NLP Maps Meeting
05/23

Andrew Simon, Meenal Rawlani, Shijie Zhang

Roadmap

- Summary of Model Performance
- Discussion of strategy advanced meta selection algorithm
- Final objectives moving forward

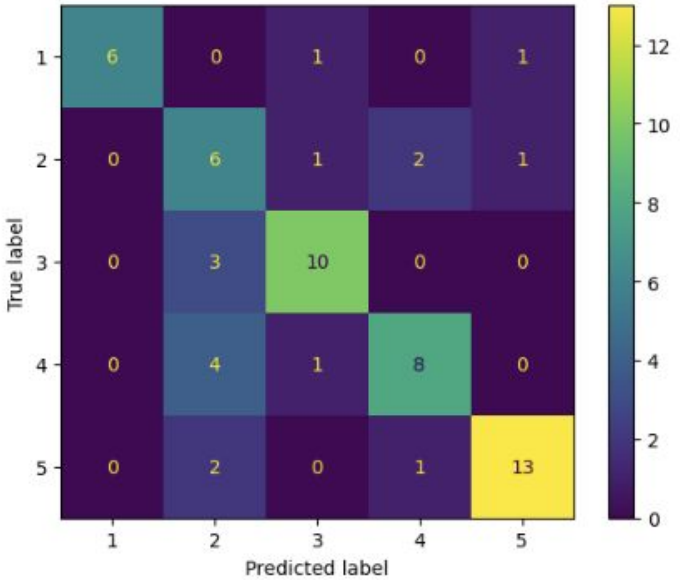
Word2Vec (Summary)

- Sampled evenly across each sheet, concatenated into a single pandas dataframe
- Used both pre-trained and untrained models
  - Untrained model performed significantly worse, with a 0.55 accuracy score compared to the pre-trained score of 0.72
  - Will use exclusively pre-trained models moving forward
- Implemented Tensorflow, DecisiontreeClassifier, RandomForest, LogisticRegression, and SVM classifiers
  - RandomForest consistently performed best out of each model tested
  - Unable to perform meaningful classification with Tensorflow

Word2Vec (Results)

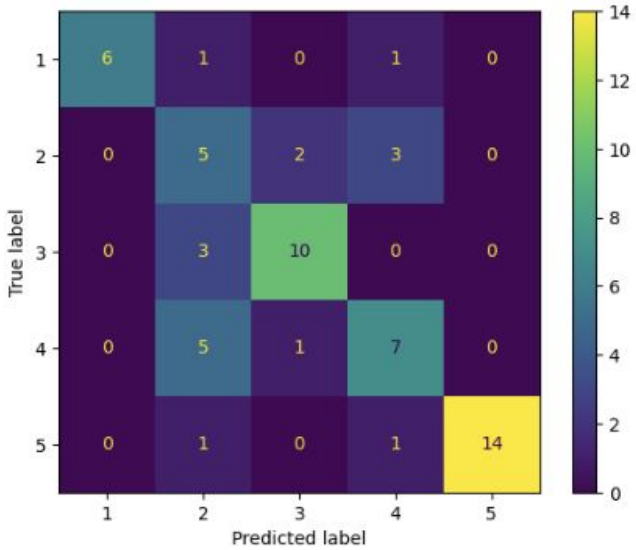
Classifier	Accuracy
Tensorflow	0.17
DecisionTree	0.42
RF	0.72
LR	0.70
SVM	0.67

Confusion Matrix of RF

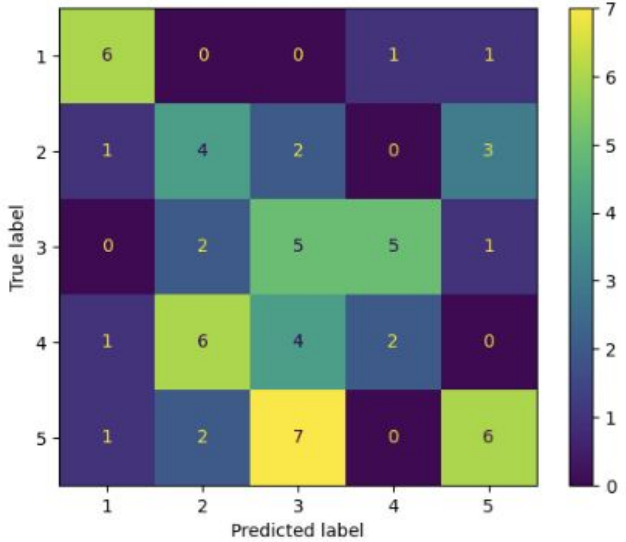


Word2Vec (Results cont.)

Logistic Regression

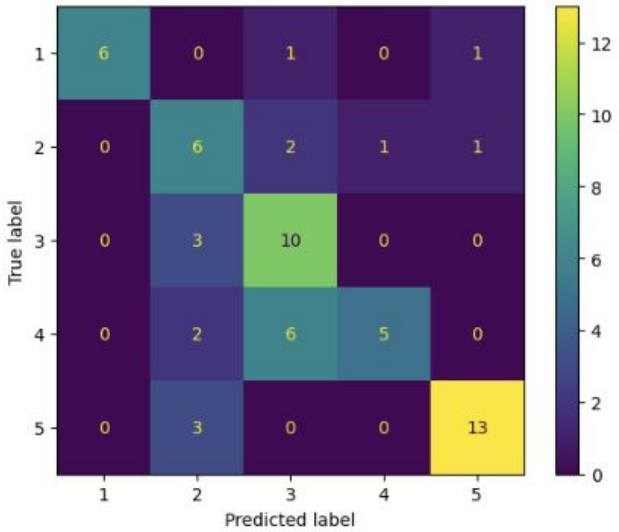


Decision Tree



Word2Vec (Results cont.)

SVM



Word2Vec (Summary)

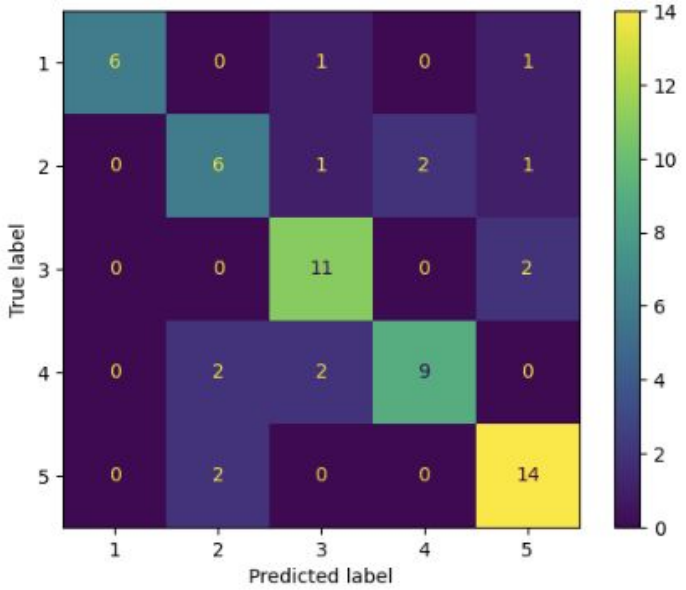
- Near misses are classified with no false positives across all reports
- Model performs exceptionally well classifying level 5 incidents, for certain random states there were no false negatives
- Fair performance in classifying level 2-4 incidents
- Model struggles most with classifying level 2 incidents

ELMo (Summary)

- Embeddings are an order of magnitude more computationally expensive than any other method tested
- Highest accuracy across all models tested
- Almost identical results for level 1 and 5 incidents when compared to W2v, however much better at discerning level 2-4 incidents
- SVM only classifies data a level 1,3, or 5

ELMo (Results

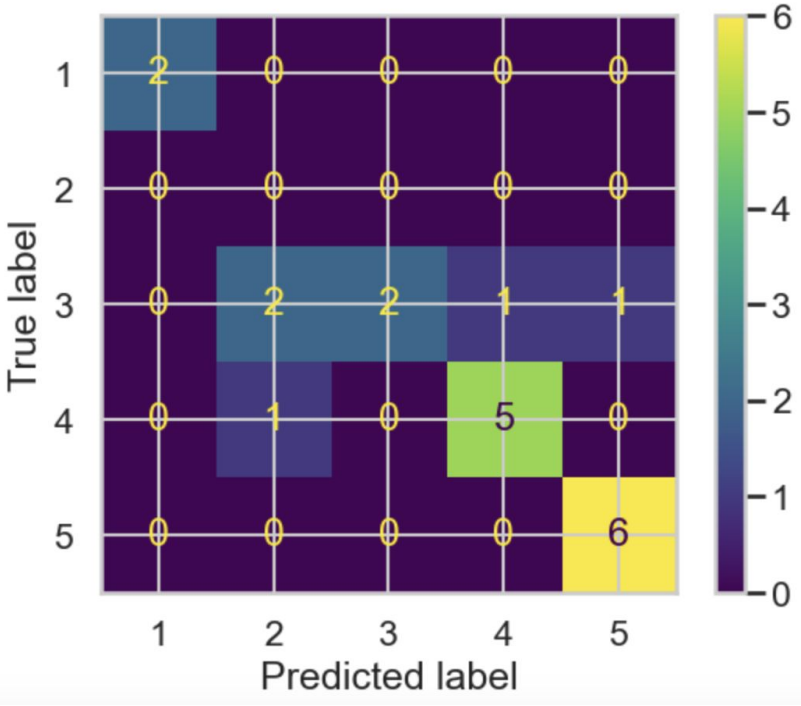
Classifier	Accuracy
DecisionTree	0.58
RF	0.77
LR	0.68
SVM	0.45

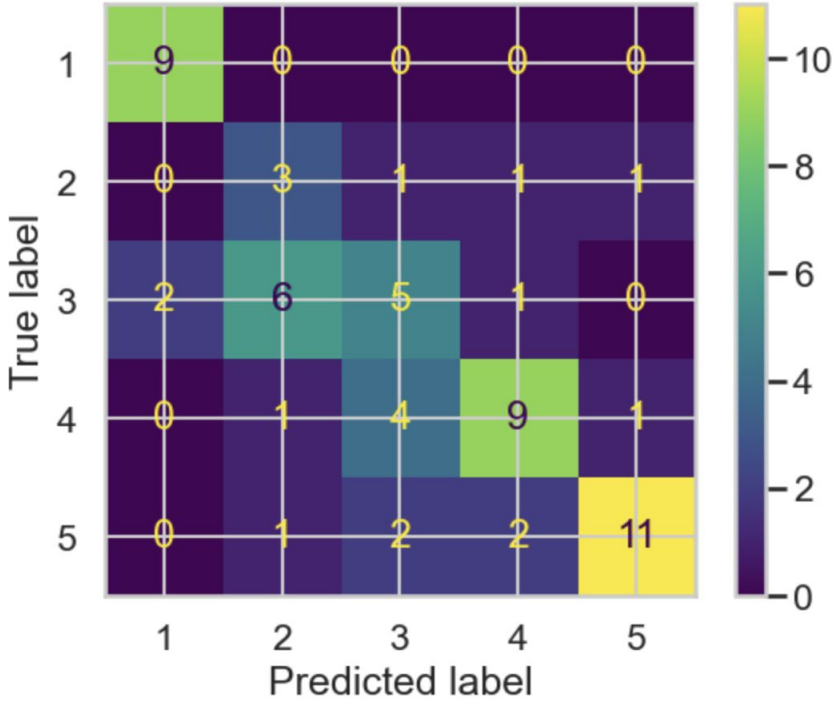


GloVe (Summary)

- Sampled evenly across each sheet, concatenated into a single pandas dataframe
- Used pre-trained vectors embedded from Wikipedia from official website
- Accuray score could be as high as 0.75, goes down to 0.6 when using all three sheets from the dataset
- Among all classifiers, RandomForest performs the best
- Further fine tuning will be done in this week

GloVe (Results)

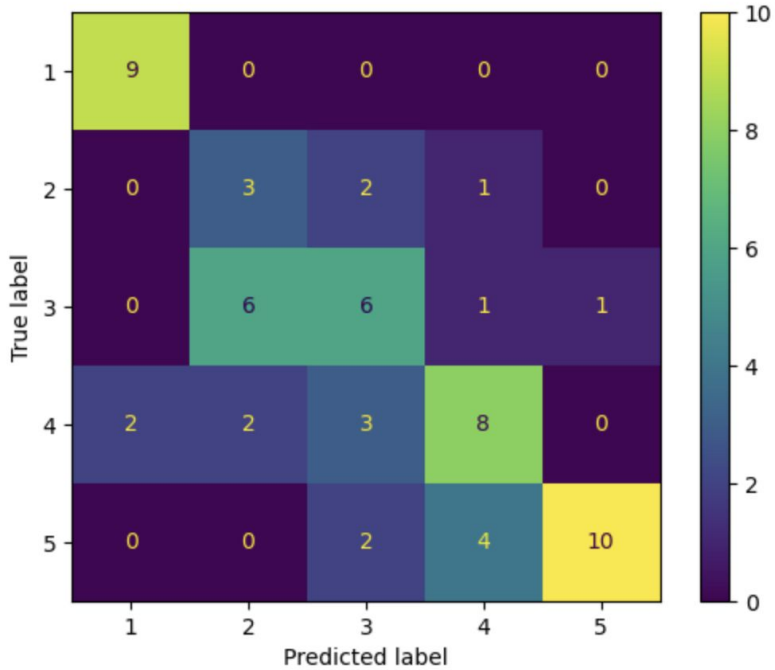




FastText (Summary)

- Sampled evenly across each sheet, concatenated into a single pandas dataframe
- The model has basically no ability to predict if using PSE dataset to train, only captured ~700 words
- Use pre-trained vectors from Common Crawl and get better results of 0.6
- RandomForest still performs best for now

FastText (Results)



- Accuray is around 0.6
- More fine tuning and trial of different classifiers will be done in this week

TF-IDF (Summary)

- The TF-IDF model used an Excel dataset, preprocessing it by combining multiple sheets into one DataFrame.
- It employed the TF-IDF vectorizer to tokenize the text and generate numerical representations.
- The Random Forest Classifier was chosen as the primary model, with hyperparameters tuned via grid search.
- Additional models like SVM and Naive Bayes were also evaluated. To enhance accuracy, a Voting Classifier combined predictions from all models.

TF-IDF (Result)

Random Forest Model accuracy: 0.63

SVM Model accuracy: 0.57

Naive Bayes Model accuracy: 0.57

Voting Classifier accuracy: 0.55

TF-IDF (Result)

Classification reports were generated to provide detailed performance metrics for each model. These reports included precision, recall, F1-score, and support for each class level. The Random Forest model consistently outperformed the other models across different metrics, demonstrating its superior performance.

In summary, the TF-IDF model, with the Random Forest Classifier as the primary model, showed promising results in classifying incident reports based on severity levels.

The model exhibited strong performance, particularly in identifying level 5 incidents, while achieving fair accuracy for level 2 to 4 incidents.

TF-IDF (Further Fine-Tuning)

Further exploration and optimization of the model could focus on improving the classification of level 2 incidents, addressing class imbalances, and considering additional feature engineering techniques.

Bag-of-Words

The Bag-of-Words model utilized a dataset obtained from an Excel file, where multiple sheets were concatenated into a single DataFrame for preprocessing. The text data was tokenized using the CountVectorizer, creating numerical representations of the reports.

The dataset was divided into training and testing sets using an 80:20 ratio. A Random Forest Classifier was trained on the training data, employing 200 estimators. The model achieved an accuracy of 73.33% on the test set.

Bag-of-Words

	precision	recall	f1-score
1	1.0000	0.8571	0.9231
2	0.5333	0.8000	0.6400
3	0.7000	0.4667	0.5600
4	0.6429	0.6429	0.6429
5	0.9333	1.0000	0.9655
accuracy			0.7333
macro avg	0.7619	0.7533	0.7463
weighted avg	0.7483	0.7333	0.7296

The Random Forest model performed particularly well in classifying level 1 and level 5 incidents, while achieving lower scores for level 2 and level 4 incidents.

Bert

The data from each sheet was split into training and testing sets using a 80:20 ratio.

The BERT tokenizer was utilized to tokenize the text data, encoding sentences with special tokens and generating input IDs and attention masks. The BERT model was then loaded and used to extract features from the tokenized data. The features were converted to numpy arrays for further processing.

A Random Forest Classifier was trained on the extracted features.

Bert

	precision	recall	f1-score
1	0.5000	0.5000	0.5000
2	0.0000	0.0000	0.0000
3	1.0000	0.3333	0.5000
4	0.7500	0.5000	0.6000
5	1.0000	0.8333	0.9091
accuracy			0.5500
macro avg	0.6500	0.4333	0.5018
weighted avg	0.8750	0.5500	0.6527

The model achieved an accuracy of 55% on the test set.

The model did a very good job in classifying level 5 incidents.

Discussion of Advanced Selection Algorithm

- BoW and TF-IDF are able to discern Level 1 and 5 incidents with high accuracy, and there is little gained from using contextual models
- Initially classify data using BoW, separate classifications by Level
- Pass levels 2-4 to a contextual embedding algorithm, merge results
- Accuracy should remain consistent, expect to see a ~40% decrease in computational expense using method

Final Objectives for last two weeks

- Fine tune models for optimized performance
- Determine if there are noticeable semantic differences causing false negatives for level 5 incidents
- Map Results across each of the three sheets as well as ensemble
- Build advanced selection algorithm