Section A

Q1.a)

SECTION-A

Q1.(a)(a) Height of Feature Map $= \frac{M-K+2P}{S} + 1$

$P=0$
$S=1$

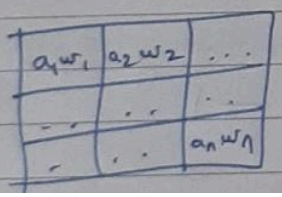
$= M-K+1$

Width of Feature Map $= \frac{N-K+2P}{S} + 1$

$= N-K+1$

$\therefore$ Dimensions $\rightarrow (M-K+1, N-K+1)$

(b) No. of elementary operations
= No. of multiplications + No. of additions



No. of multiplications
= No. of cells in kernel × No. of channels

$= K \times K \times P$

We add P because there are P channels so P different products of weights and image.

No. of additions = No. of cells in kernel $\times P - 1$

(Addition of 3 items
↳ 2 addition operations)

$= K \times K \times P - 1$

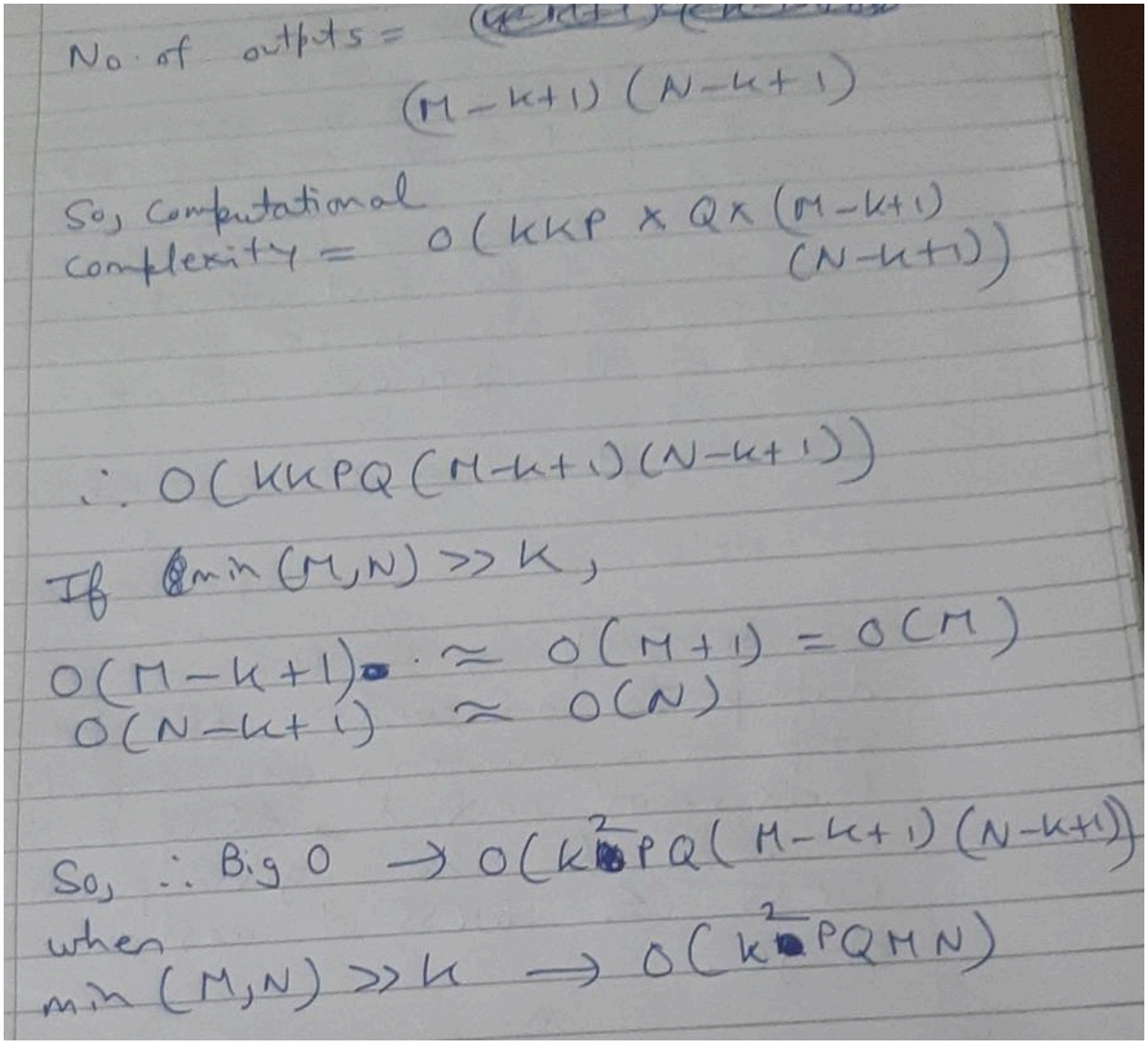
Total number of elementary operations
$= 2 \times K \times K \times P - 1$

cc) No. of multiplications per kernel per output in feature map $= K \times K \times P$

No. of additions per kernel per output in feature map $= K \times K \times P - 1$

Total no. of operations per output per kernel $= 2K \times K \times P - 1 = O(K \times K \times P)$

No. of kernels $= Q$


No. of outputs = (M-K+1)(N-K+1)
So, Computational complexity = O(KKP x Q x (M-K+1) (N-K+1))
∴ O(KKPQ(M-K+1)(N-K+1))
If min(M,N) >> K,
O(M-K+1) ≈ O(M+1) = O(M)
O(N-K+1) ≈ O(N)
So, ∴ Big O → O(K²PQ(M-K+1)(N-K+1))
when min(M,N) >> K → O(K²PQMN)

Q1.b)

~~Update Step~~

Q1.c.b) Assignment Step → We assign each data point to an existing centroid based on the minimum Euclidean distance to any of them. As long as it's not the first iteration, the WCSS will always be lower than that of the previous iteration.

For every pt. ci) → $c^{(i)} = \arg\min_j \|x^{(i)} - \mu_j\|^2$

$j \rightarrow$ Cluster no.s.

Update Step → For the set of datapoints achieved in the previous step for a cluster, we compute their new centroid, ~~Update~~ by taking mean of all the given points. The WCSS ≤ WCSS of assignment stage.

$$\forall j \quad \mu_j = \frac{\sum_{i=1}^{n} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{n} 1\{c^{(i)} = j\}}$$

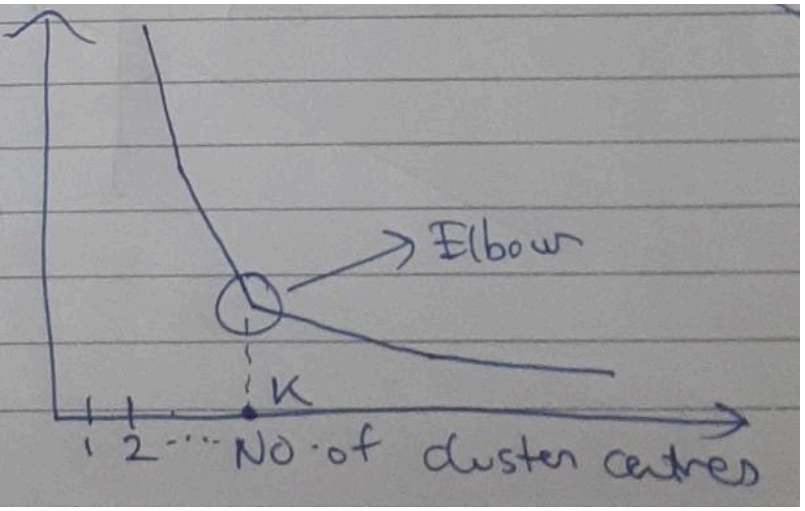
One method that helps in determining optimal number of clusters is the

'ELBOW' method. Here, the theory states that the optimal no. of clusters must strike a perfect balance between computation cost and performance. So, we try out K-Means for a bunch of different no. of cluster centers and plot the WCSS for each no. of centers.

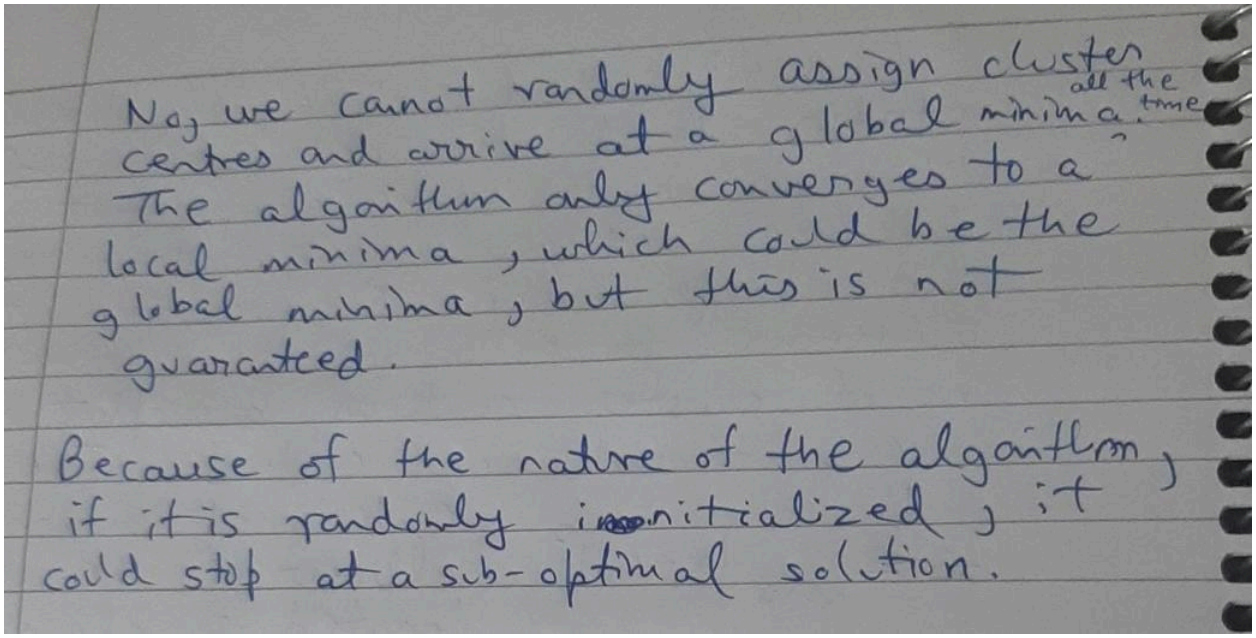
$$\sum_{i=1}^{K} \sum_{x_i \in C_j} |x_i - v_j|^2$$

The no. of centers at which the elbow of the curve is located is chosen to be the optimal no. of cluster centres.

Example →



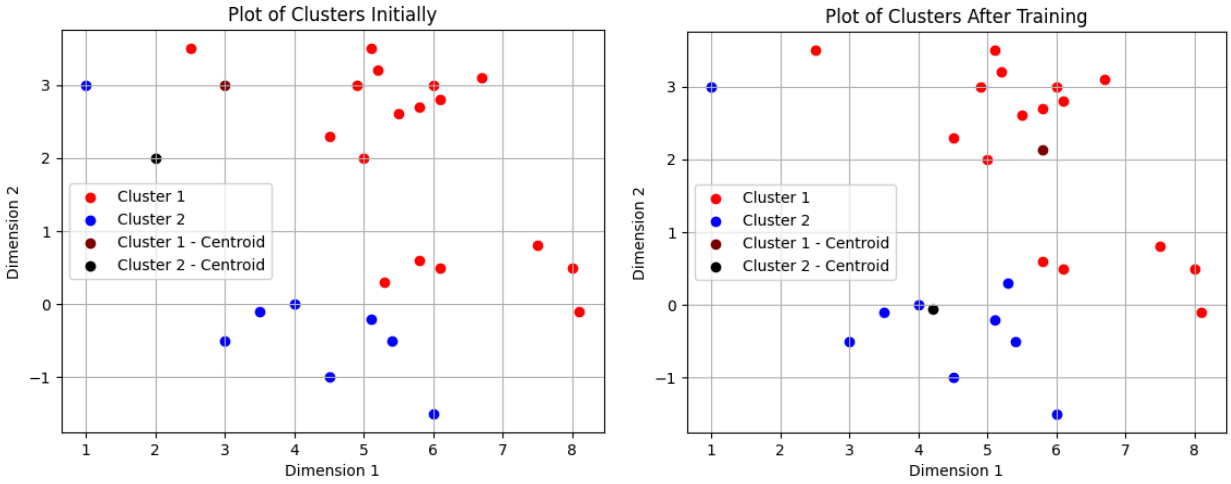
We choose K as optimal no. of cluster centres



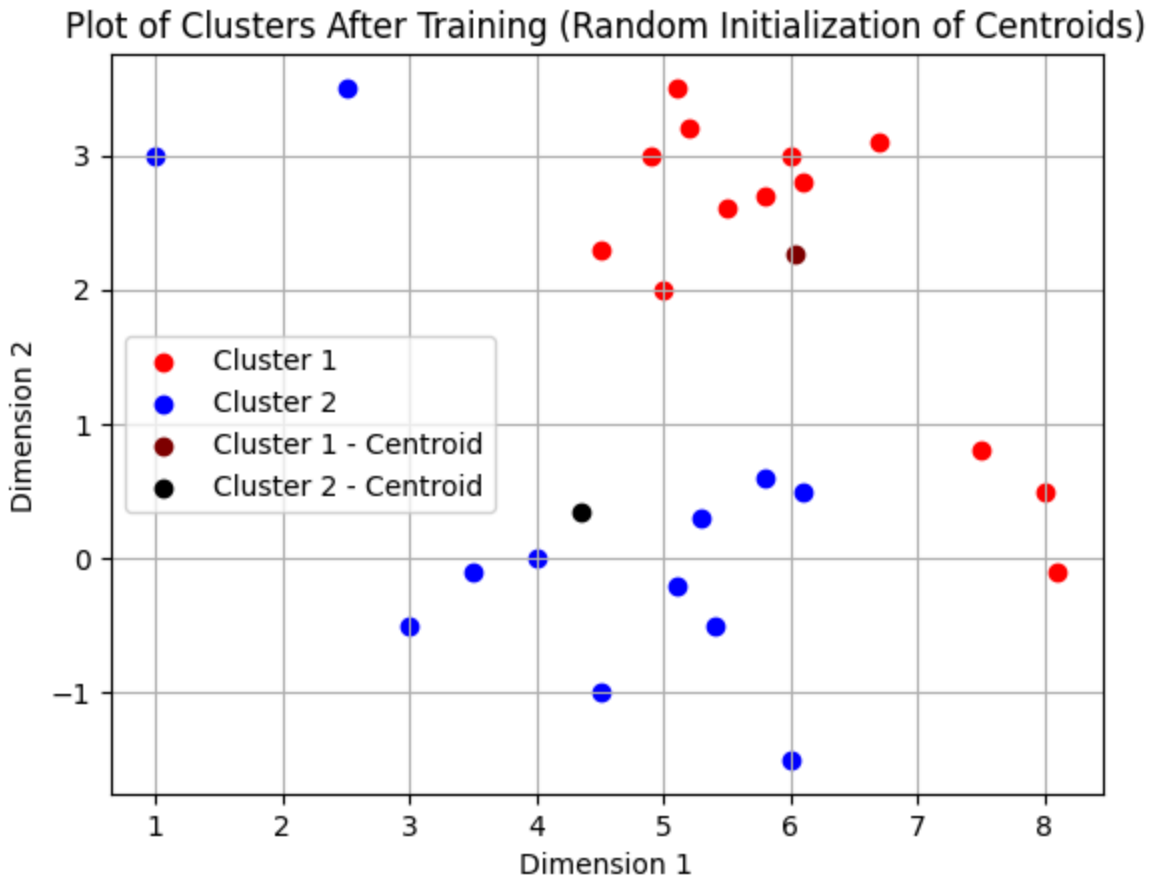
SECTION B

Q2) Assuming convergence threshold is sum of changes of centroids, not max change of an individual centroid.

b) Final Centroids -> [(5.8, 2.125), (4.199999999999999, -0.05555555555555555)]

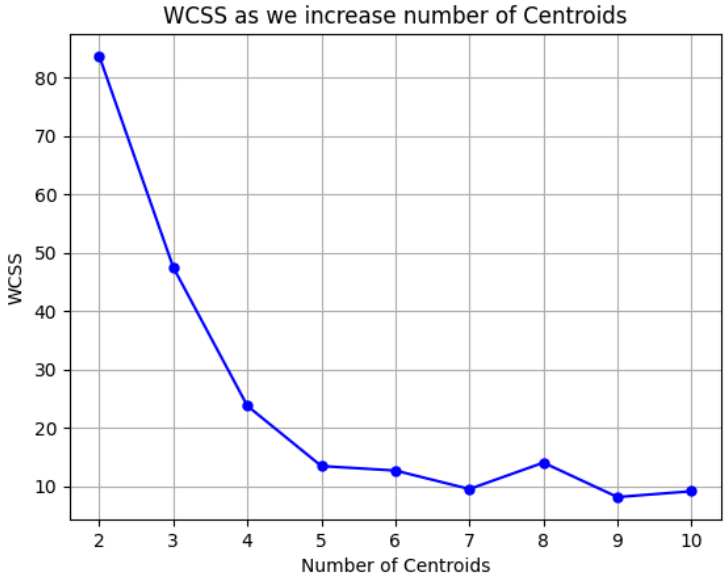


c) WCSS with provided centroids -> 83.67222222222222.
WCSS with Random Clusters -> 85.1776282051282



So, it is slightly worse, though again, depends completely on how the centroids are randomly generated. The algorithm converges to a local minima not global, hence the difference in loss.

(d)

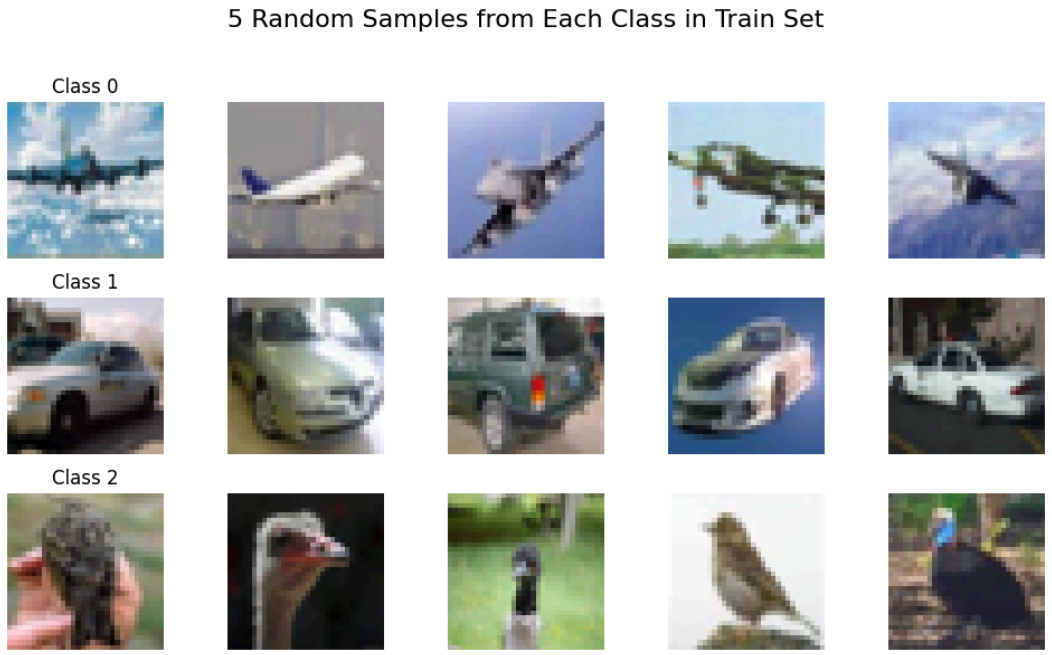


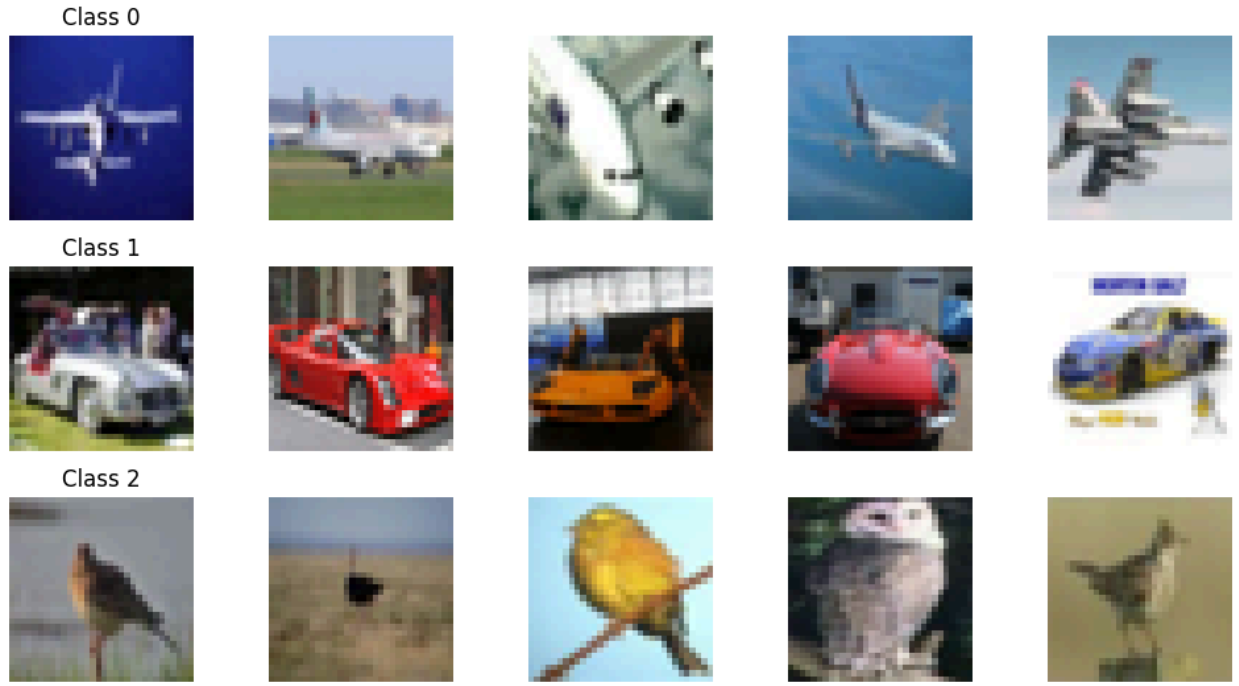
Using this, we can set M to 5, as it is clearly the Elbow.


Plot of M Clusters After Training
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 1 - Centroid
Cluster 2 - Centroid
Cluster 3 - Centroid
Cluster 4 - Centroid
Cluster 5 - Centroid
Dimension 1
Dimension 2

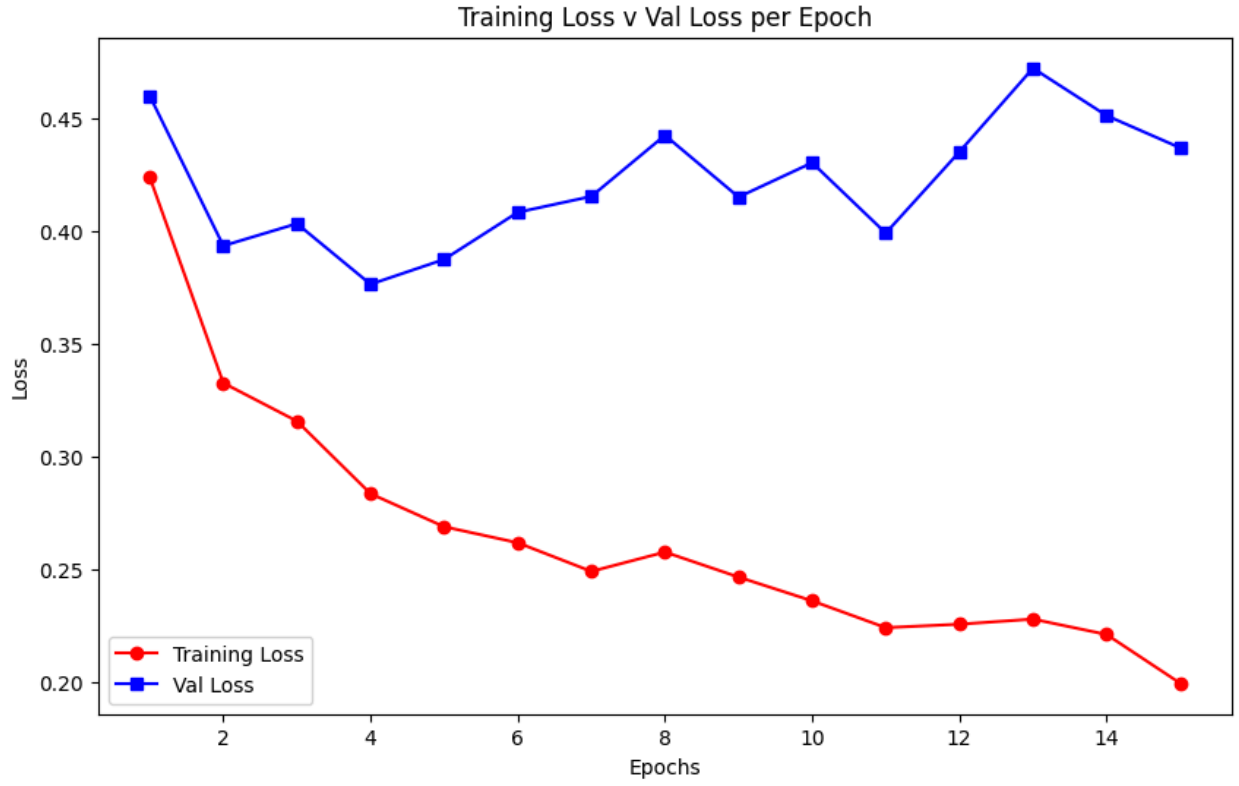
SECTION C

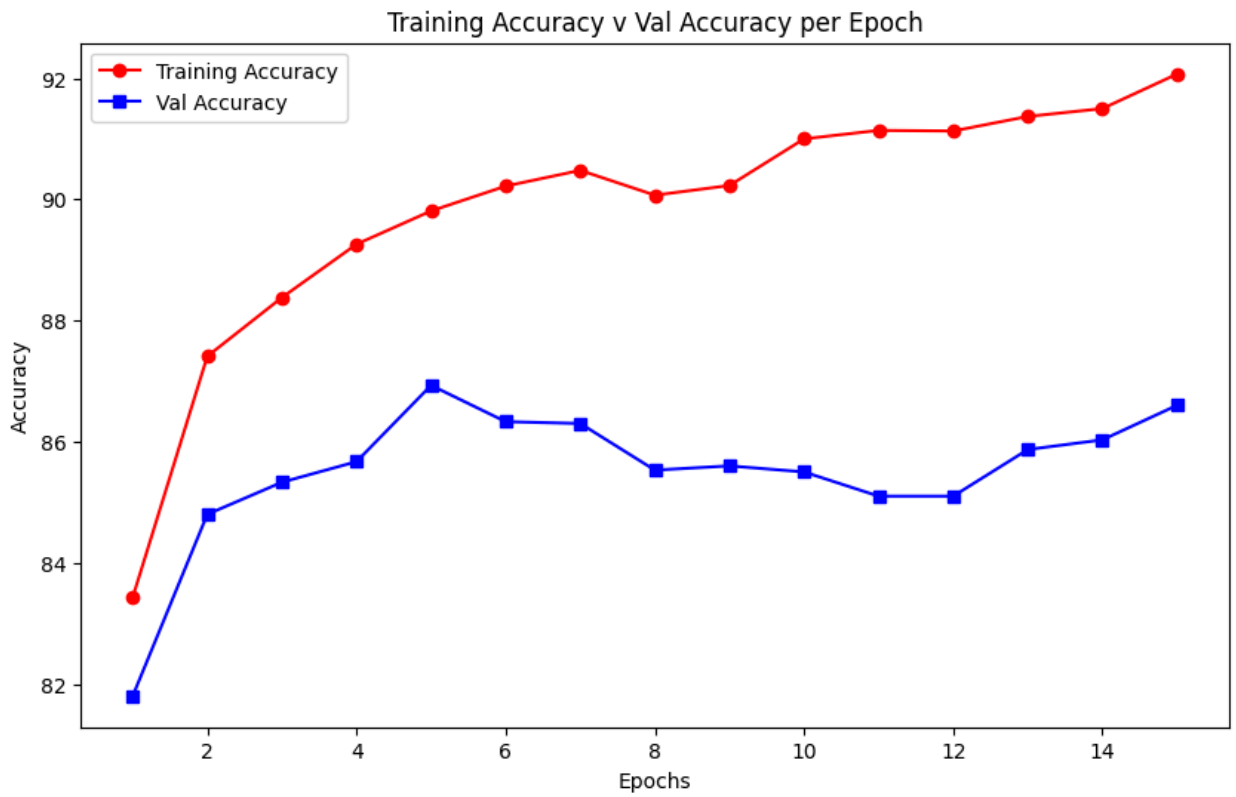
Q2)


5 Random Samples from Each Class in Train Set
Class 0
Class 1
Class 2


5 Random Samples from Each Class in Val Set
Class 0
Class 1
Class 2

Q5)

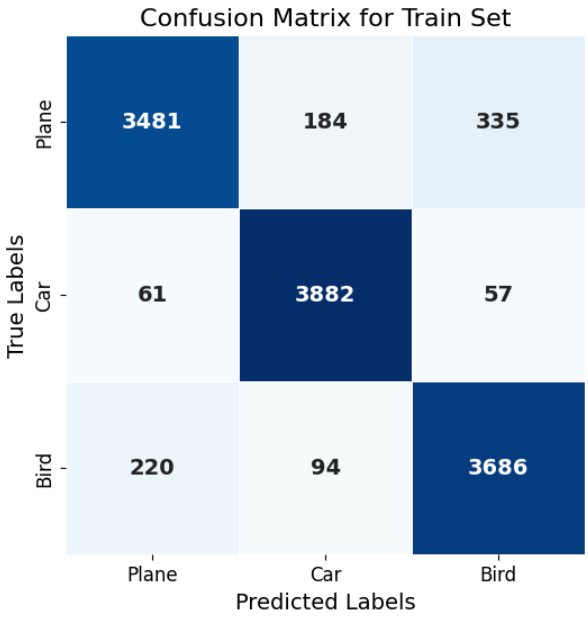

Training Loss v Val Loss per Epoch
Loss
Epochs
Training Loss
Val Loss
0.45
0.40
0.35
0.30
0.25
0.20
2
4
6
8
10
12
14

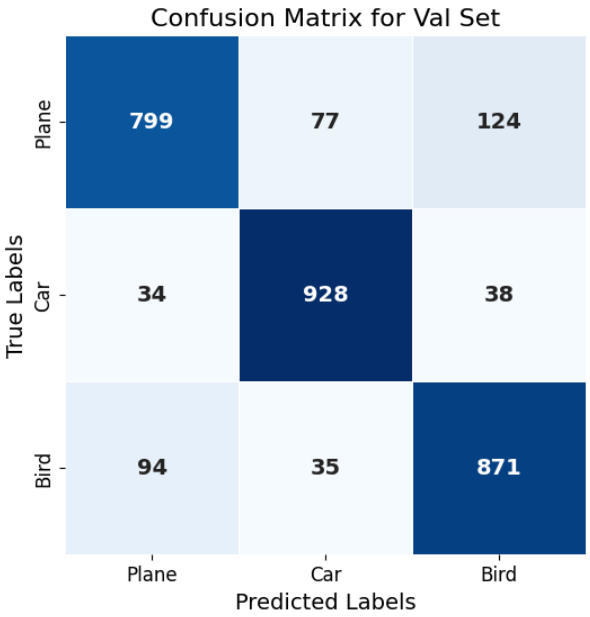


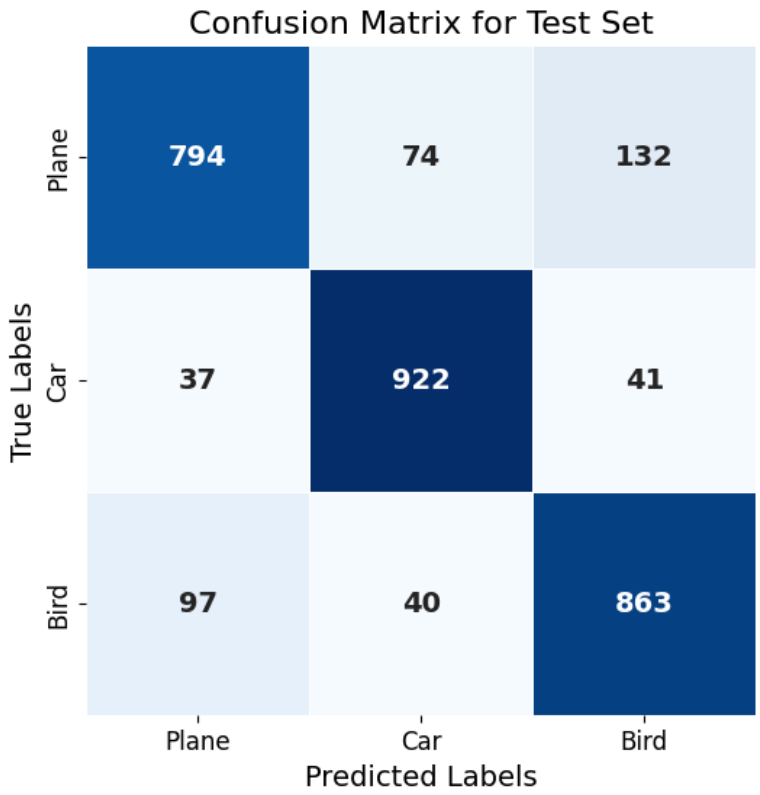

```
Observations:-

1) The val loss indicates that there is overfitting, since from epoch 8
onwards, there is an increase in val loss, which is a good indicator of
overfitting.
2) The accuracies for train are increasing steadily which is good, while
the val accuracy does stop increasing after epoch 5 and oscillates a bit,
with the model perhaps not generalizing as well as the training loss
indicates.
3) So, overall, one could say that the model should have been stopped a
bit earlier, since it's overfitting but it's still just about fine.
```

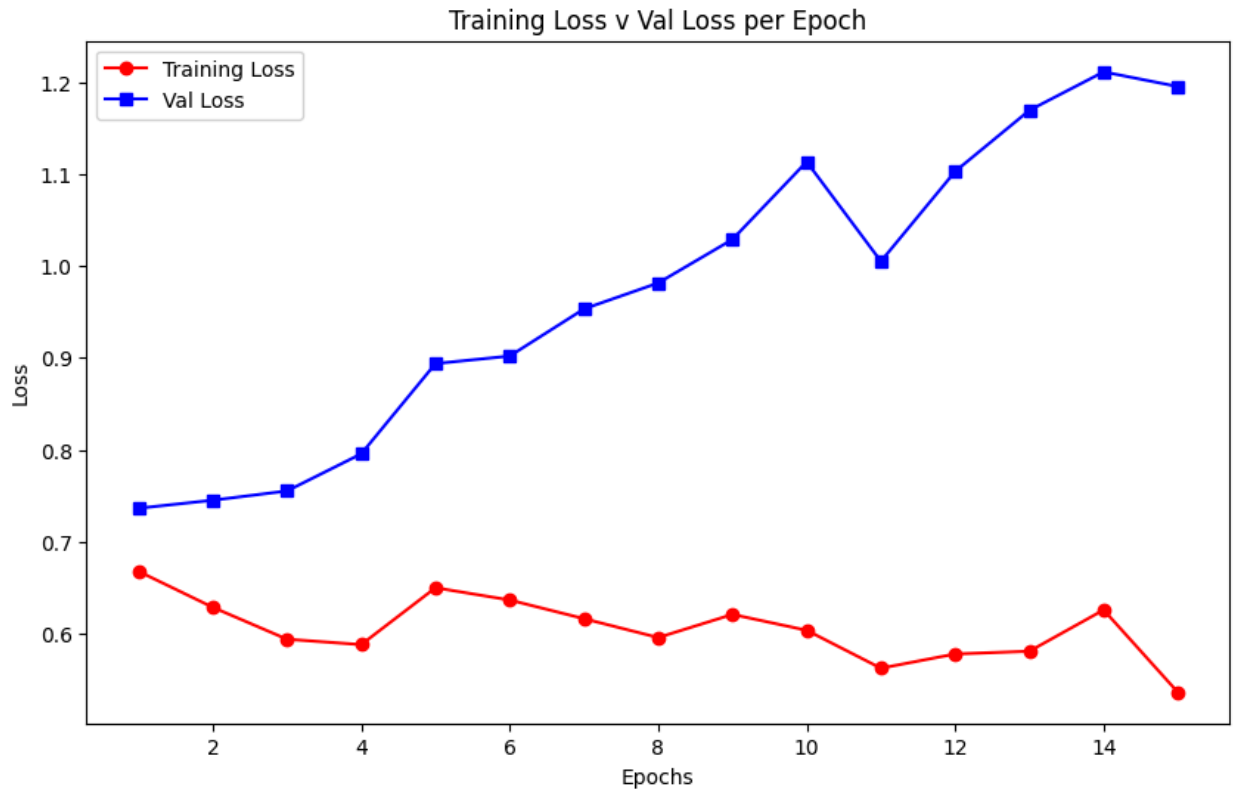
Test Accuracy -> 85.97%
Test F1 Score -> 0.8590298891067505

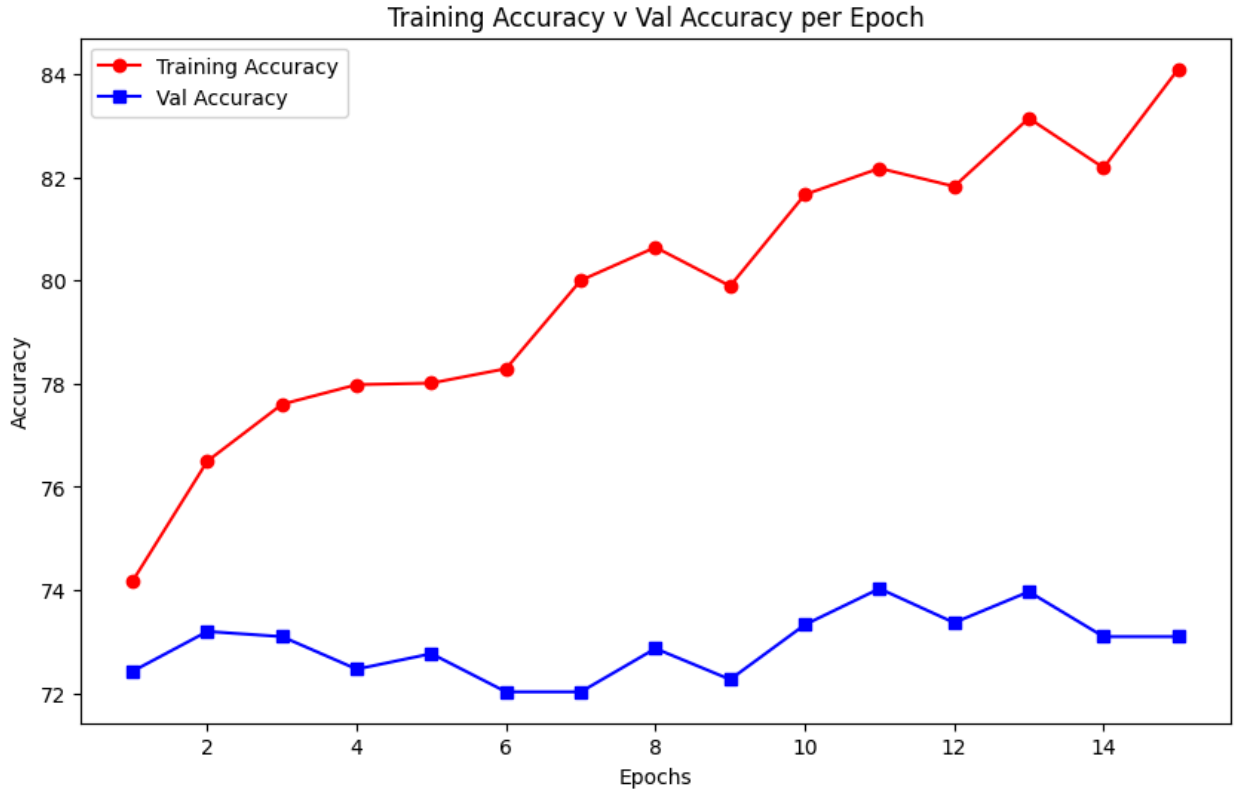

Confusion Matrix for Train Set
Plane
Car
Bird
True Labels
3481
184
335
61
3882
57
220
94
3686
Plane
Car
Bird
Predicted Labels


Confusion Matrix for Val Set
Plane
Car
Bird
True Labels
799
77
124
34
928
38
94
35
871
Plane
Car
Bird
Predicted Labels


Confusion Matrix for Test Set
Plane
Car
Bird
True Labels
794
74
132
37
922
41
97
40
863
Plane
Car
Bird
Predicted Labels

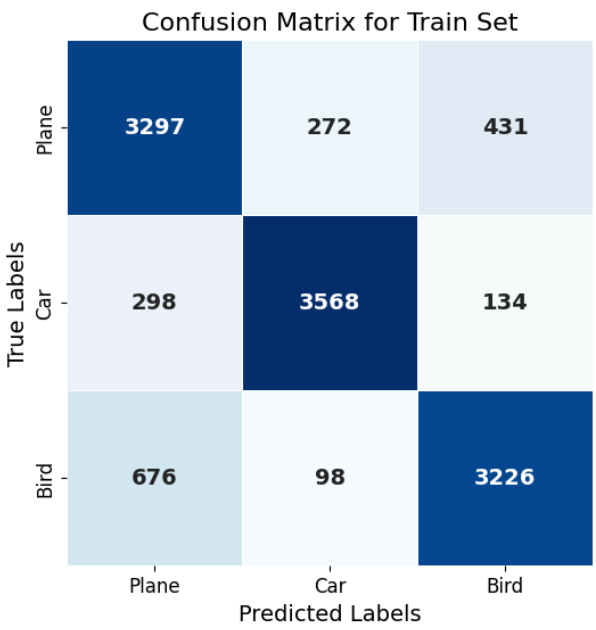
Q6)

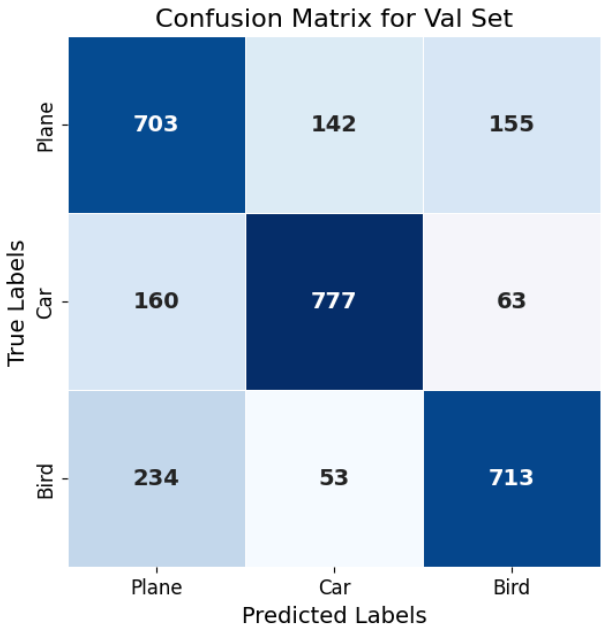


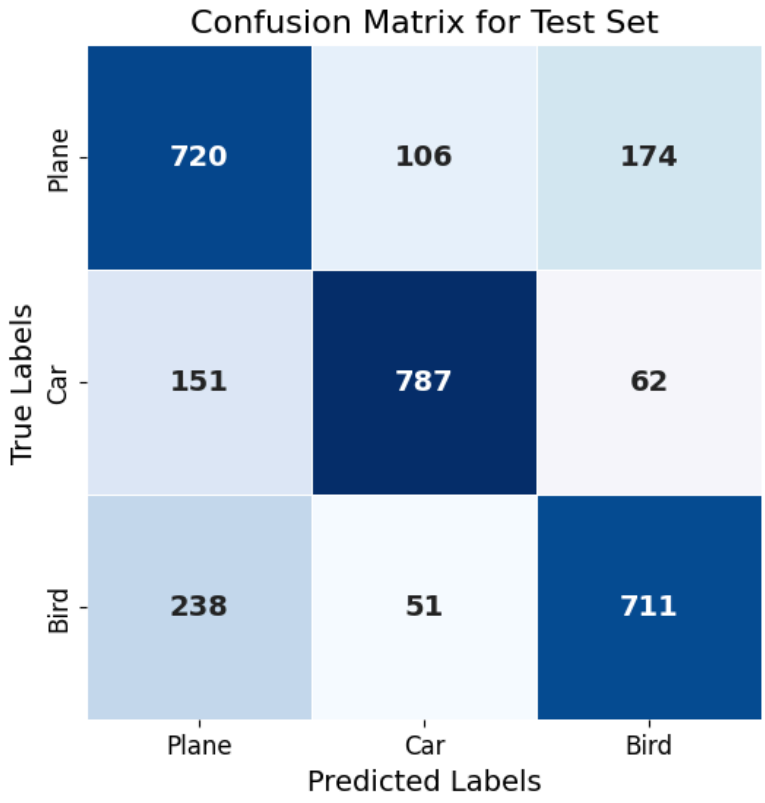


Q7) MLP:
Test Accuracy -> 73.93%

Test F1 Score -> 0.7409377694129944








```
Final Comparisons:-

1) Clearly, by looking at both accuracy and macro F1 score on the test
set, CNN does a significantly better job. There is a difference in 12% in
```

the accuracy and 0.12 in the F1 score. Clearly these 2 indicate a large difference in performance.

2) Both models do mistake birds with planes and vice versa more so than any other pair. Perhaps, this could be due to the relative similarities in the shape of the 2, which leads to this misclassification. MLP suffers much more in this regard, as one can see from the graph.

3) Predicting cars is a strong suit for both models, clearly this is a follow-up effect of part 2.
4) CNNs overfit a bit less than MLPs though by looking at the loss graphs for both, they both do overfit.
5) In MLPs, the model doesn't train as quickly as CNNs, this can also be analyzed by looking at the loss curves.
6) Clearly, CNNs do a good job at image classification, since both F1 Score and Accuracy are high.