Naïve Bayes Classifier

Palacode Narayana Iyer Anantharaman

12 Aug 2016

Classification Problems with Naïve Bayes

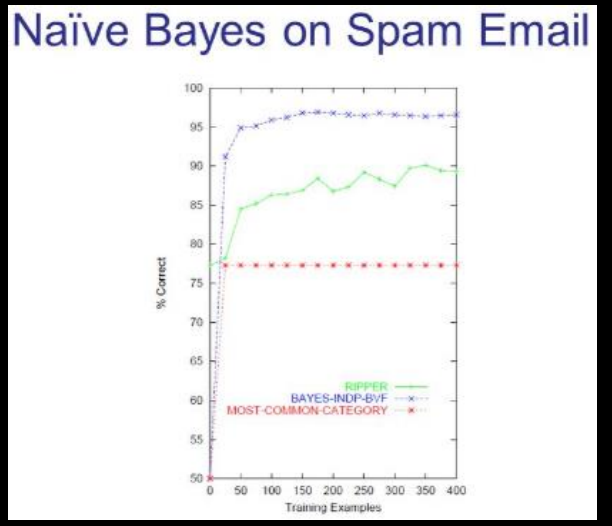
- Two step process
  - Build the model – this is the training process where we estimate model parameters

  - Use the model – here, we predict the output class given the inputs

When to use Naïve Bayes: Three Scenarios

- You have come up with a neat solution to a ML problem. Your manager wants you to do a quick demo to your CEO in the next couple of hours.

- You are assigned a classification problem similar to spam filtering. Your manager says: We need this feature in our next release, a less accurate model is okay.



- You have come up with a sophisticated deep learning based model. You submitted this for review and you are asked to benchmark your results against standard approaches

Naïve Bayes Classifier

A simple classifier model that is:

- Based on the Bayes theorem
- Uses Supervised Learning
- Easy to build
- Faster to train, compared to the other models
- Often used as a baseline classifier for benchmarking

Foundation: Bayes Theorem

$From\ Bayes\ theorem, we\ have\colon\ P(Y|X) = P(X|Y)P(Y)/P(X)$

Suppose Y represents the class variable and X_1, X_2, X_3, … X_n are inputs:

$$P(Y|X_1, \dots, X_n) = \frac{P(X_1, \dots, X_n|Y)\ \ P(Y)}{P(X_1, \dots, X_n)}$$

Assuming **$X_i \perp X_j$ given Y** for all i, j , we may write the above equation as:

$$P(Y|X_1, \dots, X_n) = \frac{P(X_1|Y)\ P(X_2|Y) \dots P(X_n|Y)\ \ P(Y)}{P(X_1, \dots, X_n)}$$ **(Naïve Assumption)**

$$P(Y|X_1, \dots, X_n) \propto P(Y) \prod_{i=1}^{n} P(X_i|Y)$$

$$\hat{Y} = argmax_y P(Y) \prod_{i=1}^{n} P(X_i|Y)$$

What is Naïve about Naïve Bayes?

- In many applications, treating one element in the input (X_i) independent of every other element in the input (X_j for all j) and also ignoring word order is quite a strong assumption

- Why?
  - The sentence: “day great is today a” is a jumbled form of “Today is a great day”. This suggests that the ordering of words is important for us to perform a semantic interpretation. NB classifier treats each word as independent and hence ignores the order, which in many cases will not hold.
  - Take a selfie of yourself, the picture looks great! What if we randomly shift the pixels around throughout the image? Though all pixels are still present in the modified image, their order is severely altered.

- But:
  - Despite the Naïve assumption, NB Classifier still works and produces accurate results for a number of applications!
    - Consider the problem of search using key words. Does the word order matter?

Estimating the model parameters

- Naïve Bayes Model: $\hat{Y} = argmax_y P(Y) \prod_{i=1}^{n} P(X_i|Y)$
- The model parameters are: $P(Y)$ and $P(X_i|Y)$ for all values of X_i
- Given a dataset that has several training examples, where each example has an input $(X_1, \ldots, X_n)$ and the expected target output (Y), we need to "learn" the model
- We can perform maximum likelihood estimates in order to determine model parameters

Naïve Bayes Classifier for the real world use cases

(Ref: Kaggle)

Kaggle Team <team@kaggle.com> Unsubscribe
to me

Aug 10 (1 day ago)

Expert interviews & top community code

View this email in your browser

kaggle newsletter

Hello!

This week we highlight the use of Bayes' theorem **to predict mobile check-in locations** in a fictional world created by Facebook. Also, will your job be usurped by robots? Watch **Anthony's TED Talk** to find out which jobs machine learning will replace, and why creativity will keep some occupations safe.

Kagglers in Berlin, don't miss a chance to **meetup** with other community members and competition winners next month.

Facebook V: Predicting Check Ins, Winner's Interview: 2nd Place, Markus Kliegl

Facebook's **5th recruiting** challenge asked Kagglers to use mobile coordinates to rank likely check in locations in a 10km by 10km artificial world. The 2nd place winner, **Markus Kliegl**, describes his elegant use of Bayes' theorem and semi-supervised learning.

Naïve Bayes Case Study (Ref: Kaggle)

Let's get technical

What preprocessing and supervised learning methods did you use?

The overall approach was to use Bayes' theorem: Given a particular data point (x, y, accuracy, time), I would try to compute for a suitably narrowed set of candidate places the probability

$$P(place|x, y, accuracy, time) \propto P(x, y, accuracy, time|place)P(place)\,,$$

and rank the places accordingly. A la Naive Bayes, I further approximated

P(x, y, accuracy, time | place) as

$$P(x, y, accuracy, time|place) \approx P(x, y|place)\cdot$$

$$P(accuracy|place)\cdot P(time\,of\,day|place)\cdot$$

$$P(day\,of\,week|place)\,.$$

I decided on this decomposition after a mixture of exploratory analysis and simply trying out different assumptions on the independence of variables on a validation set.

One challenge given the data size was to efficiently learn the various conditional distributions on the right-hand side. Inspired by the effectiveness of ZFTurbo's "Mad Scripts Battle" kernel early in the competition, I decided to start by just learning these distributions using histograms.

Document Classification With Naïve Bayes

- Document (or text in our discussion today) classification assigns a class label to the given document. Formally:
  - Given the input as document d and a set of classes $C = \{c_1, c_2, \dots, c_n\}$, predict a class $c \in C$

- Example:
  - Gmail categorizes the incoming mails in to Primary, Social, Promotions, Junk – we can define:

$$C = \{Primary, Social, Promotions, Junk\}$$

  - Assign a class label $c \in C$ for every incoming mail d

- The term document could be a plain text or even a compound multimedia document. We use the term document to refer to any entity that can be classified.

Can we build rule based models to do this?

- For instance, in email categorization, one may apply a set of if-then-else rules and determine the class of input mail
- With well written rules, in general, one can get high precision but often low recall
- Drafting a set of comprehensive rules is difficult and expensive as they need expert knowledge
- Example: Suppose I receive a mail from flipkart, should that be classified as a promotion or primary?
  - It depends!

Supervised Machine Learning

- Input
  - A document d, consisting of word tokens $w_1, w_1, \ldots, w_j$
  - A finite set of classes: $C = \{c_1, c_2, \ldots, c_n\}$
  - The training dataset: $D_{train} = \{(d_1, c_1), (d_2, c_2), \ldots, (d_m, c_m)\}$

- Output
  - A model M such that: $M: d \rightarrow c$

Bag of words representation

- A document can be considered to be an ordered sequence of words

$$M(\text{"I love my Samsung Galaxy Grand 2"}) = c$$

- Naïve Bayes classifier ignores the word order and correlations and hence we may represent a document d as bag of words or unigrams

- In a typical English sentence, there may be many words that are required for grammatical purposes and may not contribute to the classification decision.

$$M(\text{"love Samsung Galaxy Grand"}) = c$$

- We can do some pre-processing and remove such words before sending the document to the classifier

Text Classification with Multinomial Naïve Bayes

- Recall: $\hat{Y} = argmax_y P(Y) \prod_{i=1}^{n} P(X_i|Y)$ where the model parameters are: $P(Y)$ and $P(X_i|Y)$ for all values of X_i

- For the document classification problem with bag of words model, X_i is a word in the document and Y is the document class

- Estimate the model parameters as below and save the model as a table T:
  - For each class $c \in C$, determine
    - MLE: $P(C = c) = \frac{Number\ of\ documents\ that\ are\ labelled\ c}{Total\ Number\ of\ documents}$
  - For each word $w_i \in V$ and a $class$ $c_j \in C, MLE\ P(w_i|c_j) = \frac{count(w_i,c_j)}{\sum_{w \in V} count(w,c_j)}$

- Prediction: Given a new input, generate the word tokens using the **same procedure** used for training. Retrieve the model values from the Table T and compute $\hat{Y}$

Are we done? Not yet ☹

- What happens if $count(w_i, c_j)$ = 0 ?

- This can happen if the new unseen input has a word w_i, that was not encountered in the training data.

- Example: The training data contains “I love my Samsung Galaxy Grand 2” but doesn’t have the word, say, “adore”. If the unseen input is: “I adore my Samsung Galaxy Grand 2”, the entire probability computation will be zero!

Laplace Smoothing (Add 1)

- Assume any word in the vocabulary has occurred at least once
- This assumption results in the estimation:

$$P(w_i|c_j) = \frac{count(w_i, c_j) + 1}{(\sum_{w \in V} count(w, c_j)) + |V|}$$

- The above ensures that the probabilities don't go to zero
- What happens when you encounter a word that is not in the vocabulary?

Variants of Naïve Bayes

- In the previous slides we showed the MLE probability computation to be based on counts of words in each document. This is called the multinomial model.

- Multinomial is a natural fit when we solve topic classification kind of problems
  - E.g. Consider the problem of classifying a given article in to Scientific, Business, Sports.

- Sometimes, just the presence or absence of a given word in a document is adequate in order to classify. We may choose a Binarized Naïve Bayes in such cases.
  - E.g Consider the problem of sentiment analysis. If there is a word “fantastic”, it doesn’t need to be repeated in the same document in order for us to conclude the polarity of the sentiment.

- A number of applications may involve features that are real valued. We can use a Gaussian (or some other) variant for these.

Multinomial Naïve Bayes

- In a multinomial classification model, the frequency of occurrence of each word in the document is taken in to account (instead of presence/absence)

- Compute prior for classes using maximum likelihood estimates

- The algorithm to compute $P(w_i|c_j)$ is:
  - Concatenate all documents that have the class c_j, let it be text_j
  - Let n be the number of tokens in text_j and α be the constant used for smoothing
  - For each word w_k in the vocabulary, let n_k be the number of occurrences of w_k in the text_j

$$P(w_i|c_j) = \frac{n_k + \alpha}{n + \alpha|V|}$$

Binarized Multinomial Naïve Bayes

- In a binarized multinomial classification model, we count only the presence or absence of a given word (or feature) in the given document as opposed to using the frequency. That is, we clamp the word count of a word w in a document j to 1

- Compute prior for classes using maximum likelihood estimates

- The algorithm to compute $P(w_i|c_j)$ is:
  - **In each document d, keep only one instance of the given word w (Remove duplicates)**
  - Concatenate all documents that have the class c_j, let it be text_j
  - Let n be the number of tokens in text_j and α be the constant used for smoothing
  - For each word w_k in the vocabulary, let n_k be the number of occurrences of w_k in the text_j

$$P(w_i|c_j) = \frac{n_k + \alpha}{n + \alpha|V|}$$

Gaussian Naïve Bayes

- So far, we have looked at text and dealt with word occurrence counts that are discrete values

- What happens when the features are continuous valued or even vectors of continuous values? E.g images with RGB values?

- Gaussian Naïve Bayes is useful to classify such inputs

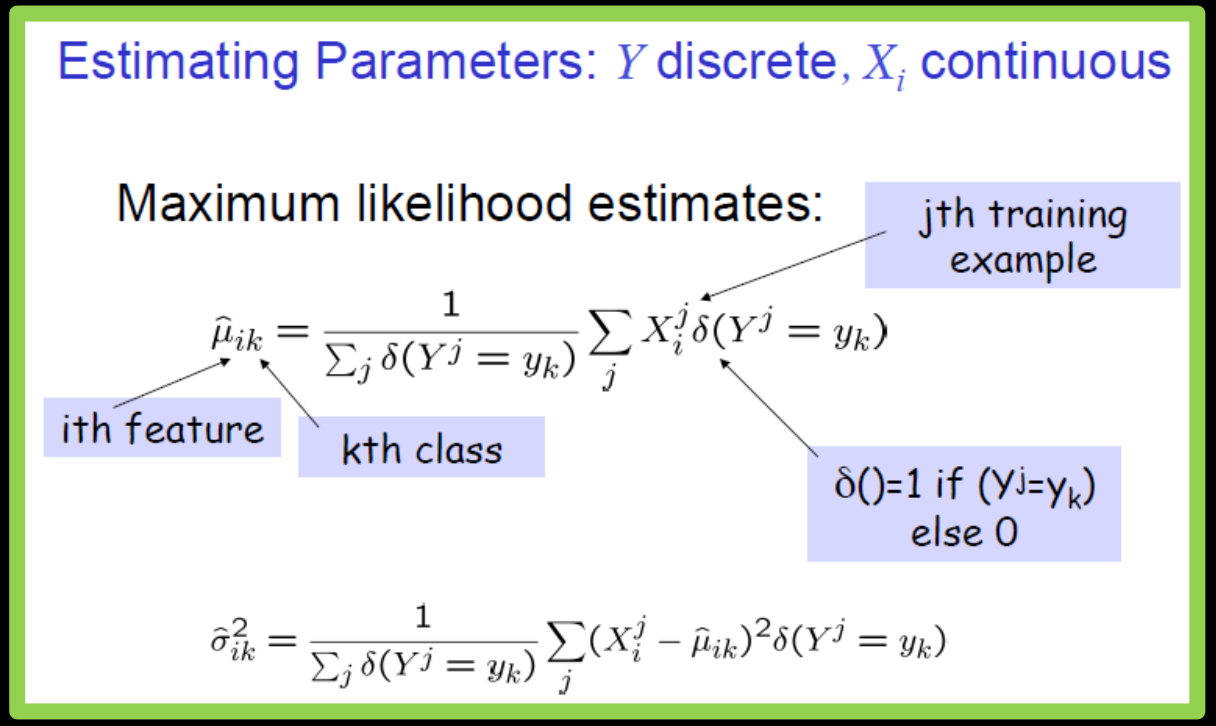
$$P(X_n|Y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp(-\frac{\left(x_i - \mu_y\right)^2}{2\sigma_y^2})$$

- Estimate the parameters $\sigma_y \; and \; \mu_y$ using maximum likelihood

Estimating Parameters (Ref: T Mitchelle)

How many parameters must we estimate for Gaussian Naïve Bayes if Y has k possible values:

$$X = X_1 X_2 \ldots X_n$$



Gaussian Naïve Bayes: Example

1	Brand	Product	cores	Clock	Size	Internal	RAM	MP	Price
2	Samsung	Galaxy Star S5280	1	1	3	4	0.5	2	
3	Samsung	Galaxy Note II N7100	4	1.6	5.5	64	2	8	
4	Samsung	Google Nexus S 4G	1	1	4	16	0.5	5	
5	Samsung	ATIV S Neo	2	1.4	4.77	16	1	8	
6	Samsung	Galaxy S II Epic 4G Touch	2	1.2	4.52	16	1	8	
7	Samsung	Galaxy Note Edge	4	2.7	5.6	64	3	16	
8	Samsung	Galaxy A3	4	1.2	4.5	16	1.5	8	

- Suppose we are required to predict the price range (high_end, mid_range, low_end) of a mobile phone given its specifications.
- We observe that some elements in the specification (e.g screen size) are continuous variables.
- We can either discretize these elements and use discrete NB classifier or we can directly use a Gaussian NB

Practical Considerations

- Probability computations in a joint distribution involve multiplying many terms that are small fractions. This might sometime cause underflow errors. Use log probabilities to avoid this issue

- Use the distribution that is natural to the problem on hand
  - The choice of the distribution to use is your decision. There is no rule that says you should use Gaussian all the time!
  - You can discretize continuous variables so that you can use Binarized, Bernoulli or Multinomial discrete Naïve Bayes. But you might lose fidelity due to discretization.

- Exercise judgement while choosing the features, you can minimize the data required by removing redundant features

Case Study: Accurate Searching on Twitter

- Twitter's search API allows keywords based search
- Suppose we search Twitter with the keyword "congress", we might end up getting tweets that pertain to Indian National Congress, Mobile World Congress, American Congress, Science Congress and so on.
- A narrow search using "exact" search phrase would improve precision but will miss many relevant and interesting tweets
- Is there a way to search the Twitter such that we get precise matches without missing interesting and relevant tweets?

Summary

- Despite the naïve assumptions, Naïve Bayes classifier is pretty useful. Do not skip it in favour of complex models without evaluating it for your application. You may be in for surprise!

- There are many variants of Naïve Bayes Classifier, the common thing about them is that all are based on Bayes theorem and make same assumptions.

- Choose binarized model if number of occurrences of a given word do not contribute to the classification process.

- If the features are continuous variables, use Gaussian Naïve Bayes or perform discretization if you get good accuracy

Code Walkthrough