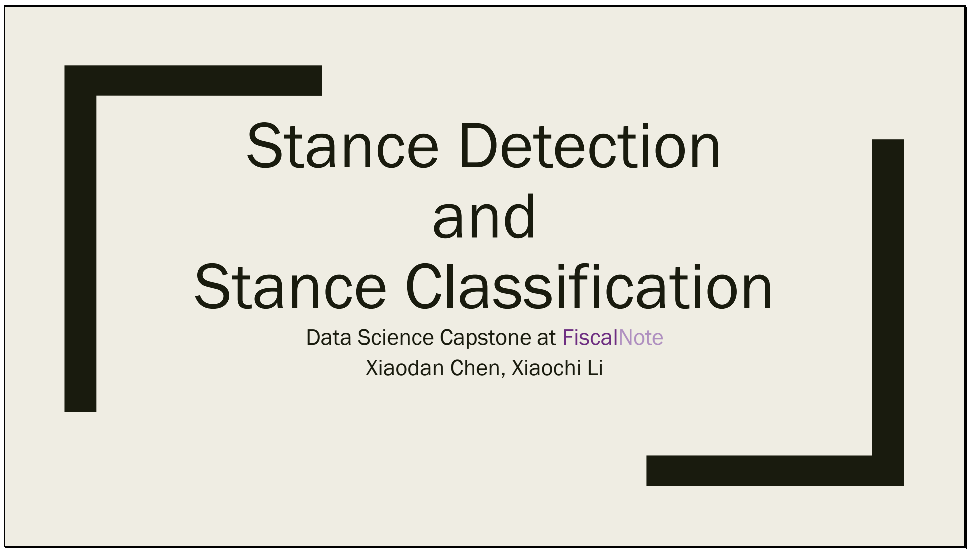
Slide 1



Stance Detection and Stance Classification

Data Science Capstone at FiscalNote

Xiaodan Chen, Xiaochi Li

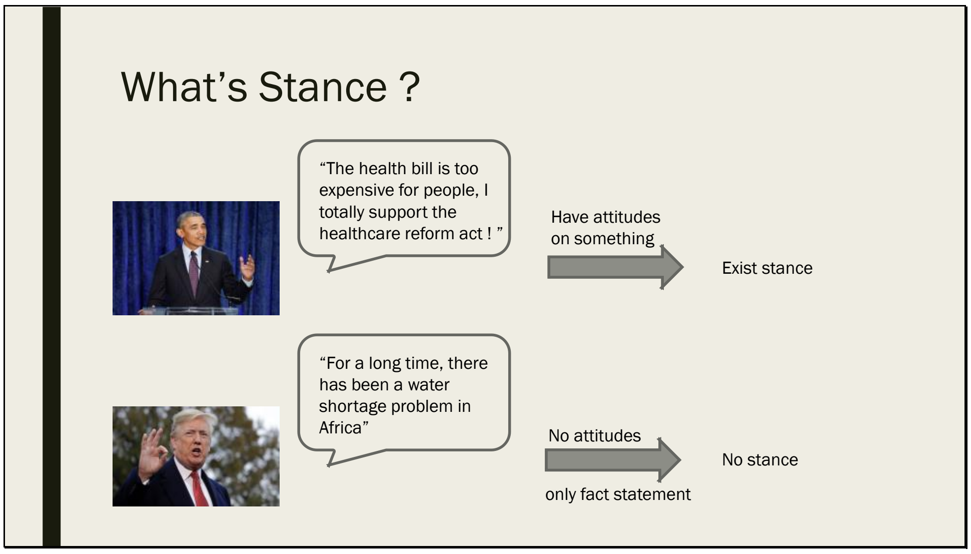
Good Afternoon, Professors.

Our capstone project is “Stance Detection and Stance Classification” at FiscalNote.

I am Xiaodan Chen.

I am Xiaochi Li.

Slide 2



What's Stance ?

"The health bill is too expensive for people, I totally support the healthcare reform act ! "

Have attitudes on something → Exist stance

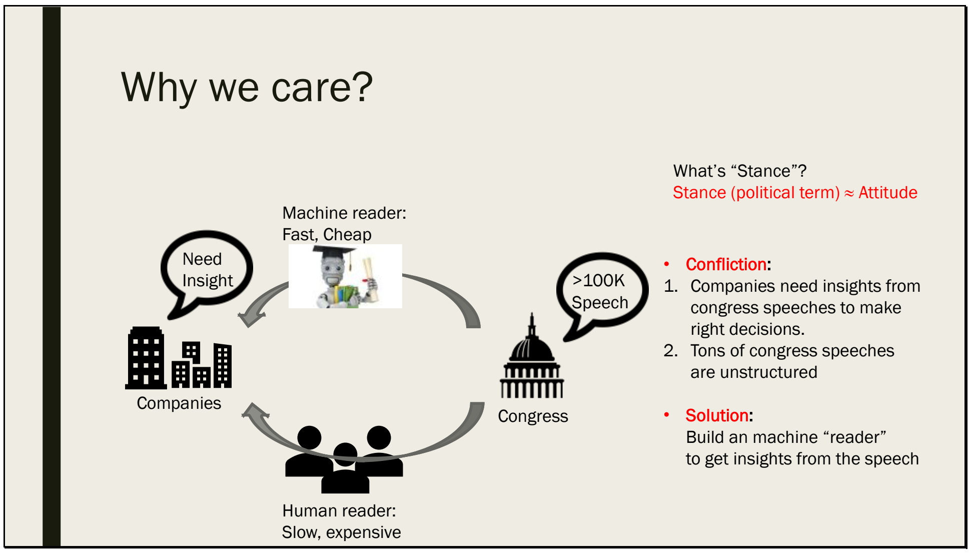
"For a long time, there has been a water shortage problem in Africa"

No attitudes → No stance

only fact statement

Firstly, what's stance?
stance is a political term; its meaning is similar to attitude.
If you have stance. That means, you have attitudes on something. You support it or you oppose it.
If you don't have stance, it means you are stating a fact, and you don't have attitude

Slide 3



Why we care?

Machine reader:
Fast, Cheap
Need Insight
Companies
>100K Speech
Congress
Human reader:
Slow, expensive

What's "Stance"?
Stance (political term) ≈ Attitude

- **Confliction:**
  1. Companies need insights from congress speeches to make right decisions.
  2. Tons of congress speeches are unstructured
- **Solution:**
  Build an machine "reader"
  to get insights from the speech

Why we care?

Because we found a confliction between the demand and supply of information.

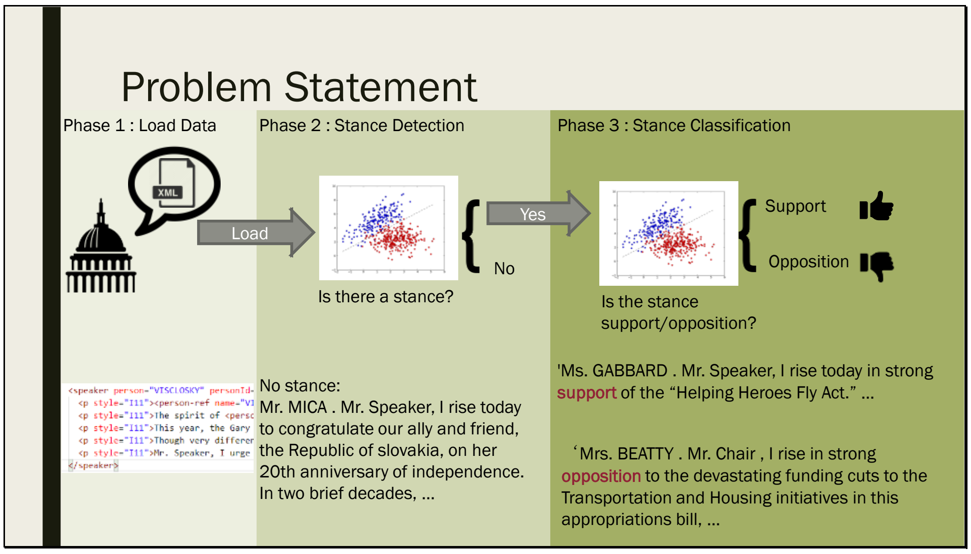
The companies need insights from congress speeches to make right decisions.

But tons of congress speeches are in unstructured format.

The companies can hire human reader to do so, but it's time-consuming and expensive.

So, we come up with a solution to build an machine reader to do that for us.

Slide 4



Problem Statement

Phase 1 : Load Data

Phase 2 : Stance Detection

Phase 3 : Stance Classification

```
<speaker person="VISCLOSKY" personId-
  <p style="I11"><person-ref name="VI
  <p style="I11">The spirit of <perso
  <p style="I11">This year, the Gary
  <p style="I11">Though very differer
  <p style="I11">Mr. Speaker, I urge
</speaker>
```

No stance:
Mr. MICA . Mr. Speaker, I rise today to congratulate our ally and friend, the Republic of slovakia, on her 20th anniversary of independence. In two brief decades, ...

'Ms. GABBARD . Mr. Speaker, I rise today in strong support of the "Helping Heroes Fly Act." ...

'Mrs. BEATTY . Mr. Chair , I rise in strong opposition to the devastating funding cuts to the Transportation and Housing initiatives in this appropriations bill, ...

Here is the problem statement.

The project is composed in three phases:

Phase one: load data, we extract speech from the XML file like this.

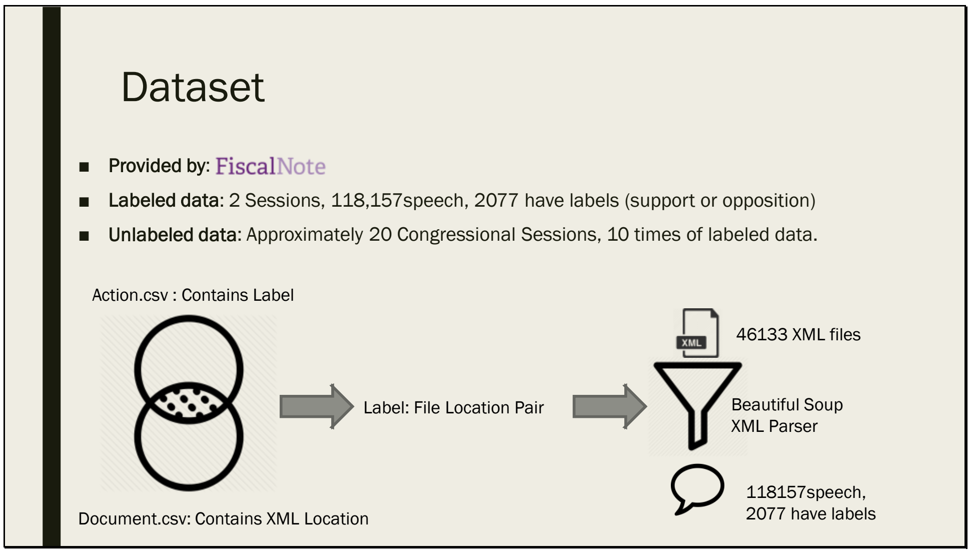
Phase two: Stance detection, we build a model to detect whether there is a stance in the speech.

And when there is stance, we send the speech to Phase three, stance classification.

We build a model to classify whether the stance is support or opposition.

Here is a sample speech contains no stance, this is a speech contain support stance, and one contain opposition stance.

Slide 5



Dataset

- **Provided by:** FiscalNote
- **Labeled data:** 2 Sessions, 118,157speech, 2077 have labels (support or opposition)
- **Unlabeled data:** Approximately 20 Congressional Sessions, 10 times of labeled data.

Action.csv : Contains Label
Label: File Location Pair
46133 XML files
Beautiful Soup XML Parser
118157speech, 2077 have labels
Document.csv: Contains XML Location

The dataset is provided by FiscalNote

The labeled data comes from 113th congress in 2013-2014.

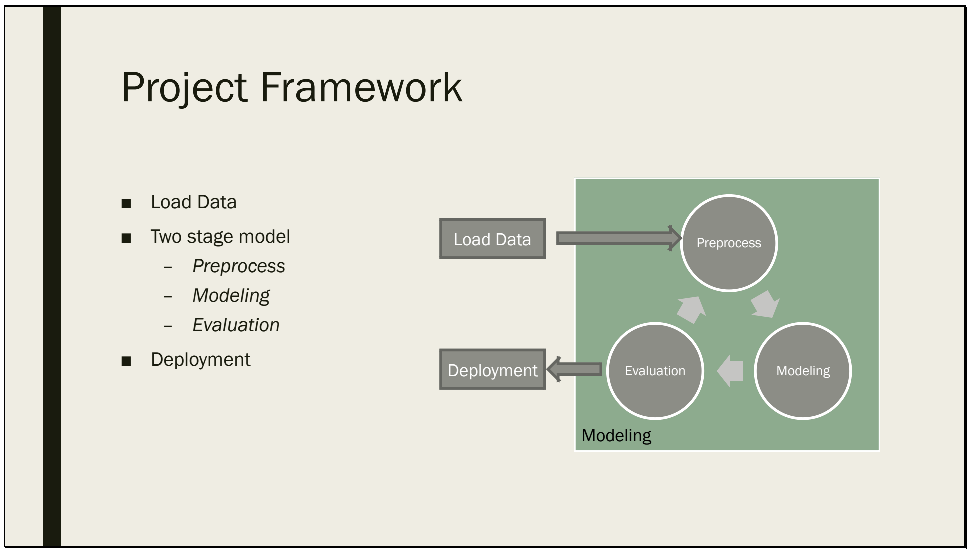
It contains one hundred and eighteen thousand speeches, and two thousand and seventy-seven of them have labels

And we have unlabeled data 10 times of labeled data.

We write a function to join the CSV file contains label and CSV file contains XML location and make label file location pair.

Then we use beautiful soup as XML parser and extract the speech we want.

Slide 6



Project Framework

- Load Data
- Two stage model
  - *Preprocess*
  - *Modeling*
  - *Evaluation*
- Deployment

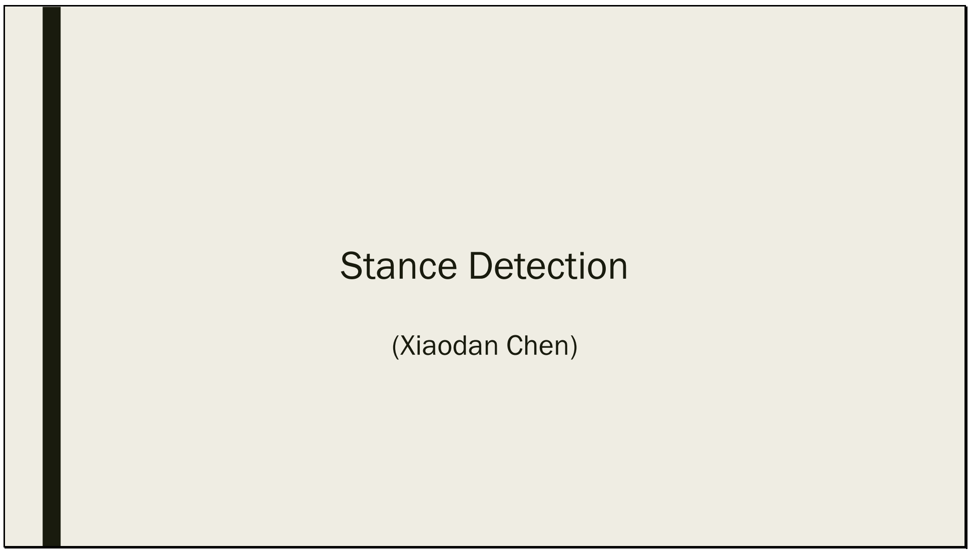
Load Data
Preprocess
Modeling
Evaluation
Deployment
Modeling

This is the project framework

Firstly, we load data.

Then we go through the preprocess-modeling-evaluation iteration until the model is good enough

Finally, we deploy the model into production environment

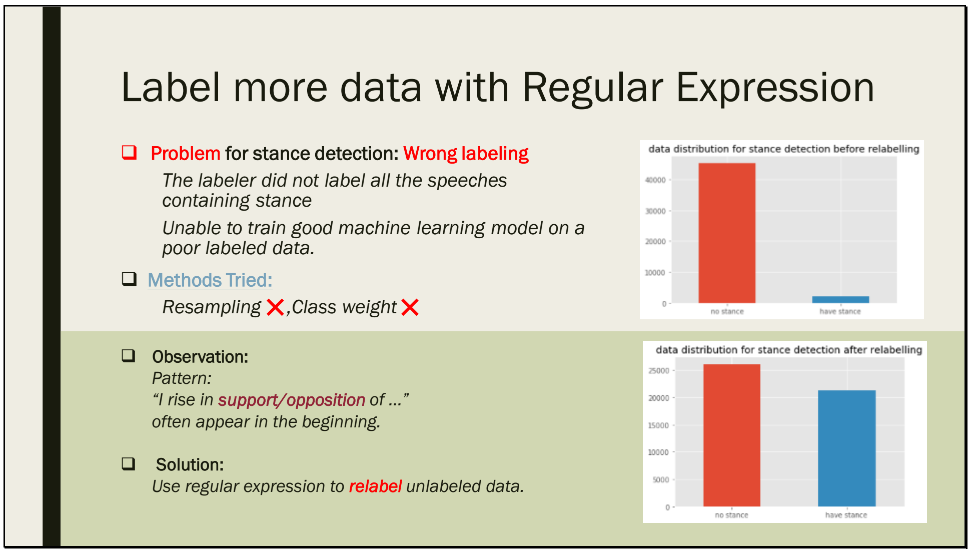


Stance Detection

(Xiaodan Chen)

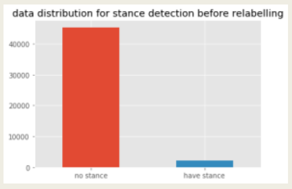
Chen: I will explain stance detection

Slide 8

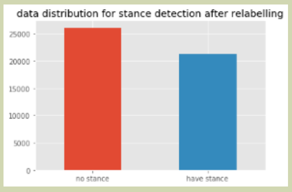


Label more data with Regular Expression

- ❑ Problem for stance detection: Wrong labeling
  - *The labeler did not label all the speeches containing stance*
  - *Unable to train good machine learning model on a poor labeled data.*
- ❑ Methods Tried:
  - *Resampling* ❌*,Class weight* ❌



- ❑ Observation:
  Pattern:
  "I rise in support/opposition of ..."
  often appear in the beginning.
- ❑ Solution:
  Use regular expression to relabel unlabeled data.



Firstly, we check the distribution of label, and we found that it is extremely imbalanced.

Most of the data are labeled as no stance. So we are unable to train good machine learning models on this data.

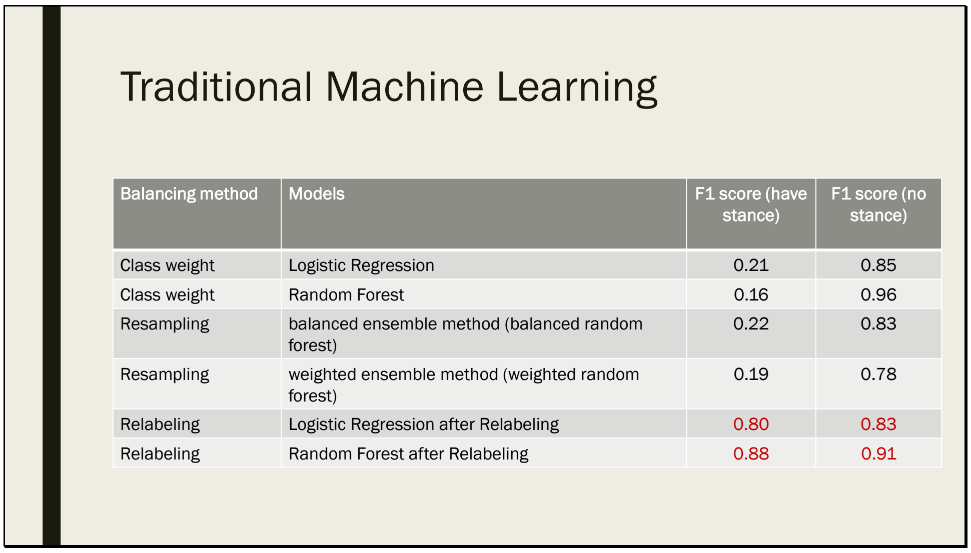
Usually, solutions to imbalance data involves resampling and changing the class

weight
We tried these methods, but the result is bad. We get pretty low F1 score for the minority class.

So, we go back and check our data again manually. A lot of speeches with strong indications of stances are not labeled.
With the observation comes our solution --- use regular expression to relabel the data.

We can see the distribution is more balanced after relabeling.

Slide 9



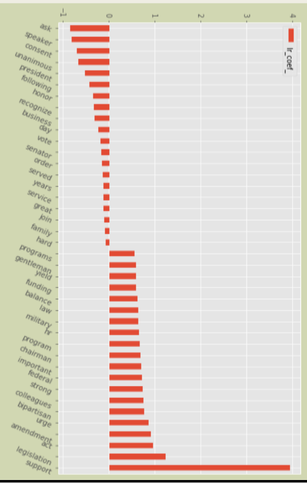
Traditional Machine Learning

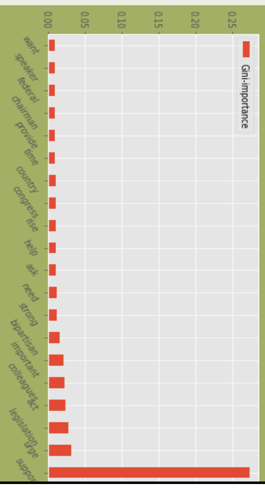
Balancing method	Models	F1 score (have stance)	F1 score (no stance)
Class weight	Logistic Regression	0.21	0.85
Class weight	Random Forest	0.16	0.96
Resampling	balanced ensemble method (balanced random forest)	0.22	0.83
Resampling	weighted ensemble method (weighted random forest)	0.19	0.78
Relabeling	Logistic Regression after Relabeling	0.80	0.83
Relabeling	Random Forest after Relabeling	0.88	0.91

this is the three methods we tried to balance the data, and we found the relabeling is most useful.

Slide 10

Logistic Regression & Random Forest



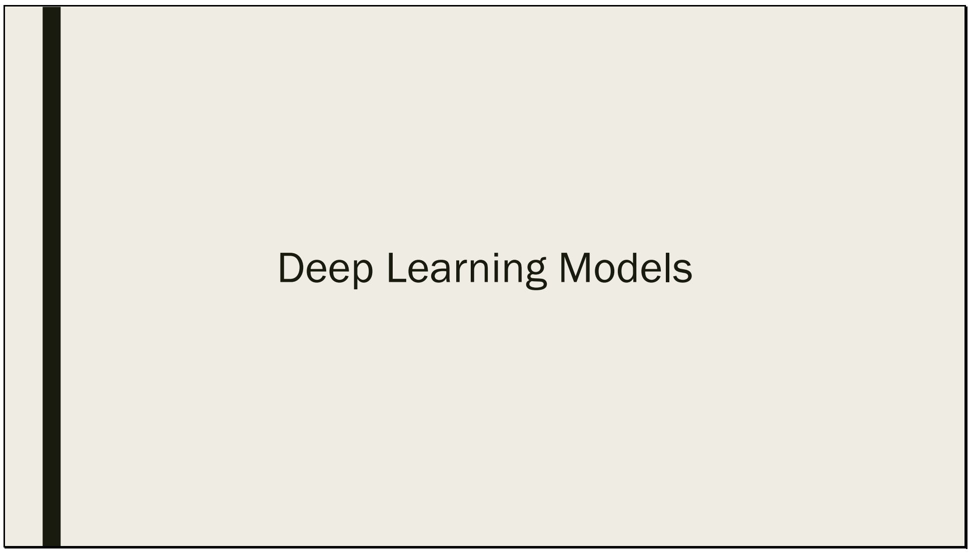


And we check the important features in logistic regression and random forest.

We found the features like ‘support’, ‘legislation’, ‘act’, ‘amendment’ are related to speeches containing stance.

And that’s reasonable

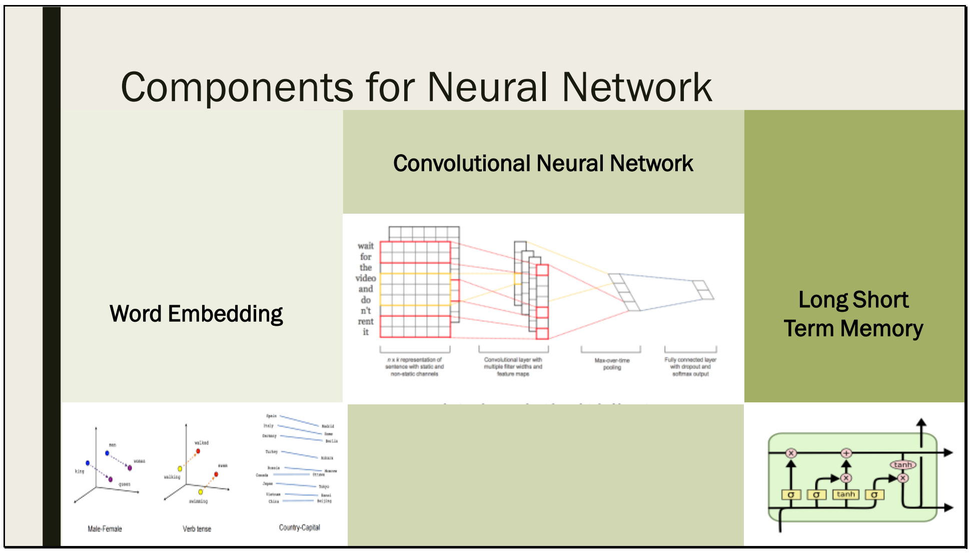
Slide 11



Deep Learning Models

Then we moved on to deep learning models

Slide 12



Components for Neural Network

Word Embedding

Convolutional Neural Network

Long Short Term Memory

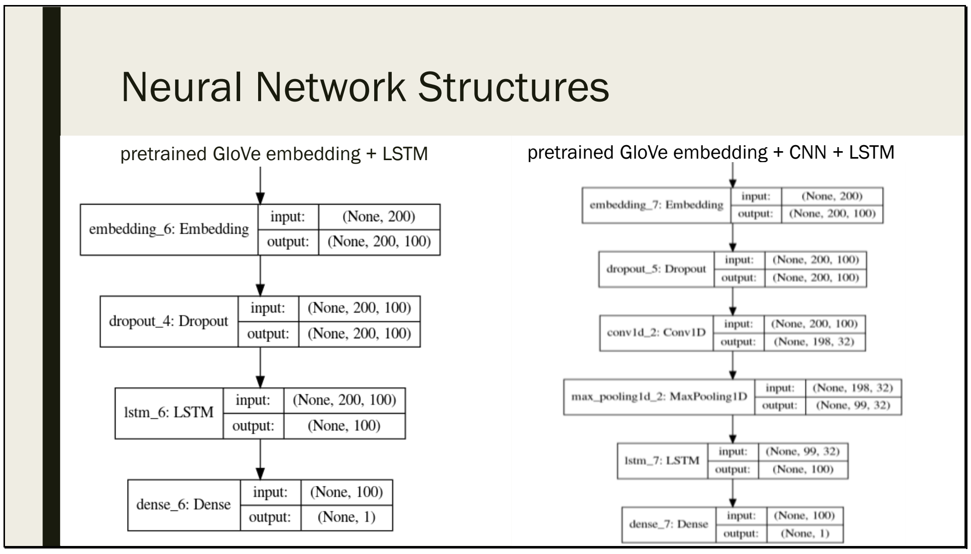
For the neural network, we have three components.

First one is word embedding layer, it is used to turn word into vector representation while keeps the semantic information.

Second one is Convolutional Neural Network, it is used to extract high level features. It is more like a n-gram method in traditional NLP.

Third one is Long Short-Term Memory, it is commonly used on sequential data like speech.

Slide 13



Neural Network Structures

pretrained GloVe embedding + LSTM
embedding_6: Embedding | input: (None, 200) | output: (None, 200, 100)
dropout_4: Dropout | input: (None, 200, 100) | output: (None, 200, 100)
lstm_6: LSTM | input: (None, 200, 100) | output: (None, 100)
dense_6: Dense | input: (None, 100) | output: (None, 1)
pretrained GloVe embedding + CNN + LSTM
embedding_7: Embedding | input: (None, 200) | output: (None, 200, 100)
dropout_5: Dropout | input: (None, 200, 100) | output: (None, 200, 100)
conv1d_2: Conv1D | input: (None, 200, 100) | output: (None, 198, 32)
max_pooling1d_2: MaxPooling1D | input: (None, 198, 32) | output: (None, 99, 32)
lstm_7: LSTM | input: (None, 99, 32) | output: (None, 100)
dense_7: Dense | input: (None, 100) | output: (None, 1)

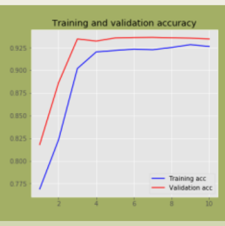
The first network structure contains a pretrained Glove embedding layer and a LSTM layer.

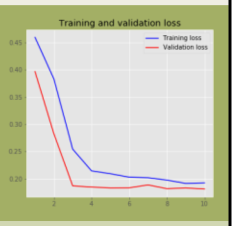
The second network structure contains an additional CNN layer between embedding layer and LSTM layer to extract high level features.

Slide 14

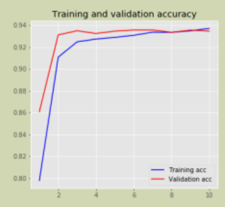
Deep Learning Result

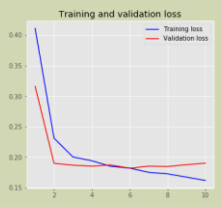
pretrained GloVe + LSTM





pretrained Glove + CNN + LSTM



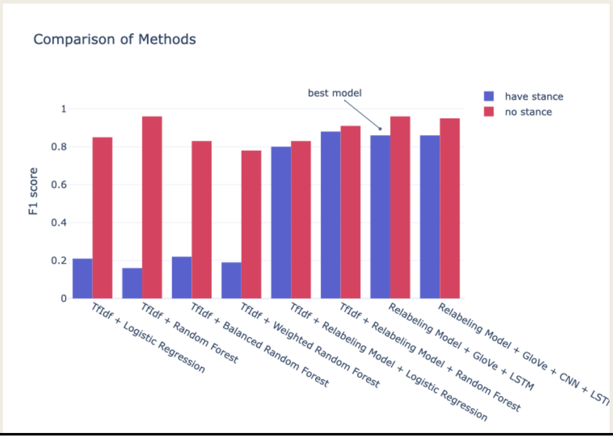


This is the validation accuracy and validation loss of both models.

The performance is pretty good, accuracy is around ninety three percent.

Slide 15

F1 score for Stance Detection Models



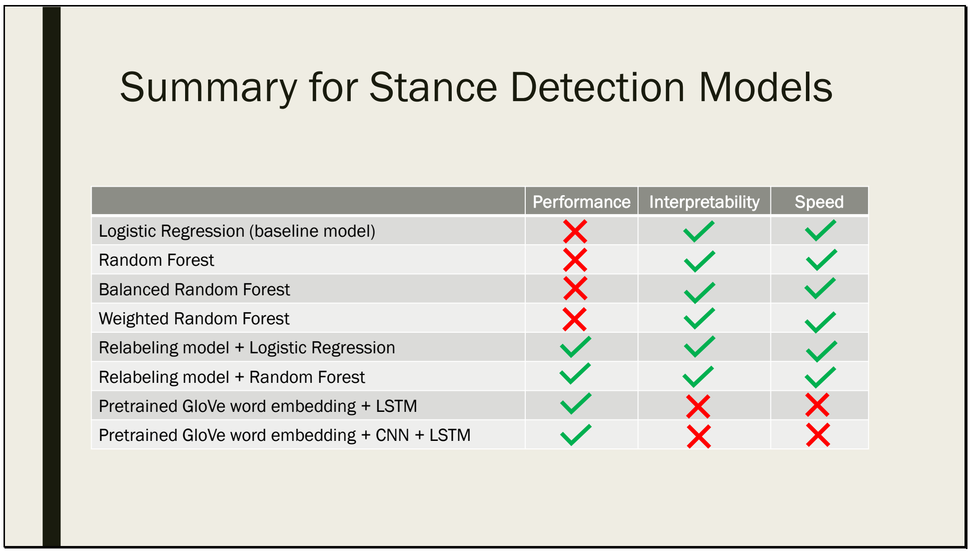
See Table

This is the F1 score for all the models.

We can see the best model should be this GloVe plus LSTM models.

It performs well for both majority and minority class.

Slide 16



Summary for Stance Detection Models

	Performance	Interpretability	Speed
Logistic Regression (baseline model)	✗	✓	✓
Random Forest	✗	✓	✓
Balanced Random Forest	✗	✓	✓
Weighted Random Forest	✗	✓	✓
Relabeling model + Logistic Regression	✓	✓	✓
Relabeling model + Random Forest	✓	✓	✓
Pretrained GloVe word embedding + LSTM	✓	✗	✗
Pretrained GloVe word embedding + CNN + LSTM	✓	✗	✗

To answer which model to choose.

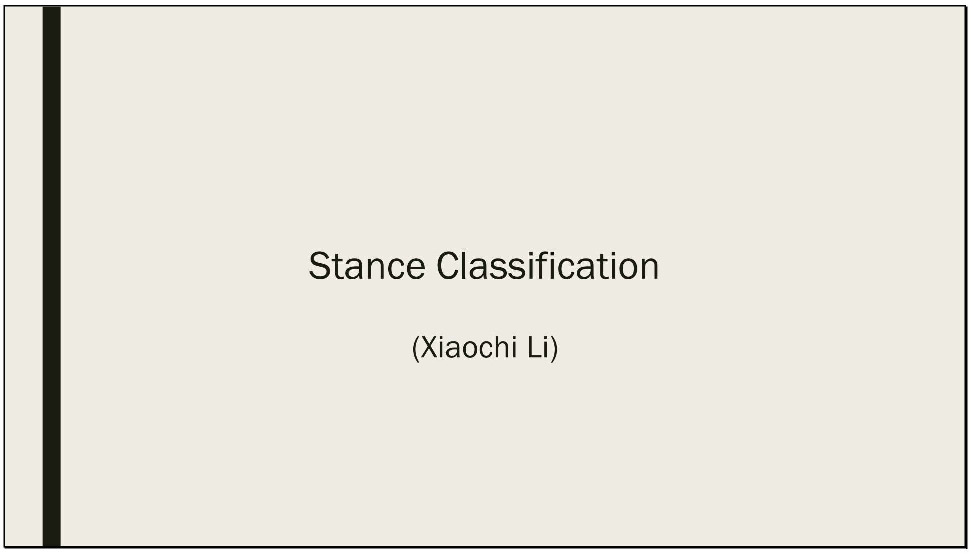
We thought it depends on the need.

If you want best accuracy and F1 score, you can use the deep learning models.

But they need hours to train and they are black box.

If you want a balance of training time, interpretability and performance, use

random forest is enough.

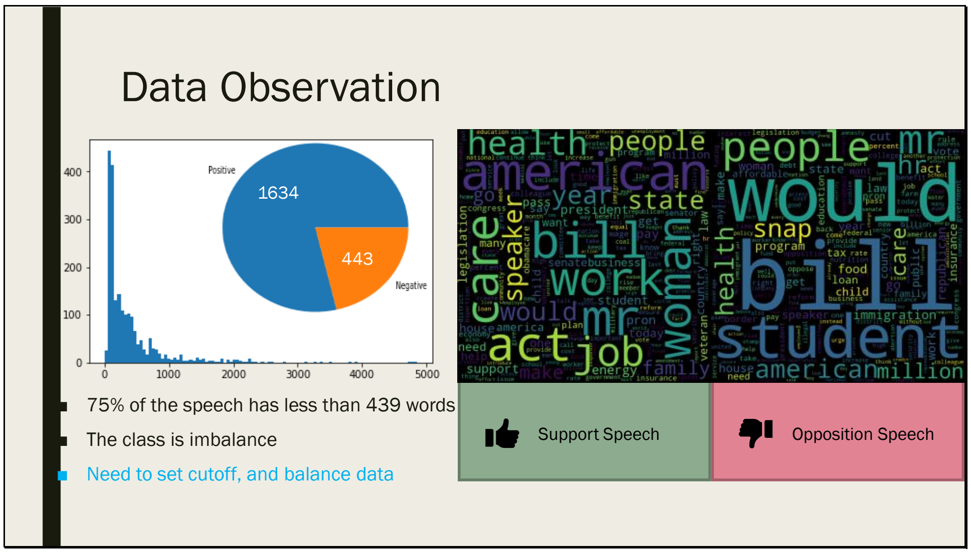


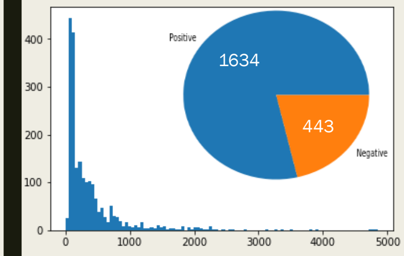
Stance Classification

(Xiaochi Li)

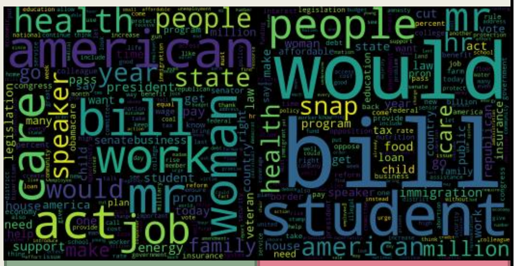
Li: I will explain stance classification from here

Slide 18


Data Observation



- 75% of the speech has less than 439 words
- The class is imbalance
- Need to set cutoff, and balance data



Support Speech

Opposition Speech

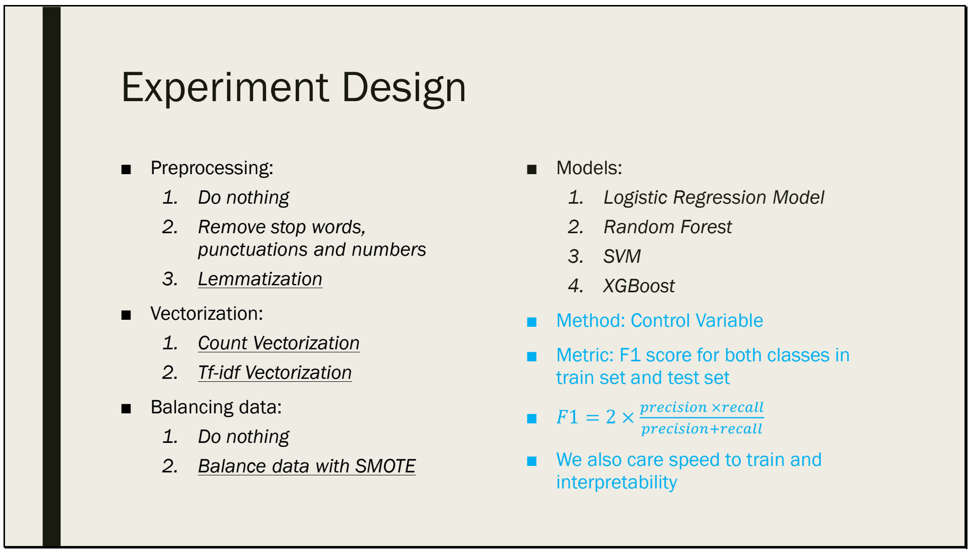
Firstly, is the data observation.

We drew the word cloud for support and opposition speech and did not found something special.

We found that 75% of the speech has less than 429 words and less than one quarter of the samples are negative.

It indicated that we need to set a cutoff for the extreme long speech and balance the data

Slide 19



Experiment Design

- Preprocessing:
  1. *Do nothing*
  2. *Remove stop words, punctuations and numbers*
  3. *Lemmatization*
- Vectorization:
  1. *Count Vectorization*
  2. *Tf-idf Vectorization*
- Balancing data:
  1. *Do nothing*
  2. *Balance data with SMOTE*
- Models:
  1. *Logistic Regression Model*
  2. *Random Forest*
  3. *SVM*
  4. *XGBoost*
- Method: Control Variable
- Metric: F1 score for both classes in train set and test set
- $F1 = 2 \times \frac{precision \times recall}{precision + recall}$
- We also care speed to train and interpretability

This is the experiment design for the stance classification task.

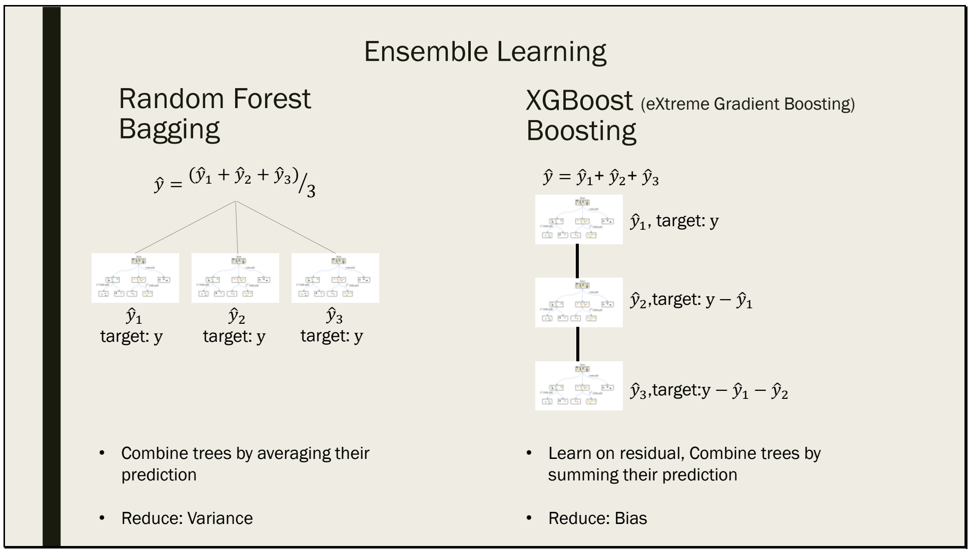
The experiment method is Control Variable, the metric is F1 score for both classes in train set and test set.

We will test all the combination of three preprocessing options, two vectorization methods, two balancing data options with four models.

So, for each model, we will test 12 different combinations.

We also care the speed to train and the interpretability of the model.

Slide 20



Ensemble Learning

Random Forest
Bagging

$\hat{y} = {}^{(\hat{y}_1 + \hat{y}_2 + \hat{y}_3)}/_{3}$

$\hat{y}_1$ target: y

$\hat{y}_2$ target: y

$\hat{y}_3$ target: y

- Combine trees by averaging their prediction
- Reduce: Variance

XGBoost (eXtreme Gradient Boosting)
Boosting

$\hat{y} = \hat{y}_1 + \hat{y}_2 + \hat{y}_3$

$\hat{y}_1$, target: y

$\hat{y}_2$,target: $y - \hat{y}_1$

$\hat{y}_3$,target:$y - \hat{y}_1 - \hat{y}_2$

- Learn on residual, Combine trees by summing their prediction
- Reduce: Bias

I would like to briefly introduce XGBoost since we have not learnt it in class.

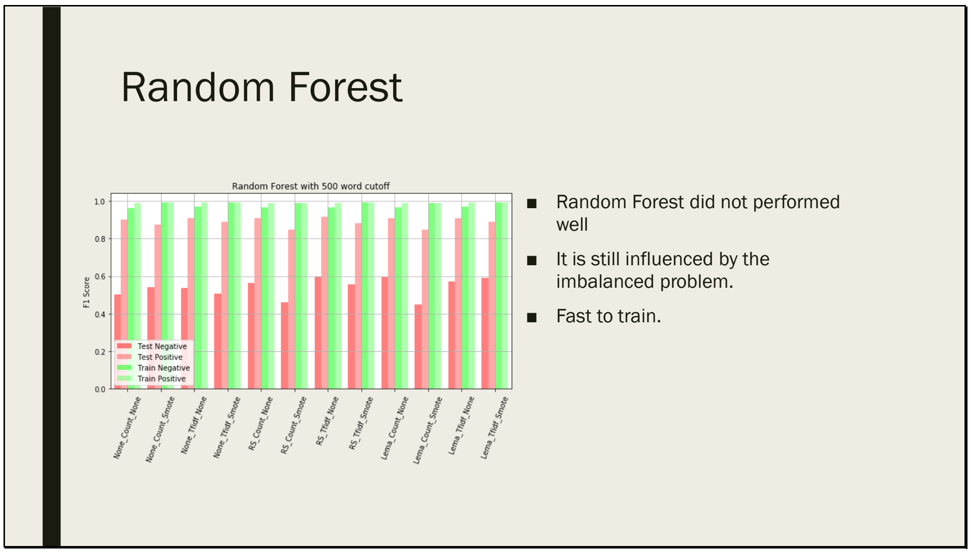
There are two main ensemble learning strategies, one is bagging, and another is boosting.

Bagging methods like Random forest combine trees parallelly and it is good at reducing the variance

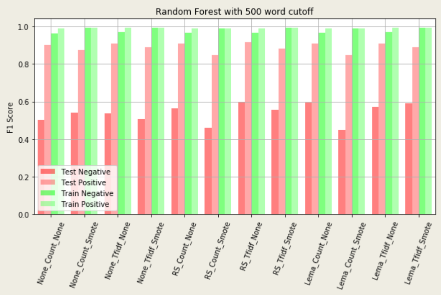
Boosting methods like Xgboost combine trees sequentially and each tree learns on the residual of the previous tree.

So, it is good at reducing bias.

Slide 21



Random Forest

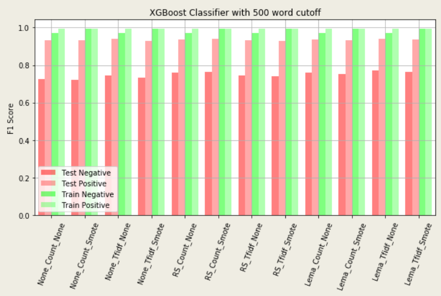


- Random Forest did not performed well
- It is still influenced by the imbalanced problem.
- Fast to train.

The first model we tested is Random Forest, we found that it did not performed well on the test set, still influenced the imbalanced problem.

Slide 22

XGBoost Classifier

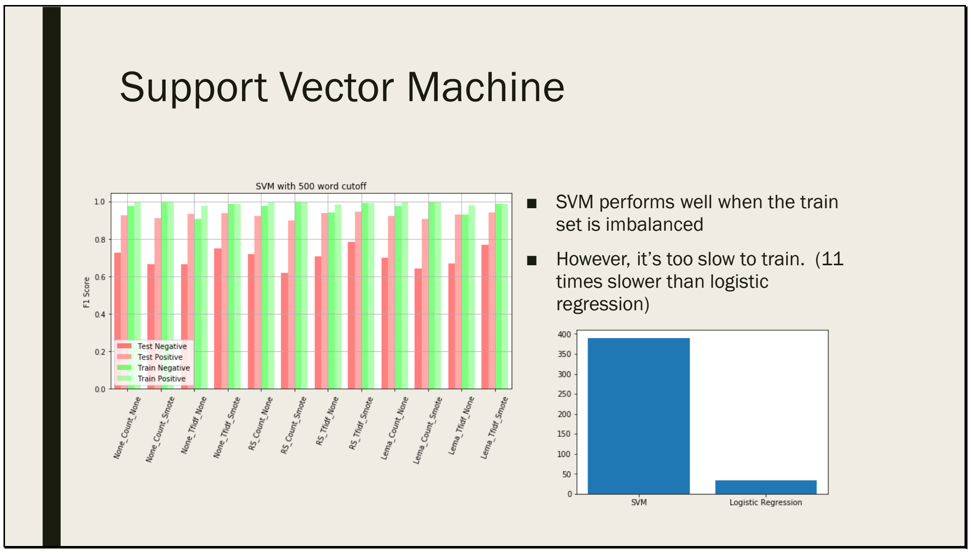


- XGBoost performs better compared to Random Forest.
- XGBoost is more focused on reducing bias, while Random Forest reduce variance in general
- Fast to train

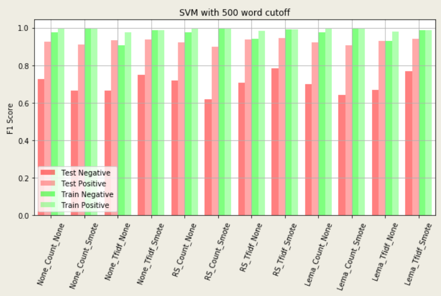
The second one is XGBoost, it performance better than Random Forest.

And we thought it is because XGBoost is more focused on reducing bias.

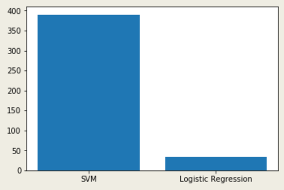
Slide 23



Support Vector Machine



- SVM performs well when the train set is imbalanced
- However, it's too slow to train. (11 times slower than logistic regression)

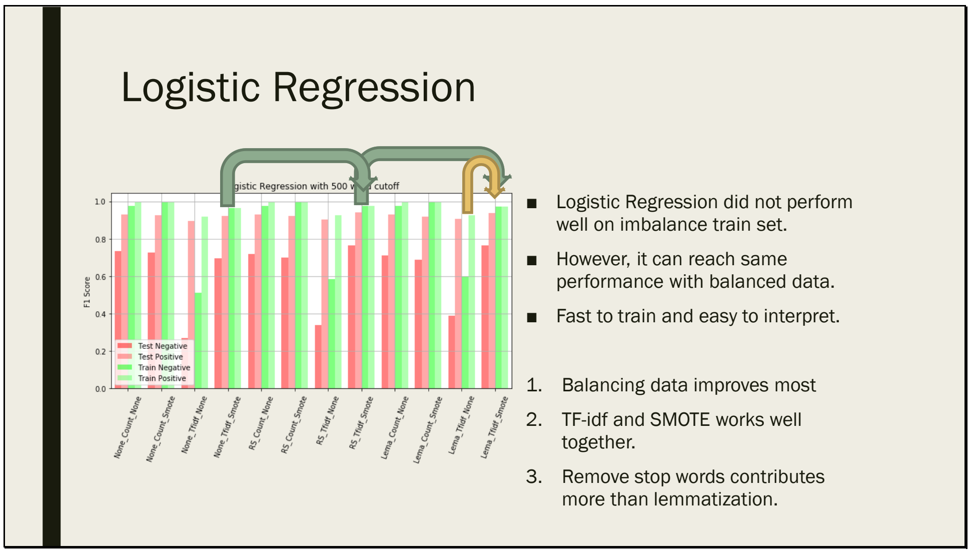


Then we tested Support Vector Machine, and we found it performs well when

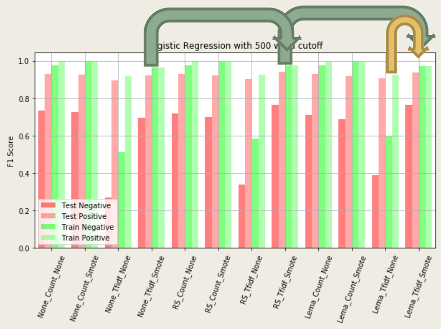
the training set is imbalanced.

However, it is too slow to train. It took 11 more times to train than other models

Slide 24



Logistic Regression



- Logistic Regression did not perform well on imbalance train set.
- However, it can reach same performance with balanced data.
- Fast to train and easy to interpret.

1. Balancing data improves most
2. TF-idf and SMOTE works well together.
3. Remove stop words contributes more than lemmatization.

Finally, we tested logistic regression.

We found that although logistic regression did not perform well when the train set is imbalanced.

But it can reach same performance with balanced data.

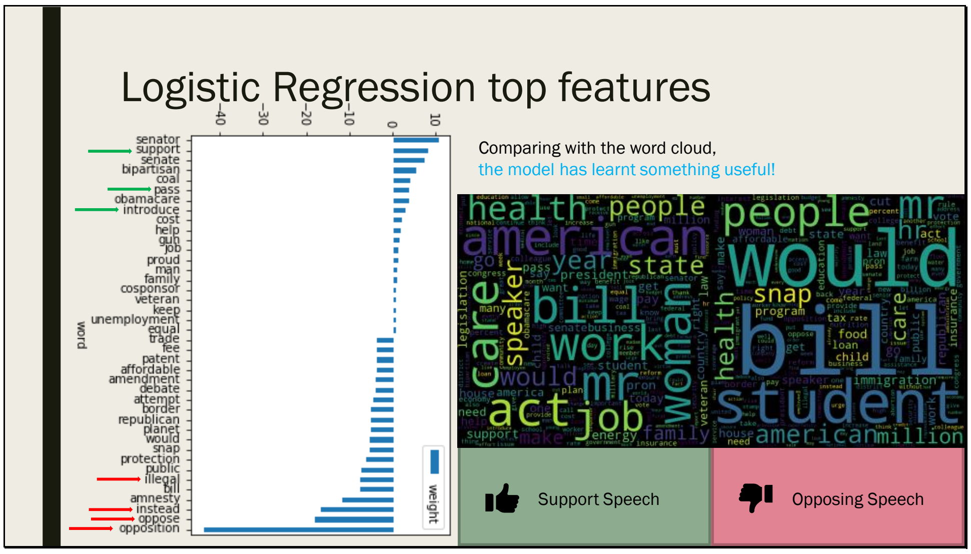
Another good thing is that it is fast to train and easy to interpret.

Balancing data improves the performance most

As for the vectorization and preprocessing, the finding is that tf-idf and SMOTE works well together.

And remove stop words contributes more than lemmatization.

Slide 25



Logistic Regression top features

Comparing with the word cloud,
the model has learnt something useful!

Support Speech

Opposing Speech

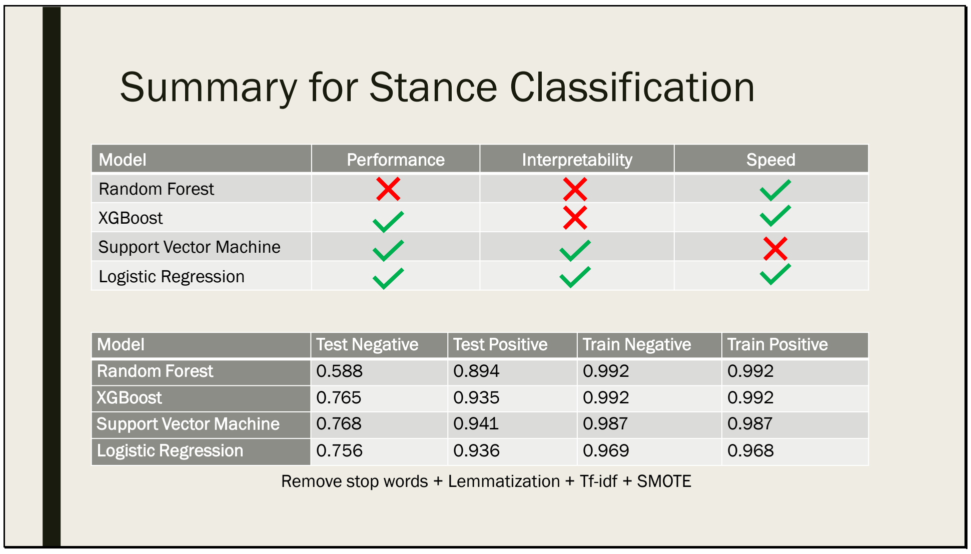
Then we check the top features for the logistic regression.

We can find some words like support, pass, introduce with positive weight

And words like opposition, oppose, instead and illegal with negative weight.

Comparing with the word cloud we saw before, we can see the model has learnt something useful.

Slide 26



Summary for Stance Classification

Model	Performance	Interpretability	Speed
Random Forest	✗	✗	✓
XGBoost	✓	✗	✓
Support Vector Machine	✓	✓	✗
Logistic Regression	✓	✓	✓

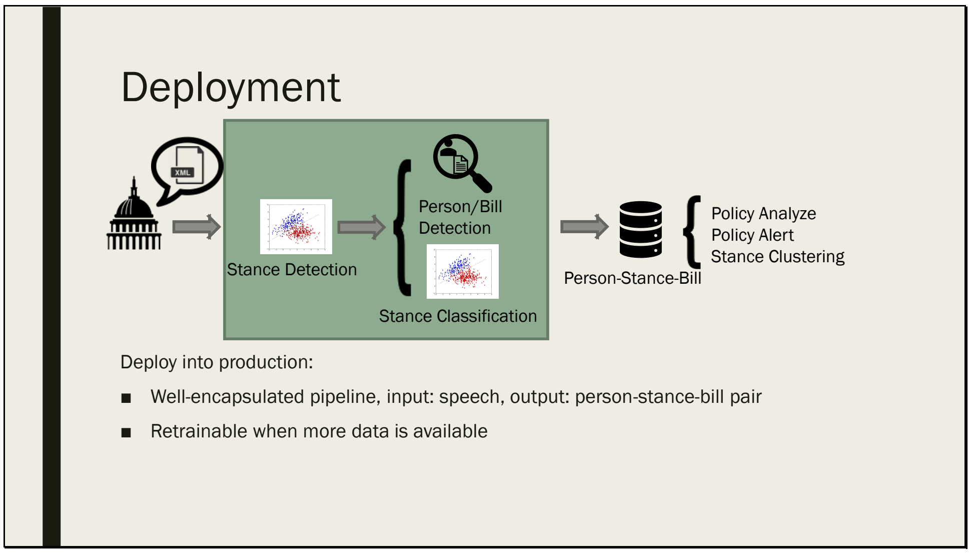
Model	Test Negative	Test Positive	Train Negative	Train Positive
Random Forest	0.588	0.894	0.992	0.992
XGBoost	0.765	0.935	0.992	0.992
Support Vector Machine	0.768	0.941	0.987	0.987
Logistic Regression	0.756	0.936	0.969	0.968

Remove stop words + Lemmatization + Tf-idf + SMOTE

After taking performance, interpretability and speed into consideration.

We chose Logistic Regression as our best model for stance classification.

Slide 27



Deployment

Deploy into production:

- Well-encapsulated pipeline, input: speech, output: person-stance-bill pair
- Retrainable when more data is available

After we have the models, we will move on to deployment.

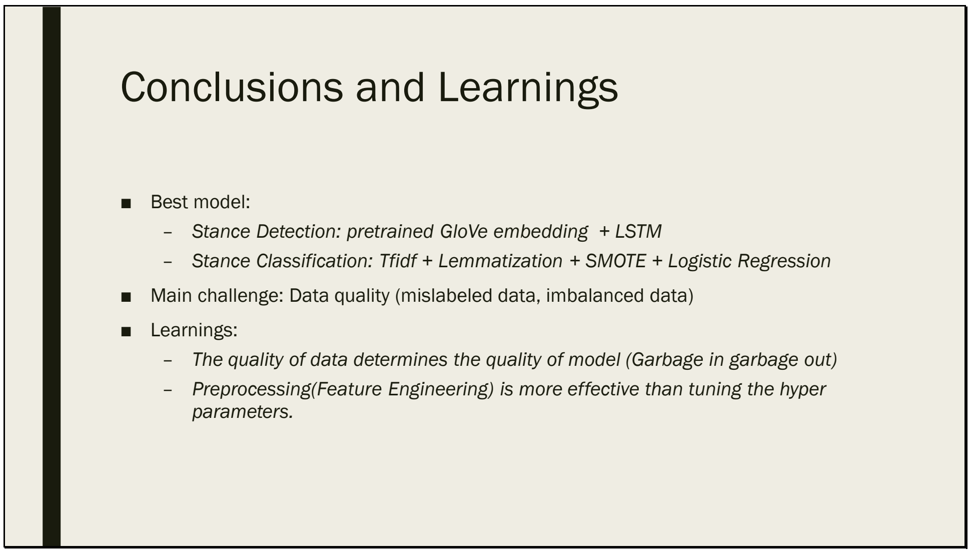
These two models will be a part of a system that takes XML format record as

input

and generate person-stance-bill pairs for further use like policy analyze, policy

alert and stance clustering.

Slide 28



Conclusions and Learnings

- Best model:
  - *Stance Detection: pretrained GloVe embedding + LSTM*
  - *Stance Classification: Tfidf + Lemmatization + SMOTE + Logistic Regression*
- Main challenge: Data quality (mislabeled data, imbalanced data)
- Learnings:
  - *The quality of data determines the quality of model (Garbage in garbage out)*
  - *Preprocessing(Feature Engineering) is more effective than tuning the hyper parameters.*

The best model we have for stance detection is pretrained GloVe embedding + LSTM

The best model we have for stance classification is Tfidf + Lemmatization +SMOTE with Logistic Regression

The main challenge we have is the data quality issue, mislabeled data for stance detection and imbalanced data for stance classification.

The learning is that the quality of data determines the quality of the model

And Preprocessing(Feature Engineering) is more effective than tune the hyper parameters.

Slide 29

Thank you

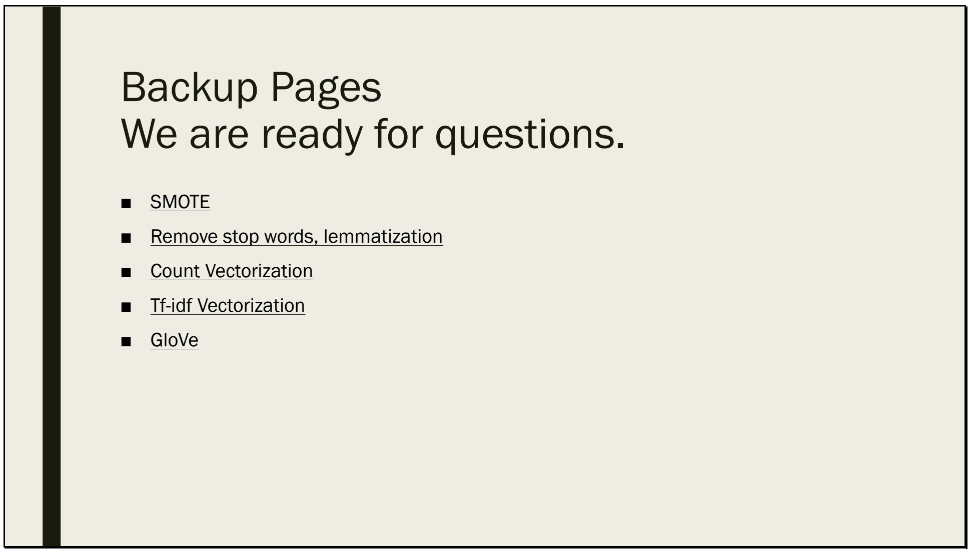
- Dr. Brian Wright and Dr. Vladimir A. Eidelman for offering this opportunity.
- Dr. Daniel Argyle for mentoring us during this project.
- Dr. Nima Zahadat for supporting us
- All the professors in the Data Science Program.

FiscalNote
FiscalNote

Finally, we would like to thank Brian and Vlad for offering this great opportunity.

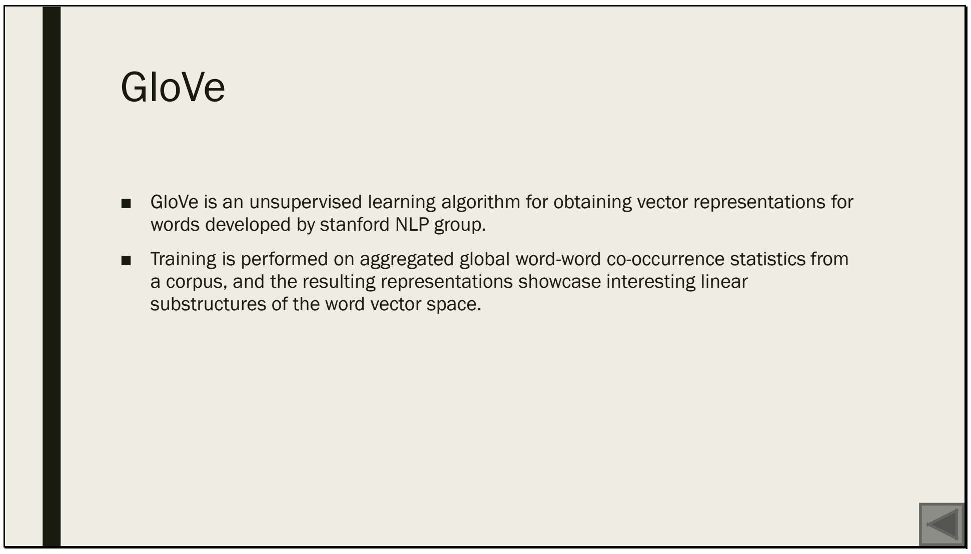
Also Daniel for mentoring us during the project.

And Nima and all the professors in the data science program



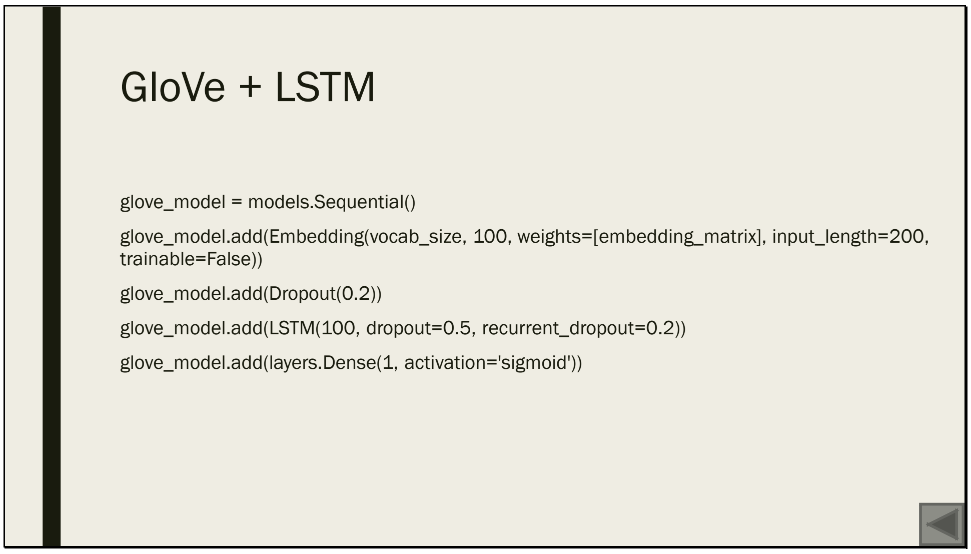
Backup Pages
We are ready for questions.

- SMOTE
- Remove stop words, lemmatization
- Count Vectorization
- Tf-idf Vectorization
- GloVe



GloVe

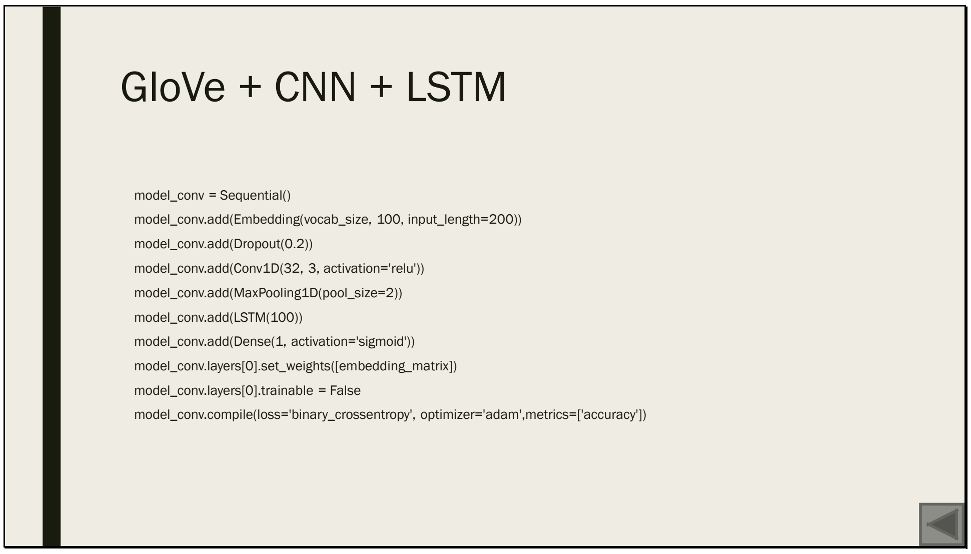
- GloVe is an unsupervised learning algorithm for obtaining vector representations for words developed by stanford NLP group.
- Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.



GloVe + LSTM

```
glove_model = models.Sequential()
glove_model.add(Embedding(vocab_size, 100, weights=[embedding_matrix], input_length=200,
trainable=False))
glove_model.add(Dropout(0.2))
glove_model.add(LSTM(100, dropout=0.5, recurrent_dropout=0.2))
glove_model.add(layers.Dense(1, activation='sigmoid'))
```

Slide 33



GloVe + CNN + LSTM

```
model_conv = Sequential()
model_conv.add(Embedding(vocab_size, 100, input_length=200))
model_conv.add(Dropout(0.2))
model_conv.add(Conv1D(32, 3, activation='relu'))
model_conv.add(MaxPooling1D(pool_size=2))
model_conv.add(LSTM(100))
model_conv.add(Dense(1, activation='sigmoid'))
model_conv.layers[0].set_weights([embedding_matrix])
model_conv.layers[0].trainable = False
model_conv.compile(loss='binary_crossentropy', optimizer='adam',metrics=['accuracy'])
```

Backup
What’s SMOTE?

- <u>S</u>ynthetic <u>M</u>inority <u>O</u>ver-sampling <u>Te</u>chnique
- Generate new samples for minority class between the existing samples

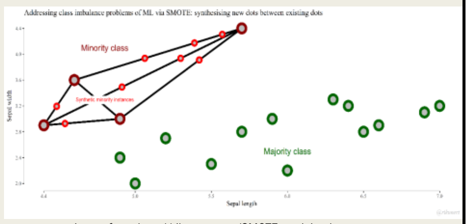
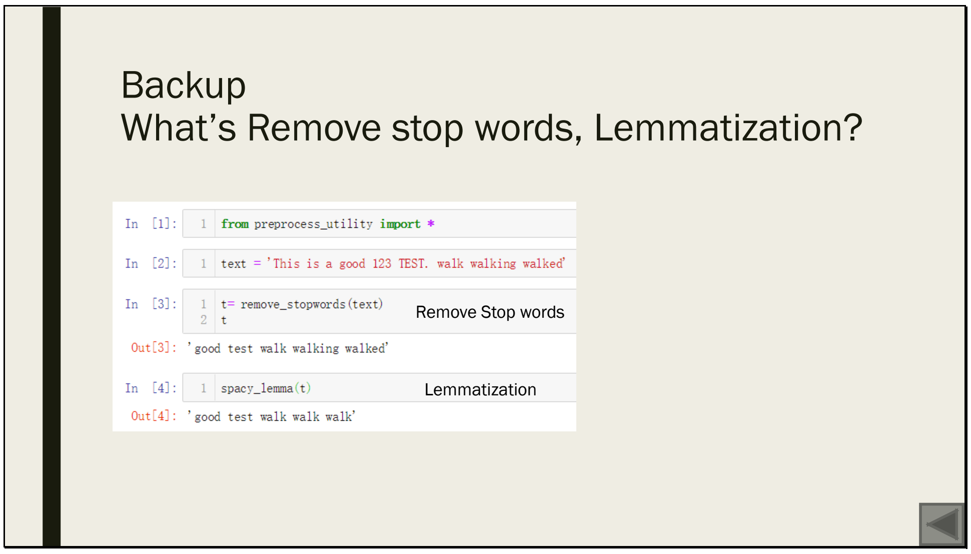


Image from: http://rikunert.com/SMOTE_explained

Slide 35



Backup
What's Remove stop words, Lemmatization?

```
In [1]:  from preprocess_utility import *

In [2]:  text = 'This is a good 123 TEST. walk walking walked'

In [3]:  t= remove_stopwords(text)
         t
```

Remove Stop words

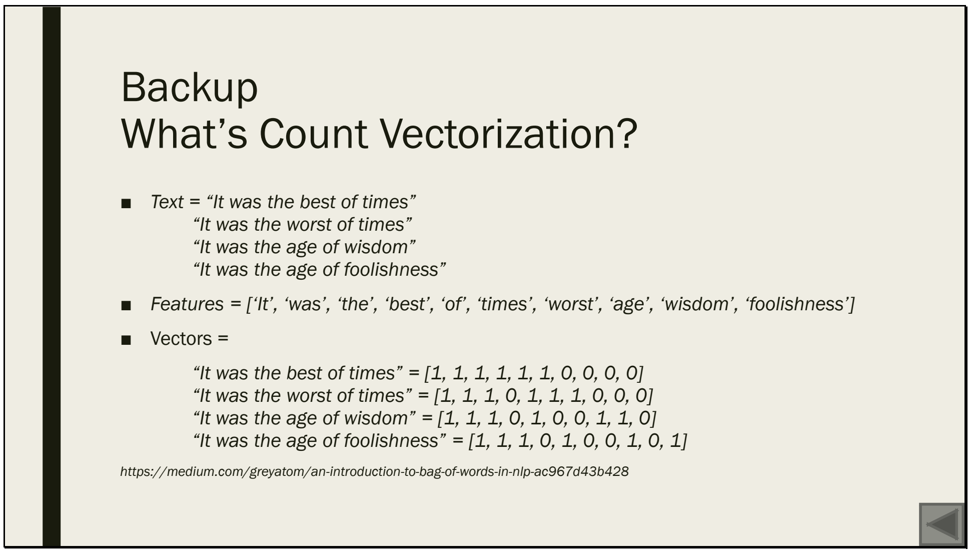
```
Out[3]: 'good test walk walking walked'

In [4]:  spacy_lemma(t)
```

Lemmatization

```
Out[4]: 'good test walk walk walk'
```

Slide 36



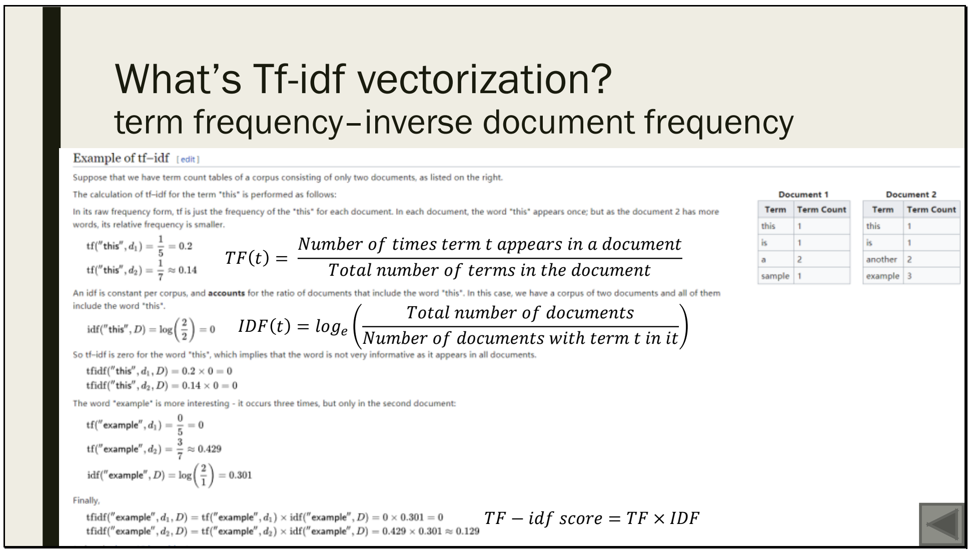
Backup
What's Count Vectorization?

- *Text = "It was the best of times"*
  "It was the worst of times"
  "It was the age of wisdom"
  "It was the age of foolishness"
- *Features = ['It', 'was', 'the', 'best', 'of', 'times', 'worst', 'age', 'wisdom', 'foolishness']*
- Vectors =

  "It was the best of times" = [1, 1, 1, 1, 1, 1, 0, 0, 0, 0]
  "It was the worst of times" = [1, 1, 1, 0, 1, 1, 1, 0, 0, 0]
  "It was the age of wisdom" = [1, 1, 1, 0, 1, 0, 0, 1, 1, 0]
  "It was the age of foolishness" = [1, 1, 1, 0, 1, 0, 0, 1, 0, 1]

https://medium.com/greyatom/an-introduction-to-bag-of-words-in-nlp-ac967d43b428

Slide 37



What’s Tf-idf vectorization?

term frequency–inverse document frequency

Example of tf–idf [edit]

Suppose that we have term count tables of a corpus consisting of only two documents, as listed on the right.

The calculation of tf–idf for the term "this" is performed as follows:

In its raw frequency form, tf is just the frequency of the "this" for each document. In each document, the word "this" appears once; but as the document 2 has more words, its relative frequency is smaller.

$$\mathrm{tf}(''\text{this}'', d_1) = \frac{1}{5} = 0.2$$
$$\mathrm{tf}(''\text{this}'', d_2) = \frac{1}{7} \approx 0.14$$

$$TF(t) = \frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document}$$

An idf is constant per corpus, and **accounts** for the ratio of documents that include the word "this". In this case, we have a corpus of two documents and all of them include the word "this".

$$\mathrm{idf}(''\text{this}'', D) = \log\left(\frac{2}{2}\right) = 0$$

$$IDF(t) = log_e\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ term\ t\ in\ it}\right)$$

So tf–idf is zero for the word "this", which implies that the word is not very informative as it appears in all documents.

$$\mathrm{tfidf}(''\text{this}'', d_1, D) = 0.2 \times 0 = 0$$
$$\mathrm{tfidf}(''\text{this}'', d_2, D) = 0.14 \times 0 = 0$$

The word "example" is more interesting - it occurs three times, but only in the second document:

$$\mathrm{tf}(''\text{example}'', d_1) = \frac{0}{5} = 0$$
$$\mathrm{tf}(''\text{example}'', d_2) = \frac{3}{7} \approx 0.429$$
$$\mathrm{idf}(''\text{example}'', D) = \log\left(\frac{2}{1}\right) = 0.301$$

Finally,

$$\mathrm{tfidf}(''\text{example}'', d_1, D) = \mathrm{tf}(''\text{example}'', d_1) \times \mathrm{idf}(''\text{example}'', D) = 0 \times 0.301 = 0$$
$$\mathrm{tfidf}(''\text{example}'', d_2, D) = \mathrm{tf}(''\text{example}'', d_2) \times \mathrm{idf}(''\text{example}'', D) = 0.429 \times 0.301 \approx 0.129$$

$$TF - idf\ score = TF \times IDF$$

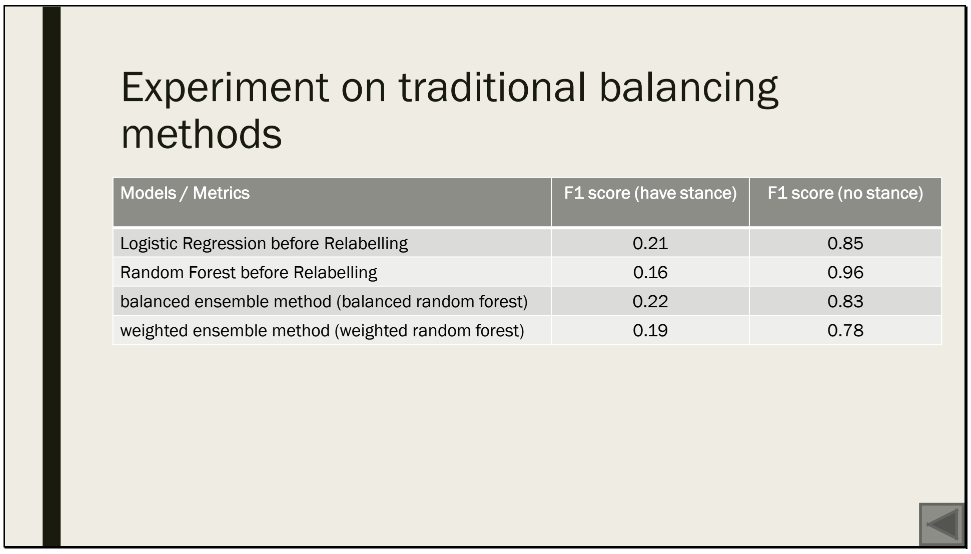
Document 1

Term	Term Count
this	1
is	1
a	2
sample	1

Document 2

Term	Term Count
this	1
is	1
another	2
example	3

Slide 38

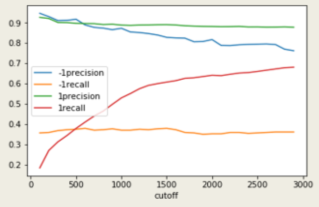


Experiment on traditional balancing methods

Models / Metrics	F1 score (have stance)	F1 score (no stance)
Logistic Regression before Relabelling	0.21	0.85
Random Forest before Relabelling	0.16	0.96
balanced ensemble method (balanced random forest)	0.22	0.83
weighted ensemble method (weighted random forest)	0.19	0.78

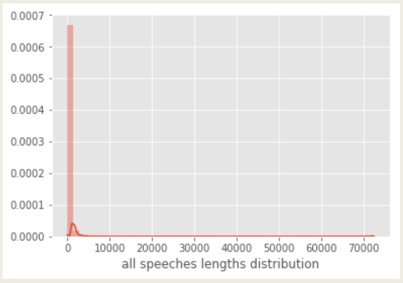
Slide 39

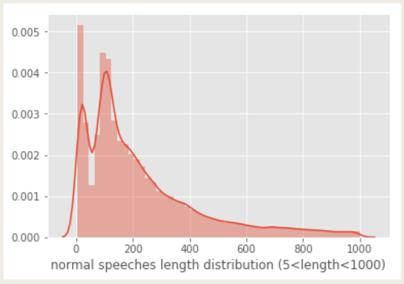
Logistic Regression Relabeling Accuracy



- We can get a good precision(~0.9) with a strict keyword detection range (500 characters from begin)
- Trade “quantity” for “quality”.

Speech Length Observation



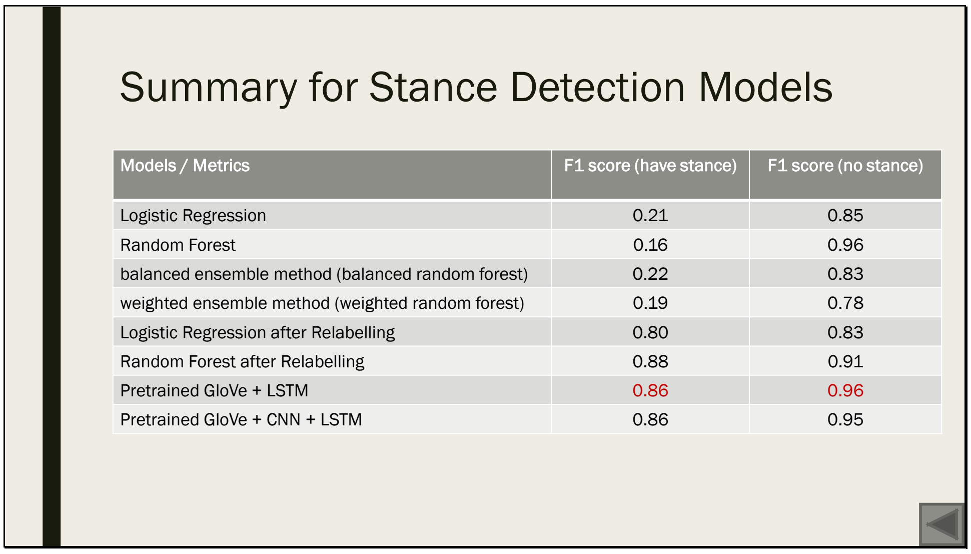


	Speech Length
Max	79382
Min	2
Avg	346

Examples of speeches (length< 5):
"Mr Heller yield floor"

Examples of speeches (length< 15):
"Had I been here, I would vote yes"

Slide 41



Summary for Stance Detection Models

Models / Metrics	F1 score (have stance)	F1 score (no stance)
Logistic Regression	0.21	0.85
Random Forest	0.16	0.96
balanced ensemble method (balanced random forest)	0.22	0.83
weighted ensemble method (weighted random forest)	0.19	0.78
Logistic Regression after Relabelling	0.80	0.83
Random Forest after Relabelling	0.88	0.91
Pretrained GloVe + LSTM	0.86	0.96
Pretrained GloVe + CNN + LSTM	0.86	0.95