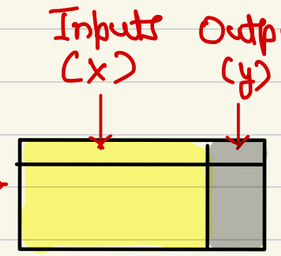
INTRODUCTION TO NLP

By: Kanav Bansal
(ThatAIGuy)

MODEL BUILDING PIPELINE

By: Kanav Bansal
(That AI Guy)

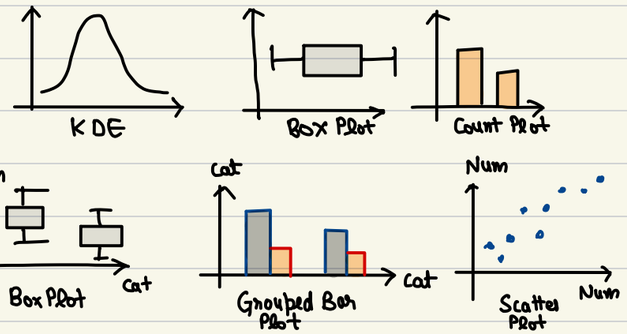
1. Load the data



3. Segregate Inputs & Outputs

* EDA *

2. Exploratory Data Analysis



Analysis

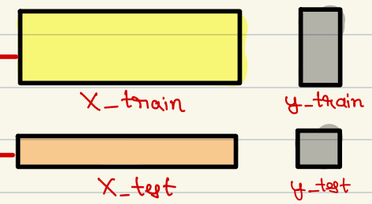
Objective:
- Identify the data distribution
- Outliers & Missing Values
- Relationships b/w inputs & outputs

* Data Preparation *

4. Split the data into train & test

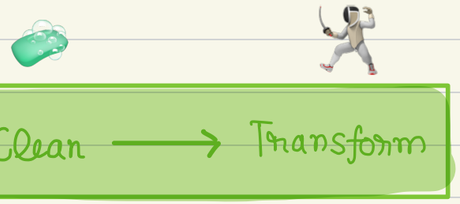
5. Apply Data Preparation on train data (i.e. X_train)

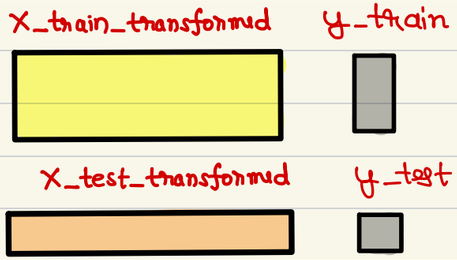
Apply fit_transform()



Apply only transform()

6. Apply Data Preparation on test data (i.e. X_test)



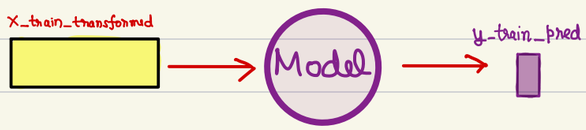


* Learning Phase *

7. Build the Model

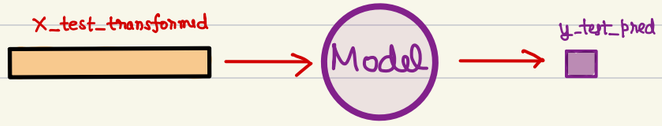
* Testing Phase *

8. Predict on Test data

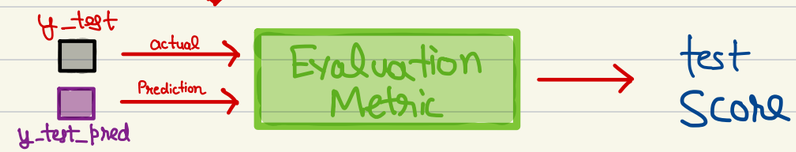


9. Evaluation

10. Predict on Test data



11. Evaluation



What is NLP?

By: Kanav Bansal
(That AI Guy)

NLP is the branch of AI that involves computers to understand, interpret & generate human language.
There are wide range of tasks in NLP like:

1. Text Classification → Spam Detection, Sentiment Analysis, Adult Content filtering, etc...
   In text classification, the goal is to classify the entire piece of text into categories.

2. Information Extraction → Named Entity Recognition, Part of Speech Tagging
   NER is a Sequence Classification task. Here the goal will be to classify each element in a sequence into categories. For eg:

Ex: Apple (ORG) is looking to launch M4 Macbooks (PRODUCT) for 1500$ (PRICE).

Ex: She (Pronoun) runs (verb) fast (Adverb).
- She: Refers to a female subject
- runs: Indicates action performed
- fast: Describes how she runs

The goal of information extraction is to extract structured information from an unstructured data (like NER from a news article)

3. Information Retrieval → Search & Retrieval
   Information Retrieval is a process of finding relevant information within a large repository of data, such as documents, websites, etc...
   Goal is to retrieve documents or information that best matches a user's query.

4. Text Summarization → News feed Creation, Case Report Summarization in a big law firm

5. Machine Translation → Translating one language to another

6. Question & Answering → Automated systems to answer customers queries, Providing Weather updates, Healthcare Chatbot that can provide information about diseases, etc...

Question Processing → Information Retrieval → Information Extraction → Answer Generation

Why Is NLP Hard?

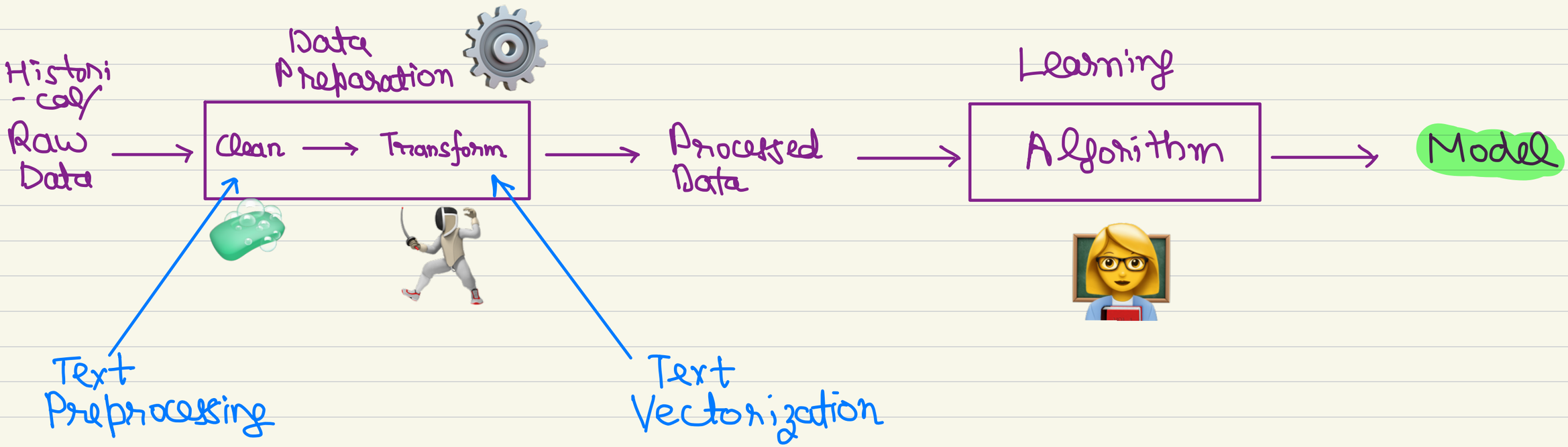
By: Kanav Bansal
(That AI Guy)

1. Ambiguity → i.e uncertainty of meaning
   Ex: The car hit the pole while it was moving
   Ex: The chicken is ready to eat.

2. Complexity of representation → Like Poems, Sarcasm, Phrases, etc...
   Ex: You have a football game tomorrow. Break a leg!
   Ex: Yeah, right, because that worked so well last time.

Data Preparation for Text

By: Kanav Bansal
(That AI Guy)



(To clean the noise from text data)

(To convert Text data to numerical representation)

By: Kanav Bansal
(That AI Guy)

Terminologies I

@ Document → It is a single piece of text. It can be a sentence, para, article, review, email body etc..

ⓑ Corpus → Corpus is a collection of documents

© Vocabulary → Set of all unique words that appear in a corpus.

ⓓ Vectorization → It is a technique to convert text data into numerical vectors.

ⓔ Document Vectors → Numerical Representation of a document

ⓕ Document Term Matrix → A DTM is a matrix where rows represent document vectors & columns represent terms from the vocabulary.

Text Vectorization Techniques (AKA Feature Extraction)

@ Bag of Words

ⓑ Term Frequency Inverse Document Frequency

© Word2Vec

ⓓ GloVe

ⓔ FastText

ⓕ ELMo

ⓖ LLM's → GPT, BERT

Text Cleaning (AKA Text Preprocessing)

@ Removing Special Characters

ⓑ Converting to lower case

© Removing Stop words

ⓓ Converting to root form

By: Kanav Bansal
(That AI Guy)

Text Vectorization → Bag of Word

Converts each document into a vector, where each element of vector is the count of a term in the document.

Corpus

Doc-1: we are learning machine learning
Doc-2: processing natural language data
Doc-3: machine learning algorithms

Step 1: Build the vocabulary (fit)

i.e. Learn the unique words in the corpus

vocabulary = {'algorithms', 'are', 'data', 'language', 'learning', 'machine', 'natural', 'processing', 'we'}

↑ Represented in sorted order

size of vocabulary = 9

Step 2: Apply transformation (aka vectorization)

For each document, count the number of times each vocabulary term appears in that document.

Doc Vector

	'algorithms'	'are'	'data'	'language'	'learning'	'machine'	'natural'	'processing'	'we'
DV-1:	0	1	0	0	2	1	0	0	1
DV-2:	0	0	1	1	0	0	1	1	0
DV-3:	1	0	0	0	1	1	0	0	0

n * d

Shape of DTM = n * d

n → # of documents

d → Size of vocabulary

Problems / Disadvantages

1. Dimensionality & Sparsity of DTM

Ques: What is a sparse Matrix?

Ans: A matrix where most entries are zeroes.

# of entries in matrix (dtm) = n*d = 3*9 = 27

Average no. of words per document = 4 words/doc

# of non-zero entries in dtm = 4 * n = 4*3 = 12

i.e Almost 50% of dtm is filled with zeroes

Ques: Why is it a problem?

Ans: Visualization, Memory Consumption, Computational Complexity, Curse of dimensionality and Interpretability.

Ques: Solution?

Ans: Remove Stop Words & Convert to root form

↳ Problem: Even after this dimensionality remains very large.

2. Sequence Info is lost

Ques: Solution?

Ans: n-grams approach

↳ Problem: Dim & Sparsity problem with this approach.

3. Out of vocabulary words can't be handled.

4. Semantics (or meaning) of words is lost. Word Similarity is not captured.

By: Kanav Bansal
(That AI Guy)

Text Vectorization → TF IDF

Converts each document into a vector, where each element of vector is the TFIDF score of a term in the document.

Corpus

Doc-1: we are learning machine learning
Doc-2: processing natural language data
Doc-3: machine learning algorithms

Step 1: Build the vocabulary (fit)

Vocabulary = {'algorithms', 'are', 'data', 'language', 'learning', 'machine', 'natural', 'processing', 'we'}

Step 2: Apply transformation (aka vectorization)

For each document, calculate the TFIDF score for each vocabulary term which appears in that document.

$$\text{TF IDF Score} = \text{TF} * \text{IDF}$$

$$TF(w_i, doc_j) = \frac{\text{No. of times } w_i \text{ occurs in } doc_j}{\text{Total no. of words in } doc_j}$$

$$IDF(w_i, corpus) = \log\left(\frac{\text{No. of docs in corpus}}{\text{No. of docs containing } w_i}\right) + 1$$

Note:

* If a word is more frequent in a document, its TF will be more.
* If a word is rare in a document, its IDF will be more.

Let's see the TFIDF score for 'learning' in Doc-1:

$TF('learning', Doc\text{-}1) = \frac{2}{5}$ $IDF('learning', corpus) = \log\left(\frac{3}{2}\right) + 1$

	'algorithms'	'are'	'data'	'language'	'learning'	'machine'	'natural'	'processing'	'we'
DV1	[0	–	0	0	$\frac{2}{5}*\{\log(\frac{3}{2})+1\}$	–	0	0	–]
DV2	[0	0	–	–	0	0	–	–	0]
DV3	[–	0	0	0	–	–	0	0	0]

Let's now compute the TF IDF score for a word which was never present in the corpus.

Doc-4: cleaning natural language

$TF('cleaning', Doc\text{-}4) = \frac{1}{3}$ $IDF('cleaning', corpus) = \log\left(\frac{3}{0}\right) + 1$

Zero Divison Error ↑

IDF Formula with Smoothing I

$$IDF(w_i, corpus) = \log\left(\frac{N+1}{DF(w_i)+1}\right) + 1$$

Significance:

Incorporating smoothing into the IDF calculation ensures that new or unknown terms, which were not present in the original corpus, are handled appropriately.

Avoids zero division error for terms/words which are not present in any document.

What's Next?

By: Kanav Bansal
(That AI Guy)

A Big Question I

Let's assume there are following vocabulary words in the corpus:

{'good', 'great', 'bad',
'wonderful', 'poor'}

Numerical Representation for each word I

	'bad'	'good'	'great'	'poor'	'wonderful'
good	0	1	0	0	0
great	0	0	1	0	0
bad	1	0	0	0	0
wonderful	0	0	0	0	1
poor	0	0	0	1	0

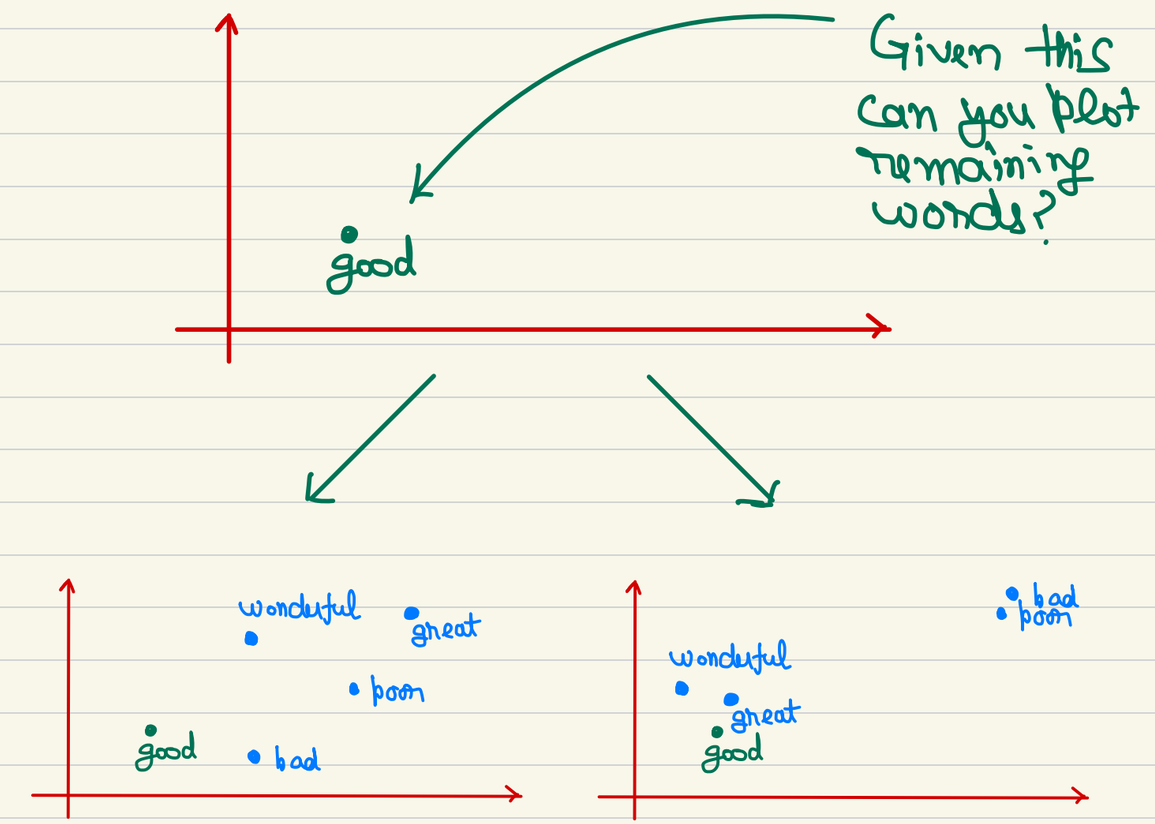
Ques: Is it possible to visualize a scatter plot for above 5 dimensional vectors?

Ans: Not directly. We can reduce the dimensionality of the data & bring it to 2D with the help of Algorithms like PCA.

d-dim vectors → [PCA] → 2 dim vectors

Ques: Assuming that all the vocabulary words can be represented using 2 dimensional vector representation, which plot do you think make more sense?

Scatter plot I



Plot - 1
(BOW & TFIDF)

Plot - 2
(W2V, GloVe, FastText, BERT, GPT, LLM's...)

By: Kanav Bansal
(That AI Guy)

Vectorization Techniques

Problems	BOW & TFIDF	W2V & Glove	GPT, BERT
1. Dimensionality & Sparsity	High	Low (AKA Embeddings)	Low (AKA Embeddings)
2. Semantic	✗	✓	✓
3. Sequence info	✗	✗	✓
4. OOV	✗	✗	✓