_state=None)
```

where `n` is the length of `dataset_x`
In the second step I calculate the three trained algorithms *Naive Bayes*, *SVM*, and *Random Forest*.

1.1 Gaussian Naive Bayes

The first trained algorithm is the the *Gaussian Naive Bayes*, and this is the *Python code*:

```
clf_gaussian = GaussianNB()
clf_gaussian.fit(X_train, y_train)
pred_gaussian = clf_gaussian.predict(X_test)
```

where:

- the `fit` method fits Gaussian Naive Bayes according to X, y
- the `predict` one performs the proper classification: i.e. predicts labels y from the training data X

1.2 SVM

The second algorithm is a *Support Vector Machine (Support Vector Classification)*. This algorithm has a C parameter, which is the penalty for each classification error. In order to find its best among different possible values, it is performed an inner cross validation (5-fold).

Possible C values are: 0.01, 0.1, 1, 10, 100

The best value of C is computed by choosing the value that maximizes the average inner F1-score. The code in the script for the best value is:

```
bestC = Cvalues[np.argmax(innerscore_SVM)]
```

Another parameter is the kernel, and the one used is the *RBF (Radial Basis Function)* kernel. The code in the script for this part is:

```
clf = SVC(C=C, kernel='rbf', class_weight='auto')
clf.fit(X_train, y_train)
return clf.predict(X_test)
```

See *Gaussian Naive Bayes* for the explanation of *fit* and *predict* methods

1.3 Random Forest Classifier

The last algorithm is the *Random Forest Classifier*, which uses a multitude of decision trees to be trained.

The algorithm has an *N estimators parameter*, i.e. the number of trees in the forest.

Possible values to be tested are 10, 100, 1000.

Also in this case, to find the best value, it is used an inner cross validation, to maximize the inner F1-score.

The other parameter is the function to measure the quality of a split. The criteria used is *gini* for the Gini impurity. The code in the script for this part is:

```
clf = RandomForestClassifier(n_estimators = n_estimators, criterion='gini', random_state=None)
clf.fit(X_train, y_train)
return clf.predict(X_test)
```

See *Gaussian Naive Bayes* for the explanation of *fit* and *predict* methods

2 Performance evaluation

To evaluate the cross-validated performance, they are used methods from the *metrics* module of *scikit* library. The three measures are: *accuracy*, *F1-score* and *AUC ROC*.

2.1 Accuracy

At the first it was to calculate the *accuracy* of each algorithm. To calculate this the formulas below are used:

- `accuracy_GM.append(metrics.accuracy_score(y_test,pred_gaussian)` for calculate the *accuracy* of the *Gaussian Naive Bayes*;
- `accuracy_SVM.append(metrics.accuracy_score(y_test,pred_SVM)` for calculate the *accuracy* of the *SVM*;
- `accuracy_RFC.append(metrics.accuracy_score(y_test,pred_RFC)` for calculate the *accuracy* of the *Random Forest Classifier*;

The result is shown in the table 1.

To better compare the three algorithms it was calculated the mean, the maximum and minimum value and the standard deviation of accuracy of 10 values for each algorithm.
The result is shown in the table 2.

From this analysis it can be seen that the accuracy of the Random Forest is better than the other two algorithms. This because comparing the mean of the three algorithms I can see that the highest mean is the mean of the Random Forest (0.9382) in fact the maximum and minimum values are close (maximum = 0.9459 and minimum = 0.9240).
The Random Forest Classifier is also the best algorithm because it has the high mean and minimum value, in fact the algorithm is best if the mean value is greater and lesser is the probability of making errors in a single test if the minimum value is high.
This result is supported by the standard deviation in fact the standard deviation of the Random Forest is lower than the other algorithms.

Gaussian Naive Bayes	SVM	Random Forest Classifier
0.9040	0.9100	0.9459
0.9000	0.9100	0.9379
0.8880	0.9040	0.9399
0.9000	0.9180	0.9439
0.9120	0.9260	0.9379
0.8760	0.8860	0.9240
0.8960	0.9080	0.9379
0.8800	0.9040	0.9320
0.8900	0.9200	0.9399
0.8840	0.8940	0.9419

Table 1: Accuracy of Gaussian Naive Bayes, SMV and Random Forest Classifier

	Gaussian Naive Bayes	SMV	Random Forest Classifier
Mean	0.8930	0.9080	0.9382
Maximum	0.9120	0.9260	0.9459
Minimum	0.8760	0.8860	0.9240
Standard deviation	0.0108	0.0113	0.0059

Table 2: Accuracy of mean, maximum, minimum and standard deviation of Gaussian Naive Bayes, SMV and Random Forest Classifier

This means that the data calculated with Random Forest algorithm are close to the mean and then the each data is not very different to the mean.

2.2 f1-score

As for accuracy at the first it was to calculate the *f1-score* of each algorithm. To calculate this the formulas below are used:

- `f1_GM.append(metrics.f1_score(y_test,pred_gaussian)` for calculate the *f1-score* of the *Gaussian Naive Bayes*;
- `f1_SVM.append(metrics.f1_score(y_test,pred_SVM)` for calculate the *f1-score* of the *SVM*;
- `f1_RFC.append(metrics.f1_score(y_test,pred_RFC)` for calculate the *f1-score* of the *Random Forest Classifier*;

The result is shown in the table 3.

Gaussian Naive Bayes	SMV	Random Forest Classifier
0.9062	0.9087	0.9450
0.8979	0.9028	0.9350
0.8880	0.9012	0.9377
0.9084	0.9216	0.9469
0.9130	0.9252	0.9378
0.8640	0.8701	0.9120
0.8988	0.9094	0.9388
0.8863	0.9066	0.9330
0.8979	0.9239	0.9423
0.8884	0.8978	0.9425

Table 3: f1-score of Gaussian Naive Bayes, SMV and Random Forest Classifier

As for *accuracy* to better compare the three algorithms it was calculated the average, the maximum and minimum value and the standard deviation of *f1-score* of 10 values for each algorithm.

The result is shown in the table 4:

As for *accuracy*, from this analysis it can be seen that the *f1-score* of the *Random Forest Classifier* is better than the other two algorithms. This because comparing the mean of the three algorithms I can see that the highest mean is the mean of the

	Gaussian Naive Bayes	**SMV**	**Random Forest Classifier**
Mean	0.8949	0.9068	0.9371
Maximum	0.9130	0.9252	0.9469
Minimum	0.8640	0.8701	0.9120
Standard deviation	0.0133	0.0152	0.0093

Table 4: f1-score of mean, maximum, minimum and standard deviation of Gaussian Naive Bayes, SMV and Random Forest Classifier

Random Forest Classifier (0.9371) in fact the maximum and minimum values are close (maximum = 0.9469 and minimum = 0.9120). The *Random Forest Classifier* is also the best algorithm because it has the high mean and minimum value, in fact the algorithm is best if the mean value is greater and lesser is the probability of making errors in a single test if the minimum value is high. This result is supported by the standard deviation in fact the standard deviation of the *Random Forest Classifier* is lower than the other algorithms. This means that the data calculated with *Random Forest Classifier* algorithm are close to the mean and then the each data is not very different to the mean.

2.3 Auc Roc

As for *accuracy* and *f1-score* at the first it was to calculate the *auc roc* of each algorithm. To calculate this the formulas below are used:

- `auc_GM.append(metrics.roc_auc_score(y_test,pred_gaussian)` for calculate the *auc roc* of the *Gaussian Naive Bayes*;
- `auc_SVM.append(metrics.roc_auc_score(y_test,pred_SVM)` for calculate the *auc roc* of the *SVM*;
- `auc_RFC.append(metrics.roc_auc_score(y_test,pred_RFC)` for calculate the *auc roc* of the *Random Forest Classifier*;

The result is shown in the table 5.

Gaussian Naive Bayes	**SMV**	**Random Forest Classifier**
0.9044	0.9099	0.9459
0.9020	0.9088	0.9386
0.8885	0.9038	0.9396
0.8977	0.9185	0.9440
0.9119	0.9260	0.9380
0.8776	0.8837	0.9206
0.8963	0.9088	0.9389
0.8792	0.9045	0.9331
0.8884	0.9199	0.9407
0.8835	0.8935	0.9421

Table 5: auc roc of Gaussian Naive Bayes, SMV and Random Forest Classifier

As for *accuracy* and *f1-score* to better compare the three algorithms it was calculated the average, the maximum and minimum value and the standard deviation of *auc roc* of 10 values for each algorithm.

The result is shown in the table 6:

	Gaussian Naive Bayes	**SMV**	**Random Forest Classifier**
Mean	0.8929	0.9077	0.9381
Maximum	0.9119	0.9260	0.9459
Minimum	0.8776	0.8837	0.9206
Standard deviation	0.0107	0.0118	0.0067

Table 6: auc roc of mean, maximum, minimum and standard deviation of Gaussian Naive Bayes, SMV and Random Forest Classifier

As for *accuracy* and *f1-score*, from this analysis it can be seen that the *auc roc* of the *Random Forest Classifier* is better than the other two algorithms. This because comparing the mean of the three algorithms I can see that the highest mean is the mean of the *Random Forest Classifier* (0.9381) in fact the maximum and minimum values are close (maximum = 0.9459 and minimum = 0.9206). *The Random Forest Classifier* is also the best algorithm because it has the high mean and minimum value, in fact the algorithm is best if the mean value is greater and lesser is the probability of making errors in a single test if the minimum value is high. This result is supported by the standard deviation in fact the standard deviation of the Random Forest is lower than the other algorithms. This means that the data calculated with *Random Forest Classifier* algorithm are close to the mean and then the each data is not very different to the mean.

3 Conclusion

From the Section 2 (**Performance evaluation**) it can see that for each method (*accuracy*, *f1-score* and *auc roc*) the best algorithm is *Random Forest Classifier*.