Using Naive Bayes As Discriminative Classifier

Published on Dec, 2020

Elie Azeraf[1] Emmanuel Monfrini[2] Wojciech Pieczynski[2]

[1]Watson Department, IBM GSB France
[2]SAMOVAR, Telecom SudParis, Institut Polytechnique de Paris

AI5002
Research Paper Presentation
Tuhin Dutta (AI21MTECH02002)

Abstract

- For classification tasks, probabilistic models can be categorized into two disjoint classes: generative or discriminative.

Abstract

- For classification tasks, probabilistic models can be categorized into two disjoint classes: generative or discriminative.
- Generative classifiers, like the Naive Bayes or the Hidden Markov Model (HMM), need the computation of the joint probability $\Pr(X, Y)$, before using the Bayes rule to compute $\Pr(X|Y)$.

Abstract

- For classification tasks, probabilistic models can be categorized into two disjoint classes: generative or discriminative.
- Generative classifiers, like the Naive Bayes or the Hidden Markov Model (HMM), need the computation of the joint probability $\Pr(X, Y)$, before using the Bayes rule to compute $\Pr(X|Y)$.
- Discriminative classifiers compute $\Pr(X|Y)$ directly regardless of the observations' law and they are used extensively with models such as Logistic Regression.

Abstract

- For classification tasks, probabilistic models can be categorized into two disjoint classes: generative or discriminative.
- Generative classifiers, like the Naive Bayes or the Hidden Markov Model (HMM), need the computation of the joint probability $\Pr(X, Y)$, before using the Bayes rule to compute $\Pr(X|Y)$.
- Discriminative classifiers compute $\Pr(X|Y)$ directly regardless of the observations' law and they are used extensively with models such as Logistic Regression.
- After Entropic Forward-Backward algorithm proved that the HMM being a generative model can also match the discriminative classifier definition, in this paper we show that Logistic Regression can be viewed as a particular case of the Naive Bayes used in a discriminative way.

Table of Contents

1.1 Review Of Important Prerequisite Concepts

1. Generative Classifier
2. Discriminative Classifier
3. Hidden Markov Model
4. Entropic Forward-Backward Algorithm
5. Gradient Descent Algorithm

1.1 Review Of Important Prerequisite Concepts

1.1.1 Generative Classifier

Generative Classifiers tries to model class, i.e., what are the features of the class. When a new observation is given to these classifiers, it tries to predict which class would have most likely generated the given observation. Mathematically, generative models try to learn the joint probability distribution, $\Pr(X, Y)$, of the inputs X and label Y, and make their prediction using Bayes rule to calculate the conditional probability, $\Pr(Y|X)$, and then picking a most likely label. Thus, it tries to learn the actual distribution of the class. An example of such classifiers is Naive Bayes.

1.1 Review Of Important Prerequisite Concepts

1.1.2 Discriminative Classifier

Discriminative Classifiers learn what the features in the input are most useful to distinguish between the various possible classes. So, if given images of dogs and cats, and all the dog images have a collar, the discriminative models will learn that having a collar means the image is of dog. An example of a discriminative classifier is logistic regression. Mathematically, it directly calculates the posterior probability $\Pr(Y|X)$ or learn a direct map from input X to label Y. So, these models try to learn the decision boundary for the model.

1.1 Review Of Important Prerequisite Concepts

1.1.3 Hidden Markov Model (HMMs)

A Markov chain is a model that tells us something about the probabilities of sequences of random variables, states, each of which can take on values from some set. The HMM is based on augmenting the Markov chain. HMMs are probabilistic sequence labeling algorithms which assigns a label to each unit in a sequence, thus mapping a sequence of observations to a sequence of labels of the same length. We know Markov Chains are used to compute probability for a sequence of observable events but when events are hidden we make use of HMMs. A (HMM) allows us to talk about both observed events (like words that we see in the input) and hidden events (like part-of-speech tags).

1.1 Review Of Important Prerequisite Concepts

1.1.3 Hidden Markov Model (HMMs)

An HMM is specified by the following components:

$$\begin{aligned} Q = q_1 q_2 ... q_N - & \quad \textit{a set of } N \textit{ states} \\ A = a_{11} ... a_{ij} ... a_{NN} - & \quad \textit{a transition probability matrix} \\ O = o_1 o_2 ... o_T - & \quad \textit{a sequence of } T \textit{ observations} \\ B = b_i(o_t) - & \quad \textit{a sequence of observational likelihoods} \\ \pi = \pi_1, \pi_2, .., \pi_N - & \quad \textit{initial probability distribution over states} \end{aligned} \tag{1}$$

1.1 Review Of Important Prerequisite Concepts

1.1.3 Hidden Markov Model (HMMs)

Two of the assumptions taken by HMMs are

- Markov assumption: The probability of a particular state depends on the previous state.

$$\Pr(q_i \mid q_1, .., q_{i-1}) = \Pr(q_i \mid q_{i-1}) \tag{2}$$

- Output Independence: The probability of an output observation o_i depends only on the state that produced the observation q_i and not on any other states or any other observations.

$$\Pr(o_i \mid q_1, .., q_i, .., q_T, o_1, ..o_i, .., o_T) = \Pr(o_i \mid q_i) \tag{3}$$

1.1 Review Of Important Prerequisite Concepts

1.1.3 Hidden Markov Model (HMMs)

When talking about HMMs (Hidden Markov Models) there are generally 3 problems to be considered:

- Evaluation problem
- Decoding problem
- Training problem

1.1.3.1 Evaluation Problem

It answers the question: what is the probability that a particular sequence of symbols is produced by a particular model? For evaluation we use two algorithms: the forward algorithm or the backwards algorithm (NOT to be confused with the forward-backward algorithm).

1.1 Review Of Important Prerequisite Concepts

1.1.3 Hidden Markov Model (HMMs)

1.1.3.2 Decoding Problem

It answers the question: Given a sequence of symbols (your observations) and a model, what is the most likely sequence of states that produced the sequence. For decoding we use the Viterbi algorithm.

1.1.3.3 Training Problem

It answers the question: Given a model structure and a set of sequences, find the model that best fits the data. For this problem we can use the following 3 algorithms:

- MLE (maximum likelihood estimation)
- Viterbi training(NOT to be confused with Viterbi decoding)
- Baum Welch = forward-backward algorithm

1.1 Review Of Important Prerequisite Concepts

1.1.3 Hidden Markov Model (HMMs)

To sum it up, we use the Viterbi algorithm for the decoding problem and Baum Welch/Forward-backward when we train our model on a set of sequences.

1.1.3.4 Viterbi decoding Vs Forward-backward algorithm

Forward-Backward gives marginal probability for each individual state, Viterbi gives probability of the most likely sequence of states. For instance if our HMM task is to predict sunny vs. rainy weather for each day, Forward Backward would tell us the probability of it being "sunny" for each day, Viterbi would give the most likely sequence of sunny/rainy days, and the probability of this sequence.

1.1 Review Of Important Prerequisite Concepts

1.1.4 Entropic Forward-Backward Algorithm

EFB algorithms are discriminative classifiers of HMMs. We consider a homogeneous HMM with the following notations:

- $\pi(i) = \Pr(x_t = \lambda_i)$
- $a_i(j) = \Pr(x_{t+1} = \lambda_j \mid x_t = \lambda_i)$
- $b_i(y_t) = \Pr(y_t \mid x_t = \lambda_i)$
- $L_{yt}(i) = \Pr(x_t = \lambda_i \mid y_t)$

The Entropic Forward-Backward algorithm consists in computing, $\Pr(x_t = \lambda_i \mid y_{1:T})$.

1.1 Review Of Important Prerequisite Concepts

1.1.4 Entropic Forward-Backward Algorithm

$$\Pr\left(x_t = \lambda_i \mid y_{1:T}\right) = \frac{\alpha_t^E(i)\ \beta_t^E(i)}{\sum_{j=1}^{N} \alpha_t^E(j)\ \beta_t^E(j)} \tag{4}$$

Entropic forward probabilities α^E is computed as:

$$\begin{aligned} \alpha_1^E(i) &= L_{y_1}(i) \\ \alpha_{t+1}^E(i) &= \frac{L_{y_{t+1}(i)}}{\pi(i)} \sum_{j=1}^{N} \alpha_t^E(j)\ a_j(i) \end{aligned} \tag{5}$$

1.1 Review Of Important Prerequisite Concepts

1.1.4 Entropic Forward-Backward Algorithm

and Entropic backward β^E:

$$\beta_T^E(i) = 1$$
$$\beta_t^E(i) = \sum_{j=1}^{N} \frac{L_{y_{t+1}}(j)}{\pi(j)} \beta_{t+1}^E(j)\ a_i(j) \tag{6}$$

This algorithm allows computing the posterior distribution directly, without using the joint distribution, or the observations' one. It also allows training the HMM with optimization algorithms as the gradient descent. Therefore, in this case, the HMM matches the discriminative classifier's definition.

1.1 Review Of Important Prerequisite Concepts

1.1.5 Gradient Descent Algorithm

Gradient descent is a way to minimize an objective function $J(\theta)$ parameterized by a model's parameters $\theta \in \mathbb{R}^d$ by updating the parameters in the opposite direction of the gradient of the objective function $\bigtriangledown_\theta J(\theta)$ w.r.t to the parameters. The learning rate η determines the size of steps we take to reach a local minimum. In other words, we follow the direction of the slope of the surface created by the objective function downhill until we reach a valley.

1.2 About

A recent work about HMM presents the Entropic Forward-Backward algorithm allows computing the posterior distribution $\Pr(x_t = \lambda_i \mid y_{1:T})$ directly, without using the joint distribution, or the observations' one. It also allows training the HMM with optimization algorithms as the gradient descent. Therefore, in this case, the HMM matches the discriminative classifier's definition. This leads to question this categorization of probabilistic classifiers, which is the aim of the paper.

1.3 Objectives

We present the objectives of the paper as:

- Naive Bayes, a popular generative model, can also match the definition of a discriminative one. Hence traditional definitions of both discriminative and generative models do not necessarily lead to disjoint categories.
- The notion of Generative-Discriminative pairs linking a generative model with its discriminative counterpart, as for example, the Naive Bayes - Logistic Regression pair, or the Hidden Markov Model - Conditional Random Fields one.

Table of Contents

2.1 Compute Label Probabilities

2.1.1 Using joint or observations' probability

Considering observations $y_{1:T} = (y_1, .., y_T)$ and a hidden variable x taking its values in Λ_x. It models the joint probability of $(x, y_{1:T})$ as:

$$\Pr(x, y_{1:T}) = \Pr(x) \Pi_{t=1}^{T} \Pr(y_t \mid x) \tag{7}$$

In the generative way, for each $\lambda_i \in \Lambda_x$, Naive Bayes distributed $\Pr(x = \lambda_i \mid y_{1:T})$ is computed as:

$$\Pr(x = \lambda_i \mid y_{1:T}) = \frac{\Pr(x = \lambda_i, y_{1:T})}{\sum_{\lambda_j \in \Lambda_x} \Pr(x = \lambda_j, y_{1:T})} \tag{8}$$

2.1 Compute Label Probabilities

2.1.1 Using joint or observations' probability

$$\Pr(x = \lambda_i \mid y_{1:T}) = \frac{\Pr(x = \lambda_i)\ \Pi_{t=1}^{T} \Pr(y_t \mid x = \lambda_i)}{\sum_{\lambda_j \in \Lambda_x} \Pr(x = \lambda_j) \Pi_{t=1}^{T} \Pr(y_t \mid x = \lambda_j)} \tag{9}$$

We see that first the joint probability $\Pr(x = \lambda_i, y_{1:T})$ is computed, and then the posterior $\Pr(x = \lambda_i \mid y_{1:T})$. The difficulties arise as the number of feature increases are:

- Assumption of independent predictor features
- Zero frequency

2.1 Compute Label Probabilities

2.1.2 Computing directly $\Pr(x = \lambda_i \mid y_{1:T})$

To simplify notations, we assume:

$$\begin{aligned} \pi(i) &= \Pr(x = \lambda_i) \\ b_i^{(t)}(y) &= \Pr(y_t \mid x_t = \lambda_i) \\ L_{y_t}^{(t)}(i) &= \Pr(x = \lambda_i \mid y_t) \end{aligned} \tag{10}$$

2.1 Compute Label Probabilities

2.1.2 Proof of computing $\Pr(x = \lambda_i \mid y_{1:T})$ directly

Re-writing (9) using (10),

$$\Pr(x = \lambda_i \mid y_{1:T}) = \frac{\pi(i) \prod_{t=1}^{T} \frac{\Pr(y_t,\ x = \lambda_i)}{\pi(i)}}{\sum_{j=1}^{N} \pi(j) \prod_{t=1}^{T} \frac{\Pr(y_t,\ x = \lambda_j)}{\pi(j)}}$$

$$\Pr(x = \lambda_i \mid y_{1:T}) = \frac{\pi(i)^{1-T} \prod_{t=1}^{T} \Pr(y_t,\ x = \lambda_i) \frac{\Pr(y_t)}{\Pr(y_t)}}{\sum_{j=1}^{N} \pi(j)^{1-T} \prod_{t=1}^{T} \Pr(y_t,\ x = \lambda_j) \frac{\Pr(y_t)}{\Pr(y_t)}} \tag{11}$$

2.1 Compute Label Probabilities

2.1.2 Computing directly $\Pr(x = \lambda_i \mid y_{1:T})$

Re-writing (9) using notations from (10),

$$\Pr(x = \lambda_i \mid y_{1:T}) = \frac{\pi(i)^{1-T} \prod_{t=1}^{T} \Pr(x = \lambda_i \mid y_t) \Pr(y_t)}{\sum_{j=1}^{N} \pi(j)^{1-T} \prod_{t=1}^{T} \Pr(x = \lambda_j \mid y_t) \Pr(y_t)}$$

$$\Pr(x = \lambda_i \mid y_{1:T}) = \frac{\pi(i)^{1-T} \prod_{t=1}^{T} L_{y_t}^{(t)}(i)}{\sum_{j=1}^{N} \pi(j)^{1-T} \prod_{t=1}^{T} L_{y_t}^{(t)}(j)} \tag{12}$$

Now, for each $\lambda_i \in \Lambda_x$, we set:

$$\delta(i) = \pi(i)^{1-T} \prod_{t=1}^{T} L_{y_t}^{(t)}(i) \tag{13}$$

2.1 Compute Label Probabilities

2.1.2 Computing directly $\Pr(x = \lambda_i \mid y_{1:T})$

From (9) we can write the joint probability in terms of $\delta(i)$ as shown below:

$$\Pr(x = \lambda_i,\ y_{1:T}) = \Pr(x = \lambda_i)\, \Pi_{t=1}^{T} \Pr(y_t \mid x = \lambda_i)$$

$$\Pr(x = \lambda_i,\ y_{1:T}) = \pi(i)\ \Pi_{t=1}^{T}\ b_i^{(t)}(y)$$

$$\Pr(x = \lambda_i,\ y_{1:T}) = \pi(i)\ \Pi_{t=1}^{T}\ \frac{\Pr(y_t,\ x = \lambda_i)}{\Pr(x = \lambda_i)} \tag{14}$$

$$\Pr(x = \lambda_i,\ y_{1:T}) = \pi(i)^{1-T}\ \Pi_{t=1}^{T}\ \Pr(y_t)\, L_{y_t}^{(t)}(i)$$

$$\Pr(x = \lambda_i,\ y_{1:T}) = \delta(i)\ \Pi_{t=1}^{T}\ \Pr(y_t)\ \ [\textit{Using}\ (13)]$$

2.1 Compute Label Probabilities

2.1.2 Computing directly $\Pr(x = \lambda_i \mid y_{1:T})$

Now, we show that one can use the Naive Bayes classifier by computing $\Pr(x = \lambda_i \mid y_{1:T})$ directly without using joint, i.e $\Pr(x = \lambda_i,\ y_{1:T})$, or observational, i.e $\Pr(y_t \mid x_t = \lambda_i)$, probability:

$$\begin{aligned} \Pr(x = \lambda_i \mid y_{1:T}) &= \frac{\Pr(x = \lambda_i,\ y_{1:T})}{\sum_{j=1}^{N} \Pr(x = \lambda_j,\ y_{1:T})} \\ \Pr(x = \lambda_i \mid y_{1:T}) &= \frac{\delta(i)}{\sum_{j=1}^{N}\ \delta(j)} \end{aligned} \tag{15}$$

Table of Contents

3.1 The Classifier

Logistic Regression is a probabilistic graphical model which can be trained using gradient descent algorithm. It is considered as a discriminative model. For each t, the multinomial logistic regression computes $\Pr(x \mid y_{1:T})$ as:

$$\Pr(x = \lambda_i \mid y_{1:T}) = \frac{\exp(W_i \cdot y_{1:T} + b_i)}{\sum_{j=1}^{N} \exp(W_j \cdot y_{1:T} + b_j)} \tag{16}$$

where for each $i \in \{1, ..., N\}$,
$W_i \in \mathbb{R}^T$, $b_i \in \mathbb{R}$, *and* $\cdot$ *denotes the matrix product*.

3.2 Particular case of the Naive Bayes

Logistic Regression with the notion of Generative – Discriminative pairs is linked with Naive Bayes. However, we are going to show that Logistic Regression is a particular case of the Naive Bayes.

Proposition

Let us consider a Naive Bayes classifier used in a discriminative way with (12). If, for each $i \in \{1, .., N\}$, $t \in 1, ..., T$:

$$L_{y_t}^{(t)}(i) = \frac{\exp\left(a_i^{(t)} y_t + c_i^{(t)}\right)}{\sum_{k=1}^{N} \exp\left(a_k^{(t)} y_t + c_k^{(t)}\right)} \tag{17}$$

with $a_i^{(t)} \in \mathbb{R}$ and $c_i^{(t)} \in \mathbb{R}$, then this classifier is a Logistic Regression.

3.2 Particular case of the Naive Bayes

We start from (12), for each $\lambda_i \in \Lambda_x$:

Proof

$$\begin{aligned}
\Pr(x = \lambda_i \mid y_{1:T}) &= \frac{\pi(i)^{1-T} \prod_{t=1}^{T} L_{y_t}^{(t)}(i)}{\sum_{j=1}^{N} \pi(j)^{1-T} \prod_{t=1}^{T} L_{y_t}^{(t)}(j)} \\
&= \frac{\pi(i)^{1-T} \prod_{t=1}^{T} \frac{\exp\left(a_i^{(t)} y_t + c_i^{(t)}\right)}{\sum_{k=1}^{N} \exp\left(a_k^{(t)} y_t + c_k^{(t)}\right)}}{\sum_{j=1}^{N} \pi(j)^{1-T} \prod_{t=1}^{T} \frac{\exp\left(a_j^{(t)} y_t + c_j^{(t)}\right)}{\sum_{k=1}^{N} \exp\left(a_k^{(t)} y_t + c_k^{(t)}\right)}}
\end{aligned} \tag{18}$$

3.2 Particular case of the Naive Bayes

Proof contd

$$
\begin{aligned}
&= \frac{\pi(i)^{1-T} \prod_{t=1}^{T} \exp\left(a_i^{(t)} y_t + c_i^{(t)}\right)}{\sum_{j=1}^{N} \pi(j)^{1-T} \prod_{t=1}^{T} \exp\left(a_j^{(t)} y_t + c_j^{(t)}\right)} \\
&= \frac{\exp\left((1-T)\log\left(\pi(i)\right) + \sum_{t=1}^{T}(a_i^{(t)} y_t + c_i^{(t)})\right)}{\sum_{j=1}^{N} \exp\left((1-T)\log\left(\pi(j)\right) + \sum_{t=1}^{T}(a_j^{(t)} y_t + c_j^{(t)})\right)}
\end{aligned} \tag{18}
$$

3.2 Particular case of the Naive Bayes

Proof contd

We set and verify that $b_i \in \mathbb{R}$ and $W_i \in \mathbb{R}^T$

- $b_i = (1 - T) \log \pi(i) + \sum_{t=1}^{T} c_i^{(t)}$
- $W_i = [\, a_i^{(1)}, \, a_i^{(2)}, \, .., \, a_i^{(T)}]$

Finally, we have proved the proposition as shown below:

$$\Pr(x = \lambda_i \mid y_{1:T}) = \frac{\exp(W_i \cdot y_{1:T} + b_i)}{\sum_{j=1}^{N} \exp(W_j \cdot y_{1:T} + b_j)} \tag{19}$$

This shows that one can view the Logistic Regression as a particular case of the Naive Bayes used in a discriminative way.

Table of Contents

4.1 Interchangeable Generative-Discriminative Models

This paper presents how to use the Naive Bayes as a discriminative model. A practical consequence is that it can be used in both generative and discriminative ways; the latter allowing arbitrary feature consideration. Moreover, we show that the Logistic Regression, usually presented as the discriminative counterpart of the Naive Bayes, can be seen as a particular case of the latter used in a discriminative way. A general conclusion is that the usual definitions of generative and discriminative models do not lead to disjoint families; and there are models that simultaneously satisfy both of them.

4.2 Future Work

An interesting perspective would consist in extending ideas of the paper to other popular generative models and examining whether they can also be used in a discriminative way.