An Overview of Word Embedding-based Machine Learning Methods in Topic Recognition Tasks

Sarah Wiegreffe
Advised by Dr. Paul Anderson
Dept. of Computer Science

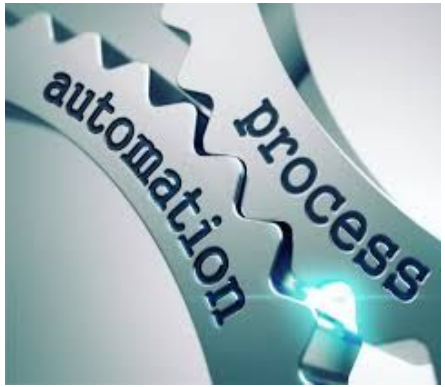
COLLEGE of CHARLESTON
COMPUTER SCIENCE

Why Natural Language Processing (NLP)?

Lots of Data

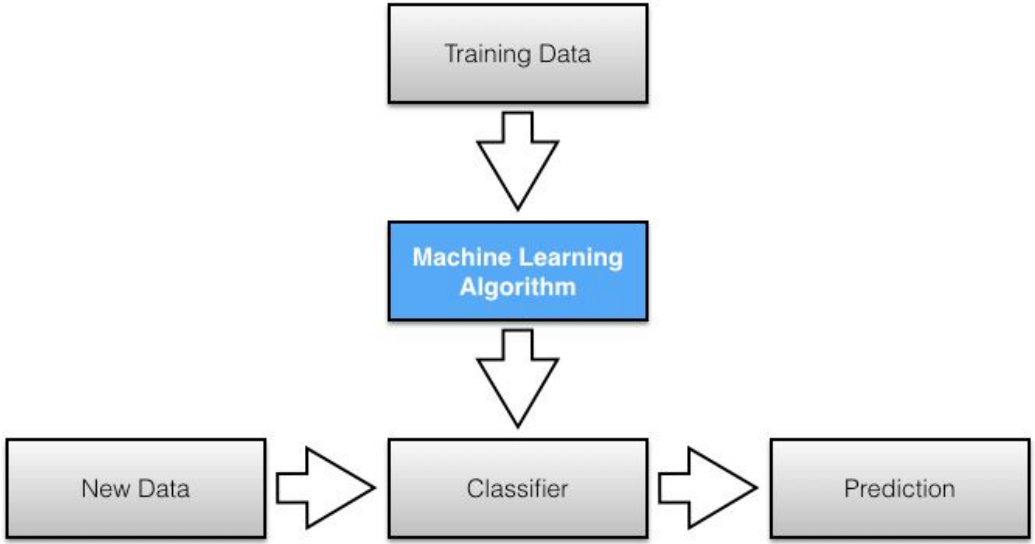
Computer Understanding

Topic Recognition



Sentiment Analysis

What is machine learning?



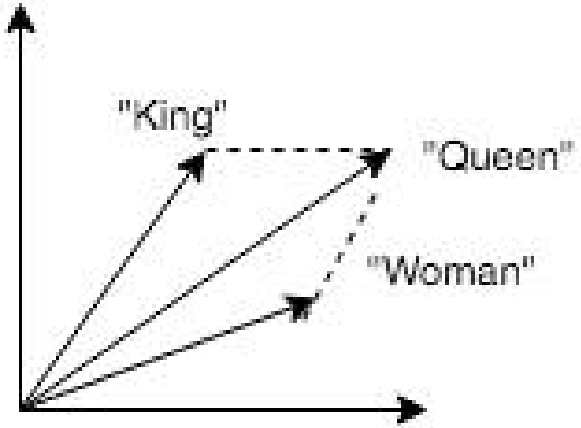
http://sebastianraschka.com/Articles/2014_intro_supervised_learning.html

Traditional machine learning algorithms need fixed-length feature vectors.

- How to generate?
- Traditionally done by Bag of Words

The dog is on the table.

Word Embeddings!

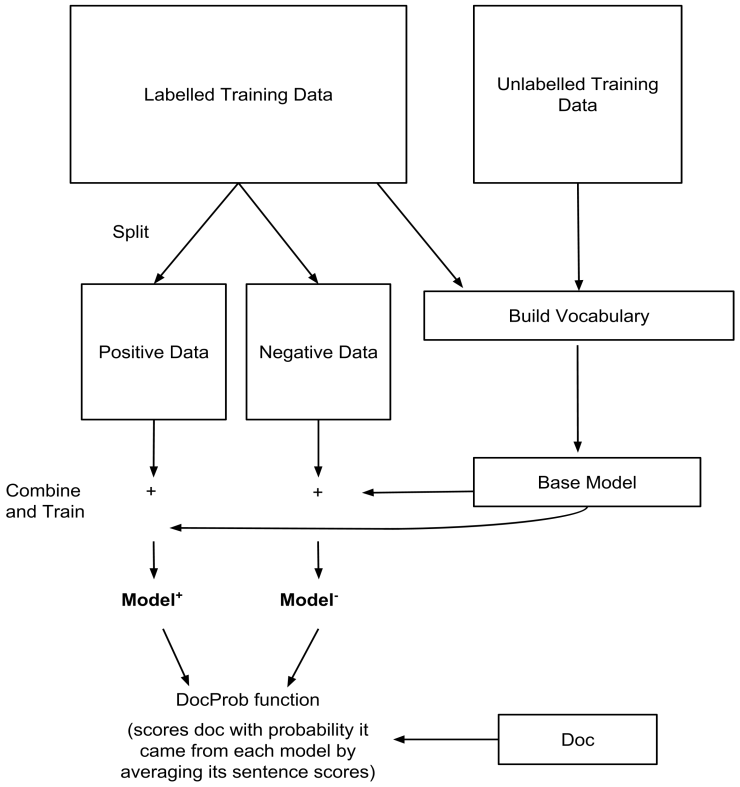


- Word2Vec
  - *Distributed Representations of Words and Phrases and their Compositionality*, Mikolov et. al.
  - Fixes word similarity problem

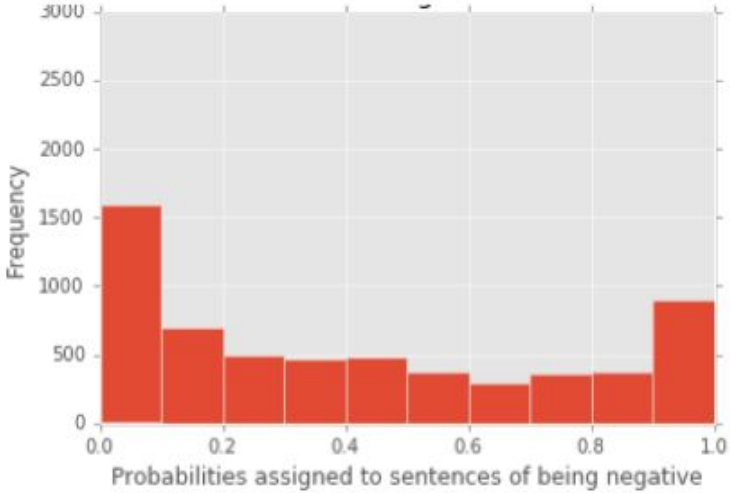
“King” - “man” + “woman” = “Queen”

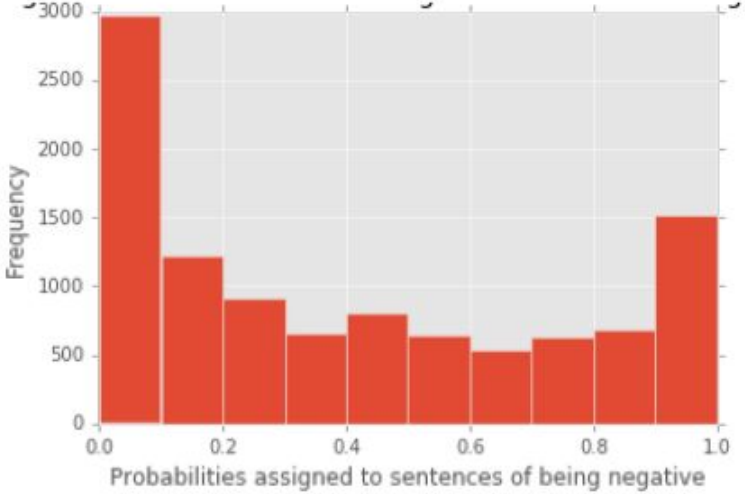
Word2Vec Inversion

- Combining preprocessing and machine learning steps
- Neural network architecture



- *Document Classification by Inversion of Distributed Language Representations* by Taddy.


3000
2500
2000
1500
1000
500
0
Frequency
0.0
0.2
0.4
0.6
0.8
1.0
Probabilities assigned to sentences of being negative


3000
2500
2000
1500
1000
500
0
Frequency
0.0
0.2
0.4
0.6
0.8
1.0
Probabilities assigned to sentences of being negative

"This movie is like the thousand \"cat and mouse\" movies that preceded it. (The following may look like a spoiler, but it really just describes a large class of movies) There is the passionate, wise main character, his goofy but well-meaning sidekick with his ill-placed attempts at humorous comments, the initially-hostile but soon softened gorgeous lady who triggers the inevitable \"unlikely\" love story, the loved ones taken hostage, and of course the careless evil adversary with his brutal minions. Everybody has seen tons of these movies already, and \"National Treasure\" is like any one of them, with only a slightly modified wrapping. Every turn of the story was easily predicted (and I can assure you I am not the sharpest tool in the shed). I am quite tired of feeling tricked for money after exiting the theater from a Hollywood movie, and if you have ever felt that way too, heed my warning; stay miles away from this movie."

Sentence Scores:

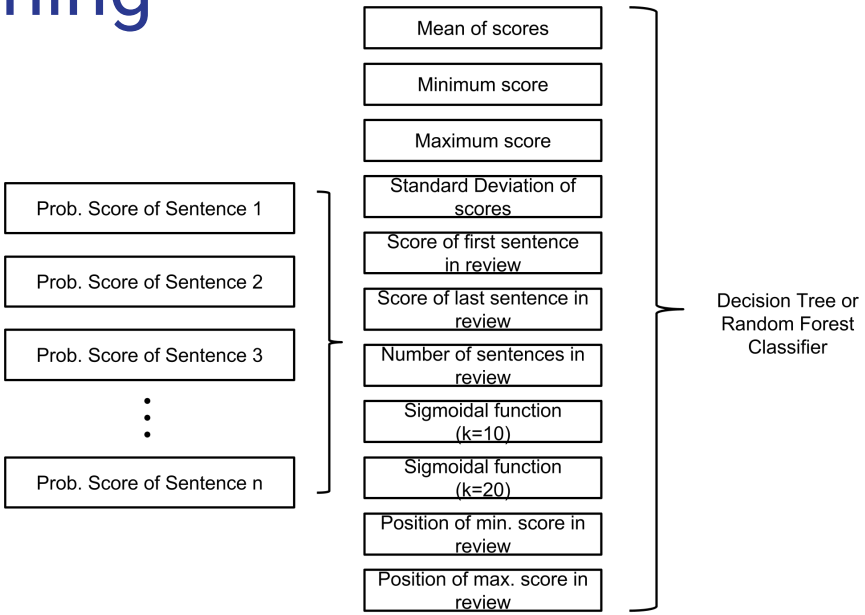
	0	1	doc
0	0.5986	0.4014	0
1	0.00792023	0.99208	0
2	0.369248	0.630752	0
3	0.445318	0.554682	0
4	0.999581	0.000419095	0

Hypothesis:

Instead of averaging together sentence scores, can allowing a computer to learn what the most important features of a review or positions of sentences are improve prediction accuracy?

- We strive to do this with as little human input as possible for generalizability.

Our Methods- employing meta-features and ensemble learning

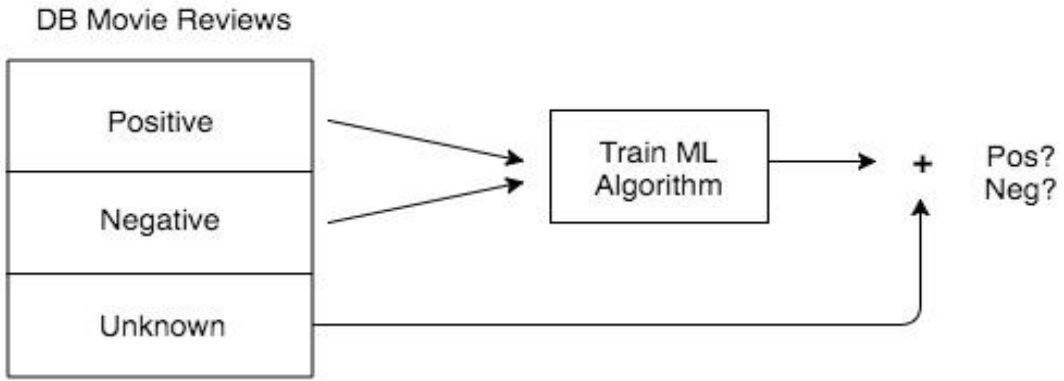


Each sentence of a review receives a probability score from the 0 model of the Word2Vec Inversion algorithm.

The generated metafeatures become a fixed-length feature vector used to train either a decision tree or a random forest classifier.

Datasets

- IMDb movie reviews (100,000)
  - Positive or negative sentiment?
- Amazon Fine Food reviews
  - 1-5 star ratings



Results

Dataset	Bag of Words + Random Forest	Word2Vec Inversion	Inversion + Decision Tree of Meta-features	Inversion + Random Forest of Meta-features	Bagging
IMDb movie reviews	**83.97%**	**87.71%**	87.98%	**88.23%**	79.97%
Amazon Fine Food reviews	**70.49%**	**73.17%**	71.03%	**75.47%**	

Next Steps

- Build confidence intervals to test more robustly
  - To account for randomness in Word2Vec model construction
- Rerun all models on a third dataset
  - Twitter data
- Submit for publication in May

Summary/Overview

- Word2Vec Inversion methods show slight improvement over current methods.
- With more robust testing, we hope to discern if they can beat baseline models more generally.
- If so, this is a great improvement not only for achieving higher accuracy of classification, but also for simplifying the process of building machine learning classifiers on text.

Tools

jupyter Pokitdok Python Last Checkpoint: 03/02/2017 (unsaved changes)

Control Panel | Logo

File Edit View Insert Cell Kernel Widgets Help

Python [default]

Code | CellToolbar

In [1]:
```
import json
import pandas as pd
import numpy as np
import pickle
#https://docs.python.org/2/library/json.html #JSON for Python
```

In [4]:
```
with open("sample.jsonl", 'r') as f:
    the_data = [json.loads(l) for l in f]
```

In [4]:
```
type(the_data[0])
```

Out[4]:
```
dict
```

In [5]:
```
medicarePUF1 = pd.read_table('./Big_Files/Medicare_Provider_Util_Payment_PUF_CY2014.txt', skiprows = [1])
#medicarePUF2 = pd.read_table('Medicare_Util_Payment_Abbrev_PUF.txt', skiprows = [1]) #for smaller file
```

```
/Users/kaya/anaconda/lib/python3.5/site-packages/IPython/core/interactiveshell.py:2717: DtypeWarning: Columns (10) ha
ve mixed types. Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

In [6]:
```
def PUFtoJSON(j):
    PUF = {}
    cols = medicarePUF1.columns.tolist()[16:]
    for col in cols:
        PUF[col] = str(medicarePUF1[col][j])
    return PUF
    print (PUF)
```

Thank you to Dr. Anderson, the Data Science program, the Computer Science department, the Honors College, and Dr. Jihad Obeid in the MUSC Biomedical informatics Center.