1. Working with text

2. Working with unbalanced datasets

3. Hyperparameter tuning

4. If we have time, we will briefly discuss strategy

Feature Extraction:

The house plans to vote on the Senate’s bipartisan bill but is scrapping an earlier plan to put Democrats’ social policy bill first.

Convert text into a feature vector.

Documents as feature vectors

...use leader said they

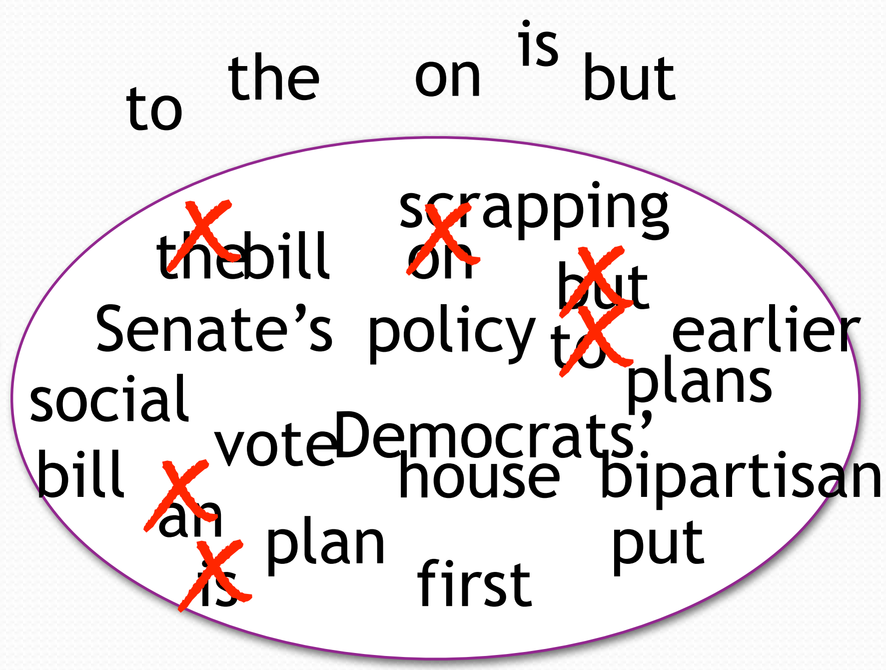
The actress is drawing Oscar talk

The house plans to vote on the Senate's bipartisan bill but is scrapping an earlier plan to put Democrats' social policy bill first.

(NYTimes Nov. 5, 2021)

$\mathbf{x}$

Bag of words



$$\begin{array}{cl} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 0 \\ 0 \\ 1 \\ \vdots \end{bmatrix} & \begin{array}{l} \text{politics} \\ \text{house} \\ \text{bill} \\ \text{president} \\ \text{leader} \\ \text{social} \\ \vdots \end{array} \end{array}$$

$\Phi(\mathbf{x})$

Approach from mit.edu/6.034

Transform the feature vectors to emphasize more “relevant” words

Approach from mit.edu/6.034

Turning text into a feature vector

- Document is natively text
- Must represent as a numeric vector
- Represent by word counts
  - Enumerate all words
  - Each document is count of frequencies
- Stopwords

Discussion Questions

- Is the absolute number of times a word appears the correct metric?
- What about the length of the document?
- What about the frequency of the word?
- What words "matter"?

the, for, a, in

convolutional, gradient

- Perhaps:
  - if a word appears frequently, it is important (give it a high score)
  - If a word appears in many documents, it is not important (give it a low score)

Ideas:

$$\text{TF}_{\text{"this"},d_1} = \frac{1}{5} = 0.2 \qquad \text{IDF}_{\text{"this"}} = \log\left(\frac{2}{2}\right) = 0$$
$$\text{TF}_{\text{"this"},d_2} = \frac{1}{7} \approx 0.14$$

$$\text{TF}_{\text{"example"},d_1} = \frac{0}{5} = 0 \qquad \text{IDF}_{\text{"example"}} = \log\left(\frac{2}{1}\right) = 1$$
$$\text{TF}_{\text{"example"},d_2} = \frac{3}{7} \approx 0.429$$

Example modified from https://en.wikipedia.org/wiki/Tf%E2%80%93idf

- How can we categorize how important a word is in a document?
- Perhaps:
  - if a word appears frequently, it is important (give it a high score)
  - except if the word appears in many documents, it is not important (give it a low score)
- Steps:
  - Count the frequency of every word in the document
  - Determine how much information a word provides: Inverse Document Frequency (IDF)

convolutional, gradie

the, for, a, in

Term frequency

$$TF_{i,n} = \frac{\text{num times word } i \text{ in doc } n}{\text{total num words in doc } n}$$

The more common a word is the lower its IDF score

Inverse doc frequency

$$IDF_i = \log\left[\frac{\text{Total num docs in corpus}}{\text{Num docs with word } i}\right]$$

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

Ideas:

$$\text{TF}_{\text{"this"},d_1} = \frac{1}{5} = 0.2 \qquad \text{TF}_{\text{"this"},d_2} = \frac{1}{7} \approx 0.14 \qquad \text{IDF}_{\text{"this"}} = \log\left(\frac{2}{2}\right) = 0$$

$$\text{TF}_{\text{"example"},d_1} = \frac{0}{5} = 0 \qquad \text{TF}_{\text{"example"},d_2} = \frac{3}{7} \approx 0.429 \qquad \text{IDF}_{\text{"example"}} = \log\left(\frac{2}{1}\right) = 1$$

Example modified from https://en.wikipedia.org/wiki/Tf%E2%80%93idf

Frequency is relative to the size of the document

- How can we categorize how [illegible] ord is in a document?
- Perhaps:
  - if a word appears frequently, it is important (give it a high score)
  - except if the word appears in many documents, it is not important (give it a low score)
- Steps:
  - Count the frequency of every word in the document
  - Determine how much information a word provides: Inverse Document Frequency (IDF)

convolutional, gradie

the, for, a, in

Term frequency

$$TF_{i,n} = \frac{\text{num times word } i \text{ in doc } n}{\text{total num words in doc } n}$$

The more common a word is the lower its IDF score

Inverse doc frequency

$$IDF_i = \log\left[\frac{\text{Total num docs in corpus}}{\text{Num docs with word } i}\right]$$

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

Term Frequency – Inverse Document Frequency

- Use TF-IDF weight for vectors:

$$X[n,i] = TF_{i,n} \times IDF_i$$

Document weight vector

Term frequency

$$TF_{i,n} = \frac{\text{num times word } i \text{ in doc } n}{\text{total num words in doc } n}$$

Inverse doc frequency

$$IDF_i = \log\left[\frac{\text{Total num docs in corpus}}{\text{Num docs with word } i}\right]$$

$$\text{TF}_{\text{"this"},d_1} = \frac{1}{5} = 0.2$$

$$\text{TF}_{\text{"this"},d_2} = \frac{1}{7} \approx 0.14$$

$$\text{IDF}_{\text{"this"}} = \log\left(\frac{2}{2}\right) = 0$$

$$\text{TF}_{\text{"example"},d_1} = \frac{0}{5} = 0$$

$$\text{TF}_{\text{"example"},d_2} = \frac{3}{7} \approx 0.429$$

$$\text{IDF}_{\text{"example"}} = \log\left(\frac{2}{1}\right) = 1$$

$$\text{TF-IDF}_{\text{"this"},d_1} = 0.2 \times 0 = 0$$

$$\text{TF-IDF}_{\text{"this"},d_2} = 0.14 \times 0 = 0$$

$$\text{TF-IDF}_{\text{"example"},d_1,D} = 0. \times 1 = 0$$

$$\text{TF-IDF}_{\text{"example"},d_2} = 0.429 \times 1 = 0.429$$

Example modified from https://en.wikipedia.org/wiki/Tf%E2%80%93idf

Term Frequency – Inverse Document Frequency

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

Term	TF-IDF Example 1	TF-IDF Example 2
this	0	0
is	0	0
a	0.4	0
sample	0.2	0
example	0	0.429
another	0	0.286

$$\text{TF-IDF}_{\text{"is"},d_1} = 0.2 \times 0 = 0$$

$$\text{TF-IDF}_{\text{"is"},d_2} = 0.143 \times 0 = 0$$

$$\text{TF-IDF}_{\text{"a"},d_1} = 0.4 \times 1 = 0.4$$

$$\text{TF-IDF}_{\text{"a"},d_2} = 0 \times 1 = 0$$

$$\text{TF-IDF}_{\text{"sample"},d_1} = 0.2 \times 1 = 0.2$$

$$\text{TF-IDF}_{\text{"sample"},d_2} = 0 \times 1 = 0$$

$$\text{TF-IDF}_{\text{"another"},d_1} = 0.286 \times 1 = 0.286$$

$$\text{TF-IDF}_{\text{"another"},d_2} = 0 \times 1 = 0$$

This is a very small example. Obviously we would usually compute the TF-IDF for a large collection of documents

Example modified from https://en.wikipedia.org/wiki/Tf%E2%80%93idf

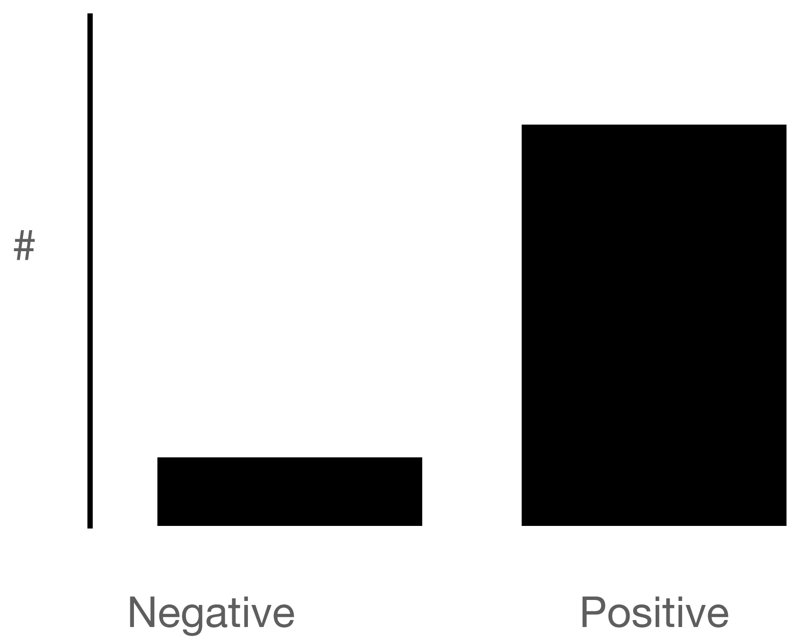
Learn more at:

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

1. Working with text
2. Working with unbalanced datasets
3. Hyperparameter tuning
4. If we have time, we will briefly discuss strategy

Imbalanced Data

Asymmetric data - Potentially not do well on the minority class



Mild: 20-40%
Moderate: 1-20%
Extreme: <1%

https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data

https://www3.nd.edu/~dial/publications/hoens2013imbalanced.pdf
https://neptune.ai/blog/how-to-deal-with-imbalanced-classification-and-regression-data

Imbalanced Data

Basic Approaches

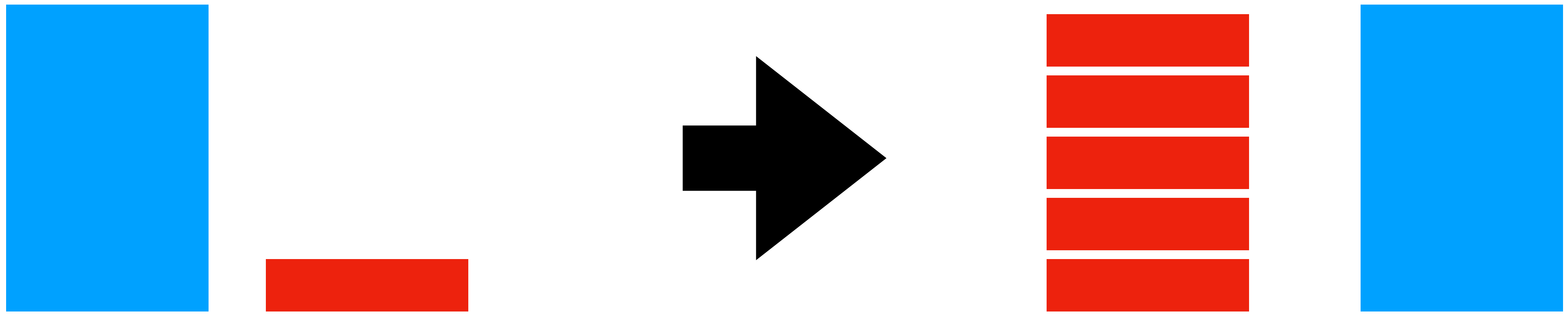
- Do nothing. See if your hypothesis works.
- Under-sampling: Reduce the number of the majority class
- Over-sampling: Increase the number in the minority class
- Weighted Learners. Create a weighted objective function.

Oversampling

Increase the number of examples in the minority class

- Duplicate minority class examples
- Generate new synthetic data
    - e.g. SMOTE, ADASYN

Advantage: use all the original data



https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/

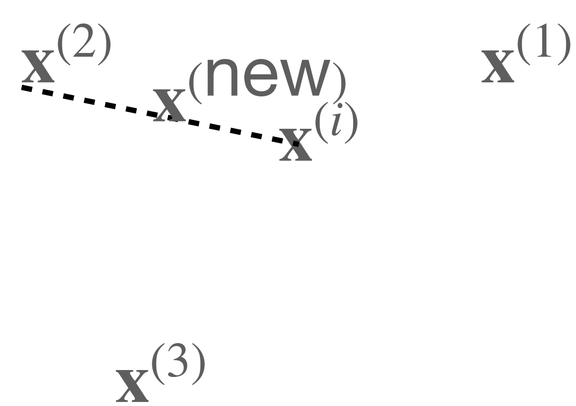
SMOTE

Create synthetic minority class training examples. There are many variations on the following basic algorithm (including ADASYN).

Creates new examples by:

- Choosing a random example $\mathbf{x}^{(i)}$ from the minority class and finding the k closest minority training examples to it

- Randomly selects $\mathbf{x}^{(j)}$ one of the k closest minority examples and creates a new example: $\mathbf{x}^{(\text{new})} = \mathbf{x}^{(i)} + \lambda(\mathbf{x}^{(j)} - \mathbf{x}^{(i)})$ where λ is randomly chosen in the range $[0,1]$

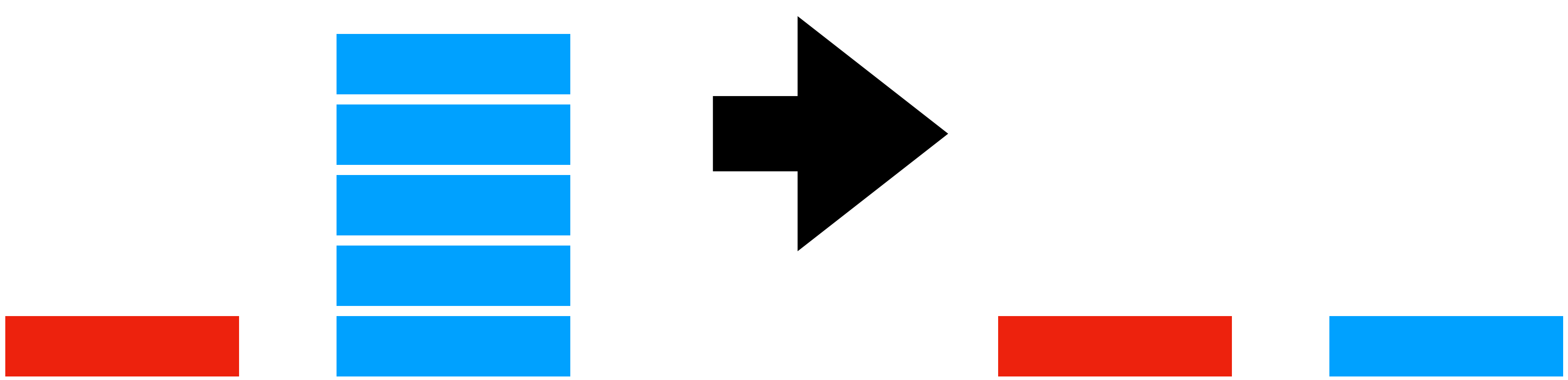


Learn more: https://www.kaggle.com/code/residentmario/oversampling-with-smote-and-adasyn/notebook

Undersampling

If there are m minority-class elements, use only m of the majority-class examples. Some ways to select the majority-class examples

- Randomly
- Selectively. The goal is to remove borderline examples and redundant noisy examples. e.g., Tomek links and condensed nearest neighbor



https://www.baeldung.com/cs/ml-stratified-sampling

Weighted Learners

Popular technique!!

- Incorporate different penalties for different classes in objective function
- Higher weight for the minority class, smaller weight for the minority class

Logistic regression

$$\ell(\mathbf{w}) = \sum_{i=1}^{N} \left[\text{weight}_1 y^{(i)} \ln \sigma(\mathbf{w}^T \mathbf{x}^{(i)}) + \text{weight}_0 (1 - y^{(i)}) \ln \left(1 - \sigma(\mathbf{w}^T \mathbf{x}^{(i)})\right) \right]$$

Support vector machine

$$\min_{w_0, \mathbf{w}, \{\xi^{(i)}\}_{i=1}^N} \|\mathbf{w}\|_2^2 + \text{weight}_1 \, C \sum_{j: y^{(j)} = +1} \xi^{(j)} + \text{weight}_0 \, C \sum_{j: y^{(j)} = -1} \xi^{(j)}$$

$$\text{subject to } y^{(i)}(w_0 + \mathbf{w}^T \mathbf{x}^{(i)}) \geq 1 - \xi^{(i)} \text{ for all } i = 1, \ldots, N$$

$$\xi^{(i)} \geq 0$$

Learn more at:

https://chrisalbon.com/code/machine_learning/logistic_regression/handling_imbalanced_classes_in_logistic_regression/

https://chrisalbon.com/code/machine_learning/support_vector_machines/imbalanced_classes_in_svm/