Machine Learning

Catalano Alice Maria 5157341

Abstract—Naive Bayes is an algorithm for classification based on Bayes' Theorem with an assumption of independence among predictors. This is a MATLAB implementation of its behaviour. Laplace smoothing is applied.

Classification

In statistics, classification is the problem of identifying the category of an observation, based on a training set of data with observations already classified. The classifiers implement a classification algorithm, that maps input data to a category y.

Naive Bayes Classifier

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}$$

A Naive Bayes classifier is a probabilistic machine learning model that's used for classification task. The classifier is based on the Bayes theorem:

H is the hypothesis
X is the experimental observation
P(H) is the a priori probability of hypothesis H
P(X) is the marginal probability of X
P(X|H) is the likelihood of observing X when H is verified

$$P(X) = \sum_i P(X|H_i)P(H_i)$$

Using Bayes theorem means finding the probability of H happening, given that X has occurred, assuming that the predictors/features are independent. That is that the presence of one particular feature does not affect the others. For this reason, gets the adjective of naive.
The marginal probability of X, the probability of observing X in any case, is computed as:
i indicates the classes.

$$\begin{aligned} g_i(\mathbf{x}) &= P(t_i)[P(x_1|t_i) \times P(x_2|t_i) \times \ldots \times P(x_d|t_i)] \\ &= P(t_i)\prod_{j=1}^{d} P(x_j|t_i) \end{aligned}$$

Now because the "naïve" assumption is made, the classifier has the following outlook:
Where:
$x=[x_1, x_2, \ldots, x_d]$
$P(t_i|x) \propto P(x | t_i)P(ti) = P(x_1, x_2, x_3, \ldots, x_d | t_i) P(t_i)$
So the naive assumption is $Pr(x_1, \ldots, x_d | t_i) = Pr(x_1 | t_i) Pr(x_2 | t_i) \cdots Pr(x_d | t_i)$

Implementation

1. Data preprocessing

The data was downloaded from the Weather data set and the Weather data description. For use with Matlab it is convenient to convert all attribute values into integers>=1, conversion done by plain text editor.

The attributes are

Outlook (overcast, rainy, sunny)=[1,2,3]
Temperature (hot, cool, mild)=[1,2,3]
Humidity (high, normal)=[1,2]
Windy (false, true)=[1,2]

Classes:

Play (yes,no)=[1,2]

After the conversion the data file could be found in the DataSet.txt and uploaded directly in the Main.m function and divided in training set (*training*) and test set (*test*).

The training set will be composed of 10 random observation and a target set of data while the test set will be composed of the 4 remaining observation.

2. Training phase

The training task was implemented in the function ('Naive_trainer.m') that implements also a Laplace smoothing improvement. Due to this additive smoothing a priori information is needed: in the data preparation set, the information about the number of levels must be added. This means that for each data column the number of possible different values for that column is added; then to compute probabilities, Laplace smoothing is introduced in the formulas by adding terms that consider your prior belief. Since the classifier **don't know anything**, the prior belief is that all values are **equally probable**.

3. Testing phase

Once the Naive Bayes classifier has been trained it can accept the testing set and classify all the observation by computing the Bayes theorem. Two posteriori probability are computed ('Naive_classifier.m') and the higher one is chosen to complete the classification algorithm.

4. Error calculation

The testing set is a file composed, as well as the training set, of as many rows as the observation and as many columns as the attribute plus one more last target column for the class which can be used to validate the classifier. Once the classifier finishes the classification an error is computed to check the validity.

Conclusions

The error rate appears to float between 0.5 and 1, reaching the minor value of 0.

The error rate is demonstrated to be acceptable, confirming the efficiency of this kind of classification, even if is not that effective with a small set of data. Thanks to the smoothing algorithm the results are improved.