A sample video of the code running can be find at https://www.youtube.com/watch?v=fcmCkQOhk6k

Using David's and Aaron's code from the project walkthrough video, I managed to find a smooth path for the car.

As in the video I also looked for the car ahead in the same lane and adjusted speed, but I broke down the distances (from the car ahead to the self-driving car), and adjusted the speed by bigger increments as the distance was greater, but only while the velocity of the self-driving car was larger than the car ahead. This resulted in the car's speed adapting to the speed from the car ahead. (code lines 280-333)

Next the code looks at all cars in all lanes and evaluates if lane changes are possible. And I assign costs to each individual lane change. (334-440)

Behavior Planner (441-541);

My strategy is for the car to stay in lane 1, because then there are two lanes available if there is any slower car ahead. From lane 0 or 1, the car will always change lane to lane 1 whenever traffic allows.

If the car is in lane 1 and a slower car would be ahead, and both lane 0 and 1 are free, the car will change lane to the fastest lane (lane-speed determined by the slowest car in the lane up to 250 meters ahead). If only one lane available, it will change to that lane. Availability of the lane determined by the cost calculation mentioned above.

Sometimes when the car changed lane through a corner, the jerk would exceed the limits. To combat this, I Increased the distance between the points that I fed into spline.h to 40 m, and went with 80 points for my path.

I also declare a variable 't' (code line 201), which I use as a time indicator. It starts at 0 with each lane change, and allows the next lane change only after 30 increments (about the time it takes to change lanes). This to prevent the car from trying to changing lanes before a change is completed, resulting swerving in and out of lane.

Reflection;

I did not use a Gaussian Naive Bayes classifier to predict the behavior of the other cars, but only got away with it because the costs of changing lane are set steep, so the car will only change lanes as traffic is clear. Sometimes this would get the car stuck in traffic, although the gap in the traffic in the adjacent lane seems sufficient to allow a change. The classifier could give predictions of the behavior of the other car, and this could be incorporated in the behavior planner to allow for lane changes in denser traffic.

The car also sometimes would be better off to find a gap to change lanes by slowing speed and dropping back until the gap would be available for lane change. A Process Model could be introduced to program for such maneuvers.

I Increased the distance between the points that I fed into spline.h to 40 m, and went with 80 points for my path.

I also looked for the car ahead in the same lane and adjusted speed, but I broke down the distances (from the car ahead to the self-driving car), and adjusted the speed by bigger increments as the distance was greater, as long as the velocity of the self-driving car was larger than the car ahead. This resulted in the car adapting the speed from the car ahead.

Increased the amount op points from 50 to 80, with 40 meters spacing in between ‘anker’ point (to feed into spline.h.

I added t