Machine Learning, assignment no. 2

ALAN RAMPONI, student ID: 179850, University of Trento

This assignment is about the performance comparison of three different classification algorithms: naive Bayes, support vector machine and random forest on an artificial binary classification dataset created on purpose. After generating the dataset, the three different algorithms were trained using the k-fold cross validation (with k=10). For the last two mentioned algorithms an inner k-fold cross validation (with k=5) was needed in order to choose the best parameter among those there were given. The evaluation was performed using some well-known metrics such as accuracy, F1-score and AUC ROC.

Categories and Subject Descriptors: [**Computing Methodologies**]: Machine Learning—*Naive Bayes, Support vector machines, Classification trees, Cross-validation*

1. INTRODUCTION

In Machine Learning, classification is the problem of categorizing new observations in a fixed set of categories given a training set containing samples whose category membership is known. In the last few decades, several algorithms were developed in order to satisfy this very purpose. In this assignment are described some of these techniques that were compared each other to accomplish a common goal. An artificial dataset was generated in order to meet our needs, and the algorithms were trained in it, using the *k-fold cross validation*. The evaluation step, described in depth later, was performed using *accuracy*, *F1-score* and *AUC ROC* as measures of performances.

2. THE ARTIFICIAL DATA AND ITS SAMPLING

The first step was to create an artificial binary classification dataset in order to train the algorithms in it. To meet the goal, the dataset was purposely generated using the *scikit-learn* library [Cournapeau 2007] (used also for almost the entire code), one of the most famous ML Python libraries. The number of samples and features can be chosen by command line by typing $python\ assignment\_2.py\ 1000\ 10$ (i.e. the values requested for this task, $n\_samples$ and $n\_features$ respectively)[1]. An example of the resulted artificial data can be seen in Table I.

Table I. Artificial data set: a subset of the data generated.

feature_1	**feature_2**	**feature_3**	**...**	**feature_10**	**TARGET**
0.15899894	-0.46799283	-0.30431862	...	0.34664386	*Class 1*
-0.16077326	1.37803962	0.67200051	...	0.93731737	*Class 0*
-0.78435635	-1.50706574	0.80746839	...	1.11410509	*Class 0*
...	...	...	...	...	...

According to our purposes, the original dataset was splitted into some subsets named training and test sets using the *k-fold cross validation*, with k=10, shuffling the data before splitting it into batches. This process was repeated 10 times, according to the parameter k, to train the different classifiers on each of the training subsamples of data and by evaluating their performances by comparing the

[1]The results presented in this report refers to an execution with a dataset of dimension 1000 (observations) x 10 (features).

expected class of the remaining test subsamples with the predicted one. Then, the overall performance of each algorithm was given by the average of the performances of all k iterations. This allowed to give us a more robust analysis of the outcomes.

3. THE CLASSIFICATION TASK

In order to predict the output classes, different algorithms were trained on the artifical data. In particular, the classification algorithms that were used are *naive Bayes*, *support vector machine* and *random forest*. Each of them has different peculiarities:

3.1 Naive Bayes classifier

A naive Bayes is one of the simplest probabilistic classifiers based on the Bayesian theorem. First, it evaluates the prior probabilities of the labels that are based on previous experience. Thus, given a binary classification problem in which the labels are A and B, their prior probabilities are the following:

$$P_{prior}(A) = \frac{|A|}{|A| + |B|} \qquad P_{prior}(B) = \frac{|B|}{|A| + |B|}$$

Second, in order to classify a new sample, the algorithm computes a likelihood measure using the k nearest points (chosen a priori) in the space where the new sample is placed. The likelihood is given by the amount of points in the neighborhood belonging to each label over the total amount of elements belonging to the label that is taken into account. Given a sample x, the likelihood of x given A and the likelihood of x given B are respectively the following:

$$L(A|x = A) = P(x = A|A) = \frac{|A_{neighborhood}|}{|A|} \qquad L(B|x = B) = P(x = B|B) = \frac{|B_{neighborhood}|}{|B|}$$

Then, the classification of the sample x is given by the combination of both the prior probability and the likelihood, in order to form a posterior probability using the Bayes' rule:

$$P_{posterior}(x = A) = P_{prior}(A) * L(A|x = A) \qquad P_{posterior}(x = B) = P_{prior}(B) * L(B|x = B)$$

This gives us the output label of the sample x, that is determined by the label that achieves the largest posterior probability. In this assignment a *Gaussian* naive Bayes is used because the continuous values associated with each class are distributed according to a Gaussian distribution.

3.2 Support vector machine classifier

A support vector machine is a non-probabilistic binary linear classifier that defines a discriminative hyperplane that is used by the algorithm in order to categorize new examples into one category or the other. In fact, giving a set of training examples, the algorithm builds a model that is a mapping of the examples as points in space, whose classes are divided by a margin that is as wide as possible. Thus, new examples are putted into the same space and they are predicted to belong to a class based on which side of the margin they fall on.

In this assignment a list of C penalty parameters of the error term ($1e{-}02, 1e{-}01, 1e00, 1e{+}01, 1e{+}02$) was provided. Each of them represents the trade-off between the misclassification of training examples against the simplicity of the decision surface of the classifier. In particular, small C values makes the surface smooth, while high C values lead to classifying all training examples correctly by giving the model freedom to choose more samples as support vectors. In order to choose the best C penalty parameter an inner k-fold cross validation (with k=5) was performed. The choice was determined by the value that gave us the higher *F1-score*. Besides, the kind of kernel function that was used in this learning algorithm is the (Gaussian) *radial basis function* (RBF).

3.3 Random forest classifier

A random forest represents an aggregation of learning methods for classification or regression, that operates by constructing a multitude of decision trees at training time and that outputs the class that is the mode of the classes (for classification tasks) or mean prediction (for regression tasks) of the individual trees. In this very way, weak learners such as decision trees cooperate in order to form a stronger and unique learner.

In this assignment a list of *n_estimators* $(10, 100, 1000)$ was provided. Estimators correspond to the number of decision trees in the forest. Using an inner k-fold cross validation (with k=5) it was possible to choose the best number of decision trees (i.e. the one that gave us the highest *F1-score*). Moreover, the chosen function to measure the quality of a split in decision trees within the random forest is the *Gini impurity*.

4. THE EVALUATION

In order to evaluate the performances of each algorithm in this classification problem, three different goodness measures were used: *accuracy*, *F1-score* and *AUC ROC*. In depth, **accuracy** is a measure of how well a binary classification identifies or excludes a condition. In other words, it is the proportion of true (both positives and negatives) among the total number of cases that are examinated:

$$Accuracy = \frac{|TP| + |FN|}{|TP| + |FP| + |TN| + |FN|}$$

F1-score is another measure of binary classification goodness that involves other two concepts: *precision* (the number of true positives over the number of all positives) and *recall* (the number of true positives over the number of positives that should have been returned). It represents a kind of weighted average of the previous two concepts, and it reaches its best value at 1 and its worst value at 0. Formally, F1-score is defined as follows:

$$F1_{score} = 2 * \frac{Precision * Recall}{Precision + Recall}$$

The last measure used is the **AUC ROC** (*Area Under the Curve, Receiver Operating Characteristic*). ROC is a plot that explains the performances of a binary classifier as its discrimination threshold is varied. The curve is created by plotting the true positives against the false positives at various threshold settings. In particular, AUC is the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming $positive_{ranks} > negative_{ranks}$).

The results of each of the previous measures were collected for each learning algorithm in each of the k iterations of the k-fold cross validation, with k=10. After that, the final classification performance of each algorithm was given by the average of the performances on each iteration.

5. DISCUSSION

Using the methodology explained before, all the three different algorithms were evaluated. In order to do that, the script was executed several times to assess the results obtained. Four of them are represented in Figure 1, Figure 2, Figure 3 and Figure 4. As it is possible to see, the best classification algorithm is represented by the random forest, that achieved the best results in accuracy, F1-score and AUC ROC in each of the four executions. The support vector machine classifier performed slightly worse but it maintained good performances. The worst one is the naive Bayes classifier that, except for the third execution (Figure 3) in which it obtained a slightly better results than the SVM in accuracy and in AUC ROC, it performed poorly compared to the other two classification algorithms.

In the end, it is possible to state that the random forest classifier is the best one for this kind of task.

APPENDIX

In this section there are some screenshots representing the evaluation results on different executions. In particular, we can see that the random forest classifier performs better in each execution.

```
======================
Performance evaluation:
======================

ACCURACY:
=================================================================
iter    Naive Bayes             SVM                 Random Forest
1       0.95000000              0.94000000          0.99000000
2       0.94000000              0.99000000          0.97000000
3       0.90000000              0.92000000          0.97000000
4       0.98000000              0.99000000          0.99000000
5       0.96000000              0.94000000          0.96000000
6       0.92000000              0.95000000          0.99000000
7       0.98000000              0.98000000          0.98000000
8       0.91000000              0.94000000          0.97000000
9       0.96000000              1.00000000          1.00000000
10      0.94000000              0.96000000          0.99000000
=================================================================
AVG     0.94400000              0.96100000          0.98100000

F1-SCORE:
=================================================================
iter    Naive Bayes             SVM                 Random Forest
1       0.95575221              0.94444444          0.99082569
2       0.94000000              0.98969072          0.96907216
3       0.91379310              0.92592593          0.97247706
4       0.98000000              0.98969072          0.98969072
5       0.96491228              0.94642857          0.96363636
6       0.91111111              0.94505495          0.98850575
7       0.97826087              0.97826087          0.97777778
8       0.91262136              0.93750000          0.96969697
9       0.96296296              1.00000000          1.00000000
10      0.93877551              0.95833333          0.98924731
=================================================================
AVG     0.94581894              0.96153295          0.98109298

AUC ROC:
=================================================================
iter    Naive Bayes             SVM                 Random Forest
1       0.94646465              0.94141414          0.99090909
2       0.94150641              0.99038462          0.97035256
3       0.89292929              0.92121212          0.97070707
4       0.98039216              0.98979592          0.98979592
5       0.95555556              0.93737374          0.95959596
6       0.92126623              0.95292208          0.98863636
7       0.98181818              0.98181818          0.97979798
8       0.91266026              0.93990385          0.97115385
9       0.95974235              1.00000000          1.00000000
10      0.94219189              0.96105982          0.98936170
=================================================================
AVG     0.94345270              0.96158845          0.98103105
```

Fig. 1. Results of the first execution.

```
======================
Performance evaluation:
======================

ACCURACY:
=================================================================
iter    Naive Bayes             SVM                 Random Forest
1       0.92000000              0.95000000          0.95000000
2       0.84000000              0.94000000          0.96000000
3       0.89000000              0.97000000          0.96000000
4       0.85000000              0.96000000          0.97000000
5       0.92000000              0.95000000          0.96000000
6       0.83000000              0.90000000          0.90000000
7       0.83000000              0.93000000          0.93000000
8       0.89000000              0.96000000          0.95000000
9       0.93000000              0.94000000          0.96000000
10      0.89000000              0.92000000          0.97000000
=================================================================
AVG     0.87900000              0.94200000          0.95100000

F1-SCORE:
=================================================================
iter    Naive Bayes             SVM                 Random Forest
1       0.93103448              0.95652174          0.95726496
2       0.84313725              0.94230769          0.96153846
3       0.90434783              0.97435897          0.96551724
4       0.81481481              0.95000000          0.96296296
5       0.93548387              0.96000000          0.96875000
6       0.83495146              0.90384615          0.90384615
7       0.80459770              0.91954023          0.92134831
8       0.90090090              0.96296296          0.95412844
9       0.93203883              0.93877551          0.96078431
10      0.87356322              0.90697674          0.96551724
=================================================================
AVG     0.87748704              0.94152900          0.95216581

AUC ROC:
=================================================================
iter    Naive Bayes             SVM                 Random Forest
1       0.91396104              0.94561688          0.94318182
2       0.83993597              0.93957583          0.95958383
3       0.88637291              0.96511628          0.95634435
4       0.84930643              0.96258932          0.97540984
5       0.91977692              0.94916345          0.95152295
6       0.83480530              0.90566038          0.90566038
7       0.83360893              0.93695742          0.94067797
8       0.88701923              0.95833333          0.94791667
9       0.93000000              0.94000000          0.96000000
10      0.89203612              0.92118227          0.97413793
=================================================================
AVG     0.87868229              0.94241952          0.95144357
```

Fig. 2. Results of the second execution.

```
======================
Performance evaluation:
======================

ACCURACY:
=====================================================================
iter    Naive Bayes             SVM                     Random Forest
1       0.85000000              0.86000000              0.95000000
2       0.85000000              0.82000000              0.89000000
3       0.84000000              0.78000000              0.87000000
4       0.86000000              0.81000000              0.89000000
5       0.86000000              0.93000000              0.93000000
6       0.88000000              0.83000000              0.89000000
7       0.81000000              0.89000000              0.88000000
8       0.85000000              0.85000000              0.94000000
9       0.79000000              0.81000000              0.89000000
10      0.83000000              0.81000000              0.90000000
=====================================================================
AVG     0.84200000              0.83900000              0.90300000

F1-SCORE:
=====================================================================
iter    Naive Bayes             SVM                     Random Forest
1       0.84536082              0.86792453              0.95412844
2       0.84848485              0.83928571              0.89908257
3       0.80000000              0.77551020              0.84337349
4       0.81081081              0.77108434              0.87058824
5       0.84444444              0.93069307              0.92929293
6       0.86363636              0.82828283              0.88421053
7       0.77647059              0.88421053              0.87234043
8       0.84210526              0.85148515              0.94117647
9       0.75862069              0.80412371              0.88888889
10      0.81720430              0.82568807              0.90196078
=====================================================================
AVG     0.82071381              0.83782881              0.89850428

AUC ROC:
=====================================================================
iter    Naive Bayes             SVM                     Random Forest
1       0.86607143              0.86525974              0.95292208
2       0.85959596              0.81616162              0.88989899
3       0.82930757              0.78341385              0.86191626
4       0.84363178              0.81189575              0.90058848
5       0.86234494              0.93017207              0.93057223
6       0.87660256              0.83092949              0.88942308
7       0.81000000              0.89000000              0.88000000
8       0.85000000              0.85000000              0.94000000
9       0.79291717              0.81092437              0.89055622
10      0.83721397              0.80750703              0.90204737
=====================================================================
AVG     0.84276854              0.83962639              0.90379247
```

Fig. 3. Results of the third execution.

```
======================
Performance evaluation:
======================

ACCURACY:
=====================================================================
iter    Naive Bayes             SVM                     Random Forest
1       0.87000000              0.85000000              0.93000000
2       0.90000000              0.94000000              0.96000000
3       0.86000000              0.89000000              0.94000000
4       0.92000000              0.95000000              0.98000000
5       0.88000000              0.93000000              0.98000000
6       0.91000000              0.95000000              0.97000000
7       0.88000000              0.96000000              0.97000000
8       0.91000000              0.94000000              0.95000000
9       0.95000000              1.00000000              1.00000000
10      0.88000000              0.91000000              0.96000000
=====================================================================
AVG     0.89600000              0.93200000              0.96400000

F1-SCORE:
=====================================================================
iter    Naive Bayes             SVM                     Random Forest
1       0.87619048              0.85436893              0.93333333
2       0.90000000              0.94117647              0.96078431
3       0.83720930              0.86419753              0.92682927
4       0.91666667              0.94623656              0.97826087
5       0.87500000              0.92631579              0.97916667
6       0.92682927              0.96000000              0.97600000
7       0.89285714              0.96428571              0.97297297
8       0.91428571              0.94545455              0.95327103
9       0.94736842              1.00000000              1.00000000
10      0.85365854              0.89411765              0.95238095
=====================================================================
AVG     0.89400655              0.92961532              0.96329994

AUC ROC:
=====================================================================
iter    Naive Bayes             SVM                     Random Forest
1       0.87373737              0.85555556              0.93434343
2       0.90144231              0.94070513              0.96073718
3       0.86666667              0.88750000              0.94166667
4       0.92270531              0.95048309              0.97987118
5       0.87980769              0.92948718              0.97996795
6       0.90311055              0.94052123              0.96153846
7       0.87824675              0.95941558              0.97077922
8       0.91720779              0.94155844              0.95535714
9       0.95042152              1.00000000              1.00000000
10      0.87356322              0.90927750              0.95894910
=====================================================================
AVG     0.89669092              0.93145037              0.96432103
```

Fig. 4. Results of the fourth execution.

REFERENCES

David Cournapeau. 2007. Scikit-Learn: Machine Learning in Python. Python library. (2007). http://scikit-learn.org/stable/