Fake News

Uri Meir & Alon Cohen

(Boys in The Hood)

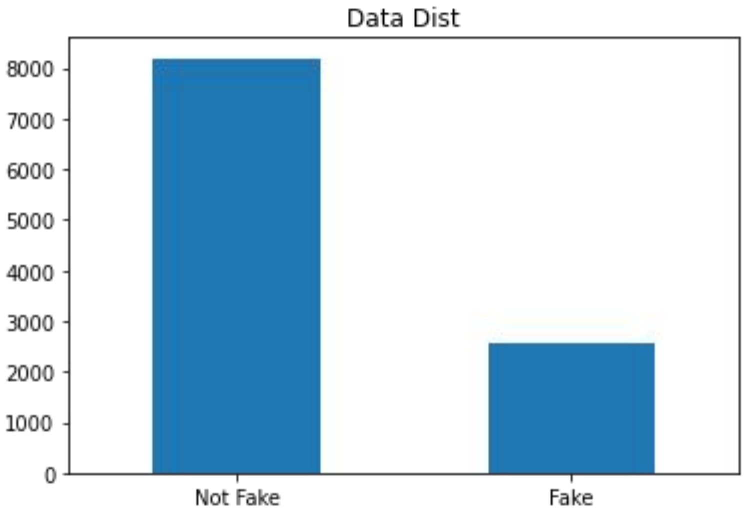
January 2021

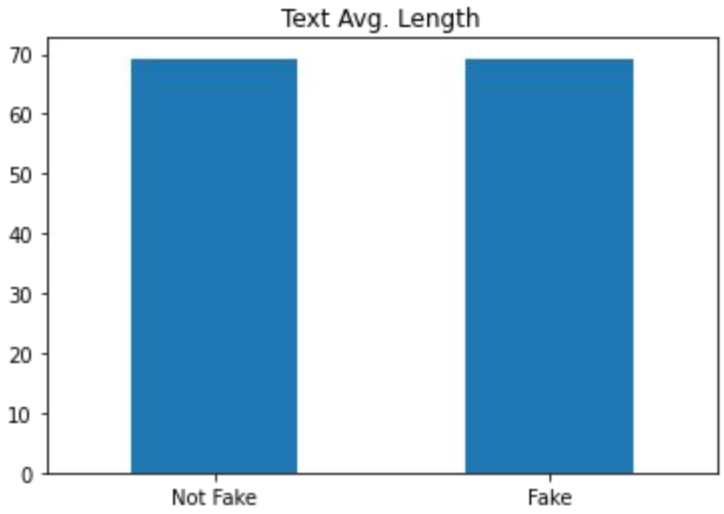
Agenda

- Exploratory Data Analysis
- Data Preprocessing
- LDA & Part-of-speech Tagging
- LDA - K-topics Selection
- Model Selection & Hyperparameter tuning
- Results & Conclusions

Exploratory Data Analysis

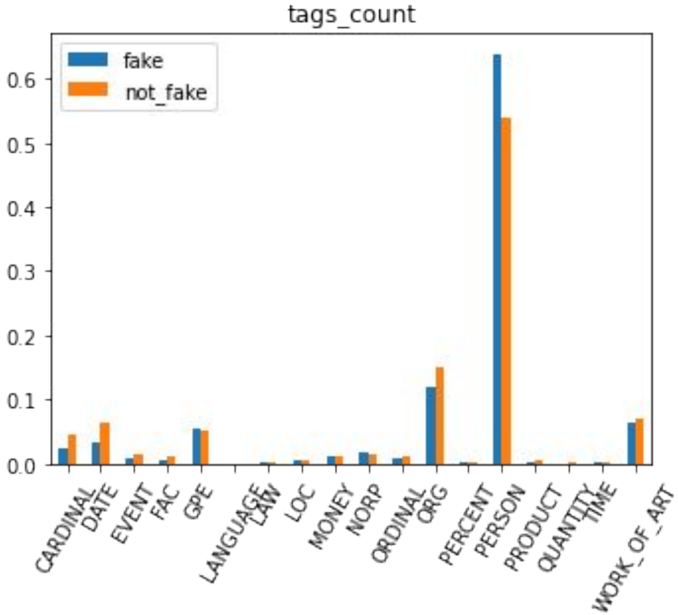
- First of all, by checking the label distribution, we see that this is an unbalanced dataset with a majority of not-fake news.
- When checking the average length of texts in both classes, we see that they are almost equal.

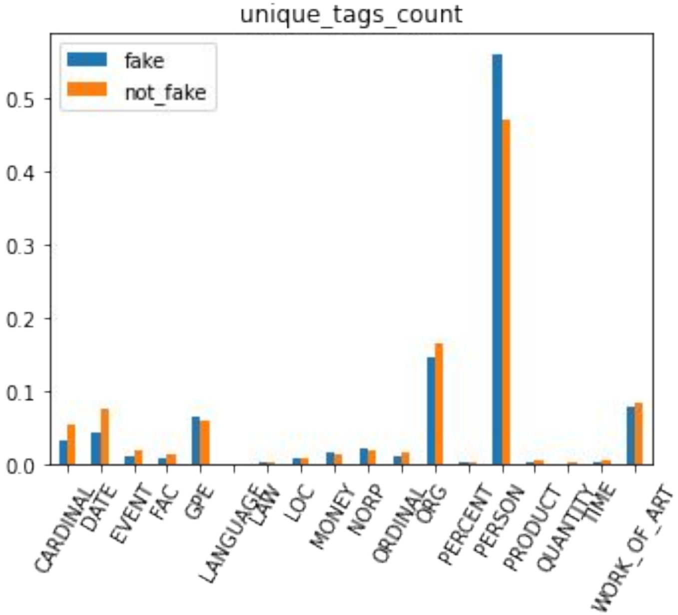




Exploratory Data Analysis

- In order to better understand the characteristics of the dataset, we used a technique called **Named entity recognition** to extract tags from each sentence.
- When visualizing the count of tags and unique tags for each class, we notice that a fake news item is more likely to involve a person compared to non-fake news, while a non-fake item is more likely to involve an organisation, dates etc..

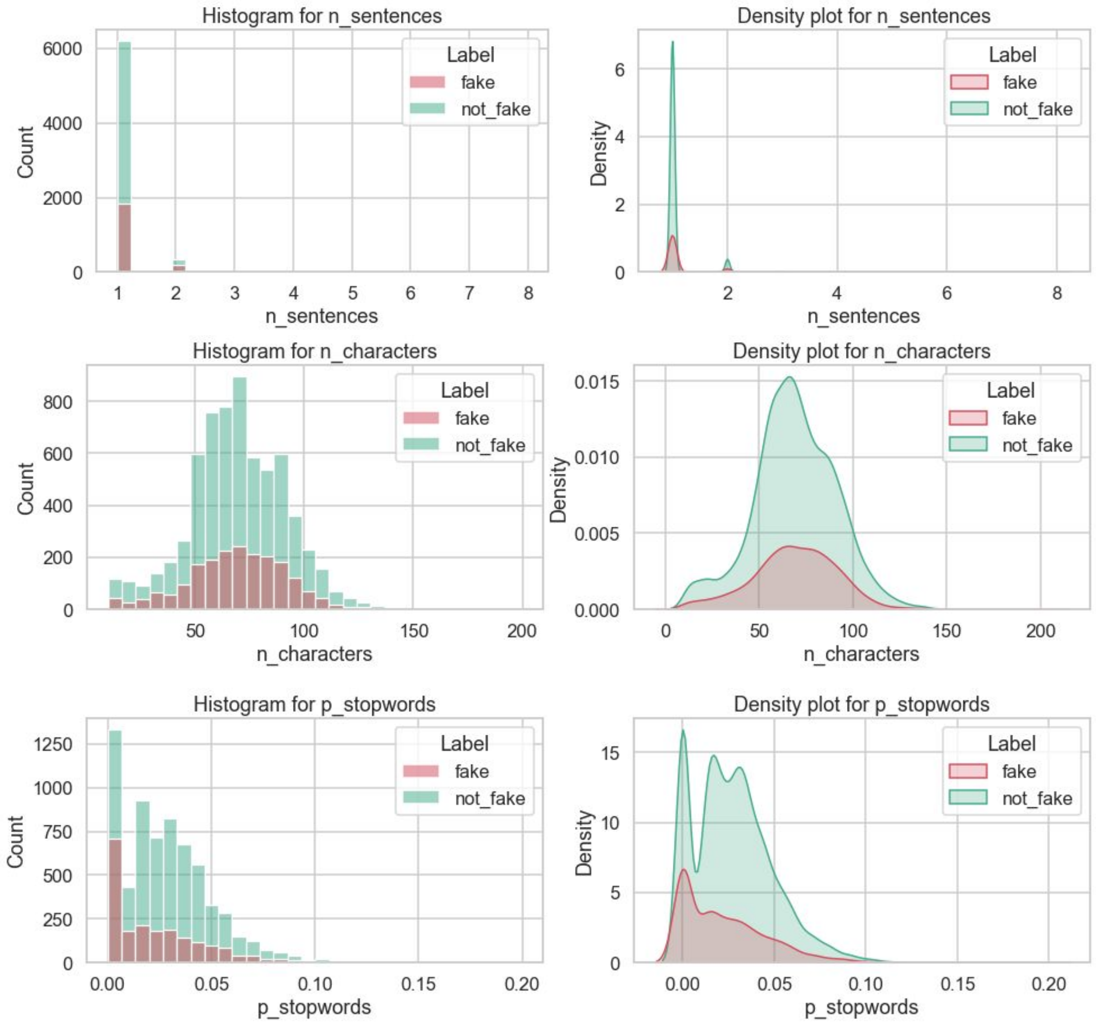




Exploratory Data Analysis

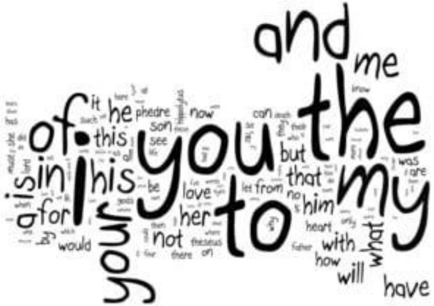
- Before continuing to preprocess the data, we extracted and visualised additional features regarding each text:
1. Number of sentences
2. Number of characters
3. Percentage of stop words

- It seems that in fake news, the percentage of stop words is usually lower than in non-fake news.



Data Preprocessing

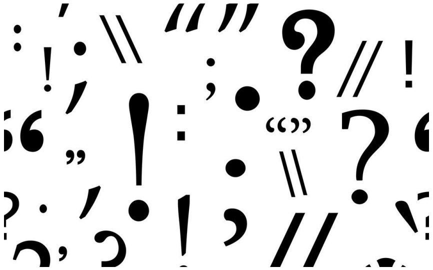
Stop Words Removal



Lower Casing

a	b	c	d	e	f	g
h	i	j	k	l	m	n
o	p	q	r	s	t	u
v	w	x	y	z		

Punctuation Removal



Lemmatization

was → (to) be
better → good
meeting → meeting

Tokenizing

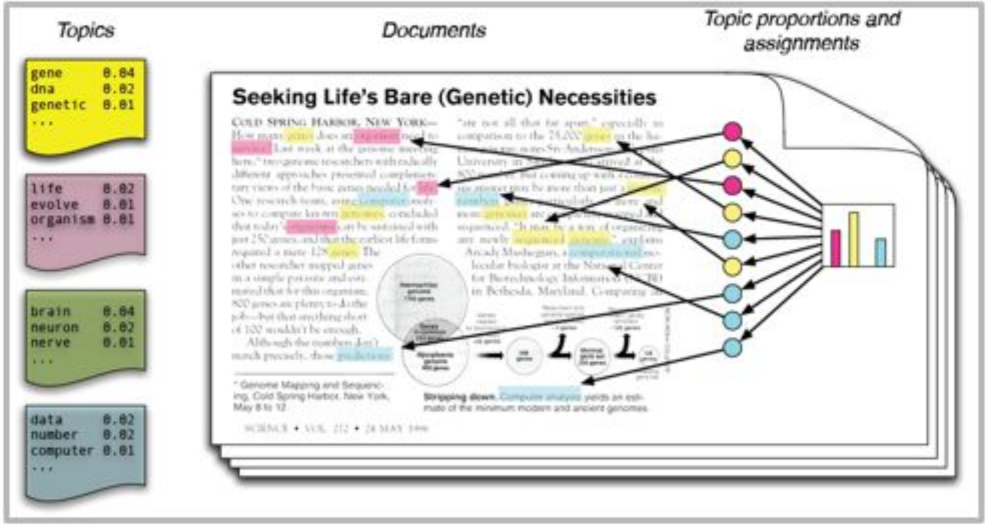
Text
"The cat sat on the mat."
↓
Tokens
"the", "cat", "sat", "on", "the", "mat", "."

TF-IDF

	Doc 1	Doc 2	...	Doc *n*
Term(s) 1	12	2	...	1
Term(s) 2	0	1	...	0
...	...	...	...	
Term(s) *n*	0	6	...	3

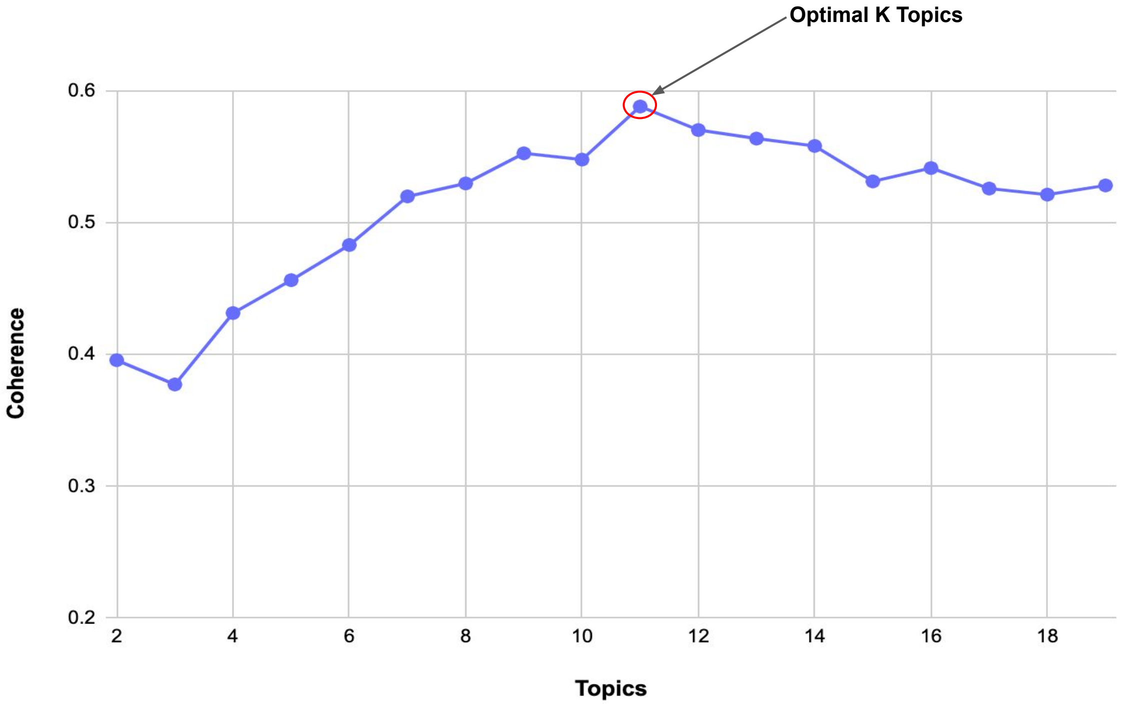
LDA - Latent Dirichlet Allocation

- As part of the features engineering part, we used LDA to extract topics from our corpus (for ex. Politics or sports).
- We first identify the optimal K topics, then we predicted the topics' probability of each record and finally we added the probabilities as new features to our matrix.
- Since LDA is unsupervised method, we run it for the train and the test together



LDA - K-Topics Selection

- In order to choose the optimal number of topics, we strived to maximize the **coherence** of the resulting representation.
- We found that **k=11** topics achieve maximal coherence.
- After choosing k, we ran the model on all of the samples, resulting in an extra 11 features for each sample, where each value is the percentage of a specific topic found in a specific text.



Final Features

	Words (TF-IDF)					Named Entity Recognition Tags				LDA Topics Probabilities			
	Term 1	Term 2	Term 3	...	Term N	NER 1	NER 2	...	NER N	Topic prob 1	Topic prob 2	...	Topic prob N
Record 1	XX	XX	XX	...	XX	XX	XX	...	XX	XX	XX	...	XX
Record 2	XX	XX	XX	...	XX	XX	XX	...	XX	XX	XX	...	XX
Record 3	XX	XX	XX	...	XX	XX	XX	...	XX	XX	XX	...	XX
...	XX	XX	XX	...	XX	XX	XX	...	XX	XX	XX	...	XX
...	XX	XX	XX	...	XX	XX	XX	...	XX	XX	XX	...	XX
...	XX	XX	XX	...	XX	XX	XX	...	XX	XX	XX	...	XX
...	XX	XX	XX	...	XX	XX	XX	...	XX	XX	XX	...	XX
...	XX	XX	XX	...	XX	XX	XX	...	XX	XX	XX	...	XX
Record N	XX	XX	XX	...	XX	XX	XX	...	XX	XX	XX	...	XX

Model selection & Hyperparameter tuning

- After completion of the preprocessing pipeline, we tried out different models with a variety of different hyperparameters to learn the training data and predict the class of unseen samples.
- This was done using the **k-fold cross validation** technique, setting different subsets of the training data as a validation sets, predicting them, and comparing them to the labels.
- As seen in the comparison below, the **SVM** classifier, optimizing loss using **stochastic gradient descent** managed to minimize all 3 important metrics: **f1 ,accuracy and recall**, hence we will choose it as the optimal model for prediction.

clf	f1_score	accuracy_score	recall_score	best_params
SGDClassifier	61.04%	83.27%	54.97%	{'penalty': 'l2', 'n_jobs': -1, 'alpha': 0.0001}
LogisticRegression	60.76%	83.13%	54.78%	{'solver': 'liblinear', 'C': 29.763514416313132}
MultinomialNB	38.38%	80.90%	24.95%	{'alpha': 0.25}
RandomForestClassifier	52.60%	80.90%	44.44%	{'max_samples': None, 'max_features': 'sqrt', 'criterion': 'gini'}
XGBClassifier	51.01%	80.90%	41.72%	{'min_child_weight': 3, 'max_depth': 10, 'gamma': 0.1, 'eta': 0.2, 'colsample_bytree': 0.3}
AdaBoostClassifier	36.92%	79.83%	24.76%	{'learning_rate': 0.75}

Results and conclusions

- After choosing the stochastic gradient descent classifier, we used it to learn the preprocessed training set together with its labels.
- After that, we passed the test set through the preprocessing pipeline and used the classifier to predict whether each sample is fake news or not.
- The **F-score** for our predictions on the test set was **61.61%**

- In this analysis we explored natural language preprocessing methods that were new to us
- We found out that the straightforward **Bag-of-Words** approach can be improved by extracting additional features for each text using the **Named entity recognition** and **Latent Dirichlet Allocation** methods.
- Since the F-score for our final predictions (61.61%) was not lower than the cross validated F-score we received when choosing the optimal model (61.04%), we conclude that our model manages to generalize and doesn't overfit the training data.