Naives bayes Section

001 - Introduction to NLP

DATE:

- In this section we will begin a dissussion on using raw string text for machine learning models.
- The general idea is general is known as "Natural language processing"

Section Overview

- Naive Bayes Algorithm and NLP
- Extracting features from text Data
- Text Classification project

— Keep in mind:-

This section focus on supervised learning text tasks. We will discuss unsupervised text task later on.

2 - Naive Bayes Algorithm : Bayes Theorm

- Naive Bayees is the shorthand for a set of algorithm that use Bayes' Theorm for supervised learning classification.
- Bayes Theorm is a probability formula that leverages previously known probablities to define probability of related events occuring.

CamScanner

DATE:

- Naive Bayes method are set of supervised learning algorithm based on applying Bayes' Theorm.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

- 1700s Thomas Bayes was a presbyterian minister in England who studied theology, statistics, and logic.

$P\cancel{(A|B) =}$ B

Bayes theorm was published after his death, Richard price edited and published his notes.

- Bayes Theorm
  - A and B are events
  - P(A|B) is probability of event A given that B is true
  - P(B|A) is probability of event B given that A is True
  - P(A) probability of A occuring.
  - P(B) is probability of B occuring

003 - Model Algorithms

- we model the probability of belonging to a class a vector of features -

$X = (x_1, \dots, x_n)$

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)} \Rightarrow P(C_k|X) = \frac{P(C_k)P(X|C_k)}{P(X)}$$

$$\Rightarrow P(C_k|X) = \frac{P(C_k, x_1, \dots, x_n)}{P(X)}$$

The numerator is equivalent to a joint probability model:

$\cancel{P(C_k|X)=}$

The chain rule can rewrite this numerator as a series of product of conditional probabilities:

$$P(C_k, x_1, \dots, x_n) = P(x_1, \dots, x_n, C_k)$$

$$= P(x_1|x_2, \dots, x_n, C_k)\,P(x_2, \dots, x_n, C_k)$$

$$= P(x_1|x_2, \dots, x_n, C_k)\,P(x_2|x_3, \dots, x_n, C_k)$$

- Finally we need to make an assumption: we assume all X features are mutually independent of each other.
- Allowing for this conditional probability

$$P(x_i|x_{i+1}, \dots, x_n, C_k) = P(x_i|C_k)$$

- Then the joint model (the full Naive Bayes model) is fully written as:

$$p(C_k \mid x_1, \ldots, x_n) \propto p(C_k, x_1, \ldots, x_n)$$
$$\propto p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)$$
$$\propto p(C_k) \prod^{n} p(x_i \mid C_k),$$

~~where $\propto$ denotes proportionality~~

Multiple variations of Naive Bayes model, including:

- multinomial Naive Bayes
- Guassain Naive Bayes
- Complement Naive Bayes
- Bernoulli Naive Bayes
- Categorical Naive Bayes

Multinomial Naive Bayes) we will focus

Slide check kro

- Note how a higher alpha values will be more "smoothing"; giving each word less distinct importance.
- Now lets move on to focusing on feature extraction in general.

4 # Feature Extraction Part One

(M)(T)(W)(T)(F)(S) DATE:

— Main Method of feature Extraction
- Count Vectorization
- TF-IDF
  - Term Frequency-Inverse Document Frequency

- Count vectorization treats every word as a feature, with the frequency counts acting as a "strength" of the feature/word.
- For larger documents, matrics are stored as a sparse matrix to save space, since so many values will be zero.

- stopwords — a, the,

- TF-IDF → will count and inverse multiple document

#7 Coding

```
text = ['This is a line',
        "This is another line",
        "Comple         ]

from sklearn.feature_extraction.text
        import CountVectorizer
cv = CountVectorizer()
```

M T W T F S text DATE:

cv. fit_transfer( )

sparse_matrix. to dense ( )
cv. vocabulary_

tfidf = TfidTransformer( )
sparce_matrix
tf. fit_transform (~~text~~)

import TfidVectorizer

tv = tfidVectorizer( )
tv_result = tv_fit_transfer (text)