CS6301: R For Data Scientists

LECTURE 19: NAÏVE BAYES

Classification Problems

We now turn our attention to classification problems …

In this setting, we have predictor variables as before (real, nominal, categorical), but our response variable is categorical

We will often assume a binary response for now, but the methods we will explore can work with multi-valued response variables

Most of these methods (often called *classfiers*) will rely on approximating a probability for the (categorical) value of y, as opposed to approximating the value directly

The method we look at here is Naïve Bayes, and (despite the name) is still a widely used technique

Classification Problems - Notes

Classification problems tend to be easier to solve than regression problems

These come up in many situations (default vs no default, spam vs ham, win vs lose)

Often times it is a good idea to see if a regression problem can be converted to a classification problem ….

- Instead of predicting the person's age, can we predict adult versus child?

Suppose we are given data which is *not* labeled … can we still do a classification problem?

- Yes – do clustering to find an optimal number of clusters, and assign cluster numbers to the data points. Then build a classifier to predict the cluster a data point belongs to

Bayes Theorem

Recall from probability theory that Bayes Rule (or Bayes Theorem) related conditional probabilities: If *A* and *B* are events, then

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

We often use the Law of Total probability to replace the *P(A)* term in the denominator …

How can we use this to approximate a value for *y*? Idea: Given the training set, we will try to approximate the probabilities for the values of *y* … then pick the value with the largest probability

Bayes Optimal Classifier

Let's state this more formally:

Suppose our response variable *y* takes on values $\{v_1, v_2, \ldots v_k\}$ and we have *n* predictors (this is the general case).

Suppose we are given a new predictor value $\mathbf{a} =< a_1, a_2, \ldots, a_n >$, and suppose each attribute is categorical. We want to find

$$\max_j P(v_j | < a_1, a_2, \ldots, a_n >)$$

This is called the *Bayes Optimal Classifier* (or just Bayes Classifier)

Naïve Bayes Classifier

What happened to the training data? We will see how it is used later …

Now we can use Bayes to rewrite the conditional probability …

$$v_{pred} = \max_j \frac{P(< a_1, a_2, \dots, a_n >| v_j)\, P(v_j)}{P(< a_1, a_2, \dots, a_n >)} = \max_j P(< a_1, a_2, \dots, a_n >| v_j)\, P(v_j)$$

Note we dropped the denominator since it does not depend upon *j*

So given a new observation, all we need to do is approximate the probabilities on the right (for each *j*) and then take the largest value. Note: <u>There is no model!</u>

Naïve Bayes Classifier

Problem: The conditional probabilities $P(< a_1, a_2, \dots, a_n > | v_j)$ are hard to find, unless we sample all possible predictor values.

Key assumption: The variables that make up the attribute tuple are conditionally independent, so we can calculate the conditional probability as

$$P(< a_1, a_2, \dots, a_n > | v_j) = \prod_i P(a_i \mid v_j)$$

So we now can define the Naïve Bayes classifier as

$$v_{NB} = \max_j P(v_j) \prod_i P(a_i \mid v_j)$$

Naïve Bayes Classifier

How do we calculate the probabilities?

Here is where the training data comes in …

For $P(v_j)$, go to the training set and calculate what proportion takes on value v_j.

For $P(a_i \mid v_j)$, go to training set and …

- For each $vj_,$ calculate what proportion of examples have value a_i for the i^{th} attribute.

Note this can be done quickly!

Naïve Bayes Classifier

Does it work? Yes!

The “independence” assumption does not seem to hurt in practice

Naïve Bayes is a good classifier for larger data sets, because it tends to be fast

Another advantage – this approach is easy to evolve, since there is no model

- If data is changing or flowing, simply update the probabilities – do not use the entire dataset to compute a model

Downside – not good at predicting response variables that depend on combinations of predictors

Problem – Insufficient Training Data

Recall how we calculated $P(a_i \mid v_j)$: Go to training set and …

- For each $v_{j,}$ calculate what proportion of examples have value a_i for the ith attribute.

What is only a few samples (in a large training set) have this attribute value? We could get a very small probability.

- If the actual probability is small and the sample size is not large enough, we may not have any examples of this attribute value, so we would conclude the probability is zero!
- If we get a new sample with this value, we may not even be able to classify it
- Due to the nature of Naïve Bayes, we could get misleading results

Insufficient Training Data

This is typically handled by modifying the probability calculation

Original calculation:

$$P(a_i \mid v_j) = \frac{number\ samples\ with\ a_i, v_j}{number\ of\ samples\ with\ v_j}$$

Let n_c be the number of samples with a_i and v_j and let n be the number of samples with v_j.

We can define a modified probability by

$$\frac{n_c + mp}{n + m}$$

Here m is a weight (usually determined by the problem) and p is a prior probability for the attribute

Numerical Predictors – Gaussian Bayes

Gaussian Bayes assumes that, for each value of *y*, the predictors are normally distributed (multi-variate normal distribution)

The training data is used to calculate approximate means and covariance matrices for these distributions

Then, given a new observation, we use the approximate distributions and the prior distributions to approximate the Bayes Classifier

Fast and flexible, but does assume a normal distribution for the predictors for each value

Gaussian Bayes Classifier

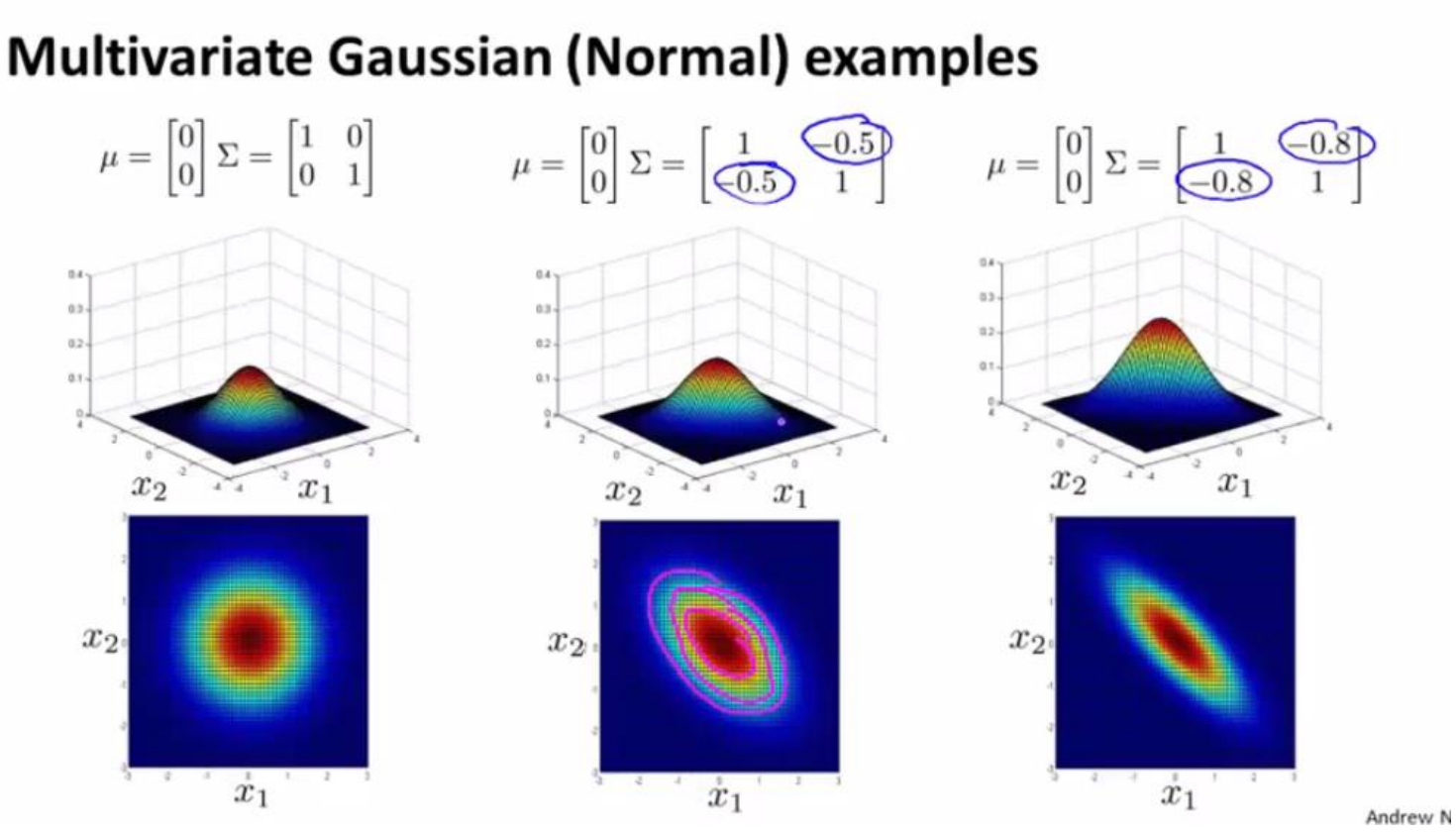
We now find

$$P(v_j | < a_1, a_2, \ldots, a_n >) = \frac{P(< a_1, a_2, \ldots, a_n > | v_j)\, P(v_j)}{P(< a_1, a_2, \ldots, a_n >)}$$

$$\approx \frac{1}{(2\pi)^{\frac{n}{2}} ||\boldsymbol{\Sigma}_i||^2} \exp\left[-\frac{1}{2}(\boldsymbol{a} - \boldsymbol{\mu_i})^t \Sigma_{\boldsymbol{i}} (\boldsymbol{a} - \boldsymbol{\mu}_i)\right] P(v_j)$$

Here we are assuming a multivariate normal distribution; $\boldsymbol{\mu}_i$ is the mean vector and $\boldsymbol{\Sigma}_i$ is a covariance matrix. These are estimated from the training data.

Multivariate Gaussian Distribution



Naïve Bayes in R

The classic implementation is in the package **e1071**

Works with categorical or numeric data

There is a new (Jan 2018) “high performance” package for Naïve Bayes … **naivebayes**

Links

https://www.r-bloggers.com/understanding-naive-bayes-classifier-using-r/

https://cran.r-project.org/web/packages/naivebayes/naivebayes.pdf

https://www.r-bloggers.com/naive-bayes-classification-in-r-part-2/

https://mdav.ece.gatech.edu/ece-6254-spring2017/notes/04-plugin-rules.pdf