"Alexandru Ioan Cuza" University of Iași
Faculty of Computer Science

Support Vector Machines

May 2020

Contents

1 SVM Algorithm **2**
1.1 Approach 2
1.2 Algorithm 2
1.2.1 Training 2
1.2.2 Classifying the new set of data 2
1.3 Optimization 2
1.4 Statistics 3
1.4.1 Accuracy 3
1.4.2 Time 4
1.5 Conclusion 5

1 SVM Algorithm

1.1 Approach

Our module takes the tweets from the common database as a training set for the SVM algorithm and integrates into the application the trained model and predictions . After the training is done, we use our model to predict if a new given tweet is either fake or true. The algorithm is based on analyzing the selected features of the previous dataset and the verdict labeled to them.

1.2 Algorithm

1.2.1 Training

Given a set of training examples, a Support Vector Machine is capable of building a trained model in order to classify a new set of data. In the training process we use and modify labeled data so it fits our algorithm, almost 6000 true and fake tweets, each of them being characterised by integers representing the tweet's positivity, the important words rate, the number of uppercase and lowercase letters, the number of words starting with a capital, the number of words containing capitals only, and the verdict: 0, in case the tweet is fake, 1 in case the tweet is true.

By analyzing this data, a trained model was built, ready to classify a new set of tweets as being true or fake.

1.2.2 Classifying the new set of data

The information, for training the model and deciding which features are more significant, is retrieved using tf-idf method . By doing this, the words with low frequency in the text form of tweets are better analyzed. When given a new data set, our algorithm successfully classify the tweets in 2 categories: fake and true.

1.3 Optimization

Optimization is especially important for a machine learning based module, and that is the reason data is well organized and selected before training . Moreover, Grid Search is an important tool used in our module, in order to optimize precision. It was used to dynamically determine the most optimal parameters for the SVC function, which was used to train the model . Another important aspect for time optimization is creating a background process for the main functions of the module, synchronise() and predict () .

1.4 Statistics

1.4.1 Accuracy

The accuracy of our results is influenced mostly by the parameters used by the SVC function. Parameters like kernel, degree, gamma can lead to different results depending on the data we use the algorithm on, this is why it is very important to find the right parameters. In order to do this, we have implemented a function using GridSearchCV and a list of all the parameters. Each possible parameter is used with a different value multiple times. Eventually the perfect combination is returned: for our set of data, the parameters that lead us to the most accurate results are c=0.1, gamma=0.2, kernel='rbf'. The precision for 0 labeled tweets is 1.0 and the precision for 1 labeled tweets is 0.92. The following bar chart shows the average precision resulted by using different parameters:

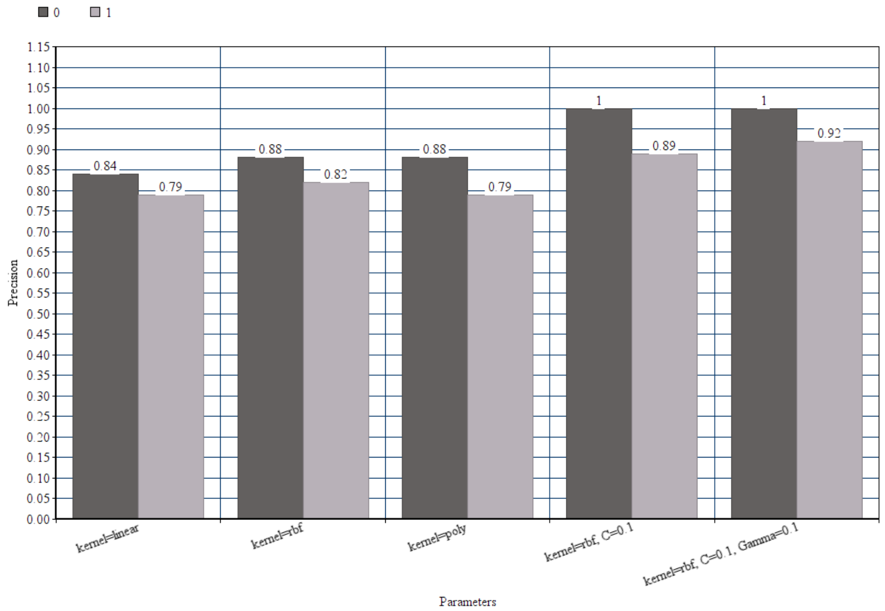

Figure 1: Precision statistics

1.4.2 Time

We don’t train our model everytime we run the program. We search it and load it from the database. The training procedure begins only if the model does not exist in the database (or in case it isn’t successfully loaded). Furthermore, the ‘synchronize’ and ‘predict’ functions run on a background process, ‘worker.py’. This way, the running time is decreased.

The whole project was run 135 times. This way we found out that the average running time of our module is 2.96 seconds. The following pie chart contains the running times of our module during the whole experiment.

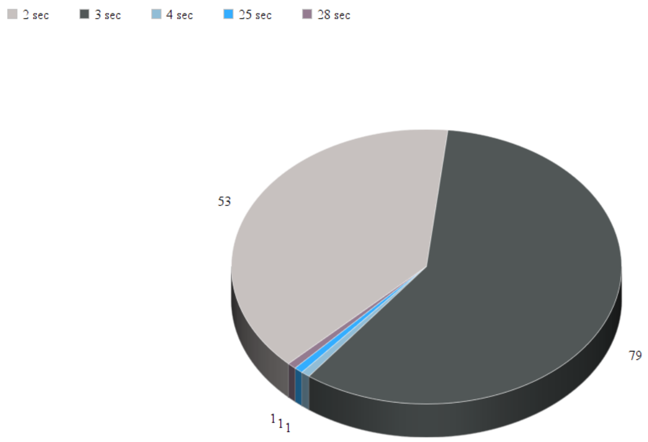

Figure 2: Time statistics

1.5 Conclusion

Our module successfully integrates a trained model and prediction into the application using some specific methods for a high optimization in time and precision . The most important of them are GridSearch, vectorizing the input with tf-idf, balancing the dataset and creating a background process when included to the final application. It successfully predicts the verdict for a new data set, having the best possible accuracy.