The goals/steps of this project are the following:

- Use the simulator to collect data of good driving behavior.
- Build, a convolution neural network in Keras that predicts steering angles from images.
- Train and validate the model with a training and validation set.
- Test that the model successfully drives around track one without leaving the road.
- Summarize the results with a written report.

Rubric points

Here I will consider the rubric points individually and describe how I addressed each point in my implementation.

Files Submitted & Code Quality

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 in the simulator
- **model.h5** : Containing a trained convolution neural network.
- **writeup_report.md** : Summarizing the results

Note:

On my first iteration, I tried LeNet model and nVidia Autonomous Car Group model.

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

1. An appropriate model architecture has been employed

My initial approach was to use LeNet, but it was hard to have the car inside the street with three epochs After this, I decided to try the nVidia Autonomous Car Group model, and the car drove the complete first track after just five training epochs

A model summary is as follows:

Layer (type) Output Shape Param # Connected to

==
==

lambda_1 (Lambda) (None, 160, 320, 3) 0 lambda_input_1[0][0]

---

cropping2d_1 (Cropping2D) (None, 65, 320, 3) 0 lambda_1[0][0]

---

convolution2d_1 (Convolution2D) (None, 31, 158, 24) 1824 cropping2d_1[0][0]

---

convolution2d_2 (Convolution2D) (None, 14, 77, 36) 21636
convolution2d_1[0][0]

---

convolution2d_3 (Convolution2D) (None, 5, 37, 48) 43248
convolution2d_2[0][0]

---

convolution2d_4 (Convolution2D) (None, 3, 35, 64) 27712
convolution2d_3[0][0]

---

convolution2d_5 (Convolution2D) (None, 1, 33, 64)   36928
convolution2d_4[0][0]

---

flatten_1 (Flatten)   (None, 2112)   0   convolution2d_5[0][0]

---

dense_1 (Dense)   (None, 100)   211300   flatten_1[0][0]

---

dense_2 (Dense)   (None, 50)   5050   dense_1[0][0]

---

dense_3 (Dense)   (None, 1)   51   dense_2[0][0]

==

Total params: 347,749

Trainable params: 347,749

Non-trainable params: 0

---

2. Attempts to reduce overfitting in the model

I decided not to modify the model by applying regularization techniques like Dropout or Max pooling. Instead, I decided to keep the training epochs low: only 5 epochs. In addition to that, I split my sample data into training and validation data. Using 80% as training and 20% as validation. I also shuffled the data.

3. Model parameter tuning

The model used an Adam optimizer, so the learning rate is tuned automatically

4. Appropriate training data

Training data is collected using training mode. Steep turns data is also recorded so make turns more smooth.

Model Architecture and Training Strategy

1. Solution Design Approach

I first tried LeNet model with 5 epochs and data provided from UdaCity. The car was taking 25 turn and then it went straight to lake.

To normalize the data I used lamda layer but car was not able to take turn after the bridge.

Adding Cropping layer was having no effect on the car.

Then for second I Shifted to nVidia Autonomous Car Group. The model worked Fine.

2. Final Model Architecture

The final model architecture is nVidia Autonomous Car Group as shown in the following image:

```
Layer (type)                     Output Shape          Param #     Connected to
================================================================================================
lambda_1 (Lambda)                (None, 160, 320, 3)   0           lambda_input_1[0][0]
____________________________________________________________________________________________________
cropping2d_1 (Cropping2D)        (None, 65, 320, 3)    0           lambda_1[0][0]
____________________________________________________________________________________________________
convolution2d_1 (Convolution2D)  (None, 31, 158, 24)   1824        cropping2d_1[0][0]
```

```
____________________________________________________________________________________________________
convolution2d_2 (Convolution2D)  (None, 14, 77, 36)    21636
convolution2d_1[0][0]
____________________________________________________________________________________________________
convolution2d_3 (Convolution2D)  (None, 5, 37, 48)     43248
convolution2d_2[0][0]
____________________________________________________________________________________________________
convolution2d_4 (Convolution2D)  (None, 3, 35, 64)     27712
convolution2d_3[0][0]
____________________________________________________________________________________________________
convolution2d_5 (Convolution2D)  (None, 1, 33, 64)     36928
convolution2d_4[0][0]
____________________________________________________________________________________________________
flatten_1 (Flatten)              (None, 2112)          0           convolution2d_5[0][0]
____________________________________________________________________________________________________
dense_1 (Dense)                  (None, 100)           211300      flatten_1[0][0]
____________________________________________________________________________________________________
dense_2 (Dense)                  (None, 50)            5050        dense_1[0][0]
____________________________________________________________________________________________________
dense_3 (Dense)                  (None, 1)             51          dense_2[0][0]
====================================================================================================
```

2. Creation of the Training Set & Training Process

To capture good driving behaviour, I first recorded two laps on track one using center lane driving. Here is an example image of center lane driving:

I collected some turn data explicitly to make turn smoother