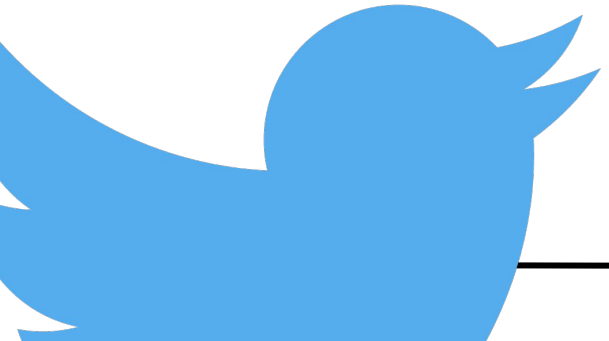
Mapping Disasters using Twitter



Presented by:

Dae Han
Sophia Alice
Sonam Thakkar

Problem statement

We used **tweets** from Twitter
to create **a classification model**
that filters out tweets that are reporting about a fire
to discover **new ways to help respond to disasters.**

These tweets were displayed on a map.

Beyond aerial, satellite imaging

Beyond aerial, satellite imaging

Each tweet scapers have its ups and downs

Tweepy	GetOldTweets3	Python-Twitter
Strict scrape rate limit	No scrape rate limit	Strict scrape rate limit
Geo. coordinate ‘search’ not available	Geo. coordinate not available	Geo. coordinate ‘search’ available

Each tweet scapers have its ups and downs

Tweepy

Strict scrape rate limit

Geo. coordinate ‘search’ not available

GetOldTweets3

No scrape rate limit

Geo. coordinate not available

Python-Twitter

Strict scrape rate limit

Geo. coordinate ‘search’ available

Each tweet scapers have its ups and downs

Tweepy

Strict scrape rate limit

Geo. coordinate ‘search’ not available

GetOldTweets3

No scrape rate limit

Geo. coordinate not available

Model Train Set

Python-Twitter

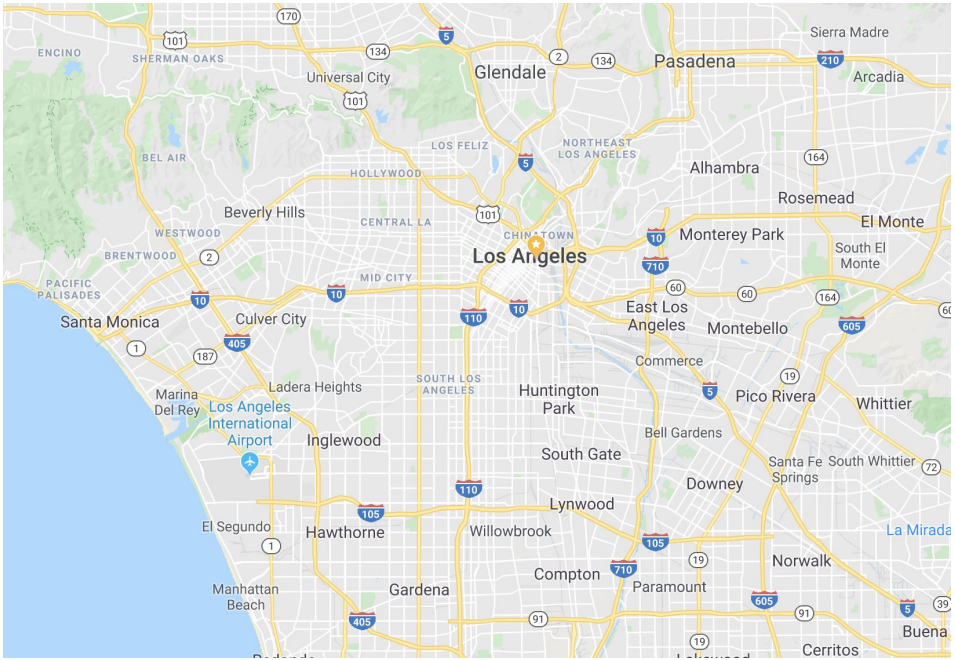
Strict scrape rate limit

Geo. coordinate ‘search’ available

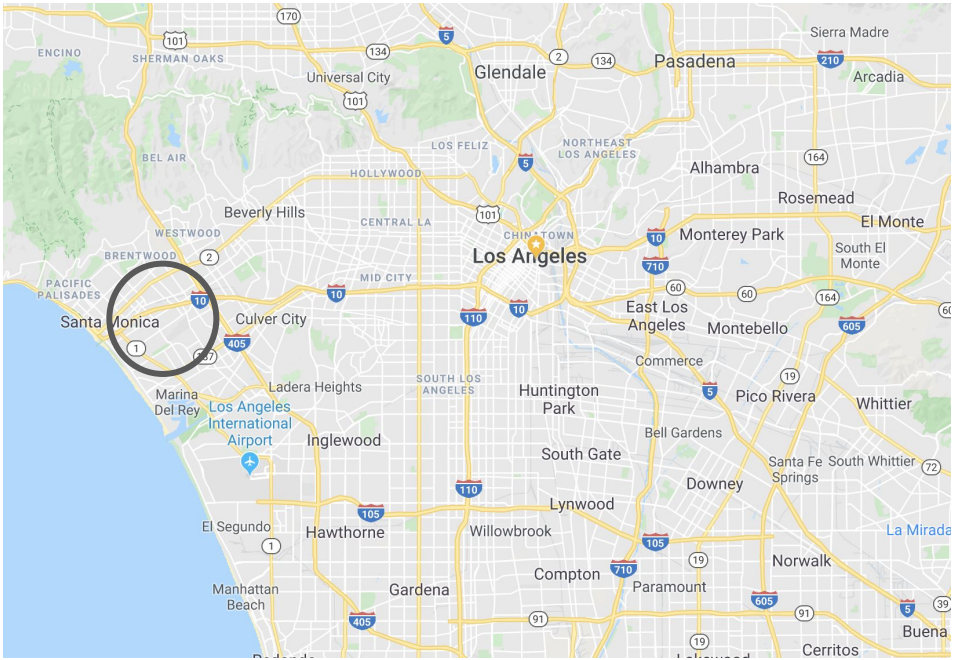
Each tweet scapers have its ups and downs

Tweepy	GetOldTweets3	Python-Twitter
Strict scrape rate limit	No scrape rate limit	Strict scrape rate limit
Geo. coordinate ‘search’ not available	Geo. coordinate not available	Geo. coordinate ‘search’ available
	**Model Train Set**	**Test Set / Map**

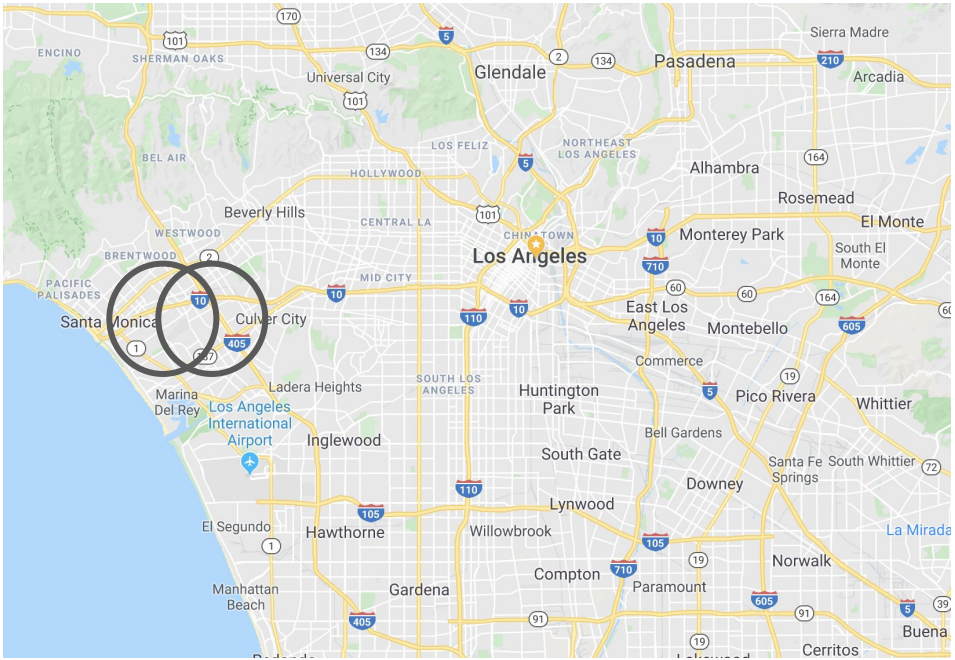
“Only 1~2 % of tweets are geotagged”
- Twitter



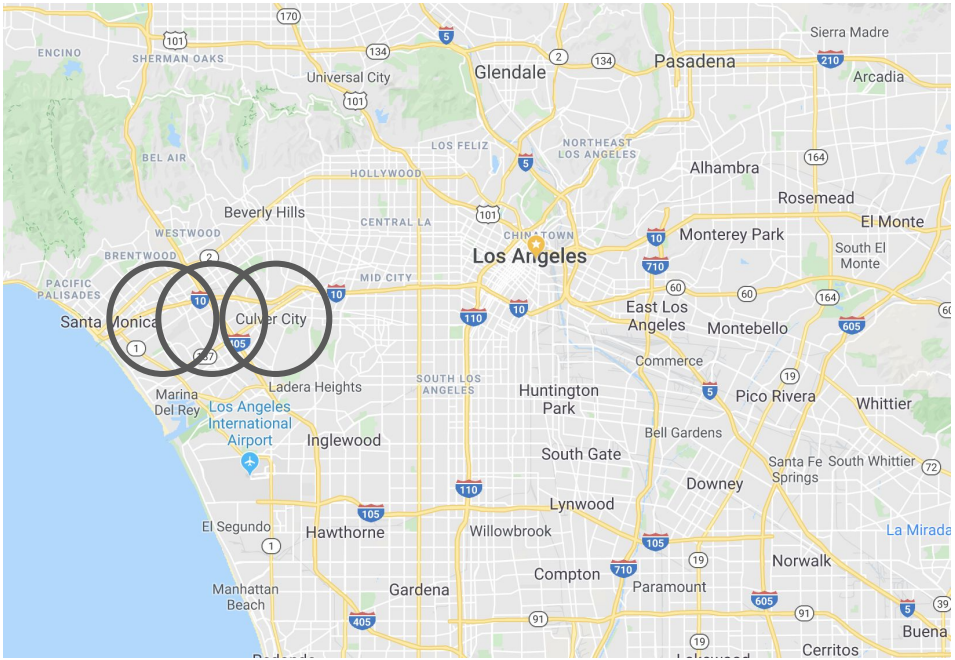
“Only 1~2 % of tweets are geotagged”
- Twitter



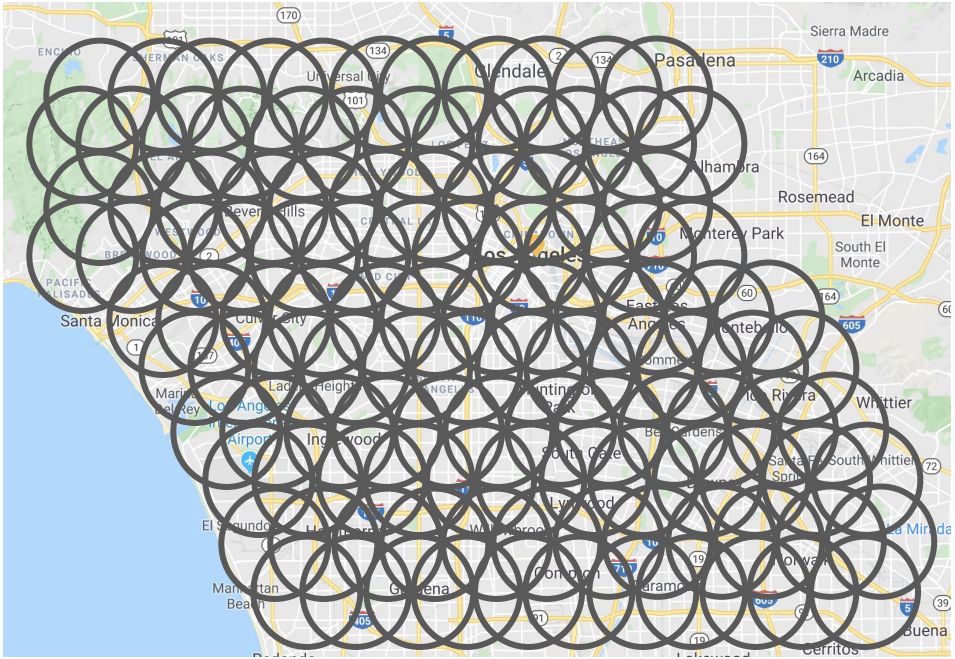
“Only 1~2 % of tweets are geotagged”
- Twitter



“Only 1~2 % of tweets are geotagged”
- Twitter



“Only 1~2 % of tweets are geotagged”
- Twitter

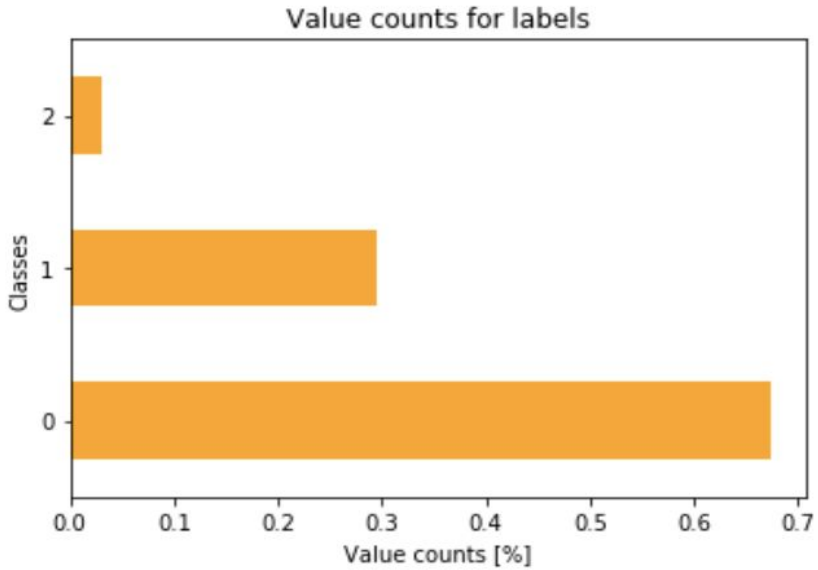


Few tweets are discussing a disaster.

0: Tweets not reporting about a fire

1: Tweets reporting about a fire

2: Ambiguous tweets —> Dropped



3:7 ratio of Class 0 and 1 to reflect the real life tweets in a train set .

Modeling

Minimize **false negatives**

=

Optimize **sensitivity**

Logistic Regression	Random Forest	Multinomial Naive Bayes
TF-IDF Vectorizer	TF-IDF Vectorizer	Count Vectorizer

Modeling

Minimize **false negatives**

=

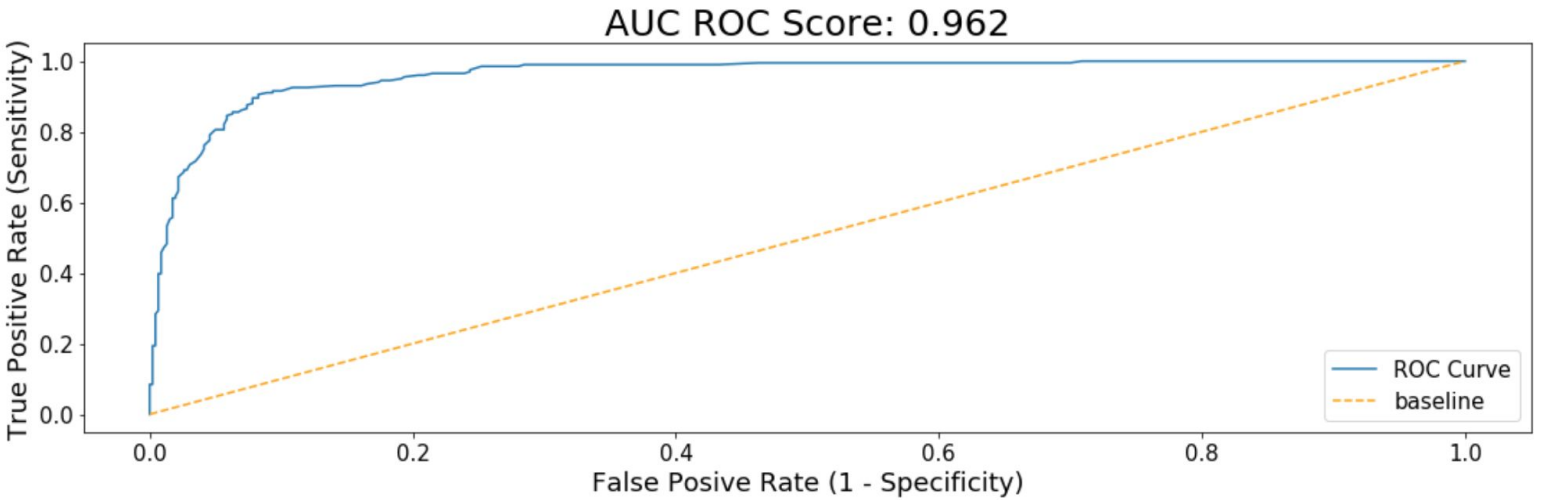
Optimize **sensitivity**

Logistic Regression	Random Forest	Multinomial Naive Bayes
TF-IDF Vectorizer	TF-IDF Vectorizer	Count Vectorizer

Logistic Regression (TF-IDF Vectorizer)

Accuracy (test set): 86 %

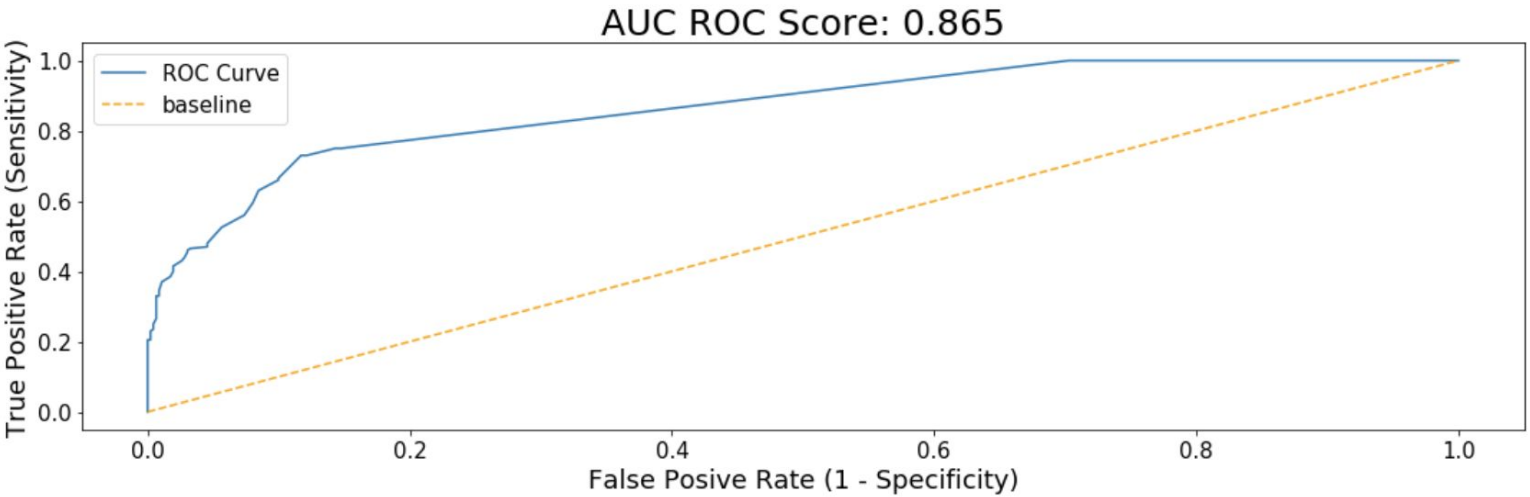
Sensitivity: 62%



Random Forest (TF-IDF Vectorizer)

Accuracy (test set): 76 %

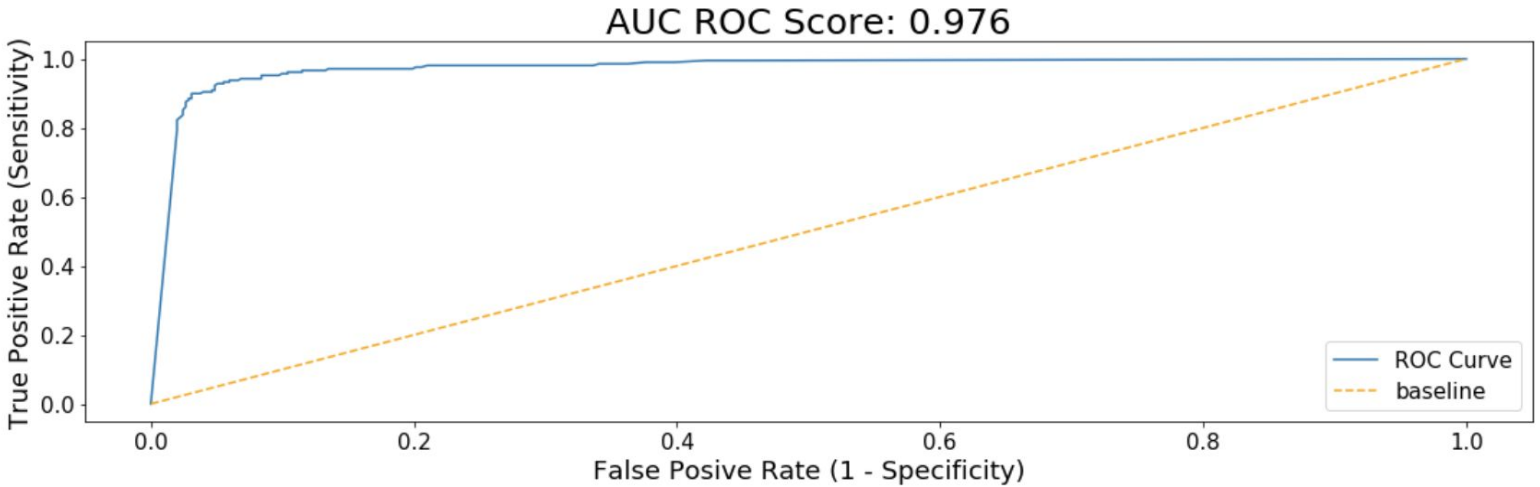
Sensitivity: 21%



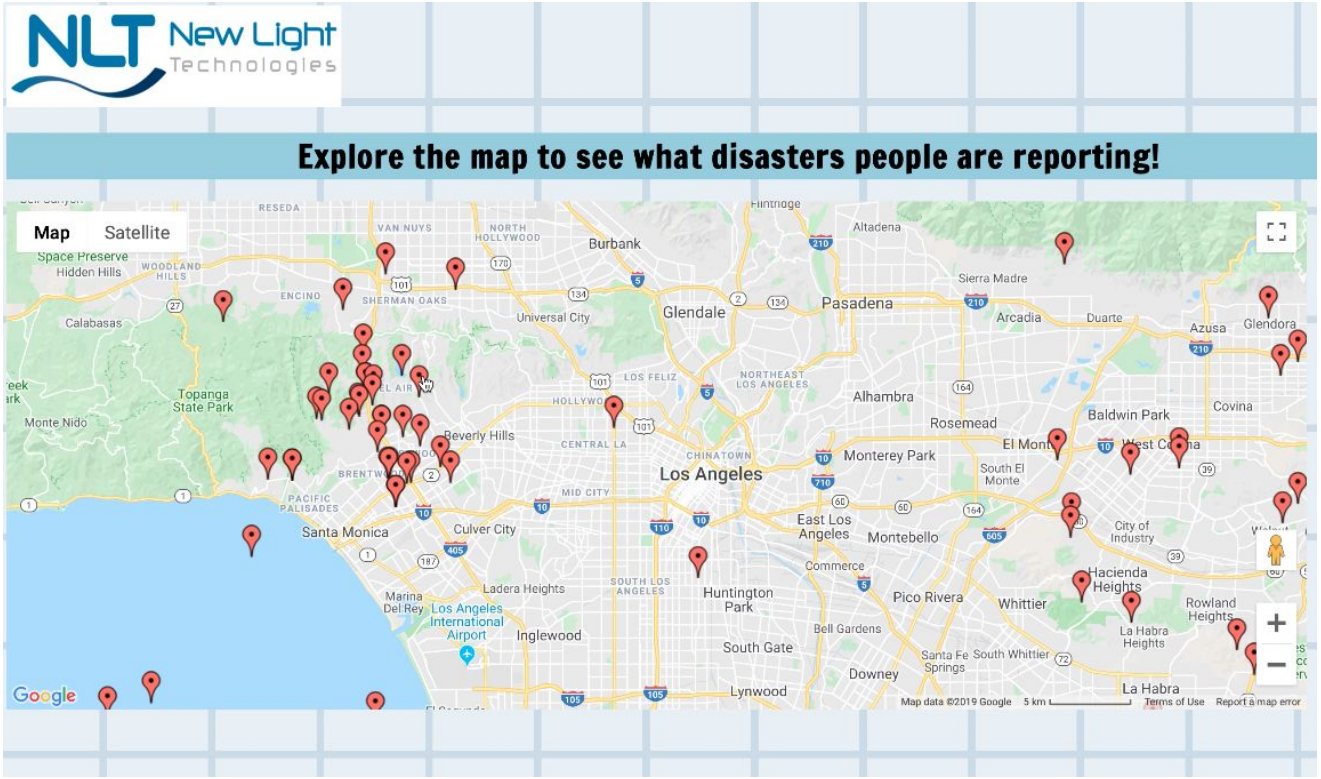
Multinomial Naive Bayes (Count Vectorizer)

Accuracy (test set): 84 %

Sensitivity: 90%



Flask app / demo



Conclusion

Obstacles

- Scraping takes a long time due to rate limit
- Geotags are scarce.

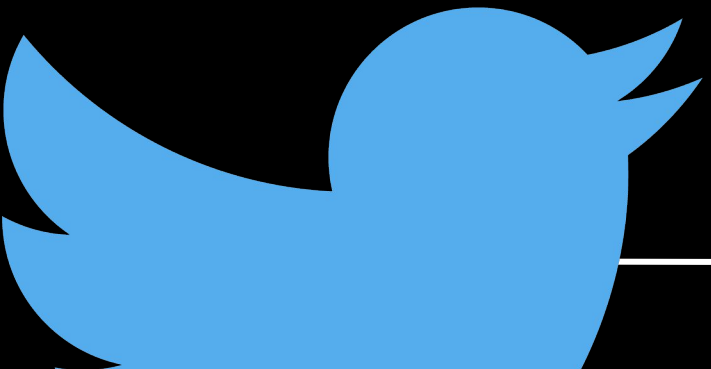
- Inaccurate geo-coordinates (Aka. Ocean Fire)

Next steps

- Run scraper daily, ahead of the time
- Enterprise API access for real-time track

- Create a boundary line and filter out coordinates outside the boundary.

- Make rating available for users to learn which tweets were useful
  —> feed these back to the train set



Source

1. http://www.newlighttechnologies.com
2. https://www.twitter.com
3. https://www.eos.org