Assignment 1

Team member 1:
Name: Varun Vaddi
PeopleSoft number: 2347481
Name on Kaggle leaderboard: Vamsi_n_Varun
Contribution Description: Developed the code along with Vamsi for both perceptron and SoftMax. Ran 3 models of Perceptron & 4 models of SoftMax (but when merging both the code models, ran into issue where the name of all 8 files of SoftMax are named as vyeruban, couldn't re-run due to time constraints). Prepared the Report document for submission.

Team member 2:
Name: Vamsi Krishna Yerubandi
PeopleSoft number: 2351142
Name on Kaggle leaderboard: Vamsi_n_Varun
Contribution Description: Developed the code along with Varun for both perceptron and SoftMax. Ran 3 models of Perceptron & 4 models of SoftMax. Prepared the Kaggle submission and Zip file.

Theory

A1	- Perceptron - Logistic Regression - Stochastic Gradient Descent
A2	**<u>Gradient Descent:</u>** -It computes Gradient of Loss function for the whole training dataset. - As a result, convergence is slow, as it need to run whole dataset for each iteration. - Expensive and takes more time to process for large datasets. **<u>Stochastic Gradient Descent (SGD):</u>** -It computes Gradient of Loss function for the randomly selected subset of the training dataset. - Convergence is quicker. -Less time to process large datasets, as it only considers a small subset of data. In simple terms, Gradient Descent is more stable but slower, while SGD is quicker but can be a bit more unpredictable.
A3	Binary Cross-entropy Loss/ Log Loss

③ Loss function - Logistic Regression :-

$$\hat{y} = \sigma(w^T x + b) = \text{sigmoid}(w^T x + b)$$

the loss in Logistic Regression is called as "Log Loss" (or) "Binary cross-entropy loss". It measures the difference (∇) between Predicted probability & true label, thereby Penalizing incorrect classifications

$$L(w, b ; D) = \frac{-1}{n} \sum_{i=1}^{n} \left(y^{(i)} \cdot \log \hat{y}^{(i)} + (1 - y^{(i)}) \cdot \log (1 - \hat{y}^{(i)}) \right)$$

where,

$y^{(i)}$ = true label

$\hat{y}^{(i)}$ = predicted probability.

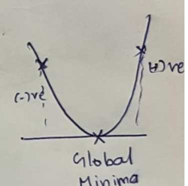
σ = sigmoid function

$L(w, b ; D)$ = Negative Likelihood function

A4 Derive the SGD for Log Reg

④ Stochastic Gradient Descent :-

$$L(w, b, (x^{(i)}, y^{(i)})) = (w \cdot x^{(i)} + b - y^{(i)})^2$$



$$w_{new} = w - \frac{\partial L}{\partial w}(w, b, (x^{(i)}, y^{(i)}))$$

$$b_{new} = b - \frac{\partial L}{\partial b}(w, b, (x^{(i)}, y^{(i)}))$$

$$\boxed{L(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2n} \sum_{i=1}^{n} (y_i^{(i)} - \hat{y}_i^{(i)})^2}$$

$n \rightarrow$ No. of data points

but, we know that

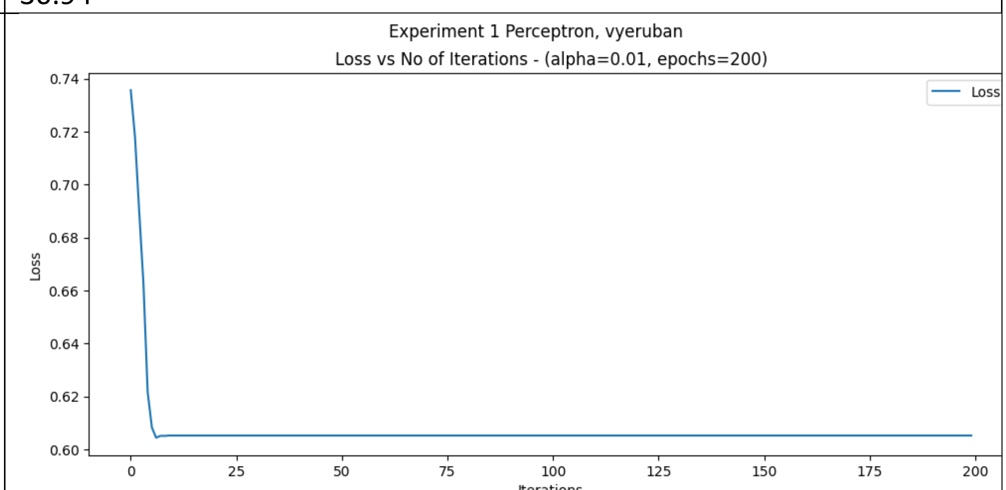
$$\boxed{\hat{y}^{(i)} = wx + b}$$

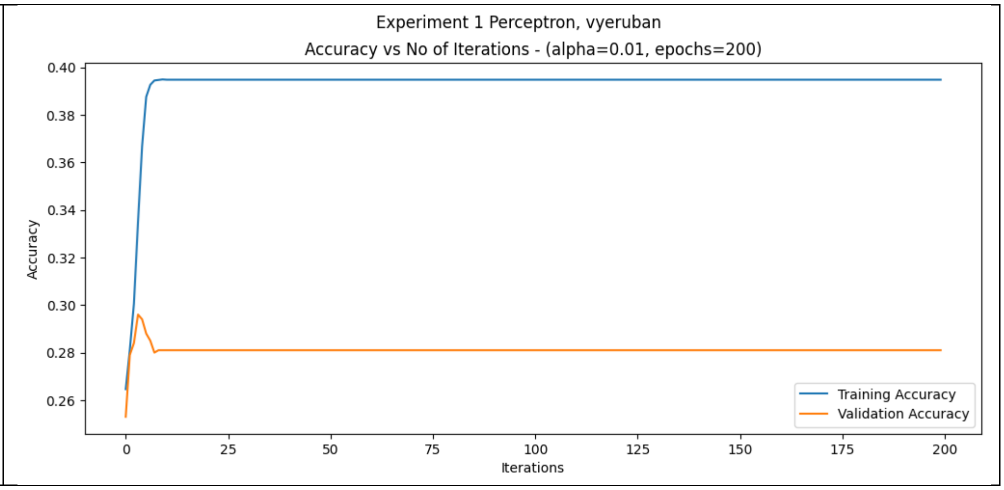
	Computing gradients for Loss function wrt (w) $\frac{\partial L}{\partial w} = -\frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)}).x_i$ Computing gradient for Loss function wrt (b) $\frac{\partial L}{\partial b} = -\frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})$ $w_{new} \leftarrow w_{old} - \eta . \frac{\partial L}{\partial w}$ $b_{new} \leftarrow b_{old} - \eta . \frac{\partial L}{\partial b}$ $\eta \longrightarrow$ Learning rate.

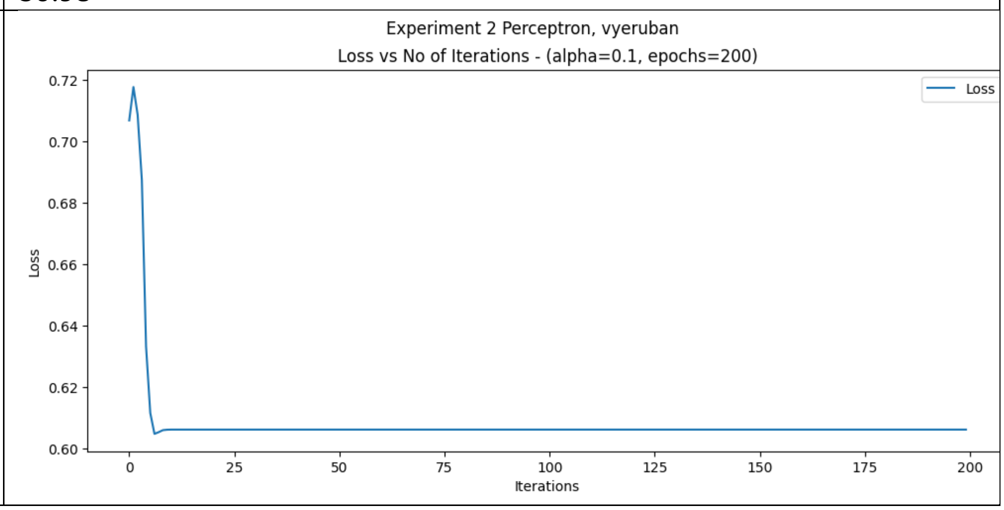
Perceptron:

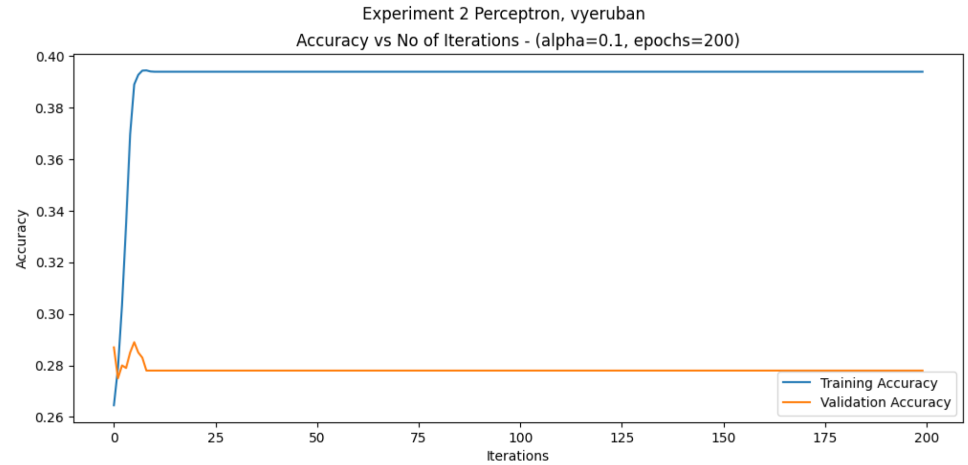
You should also mention whether adding a learning rate decay helped and how you implemented this decay.

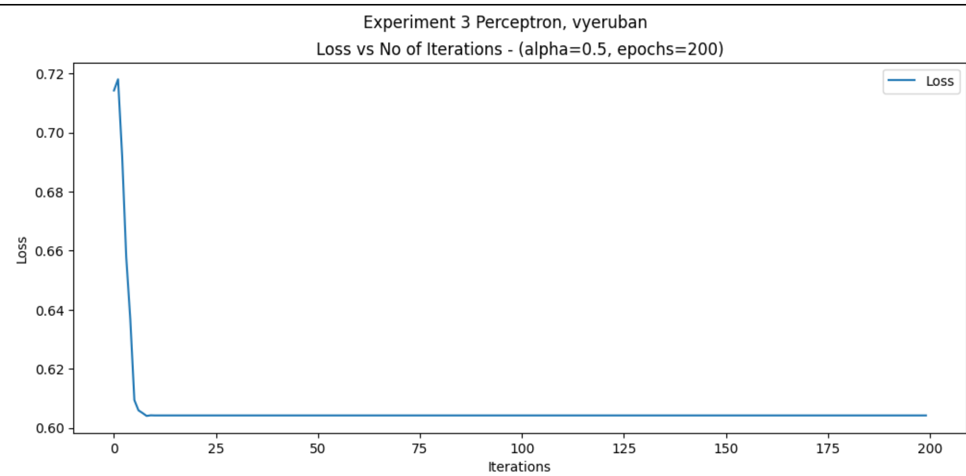
Experimentation

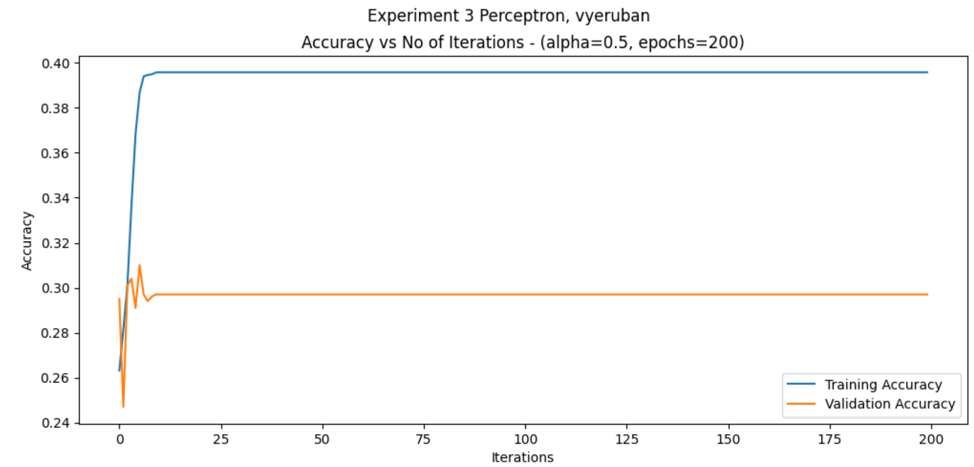
Experiment #	1
Hyperparameters	Learning rate = 0.01, epoch = 200, decay = 0.8
Training accuracy	39.47
Validation accuracy	28.1
Test accuracy	30.94
Loss vs number of iterations plot using matplotlib	

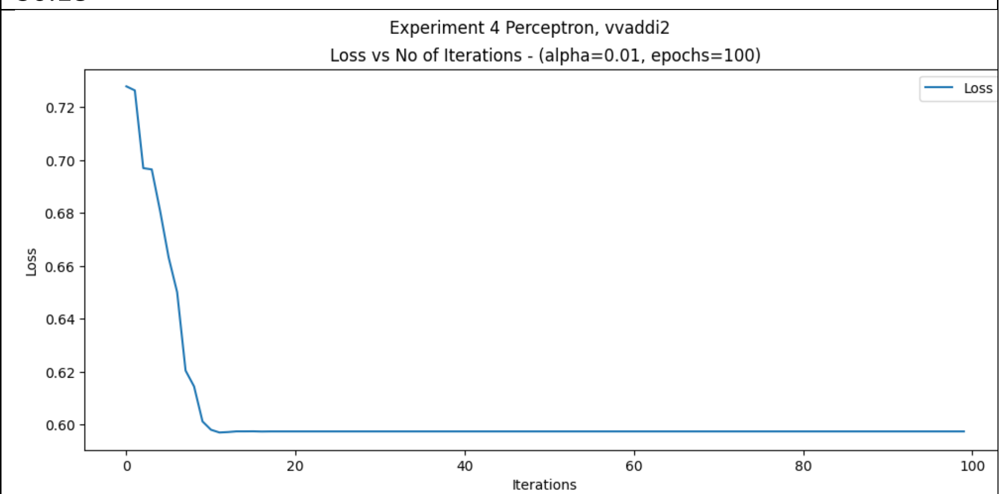
Training and validation accuracy vs number of iterations plot	

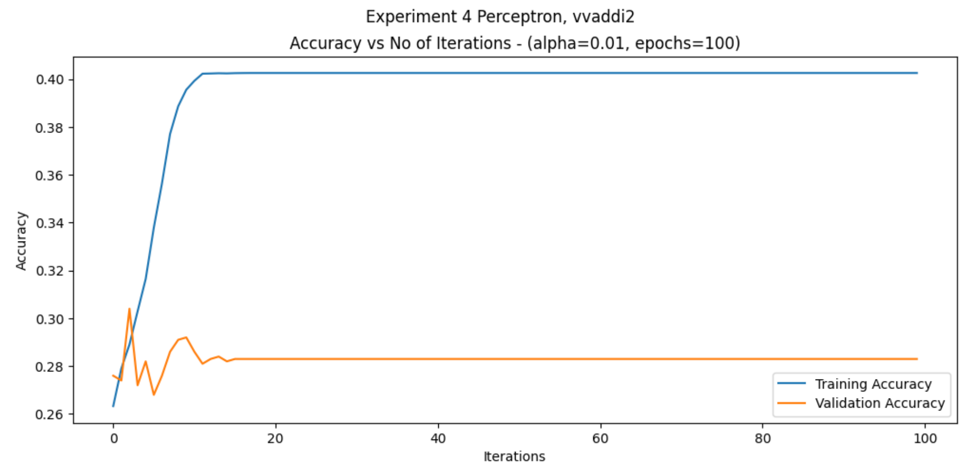
Experiment #	2
Hyperparameters	Learning rate = 0.1, epoch = 200, decay = 0.8
Training accuracy	39.38
Validation accuracy	27.8
Test accuracy	30.98
Loss vs number of iterations plot using matplotlib	

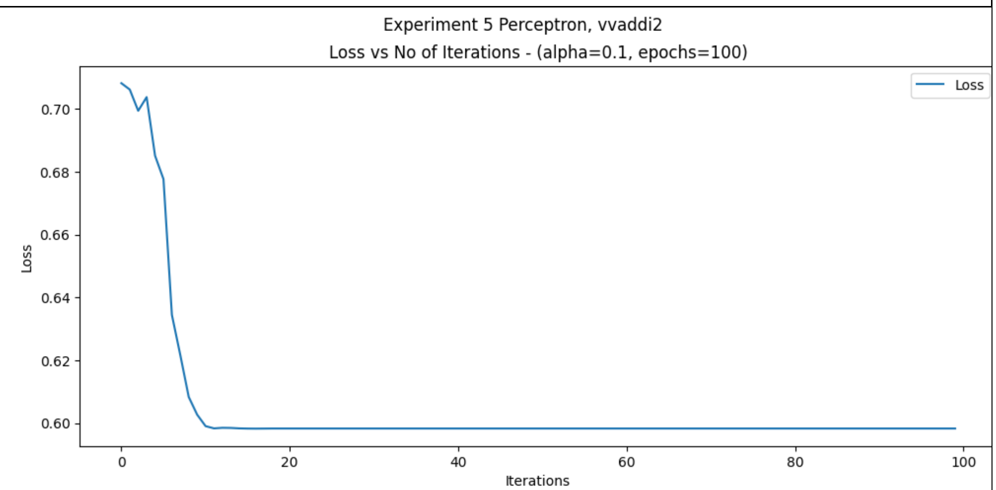
Training and validation accuracy vs number of iterations plot	

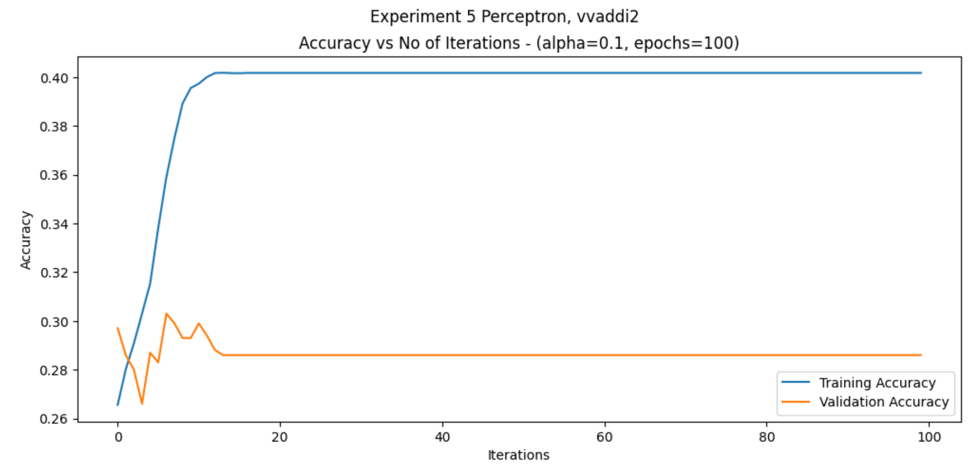
Experiment #	3
Hyperparameters	Learning rate = 0.5, epoch = 200, decay = 0.8
Training accuracy	39.57
Validation accuracy	29.7
Test accuracy	30.92
Loss vs number of iterations plot using matplotlib	

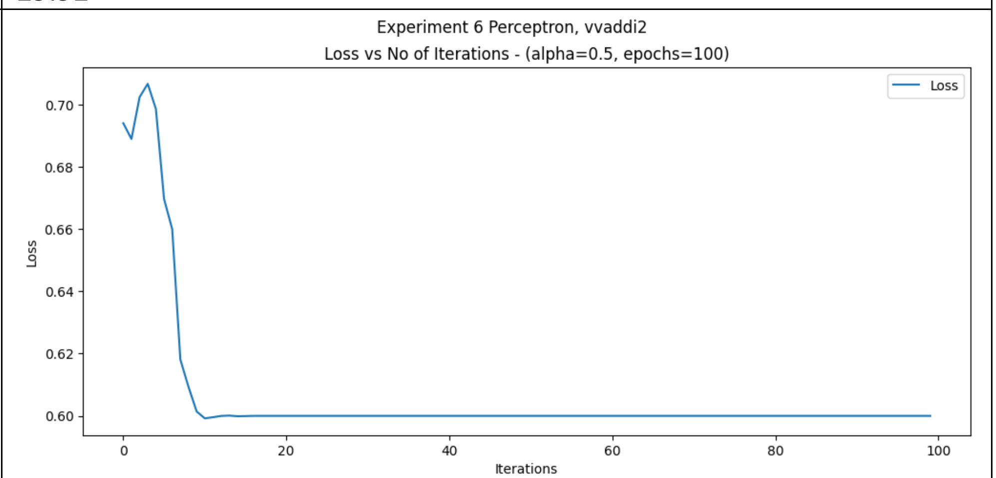
Training and validation accuracy vs number of iterations plot	

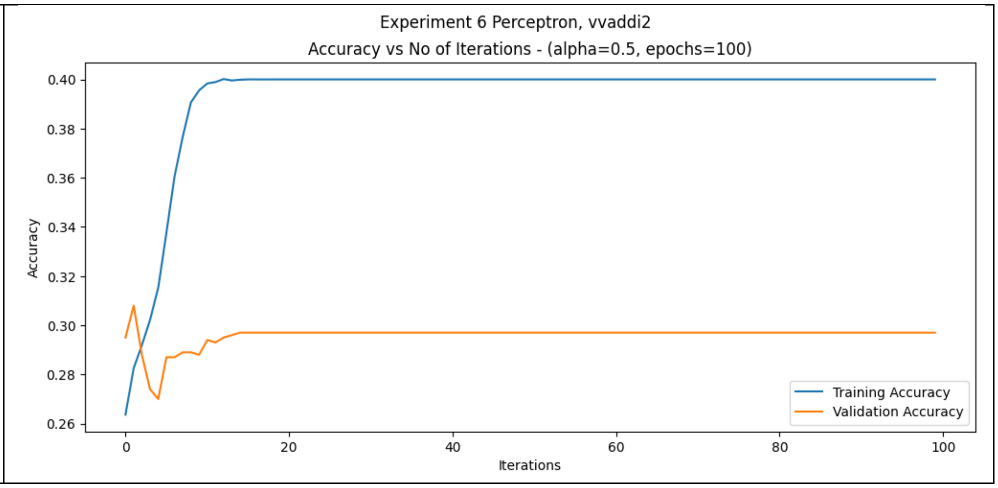
Experiment #	4
Hyperparameters	Learning rate = 0.01, epoch = 100, decay = 0.2
Training accuracy	40.25
Validation accuracy	28.3
Test accuracy	30.18
Loss vs number of iterations plot using matplotlib	

Training and validation accuracy vs number of iterations plot	

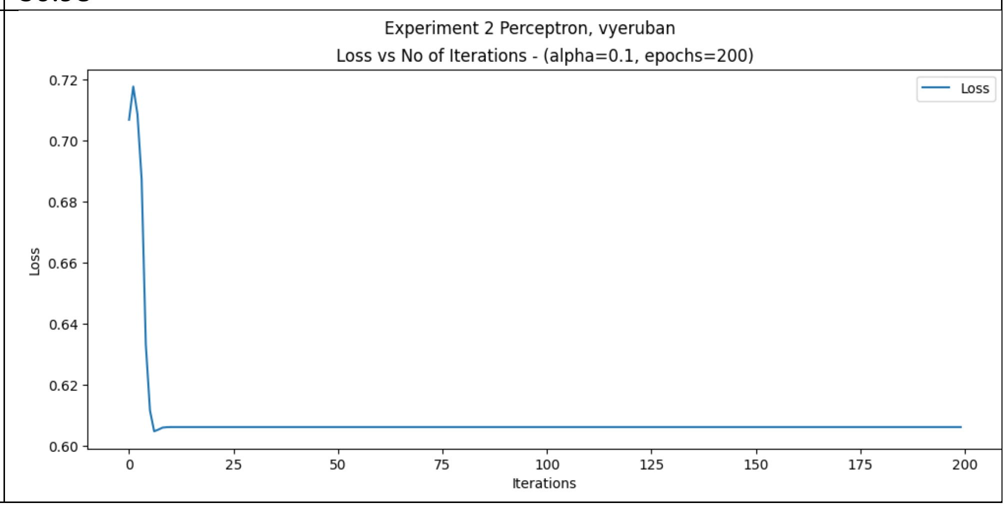
Experiment #	5
Hyperparameters	Learning rate = 0.1, epoch = 100, decay = 0.2
Training accuracy	40.17
Validation accuracy	28.6
Test accuracy	30.56
Loss vs number of iterations plot using matplotlib	

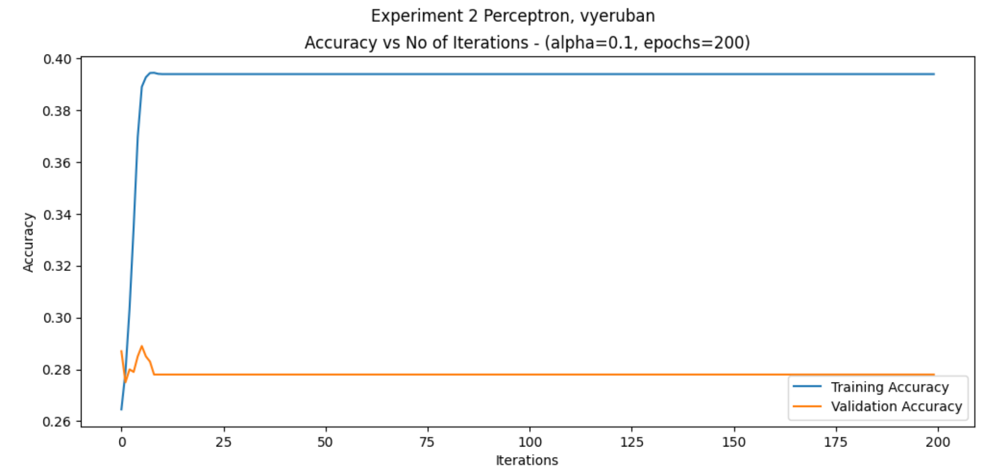
Training and validation accuracy vs number of iterations plot	

Experiment #	6
Hyperparameters	Learning rate = 0.5, epoch = 100, decay = 0.2
Training accuracy	40
Validation accuracy	29.7
Test accuracy	29.92
Loss vs number of iterations plot using matplotlib	

Training and validation accuracy vs number of iterations plot	

Hyperparameters of Kaggle model

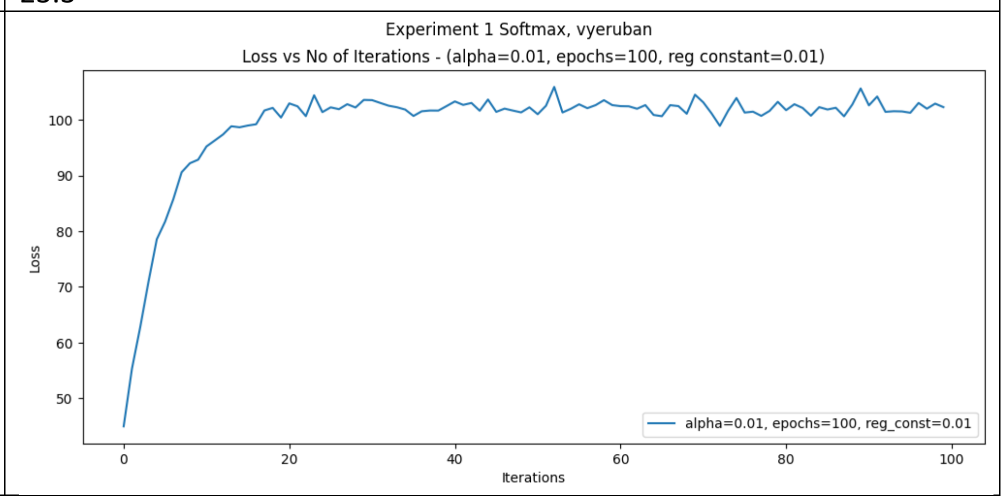
Optimal hyperparameters	Learning rate = 0.1, epoch = 200, decay = 0.8
Training accuracy	39.38
Validation accuracy	27.8
Test accuracy	30.98
Loss vs number of iterations plot	

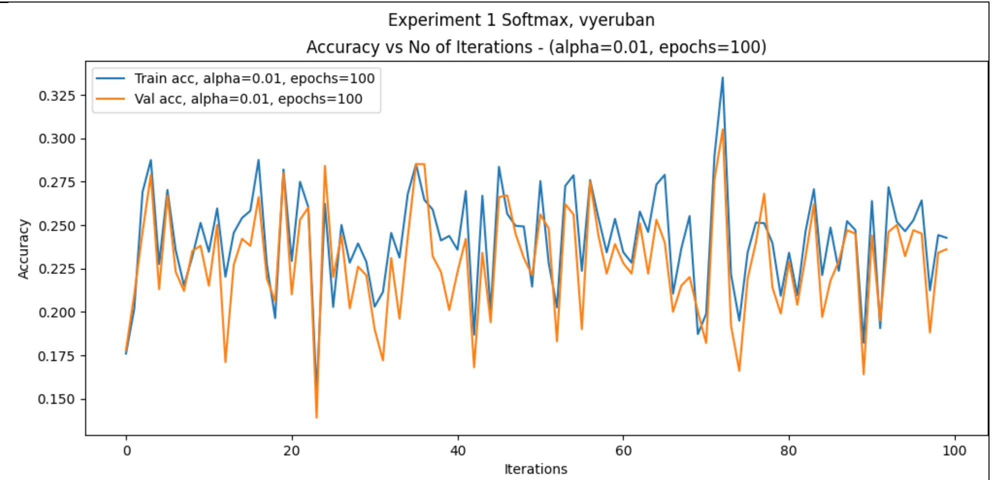
Training and validation accuracy vs number of iterations plot	

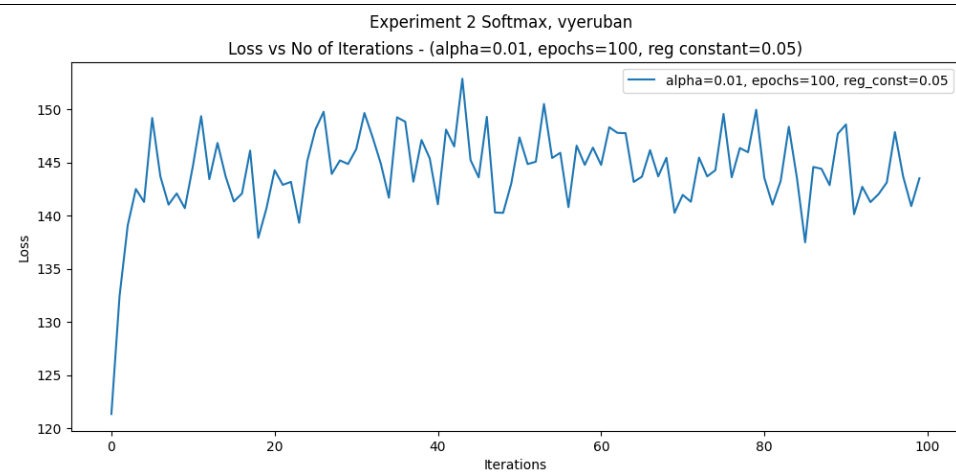
Softmax:

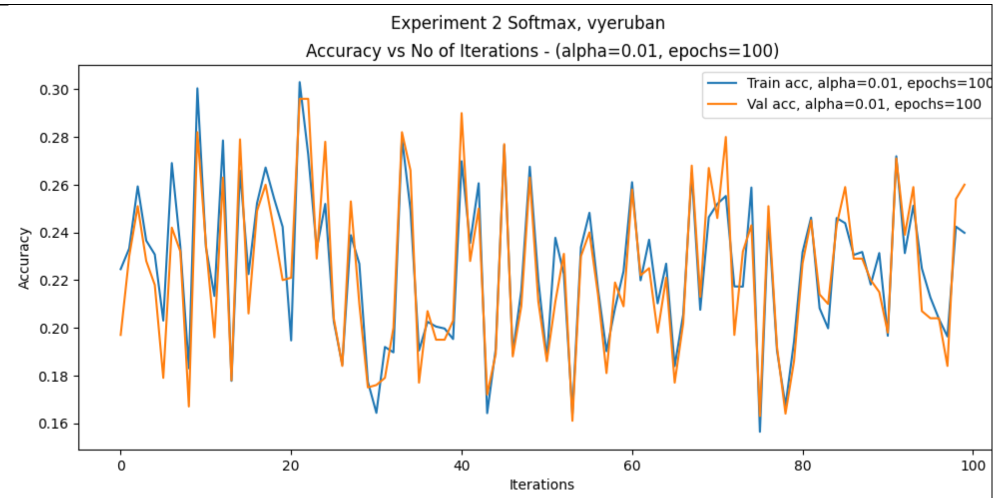
Also report your training, validation, and testing accuracy with your optimal hyperparameter setting.

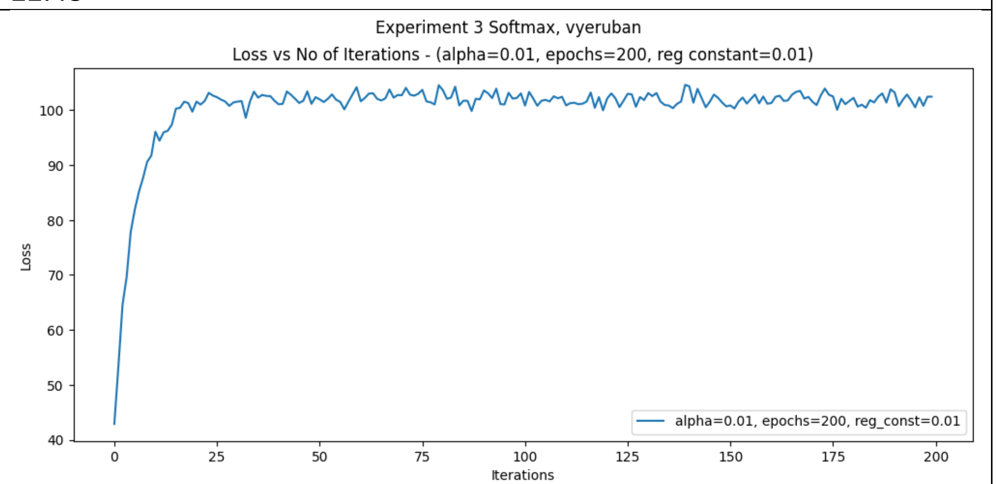
Experimentation

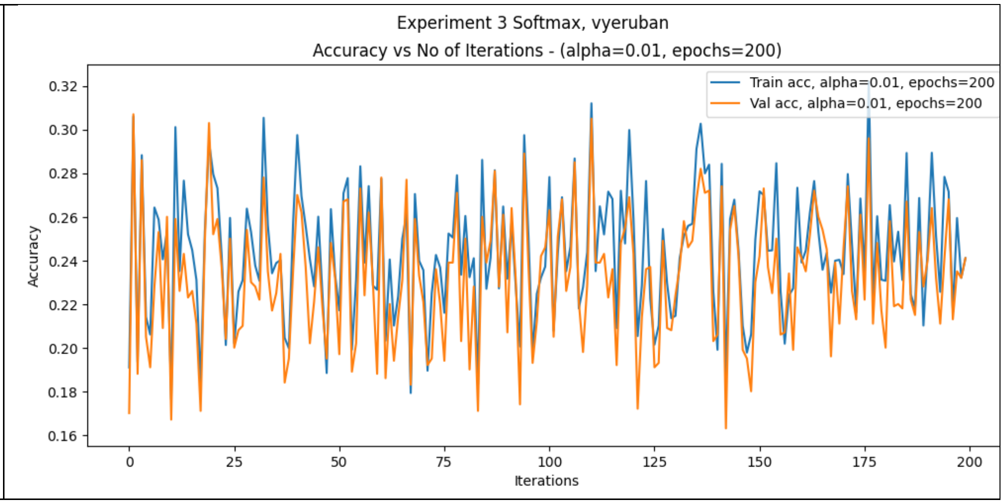
Experiment #	1
Hyperparameters	Learning rate = 0.01, epoch = 100, regularization constant = 0.01
Training accuracy	24.25
Validation accuracy	23.59
Test accuracy	23.3
Loss vs number of iterations plot using matplotlib	

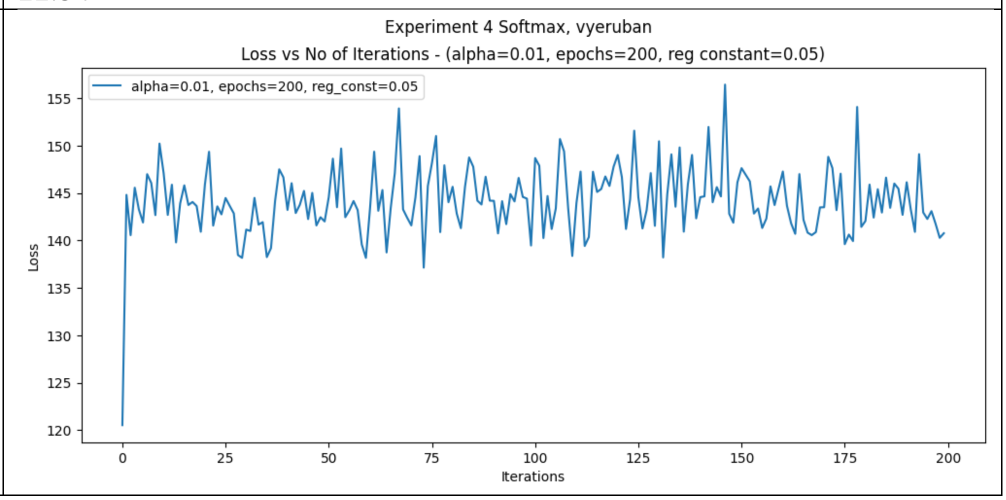
Training and validation accuracy vs number of iterations plot	

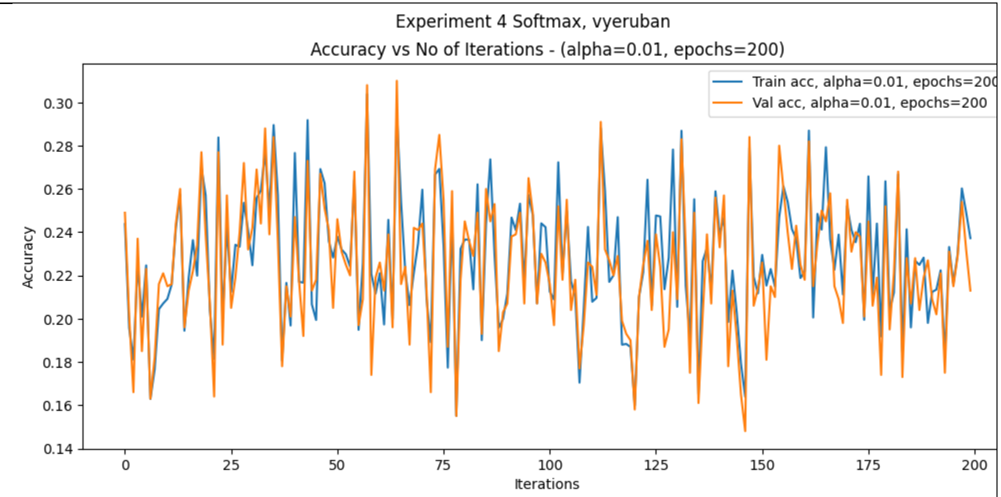
Experiment #	2
Hyperparameters	Learning rate = 0.01, epoch = 100, regularization constant = 0.05
Training accuracy	23.98
Validation accuracy	26
Test accuracy	23.7
Loss vs number of iterations plot using matplotlib	

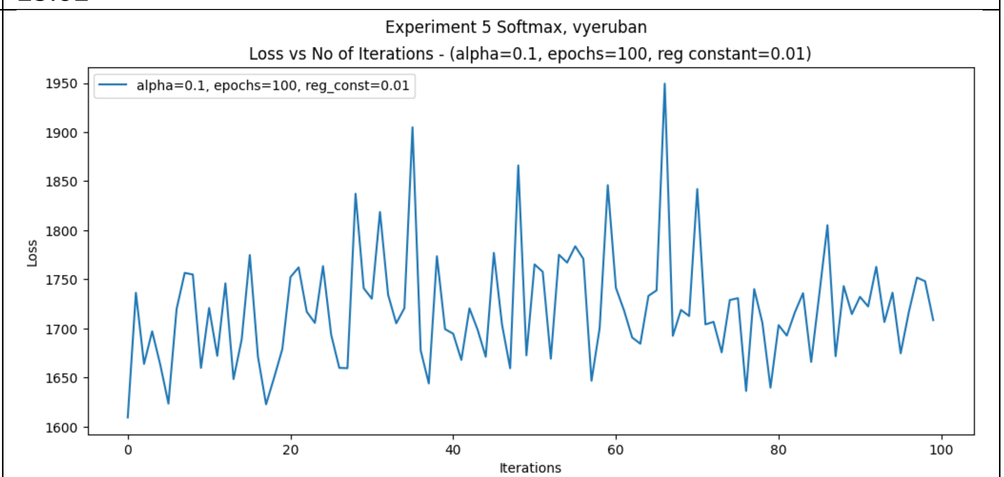
Training and validation accuracy vs number of iterations plot	

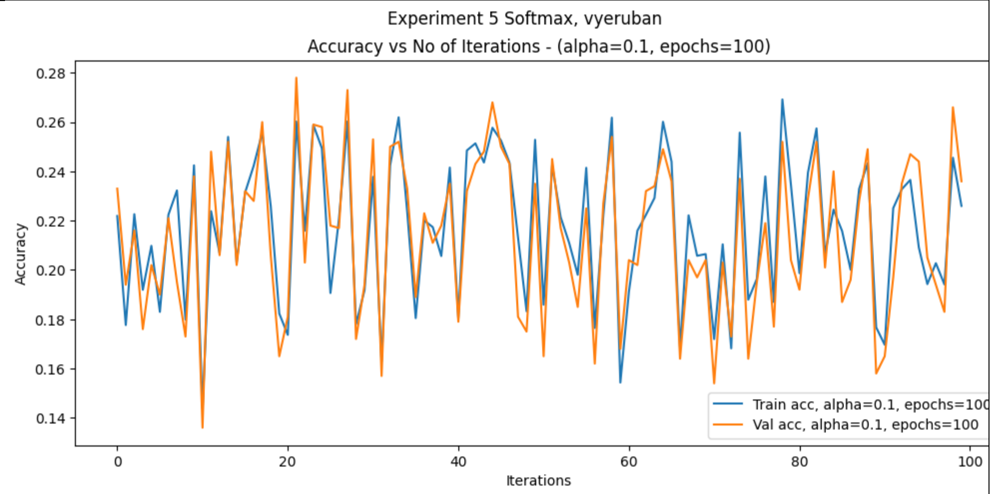
Experiment #	3
Hyperparameters	Learning rate = 0.01, epoch = 200, regularization constant = 0.01
Training accuracy	24.14
Validation accuracy	24.09
Test accuracy	22.48
Loss vs number of iterations plot using matplotlib	

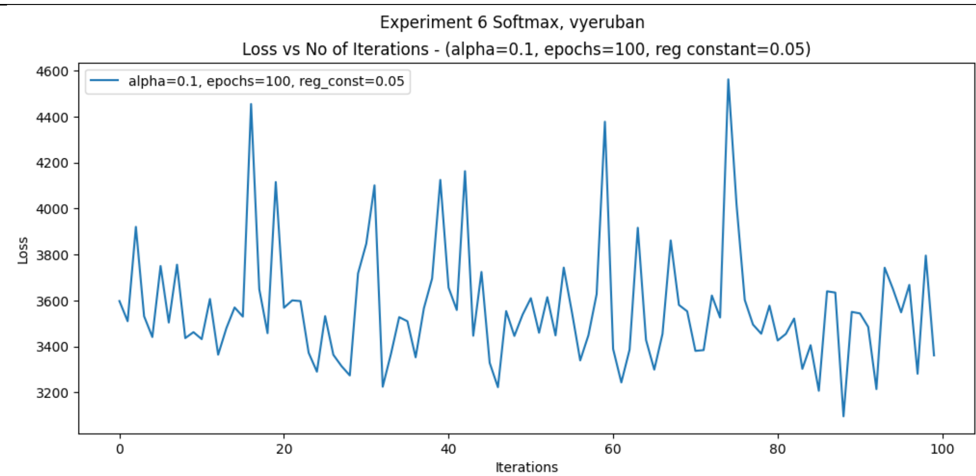
Training and validation accuracy vs number of iterations plot	

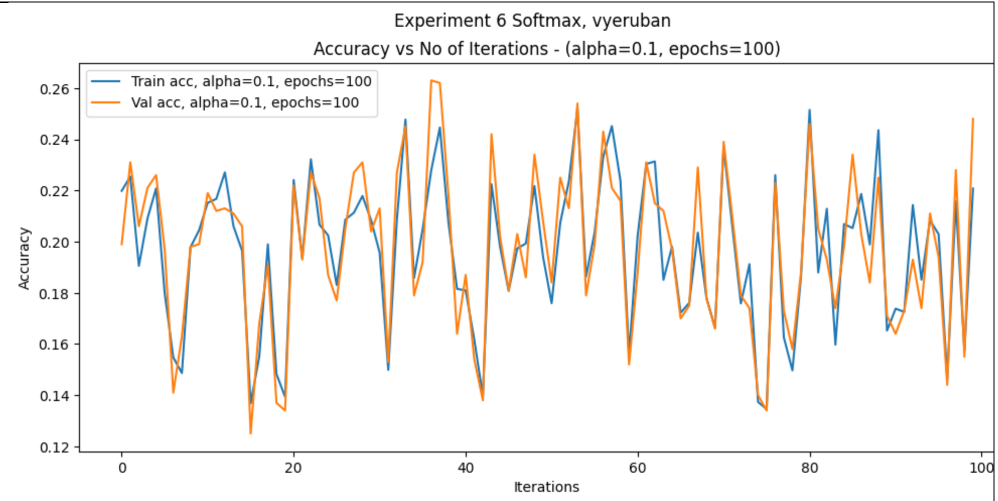
Experiment #	4
Hyperparameters	Learning rate = 0.01, epoch = 200, Regularization constant = 0.05
Training accuracy	23.72
Validation accuracy	21.3
Test accuracy	22.54
Loss vs number of iterations plot using matplotlib	

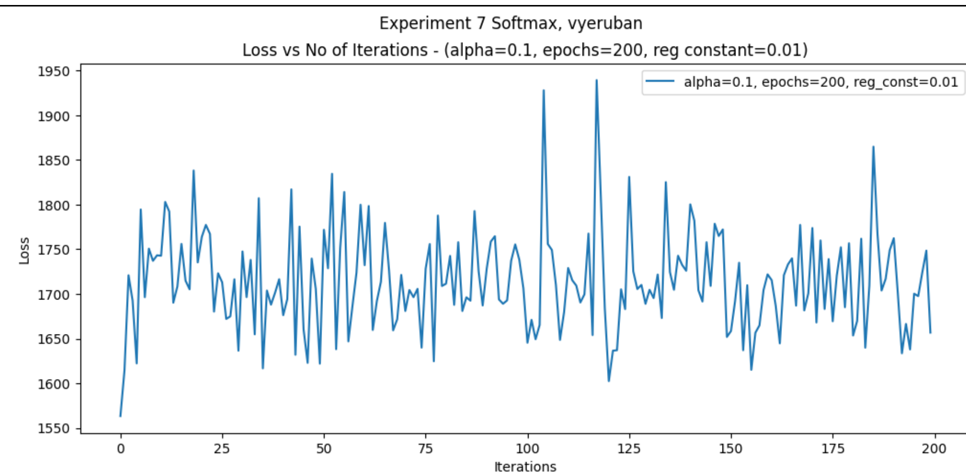
Training and validation accuracy vs number of iterations plot	

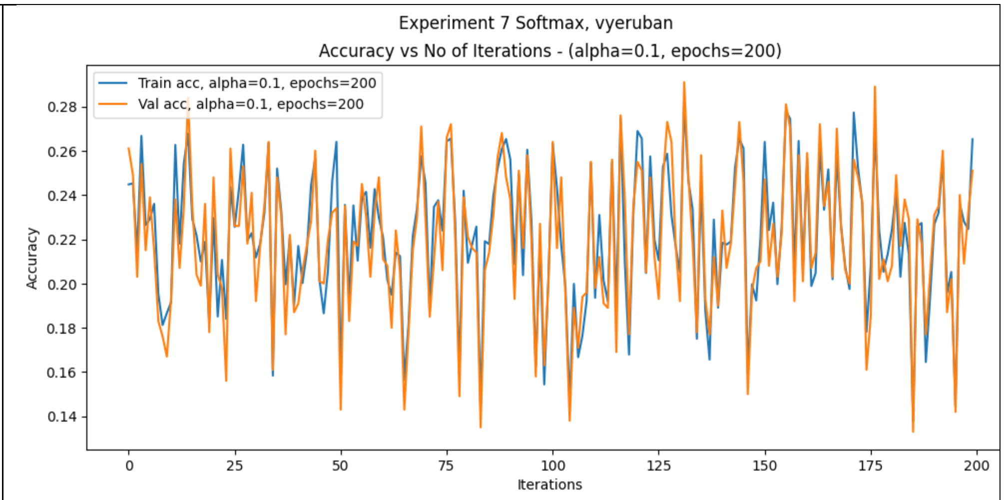
Experiment #	5
Hyperparameters	Learning rate = 0.1, epoch = 100, regularization constant = 0.01
Training accuracy	22.6
Validation accuracy	23.59
Test accuracy	23.02
Loss vs number of iterations plot using matplotlib	

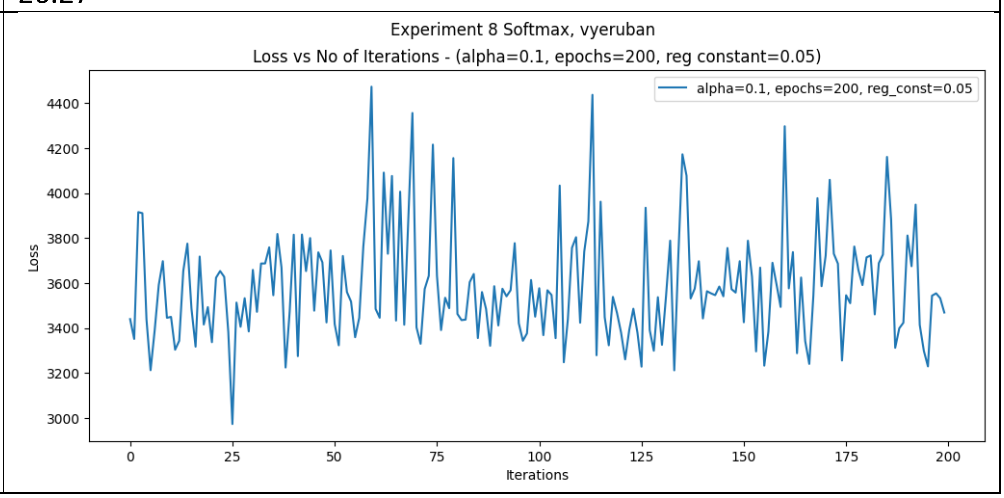
Training and validation accuracy vs number of iterations plot	

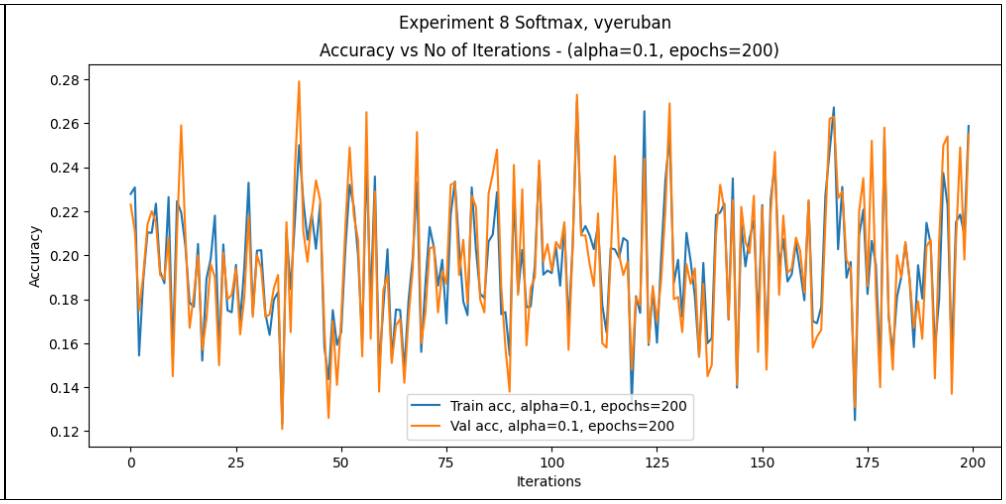
Experiment #	6
Hyperparameters	Learning rate = 0.1, epoch = 100, regularization constant = 0.05
Training accuracy	22.07
Validation accuracy	24.8
Test accuracy	21.6
Loss vs number of iterations plot using matplotlib	

Training and validation accuracy vs number of iterations plot	

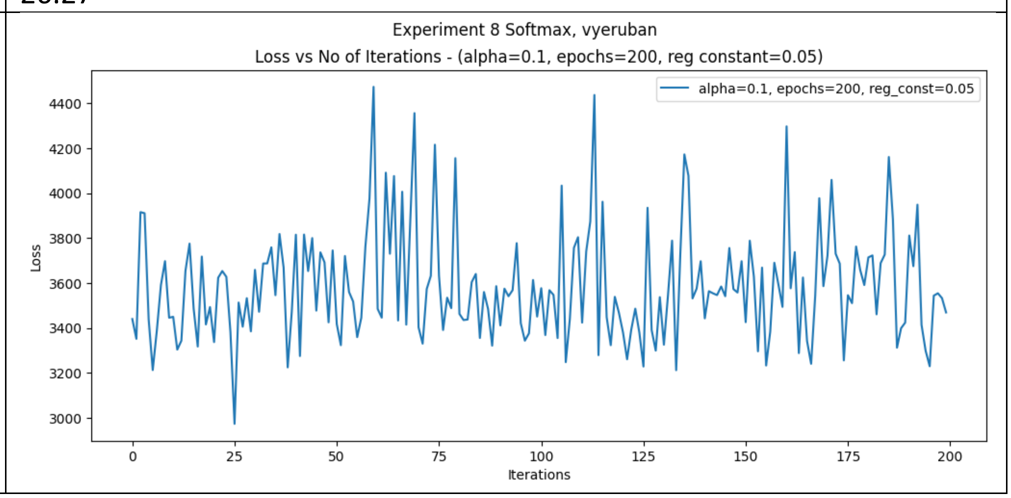
Experiment #	7
Hyperparameters	Learning rate = 0.1, epoch = 200, regularization constant = 0.01
Training accuracy	26.51
Validation accuracy	25.1
Test accuracy	25.31
Loss vs number of iterations plot using matplotlib	

Training and validation accuracy vs number of iterations plot	

Experiment #	8
Hyperparameters	Learning rate = 0.1, epoch = 200, regularization constant = 0.05
Training accuracy	25.85
Validation accuracy	25.5
Test accuracy	26.27
Loss vs number of iterations plot using matplotlib	

Training and validation accuracy vs number of iterations plot	

Hyperparameters of Kaggle model

Optimal hyperparameters	Learning rate = 0.1, epoch = 200, regularization constant = 0.05
Training accuracy	25.85
Validation accuracy	25.5
Test accuracy	26.27
Loss vs number of iterations plot	

Training and validation accuracy vs number of iterations plot