University of Pittsburgh
CS2750 Machine Learning
Professor Milos Hauskrecht

Handout 4
February 10, 2022

Problem assignment 4

Due: Thursday, February 17, 2022

In this assignment we explore "Pima" classification dataset and use it to learn and evaluate a number of classification models. The dataset is originally from the UC Irvine machine learning repository:
$http://www1.ics.uci.edu/\sim mlearn/MLRepository.html$. Please recall we performed initial exploratory analysis of the Pima dataset in Homework 1.

The pima dataset is given to you in three files: *pima.csv* that consists of the complete dataset, and *pima_train.csv* and *pima_test.csv* that split the original dataset to the training and testing sets.

Problem 1. Logistic regression model

You have been given file *hw4_problem1.py* that loads pima training and testing datasets, learns a logistic regression model from the training data, applies it to make predictions on the test data, and calculates evaluation statistics on the test data. At the end it plots additional assessment of the model in terms of the ROC and PR curves. Please familiarize yourself with the initial code. The code has been built with the help of scikit-learn python library and logistic regression model implemented therein.

Please modify *hw4_problem1.py* so that it calculates all of the following statistics:

- confusion matrix, misclassification error, SENS (recall), SPEC, PPV (precision), NPV on the test data
- confusion matrix, misclassification error and SENS (recall), SPEC, PPV (precision), NPV on the training data

Report the above results and submit the modified code. Inspect the results and assess whether the model overfits the training data. Explain your conclusion.

Problem 2. Naive Bayes model

The Naive Bayes model defines a generative classifier model in which all features are independent given the class label. In such a case the class-conditional densities over many input variables can be decomposed into a set of independent class-conditional densities, one for every input variable. For example, the conditional probability of an input $\mathbf{x} = \{x_1, x_2, \cdots, x_d\}$ given class 1 in the Naive Bayes model is decomposed as:

$$p(\mathbf{x}|y = 1) = \prod_{i=1}^{d} p(x_i|y = 1).$$

One important concern is the choice of an appropriate parameterization of class-conditional densities. Typically we do not choose the distributions arbitrarily, instead we want to make a good educated guess. Exploratory data analysis can help us greatly to recognize types of densities that appear to match the data the best.

Problem 2.1. Exploratory data analysis

We have performed the exploratory analysis of the Pima dataset in Problem set 1. Here we reuse the programs created there and apply them to study the density models we choose to parameterize our Naive Bayes model.

Part a. Modify and submit a program (*hw4_problem2.py*) such that:

- Loads the full pima dataset and its components X and Y. Divides X into two subsets - one with all examples with class "0", and another with all examples with class "1".
- Analyzes examples in two subsets using histograms. Histograms should give you more information about the shape of the distribution of attributes. Include histograms for the

Part b. Include histograms for each input attribute (1-8) and two classes in the report. What distribution/density would you use to fit the values of attributes 1 to 8 in the pima dataset? Choices one typically considers are Bernoulli, Binomial, Multinomial, Normal, Poisson, Gamma, exponential distributions.

Problem 2.2. The Naive Bayes classifier

The learning of the Naive Bayes model corresponds to the estimation of parameters of class-conditional distributions $p(x_i|y = 1), p(x_i|y = 0)$ for all input components i from data and

estimation of class priors $p(y = 1)$, $p(y = 0)$. Thus, the learning boils down to a number of 'smaller' density estimation problems.

Assume that class-conditional densities for pima dataset have the following form:

- Class-conditionals for inputs [1 5 7 8] take the form of exponential distribution. The exponential distribution is defined as:

$$p(x|\mu) = \frac{1}{\mu} exp\left[-\frac{x}{\mu}\right],$$

  where μ is the parameter. (Exponential distribution is a special case of the Gamma distribution and belongs to the exponential family).

- Class-conditionals for inputs [2 3 4 6] follow univariate normal distributions:

$$p(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right],$$

  with mean and standard deviation being the two parameters.

In addition assume that priors on classes follow a Bernoulli distribution:

$$p(x|\theta) = \theta^x (1-\theta)^{(1-x)} \text{ for } x \in \{0, 1\}.$$

Part a. Modify a python program *hw4_Naive_Bayes.py* so that it implements the learning, classification, and probabilistic prediction steps of the Naive Bayes model with the above class conditional distributions. The code should support the following functions:

- fit_NB: Learns ML estimates of all parameters of the Naive Bayes model from the (training) data. Please note the parameters define the prior of the class and class conditionals. There are 16 class-conditionals ($8 * 2 = 16$), one for every input component and class label. The parameters of the class conditionals depend on the distribution form used for the specific attribute. For the Gaussian the parameters are the mean and variance. Please use an unbiased estimate of the variance. For the exponential density use the ML estimate of μ.

- $predict\_NB$: Classifies the input, that it assigns either label 0, or 1 to the input in the test data based on the Naive Bayes model. Briefly, once the parameters of the Naive Bayes model are learned (estimated) , the decision about the class for a specific input $\mathbf{x}$ can be made by designing the appropriate discriminant functions. For the generative models there are based on class posteriors, thus a classification problems

boils down to the problem of comparison of posteriors of classes for $\mathbf{x}$. These are computed through the Bayes rule:

$$p(y=1|x) = \frac{\left[\prod_{i=1}^{d} p(x_i|y=1)\right] p(y=1)}{\left[\prod_{i=1}^{d} p(x_i|y=0)\right] p(y=0) + \left[\prod_{i=1}^{d} p(x_i|y=1)\right] p(y=1)}.$$

Please note that in order to make the best posterior class choice it is sufficient to compare the following discriminant functions based on log posteriors:

$$g_0(x) = \left[\sum_{i=1}^{d} \log p(x_i|y=0)\right] + \log p(y=0) \quad (1)$$

$$g_1(x) = \left[\sum_{i=1}^{d} \log p(x_i|y=1)\right] + \log p(y=1) \quad (2)$$

- *Predict_prob*: Calculates the probabilistic output $p(y=1|x)$ for the input vector x as defined above.

Please note that there are different ways of implementing the above functions, so please pick the approach that best fits your programming style.

Part b. After implementing the learning, classification and probabilistic outputs steps (functions) apply the code to

- Learn the parameters of the Naive Bayes model. Report the parameters of the model in the report.
- Apply/classify the inputs in the test data. One the class is assigned, calculate and report the confusion matrix, misclassification error, SENS, SPEC, PPV, NPV of the model. Please see the code in Problem 1 to calculate the evaluation statistics.
- Apply the model to calculate the probabilistic output $p(y=1|x)$ and use it to plot ROC and PR curves. Include the graphs in the report.

Submit the final code with your assignment.

Part c. Compare the classification models learned in Problem 1 and Problem 2. Which one is better? Explain the reasoning behind your choice.