CS 446/ECE 449: Machine Learning

Lecture 2: Naive Bayes

Han Zhao
01/18/2024

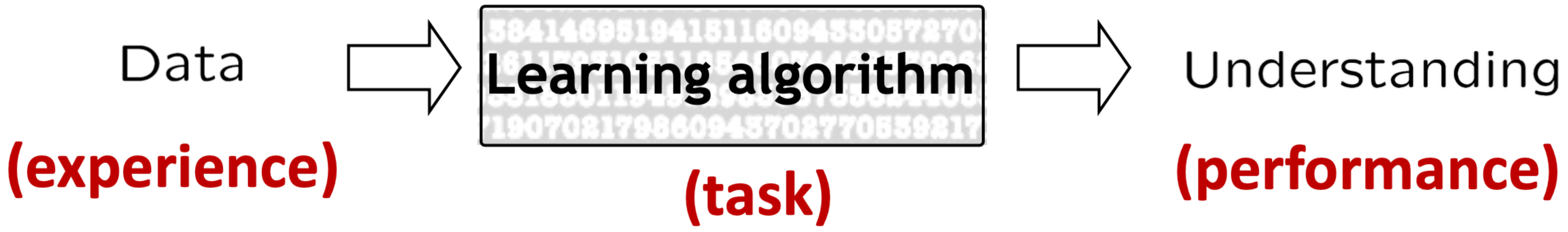
ILLINOIS
Computer Science
GRAINGER COLLEGE OF ENGINEERING

Recap: Machine Learning

How to obtain such a model/predictor? Let's recall the definition of machine learning:

> "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."
>
> — Tom M. Mitchell

- E: data
- T: task of interest
- P: objective function

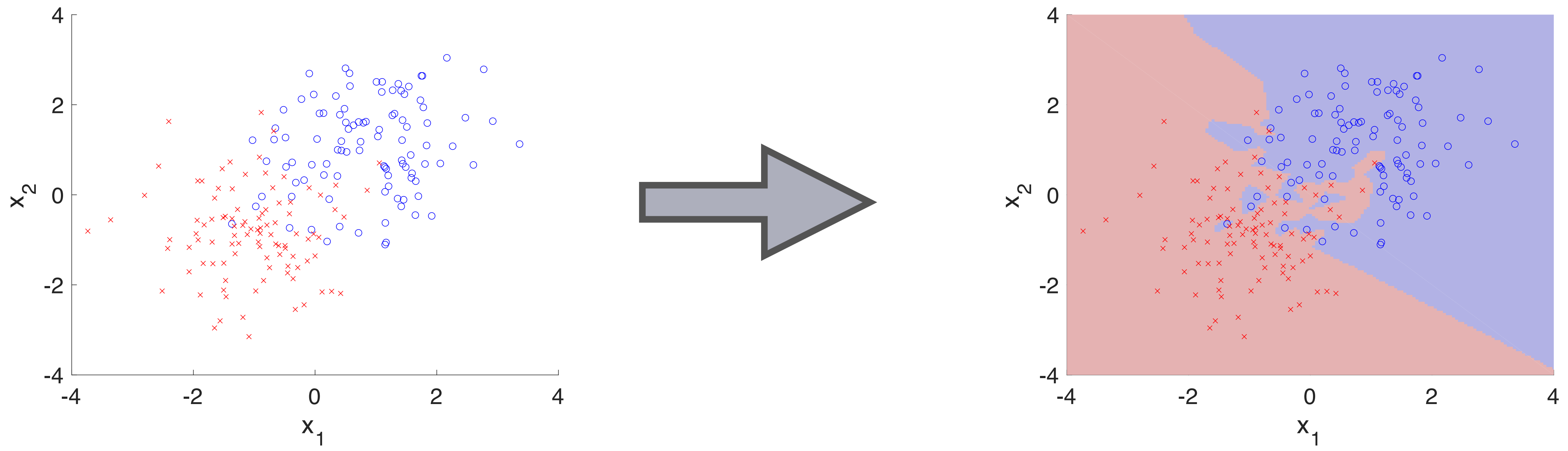


Recap: Nearest Neighbor

We have a huge dataset from ImageNet that contains 1M images: $\mathscr{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{1,000,000}$. Now here comes a new image x, how can we predict/determine its label $\hat{y}$?

A simple idea: compare the similarity of x with all the existing images, and use the label ($y^{(c)}$) of the closest image $x^{(c)}$ as the prediction:

$$\hat{y} = y^{(c)} \quad \text{where} \quad c = \arg\min_{i \in [n]} \|x - x^{(i)}\|_2$$



Lecture Today

- (Conditional) Independence
- (Discrete/Continuous) Naive Bayes

Probability and Estimation

We have a huge dataset from ImageNet that contains 1M images: $\mathscr{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{1,000,000}$. Now here comes a new image x, how can we predict/determine its label $\hat{y}$?

- Recall: $X \in \mathbb{R}^d, Y \in [k]$
- In order to answer the prediction problem, we can try to model the joint distribution between the input X and output Y

$$\Pr(X_1, \ldots, X_d, Y)$$

Why?

Probability and Estimation

We have a huge dataset from ImageNet that contains 1M images: $\mathscr{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{1,000,000}$. Now here comes a new image x, how can we predict/determine its label $\hat{y}$?

- Recall: $X \in \mathbb{R}^d, Y \in [k]$
- In order to answer the prediction problem, we can try to model the joint distribution between the input X and output Y

$$\Pr(X_1, \ldots, X_d, Y)$$

- In order to answer the classification query, it suffices to compute the conditional probability:

$$\Pr(Y \mid X_1, \ldots, X_d)$$

for any potential events over $X_1, \ldots, X_d, Y$

Probability and Estimation

Example: Lung cancer prediction (let $\mathscr{D}$ be the dataset)

#	Gender	Smoke	Lung Cancer?
1000	M	Y	Yes
5000	M	Y	No
100	M	N	Yes
10000	M	N	No
800	F	Y	Yes
5000	F	Y	No
50	F	N	Yes
10000	F	N	No

$$\Pr_{\mathscr{D}}(\text{Gender} = \text{M}) = \frac{1000 + 5000 + 100 + 10000}{1000 + 5000 + 100 + 10000 + 800 + 5000 + 50 + 10000} = 0.5039$$

$$\Pr_{\mathscr{D}}(\text{Cancer} = \text{Y}) = \frac{1000 + 100 + 800 + 50}{1000 + 5000 + 100 + 10000 + 800 + 5000 + 50 + 10000} = 0.0610$$

Probability and Estimation

Example: Lung cancer prediction (let $\mathscr{D}$ be the dataset)

#	Gender	Smoke	Lung Cancer?
1000	M	Y	Yes
5000	M	Y	No
100	M	N	Yes
10000	M	N	No
800	F	Y	Yes
5000	F	Y	No
50	F	N	Yes
10000	F	N	No

Given the data $\mathscr{D}$, what is the probability that a male patient who smokes will get cancer?

$$\Pr_{\mathscr{D}}(\text{Cancer} = \text{Y} \,|\, \text{Gender} = \text{M}, \text{Smoke} = \text{Y})$$

Probability and Estimation

Example: Lung cancer prediction (let $\mathscr{D}$ be the dataset)

Given the data $\mathscr{D}$, what is the probability that a male patient who smokes will get cancer?

$$\Pr_{\mathscr{D}}(\text{Cancer} = \text{Y} \,|\, \text{Gender} = \text{M}, \text{Smoke} = \text{Y})$$

#	Gender	Smoke	Lung Cancer?
1000	M	Y	Yes
5000	M	Y	No
100	M	N	Yes
10000	M	N	No
800	F	Y	Yes
5000	F	Y	No
50	F	N	Yes
10000	F	N	No

Probability and Estimation

Example: Lung cancer prediction (let $\mathscr{D}$ be the dataset)

Given the data $\mathscr{D}$, what is the probability that a male patient who smokes will get cancer?

$$\Pr_{\mathscr{D}}(\text{Cancer} = \text{Y} \mid \text{Gender} = \text{M}, \text{Smoke} = \text{Y})$$

Based on the data, we know that

$$\Pr_{\mathscr{D}}(\text{Cancer} = \text{Y} \mid \text{Gender} = \text{M}, \text{Smoke} = \text{Y}) = \frac{1000}{1000 + 5000} = \frac{1}{6}$$

Table look-up works for our toy example. But how about in general?

For example, consider the case where we have 100 binary attributes as our features. How many different feature combinations we may have?

- # rows in the table? $\geq 2^{100} \geq 10^{30}$
- # people in the world ~ $8B = 8 \cdot 10^9$
- Fraction of rows with 0 examples $\geq 99.999\,\%$

Probability and Estimation

Table look-up works for our toy example. But how about in general?

For example, consider the case where we have 100 binary attributes as our features. How many different feature combinations we may have?

- # rows in the table? $\geq 2^{100} \geq 10^{30}$
- # people in the world ~ $8B = 8 \cdot 10^9$
- Fraction of rows with 0 examples $\geq 99.999\,\%$

This is known as a combinatorial explosion or the curse of dimensionality

Let’s see what do we need to estimate in order to answer any query

Bayes formula: for any two events A, B where $\Pr(B) > 0$, then

$$\Pr(A \mid B) = \frac{\Pr(B \mid A)\Pr(A)}{\Pr(B)}$$

Probability and Estimation

Bayes formula: for any two events A, B where $\Pr(B) > 0$, then

$$\Pr(A \mid B) = \frac{\Pr(B \mid A) \Pr(A)}{\Pr(B)}$$

For general classification query, we need to compute

$$\begin{aligned}
\Pr(Y \mid X_1, \ldots, X_d) &= \frac{\Pr(Y) \Pr(X_1, \ldots, X_d \mid Y)}{\Pr(X_1, \ldots, X_d)} \\
&= \frac{\Pr(Y) \Pr(X_1, \ldots, X_d \mid Y)}{\sum_{y \in [k]} \Pr(Y = y, X_1, \ldots, X_d)} \\
&= \frac{\Pr(Y) \Pr(X_1, \ldots, X_d \mid Y)}{\sum_{y \in [k]} \Pr(X_1, \ldots, X_d \mid Y = y) \Pr(Y = y)}
\end{aligned}$$

- The marginal probability $\Pr(Y = y)$ is easy to estimate for any $y \in [k]$
- However, due to the combinatorial explosion, we cannot accurately estimate $\Pr(X_1, \ldots, X_d \mid Y)$ since it contains $2(2^d - 1)$ quantities

Conditional Independence

Assumption: all the feature attributes are statistically independent conditioned on the label.

$$\forall x_1, \ldots, x_d, y, \;\; \Pr(X_1 = x_1, \ldots, X_d = x_d \mid Y = y) = \prod_{i=1}^{d} \Pr(X_i = x_i \mid Y = y)$$

Why does it help?

- Instead of estimating the joint probability over $X_1, \ldots, X_d$ given Y, now we only need to estimate each marginal probability over $X_i, \forall i \in [d]$ separately
- The number of parameters to be estimated reduces from $2(2^d - 1)$ to $2d$
- When each X_i is binary, it suffices for us to estimate $\Pr(X_i = 1 \mid Y = y)$ for each $y \in [k]$

Maximum Likelihood Estimation

Question: given a dataset $\mathscr{D}$ that contains n instances of $X_1, \ldots, X_d, Y$, where $\forall i \in [d], X_i \in \{0,1\}$ and $Y \in [k]$. How we can estimate $\Pr(X_i = 1 \mid Y = y)$?

Let $\Pr(X_i = 1 \mid Y = y) = p_{iy} \in [0,1]$, there is only one parameter to be estimated.

Intuition:

Just do the counting!

- Count # instances in $\mathscr{D}$ with label y, denoted as c_y
- Count # instances in $\mathscr{D}$ with label y and $X_i = 1$, denoted as c_{iy}
- Construct our estimate $\hat{p}_{iy} = c_{iy}/c_y$

Repeat the above process for all X_i and $y \in [k]$. We then finish the estimation of all the conditional probabilities $\Pr(X_i = 1 \mid Y = y)$

Naive Bayes Classifier

Assumption: all the feature attributes are statistically independent conditioned on the label.

$$\forall x_1, \ldots, x_d, y, \ \Pr(X_1 = x_1, \ldots, X_d = x_d \,|\, Y = y) = \prod_{i=1}^{d} \Pr(X_i = x_i \,|\, Y = y)$$

NB algorithm (training):

Input: a dataset $\mathscr{D}$ that contains n labeled instances

1. Count # instances in $\mathscr{D}$ with label y, denoted as c_y
2. For each $i \in [d]$, count # instances in $\mathscr{D}$ with label y and $X_i = 1$, denoted as c_{iy}
3. Construct the estimate $\hat{p}_{iy} = c_{iy}/c_y$
4. Construct the estimate $\hat{p}_y = c_y/n$

Output: $\hat{p}_y, \hat{p}_{iy}, \forall i \in [d], y \in [k]$

Naive Bayes Classifier

Given a trained NB classifier, how should we make a prediction for a new instance $(x_1, \ldots, x_d)$?

NB algorithm (test):

Input: the trained model $\hat{p}_y, \hat{p}_{iy}, \forall i \in [d], y \in [k]$

$$\hat{y} \leftarrow \arg\max_{y \in [k]} \Pr_{\mathcal{D}}(Y = y \mid X_1 = x_1, \ldots, X_d = x_d)$$

$$= \arg\max_{y \in [k]} \hat{p}_y \prod_{i=1}^{d} \hat{p}_{iy}^{x_i} (1 - \hat{p}_{iy})^{1 - x_i}$$

Output: $\hat{y}$

Naive Bayes Classifier

Properties of the Naive Bayes classifier:

Claim: when $\mathscr{Y} = \{0,1\}$, i.e., $k = 2$, then Naive Bayes is a linear classifier

Proof: recall the decision rule of the NB classifier:

$$\hat{y} \leftarrow \arg\max_{y \in [k]} \hat{p}_y \prod_{i=1}^{d} \hat{p}_{iy}^{x_i} (1 - \hat{p}_{iy})^{1-x_i}$$

For $k = 2$, equivalently, the predicted output will be 1 iff

$$\log \hat{p}_1 + \sum_{i=1}^{d} x_i \log \hat{p}_{i1} + (1 - x_i) \log(1 - \hat{p}_{i1}) \geq \log \hat{p}_0 + \sum_{i=1}^{d} x_i \log \hat{p}_{i0} + (1 - x_i) \log(1 - \hat{p}_{i0})$$

Furthermore, note that $x_i \in \{0,1\}$, then we can compactly write the decision rule as: predicting 1 iff

$$\sum_{i=1}^{d} x_i \log \frac{\hat{p}_{i1}}{\hat{p}_{i0}} + (1 - x_i) \log \frac{1 - \hat{p}_{i1}}{1 - \hat{p}_{i0}} + \log \frac{\hat{p}_1}{\hat{p}_0} \geq 0$$

This is a linear function of the input vector $x = (x_1, \ldots, x_d)$.

Naive Bayes Classifier

Properties of the Naive Bayes classifier:

NB classifier is a parametric model, i.e., there is a fixed number of parameters to be estimated: $\hat{p}_y, \hat{p}_{iy}, \forall i \in [d], y \in [k]$

Comparison: Nearest Neighbor does not have a fixed number of parameters to be estimated

Naive Bayes Classifier

Properties of the Naive Bayes classifier:

NB classifier is a generative model: it models the joint distribution between X and Y under the conditional independence assumption:

- Once the model has been trained, we can compute the estimated joint probability $\Pr_{\mathcal{D}}(X_1, \ldots, X_d, Y)$
- We can generate new data (sampling) from the estimated model:
  1. Sample the label $Y = y$ according to $\Pr_{\mathcal{D}}(Y)$
  2. For each feature $i \in [d]$, independently sample X_i according to $\Pr_{\mathcal{D}}(X_i \mid Y = y)$

Comparison: We cannot generate new data from the Nearest Neighbor classifier

Naive Bayes Classifier

Limitations of the NB classifier:

- The conditional independence assumption could be too strong to hold in practice (consider an image as an example)
- The Maximum Likelihood Estimator of $\hat{p}_y$ and $\hat{p}_{iy}$ could be too stringent: it will assign 0 probability to any future instance that has an feature which does not appear before

Laplace smoothing: add pseudo-count ($\lambda > 0$) to the data $\mathscr{D}$

$$\hat{p}_y = \frac{c_y + \lambda}{n + \lambda k} \qquad \hat{p}_{iy} = \frac{c_{iy} + \lambda}{c_y + \lambda |X_i|}$$

Intuition: we are adding $\lambda > 0$ data points for each value of $Y, X_1, \ldots, X_d$ to our dataset $\mathscr{D}$

Later in this course: this is also known as the Maximum-a-Posteriori (MAP) estimate (vs the Maximum Likelihood Estimator (MLE)).

Naive Bayes Classifier

Extension to continuous distributions, i.e., $X_i \in \mathbb{R}$:

Assumption: all the feature attributes are statistically independent conditioned on the label.

$$\forall x_1, \ldots, x_d, y, \;\; \Pr(X_1 = x_1, \ldots, X_d = x_d \mid Y = y) = \prod_{i=1}^{d} \Pr(X_i = x_i \mid Y = y)$$

However, instead of modeling $\Pr(X_i \mid Y = y)$ as a binary (categorial) distribution, in the continuous case we assume

$$\Pr(X_i \mid Y = y) = \mathcal{N}(\mu_{iy}, \Sigma_{iy})$$

- The distribution $\mathcal{N}(\mu_{iy}, \Sigma_{iy})$ is a univariate Gaussian distribution with mean μ_{iy} and covariance matrix Σ_{iy}
- Similar to the discrete case, we only need to estimate μ_{iy}, Σ_{iy} separately for each $i \in [d], y \in [k]$

HW1: derive the Gaussian NB classification formula