Text Representation

Luciano Barbosa

Centro de Informática UFPE

Universidade Federal de Pernambuco

Example of Text Classification: Spam

From: "" <takworlld@hotmail.com>
Subject: real estate is the only way... gem oalvgkay

Anyone can buy real estate with no money down

Stop paying rent TODAY !

There is no need to spend hundreds or even thousands for similar courses

I am 22 years old and I have already purchased 6 properties using the methods outlined in this truly INCREDIBLE ebook.

Change your life NOW !

===

Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm

===

Supervised Learning

- Goal: to infer a function from examples to predict classes of new examples
- Two phases:
  - Training: learn the function from examples
  - Execution: use the function to predict the class of a given instance

Supervised Model

- Training set: instances and labels
- Instance represented by its feature vector
- Learn function f(x)=y that best predicts the value of y given x
- For categorical y -> classification
- For numerical y -> regression

	viagra	learning	the	dating	nigeria		*spam?*
$\vec{x}_1 = ($	1	0	1	0	0	)	$y_1 = 1$
$\vec{x}_2 = ($	0	1	1	0	0	)	$y_2 = -1$
$\vec{x}_3 = ($	0	0	0	0	1	)	$y_3 = 1$

Spam Classification

- Training instances

	viagra	learning	the	dating	nigeria	*spam?*
$\vec{x}_1 = ($	1	0	1	0	0)	$y_1 = 1$
$\vec{x}_2 = ($	0	1	1	0	0)	$y_2 = -1$
$\vec{x}_3 = ($	0	0	0	0	1)	$y_3 = 1$

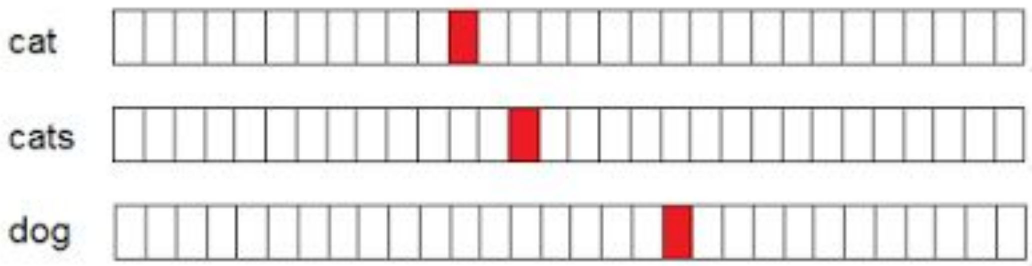
- Features
  - Words: viagra, learning, the, dating, nigeria
  - Occurrence: 1 or 0
- Class y: spam (1) or non-spam (-1)

Features

- Great importance in the classification result
- Important: high correlation with the classification output
  - Ex1: rain forecast: temperature, humidity
  - Ex2: sentiment analysis: words with polarity (negative/positive)
- Text classifiers can use any type of feature: words, punctuation, capitalization, etc.

Feature Representation in Text: One Hot Encoding

- Each word mapped to a dimension and represented by a vector
- Example
  - V = {dog, bites, man}
  - D_1: "dog bites man" = {[1,0,0],[0,1,0],[0,0,1]}
  - D_2: "man bites dog" = {[0,0,1],[0,1,0],[1,0,0]}
- Dimensionality: size of vocabulary
- Naïve embedding
- Similar words have different representations



Img-Source: http://veredshwartz.blogspot.com.br/2016/01/representing-words.html

Feature Representation in Text: Bag of Words

- Each document represented by a vector
- Each dimension: a word with its respective weight
- Dimensionality: size of vocabulary
- Pros: Simple and very effective
- Cons:
  - Orderless
  - No notion of semantic similarity

D_1 : "the cat sat on the hat"
D_2 : "the dog ate the cat and the hat"

V	[the, cat, sat, on, hat, dog, ate, and]
D_1	[2, 1, 1, 1, 1, 0, 0, 0]
D_2	[3, 1, 0, 0, 1, 1, 1, 1]

Term Weighting

- Terms in a document are not equally useful for describing its content
- Ex: frequent words in the document -> important
- Ex: words that appear in all documents of the collection -> not important
- Weight used to characterize the importance of the term

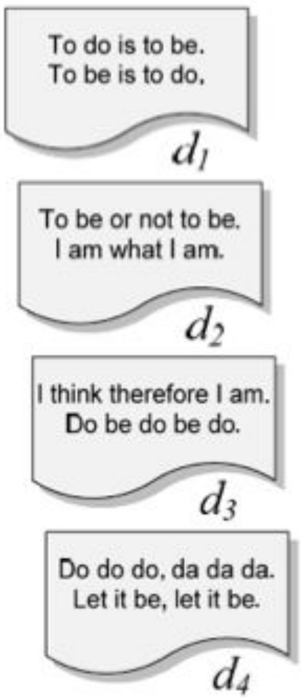
TF - Term Frequency

- The importance of a word is proportional to its frequency in a document

	tf weight
binary	{0,1}
raw frequency	$f_{i,j}$
log normalization	$1 + \log f_{i,j}$
double normalization 0.5	$0.5 + 0.5 \frac{f_{i,j}}{max_i f_{i,j}}$
double normalization K	$K + (1 - K) \frac{f_{i,j}}{max_i f_{i,j}}$

TF - Term Frequency

- Example using the log variation



Vocabulary		$tf_{i,1}$	$tf_{i,2}$	$tf_{i,3}$	$tf_{i,4}$
1	to	3	2	-	-
2	do	2	-	2.585	2.585
3	is	2	-	-	-
4	be	2	2	2	2
5	or	-	1	-	-
6	not	-	1	-	-
7	I	-	2	2	-
8	am	-	2	1	-
9	what	-	1	-	-
10	think	-	-	1	-
11	therefore	-	-	1	-
12	da	-	-	-	2.585
13	let	-	-	-	2
14	it	-	-	-	2

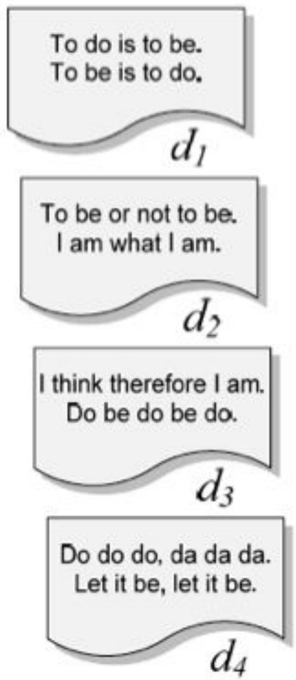
IDF - Inverse Document Frequency

- The importance of a word is inversely proportional to its frequency in a collection:
  - n_i: number of occurrences of a word in the documents
  - N: total number of documents

	idf weight
unary	1
inverse frequency	$\log \frac{N}{n_i}$
inv frequency smooth	$\log(1 + \frac{N}{n_i})$
inv frequeny max	$\log(1 + \frac{max_i n_i}{n_i})$
probabilistic inv frequency	$\log \frac{N-n_i}{n_i}$

IDF - Inverse Document Frequency

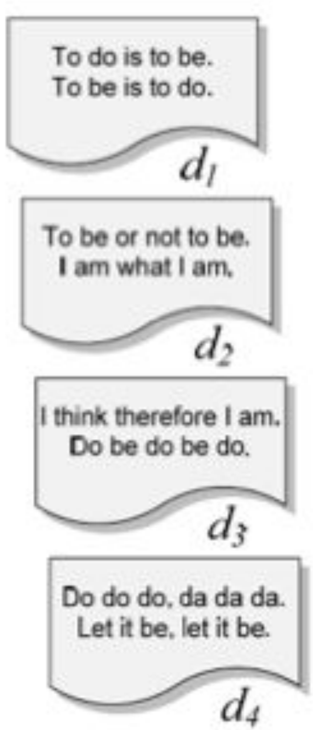
- Example using the log variation



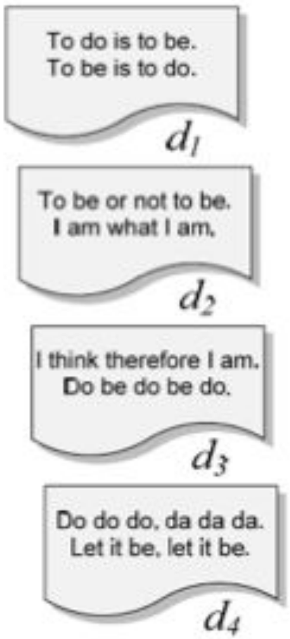
	term	n_i	$idf_i = \log(N/n_i)$
1	to	2	1
2	do	3	0.415
3	is	1	2
4	be	4	0
5	or	1	2
6	not	1	2
7	I	2	1
8	am	2	1
9	what	1	2
10	think	1	2
11	therefore	1	2
12	da	1	2
13	let	1	2
14	it	1	2

TFIDF

- Combination of tf and idf: tf x idf



		d_1	d_2	d_3	d_4
1	to	3	2	-	-
2	do	0.830	-	1.073	1.073
3	is	4	-	-	-
4	be	-	-	-	-
5	or	-	2	-	-
6	not	-	2	-	-
7	I	-	2	2	-
8	am	-	2	1	-
9	what	-	2	-	-
10	think	-	-	2	-
11	therefore	-	-	2	-
12	da	-	-	-	5.170
13	let	-	-	-	4
14	it	-	-	-	4



		d_1	d_2	d_3	d_4
1	to	3	2	-	-
2	do	0.830	-	1.073	1.073
3	is	4	-	-	-
4	be	-	-	-	-
5	or	-	2	-	-
6	not	-	2	-	-
7	I	-	2	2	-
8	am	-	2	1	-
9	what	-	2	-	-
10	think	-	-	2	-
11	therefore	-	-	2	-
12	da	-	-	-	5.170
13	let	-	-	-	4
14	it	-	-	-	4

	term	n_i	$idf_i = \log(N/n_i)$
1	to	2	1
2	do	3	0.415
3	is	1	2
4	be	4	0
5	or	1	2
6	not	1	2
7	I	2	1
8	am	2	1
9	what	1	2
10	think	1	2
11	therefore	1	2
12	da	1	2
13	let	1	2
14	it	1	2

Vocabulary		$tf_{i,1}$	$tf_{i,2}$	$tf_{i,3}$	$tf_{i,4}$
1	to	3	2	-	-
2	do	2	-	2.585	2.585
3	is	2	-	-	-
4	be	2	2	2	2
5	or	-	1	-	-
6	not	-	1	-	-
7	I	-	2	2	-
8	am	-	2	1	-
9	what	-	1	-	-
10	think	-	-	1	-
11	therefore	-	-	1	-
12	da	-	-	-	2.585
13	let	-	-	-	2
14	it	-	-	-	2

Representation Learning

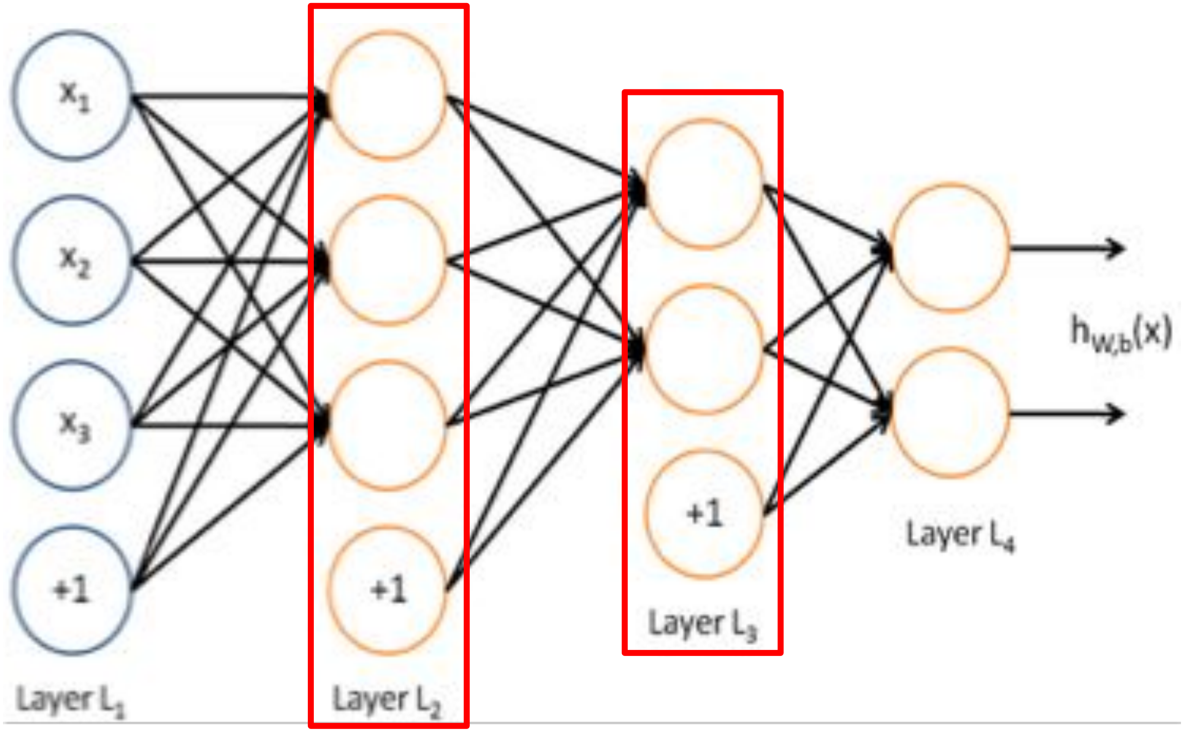
- In ML models, instances are represented by their features
- Motivation:
  - Instance/data representation is essential for effective ML models
  - Less dependent on feature engineering
  - Dimensionality reduction

- Definition:
  - Set of techniques that learn a "better" representation from the raw data

- Distributed representation or embeddings
  - Dense and low dimensional representation
  - Dimensions have no meaning

Example of Representation Learning: MLP

- Different levels of abstraction of the input



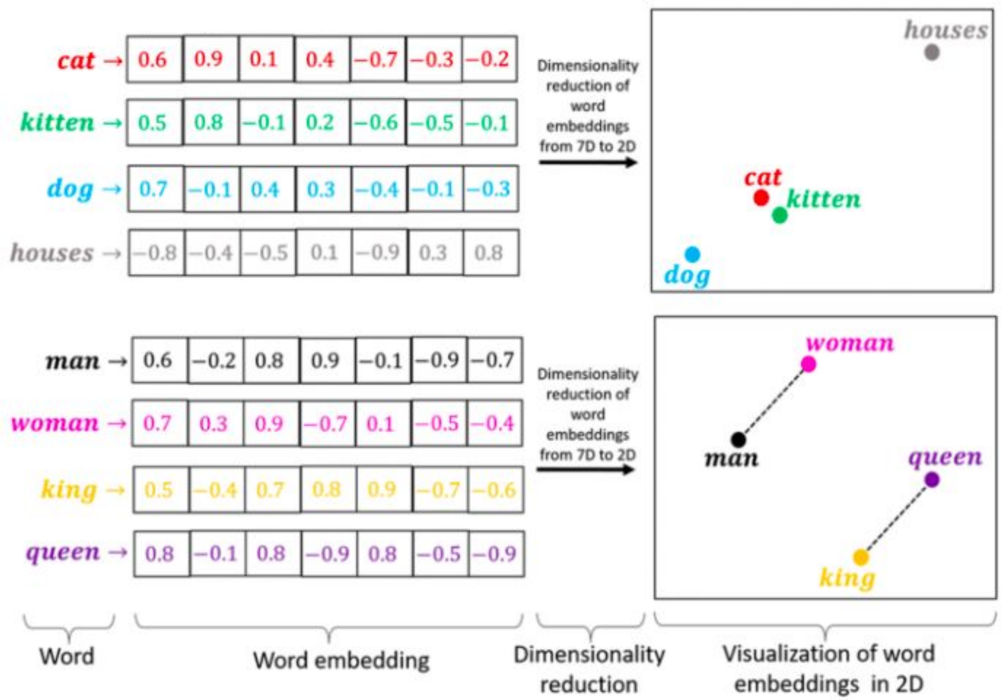
Img-Source: http://ufldl.stanford.edu/wiki/

Embeddings in NLP

- Represent a linguistic unit as a dense vector
- Linguistic unit: character, word, sentence, document
- Map semantic meaning into a geometric space (embedding space)

Word Embeddings

- Embed the context of a word in a low-dimensional vector (e.g., 100, 200)
- Similar words are close in this space



Img-Source: https://medium.com/@hari4om/word-embedding-d816f643140

Word Embeddings

- Built using dimensionality reduction techniques
  - Frequency based models (Latent Semantic Indexing)
  - Prediction based models (Neural networks)

Latent Semantic Indexing

- Build document and word representations
- Based on the co-occurrence of the words in the documents
- Dimensionality reduction: singular value decomposition (SVD)
  - $C = U\Sigma V^T$
  - C: term-document matrix
- Compute reduced C' with fewer dimensions

$C = U\Sigma V^T$

C	d_1	d_2	d_3	d_4	d_5	d_6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	1	0	0	1	1	0
tree	0	0	0	1	0	1

=

U	1	2	3	4	5
ship	−0.44	−0.30	0.57	0.58	0.25
boat	−0.13	−0.33	−0.59	0.00	0.73
ocean	−0.48	−0.51	−0.37	0.00	−0.61
wood	−0.70	0.35	0.15	−0.58	0.16
tree	−0.26	0.65	−0.41	0.58	−0.09

×

Σ	1	2	3	4	5
1	2.16	0.00	0.00	0.00	0.00
2	0.00	1.59	0.00	0.00	0.00
3	0.00	0.00	1.28	0.00	0.00
4	0.00	0.00	0.00	1.00	0.00
5	0.00	0.00	0.00	0.00	0.39

×

V^T	d_1	d_2	d_3	d_4	d_5	d_6
1	−0.75	−0.28	−0.20	−0.45	−0.33	−0.12
2	−0.29	−0.53	−0.19	0.63	0.22	0.41
3	0.28	−0.75	0.45	−0.20	0.12	−0.33
4	0.00	0.00	0.58	0.00	−0.58	0.58
5	−0.53	0.29	0.63	0.19	0.41	−0.22

Term-Document Matrix

Word Embeddings

Singular values

Doc Embeddings

Reducing to 2 Dimensions

U	1	2	3	4	5
ship	−0.44	−0.30	0.00	0.00	0.00
boat	−0.13	−0.33	0.00	0.00	0.00
ocean	−0.48	−0.51	0.00	0.00	0.00
wood	−0.70	0.35	0.00	0.00	0.00
tree	−0.26	0.65	0.00	0.00	0.00

Σ_2	1	2	3	4	5
1	2.16	0.00	0.00	0.00	0.00
2	0.00	1.59	0.00	0.00	0.00
3	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.00
5	0.00	0.00	0.00	0.00	0.00

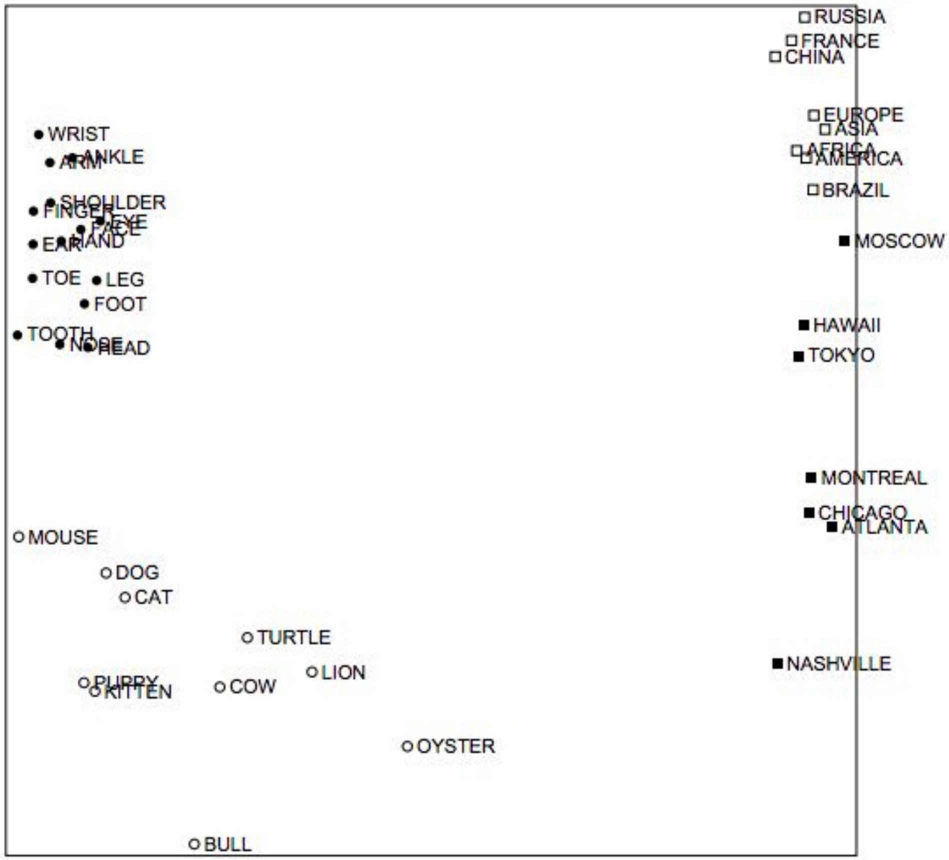
V^T	d_1	d_2	d_3	d_4	d_5	d_6
1	−0.75	−0.28	−0.20	−0.45	−0.33	−0.12
2	−0.29	−0.53	−0.19	0.63	0.22	0.41
3	0.00	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.00	0.00
5	0.00	0.00	0.00	0.00	0.00	0.00

Word Embeddings

Setting singular values to 0

Doc Embeddings

Projecting LSI Word Embeddings



Rohde et al., An Improved Model of Semantic Similarity Based on Lexical Co-Occurrence, 2005

Pros & Cons

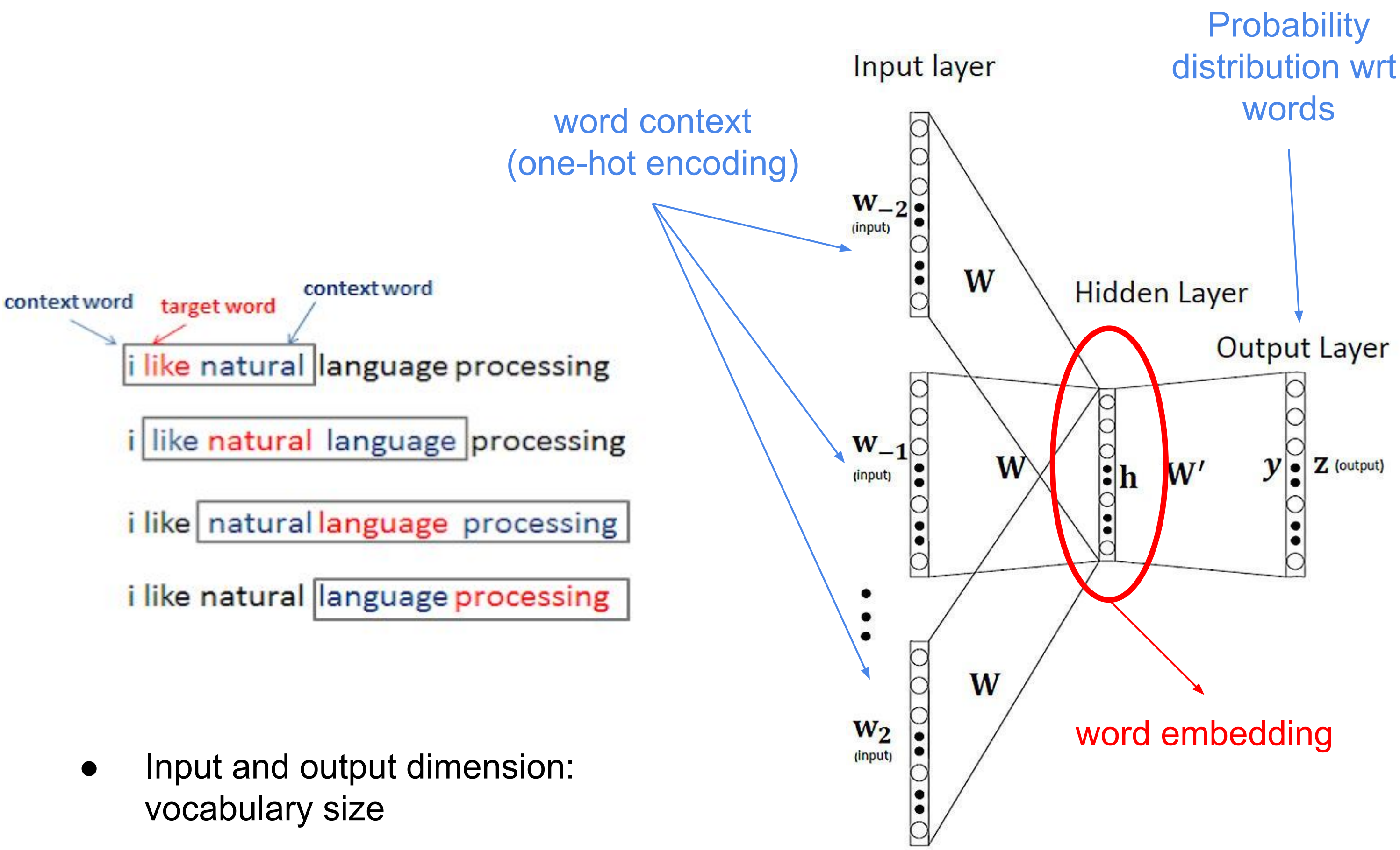
- Pros
  - Simple
  - Capture similarity
- Cons
  - Co-occurrence matrix is sparse
  - Quadratic cost (SVD)

Prediction Based Models: CBOW

- Predicts word in the context (language modeling)

Img-Source:
https://thinkinfi.com/continuous-bag-of-words-cbow-multi-word-model-how-it-works/

Prediction Based Models: CBOW

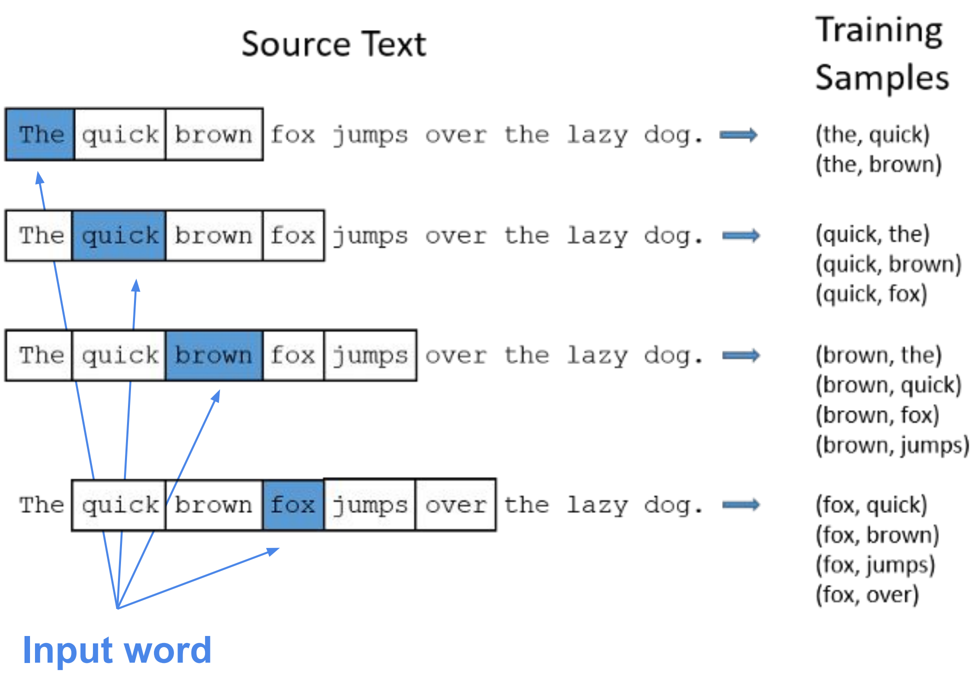


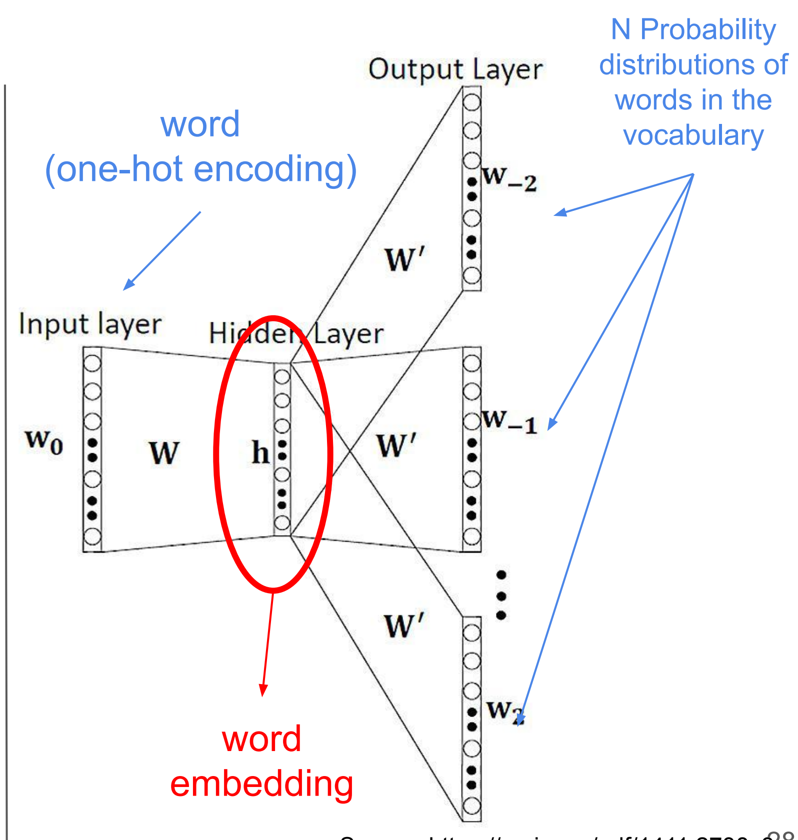
- Input and output dimension: vocabulary size

Source: https://arxiv.org/pdf/1411.2738v3.pdf (a) CBOW

Prediction Based Models: Skip-gram

- Predicts the context of a word



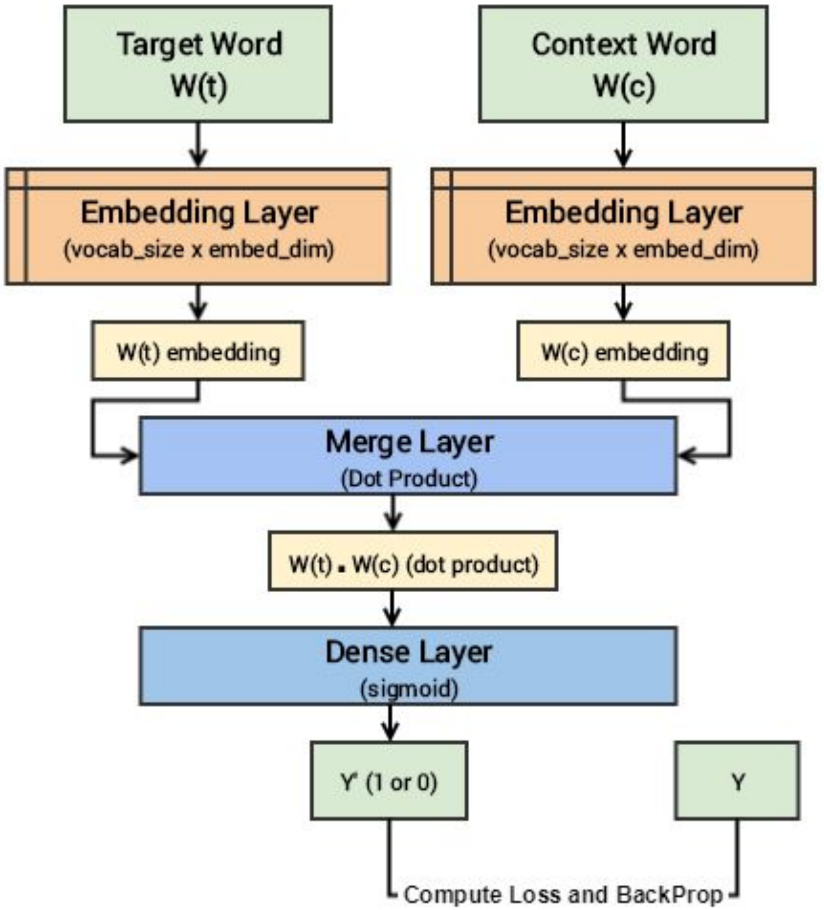


(b) Skip-Gram

Source: https://arxiv.org/pdf/1411.2738v3.pdf

Skip-gram with Negative Sampling

- Previous models are very costly
- Predicts words as neighbors
- Negative examples: words that are not neighbors
- Inputs:
  – Word and its context word
  – Representation: one-hot encoding
- Output:
  – Probability of matching



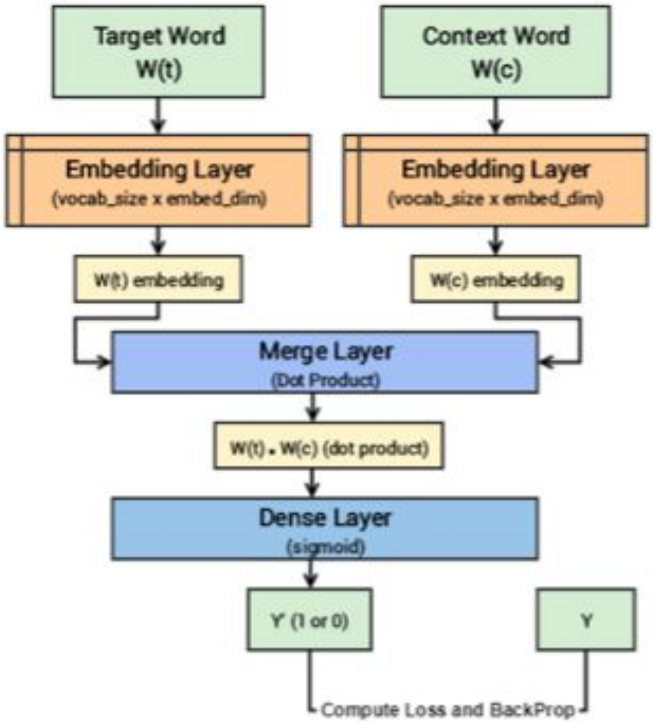
Img-Source: http://arxiv.org/pdf/1301.3781.pdf

Skip-gram vs Negative Sampling

Negative Sampling

input word	output word	target
make	shalt	1
make	aaron	0
make	taco	0

Source: https://jalammar.github.io/illustrated-word2vec/



Img-Source: http://arxiv.org/pdf/1301.3781.pdf

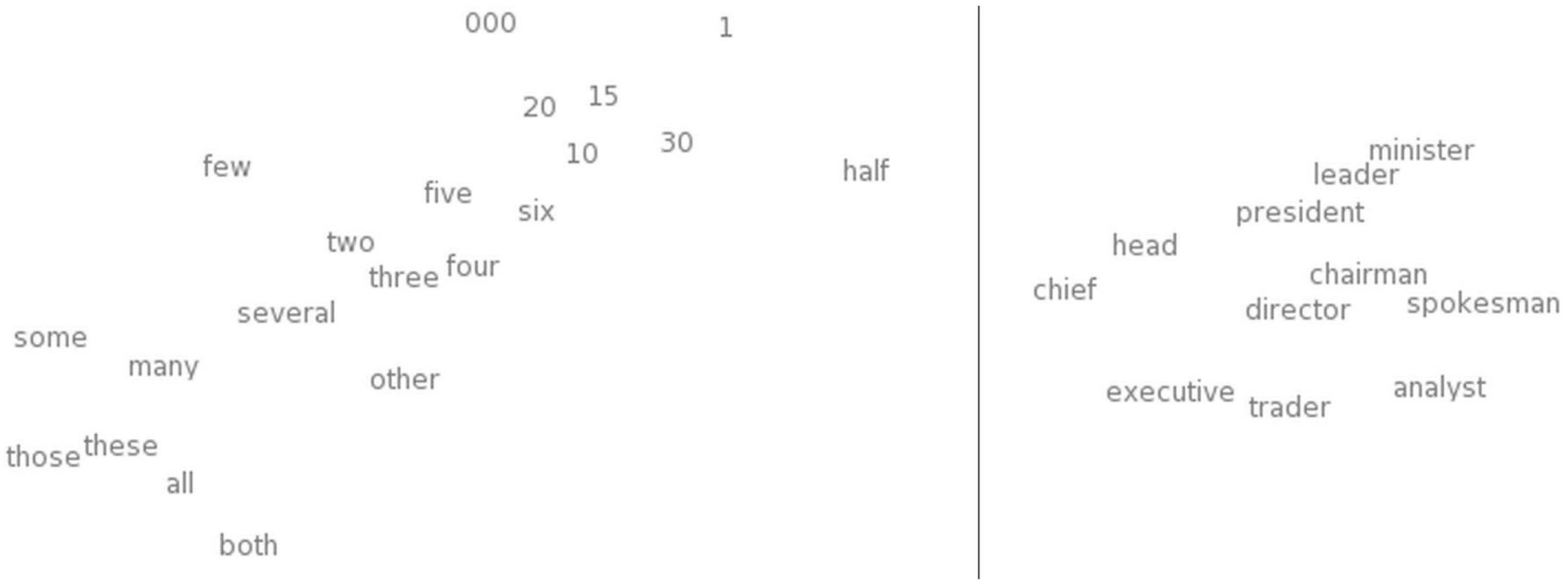
Word Embeddings

- Example: closest words

FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

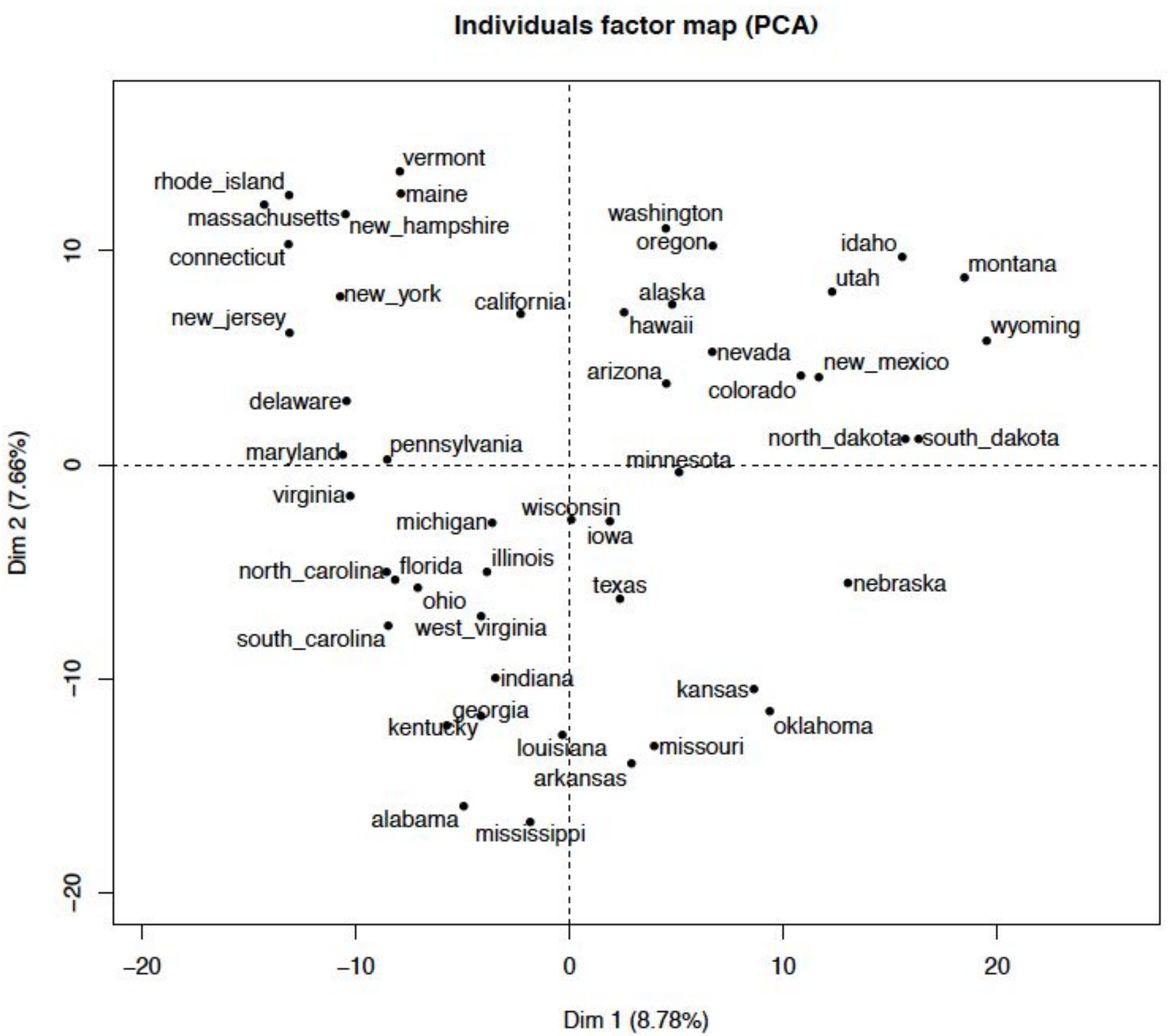
Img-Source: http://arxiv.org/pdf/1301.3781.pdf

Word Embeddings in 2D



Img-Source: http://metaoptimize.s3.amazonaws.com/cw-embeddings-ACL2010/embeddings-mostcommon.EMBEDDING_SIZE=50.png

Word Embeddings in 2D



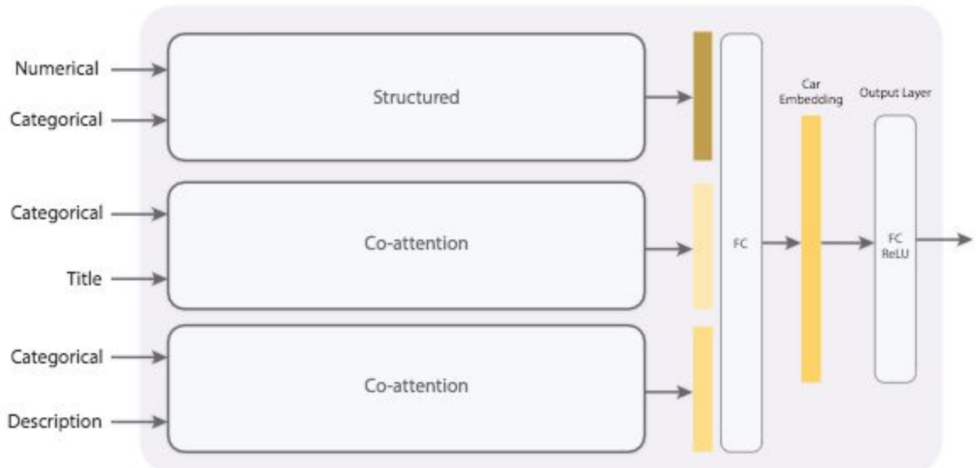
Impact of Word Embeddings

- In most networks for text, word embeddings are the basis
- Having good word embeddings increases significantly your performance
- Which is the best is hard to tell
  - Try all available
  - Use pre-trained vectors available (you can also create yours)
- Corpus selection and tuning the parameters for the task at hand
  - E.g. for sentiment, *good* and *bad* should be far away in vector space

Existent Tools

- Demo:
  - https://rare-technologies.com/word2vec-tutorial/
- Word2Vec
  - https://code.google.com/p/word2vec/
- Doc2Vec
  - Learns dense representations for phrases, sentences and documents
  - https://groups.google.com/forum/#!topic/word2vec-toolkit/Q49FIrNOQRo or Gensim
- GloVe:
  - http://nlp.stanford.edu/projects/glove/

Embeddings for Car Price Prediction

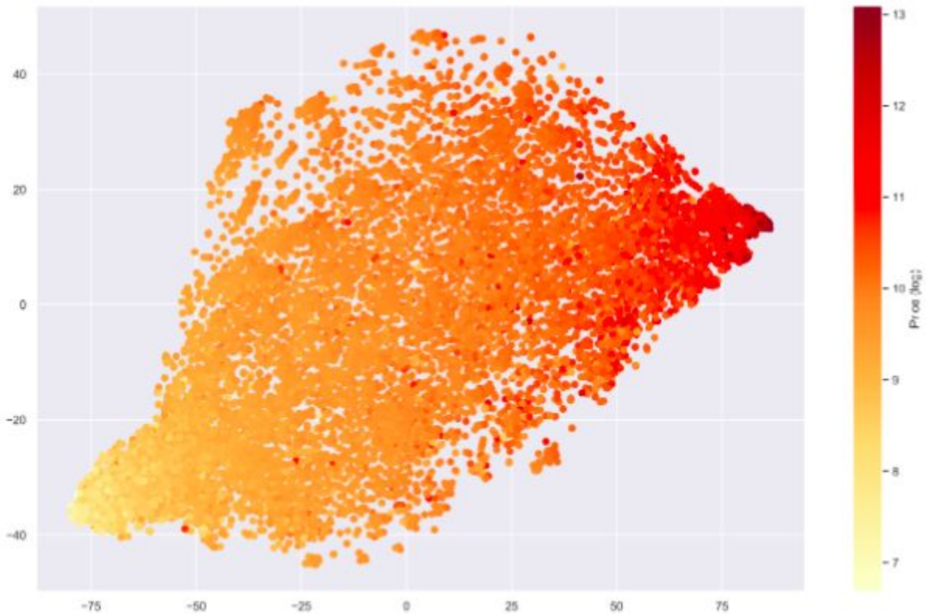


Model

Model	Features	RMSE	MALE
Linear Regression	STR	15,288	0.231
	TXT	39,801	0.217
	STR+TXT	18,555	0.171
	CE	12,926	0.125
SVR	STR	15,410	0.228
	TXT	31,140	0.249
	STR+TXT	17,672	0.185
	CE	13,076	0.127
Random Forest	STR	13,940	0.179
	TXT	16,909	0.193
	STR+TXT	15,440	0.168
	CE	**12,302**	**0.117**
LightGBM	STR	14,929	0.185
	TXT	14,601	0.177
	STR+TXT	13,560	0.130
	CE	12,305	0.120
H2O AutoML	STR	15,351	0.258
	TXT	18,644	0.283
	STR+TXT	20,341	0.299
	CE	12,439	0.118
Regression Layer	CE	13,949	0.126

Results

Embeddings



Car Embeddings

Query	Top 5	Average Price (USD)
Lamborghini	-	271,591
	Ferrari	137,567
	Aston	119,979
	Aston Martin	124,978
	Hummer	17,253
	Audi	23,796
Rolls-royce	-	116,298
	Ferrari	137,567
	Bentley	87,142
	Aston	119,979
	Jaguar	23,795
	Studebaker	30,290
Volkswagen	-	12,429
	Ford	19,112
	Volvo	20,431
	Porsche	51,561
	Cadillac	20,480
	Saturn	4,536
Saturn	-	4,536
	Saab	5,484
	Studebaker	30,290
	Honda	14,979
	Land	30,755
	Nissan	14,379

Brand Embeddings

Classification Evaluation Metrics

- Precision
- Recall
- F1 (F-measure)
- Accuracy

	in the class	not in the class
predicted to be in the class	true positives (TP)	false positives (FP)
predicted to not be in the class	false negatives (FN)	true negatives (TN)

$$\text{precision:} P = TP/(TP + FP)$$

$$\text{recall:} R = TP/(TP + FN)$$

$$F_1 = \frac{1}{\frac{1}{2}\frac{1}{P} + \frac{1}{2}\frac{1}{R}} = \frac{2PR}{P + R}$$

Accuracy = (TP+TN)/(TP+TN + FN + FP)

Evaluation Strategies

- Holdout set: not used for training
- Training/validation/test
- Cross-validation

Evaluation Strategies

- Holdout set: not used for training
- Training/validation/test
- Cross-validation