5-10 pages

**Problem**

Out project is inspired by the Kaggle-competition “Partly Sunny with a Chance of Hashtags"[[1]](#footnote-1). The task is to classify tweets talking about the weather in time, sentiment and kind of weather (“cloudy”, “sunny” etc.). For simplicity, we will focus only on kind of weather. This choice is sensible as we are provided with a dataset containing only tweets concerning weather. For temporal or sentiment classification it would be more natural to use a general dataset.

Our motivation for the choice of this problem stems from our interest in Natural Language Processing. Furthermore, we decided to implement a Naïve Bayes Classifier ourselves instead of using a toolkit because this helps us to understand it more thoroughly.

**Data**

The data we will use are annotated tweets obtained from Kaggle, which contain weather information. The data, along with a description, can be obtained here: <https://www.kaggle.com/c/crowdflower-weather-twitter/data>

The data has the following format (tweet ID, text, location):

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **tweet** | **state** | **location** |
| 4 | Edinburgh peeps is it sunny?? #weather | | birmingham |
| 5 | SEEVERE Tâ€™STORM WARNING FOR TROUSDALE, NORTHWESTERN CLAY, MACON, SOUTHEASTERN SUMNER <http://bit.ly/g6ZQzw> | | Nashville |
| 7 | @Agilis1 sport or traditional climbing? Thats intense. Send some of that weather to the north! | | Midwest |
| 8 | #WEATHER: 10:53 am : 63.0F. Feels 61F. 30.07% Humidity. 16.1MPH South Wind. | tennessee | Nashville, TN, USA |

**Evaluation of Results**

Our classifier can be tested by uploading its predictions to kaggle. The “kind of weather”-task requires to label the tweets with the following classes:

k1,"clouds"

k2,"cold"

k3,"dry"

k4,"hot"

k5,"humid"

k6,"hurricane"

k7,"I can't tell"

k8,"ice"

k9,"other"

k10,"rain"

k11,"snow"

k12,"storms"

k13,"sun"

k14,"tornado"

k15,"wind"

Multiple classes for one tweet are possible. Kaggle requires not to upload a classification but probabilities for each class. For this reason it comes in handy that we decided to implement Naïve Bayes ourselves as we can modify it to output probabilities instead of classes. The following sample shows the desired format for predictions, probabilities for each of the 15 possible classes:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TweetID** | **k1** | **k2** | **k3** | **k4** | **k5** | **k6** | **k7** | **k8** | **k9** | **k10** | **k11** | **k12** | **k13** | **k14** | **k15** |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0.604 | 0 | 0.196 | 0 | 0 | 0 | 0.201 | 0 | 0 |
| 11 | 0 | 0.203 | 0.176 | 0 | 0.376 | 0 | 0.421 | 0 | 0.176 | 0 | 0 | 0 | 0 | 0 | 0.579 |
| 13 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

In this example many of the probabilities happen to add up to 1 in total but this is a coincidence; the the sum of the probabilities for tweet 11 is 1.931 for example.

**Approach**

**Solution**

**Evaluation of Approach and Solution**

**INTERMEDIATE REPORT:**

**Methods**

We will first preprocess the tweets: tokenize words, do simple spelling correction, use the nltk-toolkit for stemming. Then, we will use the bag-of-words approach to obtain the features. A Naïve Bayes Classifier will be fed with these features and output the likelihood for each class separately. The predictions are done separately because the prediction-labels are not dependent on each other, several labels can be predicted for one tweet.

(See: <http://www.kaggle.com/c/crowdflower-weather-twitter>).

**Possible Obstacles**

**Word Misspelling/Stemming:** As twitter language is often informal words are often written in many different (“wrong”) forms which impedes the performance of the classifier. For example *hot* might be written as *hottttt* to emphasize the intensity*.* A possibility to solve this problem with duplicate characters is to reduce all sequences of the same character longer than two characters to one character. Random misspellings could only be cured with a word lexicon which is outside the scope of this project. Moreover, one might want to stem words such that e.g. *cloudy, cloudier,* and *cloud* become *cloud.* This can be done using the nltk-toolkit.

**Kaggle-Evaluation:** As the Kaggle-competition is meant for three classifications (temporal, sentiment, kind of weather) it might be that the results for predictions will only give overall results. As we will only upload predictions for the kind of weather this would make it impossible to test. However, it might be that Kaggle also outputs results for each classifier separately. If not, we will have to separate the training set into test set and training set and use this data instead of uploading to Kaggle.

**Computation-Time:** The bag-of-words approach produces as many features as there are words in the dictionary. A preliminary analysis of our data (not using stemming or spelling corrections) reveals that approximately 10.000-20.000 words are used. Therefore, our classifier might take a lot of computation time.

1. https://www.kaggle.com/c/crowdflower-weather-twitter [↑](#footnote-ref-1)