Natural Language Processing with Disaster Tweets

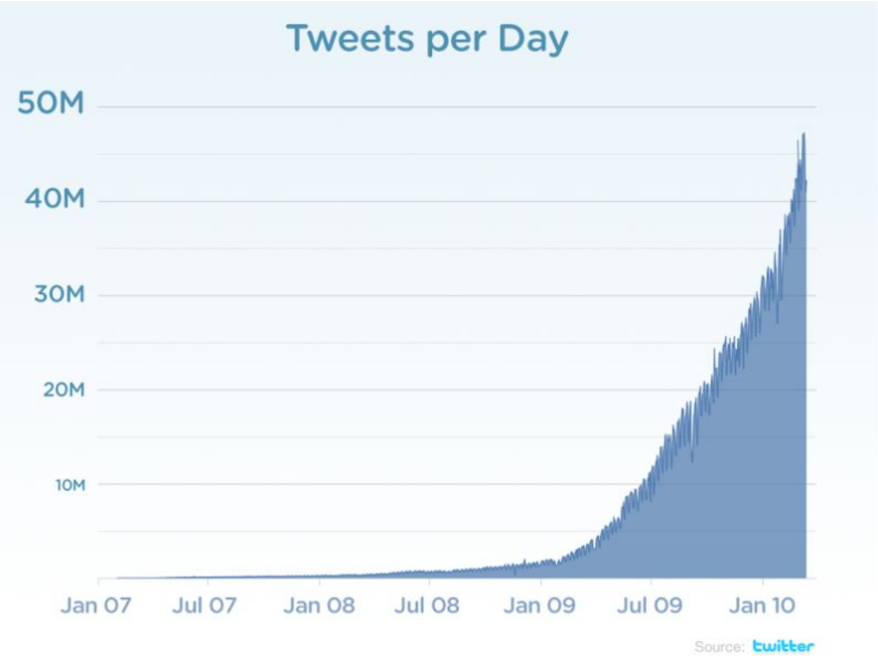
Max Tjen and Jerry Xiao

Background

- Twitter has become a go-to platform for real-time public messages
- 6,000 tweets per second

  ⇒ 500 million tweets per day

- Agencies are interested in monitoring Twitter
  - News agencies
  - Disaster relief agencies
  - Sentiment analysis: political figures, stocks, crypto, etc...



The Problem

- Dataset of over 7,000 tweets
- Each one may or may not be related to natural disaster
- Classify each tweet: is it discussing a natural disaster or not?

Significance

- Twitter is a huge trove of information

- Monitoring Tweets could be be a way to gather information about important news events in real-time

The Data

- Train data (7614 rows, labeled) for building model
- Test data (3264 rows, unlabeled) for predicting and submitting to kaggle

Given Variables:

`id` - a unique identifier for each tweet

`text` - the text of the tweet

`location` - the location the tweet was sent from (may be blank)

`keyword` - a particular keyword from the tweet (may be blank)

`target` - in **train.csv** only, this denotes whether a tweet is about a real disaster (`1`) or not (`0`)

The Data

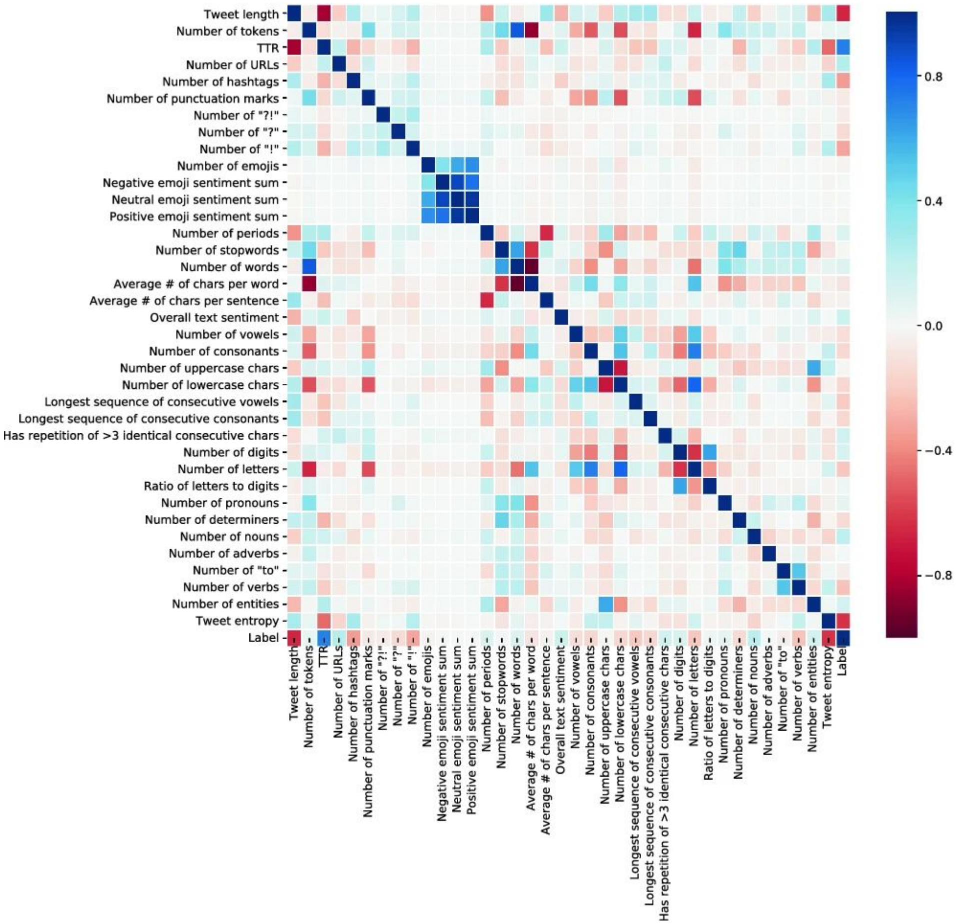
	id	keyword	location	text	target
0	1	NaN	NaN	Our Deeds are the Reason of this #earthquake M...	1
1	4	NaN	NaN	Forest fire near La Ronge Sask. Canada	1
2	5	NaN	NaN	All residents asked to 'shelter in place' are ...	1
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1
4	7	NaN	NaN	Just got sent this photo from Ruby #Alaska as ...	1
...	...	...	...	...	...
7608	10869	NaN	NaN	Two giant cranes holding a bridge collapse int...	1
7609	10870	NaN	NaN	@aria_ahrary @TheTawniest The out of control w...	1
7610	10871	NaN	NaN	M1.94 [01:04 UTC]?5km S of Volcano Hawaii. htt...	1
7611	10872	NaN	NaN	Police investigating after an e-bike collided ...	1
7612	10873	NaN	NaN	The Latest: More Homes Razed by Northern Calif...	1

7613 rows × 5 columns

Relevant Works

Hong Kong Protests: Using Natural Language Processing for Fake News Detection on Twitter:

- Dataset of tweets that are real/fake news regarding Hong Kong protests
- Feature extraction
  - Features are purely linguistic
- Tested Naive Bayes, SVMs, Decision Trees for predicting veracity of news
  - For all but random trees, top 10 correlated features are chosen and normalized
- 10 most significant features: tweet length, number of punctuation marks, number of periods, average number of characters per sentence, number of adverbs, number of “to”, number of verbs, number of entities(nouns), tweet entropy, TTR


Tweet length
Number of tokens
TTR
Number of URLs
Number of hashtags
Number of punctuation marks
Number of "?!"
Number of "?"
Number of "!"
Number of emojis
Negative emoji sentiment sum
Neutral emoji sentiment sum
Positive emoji sentiment sum
Number of periods
Number of stopwords
Number of words
Average # of chars per word
Average # of chars per sentence
Overall text sentiment
Number of vowels
Number of consonants
Number of uppercase chars
Number of lowercase chars
Longest sequence of consecutive vowels
Longest sequence of consecutive consonants
Has repetition of >3 identical consecutive chars
Number of digits
Number of letters
Ratio of letters to digits
Number of pronouns
Number of determiners
Number of nouns
Number of adverbs
Number of "to"
Number of verbs
Number of entities
Tweet entropy
Label
0.8
0.4
0.0
−0.4
−0.8

Results from Hong Kong paper

Algorithm	Class	Precision	Recall	F1 Score
Naive Bayes	Fake	90.1%	85.4%	87.6%
	Real	89.7%	92.8%	91.2%
	Average	89.9%	89.1%	89.4%
SVM	Fake	96.0%	84.0%	89.6%
	Real	89.4%	97.5%	93.3%
	Average	92.7%	90.8%	91.4%
C4.5	Fake	94.7%	84.7%	89.3%
	Real	89.8%	96.6%	93.0%
	Average	92.3%	90.6%	91.2%
Random Forest	Fake	97.5%	84.3%	90.3%
	Real	89.7%	98.4%	93.8%
	Average	93.6%	91.3%	92.1%

Relevant Works

Conclusions from Hong Kong paper:

Results indicate that tweets spreading fake news and real news differ notably in linguistic features.

Possible Caveats/Flaws:

Many tweets were translated from Chinese to english

No features related to user/network characteristics

Models tested aren’t very modern (could try NN’s)

OUR METHOD

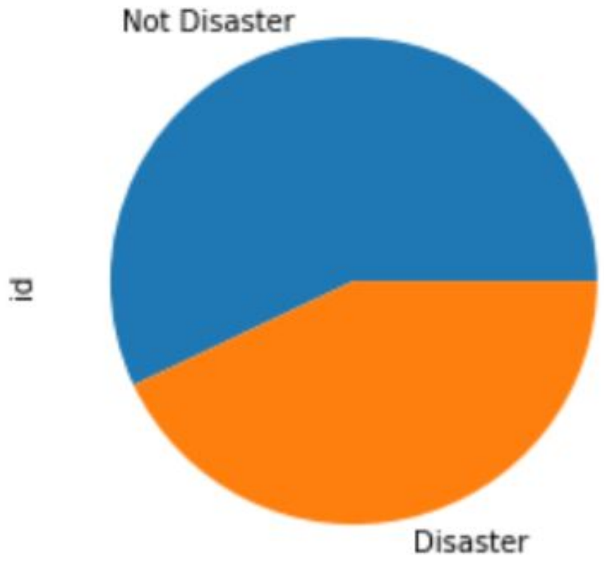
We follow the same methods as the Hong Kong paper with some modifications:

- Clean the Tweets
- Extract linguistic features
- Gather sentiment from tweets
- RandomForest to classify based on our features

Data Cleaning

```
MISSING:
id             0
keyword       61
location    2533
text           0
target         0
dtype: int64

DUPLICATE:
0
```

```
Not disaster: 57.03402075397347%
Is disaster: 42.9659792460265 3%
```



Data Cleaning

- Non printable characters were removed (emojis, symbols)
- Character references were removed (&)

For Textblob and Vader sentiment

- All letters to lowercase
- All but alphanumeric stripped out

```
                                                   text  \
0     Our Deeds are the Reason of this #earthquake M...
1                Forest fire near La Ronge Sask. Canada
2     All residents asked to 'shelter in place' are ...
3     13,000 people receive #wildfires evacuation or...
4     Just got sent this photo from Ruby #Alaska as ...
...                                                 ...
7608  Two giant cranes holding a bridge collapse int...
7609  @aria_ahrary @TheTawniest The out of control w...
7610  M1.94 [01:04 UTC]?5km S of Volcano Hawaii. htt...
7611  Police investigating after an e-bike collided ...
7612  The Latest: More Homes Razed by Northern Calif...

                                            cleanedTweets
0     our deeds are the reason of this earthquake ma...
1                 forest fire near la ronge sask canada
2     all residents asked to shelter in place are be...
3     13000 people receive wildfires evacuation orde...
4     just got sent this photo from ruby alaska as s...
...                                                 ...
7608  two giant cranes holding a bridge collapse int...
7609  aria_ahrary thetawniest the out of control wil...
7610  m194 0104 utc5km s of volcano hawaii httptcozd...
7611  police investigating after an ebike collided w...
7612  the latest more homes razed by northern califo...
```

Feature Engineering

- 36 Predictive Features

- Categories
  - Boolean
  - Overall Tweet Characteristics
  - Types of Word (grammar)
  - Sentiment
  - TTR and Entropy

```
charsToCheck = {'!', '@', '#', '?', '.', ',', 'http'}
vowels = {'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U'}
nouns = {'NN', 'NNS', 'NNP', 'NNPS'}
verbs = {'VB', 'VBD', 'VBG', 'VBN', 'VBP', 'VBZ'}
adj = {'JJ', 'JJR', 'JJS'}
adv = {'RR', 'RBR', 'RBS'}

hasLocation = []
hasKeyword = []
tweetNumberOfChars = []
specialCharacters = []
numberOfWords = []
avgCharsPerWord = []
numNumericTweet = []
numLettersTweet = []
numUpperTweet = []
numVowelsTweet = []
numConsonantsTweet = []
numNouns = []
numVerbs = []
numPrep = []
numAdj = []
numAdv = []
numProperNoun = []
```

Feature Engineering: Boolean

```
##### location
location = 0
if pd.isnull(train['location'][index]) == False:
        location = 1

##### keyword
keyword = 0
if pd.isnull(train['keyword'][index]) == False:
        keyword = 1
```

Feature Engineering: Overall Tweet

- Bulk of features
  - Number of characters
  - Number of words
  - Number of special characters
    - ! @ , # ? .
  - Average characters per word
  - Number or numeric and letters in tweet
  - Number of consonants and vowels
  - Number of special characters per sentence

```
for char in word:
    if char.isnumeric():
        numNumeric += 1
    if char.isalpha():
        numLetters += 1
        if char.isupper():
            numUpper += 1
        if char in vowels:
            numVowels += 1
        else:
            numConsonants += 1
```

```
# number of specific special characters in tweet
specialChars = []
for specialChar in charsToCheck:
    numSpecialChar = text.count(specialChar)
    specialChars.append(numSpecialChar)
```

Feature Engineering: Grammar

- NLTK package
  - Natural language toolkit
- Pass array of words into function
- Function tags each word
- Add counts for each type

```
ans = nltk.pos_tag(words)
for pair in ans:
    wordType = pair[1]
    if (wordType == 'NNP') or (wordType == 'NNPS'):
        properNounNum += 1
    elif wordType in nouns:
        nounNum += 1
    elif wordType in verbs:
        verbNum += 1
    elif wordType == 'IN':
        prepNum += 1
    elif wordType in adj:
        adjNum += 1
    elif wordType in adv:
        advNum += 1
```

Feature Engineering: Sentiment

- Representation of emotion in text
- Textblob
  - Built on NLTK
  - Sees if a word is positive, negative, or neutral and then calculates average of text
- Vader
  - Rule based (grammar and syntax conventions)
  - Dictionary of words that maps words to emotional intensity
  - Optimized for social media data - tuned for small amounts of text

```
# textblob sentiment
train["sentimentText"] = train['cleanedTweets'].apply(lambda x:
                                                       TextBlob(x).sentiment.polarity)

# vader sentiment
train["sentimentVader"] = train['cleanedTweets'].apply(lambda x:
                                                        analyser.polarity_scores(x)['compound'])
```

Feature Engineering: Sentiment

cleanedTweets	sentimentText	sentimentVader	target
Our Deeds are the Reason of this #earthquake M...	0.000000	0.2732	1
Forest fire near La Ronge Sask. Canada	0.100000	-0.3400	1
All residents asked to 'shelter in place' are ...	-0.018750	-0.2960	1
13,000 people receive #wildfires evacuation or...	0.000000	0.0000	1
Just got sent this photo from Ruby #Alaska as ...	0.000000	0.0000	1
...	...	...	...
Two giant cranes holding a bridge collapse int...	0.000000	-0.4939	1
@aria_ahrary @TheTawniest The out of control w...	0.150000	-0.5849	1
M1.94 [01:04 UTC]?5km S of Volcano Hawaii. htt...	0.000000	0.0000	1
Police investigating after an e-bike collided ...	-0.260417	-0.7845	1
The Latest: More Homes Razed by Northern Calif...	0.500000	0.0000	1

Feature Engineering: TTR and Entropy

TTR (type-token ratio) - total number of unique words divided by total number of words in a document (tweet)

Entropy - sum of all TF-IDF values of words for each tweet

- Term frequency in document - inverse document frequency across set of documents
    - The more times a word is used ⇒ it is less important (ex. ‘and’)
- Measure of word importance

```
tfidfvectorizer = TfidfVectorizer(analyzer='word',stop_words= 'english')
tfidf_wm = tfidfvectorizer.fit_transform(train['cleanedTweets'])

#retrieve the terms found in the corpora
tfidf_tokens = tfidfvectorizer.get_feature_names()

# Get TFIDF representation of tweets
df_tfidfvect = pd.DataFrame(data = tfidf_wm.toarray(),index = list(train.index.values) ,columns = tfidf_tokens)
```

Resulting Features

- hasLocation
- hasKeyword
- tweetNumberOfChars
- numberOfWords
- numEx
- numAt
- numHash
- numQ
- numPeriod
- numComma
- numLinks
- numPunc
- avgCharsPerWord
- numNumericTweet
- numLettersTweet
- numUpperTweet
- numVowelsTweet
- numConsonantsTweet
- numNouns
- numNouns
- numVerbs
- numPrep
- numAdj
- numAdv
- charsPerSentenceTweet
- numSentencesTweet
- numProperNoun
- numTo
- ttr
- entropy
- hashPerSentence
- wordsPerSentence
- atPerSentence
- commasPerSentence
- linksPerSentence
- sentimentText
- sentimentVader

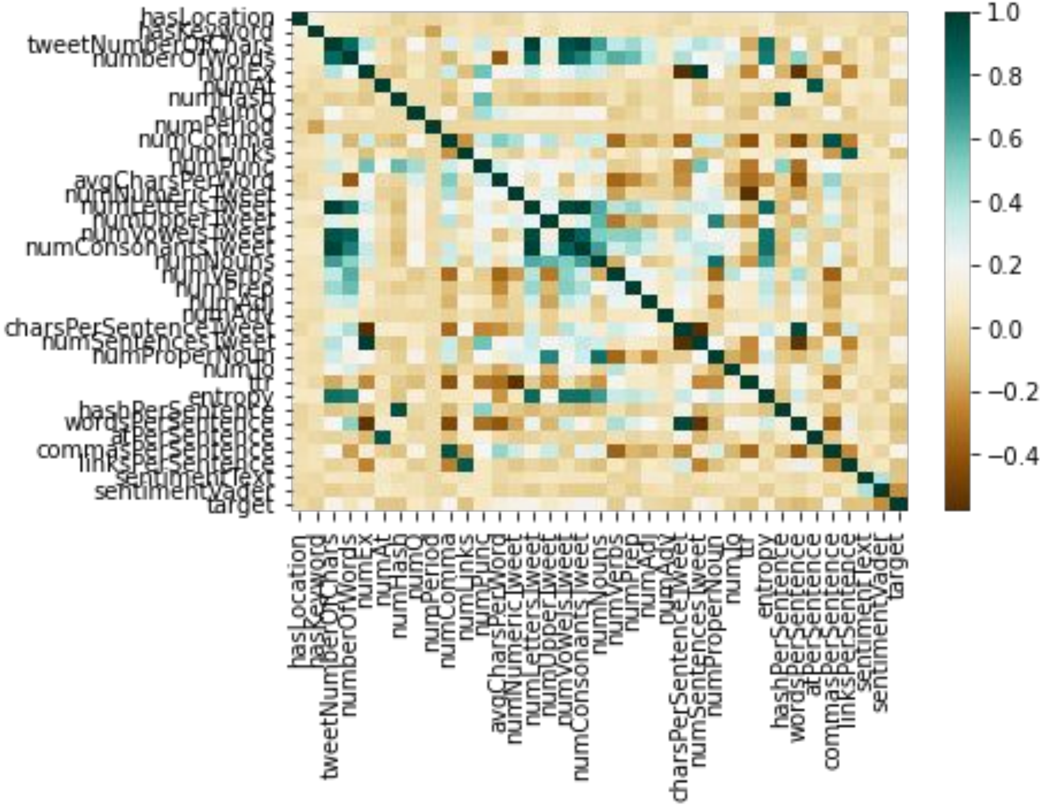
Resulting Features

	hasLocation	hasKeyword	tweetNumberOfChars	numberOfWords	numEx	numAt	numHash	numQ	numPeriod	numComma	...	ttr	entropy
0	0	0	69	13	0	0	0	1	0	0	...	1.000000	2.215113
1	0	0	38	7	1	0	0	0	0	0	...	1.000000	2.406114
2	0	0	133	22	1	0	0	0	0	0	...	0.818182	2.837094
3	0	0	65	8	0	0	0	1	1	0	...	0.875000	2.393269
4	0	0	88	16	0	0	0	2	0	0	...	0.937500	3.081889
...	...	...	...	...	...	...	...	...	...	...	...	...	...
7608	0	0	83	11	1	0	0	0	0	1	...	0.909091	2.634344
7609	0	0	125	20	2	0	0	0	0	0	...	0.800000	2.933941
7610	0	0	65	8	3	0	1	0	0	1	...	0.500000	1.410888
7611	0	0	137	19	2	0	0	0	0	0	...	0.947368	3.220383
7612	0	0	94	13	1	0	0	0	0	1	...	0.846154	2.814050

7613 rows × 37 columns

Resulting Features

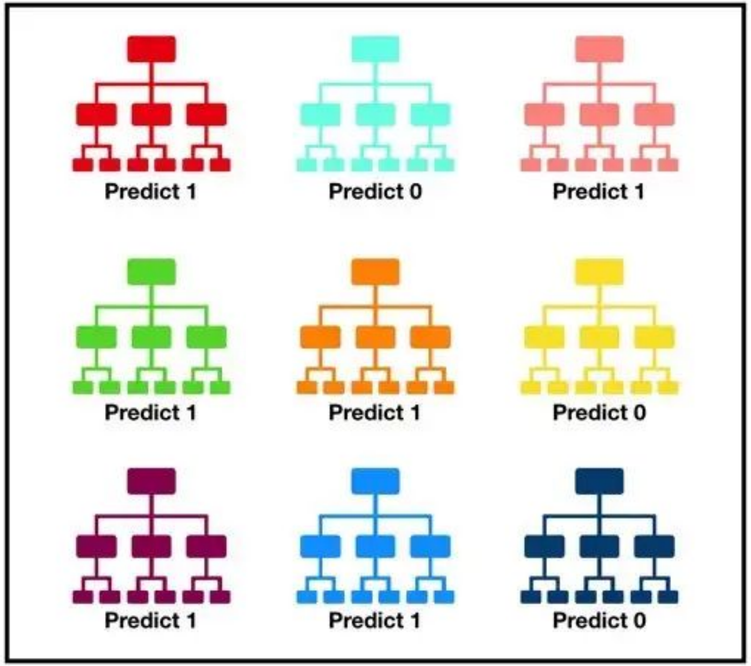
sentimentVader	0.232405
numComma	0.195455
numConsonantsTweet	0.190908
tweetNumberOfChars	0.184401
commasPerSentence	0.183376
numLettersTweet	0.182460
numNouns	0.175625
numNumericTweet	0.172301
numEx	0.156265
numSentencesTweet	0.156265
numPrep	0.155628
numVowelsTweet	0.151178
avgCharsPerWord	0.137535
entropy	0.129417
wordsPerSentence	0.117802



RandomForest Classifier

How it works:

- Many individual decision trees: each makes prediction as a vote
    - Each tree gets a random subset of data with replacement, and random subset of features
    - More diversification, lower correlation across trees



Tally: Six 1s and Three 0s
Prediction: 1

RandomForest Classifier: Default Model

80-20 train-test split

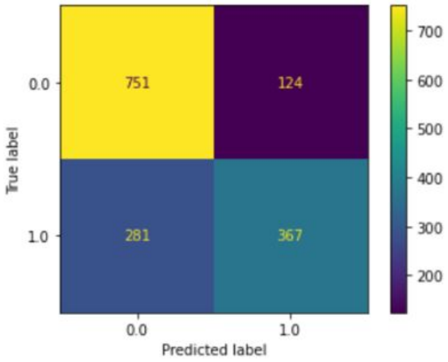
First trained on default RandomForestClassifier():

- n_estimators = 100	100 trees
- max_depth = none	no max depth of trees
- max_features = sqrt	for each split, consider sqrt(num_features)

RandomForest Classifier: Default Model

```
train accuracy: 0.9975369458128078
 test accuracy: 0.7340774786605384
Classification Report
              precision    recall  f1-score   support

         0.0       0.73      0.86      0.79       875
         1.0       0.75      0.57      0.64       648

    accuracy                           0.73      1523
   macro avg       0.74      0.71      0.72      1523
weighted avg       0.74      0.73      0.73      1523


Confusion Matrix
```

```
Out[51]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1fa21440250>
```



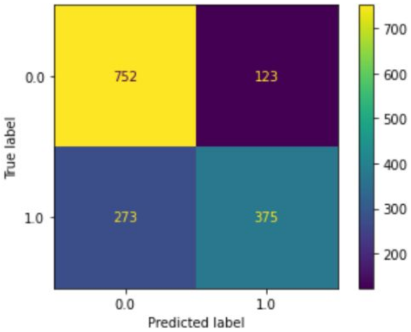
RandomForest Classifier: Optimization

Hyperparameter Optimization:

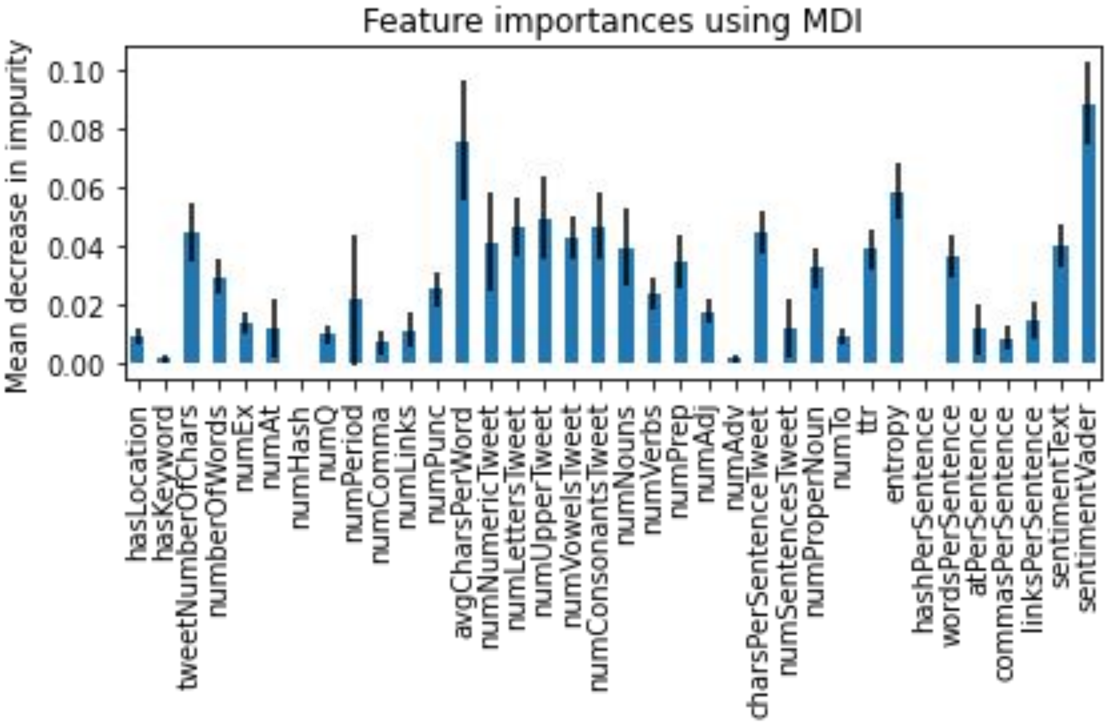
- Create a grid: n_estimators 200-2000
  max_depth 100-500, none
  max_features none, sqrt
- Randomly search grid and test parameters (tested 50 candidates)
- result: n_estimators: 200
  Max_depth: none
  max_features sqrt

Results: Optimized Model

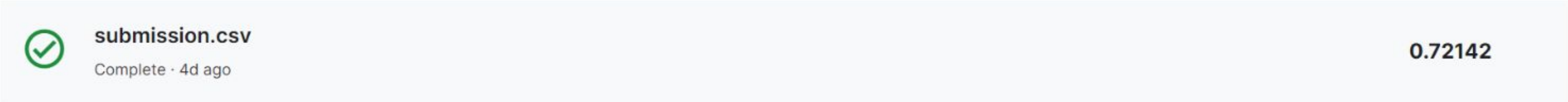
```
train accuracy: 0.9975369458128078
 test accuracy: 0.7399868680236376
Classification Report
              precision    recall  f1-score   support

         0.0       0.73      0.86      0.79       875
         1.0       0.75      0.58      0.65       648

    accuracy                           0.74      1523
   macro avg       0.74      0.72      0.72      1523
weighted avg       0.74      0.74      0.73      1523


Confusion Matrix

<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1fa2568d670>
```



Results: Optimized Model



Results: Kaggle Competition Entry



Analysis

Our accuracy is much lower than the Hong Kong paper with similar methods:

- Different implementation of entropy
- Our data is generally less formal
    - In Hong Kong paper, all tweets were trying to look like real news
    - Our data contains a lot of normal tweets from normal users
- Different data collection
    - Ours: pool of tweets chosen because of potential keyword
    - Theirs: had to separately procure fake news data and real news data: does not come from the same pool

Conclusion:

There is a difference in linguistic features between tweets regarding natural disaster and tweets not regarding natural disaster. The most significant features were:

Feature	MDI
sentimentVader	0.088286
avgCharsPerWord	0.075707
entropy	0.058457
numUpperTweet	0.049401
numConsonantsTweet	0.046705

Conclusion: Future Work

- More in-depth data cleaning
  - Twitter data generally requires more cleaning due to very relaxed grammar
  - Lots of acronyms, purposely misspelled words (yes vs. yesssss)

- Method to validate location values

- BERT (Bidirectional Encoder Representations from Transformers)

- SVM (support vector machine)

References

Di Pietro, Mauro. “Text Analysis & Feature Engineering with NLP.” *Medium*, Towards Data Science, 15 Mar. 2022, https://towardsdatascience.com/text-analysis-feature-engineering-with-nlp-502d6ea9225d.

Zervopoulos, Alexandros, et al. “Hong Kong Protests: Using Natural Language Processing for Fake News Detection on Twitter.” *IFIP Advances in Information and Communication Technology*, 29 May 2020, pp. 408–419., https://doi.org/10.1007/978-3-030-49186-4_34.