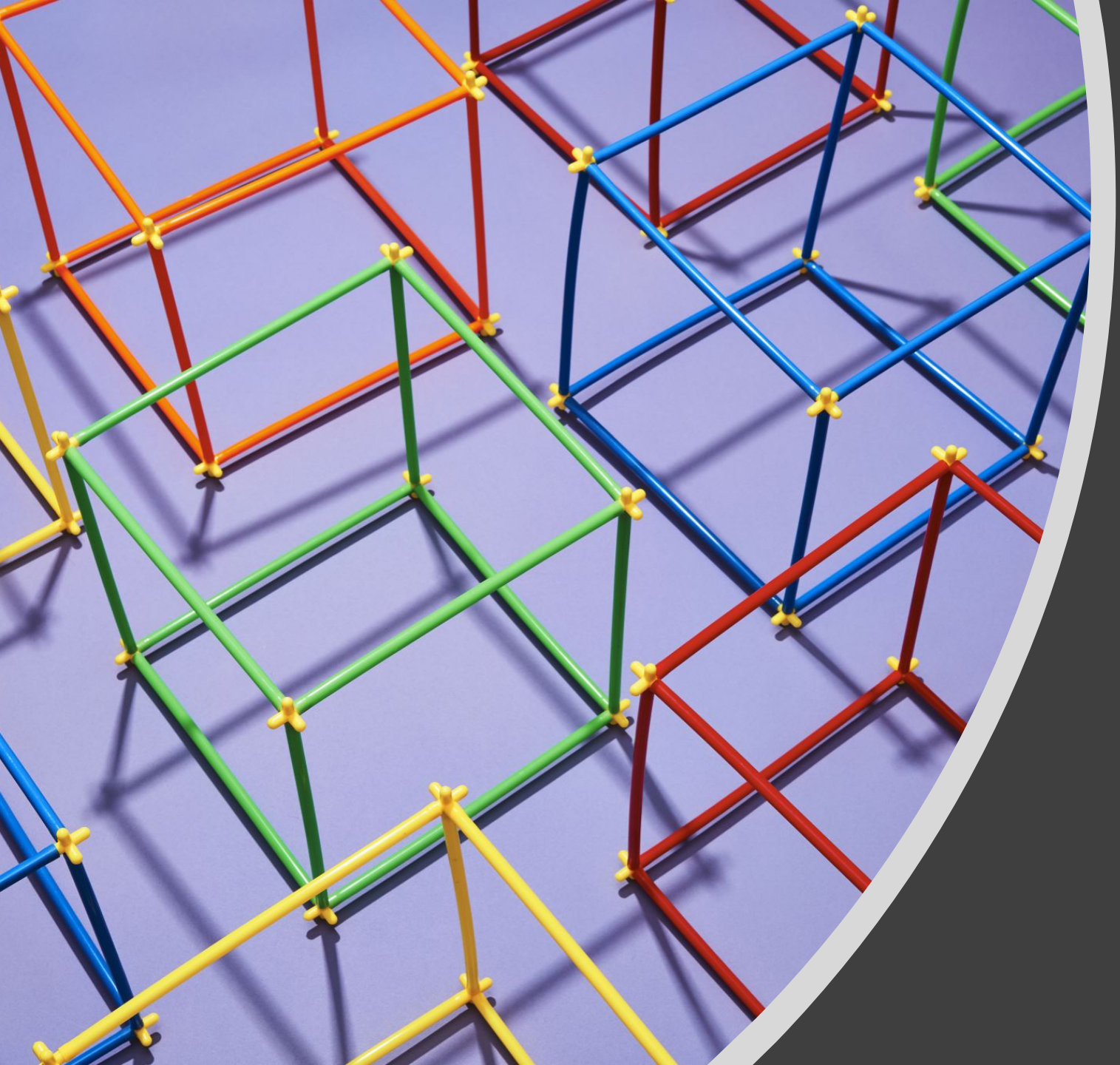


Classification in Computational Social Science

Honglin Carson Bao

baohlcs@gmail.com

www.carsonhlbao.com

Classification

In statistics, classification is the problem of identifying which of a set of categories (sub-populations) an observation (or observations) belongs to.

It is very important in computational social science!

- Toxicity
- Auto-labeling
- Ideology
- …

A brief review of classification in CSS

- Binary
- Multi-class and multi-label classification
- Advanced techniques, such as insufficient text data, imbalanced text data across classes, and some fancy deep learning models
- Model evaluation

Classification Basics

- Preprocessing
- Vectorization
- Split training/testing and load model
- Evaluation

Preprocessing

- converting text to lowercase, removing non-necessary confounding text, etc.

Vectorization

- One-hot Encoding (OHE)
- Count Vectorizer
- Bag-of-Words (BOW)
- N-grams
- Term Frequency-Inverse Document Frequency (TF-IDF)
- W2V
- …

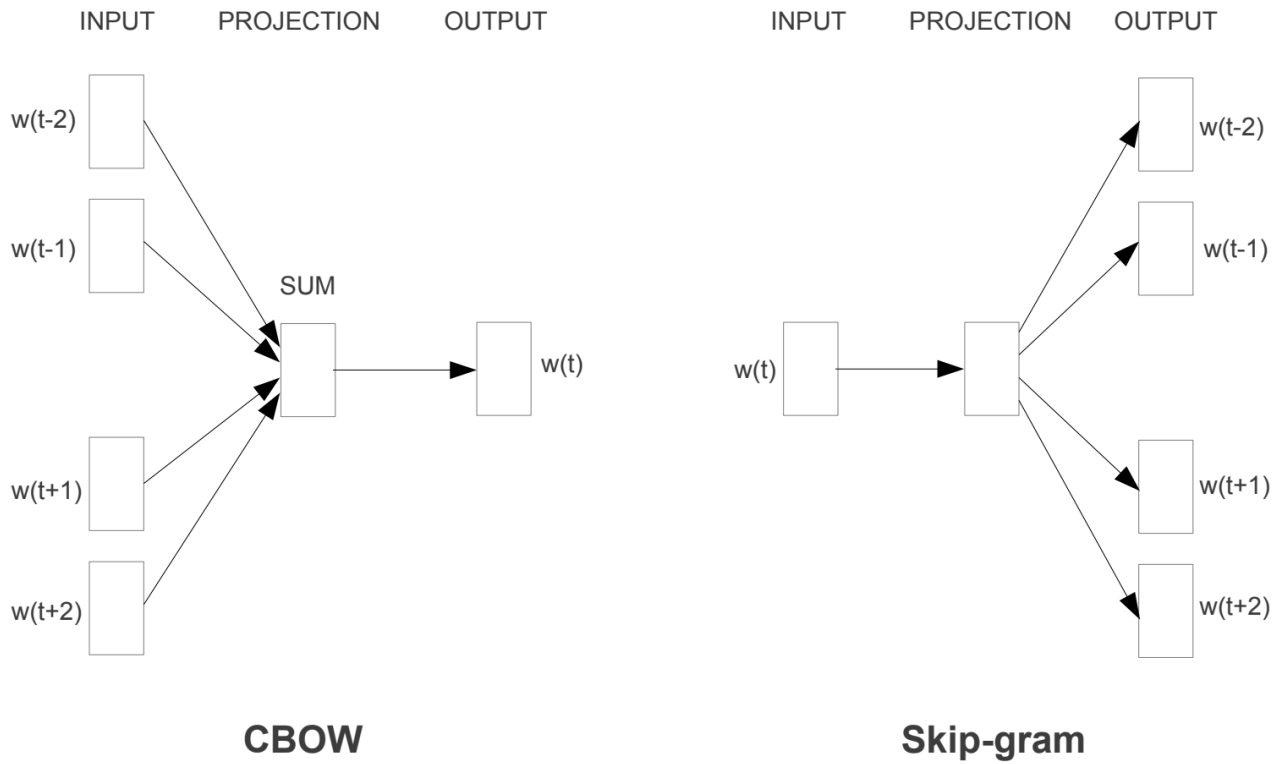
```
model = Sequential()
model.add(Embedding(MAX_NB_WORDS, EMBEDDING_DIM, input_length = X.shape[1]))
```

- input_length is the length of the input text data
- input_dim is the dimension of the text data.

For example, if the content of a piece of text data is: ['apple', 'apple', 'car’]

one-hot encoding will be [[1 0], [1 0], [0 1]]. batch_size = 3, input_dim = 2, input_length = 3.

W2V – similarities



representation of words for text analysis, typically in the form of a real-valued vector that encodes the meaning of the word such that the words that are closer in the vector space are expected to be similar in meaning.

TF-IDF method

In the TF-IDF method, a document term matrix is generated, and each column represents a single unique word. The difference in the TF-IDF method is that each cell doesn't indicate the term frequency, but the cell value represents a weighting that highlights the importance of that particular word to the document.

$$W_{x,y} = tf_{x,y^*} \log\left(\frac{N}{df_x}\right)$$

$W_{x,y}$= Word x within document y

$tf_{x,y}$= frequency of x in y

df_x= number of documents containing x

N=total number of documents

Text as Data

Label

text	name	chamber	party	account_type
[proud, work, we, done, past, 12, months, —, d...	Lori Trahan	house	D	office
[high, school, students, across, our, district...	Lori Trahan	house	D	office
[young, people, our, country, powerful, asset,...	Lori Trahan	house	D	office
[starting, 2020, right, way, thankful, everyda...	Lori Trahan	house	D	office
[2019, year, many, firsts, us, working, #ma3, ...	Lori Trahan	house	D	office
...	...	...	...	...
[ring, new, year, relax, bit, join, upcoming, ...	Kwanza Hall	house	D	campaign
[@dunkindonuts, love, location, @thehill, http...	Kwanza Hall	house	D	campaign
[xx, ≠, xy, https://twitter.com/albertmohler/s...	Warren Davidson	house	R	campaign

Model

- Instance-based learning or Model-based?

Example of instance-based learning algorithm is the k-nearest neighbor algorithm

Model-based: SVM

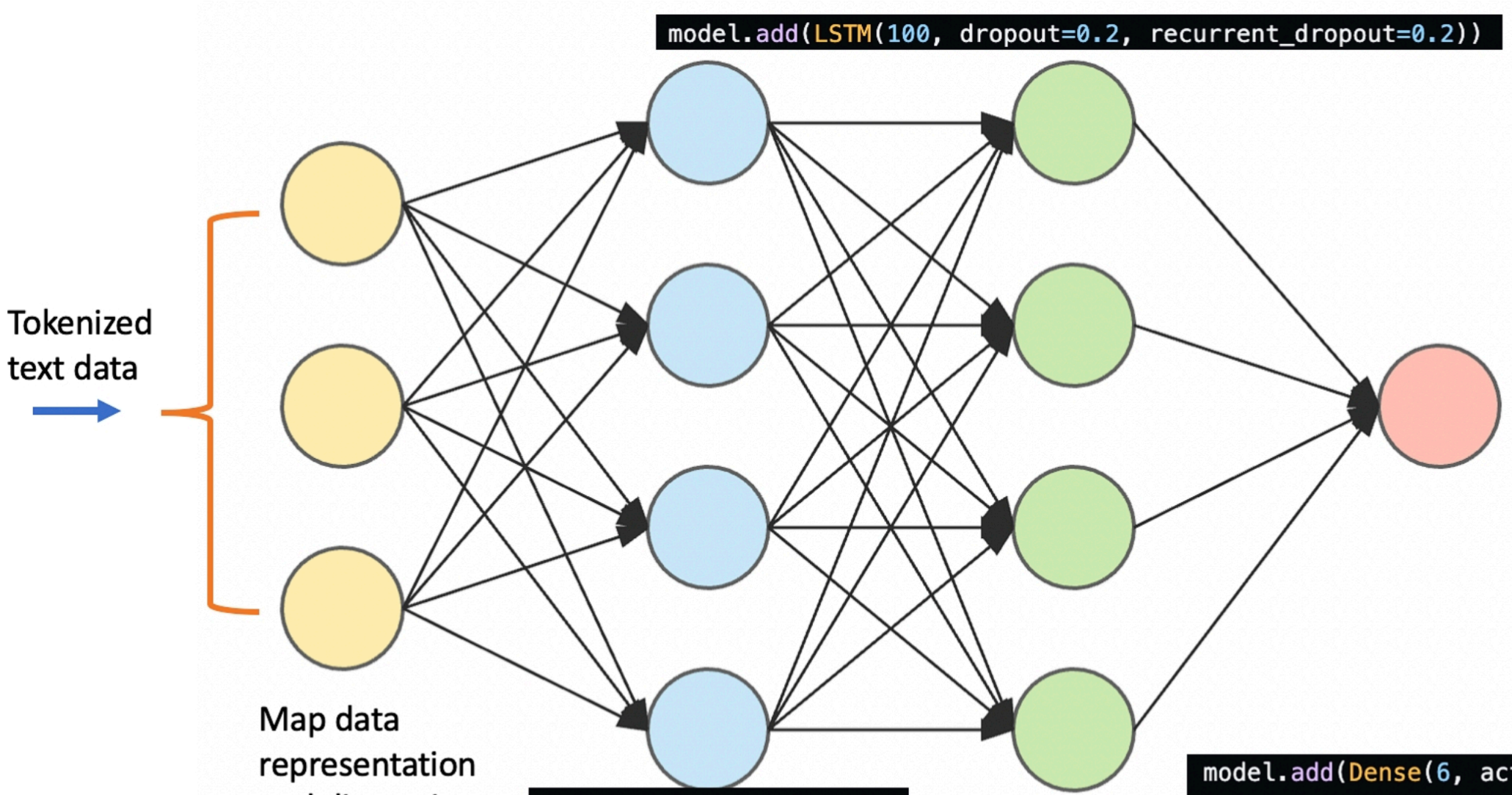
- Naive Bayes
- Logistic Regression
- SVM
- Random Forest

```
#FITTING THE CLASSIFICATION MODEL using Logistic Regression(tf-idf)
lr_tfidf=LogisticRegression(solver = 'liblinear', C=1.0, penalty = 'l2')
#lr_tfidf=LogisticRegression(solver = 'liblinear', C=10, penalty = 'l2')
lr_tfidf.fit(X_train_vectors_tfidf, y_train)  #model
y_predict = lr_tfidf.predict(X_val_vectors_tfidf)
y_prob = lr_tfidf.predict_proba(X_val_vectors_tfidf)[:,1]
print(classification_report(y_val,y_predict))
print('Confusion Matrix:',confusion_matrix(y_val, y_predict))
fpr, tpr, thresholds = roc_curve(y_val, y_prob)
roc_auc = auc(fpr, tpr)
print('AUC:', roc_auc)
```

Multiclass VS Multilabel

- If the possible labels for an example are A, B, and C, the label powerset representation of this problem is a multi-class classification problem with the classes [0 0 0], [1 0 0], [0 1 0], [0 0 1], [1 0 1], [0 1 1]. [1 1 1], where [1 0 1] denotes an instance in which labels A and C are present but label B is not

- https://scikit-learn.org/stable/modules/multiclass.html




```
model.add(Embedding(MAX_NB_WORDS, EMBEDDING_DIM, input_length = X.shape[1]))
model.add(Dense(6, activation='relu', input_dim = X_train.shape[1]))
```

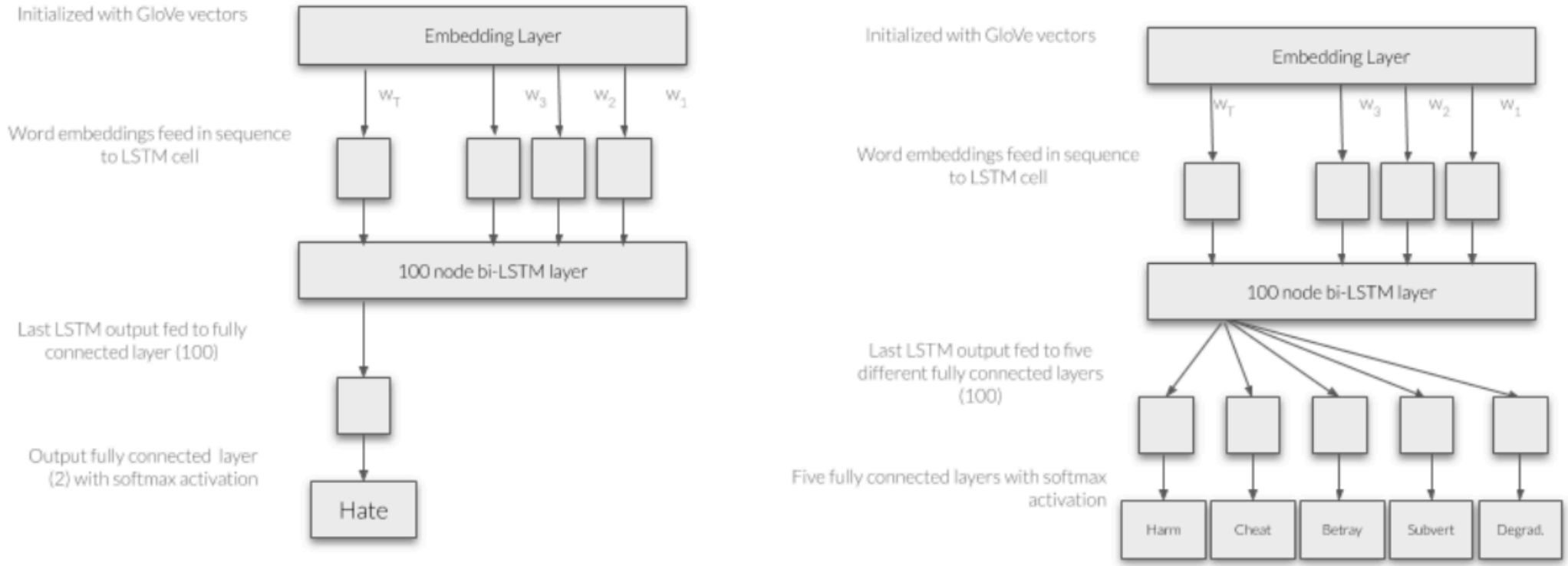


Figure 1. Visualizations of the classification models trained on Gab data and used to predict on entire Gab corpus.

Hoover, J., Atari, M., Mostafazadeh Davani, A. et al. Investigating the role of group-based morality in extreme behavioral expressions of prejudice. *Nature Communication* 12, 4585 (2021).

Sentiment Classification

- Dictionary-based Method (+/-)

$$s(i) = \sum_{j}^{n} (-1)^t * k * word(j)$$

- Comparative Agendas Project (CAP) codebook
- Polarity-based Method: Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: Sentiment classification using machine learning techniques. *EMNLP* 2002.

- fine grained

- When working with short texts such as tweets, the probability for dictionary words to appear in such a short text is low (Zirn et al., 2016). Moreover, with new words or terms being generated, a dictionary —mostly designed for formal text— soon becomes outdated (Wu et al., 2018). At the same time, extending dictionaries to improve coverage might come at the expense of lower precision.
- this approach has the downside that results will be biased by the performance of the dictionary.

Praet, Stiene, et al. “Comparing automated content analysis methods to distinguish issue communication by political parties on Twitter.” *CCR* (2021).

- We turn in the third section to analyze different tools for the automatic detection of sentiment in text. We find that although leading dictionary and supervised approaches predicted human judgments reasonably well, automatically generated sentiment dictionaries based on word embeddings surpassed these other approaches. This methodology leverages the ability of word embeddings to identify semantically related words, and we elaborate on the conditions needed for the induction of dictionaries.

Cochrane, C., Rheault, L., Godbout, J. F., Whyte, T., Wong, M. W. C., & Borwein, S. (2021). The Automatic Analysis of Emotion in Political Speech Based on Transcripts. *Political Communication*, 1-24.

We also test a third and increasingly common approach which involves creating dictionaries automatically, without the need for hand-made curation. We use the word2vec algorithm to generate distributed vector space representations of words – commonly known as "word embeddings" (Mikolov, Sutskever et al., 2013). Next, we take advantage of a property of these embeddings, namely their ability to identify semantically related words, and generate a comprehensive list of words to populate a sentiment dictionary. As we illustrate below, this approach outperforms alternatives, is robust across some domains, and it can be very easily implemented at low cost.

The word2vec algorithm uses a shallow layer neural net to predict the occurrence of words on the basis of surrounding words (Mikolov, Sutskever et al., 2013).[5] The weights (or coefficients) from this model become the word embeddings, representations with the convenient property that words commonly used in the same contexts are close to each other in the vector space. This means that distance metrics such as cosine similarity – based on the angles between vectors – can be used to uncover semantic associations between words. In this way, we can say that the algorithm embeds the contextual likeness of words in the angular distance between their corresponding vectors. Consistent with the word-use theory of meaning (Wittgenstein, 2009), mathematical operations on these vectors yield results that correspond remarkably well with meaningful semantic patterns in language; for instance, analogies. Canonically, models trained on large corpora are able to solve that the vector for "king" minus the vector for "man" plus the vector for "woman" approximates the vector for "queen," hence capturing the semantic relatedness between pairs of words (Mikolov, Corrado et al., 2013; Pennington et al., 2014).

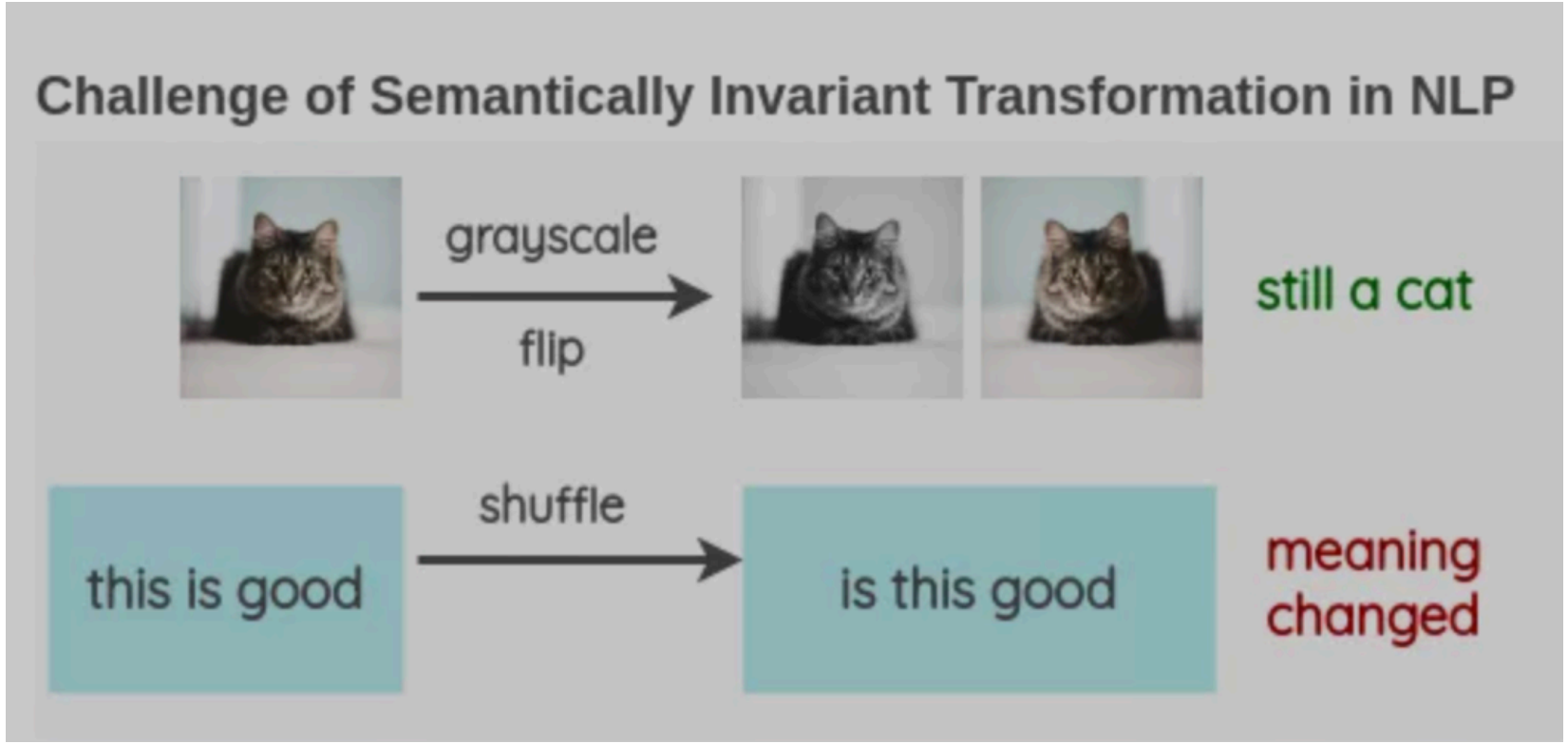
Table 2. Most Valenced Words in Induced Sentiment Lexicon

	Positive Words		Negative Words	
1.	excellent	.2630	horrible	-.3066
2.	mentorship	.2218	terrible	-.2936
3.	highquality	.2189	panic	-.2915
4.	outstanding	.2128	horrendous	-.2842
5.	invaluable	.2071	outrageous	-.2829
6.	innovative	.2012	disastrous	-.2819
7.	midwives	.2001	despicable	-.2758
8.	talents	.1978	scandalous	-.2742
9.	topnotch	.1942	horrific	-.2724
10.	collaboratively	.1917	inexcusable	-.2691
11.	fortunate	.1903	horribly	-.2635
12.	secure	.1899	indiscriminate	-.2586
13.	productive	.1894	immoral	-.2565
14.	develop	.1892	devastating	-.2559
15.	welcome	.1880	senseless	-.2551
16.	excellence	.1877	callous	-.2551
17.	ethic	.1877	deception	-.2545
18.	continue	.1872	unspeakable	-.2539
19.	worldclass	.1872	unfortunate	-.2524
20.	strengthen	.1869	unjustified	-.2510
21.	cooperative	.1865	inhumane	-.2500
22.	constructive	.1863	inexplicable	-.2493
23.	dedicated	.1839	atrocious	-.2486
24.	thoughtful	.1820	shameful	-.2483
25.	tirelessly	.1780	irresponsible	-.2474
26.	cooperatively	.1767	disgraceful	-.2468
27.	collaborative	.1763	reprehensible	-.2466
28.	build	.1762	disgusting	-.2463
29.	happy	.1752	pernicious	-.2463
30.	constructively	.1731	meanspirited	-.2459

In geometric terms, for any given word, if the angles formed between the vectors for positive words are, on average, smaller than the angles formed between the vectors for negative words, then the word is semantically closer to positive than to negative words, and we infer from that a positive sentiment for the word.

How many human labels, at least, you need?

- Unlike computer vision, augmenting limited text data is still an open question!

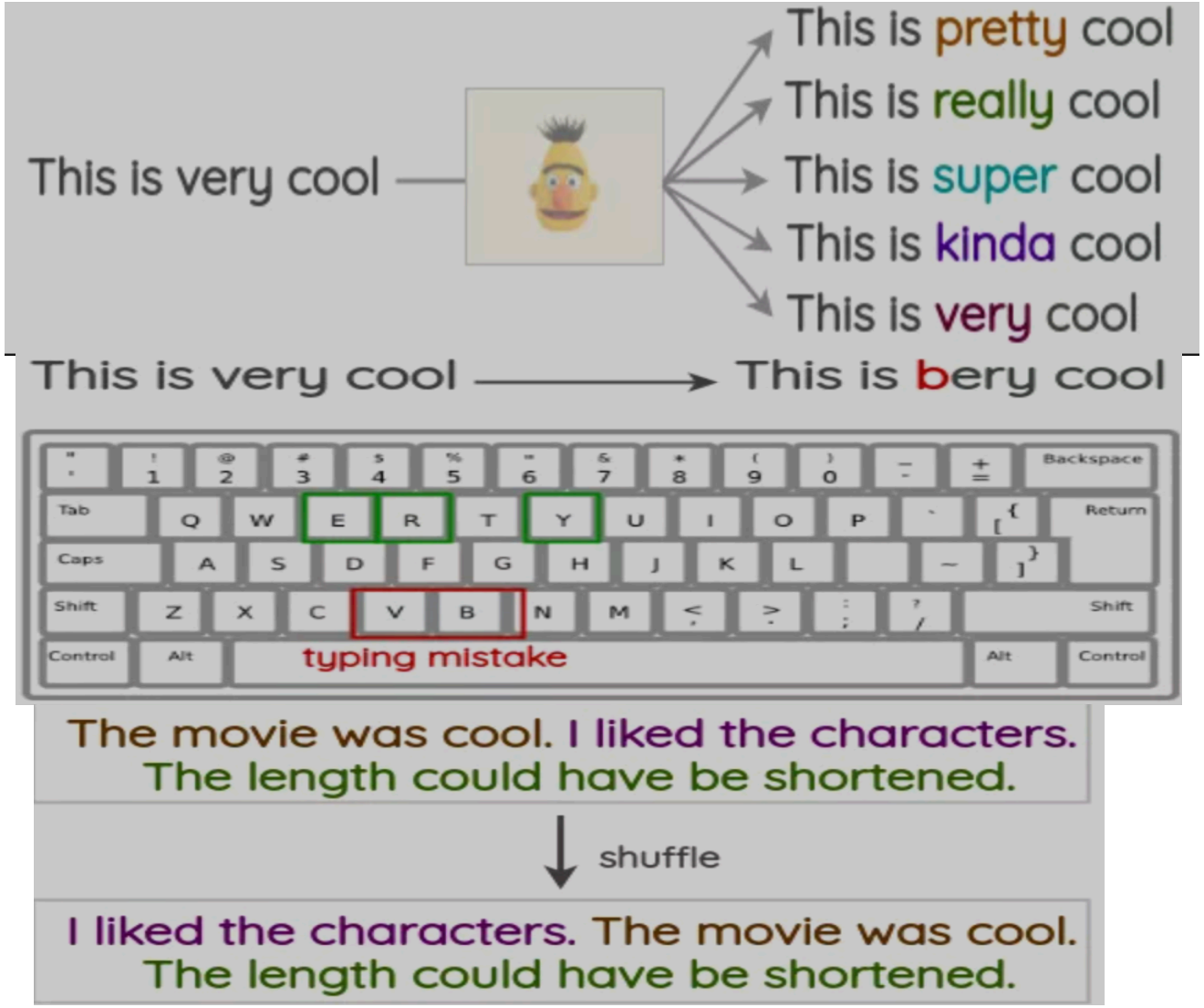


Some methods, check out this

KEY IDEA

vocabulary substitution

(w2v, dictionary, TF-IDF…)



- employing labeled data from a related task but different corpus or using hashtags or well-defined keywords as annotations instead of human coding (Hasan et al., 2014; Gupta & Hewett, 2020). Next to that, semi-supervised learning (Van Engelen & Hoos, 2020) and transfer learning (Terechshenko et al., 2020) can be relevant to train a classifier when labeled data is scarce. The latter have been shown to outperform traditional classifiers with the same amount of (coded) training data (Terechshenko et al., 2020) but are increasingly complex and computationally demanding.

- https://polmeth2020.org/event-1-vr-4
- http://web.cs.wpi.edu/~emmanuel/publications/PDFs/C25.pdf

Virtual Room 4: Text-as-Data

Date: Tuesday, July 14, 2020, 12:00pm to 1:30pm

Chair: Suzanna Linn (Penn State University)

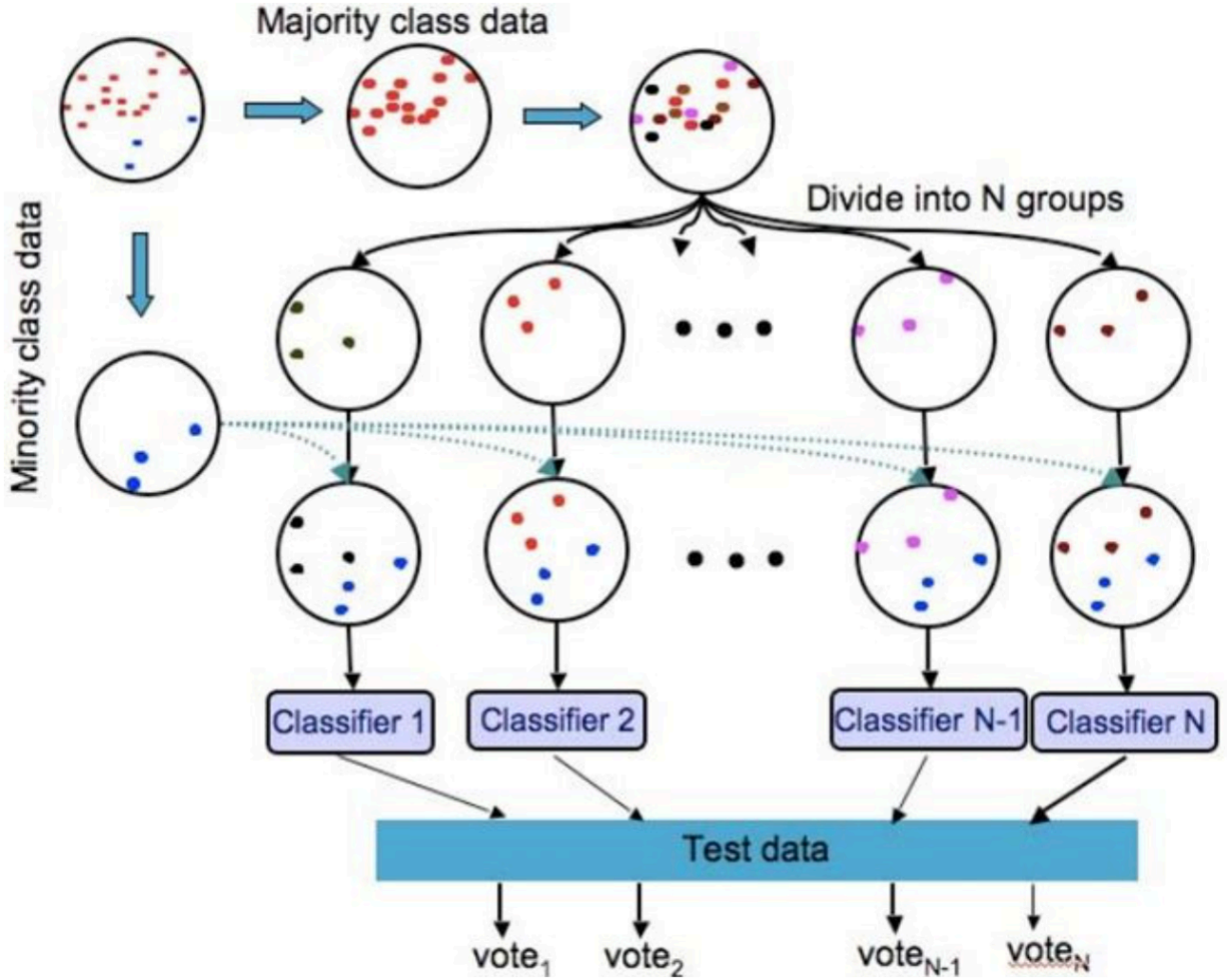
Co-Host: Justin Savoie (University of Toronto)

A comparison of methods in political science text classification: Transfer learning language models for politics

Author(s): Zhanna Terechshenko, Fridolin Linder, Vishakh Padmakumar, Fengyuan Liu, Jonathan Nagler, Joshua A. Tucker and Richard Bonneau

Discussant: Leah Windsor (Institute for Intelligent Systems, University of Memphis)

What if you have imbalanced labels?



https://www.tensorflow.org/tutorials/structured_data/imbalanced_data

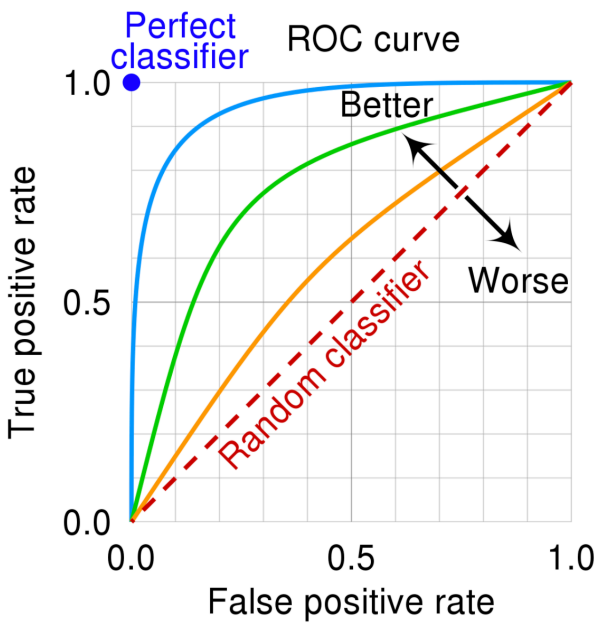
https://imbalanced-learn.org/stable/references/generated/imblearn.ensemble.EasyEnsembleClassifier.html

https://arxiv.org/pdf/2002.04592.pdf

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html

Evaluation

- Accuracy: (TP+TN)/All
- Precision: TP/(TP+FP)
- Recall (upper bound): TP/(TP+FN)
- $F_\beta = (1+\beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$
- ROC/AUC

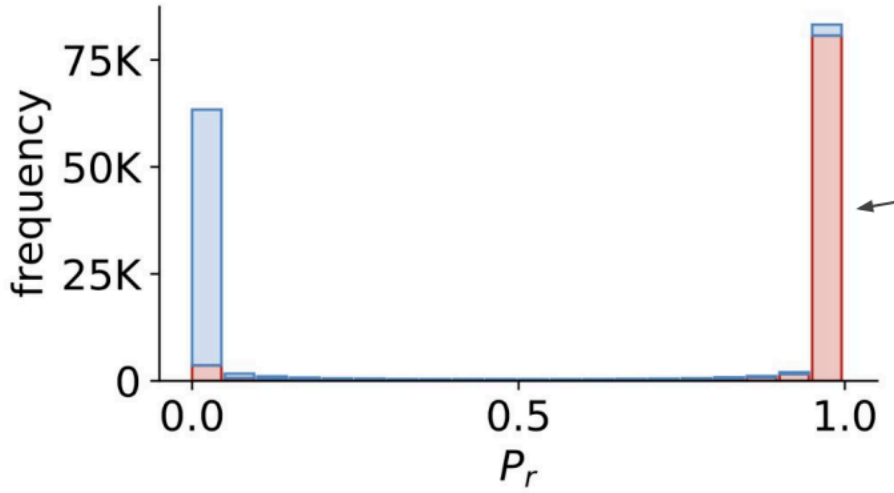


Classified

Actual	Yes	No
Yes	TP	FN
No	FP	TN

Predicting user political leaning with Hierarchical Attention Network

- Procedure:
  - Assign political leaning labels of seed users with label propagation;
  - Train HAN models (Yang et al. 2016) based on comments posted by seed users;
  - Classify the remaining users with trained HAN models.
- Results:
  - HAN models[1] with high accuracy on hold-out set (93%).
  - *Pr* >= 0.95, classify as conservative; *Pr* <= 0.05, classify as liberal.



Users with high *Pr*, the majority of them are indeed conservatives.

[1] Pre-trained HAN models are released, check our Github repo on the last page!

Cross-Partisan Discussions on YouTube: Conservatives Talk to Liberals but Liberals Don't Talk to Conservatives

Siqi Wu[1,2], **Paul Resnick**[1]
[1] University of Michigan
[2] Australian National University
{siqiwu, presnick}@umich.edu

Some fancy deep learning

- BERT
- GPT
- Switch-transformer (1600 Billion Parameters!!!)
- ……

- Why? **Pre-training**
- You have been given some initialized parameters in advance, which are not random, but learned from other similar datasets, and then your dataset is used to learn and get the parameters suitable for your dataset.
- https://ijoc.org/index.php/ijoc/article/view/10725

- Recent advances in data-enhanced dictionaries, deep learning, transfer learning and semi- supervised learning offer exciting avenues for political text classification while at the same time introducing a lot of additional complexity and requiring ever more computing power. Adapting text classification to the volatility of social media remains a delicate exercise. Therefore, a promising method to study issue communication on social media is to start from a data-driven approach and use domain knowledge to interpret and understand the results.

- Features -- coupling with expert knowledge

https://misq.org/is-ai-ground-truth-really-true-the-dangers-of-training-and-evaluating-ai-tools-based-on-experts-know-what.html

- Correlation plot
- Manual feature construction

`sklearn.feature_selection`: Feature Selection

The `sklearn.feature_selection` module implements feature selection algorithms. It currently includes univariate filter selection methods and the recursive feature elimination algorithm.

User guide: See the Feature selection section for further details.

`feature_selection.GenericUnivariateSelect`([...])	Univariate feature selector with configurable strategy.
`feature_selection.SelectPercentile`([...])	Select features according to a percentile of the highest scores.
`feature_selection.SelectKBest`([score_func, k])	Select features according to the k highest scores.
`feature_selection.SelectFpr`([score_func, alpha])	Filter: Select the pvalues below alpha based on a FPR test.
`feature_selection.SelectFdr`([score_func, alpha])	Filter: Select the p-values for an estimated false discovery rate.
`feature_selection.SelectFromModel`(estimator, *)	Meta-transformer for selecting features based on importance weights.
`feature_selection.SelectFwe`([score_func, alpha])	Filter: Select the p-values corresponding to Family-wise error rate.
`feature_selection.SequentialFeatureSelector`(...)	Transformer that performs Sequential Feature Selection.
`feature_selection.RFE`(estimator, *[, ...])	Feature ranking with recursive feature elimination.
`feature_selection.RFECV`(estimator, *[, ...])	Recursive feature elimination with cross-validation to select the number of features.
`feature_selection.VarianceThreshold`([threshold])	Feature selector that removes all low-variance features.

`feature_selection.chi2`(X, y)	Compute chi-squared stats between each non-negative feature and class.
`feature_selection.f_classif`(X, y)	Compute the ANOVA F-value for the provided sample.
`feature_selection.f_regression`(X, y, *[, center])	Univariate linear regression tests returning F-statistic and p-values.
`feature_selection.r_regression`(X, y, *[, center])	Compute Pearson's r for each features and the target.
`feature_selection.mutual_info_classif`(X, y, *)	Estimate mutual information for a discrete target variable.
`feature_selection.mutual_info_regression`(X, y, *)	Estimate mutual information for a continuous target variable.

Data are People
Julia Ticona, Annenberg Penn

- Computer science takes data for granted
- **But high-quality data is the most important thing for CSS**

(DATA COLLECTION)

- Garbage in Garbage out
- Collective voting
- Feature

Halevy, Alon, Peter Norvig, and Fernando Pereira. "The unreasonable effectiveness of data." *IEEE Intelligent Systems* 24.2 (2009): 8-12.

Thanks!

Discussion

https://github.com/hlbao/classification_in_CSS