Naive Bayes Classifier From Scratch

12 December 2016 / Machine Learning / Supervised Learning - Classification

Naive bayes is simple classifier known for doing well when only a small number of observations is available. In this tutorial we will create a gaussian naive bayes classifier from scratch and use it to predict the class of a previously unseen data point. This tutorial is based on an example on Wikipedia's naive bayes classifier page (https://en.wikipedia.org/wiki/Naive_Bayes_classifier), I have implemented it in Python and tweaked some notation to improve explanation.

Preliminaries

```
import pandas as pd
import numpy as np
```

Create Data

Our dataset is contains data on eight individuals. We will use the dataset to construct a classifier that takes in the height, weight, and foot size of an individual and outputs a prediction for their gender.

```
# Create an empty dataframe
data = pd.DataFrame()

# Create our target variable
data['Gender'] = ['male','male','male','male','female','female','female','female']

# Create our feature variables
data['Height'] = [6,5.92,5.58,5.92,5,5.5,5.42,5.75]
data['Weight'] = [180,190,170,165,100,150,130,150]
data['Foot_Size'] = [12,11,12,10,6,8,7,9]

# View the data
data
```

	Gender	Height	Weight	Foot_Size
0	male	6.00	180	12
1	male	5.92	190	11
2	male	5.58	170	12
3	male	5.92	165	10
4	female	5.00	100	6
5	female	5.50	150	8
6	female	5.42	130	7
7	female	5.75	150	9

The dataset above is used to construct our classifier. Below we will create a new person for whom we know their feature values but not their gender. Our goal is to predict their gender.

```
# Create an empty dataframe
person = pd.DataFrame()

# Create some feature values for this single row
person['Height'] = [6]
person['Weight'] = [130]
person['Foot_Size'] = [8]

# View the data
person
```

	Height	Weight	Foot_Size
0	6	130	8

Bayes Theorem

Bayes theorem is a famous equation that allows us to make predictions based on data. Here is the classic version of the Bayes theorem:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

This might be too abstract, so let us replace some of the variables to make it more concrete. In a bayes classifier, we are interested in finding out the class (e.g. male or female, spam or ham) of an observation *given* the data:

$$p(\text{class} \mid \mathbf{data}) = \frac{p(\mathbf{data} \mid \text{class}) * p(\text{class})}{p(\mathbf{data})}$$

where:

- class is a particular class (e.g. male)
- **data** is an observation's data
- $p(\text{class} \mid \mathbf{data})$ is called the posterior
- $p(\text{data}|\text{class})$ is called the likelihood
- $p(\text{class})$ is called the prior
- $p(\mathbf{data})$ is called the marginal probability

In a bayes classifier, we calculate the posterior (technically we only calculate the numerator of the posterior, but ignore that for now) for every class for each observation. Then, classify the observation based on the class with the largest posterior value. In our example, we have one observation to predict and two possible classes (e.g. male and female), therefore we will calculate two posteriors: one for male and one for female.

$$p(\text{person is male} \mid \mathbf{person's\ data}) = \frac{p(\mathbf{person's\ data} \mid \text{person is male}) * p(\text{person is male})}{p(\mathbf{person's\ data})}$$

$$p(\text{person is female} \mid \mathbf{person's\ data}) = \frac{p(\mathbf{person's\ data} \mid \text{person is female}) * p(\text{person is female})}{p(\mathbf{person's\ data})}$$

Gaussian Naive Bayes Classifier

A gaussian naive bayes is probably the most popular type of bayes classifier. To explain what the name means, let us look at what the bayes equations looks like when we apply our two classes (male and female) and three feature variables (height, weight, and footsize):

$$\text{posterior (male)} = \frac{P(\text{male})\, p(\text{height} \mid \text{male})\, p(\text{weight} \mid \text{male})\, p(\text{foot size} \mid \text{male})}{\text{marginal probability}}$$

$$\text{posterior (female)} = \frac{P(\text{female})\, p(\text{height} \mid \text{female})\, p(\text{weight} \mid \text{female})\, p(\text{foot size} \mid \text{female})}{\text{marginal probability}}$$

Now let us unpack the top equation a bit:

- $P(\text{male})$ is the prior probabilities. It is, as you can see, simply the probability an observation is male. This is just the number of males in the dataset divided by the total number of people in the dataset.
- $p(\text{height} \mid \text{female})\, p(\text{weight} \mid \text{female})\, p(\text{foot size} \mid \text{female})$ is the likelihood. Notice that we have unpacked **person's data** so it is now every feature in the dataset. The "gaussian" and "naive" come from two assumptions present in this likelihood:
    1. If you look each term in the likelihood you will notice that we assume each feature is uncorrelated from each other. That is, foot size is independent of weight or height etc.. This is obviously not true, and is a "naive" assumption - hence the name "naive bayes."
    2. Second, we assume have that the value of the features (e.g. the height of women, the weight of women) are normally (gaussian) distributed. This means that $p(\text{height} \mid \text{female})$ is calculated by inputing the required parameters into the probability density function of the normal distribution:

$$p(\text{height} \mid \text{female}) = \frac{1}{\sqrt{2\pi \text{variance of female height in the data}}}\, e^{-\frac{(\text{observation's height} - \text{average height of females in the data})^2}{2\text{variance of female height in the data}}}$$

- marginal probability is probably one of the most confusing parts of bayesian approaches. In toy examples (including ours) it is completely possible to calculate the marginal probability. However, in many real-world cases, it is either extremely difficult or impossible to find the value of the marginal probability (explaining why is beyond the scope of this tutorial). This is not as much of a problem for our classifier as you might think. Why? Because we don't care what the true posterior value is, we only care which class has a the highest posterior value. And because the marginal probability is the same for all classes 1) we can ignore the denominator, 2) calculate only the posterior's numerator for each class, and 3) pick the largest numerator. That is, we can ignore the posterior's denominator and make a prediction solely on the relative values of the posterior's numerator.

Okay! Theory over. Now let us start calculating all the different parts of the bayes equations.

Calculate Priors

Priors can be either constants or probability distributions. In our example is the simply the probability of being a gender. Calculating this is simple:

```
# Number of males
n_male = data['Gender'][data['Gender'] == 'male'].count()

# Number of males
n_female = data['Gender'][data['Gender'] == 'female'].count()

# Total rows
total_ppl = data['Gender'].count()
```

```
# Number of males divided by the total rows
P_male = n_male/total_ppl

# Number of females divided by the total rows
P_female = n_female/total_ppl
```

Calculate Likelihood

Remember that each term (e.g. $p(\text{height} \mid \text{female})$) in our likelihood is assumed to be a normal pdf. For example:

$$p(\text{height} \mid \text{female}) = \frac{1}{\sqrt{2\pi \text{variance of female height in the data}}} e^{-\frac{(\text{observation's height} - \text{average height of females in the data})^2}{2\text{variance of female height in the data}}}$$

This means that for each class (e.g. female) and feature (e.g. height) combination we need to calculate the variance and mean value from the data. Pandas makes this easy:

```
# Group the data by gender and calculate the means of each feature
data_means = data.groupby('Gender').mean()

# View the values
data_means
```

	Height	Weight	Foot_Size
Gender			
female	5.4175	132.50	7.50
male	5.8550	176.25	11.25

```
# Group the data by gender and calculate the variance of each feature
data_variance = data.groupby('Gender').var()

# View the values
data_variance
```

	Height	Weight	Foot_Size
Gender			
female	0.097225	558.333333	1.666667
male	0.035033	122.916667	0.916667

Now we can create all the variables we need. The code below might look complex but all we are doing is creating a variable out of each cell in both of the tables above.

```
# Means for male
male_height_mean = data_means['Height'][data_variance.index == 'male'].values[0]
male_weight_mean = data_means['Weight'][data_variance.index == 'male'].values[0]
male_footsize_mean = data_means['Foot_Size'][data_variance.index == 'male'].values[0]

# Variance for male
male_height_variance = data_variance['Height'][data_variance.index == 'male'].values[0]
male_weight_variance = data_variance['Weight'][data_variance.index == 'male'].values[0]
male_footsize_variance = data_variance['Foot_Size'][data_variance.index == 'male'].values[0]

# Means for female
female_height_mean = data_means['Height'][data_variance.index == 'female'].values[0]
female_weight_mean = data_means['Weight'][data_variance.index == 'female'].values[0]
female_footsize_mean = data_means['Foot_Size'][data_variance.index == 'female'].values[0]

# Variance for female
female_height_variance = data_variance['Height'][data_variance.index == 'female'].values[0]
female_weight_variance = data_variance['Weight'][data_variance.index == 'female'].values[0]
female_footsize_variance = data_variance['Foot_Size'][data_variance.index == 'female'].values[0]
```

Finally, we need to create a function to calculate the probability density of each of the terms of the likelihood (e.g. $p(\text{height} \mid \text{female})$).

```
# Create a function that calculates p(x | y):
def p_x_given_y(x, mean_y, variance_y):

    # Input the arguments into a probability density function
    p = 1/(np.sqrt(2*np.pi*variance_y)) * np.exp((-(x-mean_y)**2)/(2*variance_y))

    # return p
    return p
```

Apply Bayes Classifier To New Data Point

Alright! Our bayes classifier is ready. Remember that since we can ignore the marginal probability (the demoninator), what we are actually calculating is this:

$$\text{numerator of the posterior} = P(\text{female})\, p(\text{height} \mid \text{female})\, p(\text{weight} \mid \text{female})\, p(\text{foot size} \mid \text{female})$$

To do this, we just need to plug in the values of the unclassified person (height = 6), the variables of the dataset (e.g. mean of female height), and the function (`p_x_given_y`) we made above:

```
# Numerator of the posterior if the unclassified observation is a male
P_male * \
p_x_given_y(person['Height'][0], male_height_mean, male_height_variance) * \
p_x_given_y(person['Weight'][0], male_weight_mean, male_weight_variance) * \
p_x_given_y(person['Foot_Size'][0], male_footsize_mean, male_footsize_variance)
```

```
6.1970718438780782e-09
```

```
# Numerator of the posterior if the unclassified observation is a female
P_female * \
p_x_given_y(person['Height'][0], female_height_mean, female_height_variance) * \
p_x_given_y(person['Weight'][0], female_weight_mean, female_weight_variance) * \
p_x_given_y(person['Foot_Size'][0], female_footsize_mean, female_footsize_variance)
```

```
0.00053779091836300176
```

Because the numerator of the posterior for female is greater than male, then we predict that the person is female.

Everything on this site is avaliable on GitHub. Head on over and submit an issue. (https://github.com/chrisalbon/notes_on_data_science_machine_learning_and_artificial_intelligence/issues/new) You can also message me directly on Twitter (https://twitter.com/chrisalbon).
