Naïve Bayes Algorithm

Model formalization

Consider a problem with two classes $Y \in \{y_1, y_2\}$. Each data sample X_i is characterized by a set of J features, thus, $X_i = (X_{i1}, \dots, X_{iJ})$. The folowing table illustrates this example.

i	Y	X_{i1}	…	X_{iJ}
1	y_1	4	…	1
2	y_2	2	…	0
3	y_2	3	…	1
…	…	…	…	…

To determine the class of a given sample, we can use the Bayes law:

$$P(Y = y_l|X_i) = \frac{P(Y = y_l)P(X_i|Y = y_l)}{P(X_i)}$$

The first step in computing the above model concerns the $P(X_i|Y = y_l)$. The class prior is computed straightforwardly. Since we know X_i is composed by J features, we can re-write the above expression as:

$$P(Y = y_l|X_i) = \frac{P(Y = y_l)P\big(X_i = (X_{i1}, \dots, X_{iJ})|Y = y_l\big)}{P\left(X_i = (X_{i1}, \dots, X_{iJ})\right)}$$

If we make the naïve assumption that all $X_{i1}, \dots, X_{iJ}$ are conditionally independent given the class label, then we can make the following simplification:

$$P\big(X_i = (X_{i1}, \dots, X_{iJ})|Y = y_l\big) = \prod_{j=1}^{J} P\big(X_{ij}|Y = y_l\big)$$

In the two class scenario, we only need to determine which class model maximizes the sample likelihood, i.e.,

$$P(Y = y_l|X_1) > P(Y = y_2|X_i)$$

Formally, we have:

$$Y_i = arg \max_{y_l \in \{y_1, y_2\}} P(Y = y_l)P(X_i|Y = y_l)$$

To avoid underflow situations, a common trick is to compute the logarithm of the above expression:

$$Y_i = arg \max_{y_l \in \{y_1, y_2\}} \left[log\big(P(Y = y_l)\big) + \sum_{j=1}^{J} log\left(P\big(X_{ij}|Y = y_l\big)\right) \right]$$

Learning phase

The class prior is computed as:

$$\hat{P}(Y = y_l) = \frac{\#\{\, Y = y_l\}}{|D|}$$

Assume that each feature X_{ij} can assume a value x_k from the set of all possible values. Then we can compute the conditional

$$\hat{P}\left(X_{*j} = x_k | Y = y_l\right) = \frac{\#D\left\{X_{*j} = x_k \ AND \ Y = y_l\right\} + \alpha}{\#D\{Y = y_l\} + \alpha \cdot J}$$

Prediction phase

For an unseen sample X_n, the class posterior is computed as:

$$log\left(\hat{P}(Y = y_l)\right) + \sum_{j=1}^{J} \sum_{k=1}^{K} \delta\left(X_{jn} = x_k | Y = y_l\right)$$

where

$$\delta\left(X_{jn} = x_k | Y = y_l\right) = \begin{cases} log\left(\hat{P}(X_{*k} | Y = y_l)\right) & , if \ X_{nj} = x_k \\ 0 & , if \ X_{nj}! = x_k \end{cases}$$

Mushroom: Edible vs Poisonous

This data set includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family (pp. 500-525). Each species is identified as <u>definitely edible</u> or <u>definitely poisonous</u>.

Each mushroom sample is characterized in terms of the following aspects:

cap-shape	gill-attachment	stalk-root	veil-color
cap-surface	gill-spacing	stalk-surface-above-ring	ring-number
cap-color	gill-size	stalk-surface-below-ring	ring-type
bruises?	gill-color	stalk-color-above-ring	spore-print-color
odor	stalk-shape	stalk-color-below-ring	population
		veil-type	habitat

Each one of these features are detailed in the data files.

There is a total of 8124 instances, of which 4208 are edible and 3916 are poisonous.

Exercise

Consider the script on the last page. It defines an experimental setup for a Naïve Bayes classifier. Read the code, and implement the Naïve Bayes parameters estimation method **nb_learn** (the learning phase) and the posterior computation method **nb_probability** (the prediction phase).

Example

Consider the following dataset with two classes and 3 dimensional data.

i	Y	X_{i1}	X_{i2}	X_{i3}
1	y_1	0	0	1
2	y_2	2	2	0
3	y_1	0	1	1
4	y_2	1	2	1
5	y_2	0	3	2

The prior of each class is:

$$\hat{P}(Y = y_1) = \frac{\#\{\,Y = y_1\}}{|D|} = \frac{2}{5} \qquad \hat{P}(Y = y_2) = \frac{\#\{\,Y = y_2\}}{|D|} = \frac{3}{5}$$

Assume that each feature X_{ij} can assume a value $x_k \in \{0,1,2,3\}$. Then we can compute the conditional for each class:

$$\hat{P}\left(X_{*j} = x_k | Y = y_l\right) = \frac{\#D\left\{X_{*j} = x_k \; AND \; Y = y_l\right\} + \alpha}{\#D\{Y = y_l\} + \alpha \cdot J}$$

$$Y = y_1$$

x_k	X_{i1}	X_{i2}	X_{i3}
0	2	1	0
1	0	1	2
2	0	0	0
3	0	0	0

$$Y = y_2$$

x_k	X_{i1}	X_{i2}	X_{i3}
0	1	0	1
1	1	0	1
2	1	2	1
3	0	1	0

```
clear all;
close all;
clc;

% Load data and find unique feature values
% (this is only valid for discrete NB);
data=loaddata('agaricus-lepiota.data');
classes=data(:,1);
data=data(:,2:end);
feats = unique(data);

% Split edible and poisonous data
ix_e = find(classes=='e');
ix_p = find(classes=='p');
data_p = data(ix_p,:);
data_e = data(ix_e,:);

% Make random permutation
% and split train/test edible data
data_e = data_e(randperm(length(data_e)),:);
eTest = data_e(1:floor(length(data_e)/2),:);
eTrain = data_e(ceil(length(data_e)/2):end,:);

% Make random permutation
% and split train/test poisonous data
data_p = data_p(randperm(length(data_p)),:);
pTest = data_p(1:floor(length(data_p)/2),:);
pTrain = data_p(ceil(length(data_p)/2):end,:);

% Report data split
totTrain=size(eTrain,1)+size(pTrain,1);
totTest=size(eTest,1)+size(pTest,1);
fprintf('Training set: %i examples\n',totTrain);
fprintf('Test set: %i examples\n',totTest);

% Learn NB parameters for Edible class
edible_bayes.logPrior=log([size(eTrain,1)/totTrain]);
edible_bayes.pdm=nb_learn(eTrain,feats,1e-16);

% Learn NB parameters for Poisonois class
poisonous_bayes.logPrior=log([size(pTrain,1)/totTrain]);
poisonous_bayes.pdm=nb_learn(pTrain,feats,1e-16);

% Test edible mushrooms
edible_hypothesis = nb_probability(eTest, feats, edible_bayes);
poisonous_hypothesis = nb_probability(eTest, feats, poisonous_bayes);

e_test_labels = sum(edible_hypothesis > poisonous_hypothesis);

% Test poisonous mushrooms
edible_hypothesis = nb_probability(pTest, feats, edible_bayes);
poisonous_hypothesis = nb_probability(pTest, feats, poisonous_bayes);

p_test_labels = sum(edible_hypothesis < poisonous_hypothesis);

% Output resutls
fprintf('\nEdible detection precision: %2.1f%%\n',100*e_test_labels/(length(eTest)));
fprintf('Edible missclassified: %d\n',length(eTest)-e_test_labels);
fprintf('\nPoisonous detection precision:
%2.1f%%\n',100*p_test_labels/(length(pTest)));
fprintf('Poisonous missclassified: %d\n',length(pTest)-p_test_labels);
```