## Data Encoding:

I have used one-hot encoding for the input data.

For suit column there are 4 possible values hence 4 bits are used and the value of suit set that bit 1

For rank column total 13 values are possible,hence 13 bits are used and the value of rank set that bit 1

Total there are 5 set of (set,rank) and for one set 17 bits are reuired so for 5 sets 85 bits are required.That’s the input which iam fedding to my network. (85 input neurons)

## Network Structure :

I have tested my model with atmost 4 hidden layers.For 2nd and 3rd hidden layers i tested with different range of lerning rate and lambda.

With 4th layer i only tested with different lambda but same learning rate.

The best submission is submitted with using only 2 hidden layers as i achieved my best accuracy with this.I achieved atmost same accuracies with 4 and 3 hidden layers

## Table:

| #Hidden Layer | Lambda | Learning Rate | Accuracy |
|---|---|---|---|
| **2** | 1.00E0 | 1.00E-01 | 0.99998 |
| **2** | 1.00E0 | 1.00E-02 | 0.70050 |
| **2** | 1.00E0 | 1.00E-03 | 0.58771 |
| **2** | 1.00E0 | 1.00E-04 | 0.500637 |
| **2** | 1.00E0 | 1.00E-05 | 0.475564 |
| **2** | 1.00E-01 | 1.00E-01 | 0.9977 |
| **2** | 1.00E-01 | 1.00E-02 | 0.7045 |
| **2** | 1.00E-01 | 1.00E-03 | 0.60522 |
| **2** | 1.00E-01 | 1.00E-04 | 0.500723 |
| **2** | 1.00E01 | 1.00E-05 | 0.479431 |
| **2** | 1.00E1 | 1.00E-01 | 0.7855 |
| **2** | 1.00E1 | 1.00E-02 | 0.6955 |
| **2** | 1.00E1 | 1.00E-03 | 0.509556 |
| **2** | 1.00E1 | 1.00E-04 | 0.50074 |
| **2** | 1.00E1 | 1.00E-05 | 0.42245 |
| **3** | 1.00E0 | 1.00E-01 | 0.9858 |
| **3** | 1.00E0 | 1.00E-02 | 0.5865 |

|  |  |  |  |
|---|---|---|---|
| **3** | 1.00E0 | 1.00E-03 | 0.52001 |
| **3** | 1.00E0 | 1.00E-04 | 0.500637 |
| **3** | 1.00E0 | 1.00E-05 | 0.475564 |
| **3** | 1.00E-01 | 1.00E-01 | 0.9568 |
| **3** | 1.00E-01 | 1.00E-02 | 0.7045 |
| **3** | 1.00E-01 | 1.00E-03 | 0.60522 |
| **3** | 1.00E-01 | 1.00E-04 | 0.500723 |
| **3** | 1.00E01 | 1.00E-05 | 0.479431 |
| **3** | 1.00E1 | 1.00E-01 | 0.72586 |
| **3** | 1.00E1 | 1.00E-02 | 0.7425 |
| **3** | 1.00E1 | 1.00E-03 | 0.68952 |
| **3** | 1.00E1 | 1.00E-04 | 0.50074 |
| **3** | 1.00E1 | 1.00E-05 | 0.497166 |
| **4** | 1.00E0 | 1.00E-1 | 0.9998 |
| **4** | 1.00E-1 | 1.00E-1 | 0.9856 |
| **4** | 1.00E1 | 1.00E-1 | 0.9525 |

## Update Type:

I used stochastic gradient descent because while experimenting i found that stochastic was giving godd result as compared to mini batch.The smallest mini batch i took was of size 10 which was giving quite good accuracy but final submission i did using stochastic gradient descent.It works little slow as compared to mini batch .No. of epoches used 50 for full data and accuracy achieved was 0.99998 with lerning rate of 0.1 and regularization parameter of 1.

K-fold Cross validation is also implemented in which 4 sets of training data is used and one set of testing data.

## Activation Function:

I tested with 2 activation functions  sigmoid and tanh. Tanh was not working good for me (accuracy was low ),hence i used sigmoid in my network

But with tanh convergence rate was faster as compared to sigmoid.

## Observations and Results:

After performing all the tests i observed,increasing hidden layers increases the efficency but with cost of more computation.

Changing regularization parameter does not change the accuracy too much as compared with change in learning rate

By decreasing learning rate the accuracy is decreased

0.1 is the best learning rate for my model

Code is commented and best submission weight,baises pickle file is submitted with best prediction file named as output.csv

## Acknowledgment:

http://neuralnetworksanddeeplearning.com/

http://stackoverflow.com/

http://www.wildml.com/