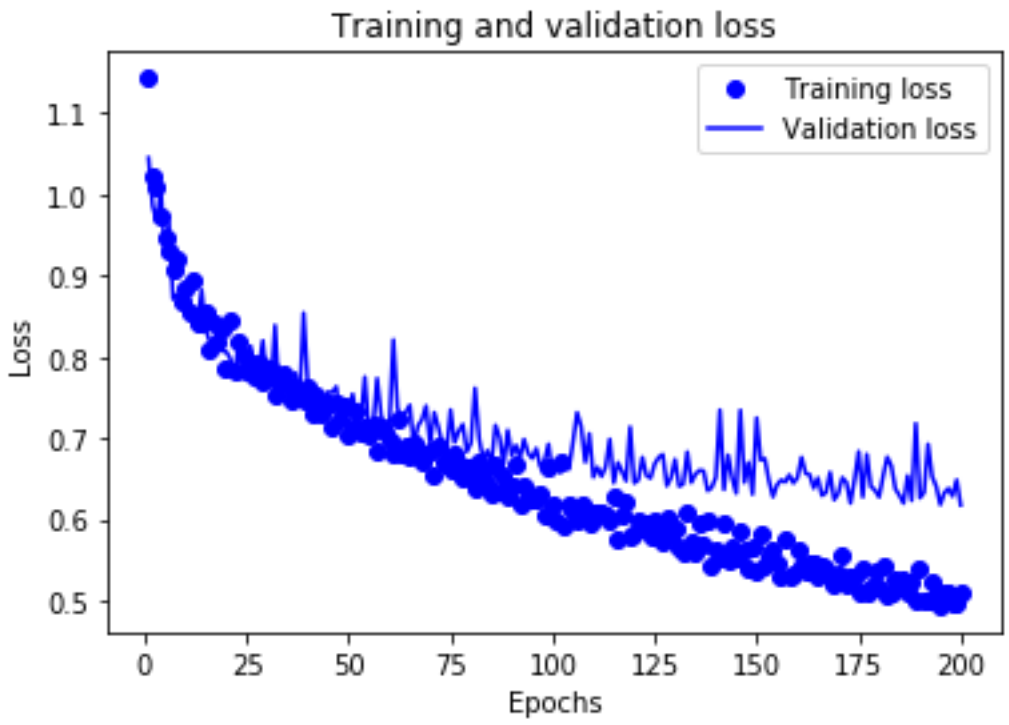
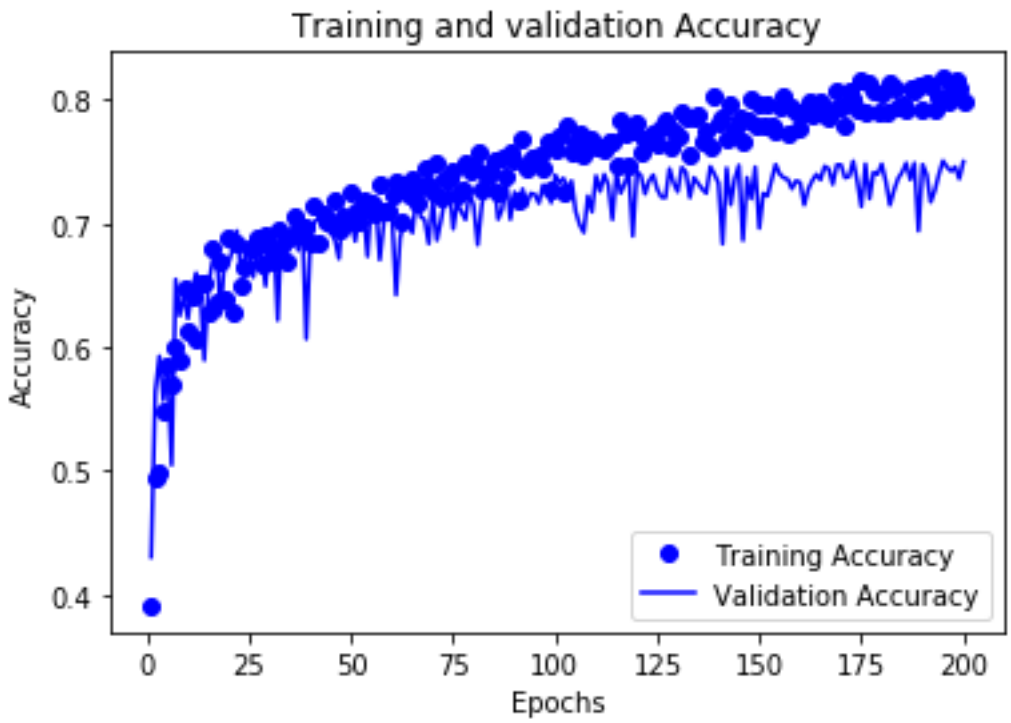
Task 1

1. Problem Statement:
    - Classify images (3 classes) from CIFAR-10 dataset.
2. Proposed method and implementation details:
    - I used deep neural network for classification.
    - Divided data into train and test (train – 80%,test-20%)
    - Reshape Train data - 50000,32*32*3 to 50000,3072
    - Reshape Test data - 10000,32*32*3 to 10000,3072
    - Rescaling – dividing pixel value by 255
    - Encoding the class labels
      Hyperparameter tuning:
    - I tried different architecture (number of hidden layers, nodes in each layer, activation function) – Following is the architecture which gave best performance on validation set
    - Its architecture is as follows :
      Input nodes : 3072
      Hidden layer 1,no of nodes : 64,activation function:”relu”
      Hidden layer 2,no of nodes : 32,activation function:”relu”
      Output nodes : 3,activation function:”softmax”
      Optimizer : RMSprop (learning rate : 0.0001)
      Loss : categorical crossentropy
      metrics='accuracy'
      epochs = 300
    - Divide train data into train and validation data and train the model on train data and choose hyperparameter which performs well on validation set.
3. Results:



As you can see training and validation loss decreases as we increase the epoch. After 128 epochs validation loss doesn't decrease significantly so we will use that value as the epoch value.



As you can see training and validation accuracy increases as we increase number of epochs. After 128 epochs validation accuracy doesn't increase significantly so we will use that value as the epoch value.

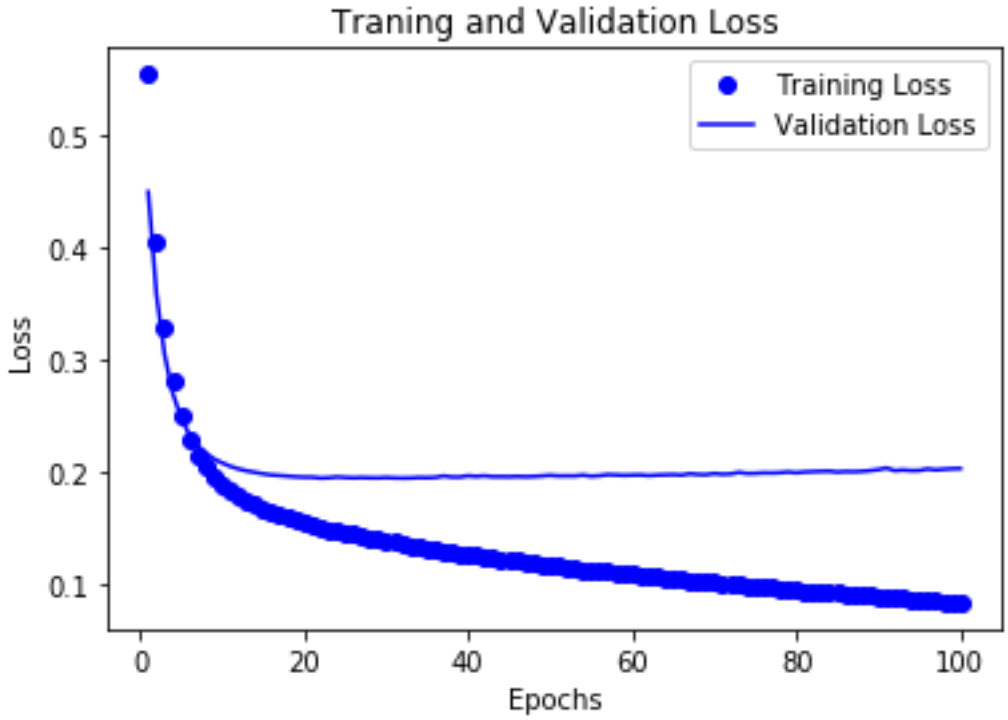
Now we will train with these hyperparamters on whole train data (train set + validation set) and test its performance on test set which we kept aside earlier.

- Test accuracy: 0.7929999828338623
- Test Loss: 0.5269214574495952
- Train accuracy: 0.8278
- Train Loss: 0.4391

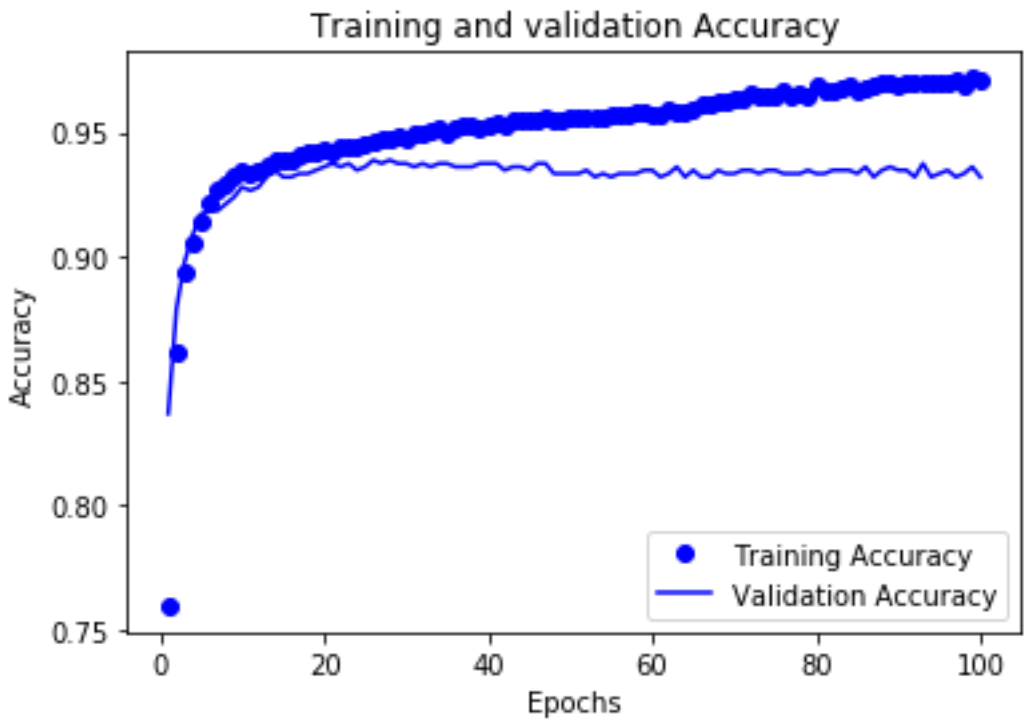
Task 2

1. Problem Statement:
    - Classify whether an email is spam or not from uci repository.
2. Proposed method and implementation details:
    - I used deep neural network for classification.
        I. Preprocessing
            - Normalize train and test data – using mean and standard deviation of train data, such that our model will converge faster.
            - Divide train data into train and validation data and train the model on train data and choose hyperparameter which performs well on validation set.
        II. Hyperparameter tuning: I tried different architecture (number of hidden layers, nodes in each layer, activation function) – Following is the architecture which gave best performance on validation set.
            - Its architecture is as follows :
                A. Input nodes : 57
                B. Hidden layer 1,no of nodes :32,activation function:”relu”
                C. Outputnodes:1, ,activation function: ”sigmoid”
                D. Optimizer : RMSprop(learning rate : 0.001)
                E. Loss : binary crossentropy
                F. metrics= accuracy
                G. epochs = 100

3. Results:



As you can see training and validation loss decreases as we increase the epoch. After 26 epochs validation loss doesn't decrease significantly so we will use that value as the final epoch value.



As you can see training and validation accuracy increases as we increase number of epochs. After 26 epochs validation accuracy doesn't increase significantly so we will use that value as the epoch value.

Now we will train with these hyperparamters on whole train data (train set + validation set) and test its performance on test set which we kept aside earlier.

Performance:

- Test Accuracy: 0.931596
- Test Loss: 0.17023
- Train Accuracy : 0.9511
- Train Loss : 0.1412

Task 3

4. Problem Statement:
    - Predict total number of violent crimes per 100K population(regression).
5. Proposed method and implementation details:
    - I used deep neural network for classification.
        III. Preprocessing
            - Divided data into train and test (train – 80%,test-20%)
            - Remove feature which had more than 80% of missing values.
            - Replace missing value of a column with mean values of its feature
            - Normalize train and test data – using mean and standard deviation of train data, such that our model will converge faster.
            - Divide train data into train and validation data and train the model on train data and choose hyperparameter which performs well on validation set.

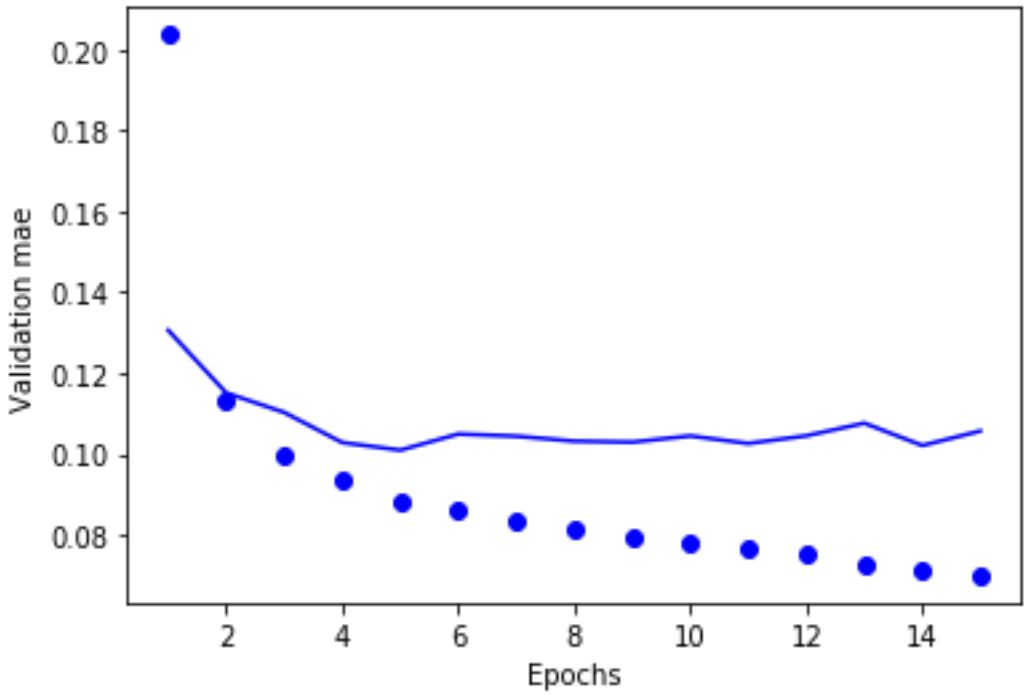
        IV. Hyperparameter tuning: I tried different architecture (number of hidden layers, nodes in each layer, activation function) – Following is the architecture which gave best performance on validation set.
            - Its architecture is as follows :
                H. Input nodes : 104
                I. Hidden layer 1,no of nodes :64,activation function:”relu”
                J. Outputnodes:1
                K. Optimizer : RMSprop

L. Loss : mse

M.metrics= mean absolute error

N. epochs = 15

6. Results:



As you can see training and validation mean absolute error decreases as we increase number of epochs. After 5th epoch validation mae doesn't decrease significantly so we will use that value as the final epoch value.

Now we will train with these hyperparamters on whole train data (train set + validation set) and test its performance on test set which we kept aside earlier.

Performance:

- Train Loss: 0.0177
- Train mae: 0.0888
- Test loss: 0.01876
- Test mae: 0.09377