Data Mining

Lab 03

Introduction to Classification Techniques

Department of Computer Science
Kristianstad University
Course: DT584C
Master's in computer science

Student: Hazrat Ali <react.dev.se@gmail.com>
Teacher: Dawit Mengistu

Date: 8 January 2020

Objective

In this lab, you shall

1. Learn how to apply various classification techniques
2. Evaluate the performance of your implementation

- Note: You may implement your own classification code from scratch or build on available tools.

Task1:

- The following table consists of training data from an employee database. The data have been generalized. For example, “31 . . . 35” for age represents the age range of 31 to 35. For a given row entry, count represents the number of data tuples having the values for department, status, age, and salary given in that row.

Department status age salary count

sales senior 31..35 46K..50K 30

sales junior 26..30 26K..30K 40

sales junior 31..35 31K..35K 40

systems junior 21..25 46K..50K 20

systems senior 31..35 66K..70K 5

systems junior 26..30 46K..50K 3

systems senior 41..45 66K..70K 3

marketing senior 36..40 46K..50K 10

marketing junior 31..35 41K..45K 4

secretary senior 46..50 36K..40K 4

secretary junior 26..30 26K..30K 6

Let status be the class-label attribute.

A. Implement a Decision Tree classification solution
B. Repeat the same problem using Naïve Bayesian classifier
C. Evaluate the performance of the two algorithms

   Repeat the above problem using ANN:

D. Design a multilayer feed-forward neural network for the given data. Label the nodes in the input and output layers.
E. Using the multilayer feed-forward neural network obtained in (a), show the weight values after one iteration of the backpropagation algorithm, given the training instance "(sales, senior, 31..35, 46K..50K)". Indicate your initial weight values and biases and the learning rate used.

Task2:

- The MNIST database of handwritten digits, available at http://yann.lecun.com/exdb/mnist/, has a training set of 60,000 examples, of which it includes a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

  Read the information provided on this site about the content of the dataset.

- Implement the K-nearest neighbor (and/or any other) classification algorithm to recognize handwritten digits.
- Repeat the classification using neural network implementation
- Repeat the classification using SVM
- Discuss the following:
  - Validation method used -
  - Selection of training samples
  - Accuracy of your results.

Introduction

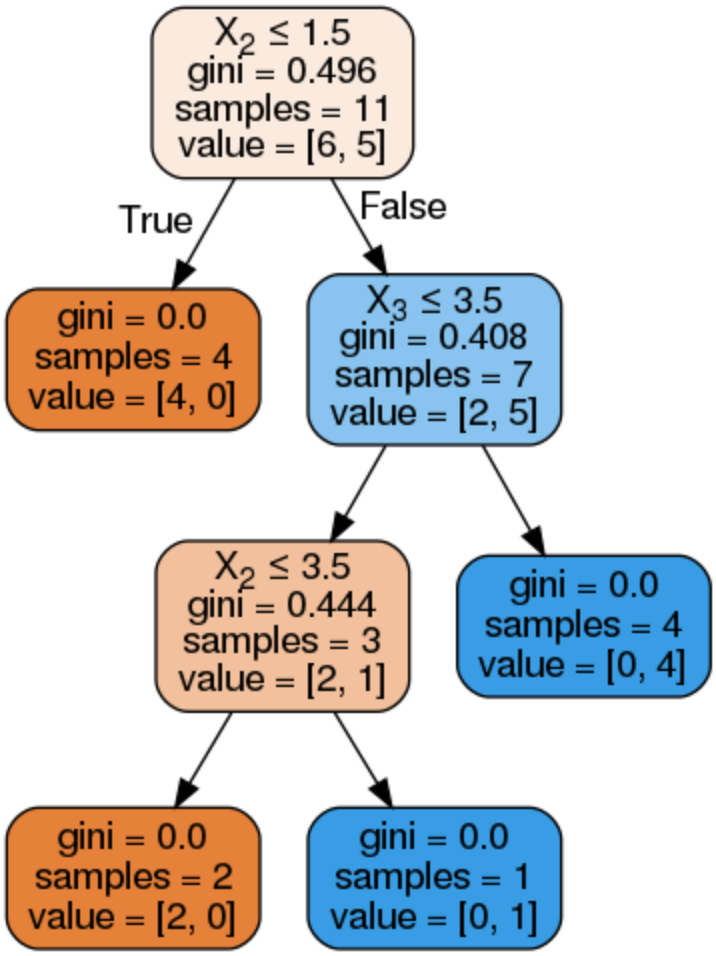
Classification is a supervised learning method in data mining. In supervised learning or classification the labels (or classes) are known in advance and the task is to classify the new data based on the model learned from training data.

It is a two step process in which case we first construct a model and then use the model for estimation. Accuracy of the model is the rate of correctly classified test samples in percentage. The test set is independent or unknown to the model otherwise it is called overfitting.

There are several classification algorithms for example decision tree, Naive Bayesian, Random forest, and neural network to name a few.

Task 1

A. In this task I implemented a decision tree solution for the given dataset. I also need to preprocess the data first and convert it into numerical form using Python libraries. I divided the dataset into target and input to the model. Then I tested by giving a value to the model and testing the result, with a score of 1.



B. In this task I implemented solution for Naive Bayesian classification. I preprocessed data to numeric form. I divided the data to 80% training and 20% testing and I got 0.66 score.

Task 2

In this task I implemented a convolutional neural network (CNN) using tensorflow python library. CNN is usually used for natural language processing and image processing tasks.

There are three layers in CNN given below.

- **Convolutional Layers:**

  This is the first layer in CNN and here we extract the features of the image

- **Pooling Layer:**

  We insert pooling layer after each convolutional layer to reduce the spatial size of the images.

- **Fully connected layer:**

  Here each node is connected to each other to determine the relationship of each parameter on the resulting label/class.

I have used the following steps / functions in the code to do this task:

1. **Load Data**

   Here we load data from mnist dataset into the dataframe.

2. **preprocess data**

   In the method we preprocess data, e.g to change it to 4 dimensional data

3. **normalize_data**

In this step we normalize data by changing the RGB code to 255 limit, and to apply float.

4. **create_model**

   Create the model and layers here.

5. **train_model**

   Here we train the model with x_train, y_train sets of data. The training data contains **60,000** of samples, while the testing data contains **10,000** samples.

6. **predict_image**

   We can predict/validate any image by providing image_index of x_train (with max 10 000)

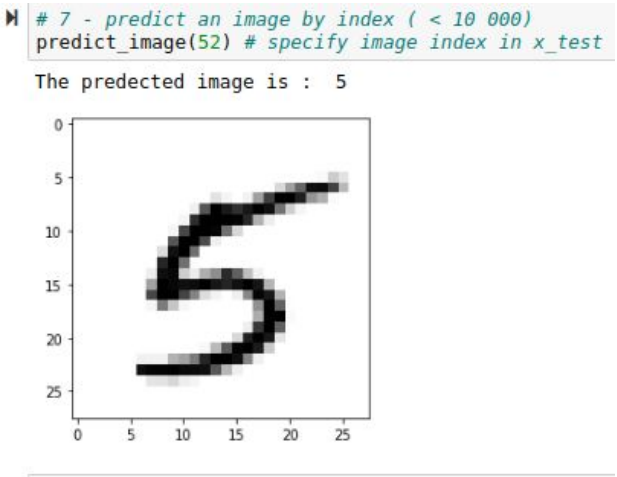


Figure : an image (5) at index is predicted as 5

Evaluation

We used the builtin method of model called evaluate and got about 99.38% success rate.