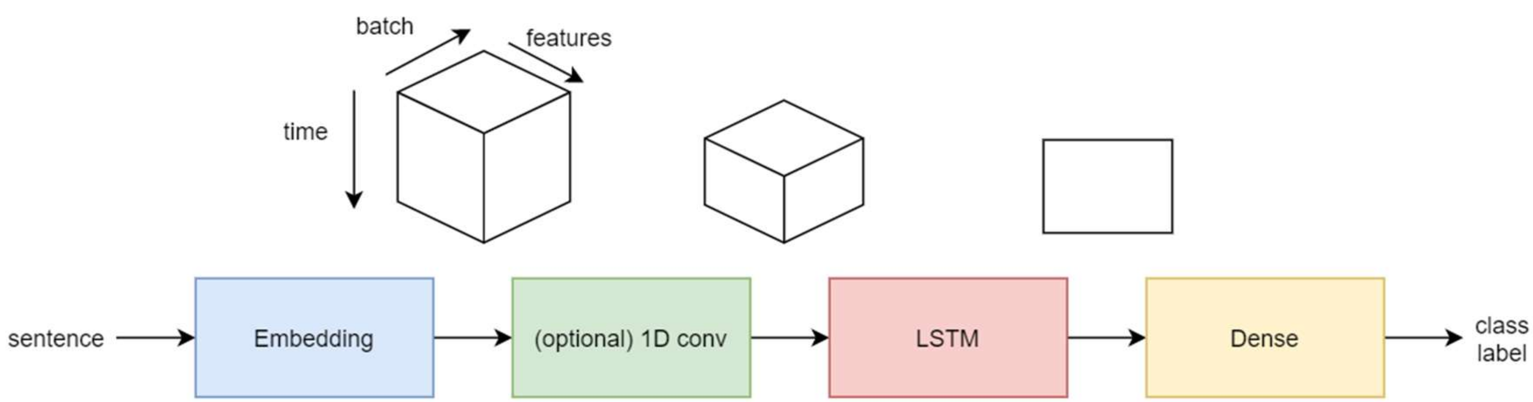
Week 4 – Text Classification

EGCO467 Natural Language and Speech Processing

1/2564

Model for text classification



Example - IMDB

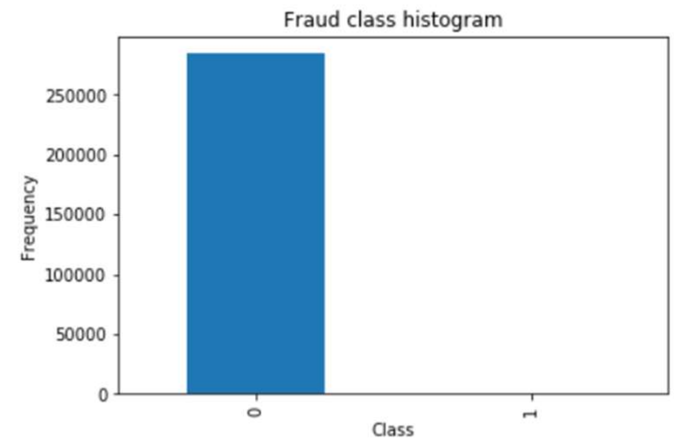
Pitfall of classification

- Imbalance data
- Credit card fraud dataset:
  - 284315 normal
  - 492 fraud
- Model that predicts 0 (normal) all the time has 0.9983 accuracy !

- Real world data always imbalanced: disease diagnosis, manufacturing defect inspection, emotional vs. neutral in text.

Out[3]:

```
0    284315
1       492
Name: Class, dtype: int64
```



TP, TN, FP and FN

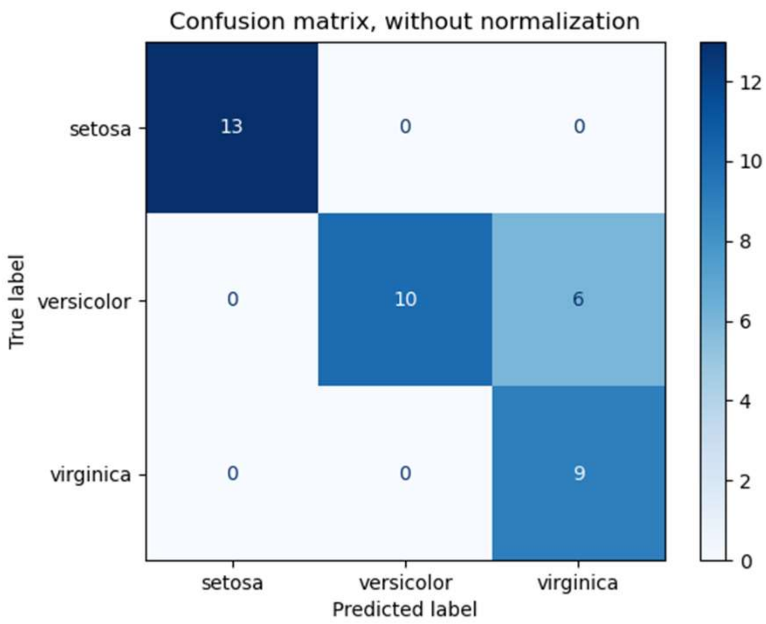
- We need something more robust than accuracy.
- TP = class 1 predicted as class 1
- TN = class 0 predicted as class 0
- FP = class 0 predicted as class 1
- FN = class 1 predicted as class 0

Confusion matrix (2 class)

		True condition	
	Total population	Condition positive	Condition negative
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error
	Predicted condition negative	**False negative**, Type II error	**True negative**

Confusion matrix (multi-class)

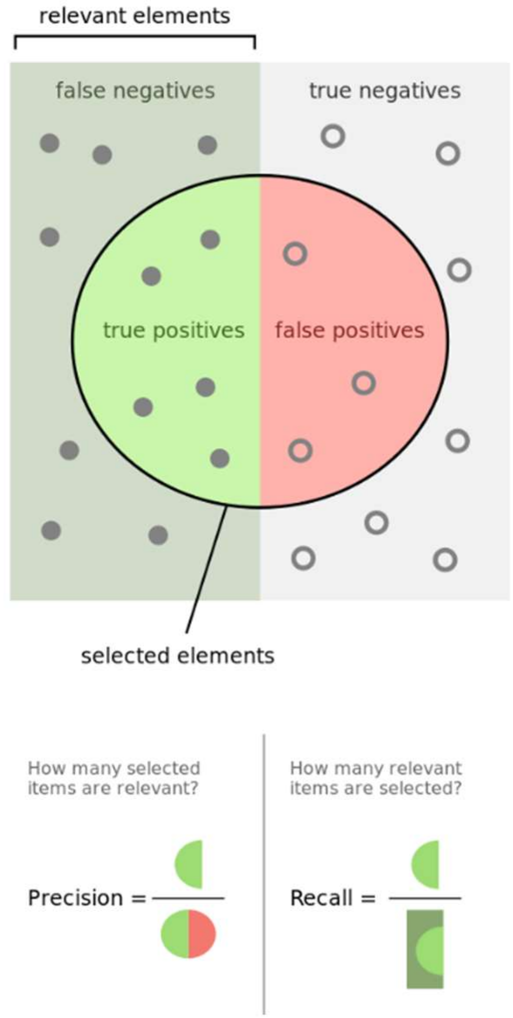
- (this is transpose of the previous slide)
- TPj = actually class j, predicted as class j
- TNj = actually not class j, predicted as not class j
- FNj = actually class j, predicted as not class j
- FPj = actually not class j, predicted as class j
- aka. one-vs.-all scheme



Precision and recall

- precision = **tp / (tp + fp)**
  denominator = everything predicted as positive

- recall = **tp / (tp + fn)**
  denominator = all positives



Accuracy is overrated

- Credit card fraud dataset:
  - 284315 normal
  - 492 fraud
- Model that predicts 0 (normal) all the time has

- TN = 284315
- FN = 492
- **TP = 0**
- FP = 0

- precision = **tp / (tp + fp)**
- recall = **tp / (tp + fn)**
- precision = 0
- recall = 0

- For a model with > 0.99 accuracy !

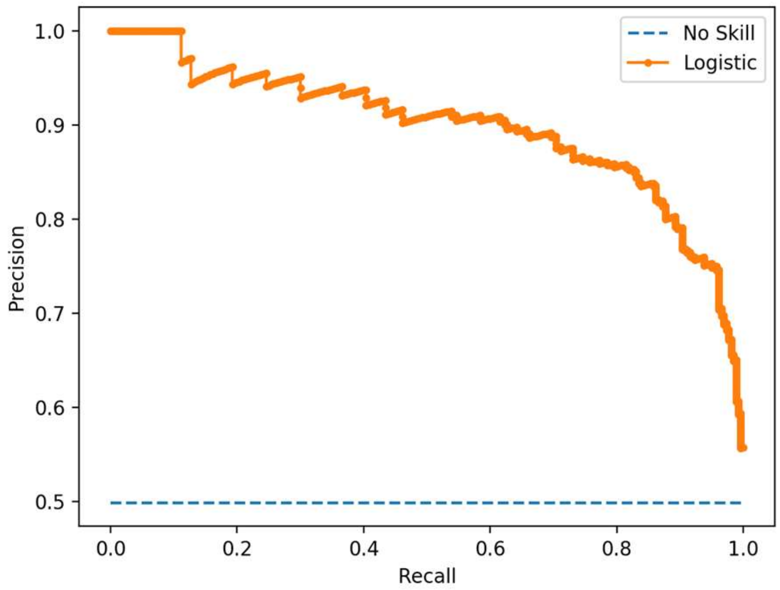
F score aka. F1

- F1 = 2*precision*recall / (precision + recall)
- combines precision/recall into one number
- gold standard for evaluating classification models

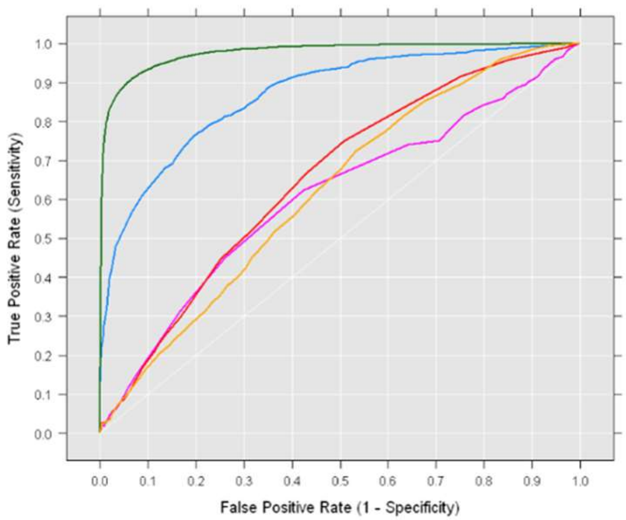
Example - Wongnai

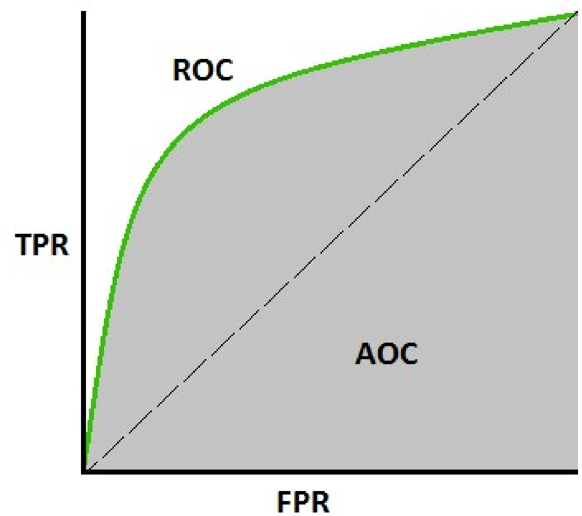
precision recall curve (PR-curve)

- vary the decision threshold t from, say 0.01 to 0.99
- p(x=1) > t => predict as positive
- low t = basically predict everything as positive => high recall, but low accuracy
- high t = do not make positive prediction unless almost 100% sure => high accuracy but low recall
- plot precision/recall value for each value of threshold



ROC, AUC ???





$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

Recall in this context is also referred to as the true positive rate or sensitivity, and precision is also referred to as positive predictive value (PPV); other related measures used in classification include true negative rate and accuracy.[11] True negative rate is also called specificity.

Multi-class F1 and PR-curve

- N classes
- do 1-vs-all for each class
- so get TP_j, TN_j, FP_j, and FN_j
- use them to calculate precision_j and recall_j
- Then calculate $F1_j$
- Overall result for every class is the average of all $F1_j$
- There are N PR-curves, one for each j

- TP_j = actually class j, predicted as class j
- TN_j = actually not class j, predicted as not class j
- FN_j = actually class j, predicted as not class j
- FP_j = actually not class j, predicted as class j

Micro vs Macro F1

- Macro F1: find $F1_j$ first, then average
- Micro F1: 2*micro precision*micro recall /(micro precision + micro recall)
- Micro: for overall measure => do well on all big classes = high
- Macro: when small classes are important => do badly on a small class = low
- Usually we care about small classes, so generally Macro

$$\text{micro precision} = \frac{\sum_{\forall j} TP_j}{\sum_{\forall j} TP_j + \sum_{\forall j} FP_j}$$

$$\text{micro recall} = \frac{\sum_{\forall j} TP_j}{\sum_{\forall j} TP_j + \sum_{\forall j} FN_j}$$

scikit learn’s classification_report

```
from sklearn.metrics import classification_report
y_true = [0, 1, 2, 2, 2]
y_pred = [0, 0, 2, 2, 1]
target_names = ['class 0', 'class 1', 'class 2']
print(classification_report(y_true, y_pred, target_names=target_names))
```

```
              precision    recall  f1-score   support
<BLANKLINE>
     class 0       0.50      1.00      0.67         1
     class 1       0.00      0.00      0.00         1
     class 2       1.00      0.67      0.80         3
<BLANKLINE>
    accuracy                           0.60         5
   macro avg       0.50      0.56      0.49         5
weighted avg       0.70      0.60      0.61         5
<BLANKLINE>
```

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

scikit learn’s confusion_matrix

- for scikit-learn:
  - rows = actual
  - columns = predicted

```
>>> from sklearn.metrics import confusion_matrix
>>> y_true = [2, 0, 2, 2, 0, 1]
>>> y_pred = [0, 0, 2, 2, 0, 2]
>>> confusion_matrix(y_true, y_pred)
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

tp1	fp1	fn1	tp2	fp2	fn2	tp3	fp3	fn3

Example

- precision = **tp / (tp + fp)**
- recall = **tp / (tp + fn)**

pred\actual	**class 0**	**class 1**	**class 2**
class 0	5	1	2
class 1	0	4	1
class 2	1	1	7
	class 0	**class 1**	**class 2**
examples	6	6	10

Dealing with imbalanced data

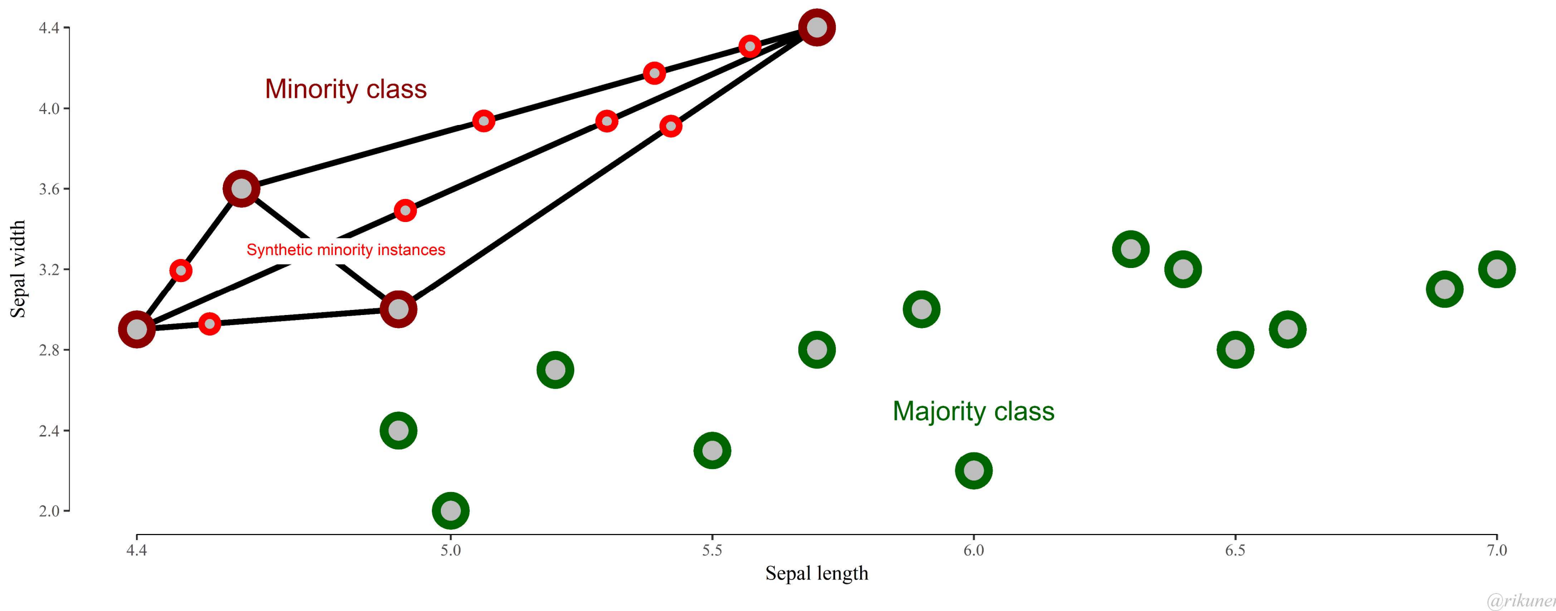
Dealing with imbalanced data

- Data
  - under-sampling (discard examples from larger class)
  - over-sampling (generate more examples for the smaller class)
- Model
  - decision trees e.g. c.45, xgboost (decision trees tend to do well on imbalance data, because we partition data anyway)
- Loss
  - weighted cross entropy (put more weight for smaller class)
- Hierarchical Models
  - combine several small classes into one, then use multiple classifiers

Over sampling

- SMOTE: Synthetic Minority Over-sampling Technique

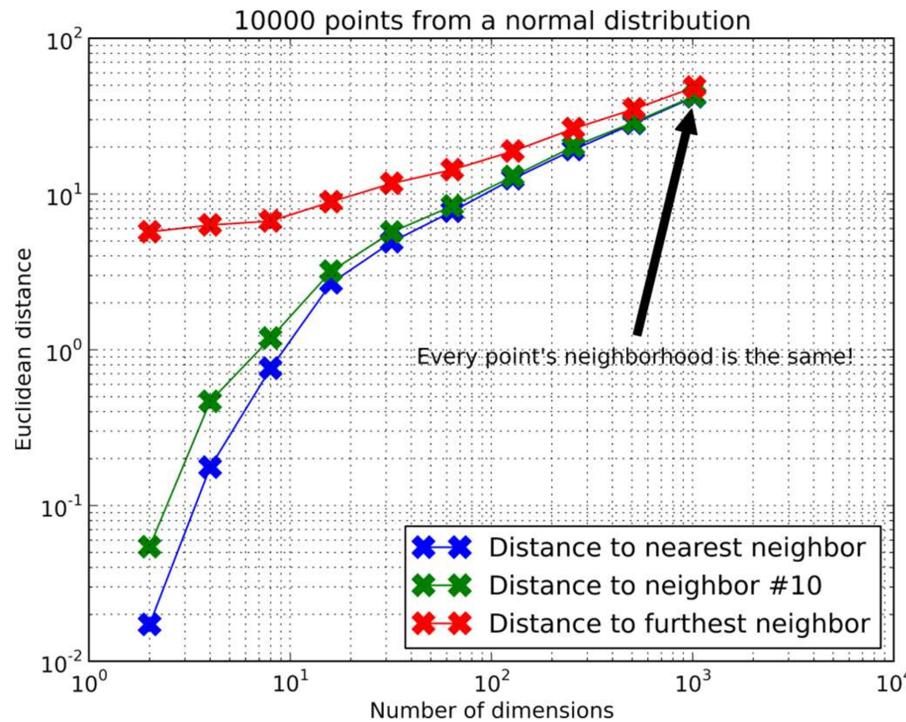
Addressing class imbalance problems of ML via SMOTE: synthesising new dots between existing dots



Chawla, Nitesh V., et al. "SMOTE: synthetic minority over-sampling technique." *Journal of artificial intelligence research* 16 (2002): 321-357.

Curse of dimensionality

- SMOTE does not scale to large feature space
- NLP has large feature space ☹
- E.g Fasttext embedding dims = 300
- Also, to use SMOTE, we have pre-embed everything
- Not practical for 10GB+ dataset



https://erikbern.com/2015/10/20/nearest-neighbors-and-vector-models-epilogue-curse-of-dimensionality.html

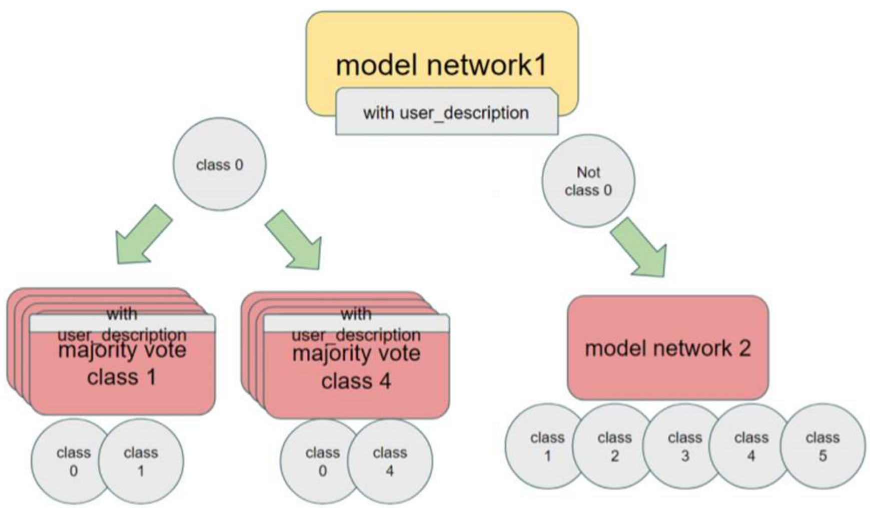
Weighted cross-entropy

$$L = -(y\log(p) + (1-y)log(1-p))$$

$$L = -(y\log(p)w + (1-y)log(1-p))$$

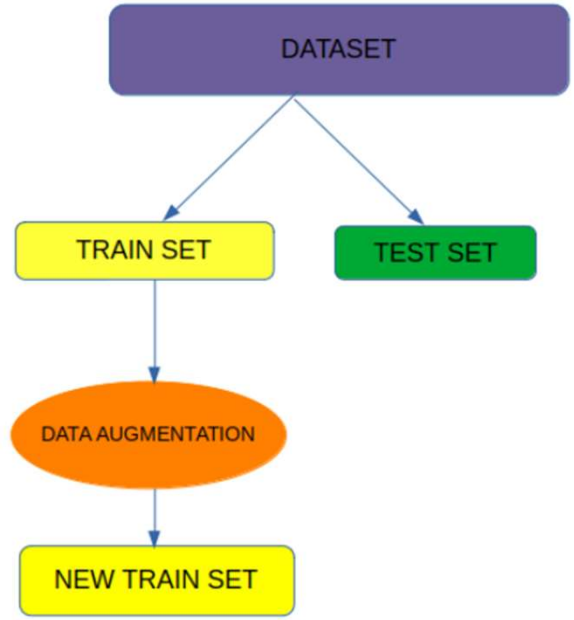
Hierarchical Models

- Group the 5 “not normal” classes together
- Train normal vs. not normal classifier
- Then another classifier just for the “not normal” classes



Over-sampling for NLP

- "Data augmentation"
- Popularized by image data
  - crop, rotate, add noise, zoom, color transforms, flip.
  - a dog upside down with some noise added is still a dog
- But how about text? A backward sentence doesn't make sense.



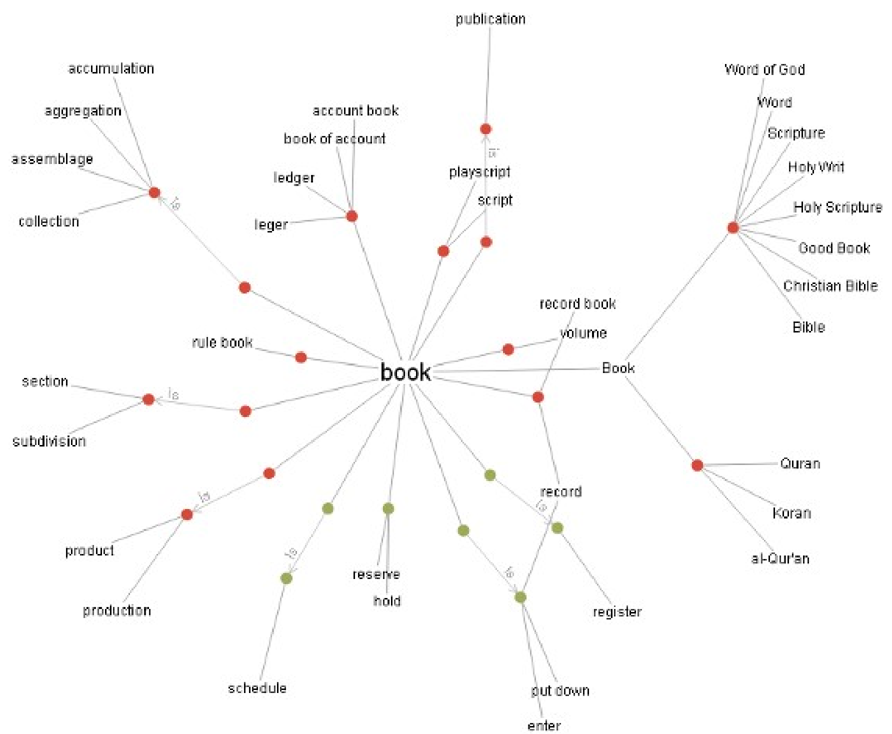
https://neptune.ai/blog/data-augmentation-nlp

NLP data augmentation

- EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks
- Back translation
- Synonym replacement (synonyms from wordnet https://wordnet.princeton.edu/)
- Random insertion
- Random swap
- Random deletion
- https://github.com/jasonwei20/eda_nlp

Wei, Jason, and Kai Zou. "Eda: Easy data augmentation techniques for boosting performance on text classification tasks." *arXiv preprint arXiv:1901.11196* (2019).

Wordnet

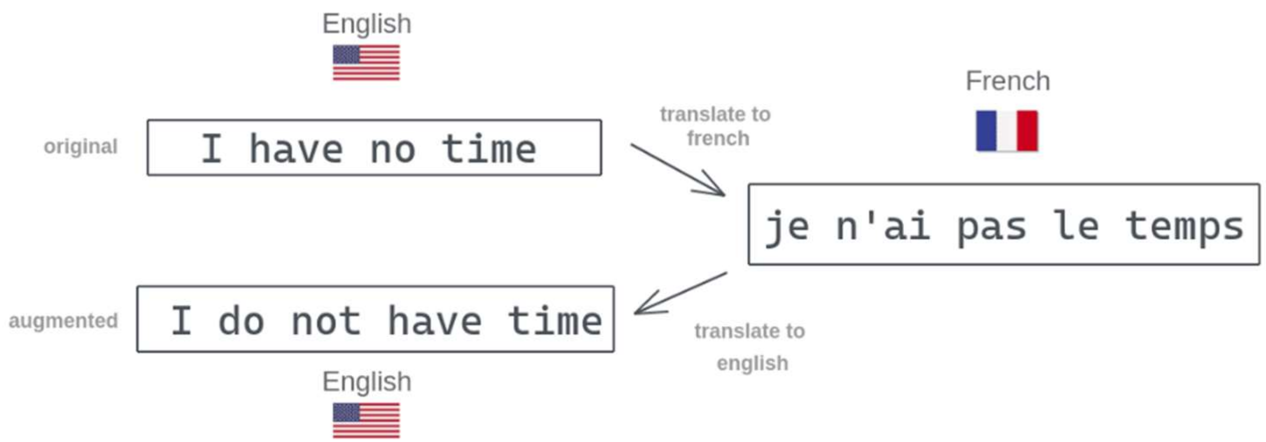


publication
accumulation
aggregation
assemblage
collection
account book
book of account
ledger
leger
playscript
script
Word of God
Word
Scripture
Holy Writ
Holy Scripture
Good Book
Christian Bible
Bible
record book
volume
rule book
book
Book
section
subdivision
Quran
Koran
al-Qur'an
record
product
production
reserve
hold
register
schedule
put down
enter

Thai wordnet

- https://python3.wannaphong.com/2017/02/wordnet-python.html
- https://sourceforge.net/p/thwnsqlite/code/ci/default/tree/LICENSE_THA_WN
- https://sourceforge.net/projects/thwnsqlite/files/thwnsqlite-201405121006.tar.bz2/download

Back translation



Synonym replacement

- Randomly choose n words from the sentence that are not stop words. Replace each of these words with one of its synonyms chosen at random.

> *This* ***article*** *will focus on summarizing data augmentation* ***techniques*** *in NLP.*

The method randomly selects n words (say two), the words *article* and *techniques*, and replaces them with *write-up* and *methods* respectively.

> *This* ***write-up*** *will focus on summarizing data augmentation* ***methods*** *in NLP.*

Random swap

- Randomly choose two words in the sentence and swap their positions. Do this n times.

*This **article** will focus on summarizing data augmentation **techniques** in NLP.*

The method randomly selects n words (say two), the words *article* and *techniques* and swaps them to create a new sentence.

*This **techniques** will focus on summarizing data augmentation **article** in NLP.*

Random deletion

- Randomly remove each word in the sentence with probability p.

> *This **article** will focus on summarizing data augmentation **techniques** in NLP.*

The method selects n words (say two), the words *will* and *techniques*, and removes them from the sentence.

> *This **article** focus on summarizing data augmentation in NLP.*

Random insertion

- Find a random synonym of a random word in the sentence that is not a stop word. Insert that synonym into a random position in the sentence. Do this n times.

*This **article** will focus on summarizing data augmentation **techniques** in NLP.*

*This article will focus on **write-up** summarizing data augmentation techniques in NLP **methods**.*

Fasttext

fasttext paper

- Enriching word vectors with subword information

- use series of binary classifications instead of 1 softmax (when V is large softmax can be a problem)
- use negative samples
- consider sub-words

Bojanowski, Piotr, et al. "Enriching word vectors with subword information." *Transactions of the Association for Computational Linguistics* 5 (2017): 135-146.

Loss function

$$\log\left(1 + e^{-s(w_t,\, w_c)}\right) + \sum_{n \in \mathcal{N}_{t,c}} \log\left(1 + e^{s(w_t,\, n)}\right),$$

- wt = target
- wc = context
- n = negative samples

set of all negative samples
(not in the context)

put negative weights for negative samples

subwords

- sub-words helps for words such as:
  - industry, industrialization
  - economy, economic
- word2vec consider them as completely separate words
- fasttext consider them to come from the same base word
- n is sub-word size

word *where* and $n = 3$ as an example, it will be represented by the character n-grams:

```
<wh, whe, her, ere, re>
```

and the special sequence

```
<where>.
```

Training Fasttext

```
>>> import fasttext
>>> model = fasttext.train_unsupervised('data/fil9')
```

- data must be:
  - text file
  - each sentence is one line
  - each word separated by space (tokenized)

```
[ ]  1 !head thwiki-tokenized-small.txt
```

```
หน้าหลัก
 วิกิพีเดีย ดำเนินการ โดย มูลนิธิวิกิมีเดีย   องค์กร ไม่ แสวง ผล กำไร   ผู้ ดำเนินการ อีก หลาย
 ได้แก่
 __NO EDITSECTION__
 ดาราศาสตร์
 ดาราศาสตร์   คือ วิชาวิทยาศาสตร์ ที่ ศึกษา วัตถุ ท้อง ฟ้า   ( อาทิ   ดาวฤกษ์   ดาวเคราะห์
 ดาวหาง   และ ดาราจักร )   รวม ทั้ง ปรากฏการณ์ ทาง ธรรมชาติ ต่าง   ๆ   ที่ เกิด ขึ้น จาก นอก ชั้น บรรยากาศ ของ โลก
 โดย ศึกษา เกี่ยว กับ วิวัฒนาการ   ลักษณะ ทาง กายภาพ   ทาง เคมี   ทาง อุตุนิยมวิทยา   และ การ เคลื่อน ที่ ของ วัตถุ ท้องฟ้า
 ตลอดจน ถึง การ กำเนิด และ วิวัฒนาการ ของ เอกภพ
 ดาราศาสตร์ เป็น หนึ่ง ใน สาขา ของ วิทยาศาสตร์ ที่ เก่าแก่ ที่สุด   นัก ดาราศาสตร์ ใน วัฒนธรรม โบราณ สังเกต การณ์ ดวงดาว บน ท้อง ฟ้า ใน เวลา กลาง คืน
```

Example - Wongnai (fasttext)

Assignment

- Do text classification on this data:
  https://github.com/PyThaiNLP/wisesight-sentiment
- Can use either LSTM or Fasttext