Naive Bayes' Algorithm and its application

Zhengqi Lin, Rong Zhang, Xi Lin, Lucy Shao and Siwen Tang

November 27, 2016

1 Introduction of the Naive Bayes algorithm and Logisitic Regression

Naive Bayes Classifier is a simple but powerful probabilistic classifier based on Bayes's theorem. The algorithm requires two assumptions. The first one is the strong conditional independence between the independent variables among dependent variable. The second one is the distribution of independent variables condition on dependent variable is a Gaussian distribution.

Naive Bayes Classifier can be applied in diverse cases, such as classifying gender, weather, document, investment, disease, etc. Given a set of data in a problem, we assign class labels to problem instances, represented as feature values, and model the problem based on the conditional probability of the features. We implemented the algorithm of Naive Bayes Classifier and applied it to find safe equity investments (stocks) and exclude risk investments. Due to the over-simplified assumptions of the Naive Bayes Classifier, we also implemented another approach, logistic regression, in complement to the Naive Bayes Classifier, which performs better when the assumptions do not hold.

Our algorithm includes two parts: Naive Bayes for continuous inputs and logistic regression. To implement the Naive Bayes algorithm, we use a fundamental equation derived from Bayes's theorem. Let Y be any discrete-valued random variable, and $X_1, X_2, ..., X_n$ be any discrete or real-valued attributes,

$$P(Y = y_k | X_1 ... X_n) = \frac{P(Y = y_k) \prod_i P(X_i | Y = y_k)}{\sum_j P(Y = y_j) \prod_i P(X_i | Y = y_j)}$$

In the case of continuous inputs X_i, we assume that for each possible discrete value y_k of Y, the distribution of each continuous X_i is Gaussian. The distribution of each X_i is defined by a mean and standard deviation that is specific to X_i and y_k. We estimate the mean and standard deviation of these Gaussians by maximum likelihood estimation.

$$\hat{\mu}_{ik} = \frac{1}{\sum_j \delta(Y^j = y_k)} \sum_j X_i^j \delta(Y^j = y_k)$$

$$\hat{\sigma}_{ik}^2 = \frac{1}{\sum_j \delta(Y^j = y_k)} \sum_j (X_i^j - \hat{\mu}_{ik})^2 \delta(Y^j = y_k)$$

In logistic regression, we take Y as a boolean variable and X as a vector containing discrete or continuous variables. We assume a parametric form for the distribution P(Y|X).

$$P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

$$P(Y=0|X) = \frac{\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}{1+\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

We can directly estimate the parameter values from the training data by maximizing the conditional data likelihood. We choose parameters W that satisfy

$$W \leftarrow \text{argmax}_W \prod_l P(Y^l|X^l, W)$$

where $W =< w_0, w_1, ..., w_n >$ is the vector of parameters, Y^l denotes the observed value of Y in the lth training example, and X^l denotes the observed value of X in the lth training example. The log of the conditional likelihood is thus given by

$$l(W) = \sum_l Y^l \ln P(Y^l = 1|X^l, W) + (1 - Y^l) \ln P(Y^l = 0|X^l, W)$$

The ith component of the vector gradient has the form

$$\frac{\partial l(W)}{\partial w_i} = \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1|X^l, W))$$

We use standard gradient ascent to optimize the weights W. Beginning with initial weights of zero, we repeatedly update the weights in the direction of the gradient, on each iteration changing every weight w_i according to

$$w_i \leftarrow w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1|X^l, W))$$

where η is a small constant (0.01) which determines the step size. With the estimation of the parameters, we can find the conditional probability of Y in each class given the feature values. The class in which Y has the largest probability is the class that we classify Y into.

2 Financial application of the algorithm

2.1 Introduction

We focus on the financial application of Naive Bayes's and Logistic Regression Algorithms. Our ultimate goal is to help investors identify companies/ stocks that are safe investments.

To achieve that, we would let users input several fundamental indicators and return the decision whether this stock is worth investing. The raw data is scraped from Finviz.com, a famous stock screener website. The stocks are drawn from middle and large companies (market captialization ranging from 2 billion to 200 billion). We transform analyst recommendation with values ranging from 1 (strong buy) to 5 (strong sell) from Finviz.com to Y that only has two possible values 1 and 0. If recommendation's value of a stock is smaller than 2.5 (hold), it is a safe investment and hence the value of 1 is assigned to Y. Otherwise, Y's value is 0. We have five continuous inputs as X, EPS (earnings per share), Volatility, Insiders' Transactions, Performance and ROI (return on investment).

We check the correlations between these five indicators to ensure that they are insignificant and so does their dependency on one another.

	EPS.this.Y	ROI	Insider.Trans	Perf.YTD	Volatility.M
EPS.this.Y	1.00000000	0.145205795	-0.059782232	-0.155219302	-0.09853701
ROI	0.14520580	1.000000000	-0.006027626	-0.006375562	-0.07981040
Insider.Trans	-0.05978223	-0.006027626	1.000000000	-0.055567490	0.06266818
Perf.YTD	-0.15521930	-0.006375562	-0.055567490	1.000000000	0.14244699
Volatility.M	-0.09853701	-0.079810396	0.062668182	0.142446990	1.00000000

The reason that we delete several significant financial indicators, such as P/S values and Market Capitalization, is that they have high correlations with the existing independent variables.

2.2 Results and Discussion

There are 1242 observations in our database. 819 of their Y-values are 1, and 423 of their Y-values are 0.

We test all 1242 observations with the two algorithms and check whether their results are consistent with the actual Y-values. Here are the statistics generated:

From the Naive Bayes Classifer, the accuracy is $812/1242 = 65.4\%$ in which 812 is the number that the algorithm produces the same values with the actual Y-values.

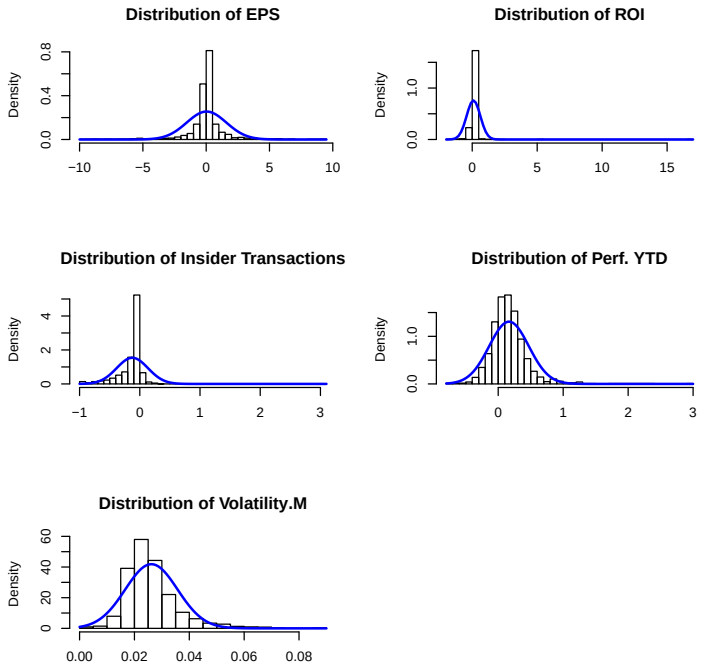
From the Logistic Regression, the accuracy is $766/1242 = 61.7\%$ in which 766 is the number that the algorithm produces the same values with the actual Y-values.

Combining the two algorithms, when they generate the same Y-value, the accuracy is $717/1098 = 65.3\%$, in which 1098 is the number that the two algorithms generate the same result, and among the 1098 results, 717 are the same with the actual Y-values.

From the investment point of view, the accuracy around 65% is acceptable. Even if this conclusion cannot help investors make final decision, it can contribute to the reviewing and screening of stocks to reduce risk.

Nevertheless, the prediction can be further enhanced. There are a few flaws in our model and we are working to fix them correspondingly:

I. The distributions of continuous independent variables are not all strictly Gussian.



From the graphs, we can see the distributions of performance and volatility are approximately normal. However, the distributions of EPS, ROI and insider transaction are mostly concentrated in a small interval. We want to find distributions that better fit these independent variables.

II. We have chosen two values of Y, 0 and 1, with the intention to generate the simple result that if the investment is safe or not. However, our Y-values are transformed from Analyst's Recommendation which range from 1 to 5. The transformation cuts relevant information from the orginial data and may cause inaccuracy in our prediction. Therefore, more classes can be assigned to Y, such as 1 to 5, to indicate degrees of investment from strong buy to strong sell and provide more specific decisions to investors.

References

[1] Tom M. Mitchell. *Machine Learning.* McGraw Hill, www.cs.cmu.edu/ tom/mlbook.html.