

SEARCH



RESOURCES

CONCEPTS

- ✓ 1. Introduction and Overview
- ✓ 2. I/O Recap
- ✓ 3. Model-Based vs Data-Driven A...
- ✓ 4. Which is Best?
- ✓ 5. Data Driven Example - Trajecto...
- ✓ 6. Trajectory Clustering 2 - Online...
- ✓ 7. Thinking about Model Based A...
- ✓ 8. Frenet Coordinates
- ✓ 9. Process Models
- ✓ 10. More on Process Models
- ✓ 11. Multimodal Estimation
- ✓ 12. Summary of Data Driven and ...
- ✓ 13. Overview of Hybrid Approach...
- ✓ 14. Intro to Naive Bayes
- ✓ 15. Naive Bayes Quiz
- ✓ 16. Implement Naive Bayes C++
- 17. Implement Naive Bayes C++ (...)
- 18. Conclusion

Knowledge

Get learning questions answered

Student Hub

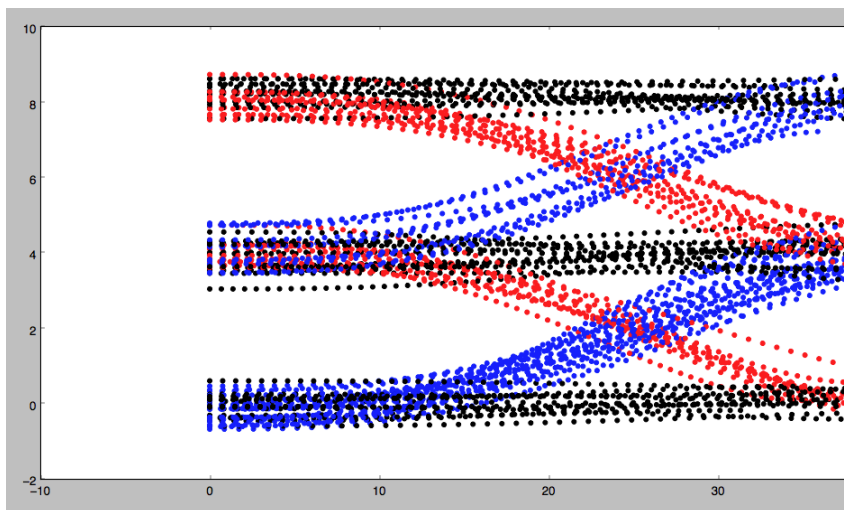
Chat with peers and mentors



Implementing Naive Bayes

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

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 car is performing given a single coordinate (sampled from the trajectories shown below).

Each coordinate contains 4 features:

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

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

Instructions

1. Implement the `train(data, labels)` method in the class `GNB` in `classifier.cpp`.

Training a Gaussian Naive Bayes classifier consists of computing and storing the mean and standard deviation from the data for each label/feature pair. For example, given the label "change lanes left" and the feature \dot{s} , it would be necessary to compute and store the mean and standard deviation of \dot{s} over all data points with the "change lanes left" label.

Additionally, it will be convenient in this step to compute and store the prior probability for each label C_k . This can be done by keeping track of the number of times each maneuver occurs in the training data.

2. Implement the `predict(observation)` method in `classifier.cpp`.