Disaster Recognition

Predicting which Tweets are about real disasters and which ones are not

Kalpa Henadhira Arachchige
Kalyan Lakshmanan
Polina Minkovski

Overview

Problem

Decipher whether or not a Tweet is about a real disaster, in order to improve the credibility and utilization of Twitter as an effective communication tool in times of emergency.

Data

10,000 manually-coded tweets, where:

1: Real disaster

0: Not real disaster

Overview

Purpose:

- Disaster readiness
- Relief organizations

Assumptions:

Two scenarios where we can misclassify

- Regular Tweet is flagged (False Positive)
- Disaster Tweet is ignored (False Negative)

What do we prefer?

- Minimizing False Negatives

Process

		Member 1: PreProcessing Experiment		Member 1: Best Final Model	
Independent EDA	Consolidate Findings	Member 2: Neural Net Experiment	Consolidate EDA	Member 2: Best Final Model	Consolidate Model, README, GitHub
		Member 3: EDA + Model Optimization		Member 3: Best Final Model	Build Streamlit
3 separate notebooks on Git	Pull/Push on Git #1	Pull/Push on Git #2	Pull/Push on Git #3	Pull/Push on Git #4 + Kaggle Submissions	Pull/Push on Git #5

DISCOVERY

DESIGN

ITERATION

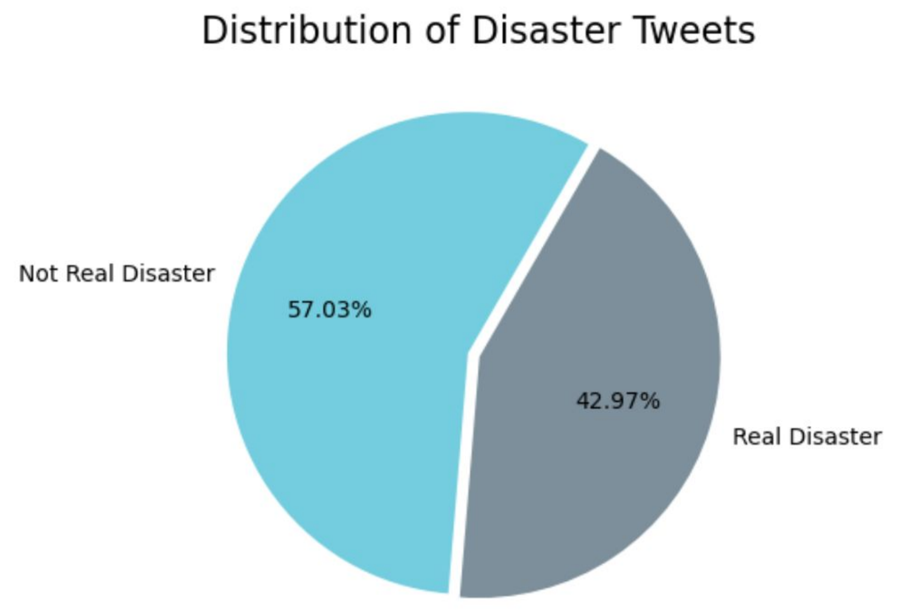
IMPLEMENTATION

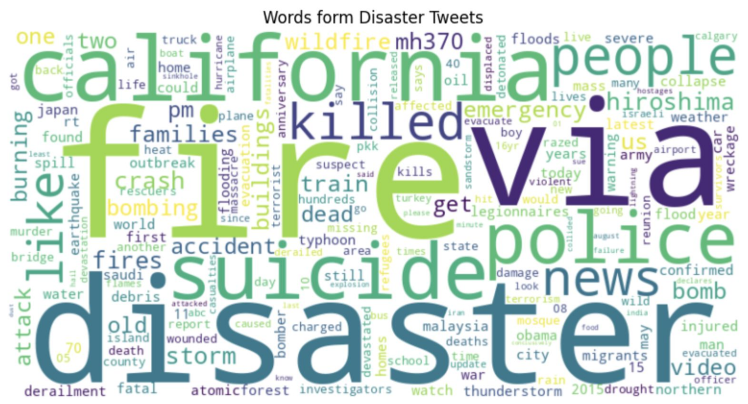
Collection, Cleaning

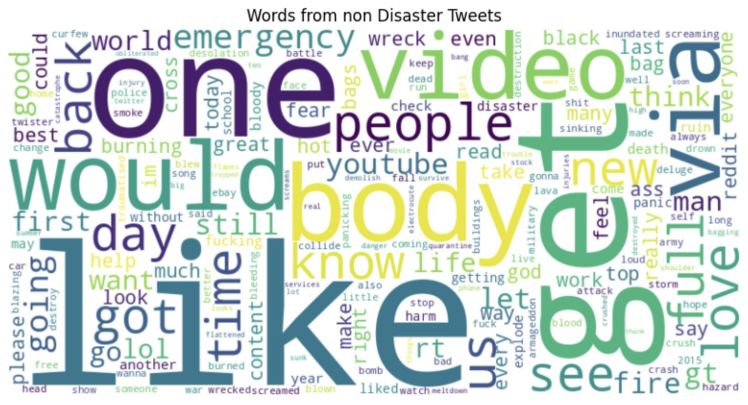
EDA ↔ Model Building

Model Deployment

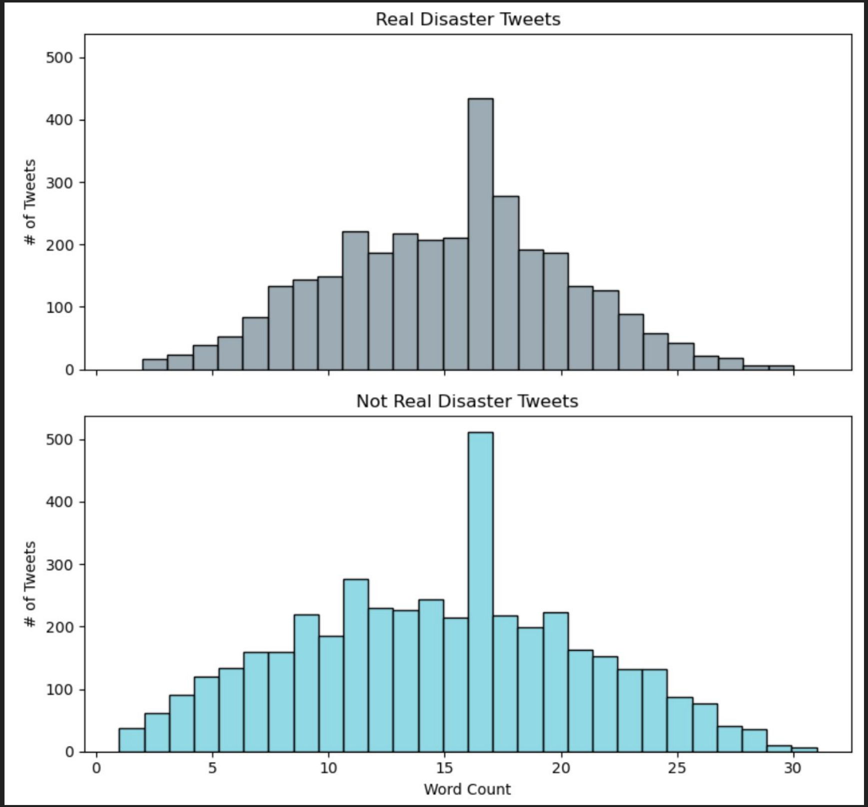
Data Overview

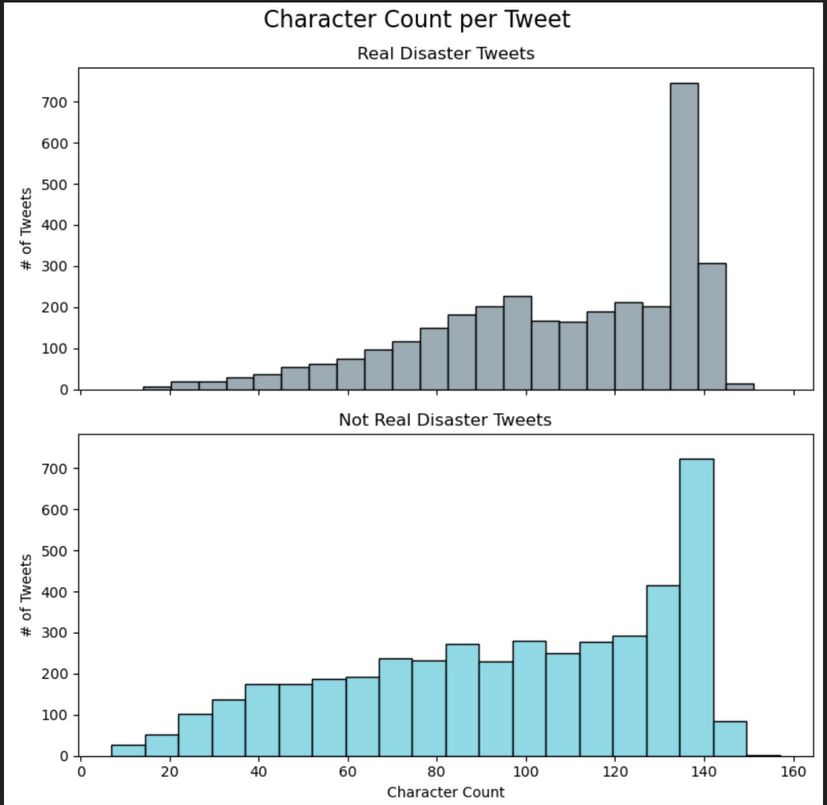






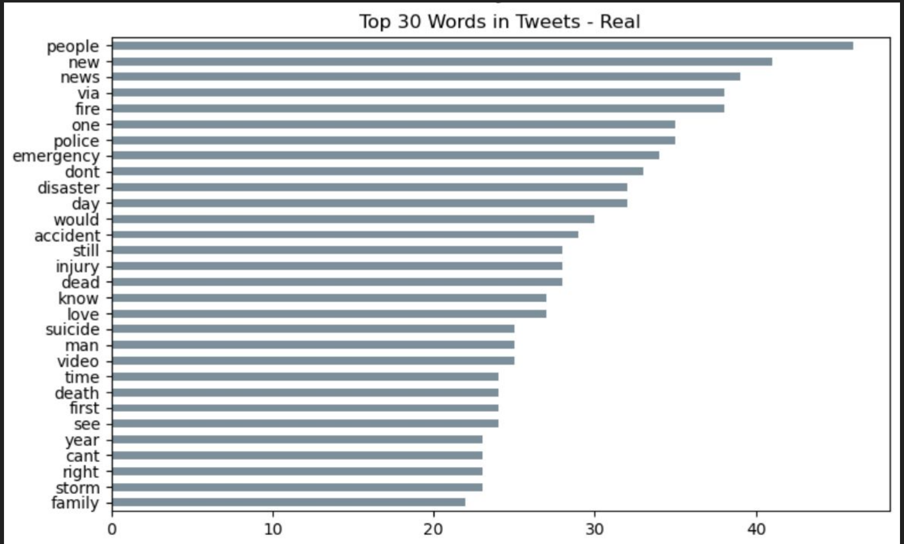
Data Overview

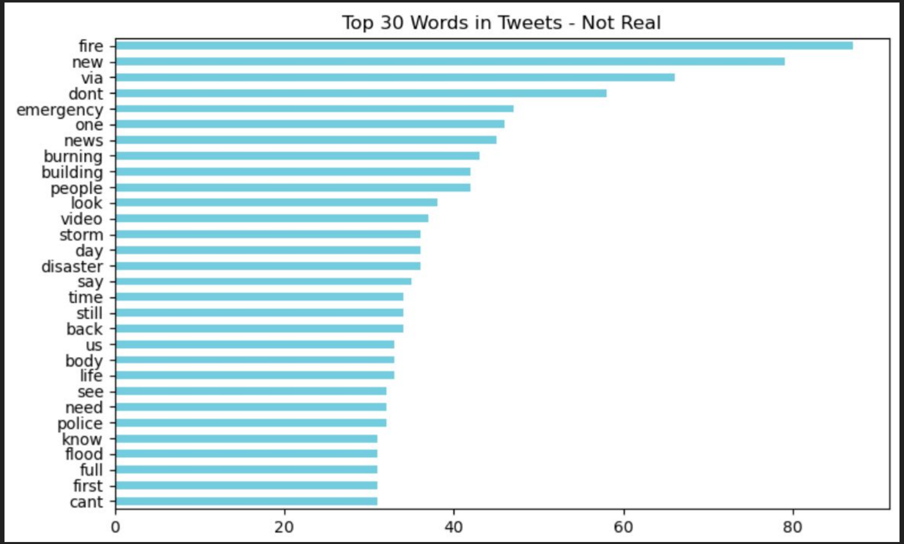




Data Overview - Most frequently used words

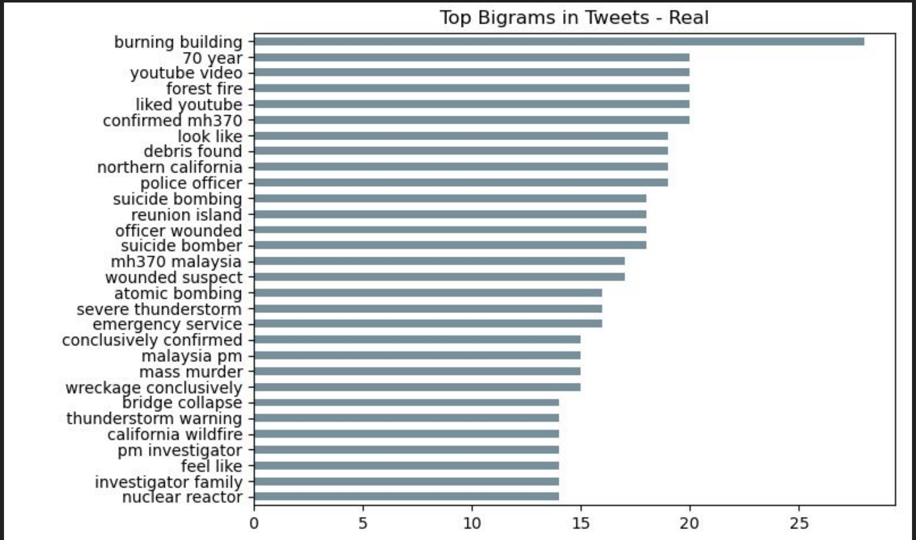
Not Real Disaster Tweets see more word repetition than Real Disaster Tweets

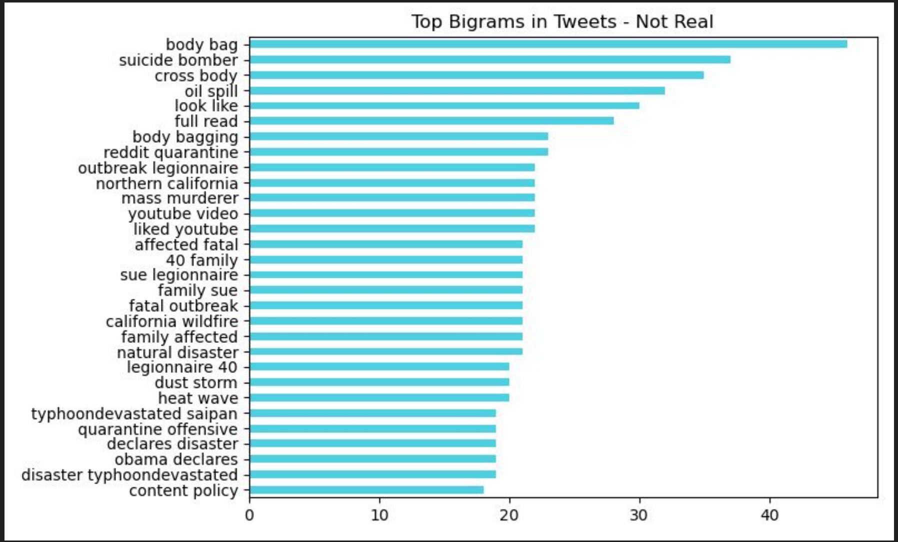




Data Overview - Bigrams

Youtube and Reddit references appear more frequently in Not Real Tweets

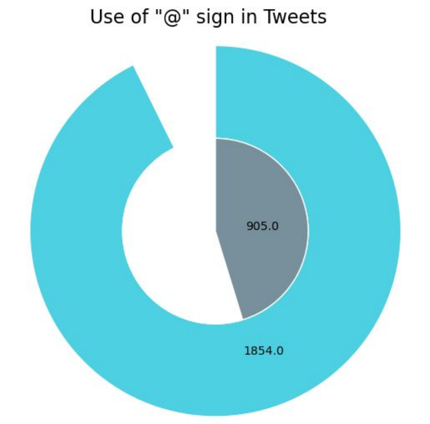


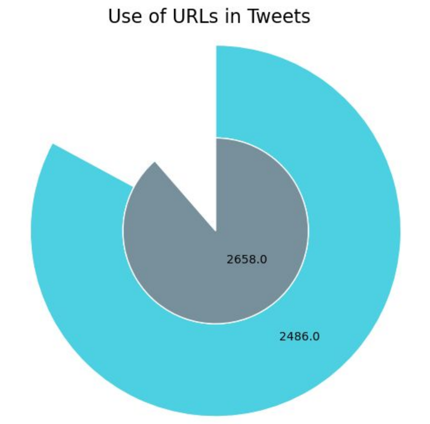


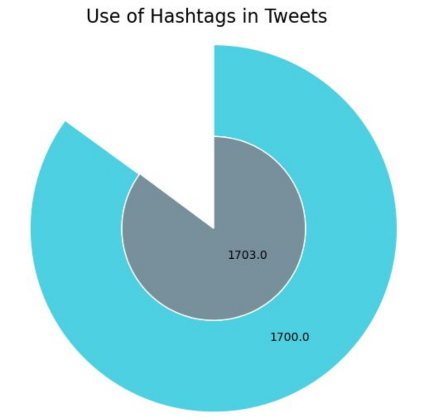
Data Cleaning

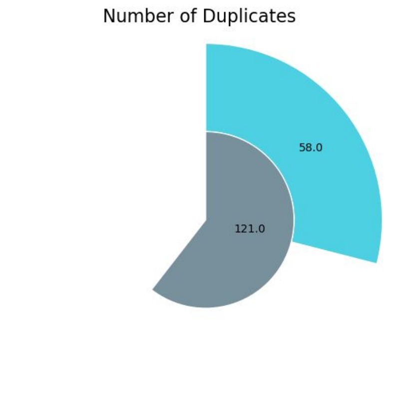
The following steps were completed as part of Data Cleaning:

1. Duplicate tweets removed*
2. Line breaks, @, URLs, Special characters and emoticons removed**
3. Stop words modified and removed
4. Lower case all text
5. Lemmatization completed







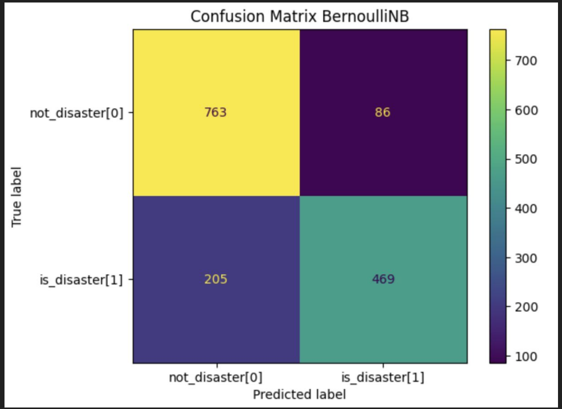


Modeling and Iterations

Model 1

- Overfitting issues
- Low sensitivity as a disadvantage
- Grid search to overcome the overfitting (high variance)

Model	Score on train	Score on test	Sensitivity	Specificity	Precision	F1 Score
logr	0.886	0.800	0.699	0.881	0.823	0.756
Randomfc	0.924	0.799	0.632	0.932	0.880	0.736
KNN	0.826	0.711	0.467	0.905	0.795	0.588
BernoulliNB	0.854	0.809	0.696	0.899	0.845	0.763
LinearSVC	0.855	0.792	0.666	0.892	0.830	0.739
Adaboost(RFC)	0.987	0.809	0.708	0.889	0.835	0.766

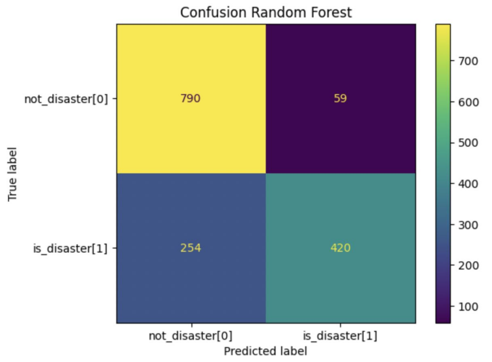


Modeling and Iterations

Random forest

```
gs_rfc.best_params_
```

```
{'rfc__bootstrap': True,
 'rfc__ccp_alpha': 0.0,
 'rfc__max_depth': None,
 'rfc__max_features': 'log2',
 'rfc__max_samples': 0.5,
 'rfc__min_impurity_decrease': 0.0,
 'rfc__min_samples_leaf': 1,
 'rfc__min_samples_split': 20,
 'rfc__n_estimators': 200,
 'rfc__oob_score': True,
 'rfc__warm_start': True,
 'vec': CountVectorizer(max_df=0.9, max_features=5000, min_df=5, ngram_range=(1, 2)),
 'vec__max_df': 0.9,
 'vec__max_features': 5000,
 'vec__min_df': 5,
 'vec__ngram_range': (1, 2),
 'vec__stop_words': None}
```

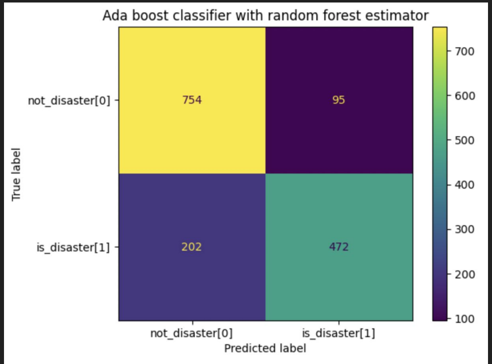


- Does gridsearch help?
- What other things to try?

Adaboost classifier with Random forest estimator

```
gs_ada.best_params_
```

```
{'ada__algorithm': 'SAMME.R',
 'ada__learning_rate': 1.0,
 'ada__n_estimators': 5,
 'vec': CountVectorizer()}
```

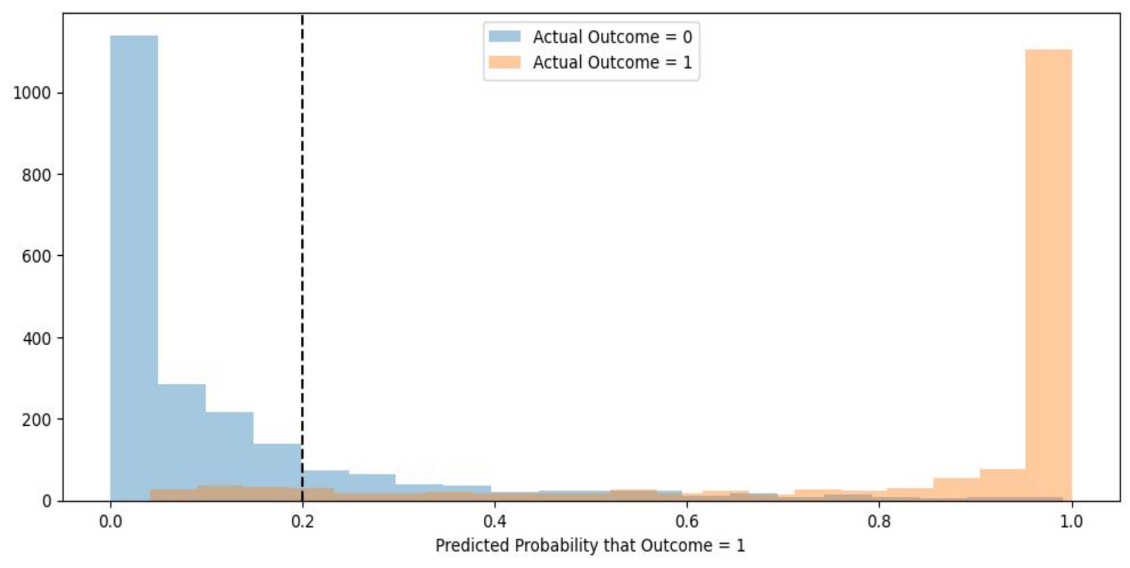


Modeling and Iterations

LSTM with GloVe Embeddings

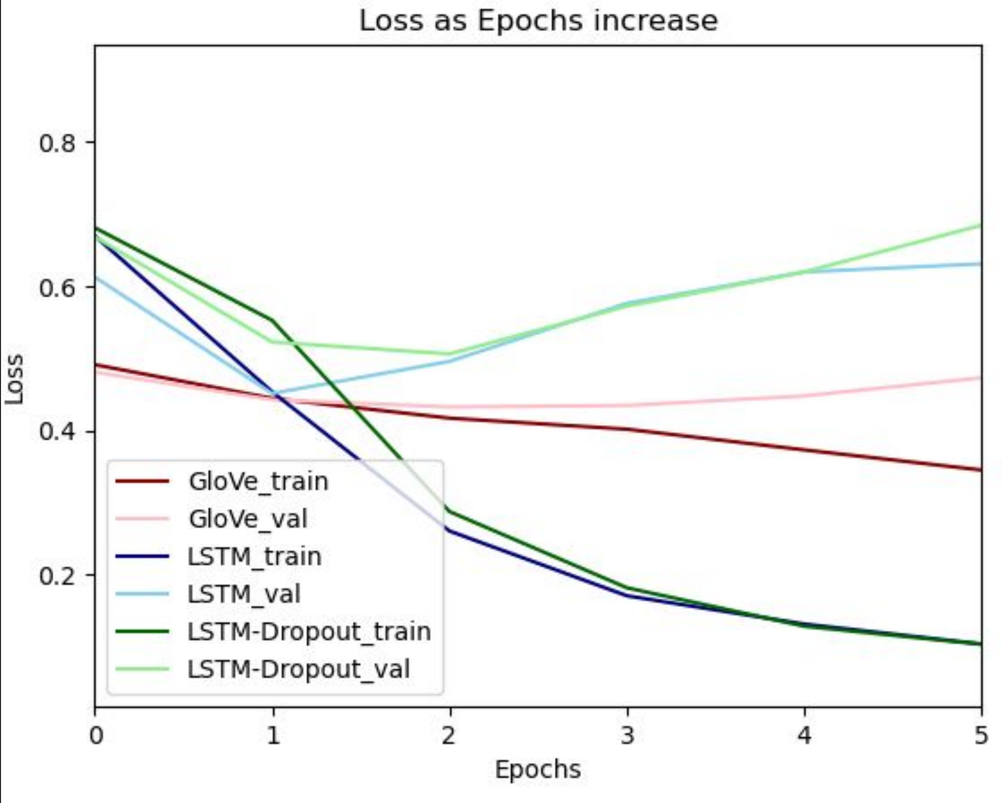
Sensitivity: 87.78%
Accuracy: 74.37%

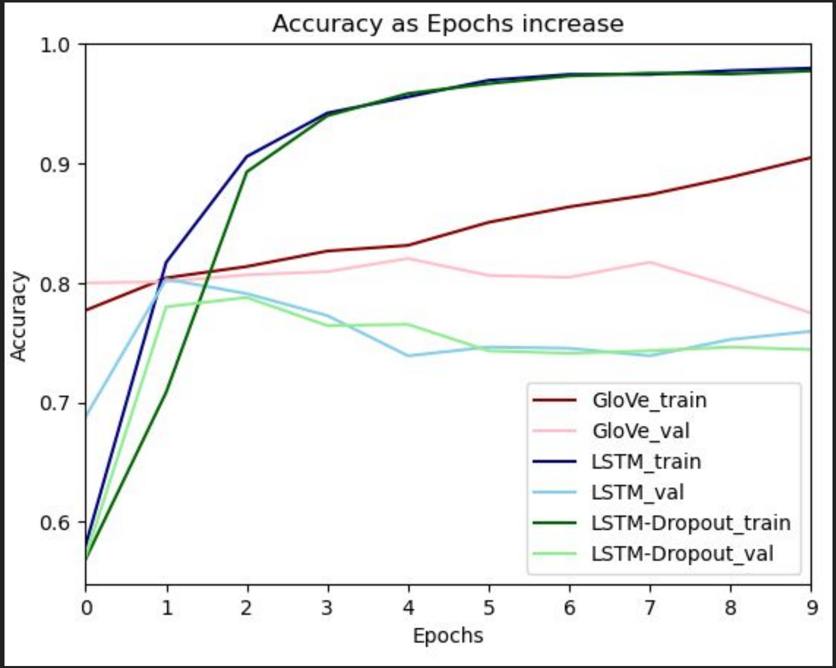
- Low Variance
- Generalizable
- GloVe Embeddings:
  Pre-trained Word ⟶ Vector
  2B Tweets, 1.2M Vocab, 25d



Aggressive 0.2 threshold for minimizing false negatives
Original – Accuracy: 84.2% and Validation Accuracy:81.6%

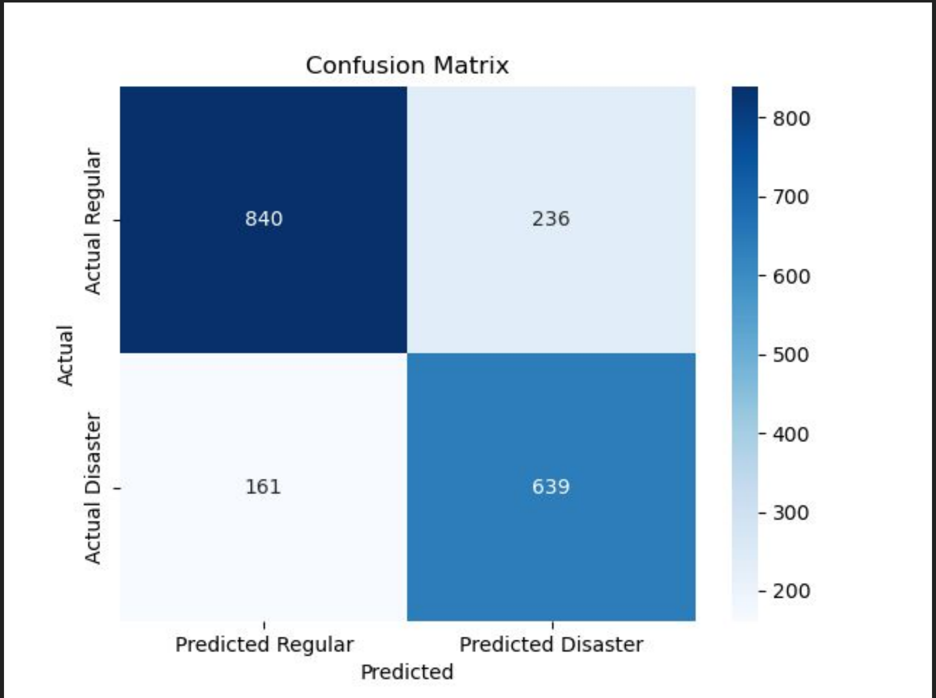
Model Comparisons



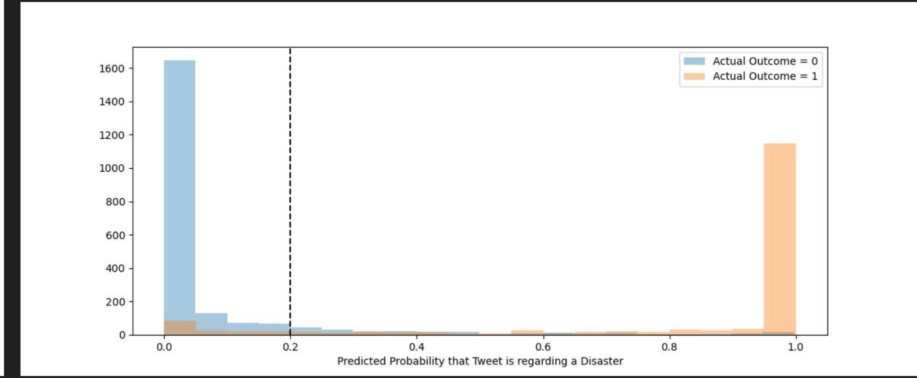


Modeling and Iterations

Model 3 - Bernoulli, CountVectorizer with GridSearch and Adjusted Threshold



With adjusted threshold:
Sensitivity: 79.88%
Specificity: 78.07%
Validation Accuracy: 78.83%

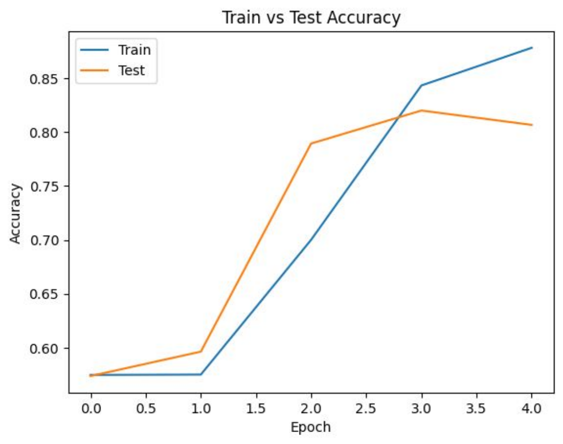


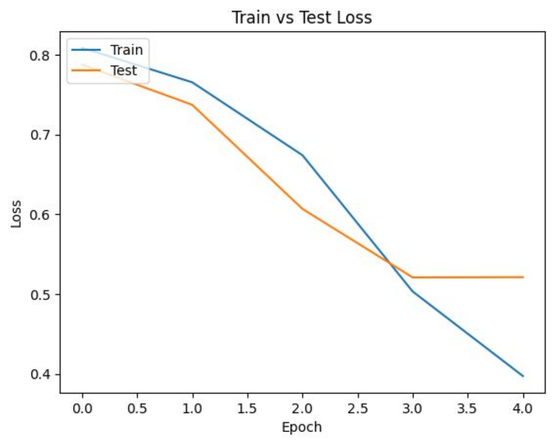
Comparing outcomes

Model	Score on train	Score on test	Sensitivity	Specificity	Precision	F1 Score
Bernoulli CVEC	0.838	0.817	0.680	0.918	0.861	0.760
Bernoulli TVEC	0.838	0.817	0.680	0.918	0.861	0.760
Logistic Regression CVEC	0.894	0.815	0.691	0.907	0.847	0.761
Logistic Regression TVEC	0.911	0.804	0.708	0.876	0.810	0.756
Random Forest CVEC	0.987	0.780	0.688	0.849	0.771	0.727
Random Forest TVEC	0.987	0.788	0.675	0.873	0.798	0.731
Decision Tree CVEC	0.881	0.746	0.671	0.802	0.716	0.693
Decision Tree TFIDF	0.891	0.735	0.659	0.791	0.701	0.67

Comparing outcomes

Model	Loss	Accuracy	Sensitivity	Specificity	Precision	F1 Score
Sequential, Bidirectional, LSTM	0.97	0.76	0.72	0.78	0.71	0.71
Sequential, Bidirectional, GRU	0.5	0.8	0.71	0.86	0.79	0.74
Sequential, Bidirectional, GRU, Regularization	0.51	0.81	0.73	0.87	0.8	0.76
Sequential, Word2Vec, EarlyStopping	0.52	0.81	0.71	0.87	0.81	0.75





Findings and Outcomes

Streamlit App - prototype, Utilization + Threshold

Future use:

- Deploy with live twitter stream - twitter stream API
- Real time information on disasters