README

First of all, we didn't store the trained models inside separate classes. Instead we store them inside the "params" dictionary that belongs to class Classifier. Hence the following contains no classes but just modules and corresponding functions.

We implemented the decision tree algorithm together. Then Bo Lei worked on Naïve Bayes and Huzihan Lu worked on neural network. Apart from coding and writing together we also talked and shared a lot of ideas.

Should you find any bugs or have any trouble, please feel free to contact us.

Decision Tree

Function data_to_exmpale:
Converts a dataset to an example set. Dataset might contain continuous values but exmaple set doesn't.

Function train_decision_tree:
Trains a decision tree from input dataset and given parameter options and store it into the "params" dictionary.

Function predict:
Predict the labels of given input dataset.

Function test:
Predict the labels of the input dataset and then test accuracy.

#The followings are supportive functions
Function Tree:
This function generates one node in the tree as a dictionary and returns it. (cited from stackoverflow, used for faster tree implementation)
Function h:
Returns entropy of input number.
Function gain:
Returns information gain for an attribute under an example set.
Function gain_ratio:
Returns information gain ratio for an attribute under an example set.
Function pretty:
Print the struture of a tree such that we can determine if the tree is good. (cited from stackoverflow)
Function plurality_value:
Returns plurality value for an example set.

Naïve Bayes

Function *train_naive_bayes***:**
In this function, we calculate the likelihood of each different feature's values. The variable 'likelihood' contains likelihood, hypothesis prior, and the number of training data. Assume we have n labels, then the variable's first n elements represents the different labels. Under each label, we will have a list of dictionaries, where each dictionary represents an attributes. The keys are the value of those attributes, and the values are the number of that attribute value under the exact label. The n+1th element is the hypothesis prior of each class. The n+2th element is the amount of train data.

Function *predict:*
In this function, we calculate the predicted class of one test object. Using naïve Bayes model, we check the object's feature one by one to get the likelihood. Here we have two assumptions:
(1) In case of any p(wi|hi) = 0, which will lead the whole product to be 0, if one feature of the test object is not in the training data, we simply set the number of appearance as 1. Thus we need to add 1 to the total number in order to get the corresponding probability.
(2) Since probability is less than 1, if we have many probabilities multiplied, this will lead the computer to regard the result as 0. To avoid this problem, we use log(p) instead of p.
The function will return the predicted class of the test object.

Function *test:*
In this function, we calculate the predicted class of each of the test object. Combined with the actual class, we compute the accuracy, precision and recall of methods.

Function *getLabels*:
This function gets list of labels e.g. [0,1,2]

Function *getNumEachLabel*:
This function gets the numbers of data in each class e.g.[12,23,43]

Function *priorProbility*：
This function gets the hypothesis prior of each class e.g. [0.1,0.4,0.5]

Function *test:*
This function test the test data and return the accuracy, precision and recall of methods.

Neural Network

Function train_neural_net:
Trains a neural network from input dataset and given parameter options by calling back_prop_learning, and store it into the "params" dictionary. Initializations of network and weights are done here.
Function forward_propagation:
For a given example(entry), use forward propagation to compute activation values for each node in the neural network.
Function back_prop_learning:
Iteratively computes the errors from predicted and observed labels and use backward propagation to update the weights till finally converge.
Function Predict:
Predict the labels of given data using the trained model.
Function Test:
Predict the labels of given data using the trained model, and compute accuracy.
#The followings are supportive functions
Function g:
Returns the activation function of given input by computing sigmoid function.
Function dg:
Returns the derivative of activation function.
Function threshold:
This function thresholds a number and return 0 or 1.
Function compute_error:
This function returns the error of a model by computing the squared difference between predicted and expected values.