MACHINE LEARNING MINI – PROJECT

Supervised Learning

FLOWER SPECIES CLASSIFICATION

Shivam Sahu - 1901188
Indian Institute of Information Technology Guwahati

Outline

- Introduction
  - Aim & Problem Definition
  - Application
  - More About Dataset
- Literature Survey
- Feature Extraction
- Applying ML Model (Discussed in class) & Result Analysis
  - Logistic Regression(Stochastic & batch mode)->Using inbuilt function sklearn
  - Perceptron Learning Algorithm -> Using inbuilt function sklearn
  - Single layer perceptron [SLP] -> using my own code
  - Multi layer perceptron [MLP] -> using my own code
  - Hyper parameter tuning[For all Algo]
  - Checking Overfitting [For all Algo]
  - K-Fold Cross validation[For all Algo]
- All algo result brief Comparison

Problem Definition & Aim

- Multiclass classifier, based on the Flowers Dataset. The data was obtained from the University of Oxford's Department of Engineering Science.
- We will use the FLOWER dataset. 8 classes of flower species, each having 80 images. So, totally we have 640 images to train, test and validation our model.
- Aim : This project aims to evaluate the use of machine learning techniques to classify eight species of flowers
- Importance In ML : The main purpose of flower recognition is to make judgments of flower category though some flower attributes, such as colour, texture and shape, which plays an important role in the fields of forestry informatization and plant medicine .

Application

Flower classification has various applications

- **Search engine which queries images based on text or keywords**
- **It can be helpful in flower searching for patent analysis and in floriculture.**
- **The floriculture industry consist of flower trade means selling and buying flowers**
- **Ayurveda treatment**
- **Seed production, potted plants and extraction of essential oil from flowers.**

Research Paper - 1

Research Paper	Year of publication	Methodology	Performance Measure
Bilinear pyramid network for flower species categorization	**2020**	Convolutional neural network (CNN)	-**Accuracy**

Conclusion : **By** Cheng Pang · Wenhao Wang · Rushi Lan · Zhuo Shi · Xiaonan Luo & Paper Link

- They proposed Bilinear Pyramid Network (BPN) for flower categorization. The proposed BPN is built on the VGG-16 architecture. When passing an image through a CNN, feature maps keep decreasing in their sizes, forming a feature pyramid.
- They have used 2 CNN's of the bilinear model. One for feature detection and another for feature extraction.
- The training process follows the standard stochastic gradient descent (SGD) with a learning rate of 0.01, and will be terminated when reach the maximum iteration of 100000.
- They are able to achieve a top-1 precision of **99.1%** using 3 convolutional layers in the feature pyramid are used.

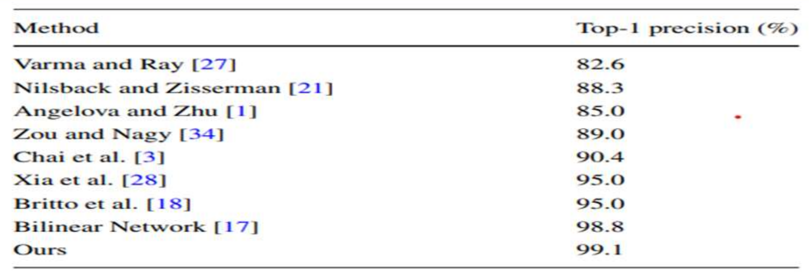


Table 4 Comparison of 17 category flower categorization

Method	Top-1 precision (%)
Varma and Ray [27]	82.6
Nilsback and Zisserman [21]	88.3
Angelova and Zhu [1]	85.0
Zou and Nagy [34]	89.0
Chai et al. [3]	90.4
Xia et al. [28]	95.0
Britto et al. [18]	95.0
Bilinear Network [17]	98.8
Ours	99.1

Research Paper - 2

Research Paper	Year of publication	Methodology	Performance Measure
Classification of flower image based on attention mechanism & multi-loss attention network	**2021**	multi-loss spatial attention network, multi-loss channel attention network & multi-loss multiattention network	-**Accuracy** **98%**

Conclusion : By Mei Zhang , Huihui Su, Jinghua Wen& Link

- This paper proposes the multi-loss spatial attention network (MLSAN), the multi-loss channel attention network (MLCA) and the multi-loss multiattention network (MLMAN) three network models, with Xception as the basic network, channel attention and spatial attention are added to Xception to improve Xception's ability to locate and extract features of flower image regions.
- They are able to achieve a maximum accuracy of 98.03% Using MLCSCAN method.

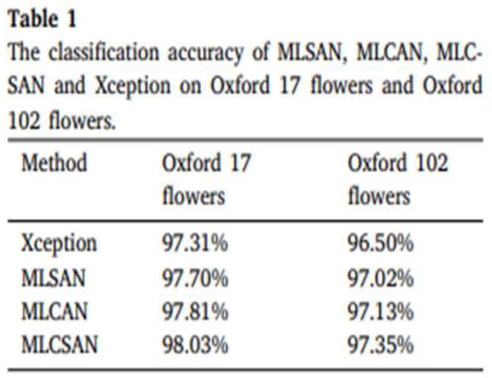


Table 1
The classification accuracy of MLSAN, MLCAN, MLCSAN and Xception on Oxford 17 flowers and Oxford 102 flowers.

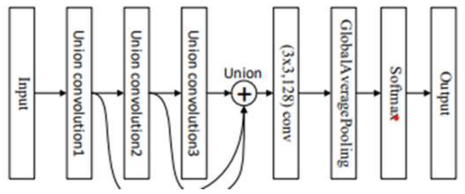
Method	Oxford 17 flowers	Oxford 102 flowers
Xception	97.31%	96.50%
MLSAN	97.70%	97.02%
MLCAN	97.81%	97.13%
MLCSAN	98.03%	97.35%

Research Paper - 3

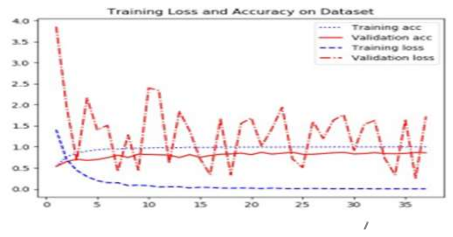
Research Paper	Year of publication	Methodology	Performance Measure
Union-net: A deep neural network model adapted to small data sets	**2020**	convolutional neural network (CNN)	-**Accuracy** 87%

Conclusion: By Qingfang He1 , Guang Cheng1 , Zhiying Lin1 & paper Link

- it is based on the convolutional neural network (CNN). Convolutional network units with different combinations of the same input form a Union module. Each Union module is a convolutional layer
- A "3-layer" neural network is formed through the serial input and output between the three modules. The outputs of the three Union modules are fused and added as the input of the last convolutional layer, thus forming a complex network with a 4-layer network structure



- Union-net uses a maximum pooling layer in the Union1 module. Max pooling can extract effective features, reduce dimensions and parameters and remove noise.
- After the initial input data of the model is processed by Union1's internal convolution structure, the output features are pooled and filtered to provide more effective data for subsequent processing.
- They are able to achieve an accuracy of **87%** with 10-fold cross-validation.



More About Dataset

- The oxford flowers dataset consists of 17 different flower species namely :

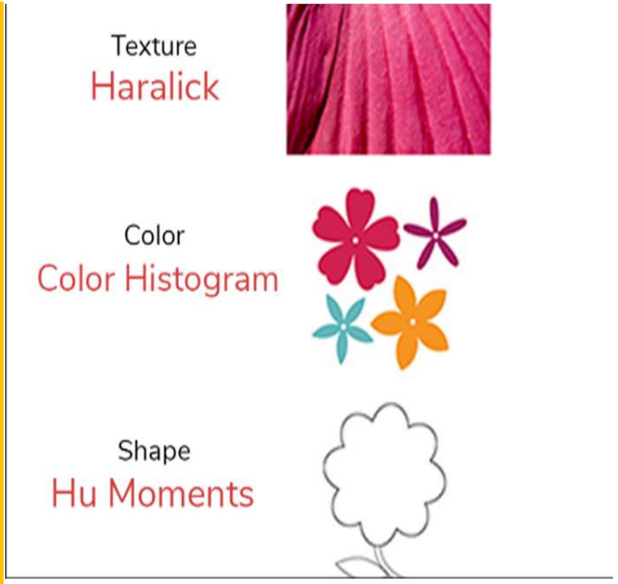
1. Buttercup	6. Daisy	11. Windflower	16. Tulip
2. Sunflower	7. Dandelion	12. LilyValley	17. Cowslip
3. Snowdrop	8. Fritillary	13. Bluebell	
4. Coltsfoot	9. Iris	14. Crocus	
5. Daffodil	10. Pansy	15. Tigerlily	

But here(According to Ma’am), I will experiment by considering 8 classes as shown below:

1. Daisy	2. Sunflower	3. Dandelion	4. Snowdrop
5. Fritillary	6. Daffodil	7. Tigerlily	8. Coltsfoot

Feature Extraction From Images

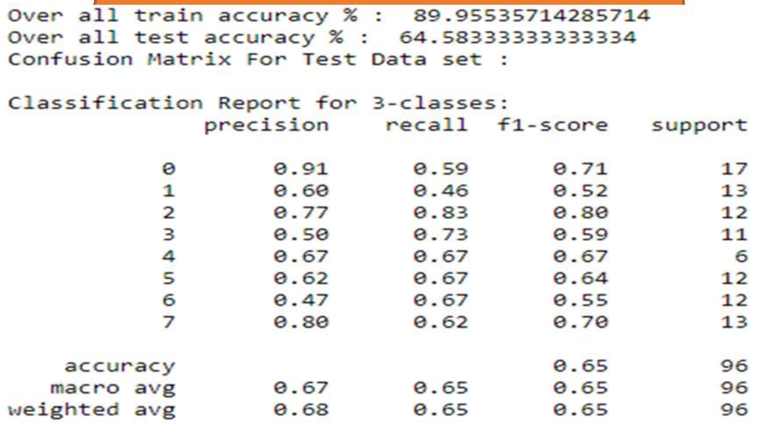
1. Features are the information or list of numbers that are extracted from an image.
2. When deciding about the features that could quantify plants and flowers, we could possibly think of Colour, Texture and Shape as the primary ones.
3. In this Project I will be used three inbuilt libraries for features extraction :
   - **Colour Histogram that quantifies colour of the flower.**
   - **Hu Moments that quantifies shape of the flower.**
   - **Haralick Texture that quantifies texture of the flower.**



1. Total I received 532 features from one image (columns), I have total 640 sample (rows)
2. I have divided the dataset into three part : Train, Test And Validation . According the requirement I will use .
3. Train part have 60% , test part 20% and validation part 20%. Splitting percentage ratio may vary according to algo.

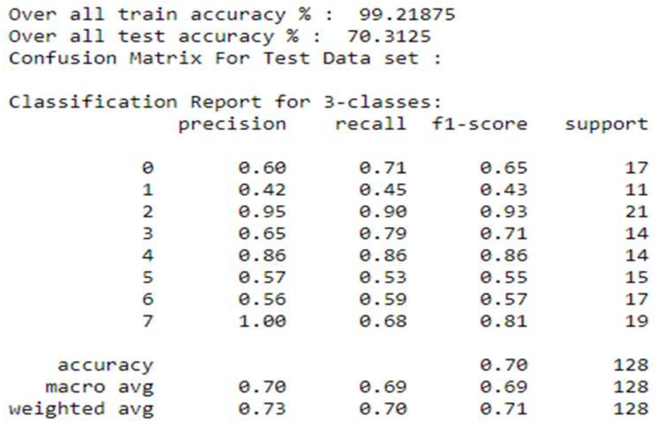
Result Analysis - Logistic Regression

Min MaxScaler Normalization



```
Over all train accuracy % :  89.95535714285714
Over all test accuracy % :  64.58333333333334
Confusion Matrix For Test Data set :

Classification Report for 3-classes:
              precision    recall  f1-score   support

           0       0.91      0.59      0.71        17
           1       0.60      0.46      0.52        13
           2       0.77      0.83      0.80        12
           3       0.50      0.73      0.59        11
           4       0.67      0.67      0.67         6
           5       0.62      0.67      0.64        12
           6       0.47      0.67      0.55        12
           7       0.80      0.62      0.70        13

    accuracy                           0.65        96
   macro avg       0.67      0.65      0.65        96
weighted avg       0.68      0.65      0.65        96
```

StandardScaler Normalization

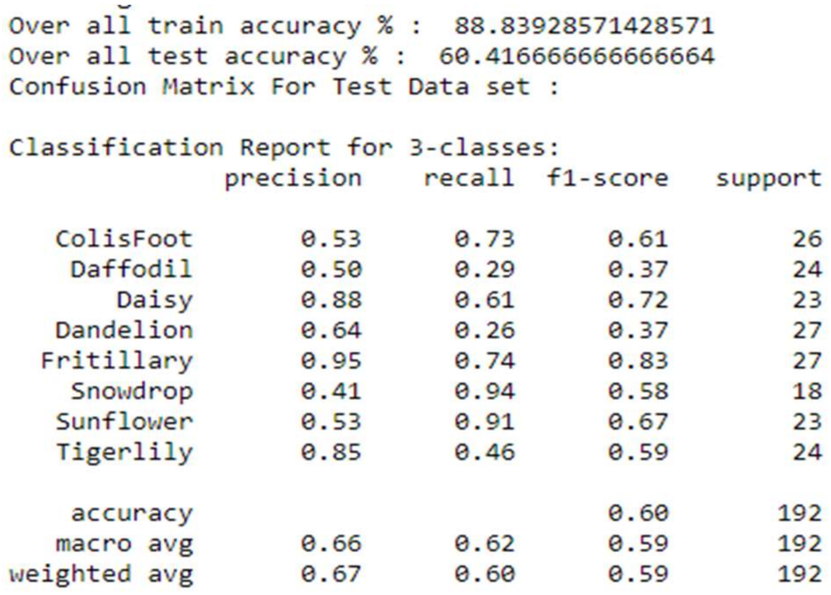


```
Over all train accuracy % :  99.21875
Over all test accuracy % :  70.3125
Confusion Matrix For Test Data set :

Classification Report for 3-classes:
              precision    recall  f1-score   support

           0       0.60      0.71      0.65        17
           1       0.42      0.45      0.43        11
           2       0.95      0.90      0.93        21
           3       0.65      0.79      0.71        14
           4       0.86      0.86      0.86        14
           5       0.57      0.53      0.55        15
           6       0.56      0.59      0.57        17
           7       1.00      0.68      0.81        19

    accuracy                           0.70       128
   macro avg       0.70      0.69      0.69       128
weighted avg       0.73      0.70      0.71       128
```

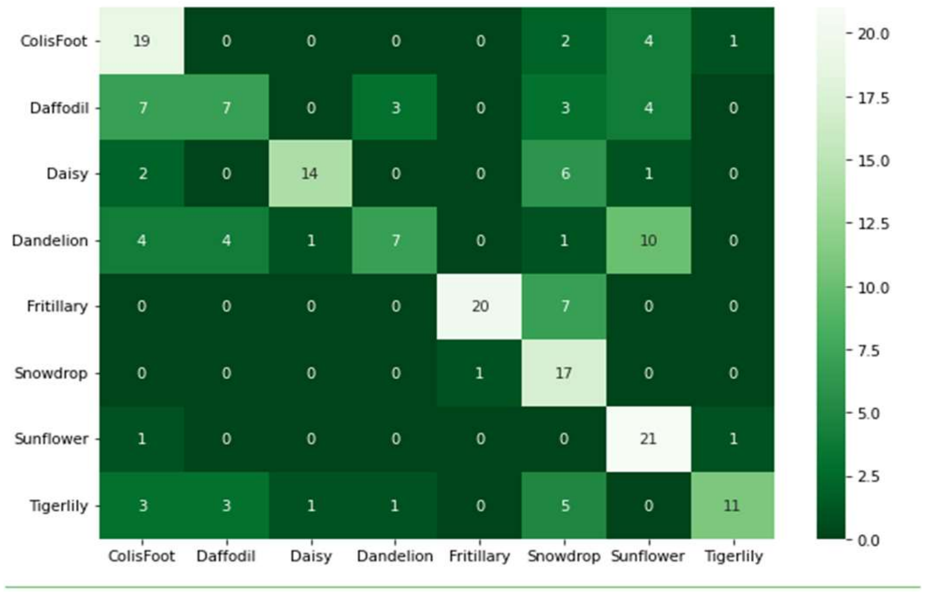
Conclusion :

- **I experimented with MinMaxScaler normalization technique.**
    - **[Batch Mode] Train Accuracy : 89.9% & Test Accuracy : 64.5%**
    - **[Stochastic Mode] Train Accuracy: 96.2% & Test Accuracy : 55%**
- **I experimented with StandardScaler normalization technique.**
    - **[Batch Mode] Train Accuracy : 99.2% & Test Accuracy : 70%**
    - **[Stochastic Mode] Train Accuracy: 97.9% & Test Accuracy : 72%**
- **I get more test accuracy in StandardScaler than MinMaxScaler.**

Result Analysis - Perceptron Learning Algorithm



```
Over all train accuracy % :  88.83928571428571
Over all test accuracy % :  60.416666666666664
Confusion Matrix For Test Data set :

Classification Report for 3-classes:
              precision    recall  f1-score   support

   ColisFoot       0.53      0.73      0.61        26
    Daffodil       0.50      0.29      0.37        24
       Daisy       0.88      0.61      0.72        23
   Dandelion       0.64      0.26      0.37        27
  Fritillary       0.95      0.74      0.83        27
    Snowdrop       0.41      0.94      0.58        18
   Sunflower       0.53      0.91      0.67        23
   Tigerlily       0.85      0.46      0.59        24

    accuracy                           0.60       192
   macro avg       0.66      0.62      0.59       192
weighted avg       0.67      0.60      0.59       192
```



	ColisFoot	Daffodil	Daisy	Dandelion	Fritillary	Snowdrop	Sunflower	Tigerlily
ColisFoot	19	0	0	0	0	2	4	1
Daffodil	7	7	0	3	0	3	4	0
Daisy	2	0	14	0	0	6	1	0
Dandelion	4	4	1	7	0	1	10	0
Fritillary	0	0	0	0	20	7	0	0
Snowdrop	0	0	0	0	1	17	0	0
Sunflower	1	0	0	0	0	0	21	1
Tigerlily	3	3	1	1	0	5	0	11

Conclusion :

- **Train Accuracy : 88.83% & Test Accuracy : 60.41%**

Result Analysis - Single layer perceptron (SLP) [one architecture]

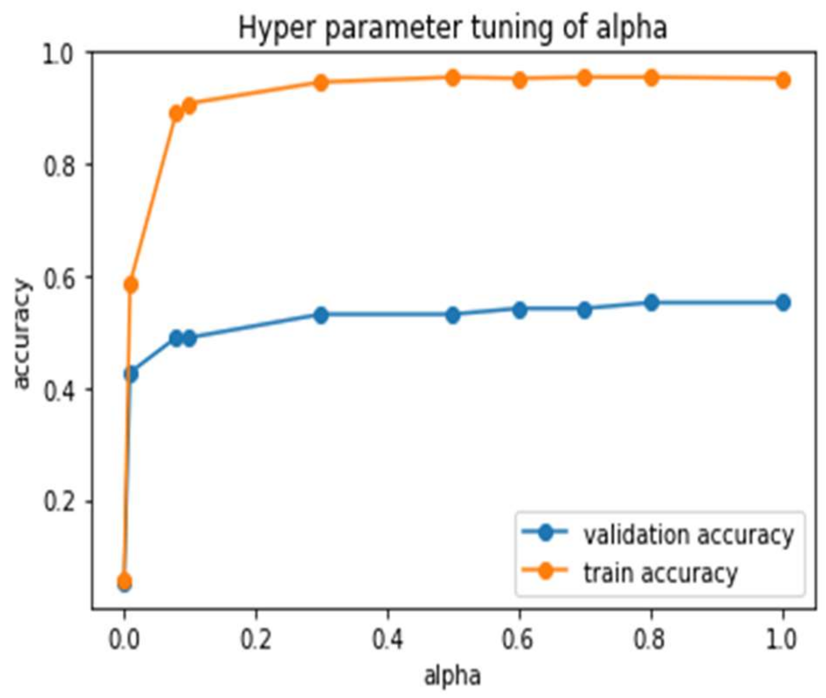
	Without tuning & checking overfitting	With tuning & checking overfitting
Train Accuracy	98%	95%
Test Accuracy	42%	45%

K- Fold Number	Train Accuracy	Test Accuracy	Test Cost	total cost
1	0.9	0.52	0.137266	0.122906
2	0.9	0.55	0.122762	0.122906
3	0.9	0.57	0.119737	0.122906
4	0.89	0.51	0.122236	0.122906
5	0.9	0.57	0.112529	0.122906

Conclusion:

- **After doing tuning of alpha(Slide number - 13), I train the model and finally after running 300 epochs I received 98% accuracy in training dataset and in testing data set I received 42% accuracy (more information in excel sheet).**
- **As I get less accuracy in test data set so I thought, need to check the overfitting problem as I shown in slide number(Slide number – 14).**
- **Finally after done the overfitting and tuning, I did 100 epochs, got 95% accuracy in training data and 45% in test data set.**
- **Applied K-fold cross validation, able to received 57% accuracy as you can see above mention table[Right side]**

Result Analysis - Single layer perceptron (SLP) [one architecture]

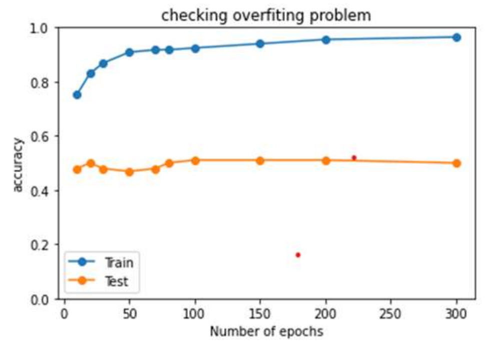


Tuning for best Alpha:

I train the model by varying the alpha value with considering 200 epochs. Then I plot graph and by seeing the graph I can say clearly that got maximum accuracy at 0.8 alpha value in validation data.

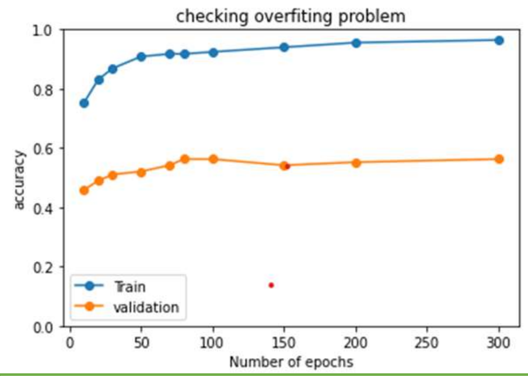
So finally best alpha values is 0.8

Result Analysis - Single layer perceptron (SLP) [one architecture]



Checking Overfitting Problem: Here I am trying to check the overfitting problem . I plot the graph of train accuracy and test accuracy by varying epochs. Finally I noticed that my model perform well in train dataset but not in test data set.

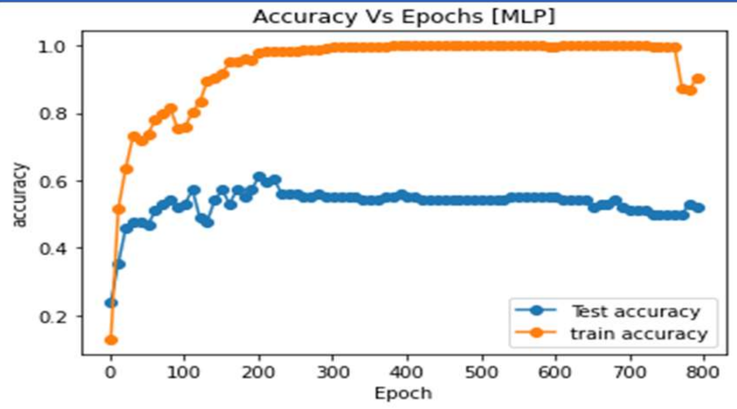
Conclusion: Train accuracy high, test accuracy very low as compare to train accuracy. I will try to solve overfitting problem…..



Overfitting: Here I am plotting the graph of train accuracy and validation accuracy by varying the epochs. Finally I noticed after 100 epochs that my validation accuracy decreases and train accuracy increases so the difference is going more ….

Conclusion: I need to stop training after 100 epochs to perform better in testing phase

Result Analysis - Multi layer perceptron (MLP)

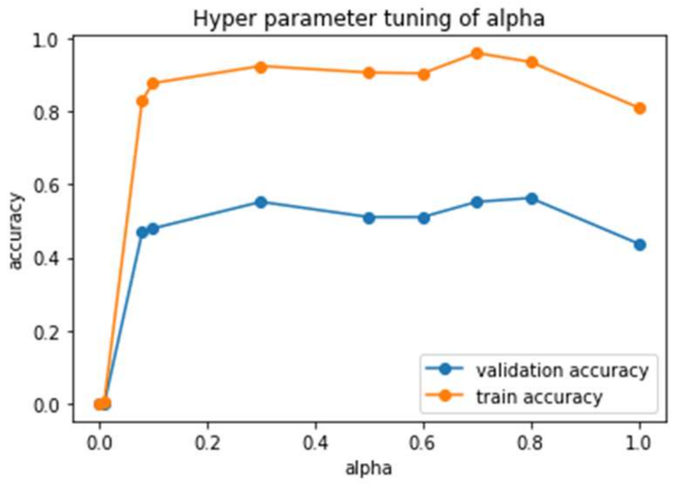


Fold Number	Train Accuracy	Test Accuracy
1	0.84	0.53
2	0.88	0.59
3	0.86	0.63
4	0.96	0.63
5	0.87	0.48

Conclusion:

- **After doing tuning (Slide number - 16), I train the model and finally after running 200 epochs I received 98% accuracy in training dataset and in testing data set I received 61% accuracy (result data in excel sheet).**
- **Applied K-fold cross validation, able to received 63% accuracy of test as you can see above mention table[Right side]**

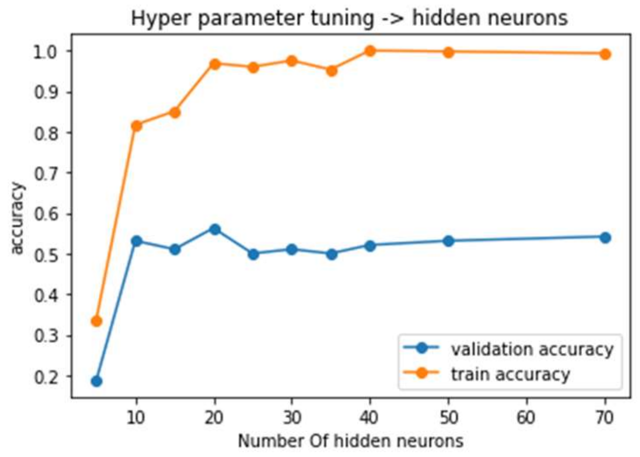
Result Analysis - Single layer perceptron (SLP) [one architecture]



Tuning for best Alpha:

I train the model by varying the alpha value with considering 200 epochs and 20 hidden neurons. Then I plot graph and by seeing the graph I can say clearly that got maximum accuracy at 0.8 alpha value in validation data.

So finally best alpha values is 0.8



Tuning for number of neurons in hidden layer:

I train the model by varying the number of neurons value with considering 200 epochs and 0.8 alpha value. Then I plot graph and by seeing the graph I can say clearly that got maximum accuracy at 20 neurons.

Brief Comparison among all previous applied algo

- **Logistic Regression: (Inbuilt - sklearn)**
  - **With MinMaxScaler normalization technique, its able to achieve 64.5% accuracy**
  - **With StandardScaler normalization technique, its able to achieve 72% accuracy**
  - **It's performing best compared to all other algorithms that are evaluated.**
- **Perceptron Learning Algorithm(PLA) : (Inbuilt - sklearn)**
  - **Train Accuracy : 88.83% & Test Accuracy : 60.41%**
- **Single Layer Perceptron:**
  - **For 5-fold it's able to achieve an accuracy of 57% with my own code (No inbuilt)**
  - **without k-fold (train : 95% ; test : 45 %) with my own code (No inbuilt)**
- **Muli Layer Perceptron:**
  - **For 5-fold it's able to achieve an accuracy of 63.23% with 1 hidden layer of 20 hidden neurons and max iterations of 500 with my own code (No inbuilt)**
  - **It can be observed that as the number of hidden neurons increases, the more linearity the trained dataset will tend to and increases the accuracy. Which is described by the cover's theorem**
- **Accuracy Comparison: Logistic Regression > MLP > PLA > SLP**