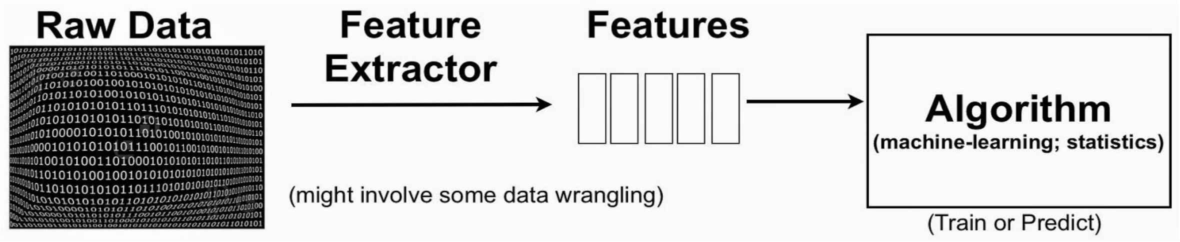
Text Feature Engineering

About the Module

- What is Feature Engineering?
- Text Feature Engineering
- Meta Text Feature Engineering
- NLP Attributes Feature Engineering
- Term Frequency and Inverse Document Frequency (TF-IDF)
- Word Embeddings

What is Feature Engineering?

- Transformation of dataset to create model inputs



- Process of deriving new features from the existing ones

Raw Features	Engineered Features
Date : 27 / 09 / 2018	Day : Thursday , Month-Date : 27 , Year : 2018 , Month : September
Text : Natural Language Processing	NumWords : 3, NumChars : 25, NumVowels : 10

Analytics Vidhya
Learn everything about analytics

Text Feature Engineering

- (Almost all) Machine Learning Algorithms cannot accept text as input
- Information present in the text data needs to be quantified into features / predictors
- **Text Feature Engineering :** Convert text to features
- Use Cases: Information Retrieval, Search Engines, Machine Learning problems

Text Feature Engineering Ideas

- Meta Attributes of Text – Words, Characters
- NLP Attributes of Text – Pos Tags, Grammar Relations
- Statistical Features – Word Frequencies / Interaction Features
- Word Vector Notations

Meta Text Features

Count / Length Features

- Sentence Counts - Number of sentences in the document
- Word Counts - Number of words in the document
- Upper Case Counts - Number of words having upper casing
- Proper Case Counts - Number of words having proper casing

Special Symbol / Entities Counts

- Character Counts - Number of characters in the document
- Punctuation Counts - Number of punctuation marks in the document
- Stop word Counts - Number of stop word keywords in the document
- Specific Category Counts - Number of domain specific words in the document

Misc features

- Interaction Features : Word Density, Character Density
- Number of spelling errors
- Number of keyword variations

Meta Text Features

Document:
"""this course is teaching us nlp. NLP is a field of data science. NLP means natural langauge processing"""

Extracted Features:

- Sentence Count : 3
- Word Count : 18
- Word Density : 6
- Character Count : 84
- Stopword Count : 6

Document to Features

- Document --> [3, 18, 6, 84, 6]

NLP Based Features

Part of speech features

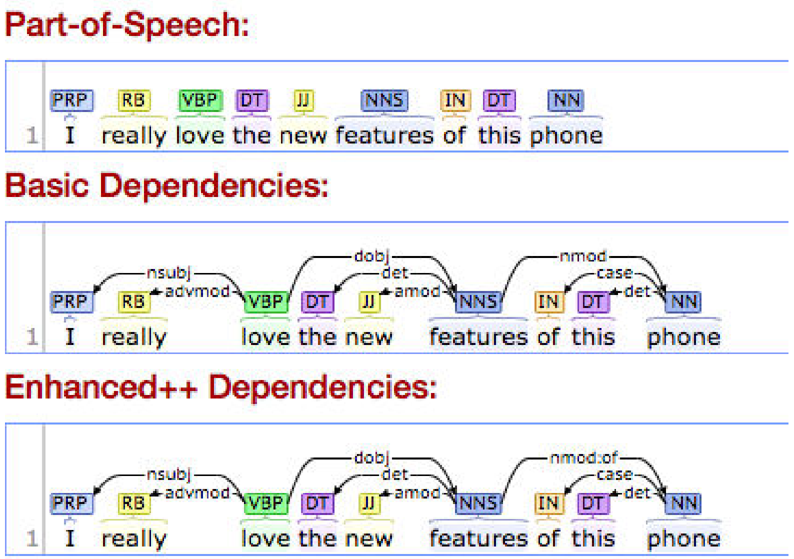
- Number of Proper Nouns
- Number of noun family words
- Number of verb family words
- Number of adjectives / adverbs

Phrases

- Number of noun phrases
- Number of verb phrases

Grammar Relations

- Subjects / objects present in the sentence
- Head word / leaf word



Document Term Matrix : Count Vectorization

Document Term Matrix

	intelligent	applications	creates	business	processes	bots	are	i	do	intelligence
Doc 1	2	1	1	1	1	0	0	0	0	0
Doc 2	1	1	0	0	0	1	1	0	0	0
Doc 3	0	0	0	1	0	0	0	1	1	1

Count Vectorization :
Rows : Document ; Columns : Words
Value : Count of Word in the Document

Some Problems :
Stopwords will have higher frequency
Important / Relevant terms will have lower frequency

Term Frequency and Inverse Document Frequency

TF: Term Frequency measures how frequently a term occurs in a document.

IDF: Inverse Document Frequency measures how many documents comprises of a specific term

- Term Frequency

$$\mathrm{tf}_{\mathrm{i,j}} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

- Inverse Document Frequency

$$\mathrm{idf}_{\mathrm{i}} = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

TF-IDF Score

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF

Term x within document y

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

df_x = number of documents containing x

N = total number of documents

- TF IDF score measures the relative importance of every word in the corpus.
- The score is used to generate the numerical representation of words in the corpus

TF-IDF Score : Example

Corpus : 10,000 documents

Terms : Delhi, Mumbai, Chennai

Document Frequency :
Delhi = 50, Mumbai = 1300, Chennai = 250

Given a document containing terms with given frequencies:
Delhi = 3; Mumbai= 2; Chennai = 1

THEN

Delhi: tf = 3/3; idf = log(10000/50) = 5.3; tf-idf = 5.3
Mumbai: tf = 2/3; idf = log(10000/1300) = 2.0; tf-idf = 1.3
Chennai: tf = 1/3; idf = log(10000/250) = 3.7; tf-idf = 1.2

Analytics Vidhya
Learn everything about analytics

TF-IDF Score

Improvement by TFIDF Score

- TF IDF score is high for the terms which are present frequently in a document but are not present in most of the other documents. Ie. Rare terms in the corpus.

- TF IDF score is lower for terms which are occurring frequently in most of the documents. Example – stop words.

Variations of TF IDF

- N-gram Level TF IDF

- Character Level TF IDF

Dealing with Sparsity

- Text features are generally represented in the form of Document Term Matrix
- Document Term Matrix is highly sparse due to large number of words
- Impact's model's performance

Tips to reduce sparsity

Text cleaning :

- Normalization
- Stop words removal

Matrix Decomposition Techniques

- SVD
- LDA
- NNMF

Singular Value Decomposition

- Singular value decomposition is a method of decomposing a matrix into three other matrices:

$$A = USV^T$$

A is an $m \times n$ matrix
U is an $m \times n$ *orthogonal* matrix
S is an $n \times n$ *diagonal matrix*
V is an $n \times n$ orthogonal matrix

- The dimensions of Document Term Matrix are reduced
- SVD finds the optimal projection to a low dimensional space by exploiting co-occurrence patterns
- Words having similar patterns are projected / or collapsed into same dimensions

- Similar to LDA: – Document Term Matrix -> Document Topic Matrix

Word Vector Notations

- Vectors : projection of word into a continuous vector space
- Quantify the vector elements using counts / tfidf scores

	Document 1	Document 2	Document 3	Document 4	Document 5	Document 6	Document 7	Document 8
Term(s) 1	10	0	1	0	0	0	0	2
Term(s) 2	0	2	0	0	0	18	0	2
Term(s) 3	0	0	0	0	0	0	0	2
Term(s) 4	6	0	0	4	6	0	0	0
Term(s) 5	0	0	0	0	0	0	0	2
Term(s) 6	0	0	1	0	0	1	0	0
Term(s) 7	0	1	8	0	0	0	0	0
Term(s) 8	0	0	0	0	0	3	0	0

Word Vector (Passage Vector)

Document Vector

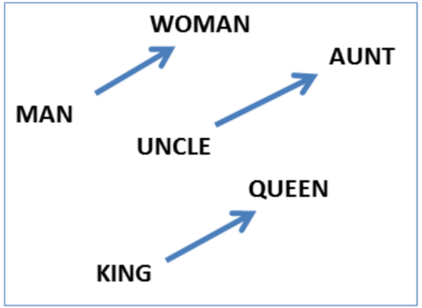
Word Embeddings

Corpus

```
array([[-0.01236233, -0.04655259,  0.00508882, ..., -0.00993368,
         0.01379246,  0.00122126],
       [-0.03087116, -0.02232517,  0.01138248, ..., -0.02389362,
         0.02484551, -0.0087585 ],
       [-0.03504547, -0.04104917,  0.00930308, ..., -0.03002032,
         0.01539359, -0.00338876],
       ...,
       [-0.03802555, -0.017358  ,  0.02445563, ..., -0.0131221 ,
         0.02305542, -0.00747857],
       [-0.02819404, -0.04432267,  0.01159158, ..., -0.02953893,
         0.01612862, -0.0099255 ],
       [-0.0326709 , -0.0484228 ,  0.01606839, ..., -0.03584684,
         0.00761068, -0.00948259]], dtype=float32)
```

Word Vectors

Word Vectors : Context / Meaning + Relationships



Analytics Vidhya
Learn everything about analytics

Word Embeddings

Word Vectors can be obtained using following :

- Training of word embedding representations from scratch

keras.layers.Embedding(input_dim, output_dim, embeddings_initializer='uniform', embeddings_regularizer=**None**, activity_regularizer=**None**, embeddings_constraint=**None**, mask_zero=**False**, input_length=**None**)

- Pretrained Word Embedding Models :

  - Word2vec
  - Glove
  - Fasttext

Word Embeddings

Word2Vec : Combination of two shallow neural network models:

- Continuous bag of words (CBOW)
- Skip-gram model

- Continuous bag of words model is trained to predict the probability of a word given a context, which can be a single word or a group of words. While, Skip-gram model predicts the context given a word.

Example :
<WORD: ???> <Context: ate the food>
<Word: The Dogs> <Context: ???>

FastText : breaks words into several n-grams

- Example : apple : app, ppl, ple
- The word embedding vector for apple will be the sum of all these n-grams.