11/12: Naive Bayes

Discord: https://discord.gg/68VpV6

Methods of Classification

What methods have we used to classify data?

Classification (Review)

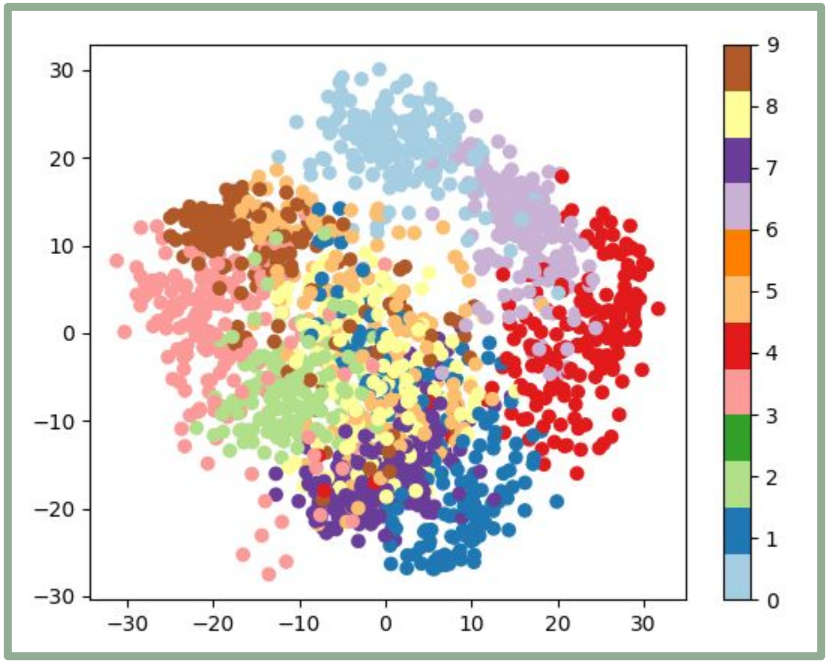
Recall some of the projects we've done with classification:

- Twitter Sentiment Analysis
- MNIST Handwritten Digit Recognition
- Chatbot (matching intents with tags)

Classification Main Problem

For all of these projects, we've used either logistic regression or some type of neural network (sequential and convolutional)

The only problem is that these methods require a lot of computation. If we want to deploy these models in an application, they'll drastically slow our performance



Ideal Classification Model

Hence, our ideal model will have the following properties:

- Ability to predict sufficiently*
- Fast and computationally inexpensive
- Can use a pre-trained model

Fortunately, **Naive Bayes** has all of these properties

Statistical Overview

What principle is Naive Bayes based on?

Intro to Probability

In probability and statistics, we refer to distributions and events as capital letters

For example:

let A = the outcome of flipping a coin

let B = the outcome of drawing from a standard deck

$$P(A = heads) = 0.5$$

$$P(B = Ace_{Hearts}) = \frac{1}{52}$$

Conditional Probability

We use the vertical bar "|" to denote conditional probability

P(A | B) = chance that A happened, given that B happened

Mathematically:

probability of A and B

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

probability of A given B

probability of B

Independence

We say that two events are independent iff:

$$P(A \mid B) = P(A)$$

For example:

let A = the outcome of flipping a coin

let B = the outcome of flipping the same coin a second time

$$P(A \mid B = tails) = P(A)$$

$$P(A \mid B = heads) = P(A)$$

Bayes Theorem

We can use what we've learned to derive Bayes Theorem:

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

This is the foundation of the Naive Bayes algorithm we'll be using!

Applying Naive Bayes

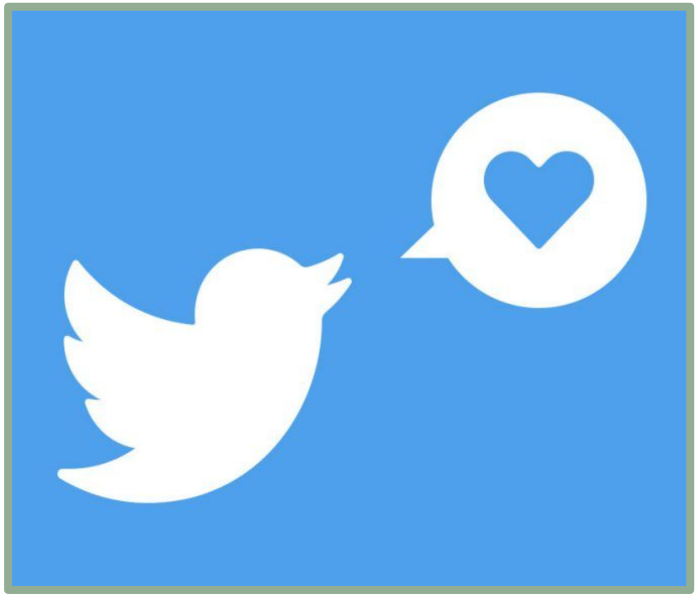
How can we apply this formula?

Project Introduction

For this week's notebook, we'll be looking at how we can apply Naive Bayes to predict if a tweet was made by Joe Biden or Kanye West, two very serious presidential contenders

Hence, we'll name our variables as follows:

- K = Kanye tweet
- B = Biden tweet



Naive Bayes Algorithm Overview

Our algorithm performs the following steps:

1. Cleans/Tokenizes the words
2. Create a frequency dictionary
3. Find the probability of each word being from Kanye/Biden
4. Calculate the “likelihood” for each word in our dictionary

Cleaning, Tokenizing, etc.

We’ve done this many many many times, but as a reminder:

- Remove excess symbols/spaces (#, @, etc)
- Deleting stopwords/unimportant words
- Making everything lowercase

```
“To the window to the wall”
```

↓

```
[“window”, “wall”]
```

Creating a Frequency Dictionary (Review)

Our dictionary will have a similar schema to when we did sentiment analysis:

Word	Biden Count	Kanye Count
Jesus	16	0
Trump	465	5
Folks	0	133
Happy	17	7

```
dict = {“word” : [biden_count, kanye_count]}
```

Conditional Word Probabilities

Now that we have a dictionary of every word, we can calculate the probability of each word coming from Biden/Kanye

$$P(word_{biden}) = \frac{freq(word_{biden})}{count_{biden}}$$

$$P(word_{kanye}) = \frac{freq(word_{kanye})}{count_{kanye}}$$

Conditional Word Probabilities: Smoothing

To make our data fit a little better, we'll introduce smoothing

$$P(word_{biden}) = \frac{freq(word_{biden}) + 1}{count_{biden} + count_{unique}}$$

$$P(word_{kanye}) = \frac{freq(word_{kanye}) + 1}{count_{kanye} + count_{unique}}$$

To do this calculation, we need to find the number of unique words in the dictionary, which will just be the length of it!

Likelihood

Now that we have the probability for each word separated by candidate, we need to combine the data into a singular value

Hence, we'll use likelihood to find how much more/less likely a word is to be from Kanye/Biden with the following formula:

$$likelihood = log(\frac{P(word_{kanye})}{P(word_{biden})})$$

Predictions and Analysis

How good is Naive Bayes?

Making Predictions

For Naive Bayes, making predictions is fairly simple

All we need to do is take the sum of the likelihood of each word in our tweet

$$pred = \sum_{i=0}^{n} likelihood(word_i)$$

We can then decide between Kanye/Biden depending on if the sum is positive/negative

Efficiency of Predictions

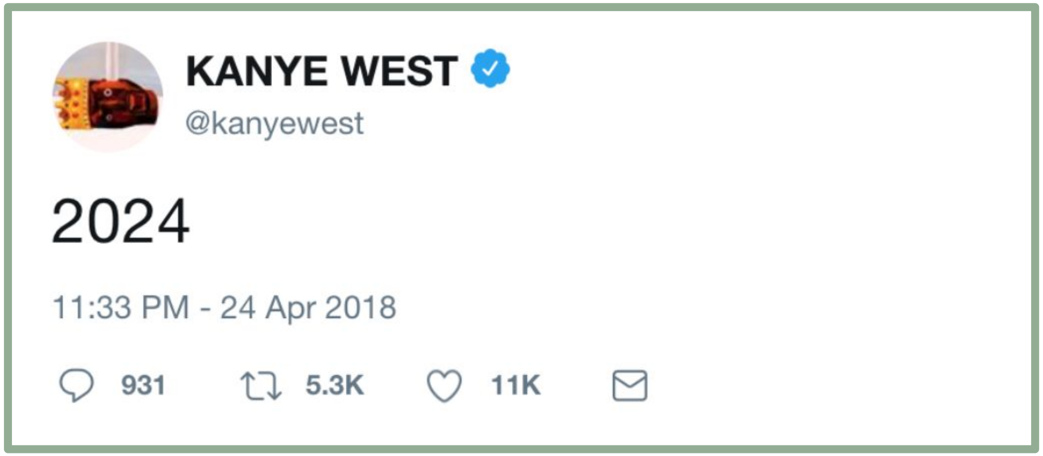
If we store our likelihoods in a dictionary, our lookup time will be very quick!

Hence, making predictions with this sum will be much quicker than the matrix multiplication needed in logistic regression and neural networks

Main Disadvantage of Naive Bayes

The main problem with Naive Bayes is that it assumes that all predictors are **mutually independent**, meaning that there is no correlation between the two inputs

However, this is clearly not the case with tweets, as there are times when Kanye and Biden are tweeting about the same thing (like the election)



KANYE WEST

@kanyewest

2024

11:33 PM - 24 Apr 2018

931 5.3K 11K

Tasks to Complete

1) Work on the notebook (**naive_bayes.zip)** in the google drive folder https://drive.google.com/drive/folders/1GsX1UOWXmZalT2NgDwmUZv7iwRAxNDiZ?usp=sharing

Try to work on it collaboratively! You might meet some people you could do a project with in the future

As always, let us know if you need any help!

SIG
NLL