Python Machine Learning Final Project

Real or not, NLP with disaster tweets

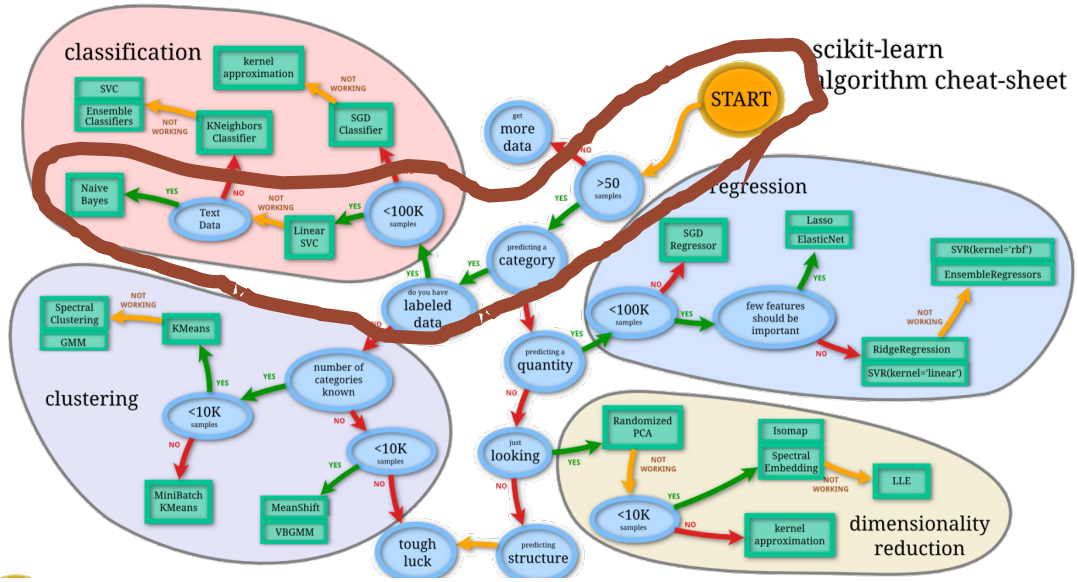
106321001 楊智成
107321002 陳靖雯
107321004 蕭名誼

- Goal:
  - To determine whether the disasters mentioned within the input tweets are real or not.
- Feature engineering:
  - Each tweet has their five entries of information:
    - id
    - location
    - key word
    - text
    - target
  - We use inbuilt function to detect if all tweets data for a single entry as values. Unfortunately, only text and target entries exist for all data. Other columns like location, key word, both lack information for some tweets, namely "null". Besides, it is quiet hard for us to simply fill in the blank, we can neither borrow information from other rows, nor extract desire information just by text data. Moreover, we do not prefer eliminate any tweet data. So, we choose text and target to be our input data.
  - By the way, we discover that the positive (label=1) and negative (label=0) instances for text entry are about equal.
- Data analysis:
  - We first convert all words within texts into lowercase and do spelling check.
  - We remove some elements that may distort our texts:
    - Square brackets: []
    - Angle brackets: <>
    - Newline: \n
    - Punctuation: , . ! ?
    - Numbers: 13000
    - Links and Special Characters
    - Stop words: "the", "is", "in", "for", "where", "when", "to", "at", … etc.
  - We also lemmatize our texts:
    We tokenize our text into separate words, then loop through each word within each text. We detect the part of speech for that word and return to its original form. For nouns, if it is a plural, we

eliminate the "s"; For verbs, if it is in past tense, v.ing or v.pp, then we return it to simple present tense.

- Unique words:
  - From original of 32017 words down to 14411 words. ☺

- Model choosing:
  - Following the advice path cheat sheet suggested,



  - We choose support vector machine classifier and Naïve Bayes as our training model. In addition, we are curious about how logistic regression would perform, so we do that as well.

- Grid search (for model fine-tuning):
  - We use grid search to do fine-tuning, to search for the best set of hyper parameter and feed into our model.
  - Logistic regression for instance:

```
(7613, 14411)
{'C': 50, 'dual': False, 'multi_class': 'ovr', 'penalty': 'l2', 'solver': 'sag', 'tol': 0.1}
0.6487339307345059
```

- Result:
  - Before data filtering:

	Average Validation Score	Precision	Recall
Logistic Regression	0. 7323	70.43%	64.96%
Support Vector Classifier	0. 7179	72.16%	55.94%
Multinomial Naïve Bayes	0. 7283	68.09%	69.21%
Voting Classifier	0. 7437	71.34%	67.44%

- Discussion:
  - Interesting part here is that logistic regressor actually performs better than support vector machine!! Perhaps, this is the strength of logistic regressor, to predict binary output from abundance of independent classes.
  - Also, just as teacher mentioned during the lecture, "Three heads are better than one". The voting result of the three classifiers indeed outperforms each and every single individual classifier, as stated in our generated result.
- After data filtering:

	Average Validation Score	**Precision**	**Recall**
Logistic Regression	0. 7049	66.90%	61.99%
Support Vector Classifier	0. 7061	70.56%	54.23%
Multinomial Naïve Bayes	0.7141	65.30%	71.41%
Voting Classifier	0. 7282	68.92%	66.92%

- Discussion: Remove stop words or not?
  - Retain all information:
    - Our Deeds are the Reason of this #earthquake May ALLAH Forgive us all
  - Remove:
    - Deeds Reason #earthquake May ALLAH Forgive

  We calculate the average accuracy on four models for both cases. We were all astounded by the discovery. The average validation score after removing stop words were 1.72 % lower than retaining all the information, that we do absolutely nothing! From this result, we speculate that "noise" plays an important role in a model.

- Submission on Kaggle:
  - We get 80.845% accuracy, and rank 396/1275