Traffic Sign Recognition**

New Writeup Template

Build a Traffic Sign Recognition Project

The goals / steps of this project are the following:

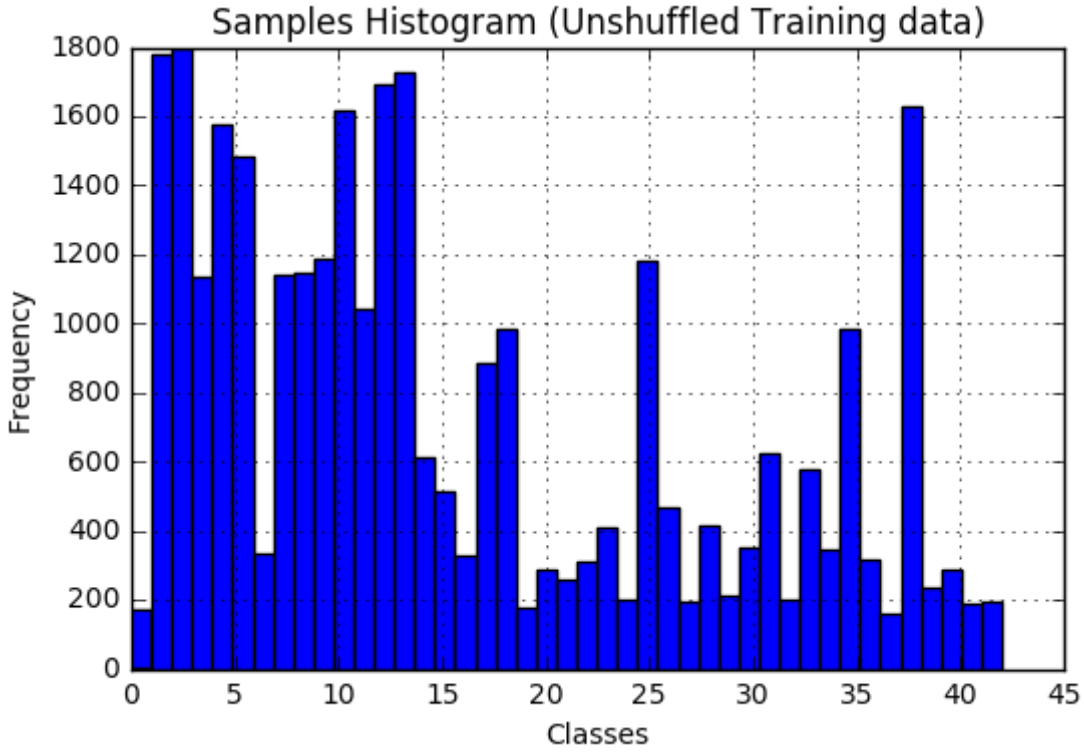
- Load the data set (see below for links to the project data set) - Done
- Explore, summarize and visualize the data set - Done
- Design, train and test a model architecture - Done
- Use the model to make predictions on new images - Done
- Analyze the softmax probabilities of the new images - Done
- Summarize the results with a written report - Done

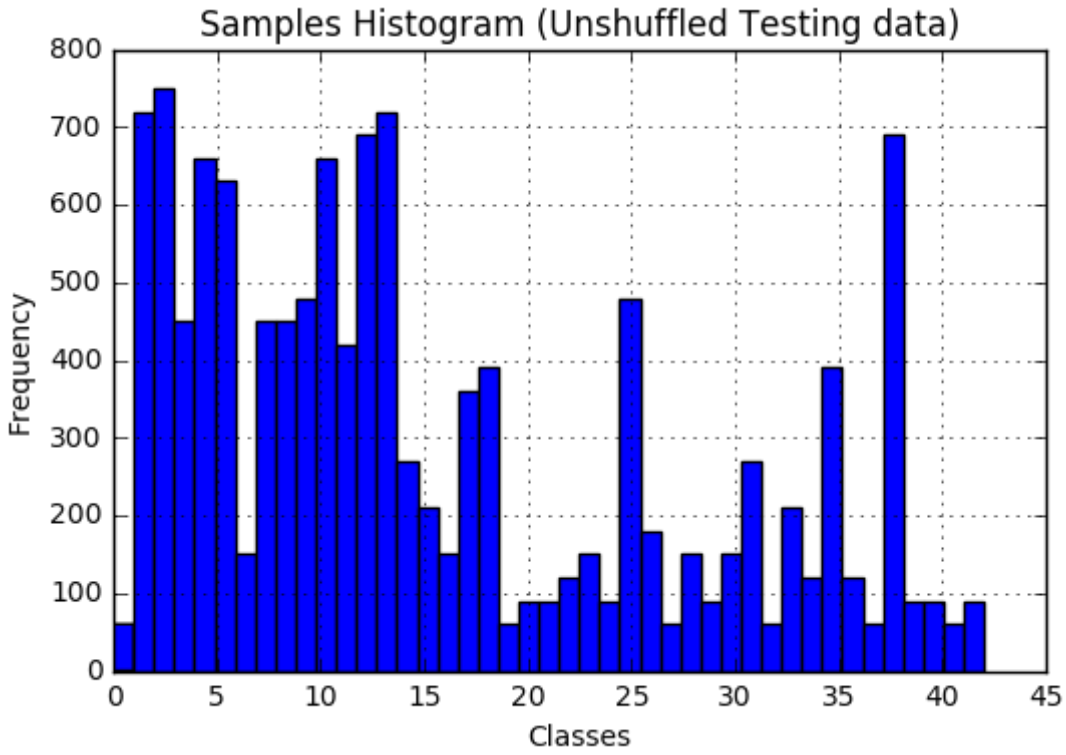
Data Set Summary & Exploration

I used the pandas library to calculate summary statistics of the traffic signs data set:

- The size of training set is 31367 samples
- The size of test set is 7842 samples for the validation set and another 12630 samples for Test Set.
- The shape of a traffic sign image is 32 x 32 x 3
- The number of unique classes/labels in the data set is 43

Exploratory visualization





Design and Test a Model Architecture

Summary:

The famous LENET architecture model with 2 Convolution layers and 3 Fully connected layers as provided from the online notes. The model has employed the following setup:

- "relu" activation,
- "dropouts" to avoid overfitting Two "pooling" layers after each convoltion layers.

The fully connected layer has 43 units for logits(network) to detect 43 different traffic signs.

More Details on Design Decisions

Layer 1: Convolutional. Input = 32x32x3. Output = 28x28x6. strides=[1, 1, 1, 1], padding='VALID' Activation - relu Avoid overfitting- dropout Pooling - maxpool

Layer 2: Convolutional. Output = 10x10x16. strides=[1, 1, 1, 1], padding='VALID' Activation - relu Avoid overfitting- dropout Pooling - maxpool Flatten. Input = 5x5x16. Output = 400.

Layer 3: Fully Connected. Input = 400. Output = 120. Activation - relu Avoid overfitting- dropout

Layer 4: Fully Connected. Input = 120. Output = 84. Activation - relu

Layer 5: Fully Connected. Input = 84. Output = 43

3. Describe, and identify where in your code, what your final model architecture looks like including model type, layers, layer sizes, connectivity, etc.) Consider including a diagram and/or table describing the final model.

The code for my final model is located in the seventh cell of the ipython notebook.

My final model consisted of the following layers:

Layer	Description
Input	32x32x3 RGB image
Convolution 3x3	1x1 stride, same padding, outputs 32x32x64
RELU	
Max pooling	2x2 stride, outputs 16x16x64
Convolution 3x3	Input = 400. Output = 120. Activation - relu
Fully connected	Input = 120. Output = 84. Activation - relu
Fully connected	Input = 84. Output = 43

Describe how, and identify where in your code, you trained your model. The discussion can include the type of optimizer, the batch size, number of epochs and any hyperparameters such as learning rate.

The code for training the model is located in the eigth cell of the ipython notebook.

Test of the Model on New Images

1. Choose five German traffic signs found on the web and provide them in the report. For each image, discuss what quality or qualities might be difficult to classify.

Here are five German traffic signs that I found on the web:

![alt text][image4] ![alt text][image5] ![alt text][image6] ![alt text][image7] ![alt text][image8]

The first image might be difficult to classify because ...

2. Model's predictions

The code for making predictions on my final model is located in the tenth cell of the Ipython notebook.

Here are the results of the prediction:

Image	Prediction
Do Not Enter Sign	Stop sign
Intersection Sign	U-turn
Do not Turn Sign	Bumpy Road
70 km/h	Yield
50 km/h	U-turn

The model was able to correctly guess 4 of the 5 traffic signs, which gives an accuracy of 80%. This compares favorably to the accuracy on the test set of ...

3. Describe how certain the model is when predicting on each of the five new images by looking at the softmax probabilities for each prediction and identify where in your code softmax probabilities were outputted. Provide the top 5 softmax probabilities for each image along with the sign type of each probability. (OPTIONAL: as described in the "Stand Out Suggestions" part of the rubric, visualizations can also be provided such as bar charts)

The code for making predictions on my final model is located in the 11th cell of the Ipython notebook.

For the first image, the model is relatively sure that this is a stop sign (probability of 0.6), and the image does contain a stop sign. The top five soft max probabilities were

Probability	Prediction
.60	Stop sign
.20	U-turn
.05	Yield
.04	Bumpy Road
.01	Slippery Road

References and Appendix:

1. CS231 Cornell University Library, Convolutional Neural Network for Image Recognition http://cs231n.github.io/ (http://cs231n.github.io/)
2. Caffe Link (http://caffe.berkeleyvision.org/) Especially lessons learned from current tested models for exmpale the work from Karpathy (http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/)

Section: Test a Model on New Images

1. Choose five German traffic signs found on the web and provide them in the report. For each image, discuss what quality or qualities might be difficult to classify.

Here are five of the ten German traffic signs that I found on the web:


70


50