IntroML Tutorial: Generative Modeling & Decision Theory

Marc Roeschlin

ETH Zurich

13 May 2019

A supervised classification problem

- Let $\mathcal{Y} = \{0, 1\}$ be the set of labels and $\mathcal{X} = \mathbb{R}^d$ a d-dimensional features space.
- Training set $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ of n labeled examples where $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}$.
- Goal: Predict most likely class y_j for a new data point $\mathbf{x}_j$

  OR
- Goal: Compute probabilities for making decisions (**Decision theory**)

Decision theory

- Act under uncertainty
- We are given:
  - Class-conditional distributions $P(y|\mathbf{x})$
  - Set of actions $\mathcal{A}$
  - Cost function that rewards/punishes our actions $C : \mathcal{Y} \times \mathcal{A} \to \mathbb{R}$

Bayesian Decision Theory

We choose the action $a^\star$ according to

$$a^\star = \operatorname{argmin}_{a \in \mathcal{A}} \mathbb{E}_y \left[C(y, a) | \mathbf{x} \right]$$

Bayesian Decision Theory

- Minimize the expected cost / minimize the risk
  (cost function, loss function, utility function)
- Infers optimal decision if $P(y|\mathbf{x})$ reflects true distribution

$$a^\star = \operatorname{argmin}_{a \in \mathcal{A}} \mathbb{E}_y \left[C(y, a) | \mathbf{x} \right] = \sum_y P(Y = y|\mathbf{x}) \cdot C(y, a)$$

For classification, expectation is computed assuming certain posterior distribution over the classes, i.e., $P(Y = y|\mathbf{x})$

Decision Theory: Homework

Task 1

You would like to classify whether an X-ray result is cancerous or normal. The cost for a correct classification is 0 and the cost for predicting that the X-ray is normal when the true label is cancer is 1000, and the cost for predicting the X-ray is cancerous when the true label is normal is 1.

Decision Theory: Homework

- Action set $\mathcal{A} = \{+1, -1\}$ and labels $\mathcal{Y} = \{+1, -1\}$, i.e., *cancer* and *no-cancer/normal*
- Cost function: $C(y, a) = \begin{cases} 0 & y = a \\ 1 & y = -1, a = +1 \\ 1000 & y = +1, a = -1 \end{cases}$
- Assumption: $P(Y = -1|X) = (1 - p)$ and $P(Y = +1|X) = p$

Decision Theory: Asymmetric Cost

- Action set $\mathcal{A} = \{+1, -1\}$ and labels $\mathcal{Y} = \{+1, -1\}$, i.e., *cancer* and *no-cancer*
- Cost function: $C(y, a) = \begin{cases} 0 & y = a \\ c_{FP} & y = -1, a = +1 \\ c_{FN} & y = +1, a = -1 \end{cases}$
  where c_{FP} is a *false positive*, i.e., we think it is cancerous, but it's not. c_{FN} is a *false negative* event: We believe the X-ray is normal, but patient has cancer.
- Assumption: $P(Y = -1|X) = p$ and $P(Y = +1|X) = (1 - p)$
- Also: $c_{FP} = 1$ and $c_{FN} = 1000$, i.e., highly asymmetric

Decision Theory: Asymmetric Cost

Assumption: $P(Y = -1|X) = (1 - p)$ and $P(Y = +1|X) = p$, $c_{FP} = 1$ and $c_{FN} = 1000$

Action that minimizes cost:
(we compute expected loss for every possible action)

$$\begin{aligned}\mathbb{E}_y\left[C(y, +1)|\mathbf{x}\right] &= P(y = -1|x) \cdot c_{FP} + P(y = +1|x) \cdot 0 \\ &= (1 - p) \cdot 1 + p \cdot 0\end{aligned}$$

$$\begin{aligned}\mathbb{E}_y\left[C(y, -1)|\mathbf{x}\right] &= P(y = -1|x) \cdot 0 + P(y = +1|x) \cdot c_{FN} \\ &= (1 - p) \cdot 0 + p \cdot 1000\end{aligned}$$

Decision Theory: Asymmetric Cost

Action that minimizes cost:

$$\mathbb{E}_y \left[C(y, +1) | \mathbf{x} \right] = (1 - p)$$

$$\mathbb{E}_y \left[C(y, -1) | \mathbf{x} \right] = p \cdot 1000$$

Ideally, we therefore predict the X-ray to be cancerous when

$$\mathbb{E}_y \left[C(y, +1) | \mathbf{x} \right] < \mathbb{E}_y \left[C(y, -1) | \mathbf{x} \right]$$

$$1 - p < 1000p$$

$$p > \frac{1}{1001}$$

Decision Theory: Summary

Classification problems can be broken down into two stages:

- *Interference stage*: learning a model based on data
- *Decision stage*: using posterior probabilities to make optimal decisions

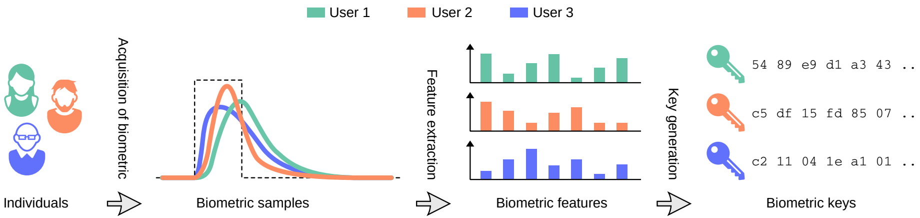
For simple loss functions, optimal decisions can be derived from the probabilities directly. However, if loss is asymmetric, for instance, decision stage is of paramount importance.

And even more so if there is a lot of uncertainty about the data — or if training set is small.

Loss Function: An example

- Feature extraction for biometric measurements
- Time-stable features can facilitate classification: Authentication and Identification
  - Authentication: usually binary classification. Verify an identity claim
  - Identification: multi-class classification. Ranking of most likely class labels.
- Reliable and unique features can support biometric key generation

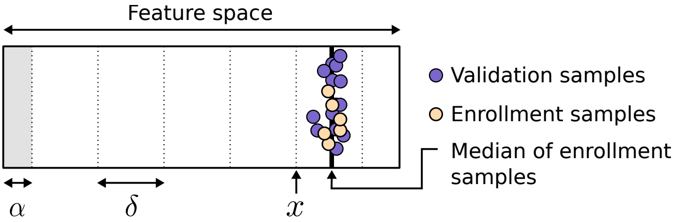
Biometric Key Generation: Why?



- Biometric privacy! (What if biometric measurement leaks?)
- **Use case**:
  Anonymous biometric authentication, symmetric encryption of storage media
- Lost popularity due to advent of TPMs, tamper-proof devices, trusted execution environments

Biometric Key Generation: Error correction

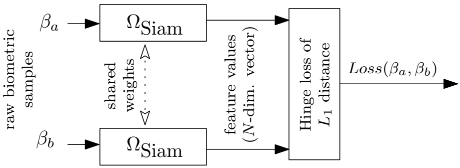
- Error correction by quantization



- Key is based on quantized values x
  - $K = \text{Hash}(x_1||x_2||x_3||\dots)$
- Template consists of user-specific offsets α
- Quantization width δ is global

Biometric Feature Extraction

- Extract features suitable for key generation
  - unique to every individual
  - distinctive across the population
- Semi-supervised approach used in Deep learning
- Deep multi-layer convolutional network in a Siamese architecture
- Let the model train itself



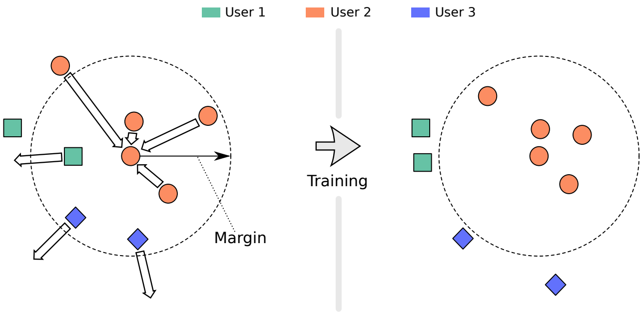
Hinge Loss Function

- Hinge loss for multi-class problem
- β_j: biometric sample belonging to class j
- Minimize Manhattan distance
- Spread samples with different labels at least m apart. Set margin to quanization width for key generation: $m = \delta$

$$Loss(\beta_a, \beta_b) = \begin{cases} |\Omega_{\mathrm{Siam}}(\beta_a) - \Omega_{\mathrm{Siam}}(\beta_b)| & \text{if } a = b \\ \max\left(0, m - |\Omega_{\mathrm{Siam}}(\beta_a) - \Omega_{\mathrm{Siam}}(\beta_b)|\right) & \text{if } a \neq b \end{cases}$$

Feature Learning

- Learn N-dimensional non-linear embedding
- Loss is defined in terms of Manhattan distance



$\rightarrow$ Ideal features for biometric key generation

Intuition for Generative Modeling

- Previously in the course, we were discussing classification problems (pixels or images). Based on data we were predicting a label (cat, pedestrian, etc.). **The question**: based on the features, which label is more probable?

- Now we are discussing models which generate data (images, etc.). **The question**: for this class how likely is this feature. It can sound like a reverse problem, but actually it is not and we will see why.

A supervised classification problem

- Let $\mathcal{Y} = \{0, 1\}$ be the set of labels and $\mathcal{X} = \mathbb{R}^d$ a d-dimensional feature space.
- Training set $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ of n labeled examples where $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}$.
- Goal: Predict class y_j for a new data point $\mathbf{x}_j$

Two fundamentally different ways

Discriminative modeling

Estimate conditional probability
$P(y = k | X = \mathbf{x})$

Generative modeling

Estimate joint probability
$P(y = k, X = \mathbf{x})$

Two fundamentally different ways

Discriminative modeling

Estimate conditional probability $P(y = k|X = \mathbf{x})$

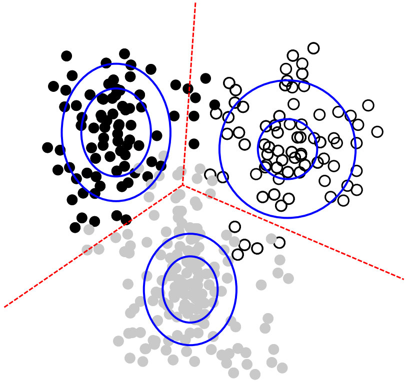
Based on features, we learn which label is more probable. We minimize the error we make in predicting new data points by comparing features.

Generative modeling

Estimate joint probability $P(y = k, X = \mathbf{x})$

We attempt to infer the distribution of features and determine how likely a specific feature value is.

Visual comparison



Discriminative modeling

Learn “decision boundaries” based on features

Generative modeling

Estimate joint distribution via class-conditional distributions

Generative modeling

Joint distribution

Estimating $P(y = k, X = \mathbf{x})$ directly is often not tractable (not enough data points).

Alternative approach

- Estimate prior distribution $P(y = k)$ on labels
- Derive conditional distribution $P(X = \mathbf{x}|y = k)$ based on available data
- Obtain posterior distribution $P(y = k|X = \mathbf{x})$

Generative modeling

Obtain posterior distribution using Bayes' rule:

$$P(y|x) = \frac{P(x,y)}{P(x)} = \frac{P(x|y)\ P(y)}{P(x)} = \frac{P(x|y)\ P(y)}{\sum_{y'} P(x|y')P(y')} = \frac{1}{Z}P(x|y)\ P(y)$$

where $Z = P(x)$

Often, a closed-form expression of the posterior is not possible.
For convenience, we can choose a **conjugate prior**: $P(y|x)$ and $P(y)$ have the same algebraic form, i.e., normal distribution.

Normalization

Obtain posterior distribution using Bayes' rule:

$$P(y|x) = \frac{P(x,y)}{P(x)} = \frac{P(x|y)\ P(y)}{P(x)} = \frac{P(x|y)\ P(y)}{\sum_{y'} P(x|y')P(y')} = \frac{1}{Z} P(x|y)\ P(y)$$

Note: Computing Z (normalization) is not necessary if we just want to determine most likely class label. $P(y|x) \sim P(x|y)\ P(y)$

Optimization Problem

- Discriminative learning optimization:

$$-\log P(Y = y | X = \mathbf{x}) \rightarrow \min_y$$

- Generative learning optimization

$$-\log P(Y, X) = -\log P(Y|X) \cdot P(X) = -\log P(Y|X) - \log P(X)$$

- Intuition: Discriminative learning tries to fit the data while generative incorporate $P(X)$ which acts similarly to a regularizer

Generative vs. Discriminative

Generative

- probabilistic "model" of each class
- decision boundary: where one model becomes more likely
- natural use of unlabeled data

Discriminative

- focus on decision boundary
- more powerful with lots of examples
- can not use unlabeled data

When to use which approach?

- If the model is well-specified (you managed to build p(x) correctly), generative modeling yields better results
- If the model is not well-specified (much more often the case), then it depends on how much data is available:
  small amount of data $\rightarrow$ *generative,*
  more data $\rightarrow$ *discriminative.*
  For more details [1].

Quiz (former exam question)

You trained a generative model and want to predict a label $y \in \{0, 1\}$ for new data point $\mathbf{x}$. Your model tells you:

- $P(Y = 1) = P(Y = 0) = 0.5$
- $P(X|Y = 0) = 0.02$
- $P(X|Y = 1) = 0.03$

To predict a label, you should compute $P(Y = 0|X)$. What is the result:

1) 0.01
2) 0.2
3) 0.4
4) Undetermined as we need to know $P(X)$

Quiz (former exam question)

Solution: $P(Y = 0|X) = 0.4$

$$\begin{aligned} P(y = 0|x) &= \frac{P(x|y = 0)\ P(y = 0)}{P(x)} \\ &= \frac{P(x|y = 0)\ P(y = 0)}{P(x|y = 0)\ P(y = 0) + P(x|y = 1)\ P(y = 1)} \\ &= \frac{0.02 \cdot 0.5}{0.02 \cdot 0.5 + 0.03 \cdot 0.5} \end{aligned}$$

Naive Bayes: A Generative Model

Model class labels as generated from categorical variable that determines class:

$$P(Y = y) \text{ where } y \in \mathcal{Y} = \{1, \dots c\}$$

Simplification / Model assumption:

$$P(X = \mathbf{x}|Y) = \prod_{i=1}^{c} P(X_i|Y)$$

Features are conditionally independent if class label is known.
This assumption might not always hold!

Naive Bayes: Feature Distribution

We still need to specify $P(X_i = x_i | Y = y)$

Example from the lecture: $P(X_i = x_i | Y = y) = \mathcal{N}(x_i | \mu_{y,i}, \sigma^2_{y,i})$

We end up with a set of parameters defining class-conditional probabilities.
Solution: We use Maximum Likelihood Estimation to determine $P(X_i = x_i | Y = y)$ and $P(Y = y)$

Naive Bayes: Homework

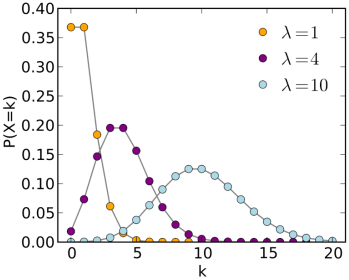
Poisson Naive Bayes

Let $\mathcal{Y} = \{0, 1\}$ be the set of labels and $\mathcal{X} = \mathbb{N}^d$ a d-dimensional features space ($\mathbb{N} = \{0, 1, 2, \dots\}$). You are given a training set $D = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ of n labeled examples $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}$.

Why is Naive Bayes model a generative model?

Naive Bayes: Poission distribution

- Maximum likelihood estimation
- $P(Z = k) = e^{-\lambda} \frac{\lambda^k}{k!}$
- Binary classification: $p_0, p_1 \in [0, 1]$, and vectors $\lambda_0, \lambda_1 \in \mathbb{R}^d$
- Constraints: $p_0 + p_1 = 1$, and λ_0, λ_1 are vectors with non-negative components



Naive Bayes: Poisson distribution

- Maximum likelihood estimation
- $P(Z = k) = e^{-\lambda}\frac{\lambda^k}{k!}$
- Binary classification: $p_0, p_1 \in [0, 1]$, and vectors $\lambda_0, \lambda_1 \in \mathbb{R}^d$
- Constraints: $p_0 + p_1 = 1$, and λ_0, λ_1 are vectors with non-negative components

$\rightarrow$ Why are λ_0, λ_1 vectors?

$\rightarrow$ What variables are Poisson distributed?

$\rightarrow$ What are the p_y's?

Possion Naive Bayes: Priors

- What is the joint distribution $P(X, Y)$ of our model?

- Use MLE to estimate the class priors and the conditional feature distributions
- n is the total number of data points, $n_1 = \sum_{i=1}^{n} y_i$ the number of times '1' was observed, and $n_0 = n - n_1$ the number of '0', accordingly.
- The MLE for $p(y) = Bernoulli(\theta)$ is simply the empirical frequency $p_y = \frac{n_y}{n}$

Possion Naive Bayes: Feature distribution

- What is the joint distribution $P(X, Y)$ of our model?

- MLE for a *Poisson*(λ) distribution is just the empirical mean
- Thus: $\lambda_{y,j} = \frac{\sum_{i=1}^{n} x_{i,j} \cdot \mathbf{1}\{y_i = y\}}{n_y}$

$\rightarrow$ We compute the mean over all samples that belong to same class

Poisson Distribution: Mean

$$E(X) = \sum_{k \geq 0} k \frac{1}{k!} \lambda^k e^{-\lambda}$$

$$\begin{aligned} E(X) &= \lambda e^{-\lambda} \sum_{k \geq 1} \frac{1}{(k-1)!} \lambda^{k-1} \\ &= \lambda e^{-\lambda} \sum_{j \geq 0} \frac{\lambda^j}{j!} \\ &= \lambda e^{-\lambda} e^{\lambda} \\ &= \lambda \end{aligned}$$

Possion Naive Bayes: Joint distribution

$$p(x, y) = p_y \prod_{j=1}^{d} e^{-\lambda_{y,j}} \frac{\lambda_{y,j}^{x_j}}{x_j!}$$

Classification

Minimize the misclassification probability of a new observation $\mathbf{x} \in \mathcal{X}$, i.e. $y_{\text{pred}} = \text{argmax}_{y \in \mathcal{Y}} P(y|X = \mathbf{x})$.

!! Note: We don't have to compute $p(x)$

- Show that the predicted label y_{pred} for $\mathbf{x}$ is determined by a hyperplane, i.e., that $y_{\text{pred}} = \left[\mathbf{a}^\top \mathbf{x} \geq b\right]$ for some $\mathbf{a} \in \mathbb{R}^d, b \in \mathbb{R}$.

Possion Naive Bayes: Decision boundary

We need to find the decision boundary:

$$
\begin{aligned}
& p(y=0|x) = p(y=1|x) \\
\iff \quad & p(x,0) = p(x,1) \\
\iff \quad & p_0 \prod_{j=1}^{d} e^{-\lambda_{0,j}} \frac{\lambda_{0,j}^{x_j}}{x_j!} = p_1 \prod_{j=1}^{d} e^{-\lambda_{1,j}} \frac{\lambda_{1,j}^{x_j}}{x_j!}
\end{aligned}
$$

Possion Naive Bayes: Decision boundary

We need to find the decision boundary:

$$
\begin{aligned}
& p(y=0|x) = p(y=1|x) \\
\iff \quad & p(x,0) = p(x,1) \\
\iff \quad & p_0 \prod_{j=1}^{d} e^{-\lambda_{0,j}} \frac{\lambda_{0,j}^{x_j}}{x_j!} = p_1 \prod_{j=1}^{d} e^{-\lambda_{1,j}} \frac{\lambda_{1,j}^{x_j}}{x_j!} \\
\iff \quad & \log\left(\frac{p_0}{p_1}\right) + \sum_{j=1}^{d} [-\lambda_{0,j} + \log(\lambda_{0,j}) x_j] = \sum_{j=1}^{d} [-\lambda_{1,j} + \log(\lambda_{1,j}) x_j]
\end{aligned}
$$

$$
0 = \log\left(\frac{p_0}{p_1}\right) + \sum_{j=1}^{d} \lambda_{1,j} - \lambda_{0,j} + \sum_{j=1}^{d} \log\left(\frac{\lambda_{0,j}}{\lambda_{1,j}}\right) x_j
$$

References I

Andrew Y Ng and Michael I Jordan.
On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes.
In *Advances in neural information processing systems*, pages 841–848, 2002.

Christopher M Bishop.
Pattern recognition and machine learning.
springer, 2006.

Christopher M Bishop et al.
Neural networks for pattern recognition.
Oxford university press, 1995.

Anastasia Makarova.
Generative vs. discriminative modeling.
Tutorial Slides.

References II

Marc Roeschlin, Ivo Sluganovic, Ivan Martinovic, Gene Tsudik, and Kasper B Rasmussen.

Generating secret keys from biometric body impedance measurements.

In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*, pages 59–69. ACM, 2016.

IntroML Tutorial: Generative Modeling & Decision Theory

Marc Roeschlin

ETH Zurich

13 May 2019