Harnessing Machine Learning for Natural Disaster Alert Systems

Edith Iyer-Hernandez, Tunde Duro-ladipo, & Kevin Aguiar

Agenda

1. Problem Statement
2. Data Gathering
3. Data Pre-processing
4. Model Building
5. Application Workflow
6. Shortcomings
7. Recommendations
8. Q&A

Problem Statement

Build a **Tsunami Alert System** that uses  's **#tsunami** livestream/API.

Designed to be **scalable**.

Data Gathering

Twitter API

Livestream of tweets containing **#tsunami**

Kaggle Dataset

50 Labeled **Tsunami** Tweets

Scraped Tweets

50 ‘alert-worthy’

≈1500 + 250

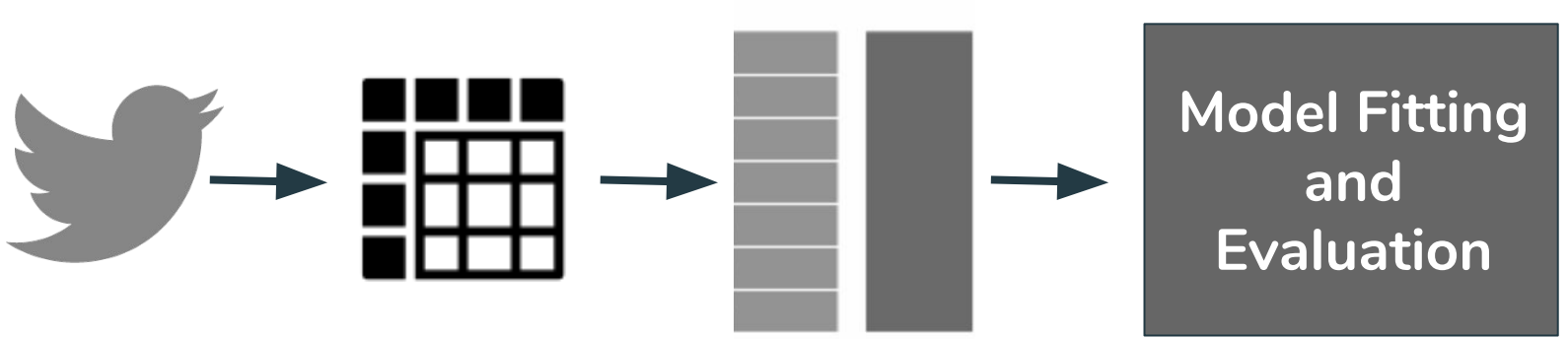
Non-Labeled ‘non-alert-worthy’ Tweets

300 Labeled Tweets

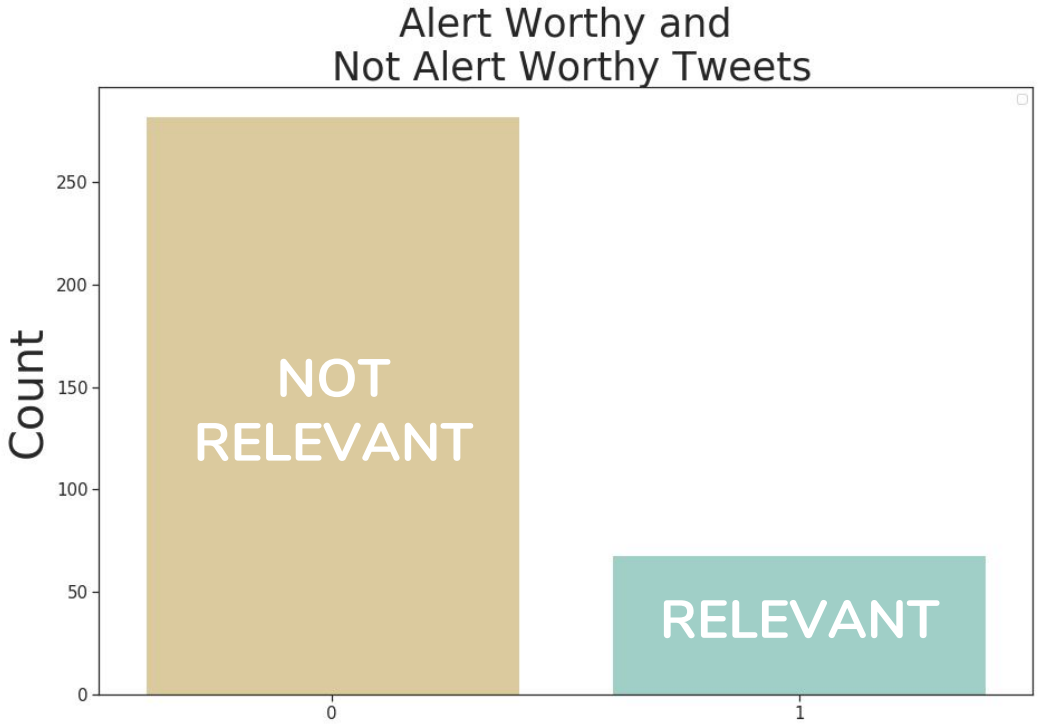
Natural Language Processing

Build a method for translating incoming tweets into vectorized features so that we can build a machine learning model.

Natural Language Processing



IMBALANCED CLASSES



IMBALANCED CLASSES

Solution: **SMOTE**

Synthetic
Minority
Oversampling
Technique

IMBALANCED CLASSES

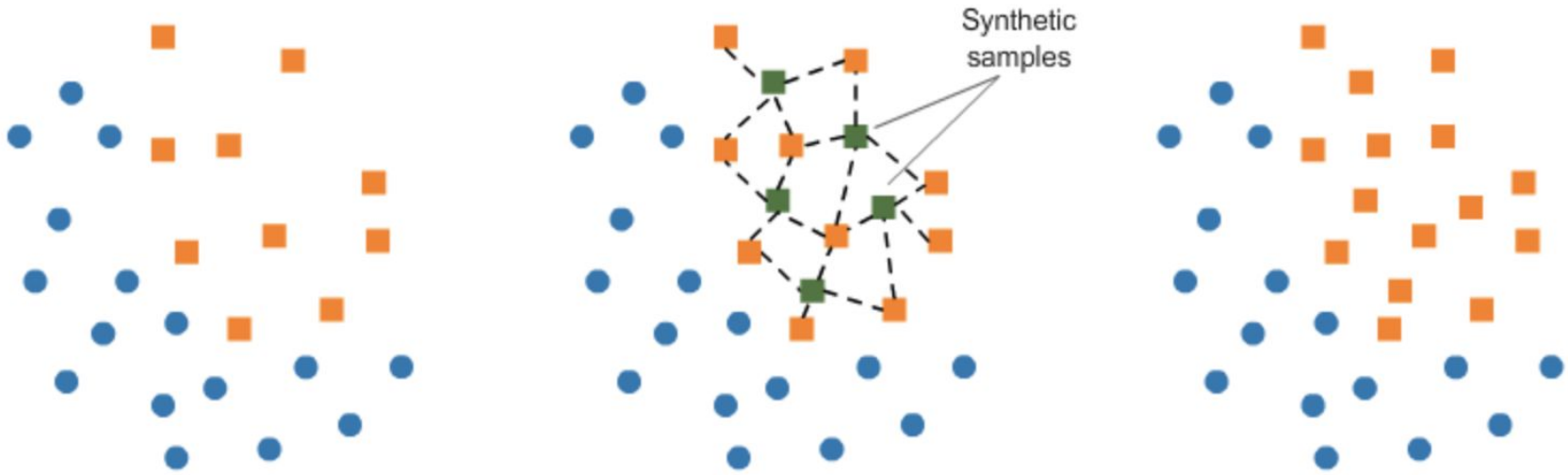
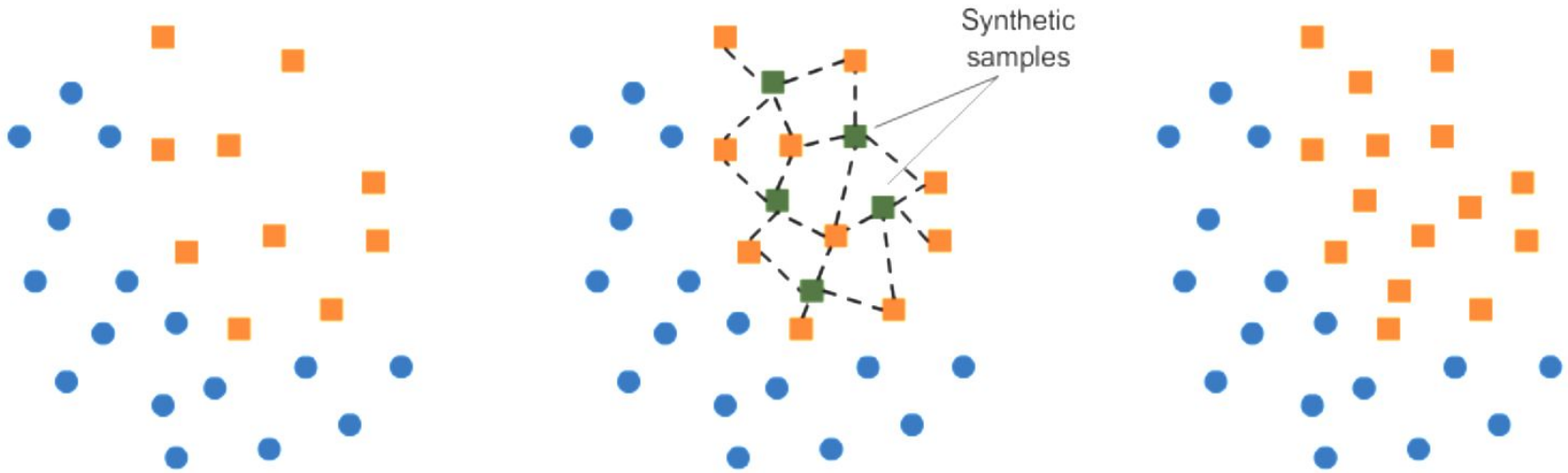


Image from: “Exploring Methods of Handling Imbalanced Data”
Author: Ashley Mighty | DSI-NYC
Edited: Tucker Allen | DSI-NYC

IMBALANCED CLASSES

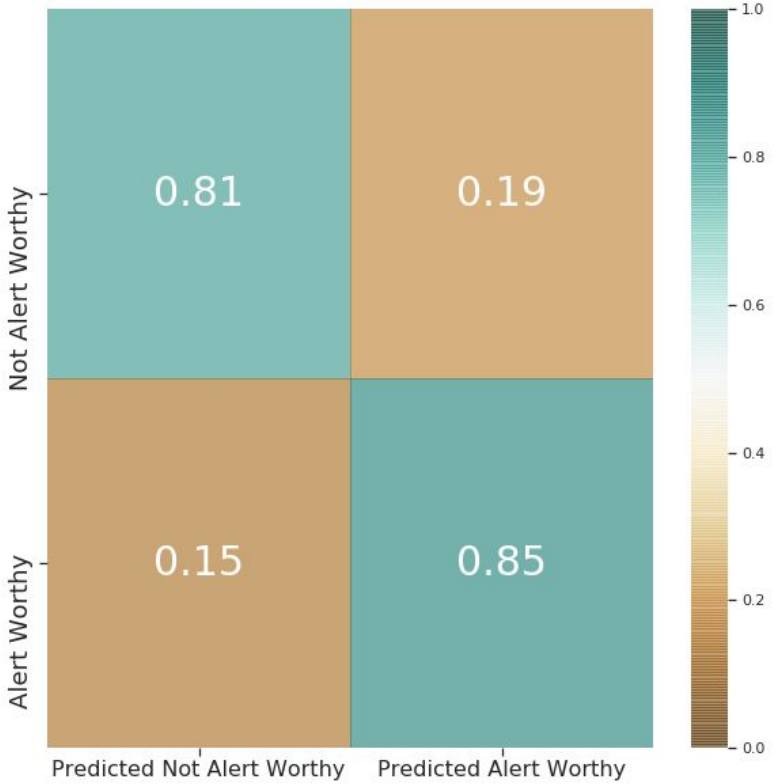


SMOTE: Less susceptible to overfitting

MODEL EVALUATION

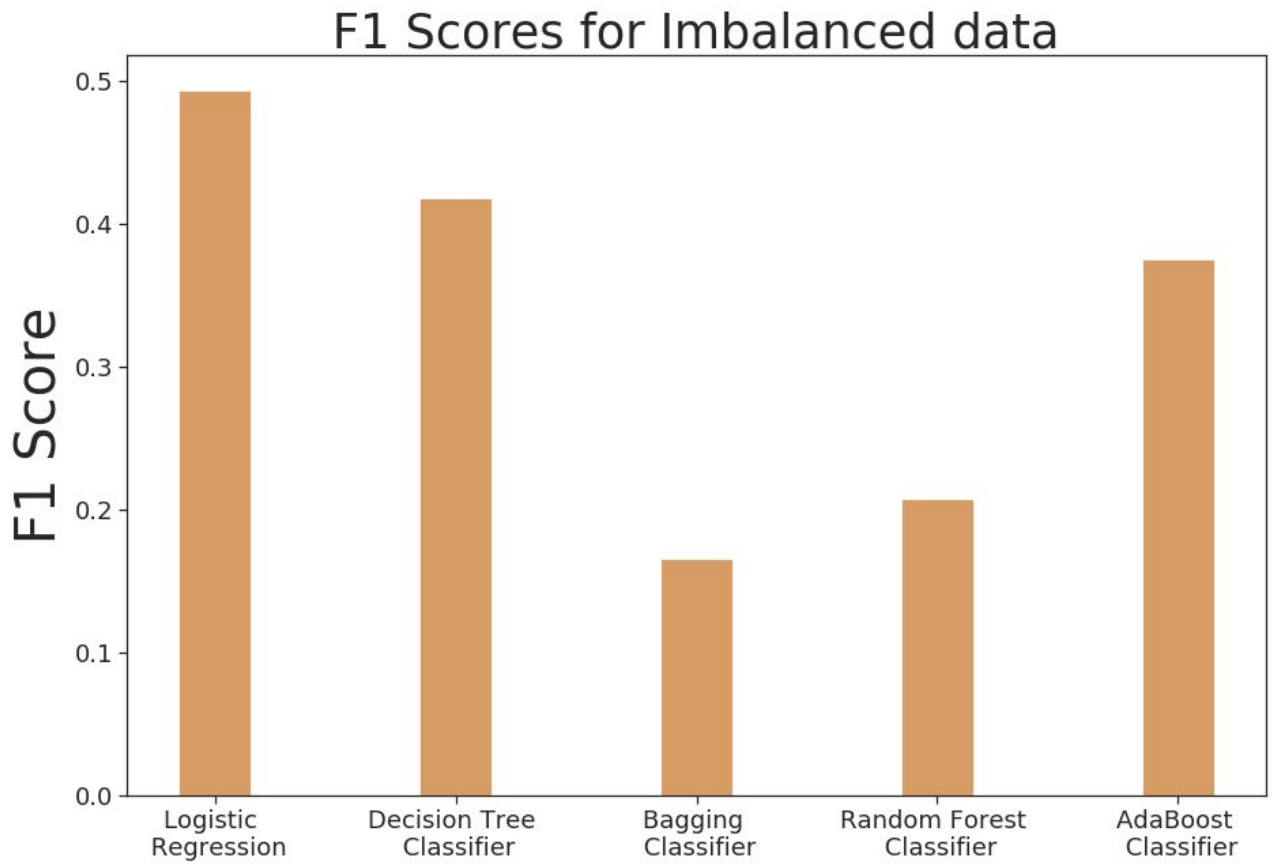
Metrics:

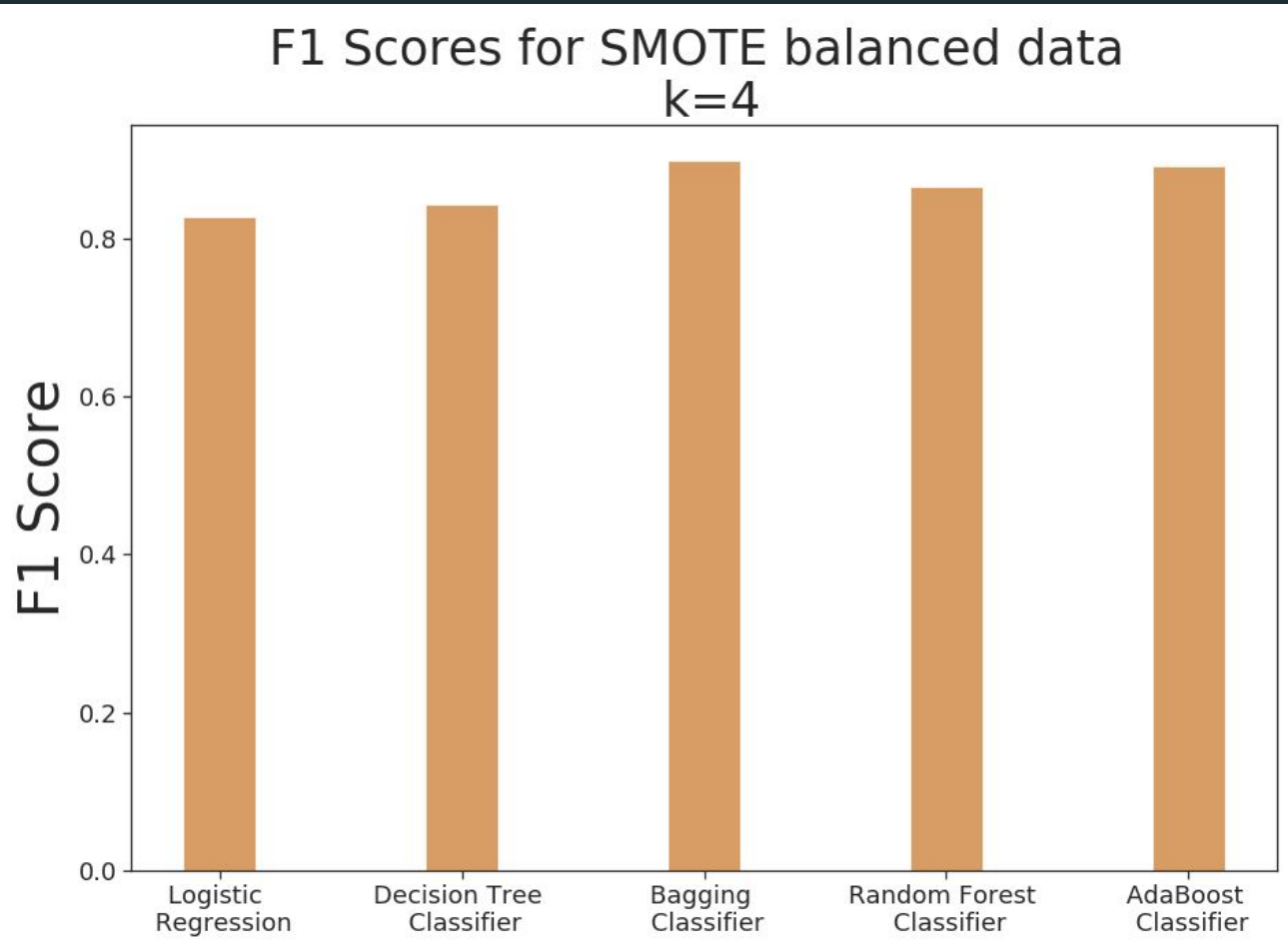
- F1 Score
  - Precision
  - Recall

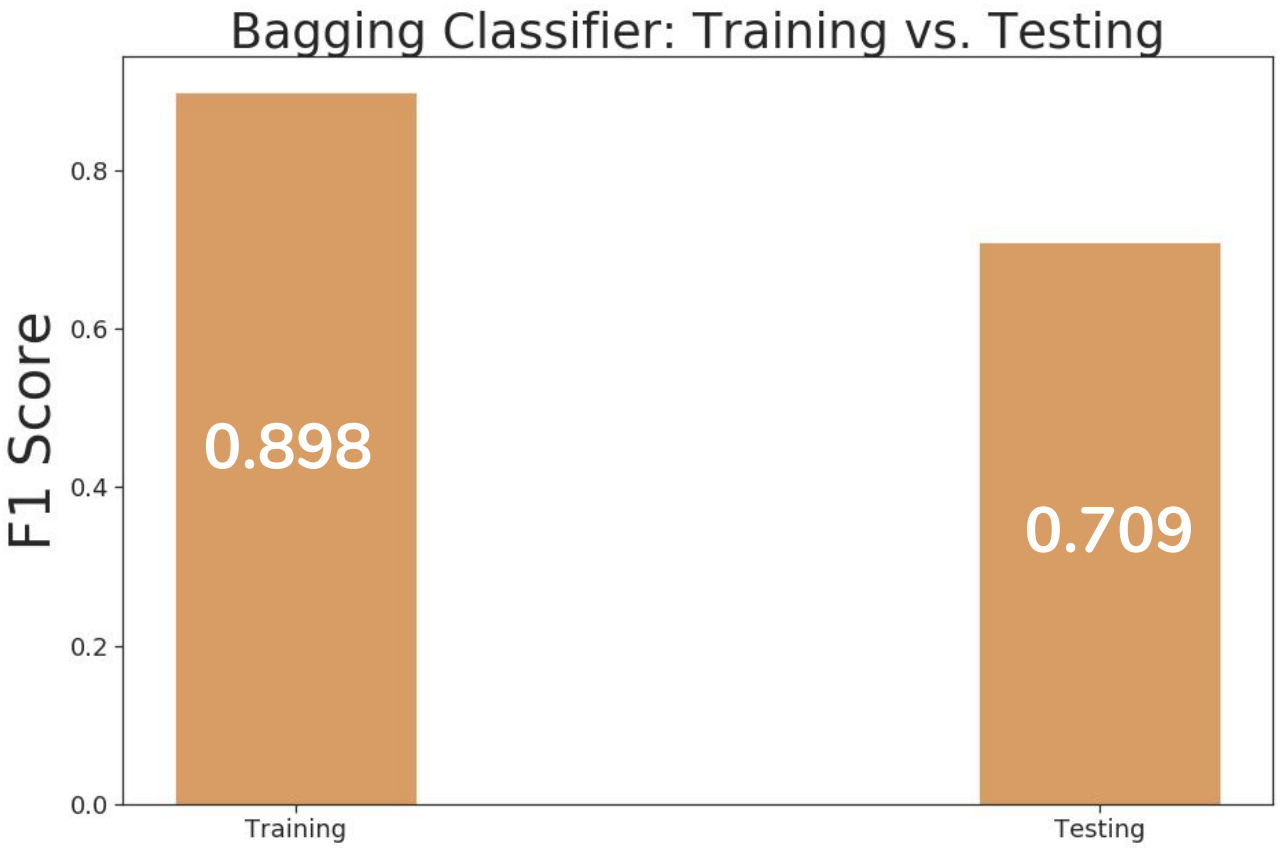


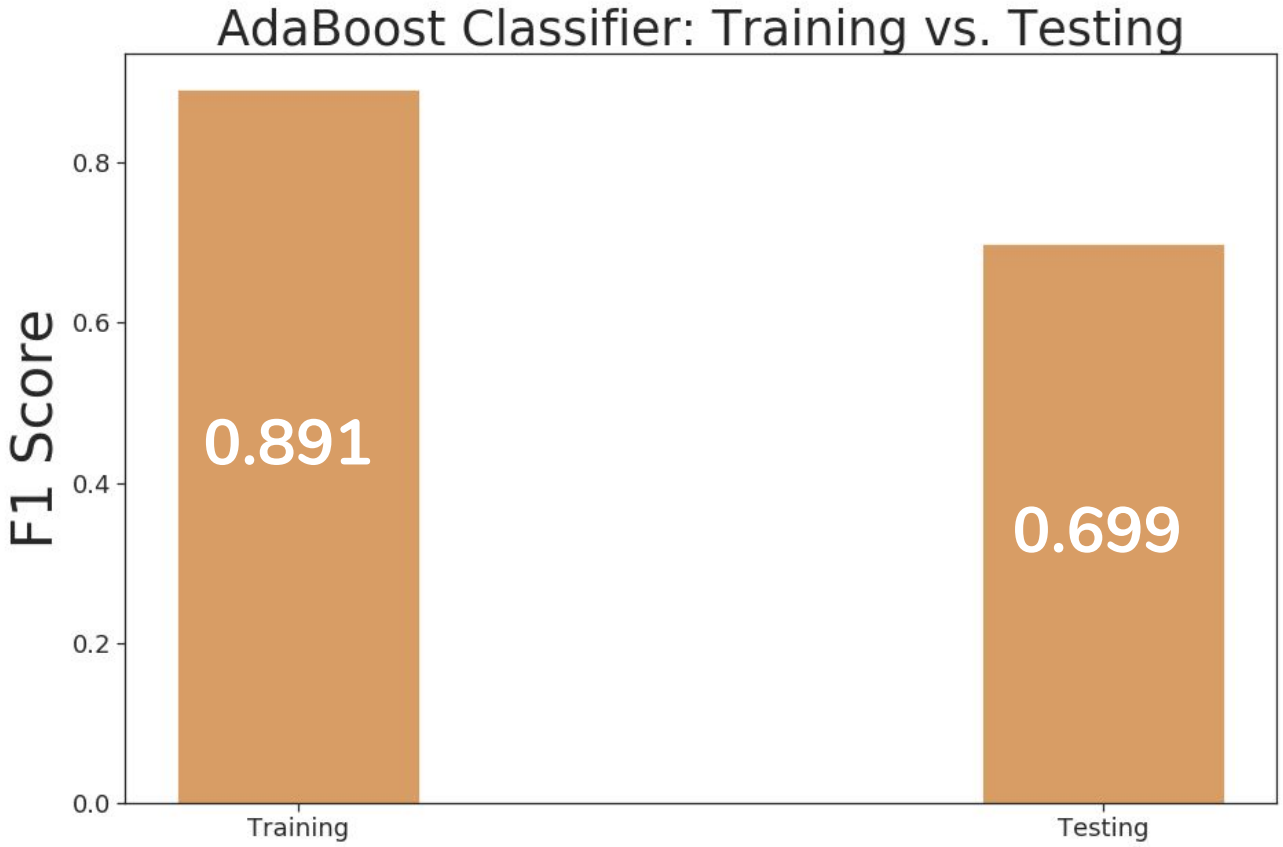
MODEL EXPLORATION

- Logistic Regression
- Decision Tree Classifier
- Bagging Classifier
- Random Forest Classifier
- AdaBoost Classifier


F1 Scores for Imbalanced data
F1 Score
0.5
0.4
0.3
0.2
0.1
0.0
Logistic Regression
Decision Tree Classifier
Bagging Classifier
Random Forest Classifier
AdaBoost Classifier


F1 Scores for SMOTE balanced data
k=4
F1 Score
0.0
0.2
0.4
0.6
0.8
Logistic Regression
Decision Tree Classifier
Bagging Classifier
Random Forest Classifier
AdaBoost Classifier


Bagging Classifier: Training vs. Testing
F1 Score
0.0
0.2
0.4
0.6
0.8
0.898
0.709
Training
Testing


AdaBoost Classifier: Training vs. Testing
F1 Score
0.0
0.2
0.4
0.6
0.8
0.891
0.699
Training
Testing

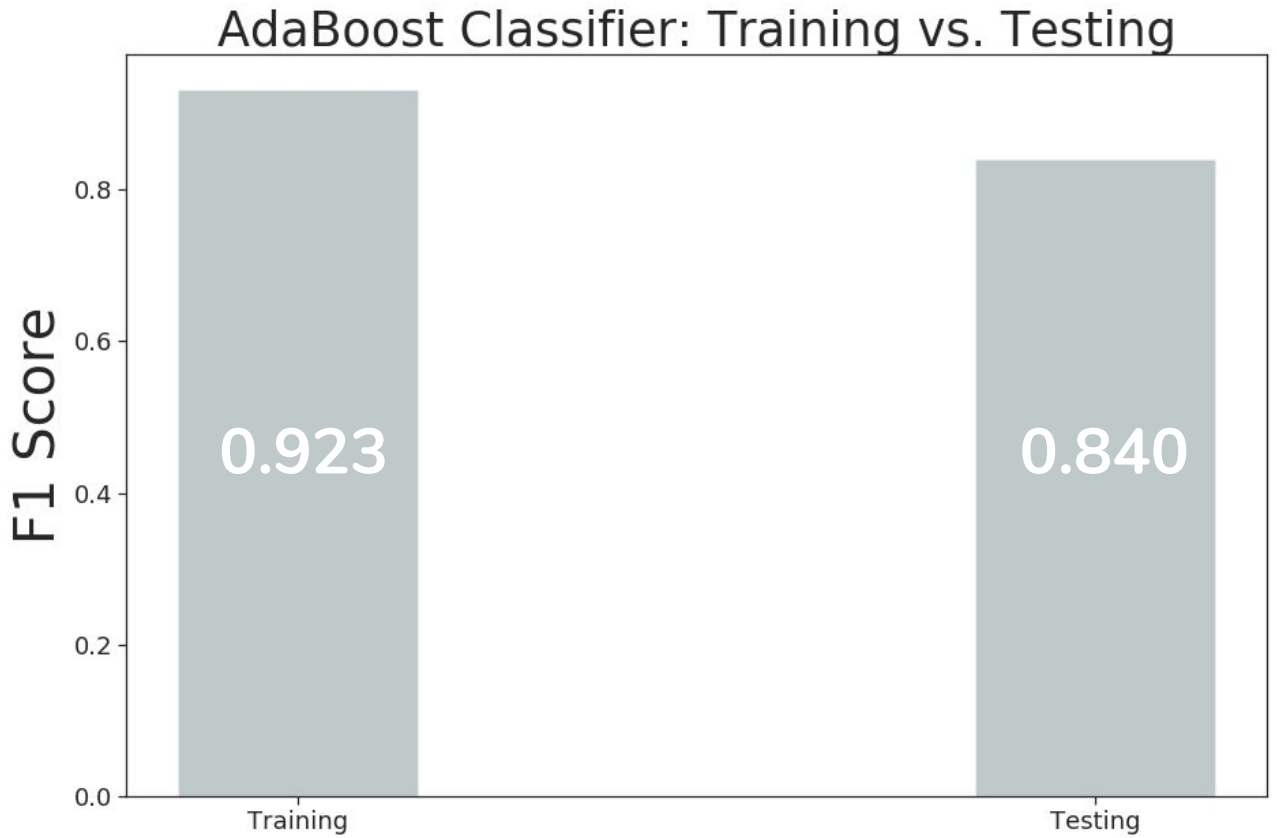
MODEL PERFORMANCE CHALLENGES

Overfitting

Transforming testing data

MODEL PERFORMANCE CHALLENGES

Overfitting	→	Switch Models
Transforming testing data	→	Avoid complex preprocessing transformations


AdaBoost Classifier: Training vs. Testing
F1 Score
0.0
0.2
0.4
0.6
0.8
0.923
0.840
Training
Testing

FINAL MODEL

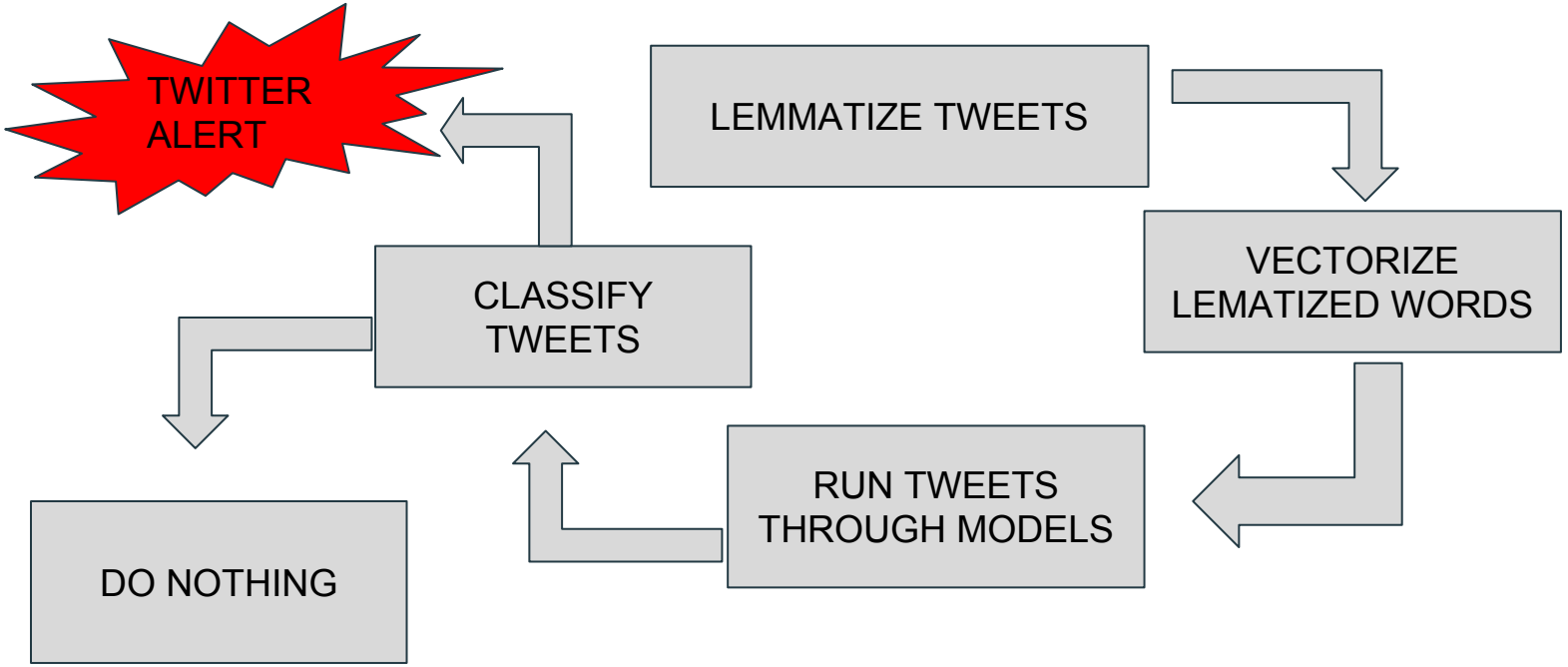
Custom Lemmatization of Tweets

CountVectorized Lemmatized Tweets

Corrected for imbalanced classes using SMOTE

AdaBoost Classifier with standard hyperparameters

HOW APPLICATION WORKS



Successes

- Model that predicts whether or not a tweet is worthy of an urgent alert with a testing f1 score = 0.8400
- Easily translatable across social media platforms
- Workflow provides a simple framework for training models on other types of natural disasters
- Simple to integrate workflow into a web app

RECOMMENDATIONS

- Twitter’s API to be used in gathering tweets.
- Other Natural disasters in modelling
- Integrate a twitter stream function - model will be constantly analyzing input of tweets

SHORTCOMINGS

How can we improve our model if we had more time

- Imbalanced Classes
- Not Enough Time!
- Labeling tweets to build model is challenging
- Label more web-scraped tweets

Stretch Goals

- Build a model that works with other languages (Indonesian, Spanish, Mandarin)
- Improve our model through GridSearch
- More Data!

Questions?