SpamDetector

Project for Data Mining and Machine Learning

Francesco Campilongo

Contents

Chapter 1

Introduction

The purpose of this documentation is to prove the machine learning algorithms utility for the detection of spam, useless messages, into dataset of emails.
This is a basic text classification problem, and so the algorithm used are very known in the lecture.
The project is carried out on python.

Chapter 2

Datasets

The datasets used for this project are two:

- E-mail dataset
- SMS dataset

Those two datasets where found on Kaggle.com both of them in a .csv format.

2.1 Data Preprocessing

The datasets found were already pretty good for the type of result this project wants to achieve. The E-mail dataset is characterize from three column a "label", "text" and "label_num", the second column contains the actual message, the first and third column are the same, but in the first one there are the actual words "ham", for the messages which are not classified as spam, and "spam", instead in the third column there is there are "0" for the ham messages and "1" for the spam messages;
The first column has been discharged since not useful for the execution of the classification algorithm.
The SMS file had need a little more work to obtain a dataset usable, because the it has five columns named "v1", "v2", "", "" and "", the first contains the words "ham", for the messages which are not classified as spam and "spam"; the second column contains the actual messages and the last three columns are basically blank, so in order to obtain a good dataset to work with the last three columns has been discharged the first two are taken in consideration, but in the first one the "ham" word is changed with a "0" and the "spam" word with a "1", in order to have the same kind of datasets between the E-mail and the SMSs.

Chapter 3

Design

The application design is pretty basic, it is written in python (version 3) does not involve any classes, just the creation of some functions in order to keep the code clean and more understandable.

3.1 Implementation

3.1.1 Libraries

The libraries involved into this application are two:

- Pandas
- Sklearn

Pandas gives the possibility to use a dataset in .csv (and many others) from the disk, implementing it on a dataframe into the main memory to use all the machine learning algorithm that are imported from Sklearn.

3.1.2 Functions

The functions are meant to implement all the algorithm used, which are the following:

- Support Vector Machine
- Multinomial Näive Bayes
- K Nearest Neighbor
- Random Forest
- AdaBoost

These functions are written to use them with different datasets, the SMS dataset and the E-Mail dataset as already mention earlier.
There is also another function that is called to show the result of all the algorithms.

3.1.3 Main code

This section is used to show some insight on the main code.
Here will be shown a bit of the dataset cleaning and processing that has been made, and described into the Data Preprocessing chapter.

```
df2 = pd.read_csv("sms_spam.csv", encoding = "ISO-8859-1")
dfsp = df2[['v1', 'v2']]
dfsp.loc[dfsp["v1"] == 'ham', "Category"] = 0
dfsp.loc[dfsp["v1"] == 'spam', "Category"] = 1
dfsp = dfsp.rename(columns={'v2': 'Content'})
dfs = dfsp[['Content', 'Category']]
```

The following code will give a look at one of the function, the one that implement the Support Vector Machine. Since these functions are quite similar there is no need to show them all here.

```
def SVM(x_train, y_trainFeat, y_testFeat):
  classifier = LinearSVC()
  classifier.fit(y_trainFeat, x_train)
  predRes = classifier.predict(y_testFeat)
  return predRes
```

The following code lines will show the functions calls, also in this case in order to avoid repetitions only few functions call will be shown.

```
# KNN
predResMailKNN = KNN(x_train, y_trainFeat, y_testFeat)
# Metrics and results
print("\tK Nearest Neighbors RESULTS")
print("Neighbors Number: 1")
show_res(actual_x, predResMailKNN)

# RF
predResMailRF = RF(x_train, y_trainFeat, y_testFeat)
# Metrics and results
print("\tRandom Forest RESULTS")
show_res(actual_x, predResMailRF)
```

Chapter 4

Results

The results are as expected very good .....scrivere roba riguardante il motivo per il quale i risultati sono expected magari elencando alcune peculiarità degli algoritmi
The metrics that shows the results goodness are: Accuracy, F Score and Confusion Matrix; Chosen in order to give a deep look at the algorithms behavior.
Here the results for what concern the E-mail dataset:

```
/-------------------SpamDetector for e-Mail-------------------/
Support Vector Machine results
Accuracy Score: 98.6473
F Score:  98.3941
Confusion Matrix:
[[716  13]
[  1 305]]

Multinomial Naive Bayes results
Accuracy Score: 91.3043
F Score:  88.5304
Confusion Matrix:
[[727   2]
[ 88 218]]

K Nearest Neighbors results
Neighbors Number: 1
Accuracy Score: 95.8454
F Score:  94.9084
Confusion Matrix:
[[718  11]
[ 32 274]]

Random Forest results
Accuracy Score: 95.4589
F Score:  94.5432
Confusion Matrix:
[[706  23]
[ 24 282]]

Adaboost results
Estimators Number: 100
Accuracy Score: 96.9082
F Score:  96.3159
Confusion Matrix:
[[709  20]
[ 12 294]]
```

Here the results for what concern the SMS dataset:

```
/----------------------SpamDetector for SMS--------------------/
Support Vector Machine results
Accuracy Score: 98.5650
F Score:  96.8657
Confusion Matrix:
[[960   0]
[ 16 139]]

Multinomial Naive Bayes results
Accuracy Score: 97.1300
F Score:  93.4249
Confusion Matrix:
[[960   0]
[ 32 123]]

K Nearest Neighbors results
Neighbors Number: 1
Accuracy Score: 94.7982
F Score:  87.0259
Confusion Matrix:
[[960   0]
[ 58  97]]

Random Forest results
Accuracy Score: 97.4888
F Score:  94.3510
Confusion Matrix:
[[959   1]
[ 27 128]]

Adaboost results
Estimators Number: 100
Accuracy Score: 97.7578
F Score:  95.0885
Confusion Matrix:
[[956   4]
[ 21 134]]
```