Generative Models for Discrete Data

Note: Unless otherwise noted all references including images are from the required textbook, Machine Learning: A Probabilistic Perspective by Kevin P. Murphy.

Generative Classifier

We can classify a feature vector x by applying Bayes rule to a generative classifier of the form

$$p(y = c|\mathbf{x}, \boldsymbol{\theta}) \propto p(\mathbf{x}|y = c, \boldsymbol{\theta})p(y = c|\boldsymbol{\theta})$$

$\boldsymbol{\theta}$ is the unknown variables of the model.

Generative Classifier

The key to using such models is specifying a suitable form for the class-conditional density $p(x|y = c, \boldsymbol{\theta})$, which defines what kind of data we expect to see in each class.

Bayesian Concept Probability

Consider how a child learns to understand the meaning of a word, such as “dog”. Presumably the child’s parents point out positive examples of this concept, saying such things as, “look at the cute dog!”, or “mind the doggy”, etc. However, it is very unlikely that they provide negative examples, by saying “look at that non-dog”.

Bayesian Concept Probability

Certainly, negative examples may be obtained during an active learning process — the child says “look at the dog” and the parent says “that’s a cat, dear, not a dog” — but psychological research has shown that people can learn concepts from positive examples alone (Xu and Tenenbaum, 2007).

Bayesian Concept Probability

We can think of learning the meaning of a word as equivalent to **concept learning**, which in turn is equivalent to binary classification. To see this, define $f(x) = 1$ if x is an example of the concept C, and $f(x) = 0$ otherwise. Then the goal is to learn the indicator function f, which just defines which elements are in the set C.

Bayesian Concept Probability

Now consider an example of concept learning called **number game**, which is partly based on Josh Tenenbaum's PhD thesis (1999).

A simple arithmetic concept C, such as "prime number" or "a number between 1 and 10" is chosen. Then you are given a series of randomly chosen positive examples $\mathcal{D} = \{x_1, \ldots, x_N\}$ drawn from C, and asked whether some new test case $\tilde{x}$ belongs to C, *i.e.,* you are asked to classify $\tilde{x}$.

Bayesian Concept Probability

For simplicity, suppose that all numbers are between 1 and 100. Now suppose you are told "16" is a positive example of the chosen concept *C*. What other numbers do you think are positive? 17? 6? 32? 99? It's hard to tell with only one example, so your predictions will be quite vague. Presumably numbers that are similar in some sense to 16 are more likely.

Bayesian Concept Probability

But similar in what way? 17 is similar, because it is “close by”, 6 is similar because it has a digit in common, 32 is similar because it is also even and a power of 2, but 99 does not seem similar. Thus some numbers are more likely than others.

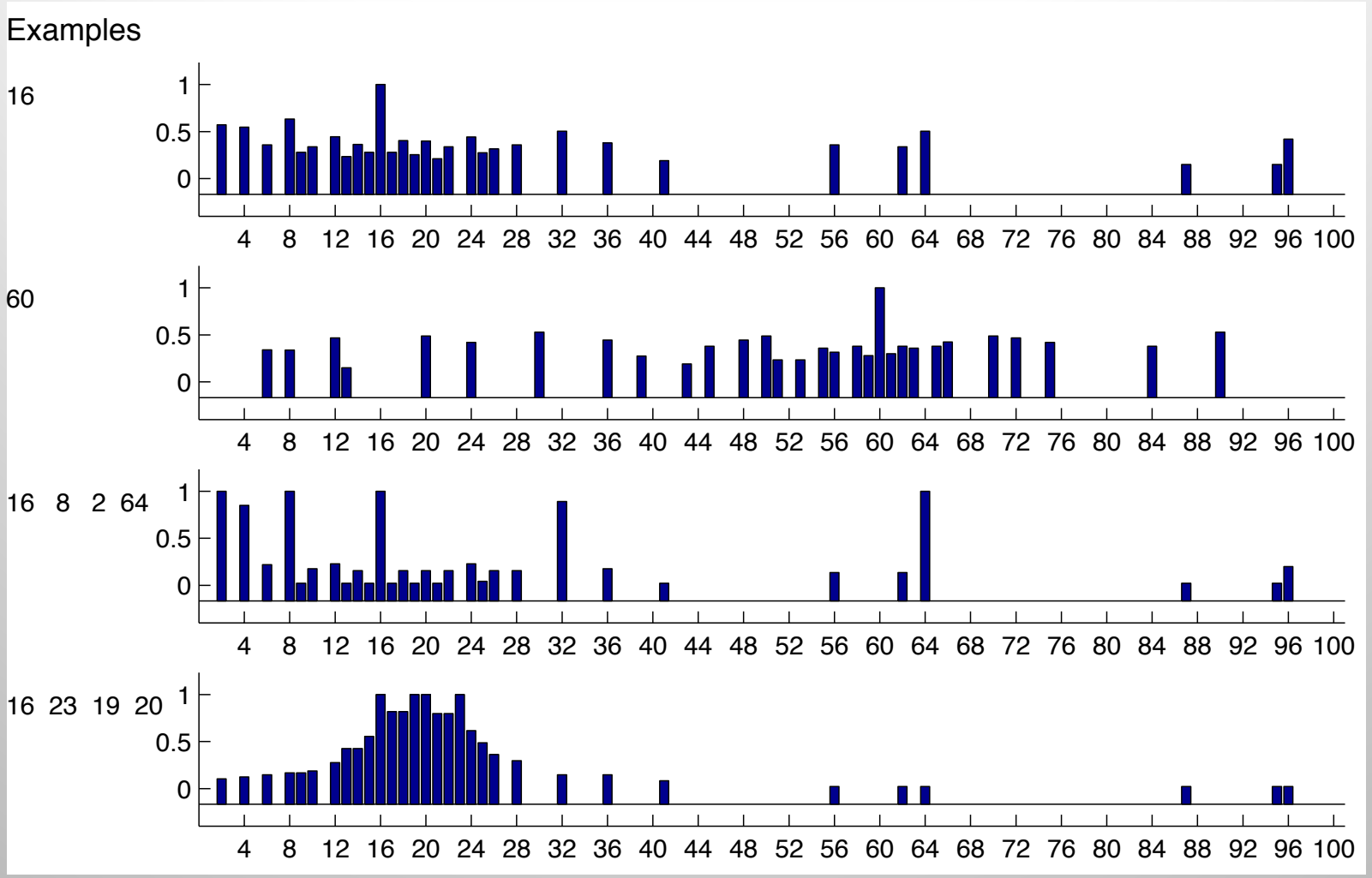
Bayesian Concept Probability

We can represent this as a probability distribution, $p(\tilde{x}|\mathcal{D})$, which is the probability that $\tilde{x} \in C$ given the data $\mathcal{D}$ for any $\tilde{x} \in \{1, \ldots, 100\}$. This is called the **posterior predictive distribution**.

Bayesian Concept Probability

What if you are now told that 8, 2 and 64 are also positive examples. Now you may guess that the hidden concept is “powers of two”. This is an example of **induction**. Given this hypothesis, the predictive distribution is quite specific, and puts most of its mass on powers of 2. If instead you are told that the data is $\mathcal{D} = \{16, 23, 19, 20\}$, you will get a different kind of generalization gradient.

Discrete Random Variables



Empirical predictive distribution over 8 humans in the number game (Tenenbaum, 1999).

Bayesian Concept Probability

How do we emulate this behavior in a machine? The classic approach to induction is to suppose we have a hypothesis space of concepts, $\mathcal{H}$, such as: odd numbers, even numbers, all numbers between 1 and 100, powers of two, all numbers ending in j (for $0 \leq j \leq 9$), etc. The subset of $\mathcal{H}$ that is consistent with the data $\mathcal{D}$ is called the **version space**. As we see more examples, the version space shrinks and we become increasingly certain about the concept (Mitchell 1997).

Bayesian Concept Probability

After seeing $\mathcal{D} = \{16\}$, there are many consistent rules; how do you combine them to predict if $\tilde{x} \in C$? Also, after seeing $\mathcal{D} = \{16, 8, 2, 64\}$, why did you choose the rule "powers of two" and not, say, "all even numbers", or "powers of two except for 32", both of which are equally consistent with the evidence?

Likelihood

We must explain why we chose $h_{two} \triangleq$"powers of two", and not, $h_{even} \triangleq$"even numbers" after seeing $\mathcal{D} = \{16, 8, 2, 64\}$, given that both hypotheses are consistent with the evidence. The key intuition is that we want to avoid **suspicious coincidences**. If the true concept was even numbers, how come we only saw numbers that happened to be powers of two?

Likelihood

Assume that examples are sampled uniformly at random from the **extension** of a concept, which is just the set of numbers that belong to it, e.g., the extension of h_{even} is $\{2, 4, 6, \ldots, 98, 100\}$. Tenenbaum calls this the **strong sampling assumption**. Given this assumption, the probability of independently sampling N items, with replacement, from h is given by

$$p(\mathcal{D}|h) = \left[\frac{1}{\text{size}(h)}\right]^N = \left[\frac{1}{|h|}\right]^N$$

Likelihood

This crucial equation embodies what Tenenbaum calls the **size principle**, which means the model favors the simplest (smallest) hypothesis consistent with the data. This is more commonly known as **Occam's razor**.

Prior

A **prior** (aka **prior probability distribution**) is the probability distribution of an uncertain quantity that represents the belief about this quantity before new information is taken into account.

Prior

Suppose $\mathcal{D} = \{16, 8, 2, 64\}$. Given this data, the concept h' =“powers of two except 32” is more likely than h =“powers of two”, since h does not need to explain the coincidence that 32 is missing from the set of examples.

Prior

However, the hypothesis h' =“powers of two except 32” seems “conceptually unnatural”. We can capture such intuition by assigning low prior probability to unnatural concepts. Of course, one person’s prior might be different than another person’s, which makes Bayesian reasoning a source of controversy.

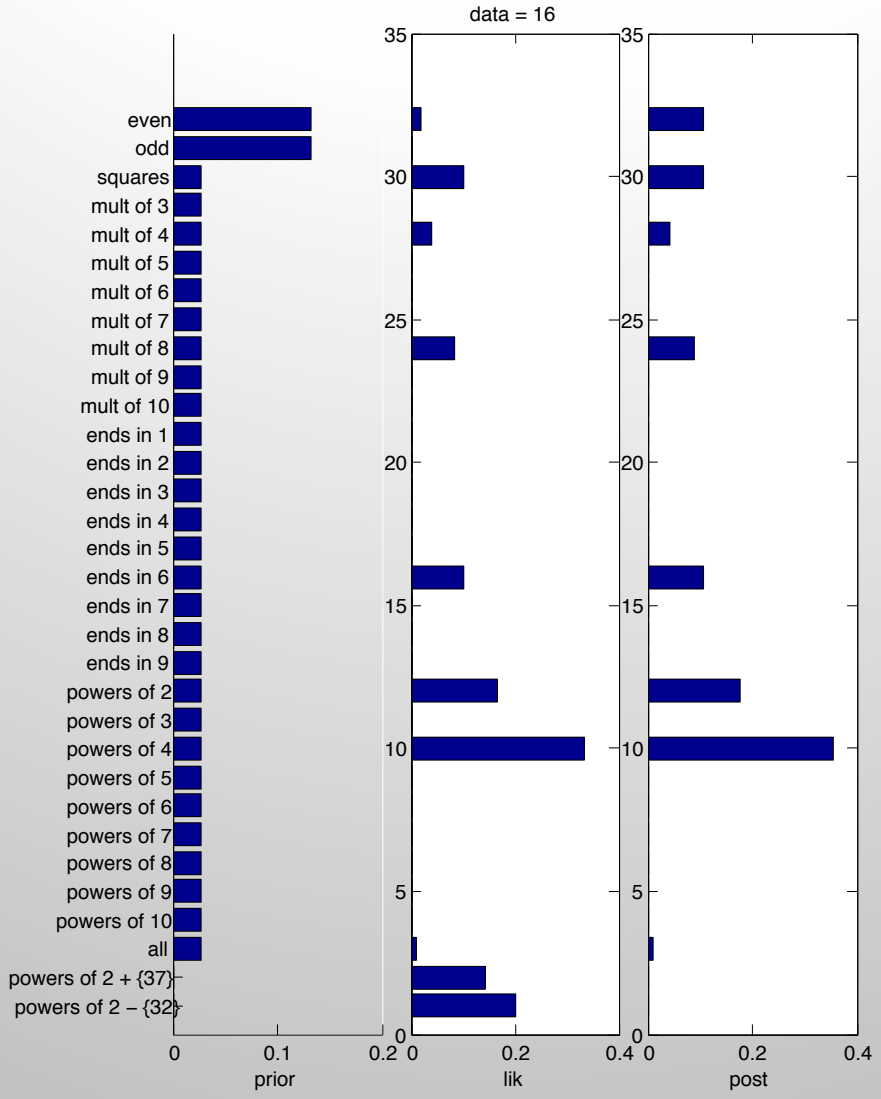
Prior

Although the subjectivity of the prior is controversial, it is actually quite useful. If you are told the numbers are from some arithmetic rule, then given 1200, 1500, 900 and 1400, you may think 400 is likely but 1183 is unlikely. But if you are told that the numbers are examples of healthy cholesterol levels, you would probably think 400 is unlikely and 1183 is likely.

Prior

So, what prior to use? Let's use a simple prior which puts uniform probability on 30 simple arithmetical concepts, such as "even numbers", "odd numbers", "prime numbers", "numbers ending in 9", etc. To make things more interesting, concepts even and odd are made more likely apriori. Two"unnatural" concepts, namely "powers of 2, plus 37" and "powers of 2, except 32", but give them low prior weight are also included.

Prior



Prior, likelihood and posterior for $\mathcal{D} = \{16\}$

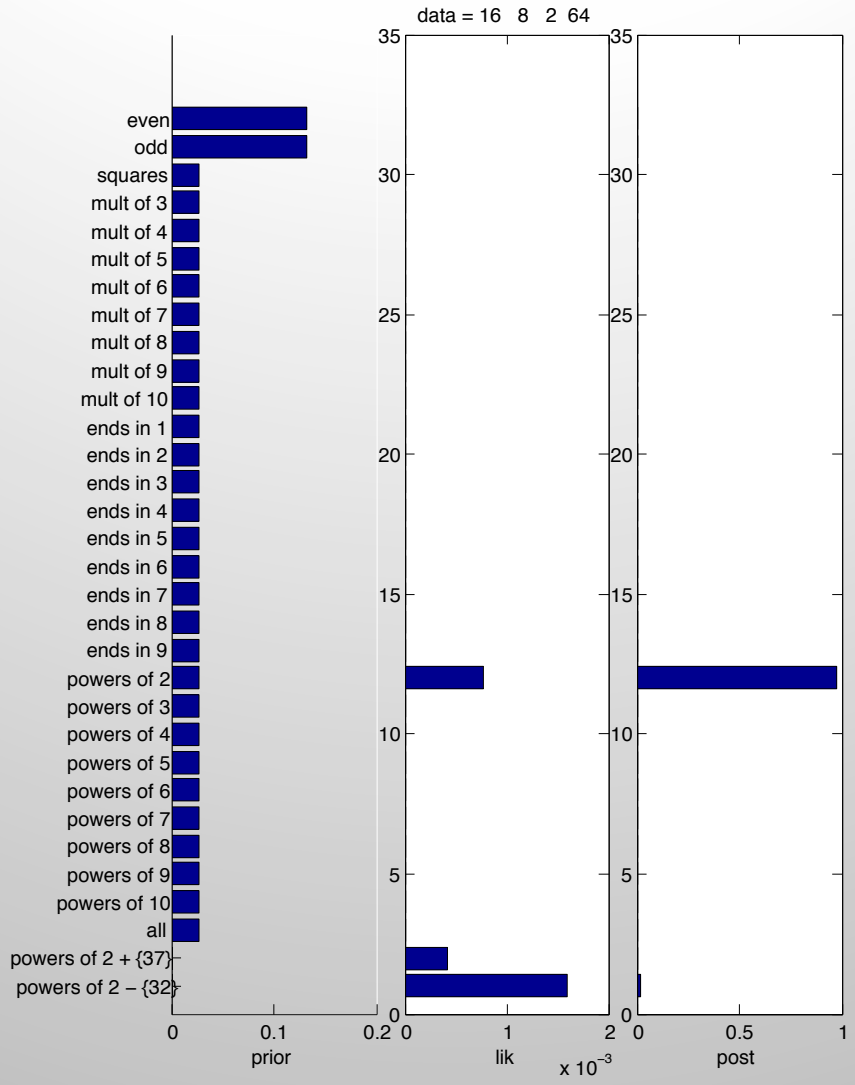
Posterior

The posterior is simply the likelihood times the prior, normalized:

$$p(h|\mathcal{D}) = \frac{p(\mathcal{D}|h)p(h)}{\sum_{h' \in \mathcal{H}} p(\mathcal{D}, h')} = \frac{p(h)\mathbb{I}(\mathcal{D} \in h)/|h|^N}{\sum_{h' \in \mathcal{H}} p(h')\mathbb{I}(\mathcal{D} \in h')/|h'|^N}$$

where $\mathbb{I}(\mathcal{D} \in h)$ is 1 if and only if all the data are in the extension of the hypothesis h.

Posterior



Prior, likelihood and posterior for $\mathcal{D} = \{16, 8, 2, 64\}$

Posterior

In general, when we have enough data, the posterior $p(h|\mathcal{D})$ becomes peaked on a single concept, namely the maximum a posteriori (MAP) estimate

$$p(h|\mathcal{D}) \rightarrow \delta_{\hat{h}^{MAP}}(h)$$

Posterior

where $\hat{h}^{MAP} = \text{argmax}_h p(h|\mathcal{D})$ is the posterior mode, and where δ is the **Dirac measure** defined by

$$\delta_x(A) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \notin A \end{cases}$$

Note that the MAP estimate can be written as

$$\hat{h}^{MAP} = \underset{h}{\text{argmax}}\, p(\mathcal{D}|h)p(h) = \underset{h}{\text{argmax}}\, [\log p(\mathcal{D}|h) + \log p(h)]$$

Posterior

Since the likelihood term depends exponentially on N, and the prior stays constant, as we get more and more data, the MAP estimate converges towards the **maximum likelihood estimate** (**MLE**).

$$\hat{h}^{mle} \triangleq \underset{h}{\operatorname{argmax}}\, p(\mathcal{D}|h) = \underset{h}{\operatorname{argmax}} \log p(\mathcal{D}|h)$$

In other words, if we have enough data, we see that the **data overwhelms the prior**.

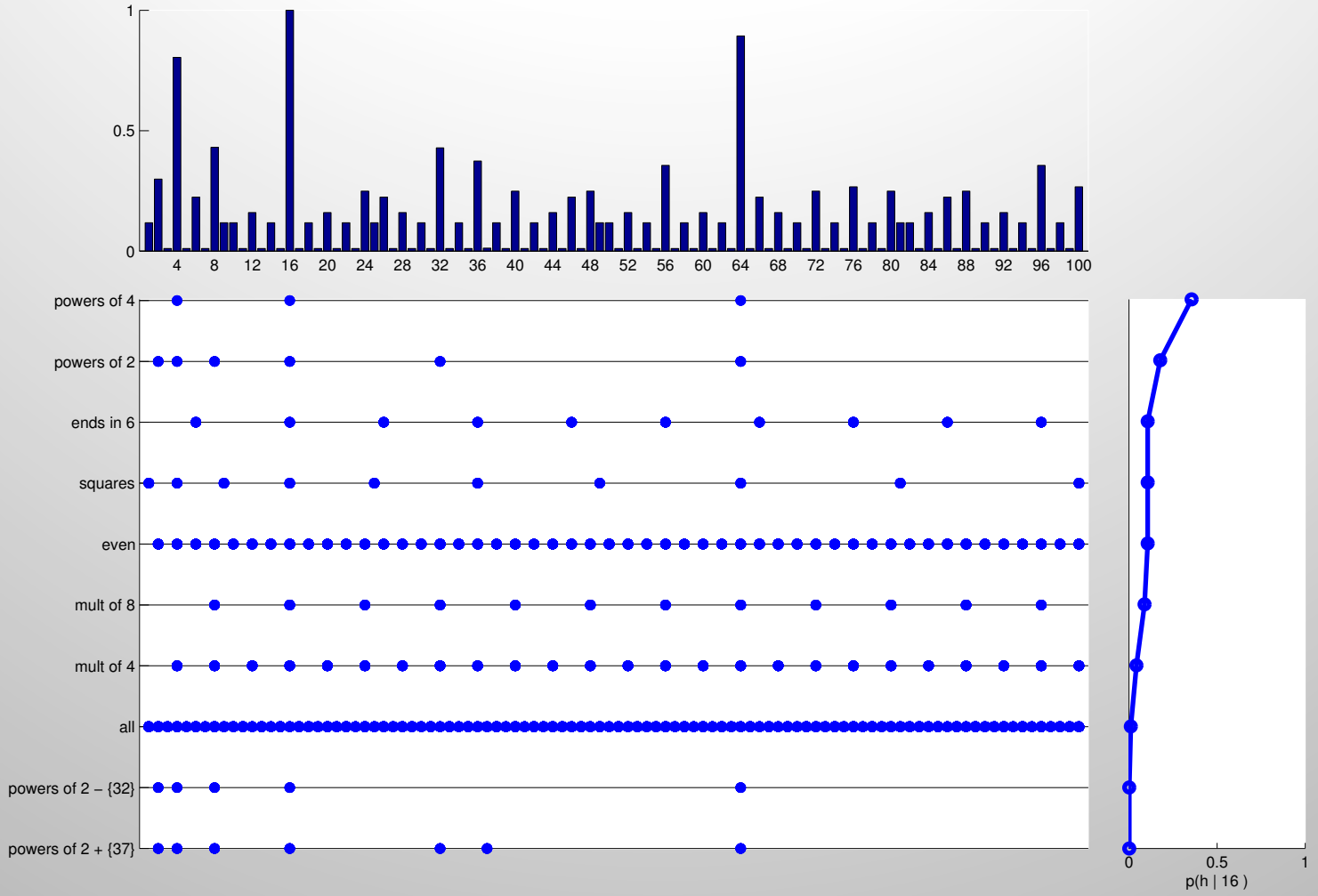
Posterior Predictive Distribution

The posterior is our **internal belief** state about the world. The way to test if our beliefs are justified is to use them to predict objectively observable quantities.

$$p(\tilde{x} \in C|\mathcal{D}) = \sum_h p(y = 1|\tilde{x}, h)p(h|\mathcal{D})$$

This is just a weighted average of the predictions of each individual hypothesis and is called Bayes model averaging (Hoeting et al., 1999).

Posterior Predictive Distribution



Predictive distributions for the model using the full hypothesis space.

Naive Bayes Classifiers

How do we classify vectors of discrete-valued features, $\mathrm{x} \in \{1, \ldots, K\}^D$, where K is the number of values for each feature, and D is the number of features?

We will use a generative approach. This requires us to specify the class conditional distribution, $p(\mathrm{x}|y = c)$.

Naive Bayes Classifiers

The simplest approach is to assume the features are **conditionally independent** given the class label. This allows us to write the class conditional density as a product of one dimensional densities:

$$p(\mathbf{x}|y=c,\boldsymbol{\theta}) = \prod_{j=1}^{D} p(x_j|y=c,\boldsymbol{\theta}_{jc})$$

The resulting model is called a **naive Bayes classifier (NBC)**.

Naive Bayes Classifiers

The model is called “naive” since we do not expect the features to be independent, even conditional on the class label. However, even if the naive Bayes assumption is not true, it often results in classifiers that work well (Domingos and Pazzani, 1997). One reason for this is that the model is quite simple (it only has $O(CD)$ parameters, for C classes and D features), and hence it is relatively immune to overfitting.

Naive Bayes Classifiers

The form of the class-conditional density depends on the type of each feature.

In the case of binary features, $x_j \in \{0, 1\}$, we can use the Bernoulli distribution:

$$p(\mathrm{x}|y = c, \boldsymbol{\theta}) = \prod_{j=1}^{D} \mathrm{Ber}\left(x_j|\mu_{jc}\right),$$

where μ_{jc} is the probability that feature j occurs in class c. This is sometimes called the **multivariate Bernoulli naive Bayes** model.

Naive Bayes Classifiers

In the case of categorical features, $x_j \in \{1, \ldots, K\}$, we can model use the multinoulli distribution:

$$p(\mathrm{x}|y = c, \boldsymbol{\theta}) = \prod_{j=1}^{D} \mathrm{Cat}(x_j|\boldsymbol{\mu}_{jc}),$$

where $\boldsymbol{\mu}_{jc}$ is a histogram over the K possible values for x_j in class c.

Naive Bayes Classifiers

In the case of real-valued features, we can use the Gaussian distribution:

$$p(\mathrm{x}|y = c, \boldsymbol{\theta}) = \prod_{j=1}^{D} \mathcal{N}\left(\mu_{jc}|\sigma_{jc}^2\right),$$

where μ_{jc} is the mean of feature j in objects of class c, and σ_{jc}^2 is its variance.

MLE for NBC

The MLE for the class prior is given by

$$\hat{\pi}_c = \frac{N_c}{N}$$

where $N_c \triangleq \sum_i \mathbb{I}(y_i = c)$ is the number of examples in class c.

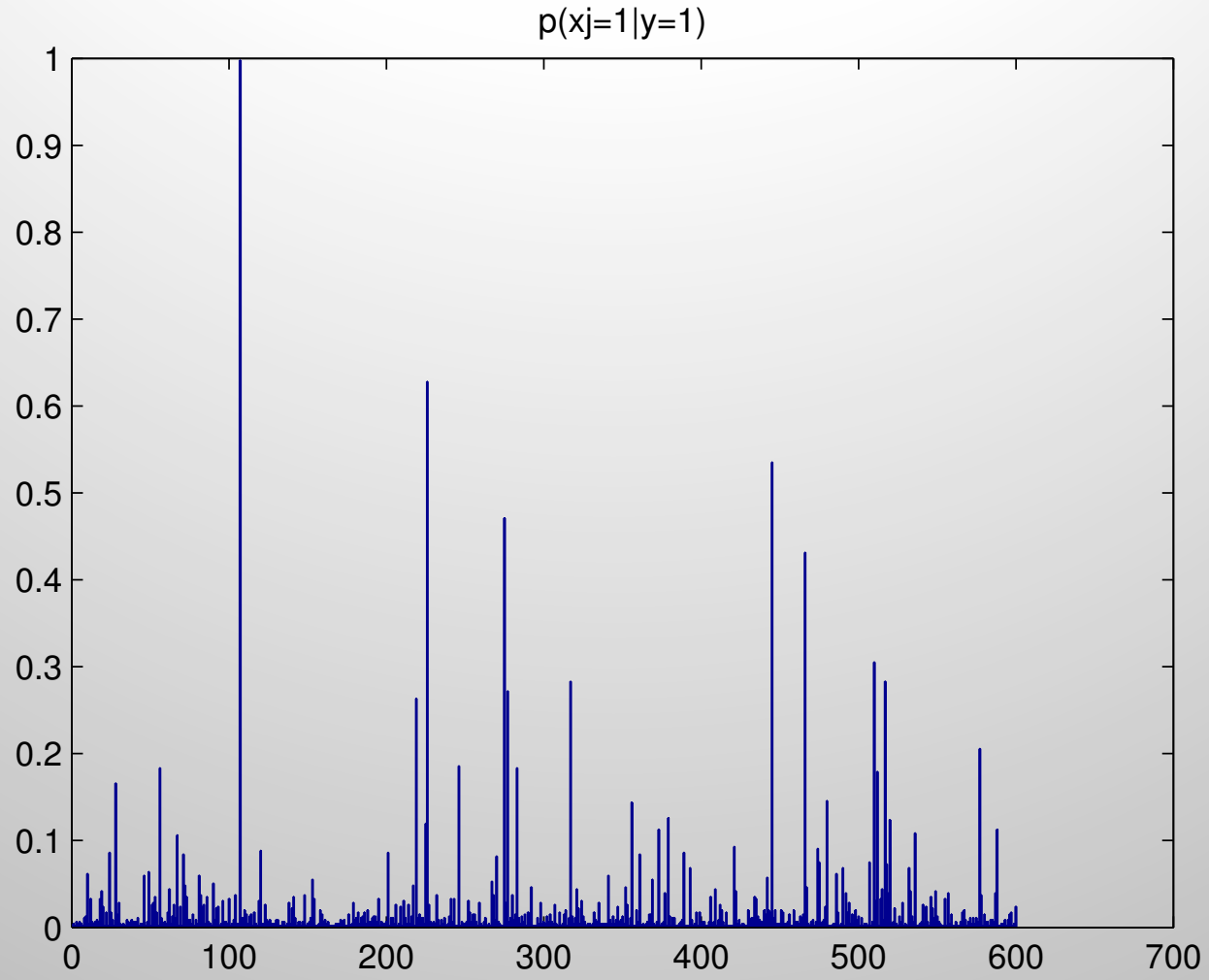
MLE for NBC

The MLE for the likelihood depends on the type of distribution we choose to use for each feature. For simplicity, let us suppose all features are binary, so $x_j|y = c \sim \text{Ber}(\theta_{jc})$. In this case, the MLE becomes

$$\hat{\theta}_{jc} = \frac{N_{jc}}{N_c}$$

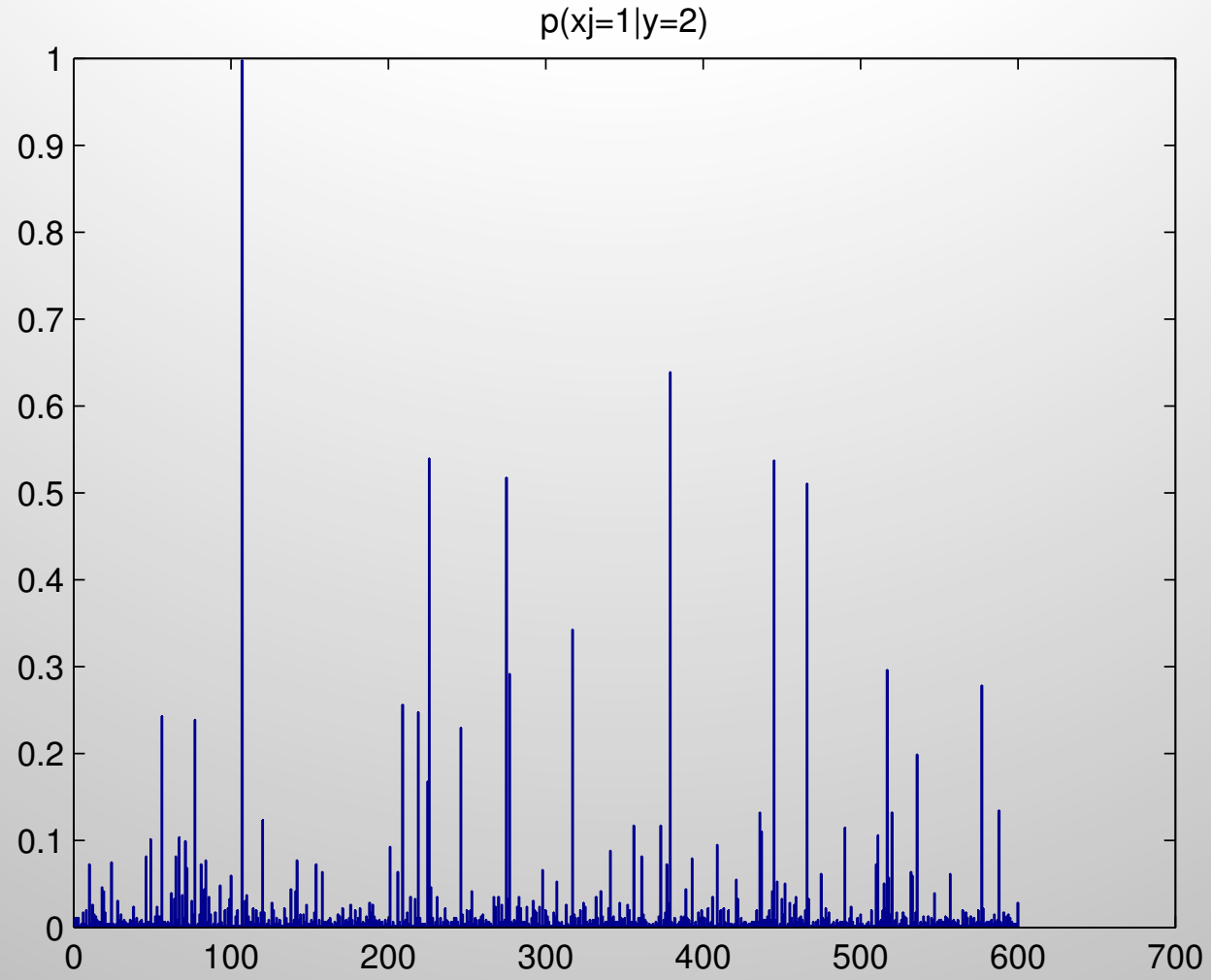
where $N_{jc} \triangleq \sum_i \mathbb{I}\left(x_{ij} = 1, y_i = c\right)$ is the number of examples in class c where j turns on.

MLE for NBC



Class conditional density for “X windows”

MLE for NBC



Class conditional density for “MS windows”

Using NBC for Prediction

At test time the goal is to compute

$$p(y = c|\mathbf{x}, \mathcal{D}) \quad \propto \quad p(y = c|\mathcal{D}) \prod_{j=1}^{D} p(x_j|y = c, \mathcal{D})$$

Feature Selection Using Mutual Information

Since an NBC is fitting a joint distribution over potentially many features, it can suffer from overfitting. In addition, the run-time cost is $O(D)$, which may be too high for some applications.

One common approach to tackling both of these problems is to perform feature selection, to remove “irrelevant” features that do not help much with the classification problem.

Feature Selection Using Mutual Information

The simplest approach to feature selection is to evaluate the relevance of each feature separately, and then take the top K, where K is chosen based on some tradeoff between accuracy and complexity. This approach is known as variable **ranking**, **filtering**, or **screening**.

One way to measure relevance is to use mutual information.

Feature Selection Using Mutual Information

Consider two random variables, X and Y. Suppose we want to know how much knowing one variable tells us about the other. We could compute the correlation coefficient, but this is only defined for real-valued random variables, and furthermore, this is a very limited measure of dependence.

Feature Selection Using Mutual Information

A more general approach is to determine how similar the joint distribution $p(X, Y)$ is to the factored distribution $p(X)p(Y)$. This is called the mutual information or MI and is given by

$$\mathbb{I}(X;Y) \triangleq \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

where $\mathbb{I}(X;Y) \geq 0$ with equality if and only if $p(X,Y) = p(X)p(Y)$. That is, the MI is zero iff the variables are independent.

Feature Selection Using Mutual Information

Mutual information between between X_j and class label Y is given by

$$I(X,Y) = \sum_{x_j} \sum_{y} p(x_j, y) \log \frac{p(x_j, y)}{p(x_j)p(y)}$$

Feature Selection Using Mutual Information

If the features are binary, the MI can be computed as follows:

$$I_j = \sum_c \left[\theta_{jc} \pi_c \log \frac{\theta_{jc}}{\theta_j} + (1 - \theta_{jc}) \pi_c \log \frac{1 - \theta_{jc}}{1 - \theta_j} \right]$$

Where $\pi_c = p(y = c)$, $\theta_{jc} = p(x_j = 1 | y = c)$, and $\theta_j = p(x_j = 1) = \sum_c \pi_c \theta_{jc}$.

Feature Selection Using Mutual Information

class 1	prob	class 2	prob	highest MI	MI
subject	0.998	subject	0.998	windows	0.215
this	0.628	windows	0.639	microsoft	0.095
with	0.535	this	0.540	dos	0.092
but	0.471	with	0.538	motif	0.078
you	0.431	but	0.518	window	0.067

5 most likely words for class 1 (X windows) and class 2 (MS Windows)
5 words with highest mutual information with class label