1. Submission includes all required files and can be used to run the simulator in autonomous mode

My project includes the following files:

model.py containing the script to create and train the model

drive.py for driving the car in autonomous mode

model.h5 containing a trained convolution neural network

writeup_report.pdf summarizing the results

2. Submission includes functional code

Using the Udacity provided simulator and my drive.py file, the car can be driven autonomously around the track by executing

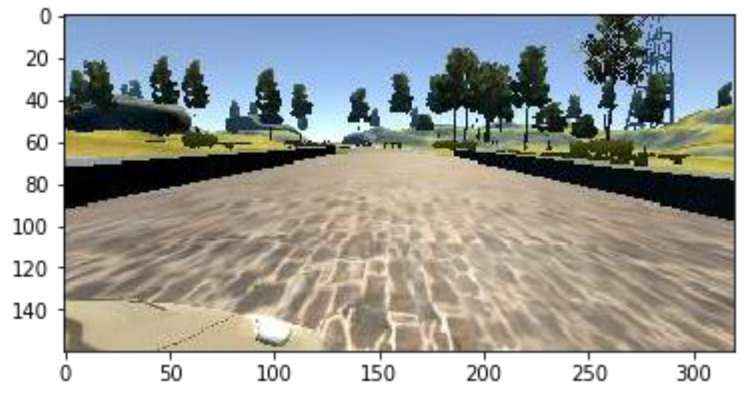
```
python drive.py model.h5
```

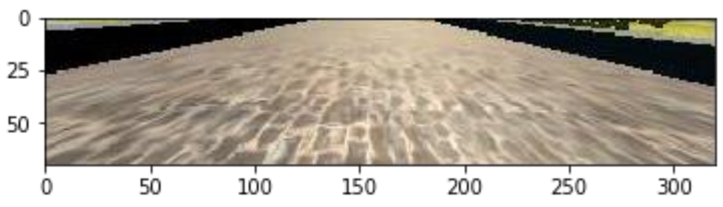
3. Submission code is usable and readable

The model.py file contains the code for training and saving the convolution neural network. The file shows the pipeline I used for training and validating the model, and it contains comments to explain how the code works.

Model Architecture and Training Strategy

My model consists of a convolution neural network. It has 3 convolution layer with filter size 8x8,5x5,5x5 and filter depth as 16,32,64 respectively. Then I added two dense layers with neurons 512 and 80 and final layer with one neuron as an output. The model includes RELU layers to introduce nonlinearity (code in line 54,56,60,63), and the data is normalized in the model using a Keras lambda layer (code line 52). The model also includes Cropping2D function at the input layer to crop the unnecessary sections of the image.





Initially I have not applied any dropout layer due to which model was overfitting and car in simulator was going out of track. Then I added dropout layer at line 59,62,65. I tried with different dropout rates and got right combination of dropout parameters. 80 % of data is used as training set and 20 % of data is used as validation set to further avoid overfitting.

The model used an adam optimizer, so the learning rate was not tuned manually.

I used training data provided by Udacity and it was sufficient to properly train model. For each frame there were three images from center, left and right camera. For each frame I decided to take randomly take any one the image from set of center, left and right images as we should take advantage of all the cameras. When I choose left image I added 0.2 as a correction to the steering angle and subtracted 0.2 when I choose right image. No change in steering angle was done when I choose center image.

My first step was to use a convolution neural network model similar to the Nvidia architecture. I thought this model might be appropriate because it is one of the famous and appropriate architecture for self-driving car. But car was not able to turn properly and it was going out of track. So I decided to add one more dense layer of 80 neurons. After adding this layer, I tried to run on simulator. The car was properly taking turns. I trained model on 20 epochs. I shuffled data for every epochs(shuffle=True in model.fit() function).

The summarized model is given below:

```
____________________________________________________________________________________________________
Layer (type)                     Output Shape          Param #     Connected to
====================================================================================================
cropping2d_1 (Cropping2D)        (None, 65, 320, 3)    0           cropping2d_input_1[0][0]
____________________________________________________________________________________________________
lambda_1 (Lambda)                (None, 65, 320, 3)    0           cropping2d_1[0][0]
____________________________________________________________________________________________________
convolution2d_1 (Convolution2D)  (None, 17, 80, 16)    3088        lambda_1[0][0]
____________________________________________________________________________________________________
activation_1 (Activation)        (None, 17, 80, 16)    0           convolution2d_1[0][0]
____________________________________________________________________________________________________
convolution2d_2 (Convolution2D)  (None, 9, 40, 32)     12832       activation_1[0][0]
```

```
____________________________________________________________________________________
______________
activation_2 (Activation)        (None, 9, 40, 32)     0           convolution2d_2[0][
0]
____________________________________________________________________________________
______________
convolution2d_3 (Convolution2D)  (None, 5, 20, 64)     51264       activation_2[0][0]
____________________________________________________________________________________
______________
flatten_1 (Flatten)              (None, 6400)          0           convolution2d_3[0][
0]
____________________________________________________________________________________
______________
dropout_1 (Dropout)              (None, 6400)          0           flatten_1[0][0]
____________________________________________________________________________________
______________
activation_3 (Activation)        (None, 6400)          0           dropout_1[0][0]
____________________________________________________________________________________
______________
dense_1 (Dense)                  (None, 512)           3277312     activation_3[0][0]
____________________________________________________________________________________
______________
dropout_2 (Dropout)              (None, 512)           0           dense_1[0][0]
____________________________________________________________________________________
______________
activation_4 (Activation)        (None, 512)           0           dropout_2[0][0]
____________________________________________________________________________________
______________
dense_2 (Dense)                  (None, 80)            41040       activation_4[0][0]
____________________________________________________________________________________
______________
dropout_3 (Dropout)              (None, 80)            0           dense_2[0][0]
____________________________________________________________________________________
______________
activation_5 (Activation)        (None, 80)            0           dropout_3[0][0]
____________________________________________________________________________________
______________
dense_3 (Dense)                  (None, 1)             81          activation_5[0][0]
====================================================================================
==============
```