CLASSIFYING USING PROBABILITY THEORY

Naïve bayes

4 January 2022, Jyothish K. J

A simple introduction

- We have a set of data points.
- We want to classify it into 2 classes C_1 and C_2.
- We'll check the probability of the data point belonging to C_1 with P_{C1} and to C_2 with P_{C2}.
- For our discussion P_{C1} and P_{C2} are probability functions that'll give us a value between 1 and 0. and we'll classify the datapoint simply as:

  If $P_{C1}(data) > P_{C2}(data)$ then data belongs to C1 and

  If $P_{C2}(data) > P_{C1}(data)$ then data belongs to C2
- We'll use **conditional probability** and **bayes rule** to obtain these probabilities.
- Along the way we'll use some naïve assumptions, thereby we say that we're classifying using **naïve bayes**.

Pre requisite:

- First thing is That probability independent of the sequence of events that are happening.

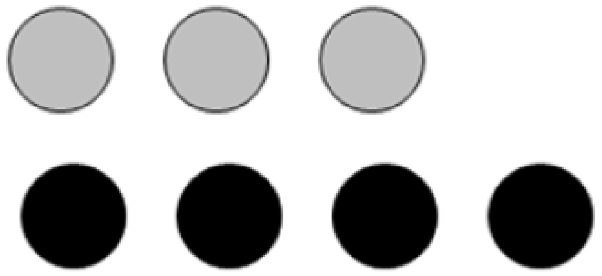

Figure 4.2 A collection has seven stones that are gray or black. If we randomly select a stone from this set, the probability it will be a gray stone is 3/7. Similarly, the probability of selecting a black stone is 4/7.

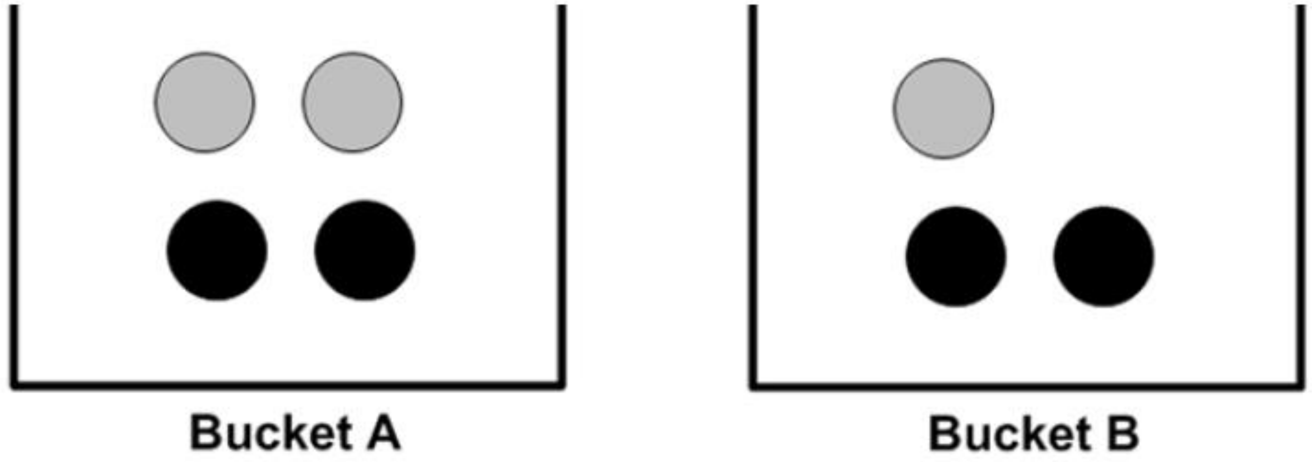

Figure 4.3 Seven stones sitting in two buckets

2 variables now: **Color** and **Bucket**

On a blindfolded draw of ball, probability of a grey ball being drawn from bucket b is same as probability of a ball being drawn from bucket b being grey.

- **Second thing** to know is **Conditional Probability.** Generally for a sequence of events a,b,c,d. The probability of such a combination of events happening is given by:

$$\mathbf{P(a,b,c,d) = P(a|b,c,d).P(b|c,d).P(c|d).P(d)}$$

Which is read as:

  - **P(a,b,c,d)** : *Probability of a,b,c,d*
  - **P(d):** Probability of d happening
  - **P(c|d) :** Probability of **c** happening such that **d** has already happened
  - **P(b|c,d) :** Probability of **b** happening such that **c** and **d** have already happened
  - **P(a|b,c,d) :** Probability of **a** happening such that **b,c and d** have already happened

- For the previous example of stones in 2 buckets,
  - The probability of a randomly drawn stone being grey and belonging to bucket B is given as:

$$\mathbf{P(gray\ and\ bucket\ B) = P(grey\ |\ bucket\ B)\ .\ P(bucket\ B)}$$

  - So:
    - Stone being from bucket B = 3/7
    - A stone from bucket B being grey = 1/3
    - Therefore a random draw may result (1/3)*(3/7)=**1/7** times in a grey stone being drawn from bucket B.

- **Third thing** we need to know is **Bayes rule**.
- We write it as:

$$P(c_i|w) = \frac{P(w|ci).p(ci)}{p(w)}$$

We'll come to this equation once we have studied the following example.

Example:

1. *We have a set of sentences with labelled as abusive(1) and non-abusive(0).*
2. *We'll split each sentence into words and know the* ***probability of a word being responsible for the classification of the sentence****.*

   We assume that each word in the sentence is independent and has equal weightage.

   What that means is: ***jalebi*** *is as likely to appear individually in a sentence as it is to appear alongside the word* ***unhealthy*** *or* ***delicious.***

   This assumption is inherently naïve for a real world scenario but still this helps get pretty acceptable prediction
3. *Now if we encounter a sentence containing the words, we can predict the probability of the sentence being abusive(1) or non-abusive(0) using the already known data.*

Example Process: 7 Steps

1. We have a list of sentences labelled as abusive (1) or non-abusive(0)
   - "My dog has flea problems help please" - 0
   - "Maybe not take him to dog park stupid" - 1
   - "My dalmatian is so cute I love him" - 0
   - "stop posting stupid worthless garbage" - 1
   - " Mr licks ate my steak how to stop him" - 0
   - Quit buying worthless dog food stupid" - 1

2. We'll split these sentences into words and create a unique vocabulary set using the above data. (*Here we have a vocabulary set of 32 elements*)

['park', 'posting', 'flea', 'cute', 'ate', 'maybe', 'not', 'has', 'worthless', 'food', 'dalmation', 'I', 'steak', 'mr', 'quit', 'stop', 'garbage', 'to', 'buying', 'problems', 'licks', 'dog', 'is', 'how', 'love', 'take', 'him', 'please', 'stupid', 'my', 'so', 'help']

3. Now we'll calculate the contribution of each word towards classification of a sentence into our class 1 (Abusive).
   - For each labelled example sentence of this class, we'll create one vector of dimensions equal to vocabulary set's cardinality (32) and initially all magnitudes = 0.

     V = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]

     Example sentence: "my dog has flea problems help please"
   - Referring to the vocabulary set we'll put a 1 corresponding to words in our sentence.

['park', 'posting', 'flea', 'cute', 'ate', 'maybe', 'not', 'has', 'worthless', 'food', 'dalmation', 'I', 'steak', 'mr', 'quit', 'stop', 'garbage', 'to', 'buying', 'problems', 'licks', 'dog', 'is', 'how', 'love', 'take', 'him', 'please', 'stupid', 'my', 'so', 'help']

In the vocab set, position of "my=30", "dog=22", "has=8", "flea=3", "problem=20", "help=32" "please=28"

Our vector(V) becomes: [0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,1]

Similar vectors are generated for each of the labelled sentences in Class 1.

4. Now we'll do a vector addition of all the vectors coming from class 1 to obtain a sum vector for this class. We'll normalize this vector (by dividing all the magnitudes by total sum of all elements) to obtain a probability vector/function for class 1.

   This will look something like:

```
([0.        , 0.05263158, 0.        , 0.10526316, 0.        ,
  0.        , 0.        , 0.        , 0.05263158, 0.        ,
  0.        , 0.10526316, 0.05263158, 0.        , 0.05263158,
  0.05263158, 0.        , 0.05263158, 0.        , 0.15789474,
  0.        , 0.        , 0.05263158, 0.05263158, 0.        ,
  0.        , 0.        , 0.05263158, 0.05263158, 0.        ,
  0.05263158, 0.05263158])
```

This matrix is going to be our probability matrix for class 1 i.e. P_{C1}.

As can be observed, that the presence word "Stupid (index = 20) is very likely to classify a sentence as abusive.

5. In similar way we obtain $\mathbf{P_{C2}}$.

We'll use these probability matrices to predict the probability of a new test sentence belonging to either of these classes.

---

6. Any input sentence can be converted to a vector of 32 dimensions as done for training sentences in step 3. Any new words will contribute to error of our analysis.

Lets say we got a vector Z for a given test sentence.

---

7. We'll take a **dot product of Z and $\mathbf{P_{C1}}$** to know the **probability of it belonging to Class 1**. And we'll take a dot product of Z and P_{C2} to know the probability of it belonging to class 2.

We'll compare the outcomes and we'll classify the sentence on the basis of greatness of the respective dot products.

Practical Considerations:

1. We assumed that words in our data were independent of each other.
2. **Underflow** is one problem (decimal numbers becoming too small such that python regards them as 0): Can be solved by using logarithm of probabilities in our calculations.

   This makes sense because:

   a) `ln(a*b) = ln(a)+ln(b)`

   b) **There is observationally no difference in character of resultant**

   - **Where there is dip in function – there's dip in log**
   - **Where there is a rise in function – there's a rise in log**
   - **Peaks are at same value**

   Etc.

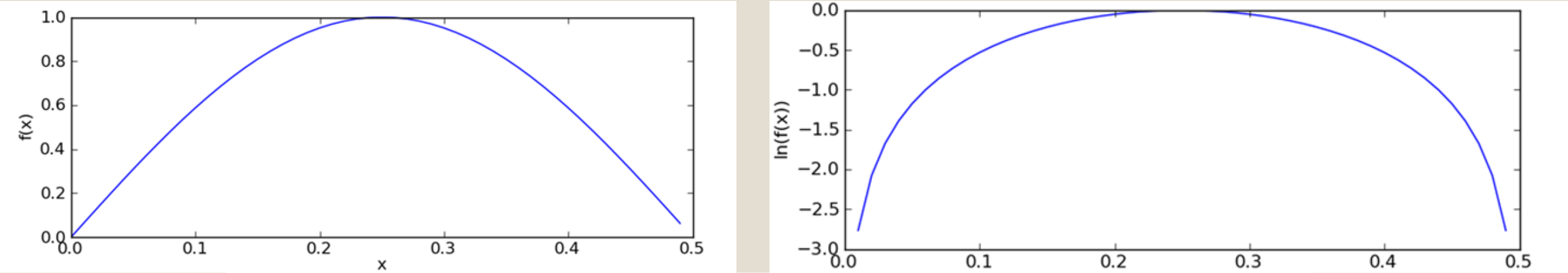

Figure 4.4 **Arbitrary functions `f(x)` and `ln(f(x))` increasing together. This shows that the natural log of a function can be used in place of a function when you're interested in finding the maximum value of that function.**

Thanks a lot.

Catch the rest of the series at-

https://github.com/satyapratheek/mltalk

Subhankar Mishra's Lab: www.niser.ac.in/~smishra