Implementing Naive Bayes

In this exercise you will implement a Gaussian Naive Bayes classifier to predict the behavior of vehicles on a highway. In the image below you can see the behaviors you'll be looking for on a 3 lane highway (with lanes of 4 meter width). The dots represent the d (y axis) and s (x axis) coordinates of vehicles as they either...

1. change lanes left (shown in blue)
2. keep lane (shown in black)
3. or change lanes right (shown in red)

Your job is to write a classifier that can predict which of these three maneuvers a vehicle is engaged in given a single coordinate (sampled from the trajectories shown below).

Each coordinate contains 4 pieces of information:

- s
- d
- $\dot{s}$
- $\dot{d}$

You also know the **lane width** is 4 meters (this might be helpful in engineering features for your algorithm).

Instructions

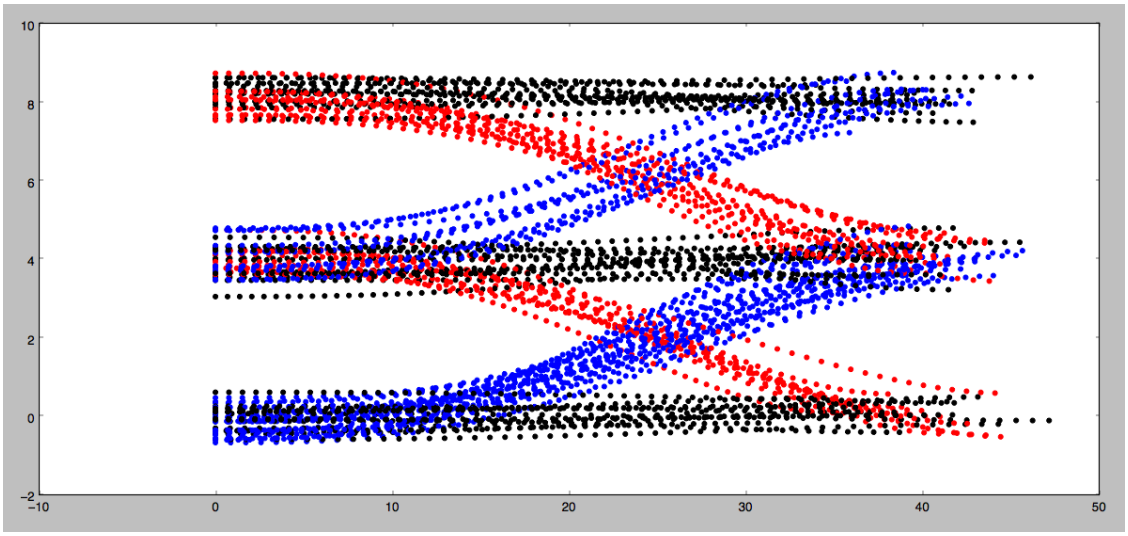
1. Implement the `train(self, data, labels)` and `predict(self, observation)` methods in the class `GNB` in `classifier.cpp`
2. When you want to test your classifier, run `Test Run` and check out the results.

NOTE: You are welcome to use some existing implementation of a Gaussian Naive Bayes classifier. But to get the **best** results you will still need to put some thought into what **features** you provide the algorithm when classifying. Though you will only be given the 4 coordinates listed above, you may find that by "engineering" features you may get better performance. For example: the raw value of the d coordinate may not be that useful. But `d % lane_width` might be helpful since it gives the *relative* position of a vehicle in it's lane regardless of which lane the vehicle is in.

Helpful Resources

- sklearn documentation on GaussianNB
- wikipedia article on Naive Bayes / GNB

written in python that you can optionally go through for extra coding practice. Also in the `python_solution` link the python solution is provided too, if you get stuck on the quiz see if you can convert the python solution to c++ and pass the classroom quiz with it. The last link `Nd013_Pred_Data` has all the training and testing data for this problem in case you want to run the problem offline.



main.cpp | classifier.cpp | classifier.h

```
#include "classifier.h"
#include <iostream>
#include <fstream>
#include <math.h>
#include <vector>

using namespace std;

vector<vector<double> > Load_State(string file_name)
{
    ifstream in_state_(file_name.c_str(), ifstream::in);
    vector< vector<double >> state_out;
    string line;

    while (getline(in_state_, line))
    {

        istringstream iss(line);
        vector<double> x_coord;
        double state1;
        double state2;
        double state3;
        double state4;
        iss >> state1;
        x_coord.push_back(state1);
        iss >> state2;
        x_coord.push_back(state2);
        iss >> state3;
        x_coord.push_back(state3);
```

RESET QUIZ TEST RUN SUBMIT ANSWER

Supporting Materials

- Nd013 Pred Data
- python_extra_practice
- python_solution

NEXT