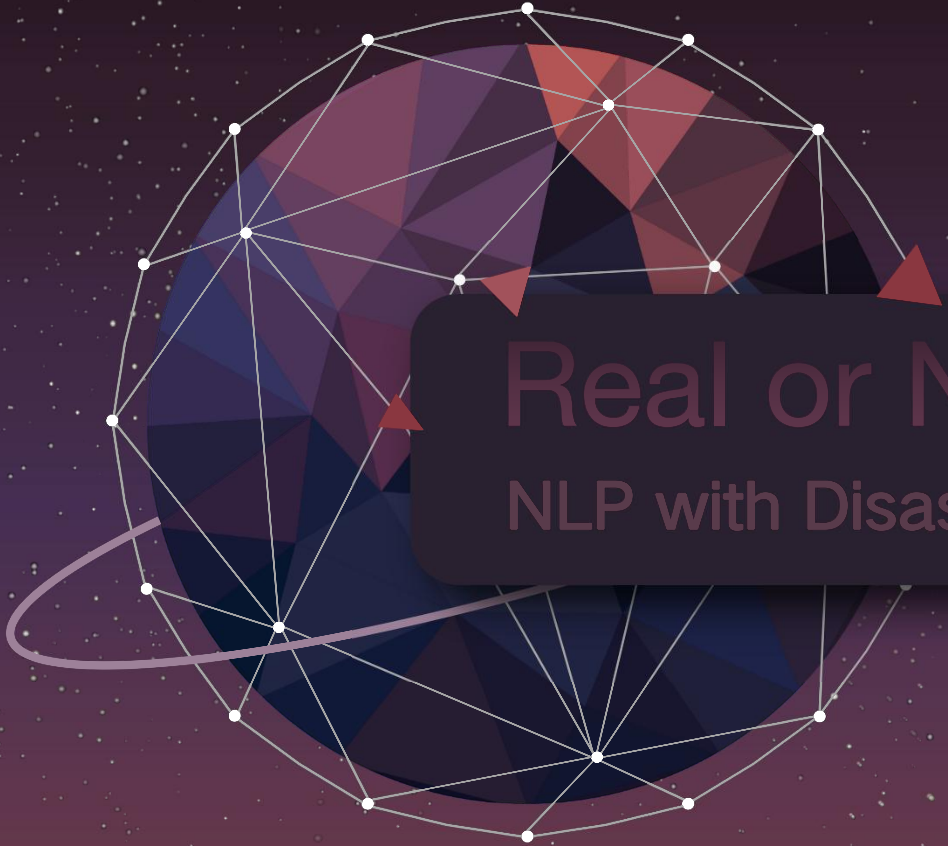


Real or Not?

NLP with Disaster Tweets

Team 8

Yan Sun

Chao Ma

Ruonan Ren

Overview

01

Use Cases

02

Goals

03

Methodology

04

TF-IDF

05

Classification Algorithms

06

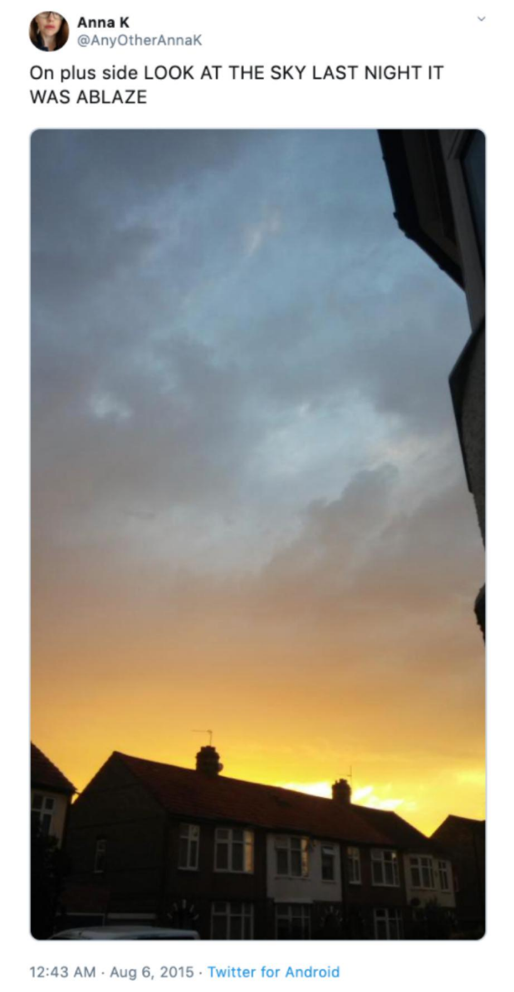
Project Results

07

Acceptance Criteria Review

01 Background

Twitter has become an important communication channel in times of emergency.
The ubiquitousness of smartphones enables people to announce an emergency they’re observing in real-time. Because of this, more agencies are interested in programatically monitoring Twitter (i.e. disaster relief organizations and news agencies).
But, it’s not always clear whether a person’s words are actually announcing a disaster.



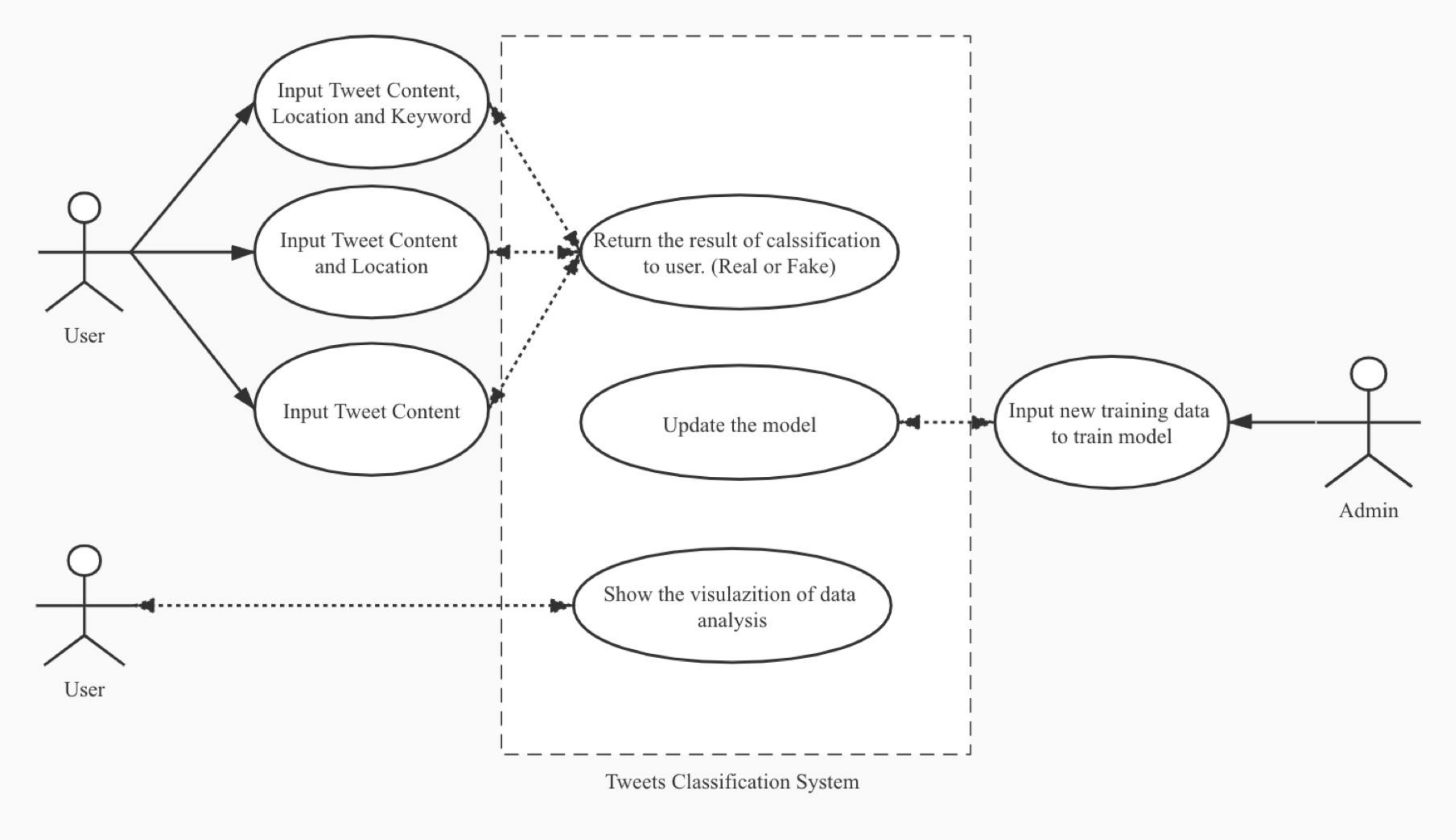
Anna K
@AnyOtherAnnaK

On plus side LOOK AT THE SKY LAST NIGHT IT WAS ABLAZE

12:43 AM · Aug 6, 2015 · Twitter for Android

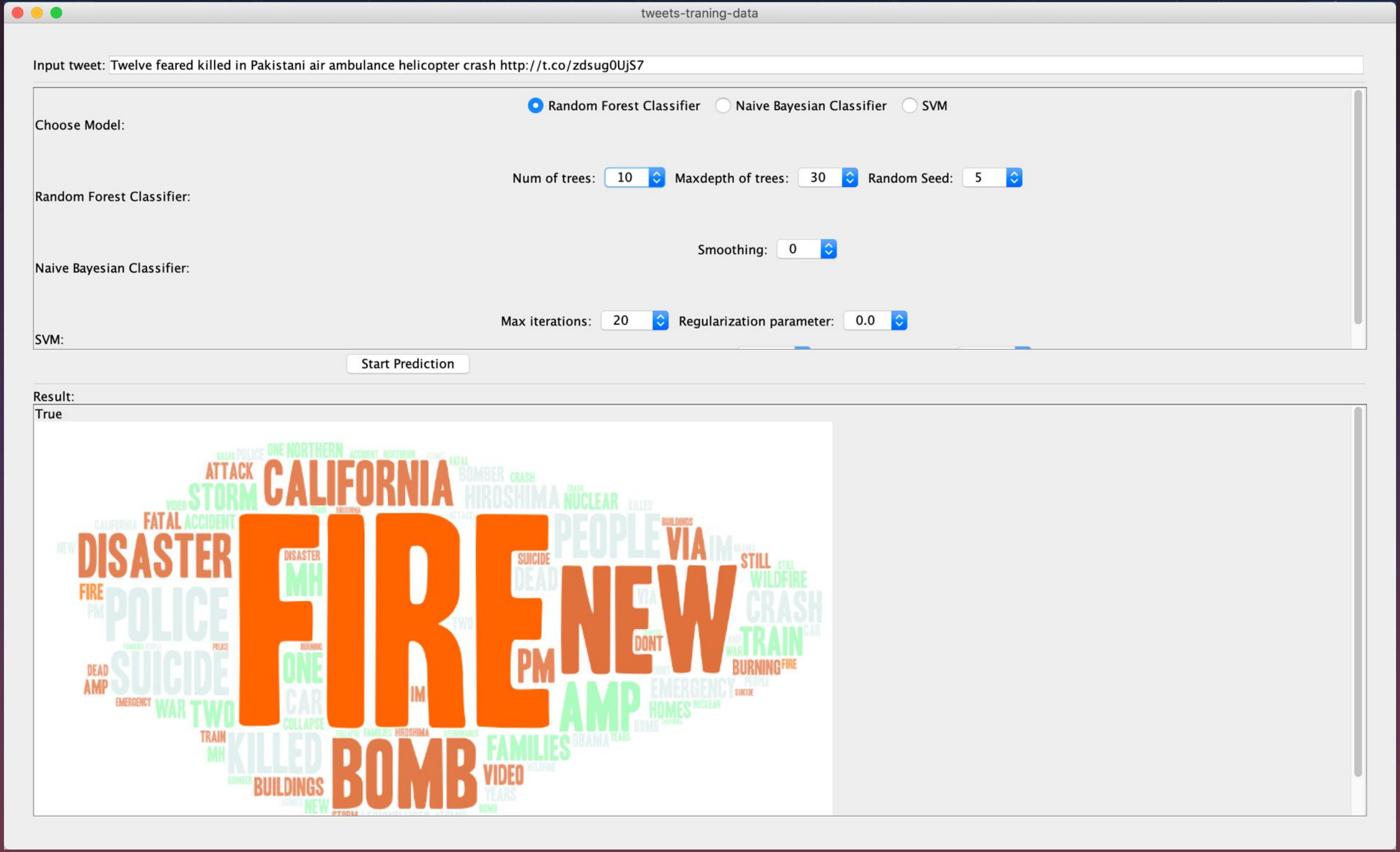
01

Uses Cases Review



01

UI Design



tweets-traning-data

Input tweet: Twelve feared killed in Pakistani air ambulance helicopter crash http://t.co/zdsug0UjS7

Choose Model: ● Random Forest Classifier ○ Naive Bayesian Classifier ○ SVM

Random Forest Classifier: Num of trees: 10 Maxdepth of trees: 30 Random Seed: 5

Naive Bayesian Classifier: Smoothing: 0

SVM: Max iterations: 20 Regularization parameter: 0.0

Result:

True

02

Goals

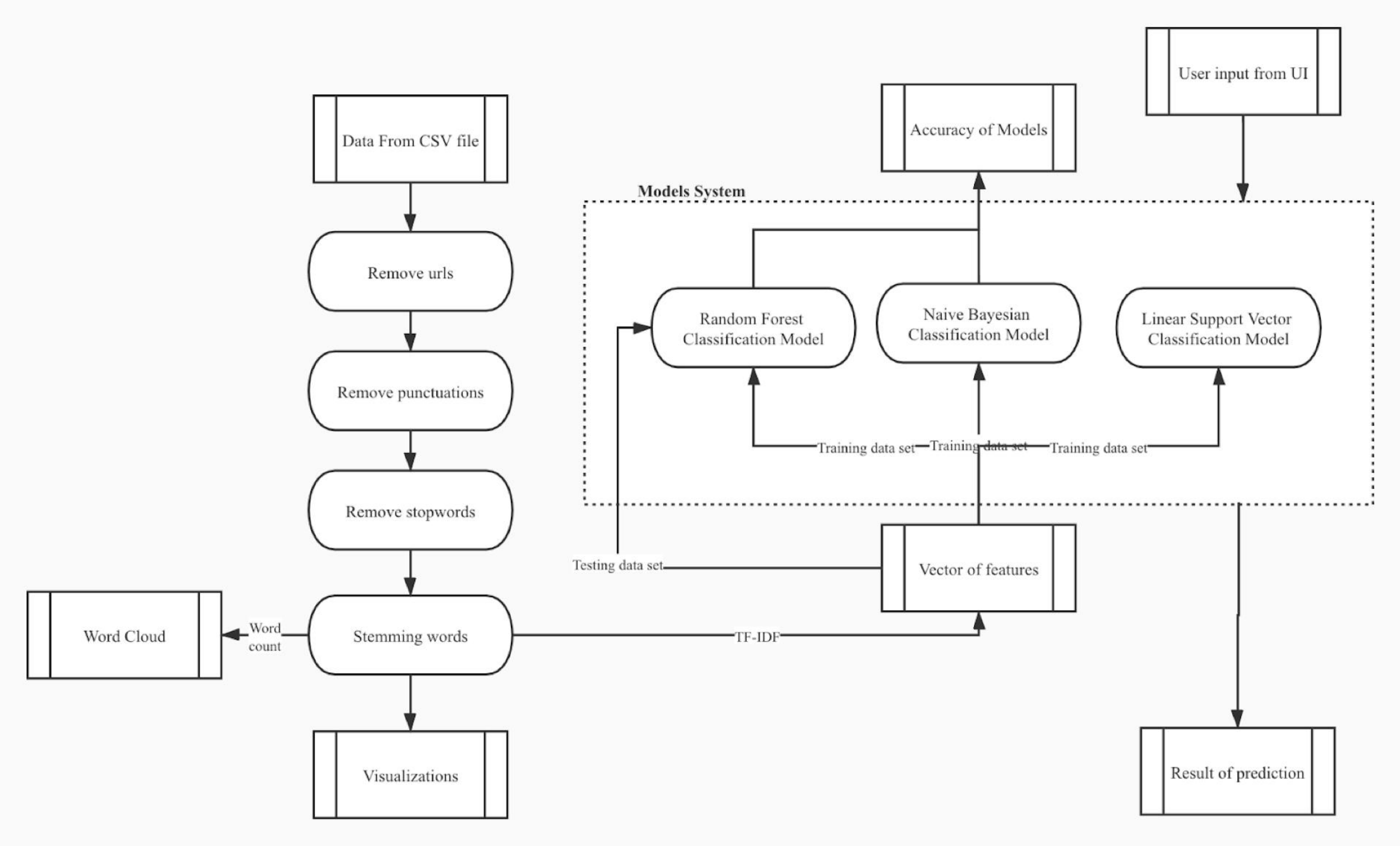
1 Create a reactive page to detect fake news on twitter.

2 Create a reactive page to analyze the characteristics of fake tweets.

3 Get a well trained model for fake tweets prediction.

03

Methodology

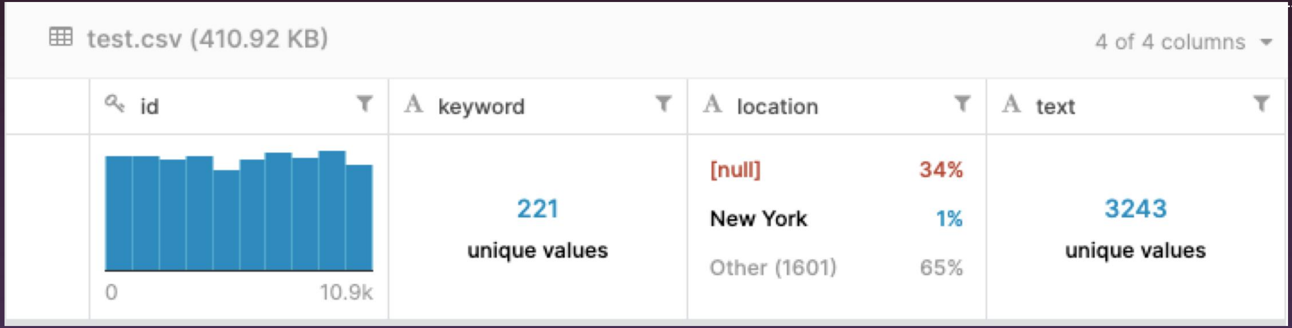


03

Methodology

- Used Zeppelin for running code line by line
- Extracted features of tweets (nature language) by TF-IDF
- Implemented 3 classification algorithms:
    - Random Forest Classifier
    - Naive Bayesian Classifier
    - Linear Support Vector Classifier
- Applied functions from Spark MLlib, Spark SQL, Spark RDD
- Designed and implemented UI by Scala Swing
- Visualized data by Vegas

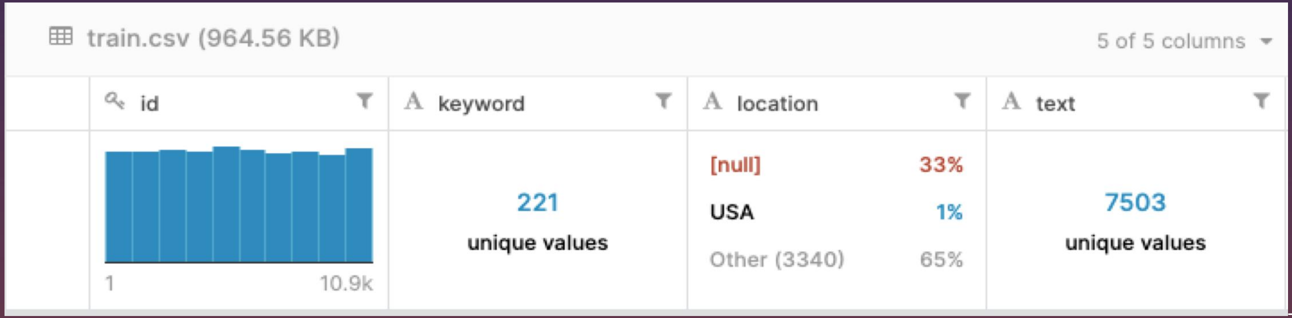
03 Data Sources



test.csv (410.92 KB)

4 of 4 columns

id	keyword	location	text
0 10.9k	221 unique values	[null] 34% New York 1% Other (1601) 65%	3243 unique values



train.csv (964.56 KB)

5 of 5 columns

id	keyword	location	text
1 10.9k	221 unique values	[null] 33% USA 1% Other (3340) 65%	7503 unique values

Data come form Kaggle competition

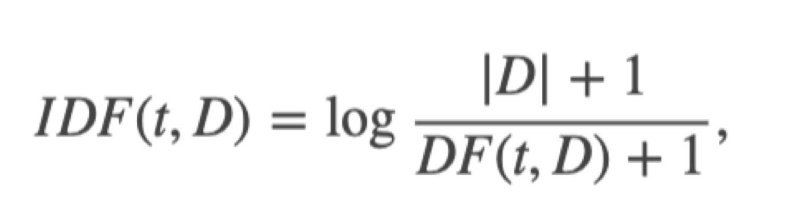
Data magnitude is more than 10,000 rows

04

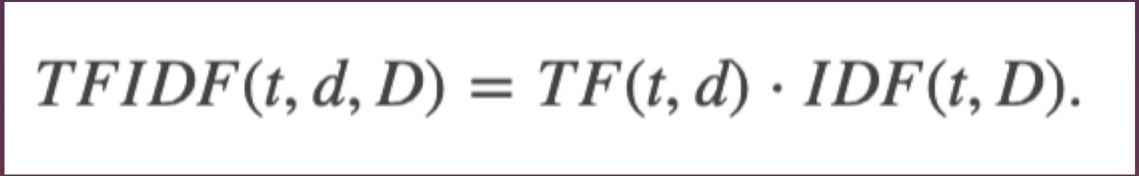
TF-IDF

Used to extract features and find keywords

- TF(t,d) is the number of times that term t appears in document d
- DF(t,D) is the number of documents that contains term t
- |D| is the total number of documents in the corpus



$$IDF(t, D) = \log \frac{|D| + 1}{DF(t, D) + 1},$$



$$TFIDF(t, d, D) = TF(t, d) \cdot IDF(t, D).$$

04

NLP

- Use Tokenizer to tokenize the tweets
- Use StopWordsRemover to remove the stop words

  Such as “ I, am, what, have, is, are….. ”
- Use HashingTF and IDF to vectorize the words
- Then we got featured data from nature language

05

Classification Algorithm

In order to classify the tweets according to if it is a real disaster tweets, we implemented three different classification algorithms:

- Random Forest Classifier
- Naive Bayesian Classifier
- Linear Support Vector Classifier

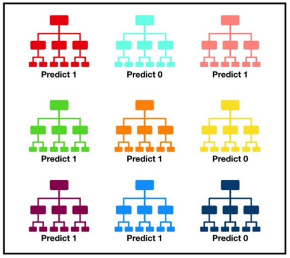
05

Random Forest Classifier

- A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.
- Given a training set X = x1, ..., xn with responses Y = y1, ..., yn, bagging repeatedly (B times) selects a random sample with replacement of the training set and fits trees to these samples
- For b = 1, ..., B:
  - Sample, with replacement, n training examples from X, Y; call these Xb, Yb.
  - Train a classification or regression tree fb on Xb, Yb.

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} f_b(x')$$

$$\sigma = \sqrt{\frac{\sum_{b=1}^{B}(f_b(x') - \hat{f})^2}{B-1}}.$$



05 Naive Bayesian Classifier

- Another assumption made here is that all the predictors have an equal effect on the outcome.
- X represents the features , y represents the label

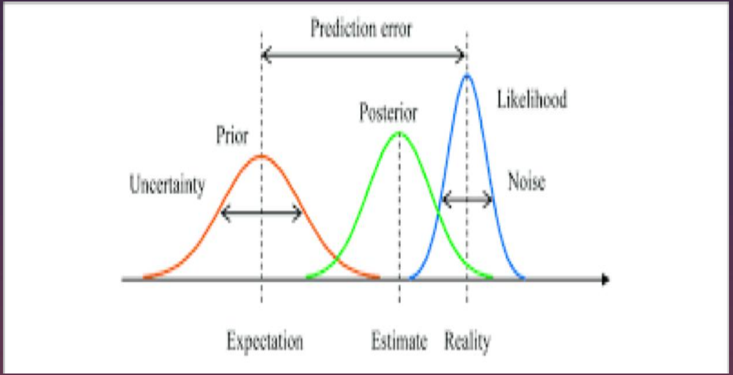
$$X = (x_1, x_2, x_3, \ldots.., x_n)$$

- According to Bayesian Theory

$$P(y|x_1, \ldots, x_n) = \frac{P(x_1|y)P(x_2|y)\ldots P(x_n|y)P(y)}{P(x_1)P(x_2)\ldots P(x_n)}$$

$$P(y|x_1, \ldots, x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y)$$

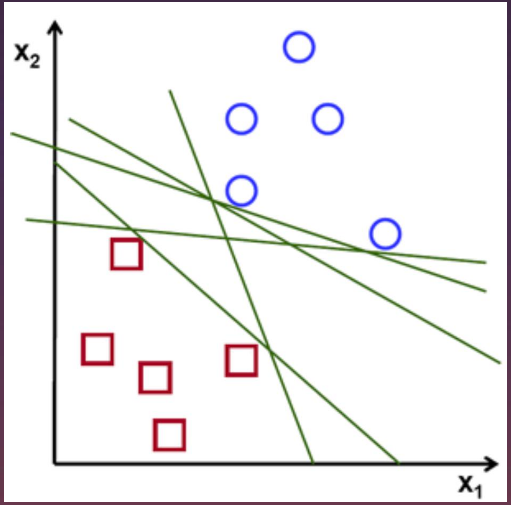
$$y = argmax_y P(y)\prod_{i=1}^{n} P(x_i|y)$$

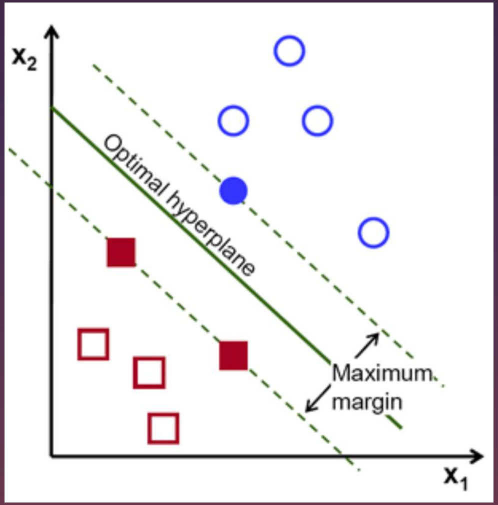


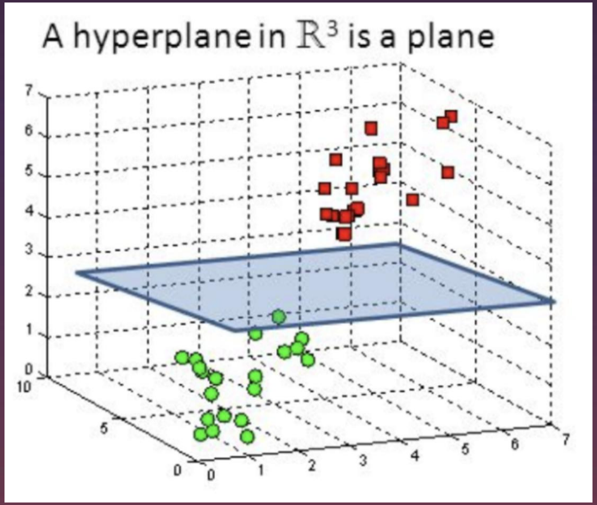
05

Linear Support Vector Classifier

- The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.







05 Linear Support Vector Classifier

- In the SVM algorithm, we are looking to maximize the margin between the data points and the hyperplane. The loss function that helps maximize the margin is hinge loss.

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases}$$

$$min_w \lambda \parallel w \parallel^2 + \sum_{i=1}^{n} (1 - y_i \langle x_i, w \rangle)_+$$

- Take partial derivatives with respect to the weights to find the gradients
- Update gradient (no misclassification)
- Update gradient (with misclassification)

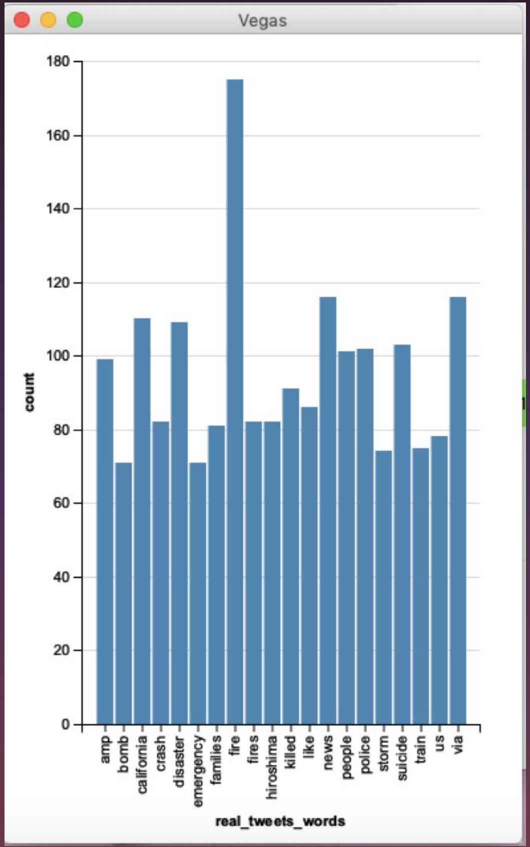
$$\frac{\delta}{\delta w_k} \big(1 - y_i \langle x_i, w \rangle\big)_+ = \begin{cases} 0, & \text{if } y_i \langle x_i, w \rangle \geq 1 \\ -y_i x_{ik}, & \text{else} \end{cases}$$

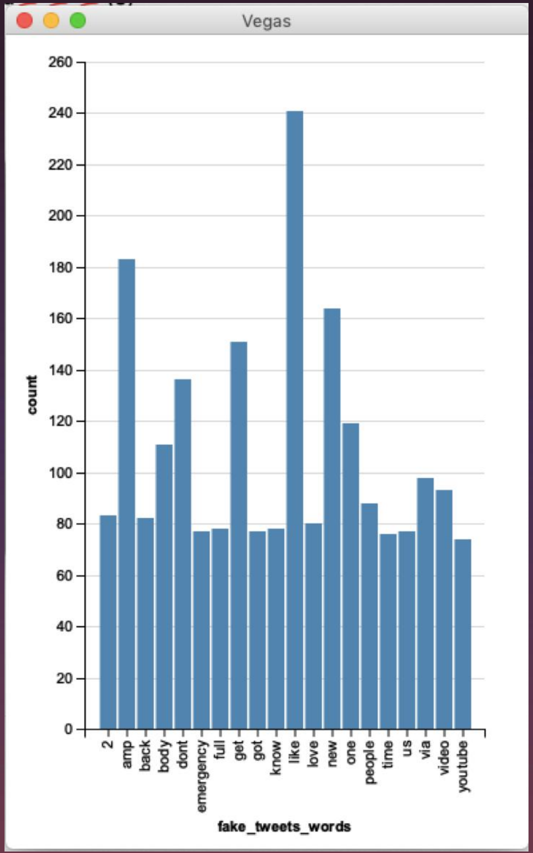
$$w = w - \alpha \cdot (2\lambda w)$$

$$w = w + \alpha \cdot (y_i \cdot x_i - 2\lambda w)$$

06

Visualization of Keyword Extraction



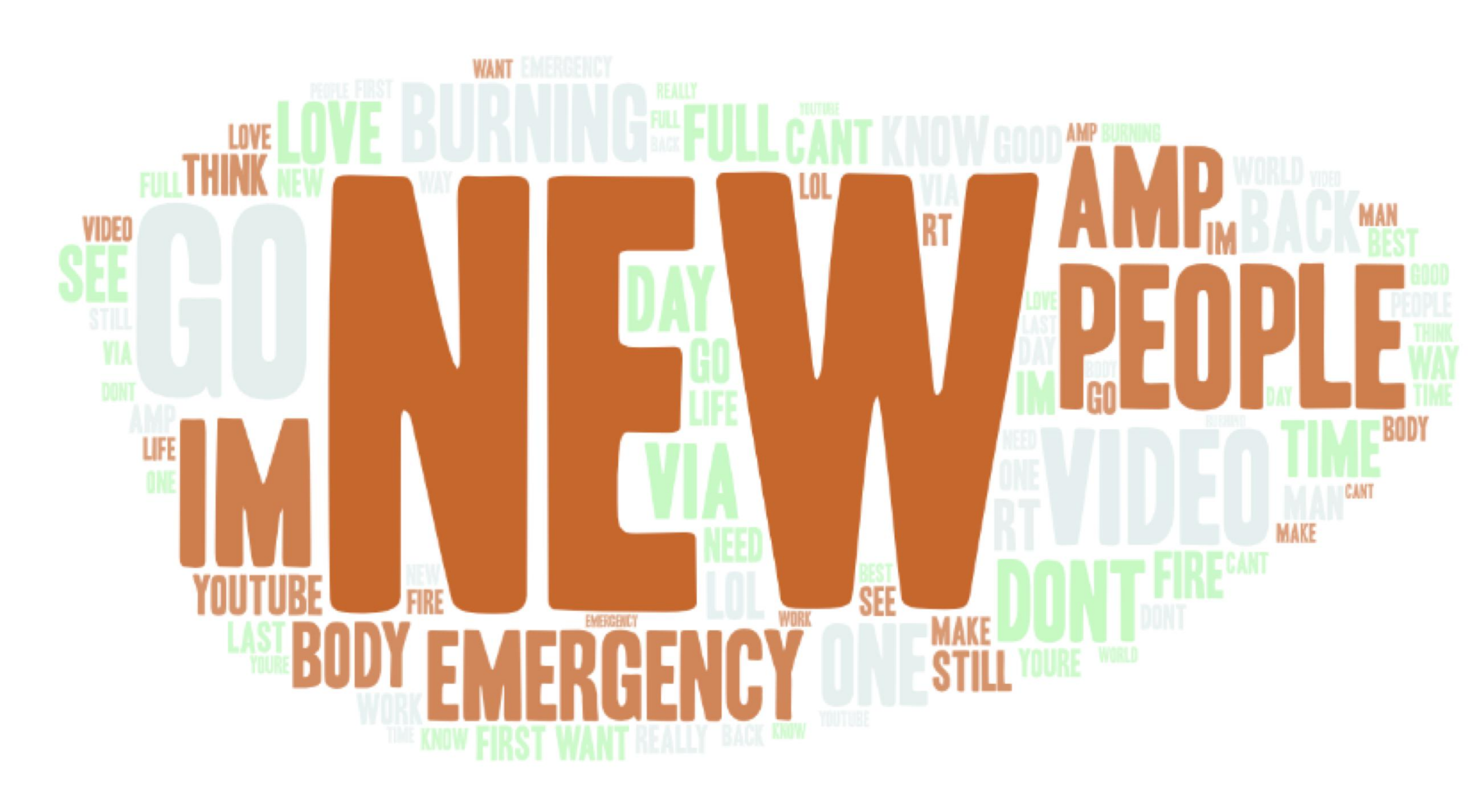


06 Real Disaster Tweets

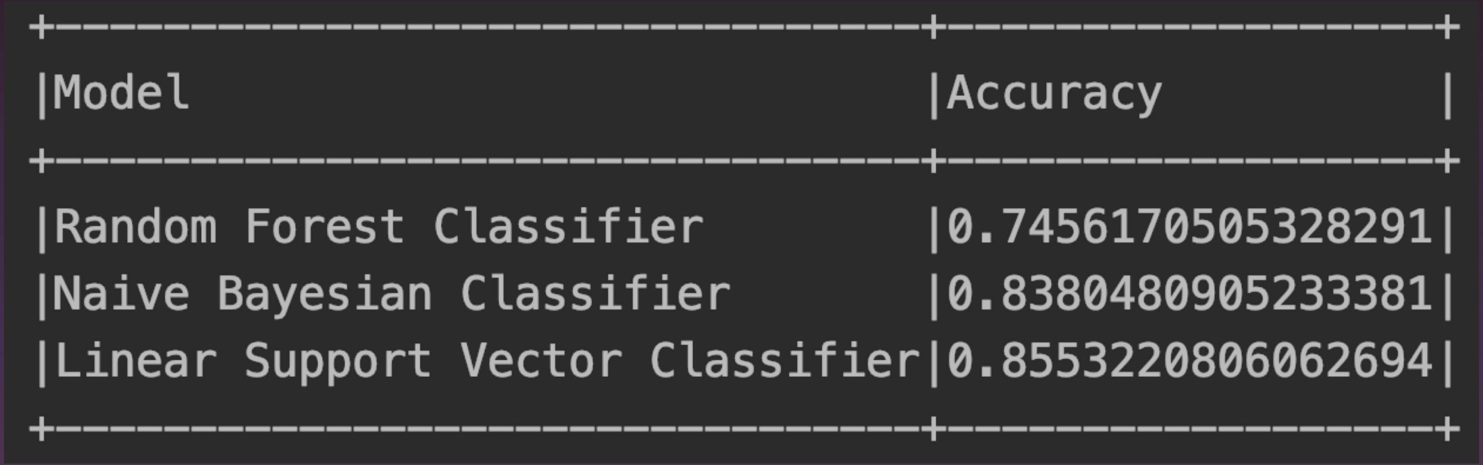
KILLED
POLICE
ONE
NORTHERN
ACCIDENT
NORTHERN
ATOMIC
FATAL
ATTACK
CALIFORNIA
BOMBER
CRASH
STORM
HIROSHIMA
CRASH
NUCLEAR
KILLED
VIDEO
TRAIN
HIROSHIMA
CALIFORNIA
FATAL
ACCIDENT
ATTACK
BUILDINGS
FIRE
PEOPLE
VIA
IM
OBAMA
NEW
DISASTER
DISASTER
SUICIDE
STILL
STILL
MH
DEAD
WILDFIRE
NEW
FIRE
POLICE
VIA
CRASH
PM
TWO
FAMILIES
PEOPLE
POLICE
BURNING
DONT
TRAIN
CAR
SUICIDE
ONE
PM
AMP
WAR
DEAD
BURNING
FIRE
AMP
IM
DONT
EMERGENCY
PEOPLE
SUICIDE
EMERGENCY
CAR
WAR
TWO
AMP
HOMES
NUCLEAR
COLLAPSE
BOMB
CALIFORNIA
TRAIN
COLLAPSE
FAMILIES
HIROSHIMA
LEGIONNAIRES
OBAMA
YEARS
KILLED
FAMILIES
MH
BOMB
VIDEO
WILDFIRE
BOMBER
BUILDINGS
YEARS
HOMES
NEW
STORM
LEGIONNAIRES
ATOMIC
BOMB

06

Fake Disaster Tweets



06 Evaluation of models



```
+--------------------------------+------------------+
|Model                           |Accuracy          |
+--------------------------------+------------------+
|Random Forest Classifier        |0.7456170505328291|
|Naive Bayesian Classifier       |0.8380480905233381|
|Linear Support Vector Classifier|0.8553220806062694|
+--------------------------------+------------------+
```

07

Acceptance Criteria Review

As a user, I am able to input Disaster Tweet content, location and keyword to get the prediction if the tweet is fake:

- The prediction accuracy for complete input data should be over 70%
- The time to respond should be under 5 seconds

As a user, I am able to show the visualization of data analysis:

- The time to respond should be under 5 seconds

Thank You