DK911b-Machine_Learning-Lab1

October 16, 2019

1 scikit-learn Lab

- scikit-learn is the leading machine learning software in Python
- scikit-learn is a project started in Paris, Inria and Telecom Paris
- scilkit-learn is easy to use and extend

2 I Tutorial

2.1 1. Install scikit-learn:

- https://scikit-learn.org/stable/install.html

2.2 2. Follow the scikit-learn tutorial

- https://scikit-learn.org/stable/tutorial/basic/tutorial.html

2.3 3. Train your first classifier

2.3.1 Import packages

```
In [3]: import numpy as np
        from sklearn import datasets
```

2.3.2 Load and parse datafile

```
In [4]: iris = datasets.load_iris()
        iris_X = iris.data
        iris_y = iris.target
        print(len(iris_X))
        np.unique(iris_y)
```

```
150
```

```
Out[4]: array([0, 1, 2])
```

2.3.3 Split data into training and test sets

```
In [5]: # Split iris data in train and test data
        # A random permutation, to split the data randomly

        np.random.seed(0)
        indices = np.random.permutation(len(iris_X))
        iris_X_train = iris_X[indices[:-10]]
        iris_y_train = iris_y[indices[:-10]]
        iris_X_test = iris_X[indices[-10:]]
        iris_y_test = iris_y[indices[-10:]]
```

2.3.4 Train a k-nearest neaighbors model

```
In [6]: # Create and fit a knearestneighbor classifier
        from sklearn.neighbors import KNeighborsClassifier
        knn = KNeighborsClassifier()
        knn.fit(iris_X_train, iris_y_train)

Out[6]: KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                             metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                             weights='uniform')
```

2.3.5 Evaluate model on test instances and compute test error

```
In [7]: from sklearn.metrics import accuracy_score
        knn.predict(iris_X_test)

Out[7]: array([1, 2, 1, 0, 0, 0, 2, 1, 2, 0])

In [8]: iris_y_test

Out[8]: array([1, 1, 1, 0, 0, 0, 2, 1, 2, 0])

In [9]: accuracy_score(iris_y_test, knn.predict(iris_X_test))

Out[9]: 0.9
```

3 II Task 1

### - What is the error of the KNN classifier trained in previous step? ### - What is the optimal parameter k for KNN classifier for iris dataset?

```
In [10]: #write a function predict(k) in which k is the number of nearest neighbors and it ret
         def predict(k):
             knn = KNeighborsClassifier(n_neighbors=k)
             knn.fit(iris_X_train, iris_y_train)
             knn.predict(iris_X_test)
             return accuracy_score(iris_y_test, knn.predict(iris_X_test))
         k_max = 140
         k_min = 1
         k_best = 0
         max_accuracy = 0

         for k in range(k_min, k_max+1):
             acc = predict(k)
             if(acc>max_accuracy):
                 max_accuracy = acc
                 k_best = k

         print(k_best, max_accuracy)

         # in fact, accuracy score is always 1.0 when k changes from 8 to 24. The Quantity of
```

```
8 1.0
```

4 III Task 2

4.0.1 - Train another two classifiers for iris dataset. The documentation for supervised learning methods available in scikit-learn: https://scikit-learn.org/stable/supervised_learning.html#supervised-learning

4.0.2 - Use cross-validation to evaluate classifiers.

4.0.3 - Compare evaluation results of the three classifiers.

```
In [11]: from sklearn import svm
         from sklearn.model_selection import cross_val_score

         clf = svm.SVC(gamma='scale')
         clf.fit(iris_X_train, iris_y_train)
         print(accuracy_score(iris_y_test, clf.predict(iris_X_test)))
         print(cross_val_score(clf, iris_X_train, iris_y_train, cv=5))

         from sklearn.linear_model import SGDClassifier
         clf = SGDClassifier()
         clf.fit(iris_X_train, iris_y_train)
         print(accuracy_score(iris_y_test, clf.predict(iris_X_test)))
         print(cross_val_score(clf, iris_X_train, iris_y_train, cv=5))

         #If we make X_train : X_test = 14 : 1, the accuracy of KNN can be 1, the other 2 clas
         #If we change the number of two sets, there is little difference between the accuracy
```

```
0.9
[1.         0.85714286 1.         1.         0.92592593]
0.9
```

```
[1.         0.82142857 0.75       0.96296296 0.92592593]
```

5 Submission:

5.0.1 This lab is due on 16th of October, 2019. Your report, in the form of Jupyter Notebook and pdf, send on: filippo.miatto@telecom-paristech.fr