Logistic and Softmax Regression for Binary and Multi-class Classification

Shubham Chaudhary
UCSD
shchaudh@ucsd.edu

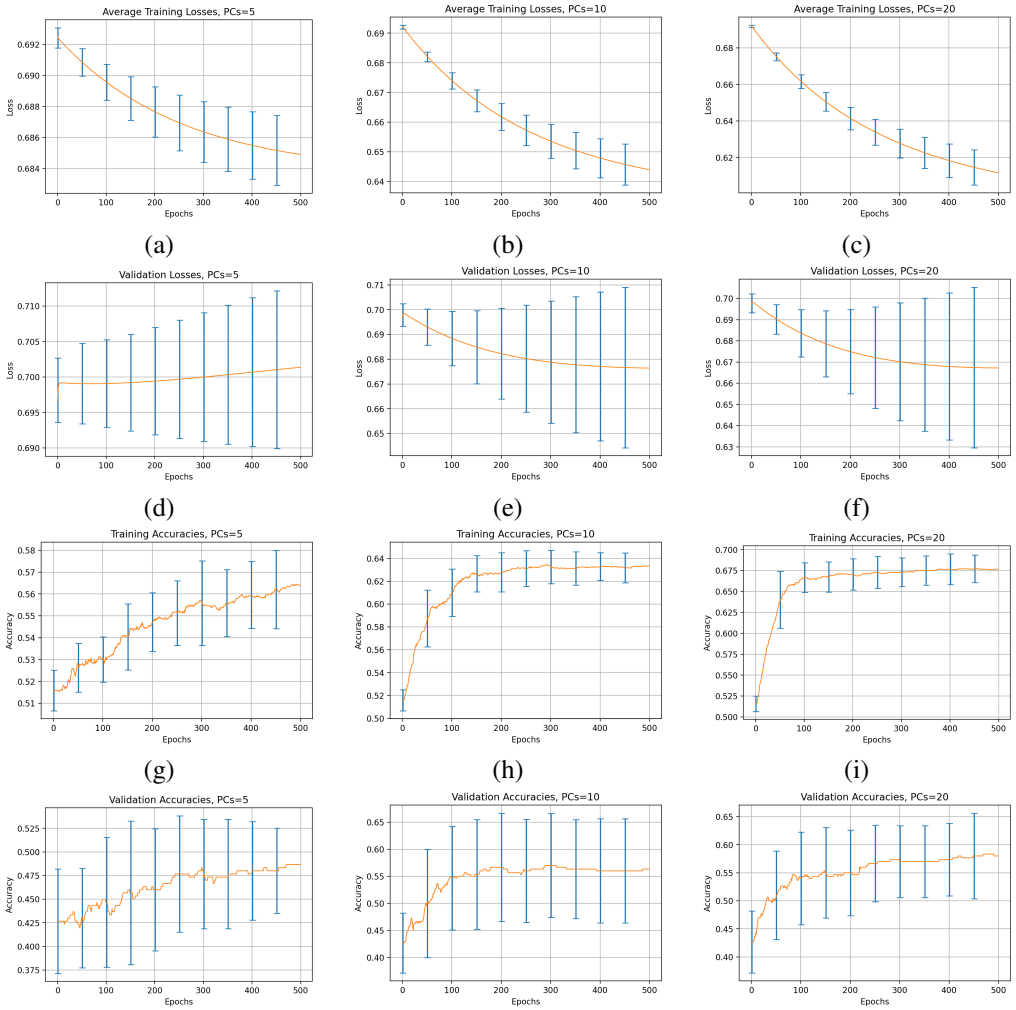
Allen Zeng
UCSD
azeng@ucsd.edu

1 Logistic Regression for Binary Classification

1.1 Convertible vs Minivan on the resized dataset

Presented in figure 1, are the statistics of Logistic regression model trained over the resized data of Convertibles and Minivans. Out of 10 folds, we observed that the validation error goes down and then up, only for some of the folds, that is why we have presented the statistics averaged over all 10-folds.

The poor accuracy over this dataset can be explained by observing the variance of the images in the dataset. In the figure 2 we have presented the first 4 Principal components of the image dataset. Observe that, due to different alignment of cars in the dataset, the principal components are fuzzy, specially at the edges of the cars which implies the lack of variance between the cars and surroundings at the borders.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Avg. Test Accuracy, PC=5: 0.60

Avg. Test Accuracy, PC=10: 0.553

Avg. Test Accuracy, PC=20: 0.566

Figure 1: Training losses, validation losses, training accuracies and validation accuracies. The average test accuracy after 10-fold cross validation is listed below the corresponding validation accuracies. Each column corresponds to 5, 10, and 20 principal components used, respectively. Training was performed using batch gradient descent.

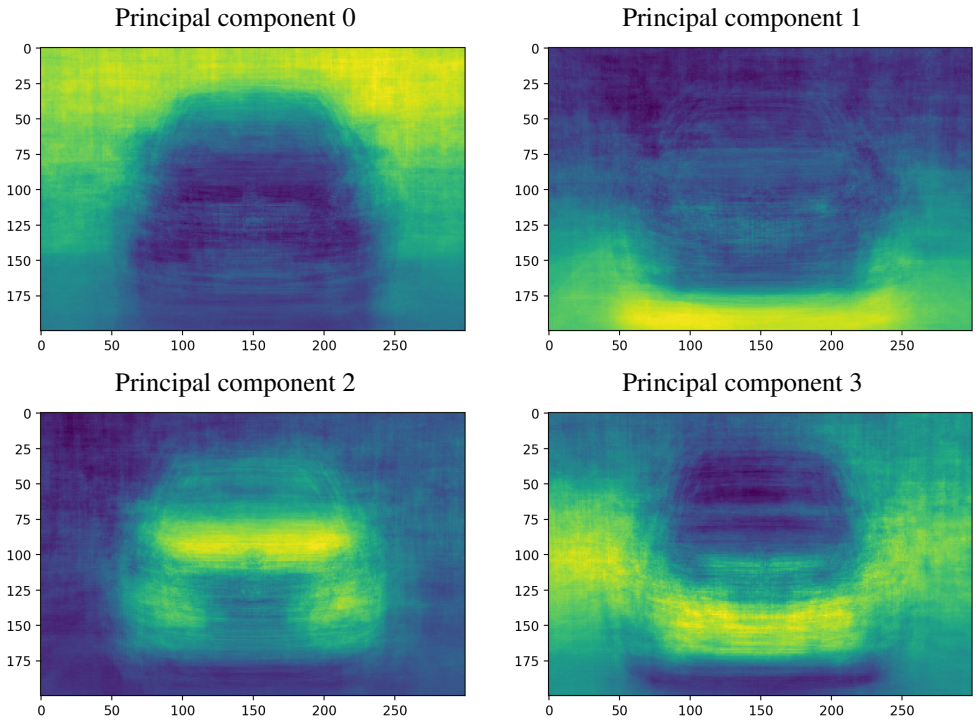


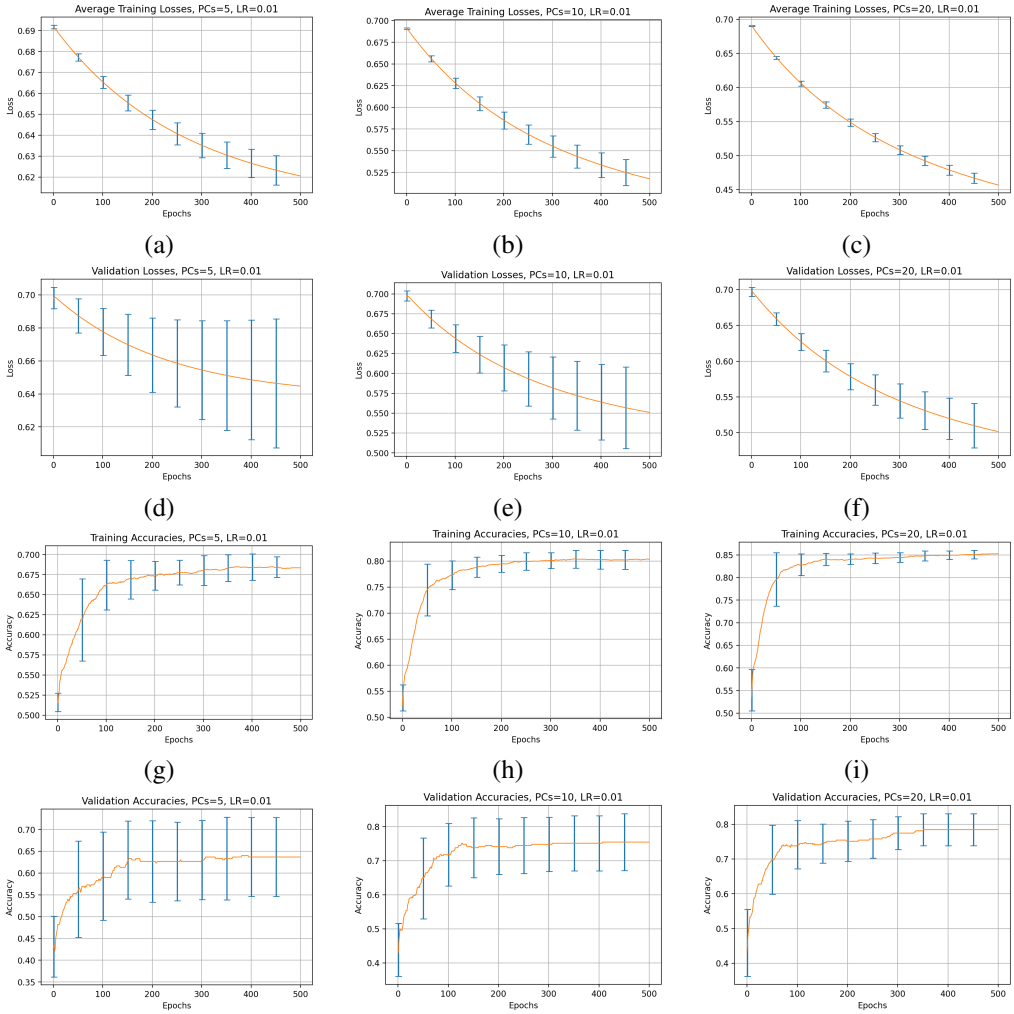
Figure 2: Images of top four principal components for resized images

1.2 Convertible vs Minivan on the aligned dataset

In order to figure out the correct set of hyper-parameters we performed a grid search over the set of learning rate(LR) and the number of Principal Components(PC). In the following table we can see how the test accuracy (averaged over 10-folds) varies with change in learning rate and number of Principal Components.

LR\PC	5	10	15	20	40
0.001	0.537	0.644	0.691	0.701	0.724
0.005	0.627	0.741	0.748	0.798	0.795
0.01	0.664	0.734	0.758	0.808	0.798
0.05	0.558	0.764	0.778	0.791	0.808
0.1	0.418	0.509	0.687	0.727	0.788

It can be observed that increasing the number of PCA components leads to an increase in accuracy but this increase is negligible for more than 20 components. Also, from this grid search we pick 0.01 as the optimal learning rate, 0.001 can be considered too low and 0.1 too high.
Using learning rate 0.01 we observed the learning behavior demonstrated in figure 3



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Avg. Test Accuracy, PC=5: 0.664(0.093)

Avg. Test Accuracy, PC=10: 0.724(0.095)

Avg. Test Accuracy, PC=20: 0.808(0.058)

Figure 3: Training losses, validation losses, training accuracies and validation accuracies. The average test accuracy after 10-fold cross validation is listed below the corresponding validation accuracies. Each column corresponds to 5, 10, and 20 principal components used, respectively. Training was performed using batch gradient descent.

Figure 4 demonstrates the first 4 principal components of the aligned image dataset. We can observe that these components, when visualized as images are much sharper(i.e. higher in contrasts) as compared to those of resized dataset.

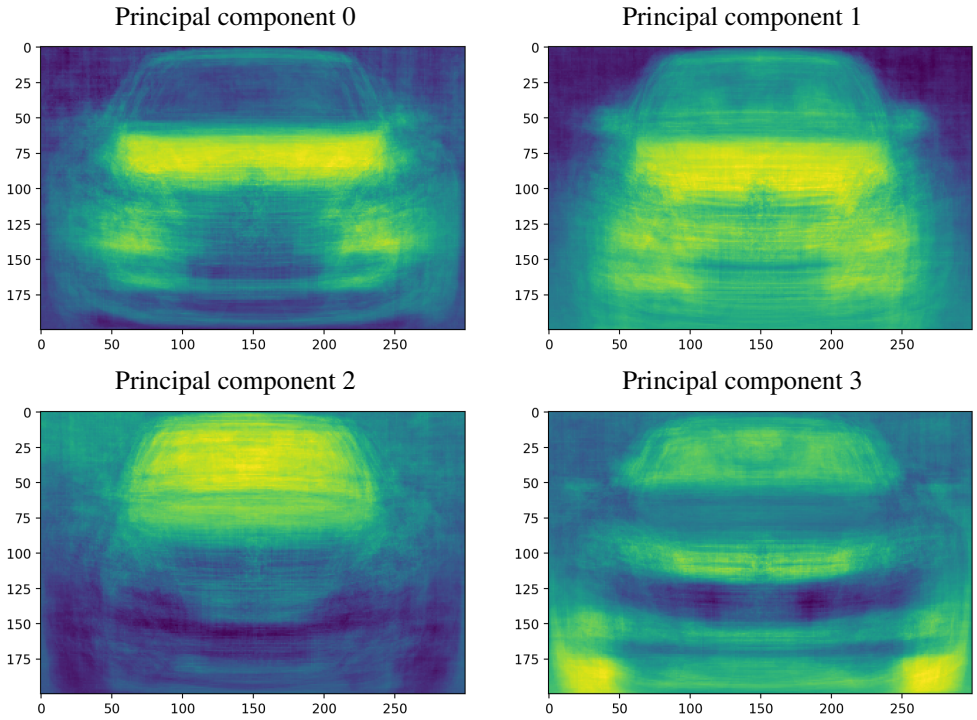


Figure 4: Images of top four principal components for aligned images

Figure 5 presents the change in training error for different learning rates. The learning rate of 0.1 is too high and we can observe that the training error oscillates very rapidly between 2 points. The learning rate of 0.001 is very low and loss corresponding to this rate remain higher that the loss that corresponds to 0.01, which is the suitable learning rate.

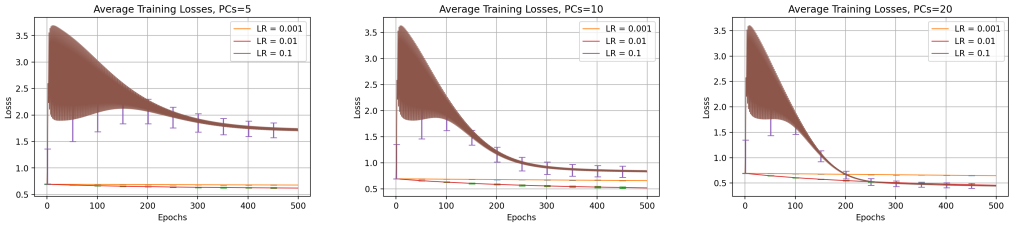
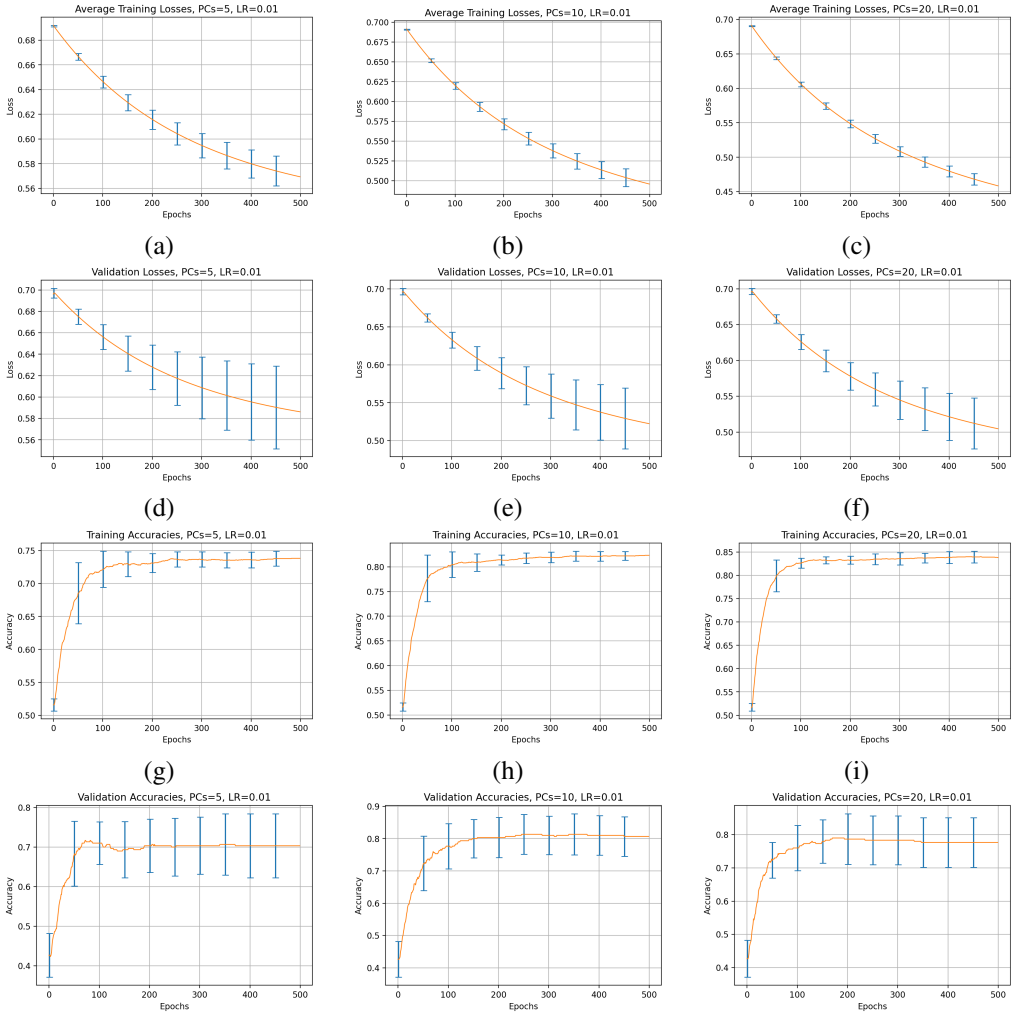


Figure 5: Training error for different learning rates

1.3 Sedan vs Pickup on the aligned dataset

Figure 6 presents the statistics of logistic regression model trained on the aligned dataset of Sedans and Pickups. The accuracy of the model over the dataset of Sedans and Pickups is better than the accuracy over the dataset of Convertibles and Minivans. The possible reason could be that the dataset of Sendans and Pickups is more separable. Since most of the images in the dataset present a front view of the cars, the difference in visual appearance from front of Sedans and Pickups could be the factor that leads to better accuracy.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Avg. Test Accuracy, PC=5: 0.717

Avg. Test Accuracy, PC=10: 0.813

Avg. Test Accuracy, PC=20: 0.770

Figure 6: Training losses, validation losses, training accuracies and validation accuracies. The average test accuracy after 10-fold cross validation is listed below the corresponding validation accuracies. Each column corresponds to 5, 10, and 20 principal components used, respectively. Training was performed using batch gradient descent.

2 Softmax Regression for Multi-class Classification

We perform multi-class classification on all four categories in our dataset. We follow their mathematical convention and summarize the equations here. We use the softmax activation function so

that the model outputs class probabilities for each input x^n. Here, y_k^n represents the probability that x^n belongs to class k.

$$y_k^n = \frac{\exp(w_k^\top x^n)}{\sum_{k'} exp(w_{k'}^\top x^n)}$$

Furthermore, we use multi-class cross-entropy as our loss function. This equation, and its gradient with respect to the model weights, is restated here.

$$E = -\sum_n \sum_{k=1}^{c} t_k^n \ln y_k^n$$
$$-\frac{\partial E^n(w)}{\partial w_{jk}} = (t_k^n - y_k^n) x_j^n$$

The equations here can be efficiently computed using matrix operations by using a one-hot encoding of the target labels t. The matrix t has elements t_k^n equal to one if the n-th input is from class k and 0 otherwise. To accommodate this change, there are separate model weights w_k for each of the k classes. After the model performs inference and outputs class probabilities y_k^n for each input x^n, the class with the highest probability is chosen as the predicted class for that input.

Similar to the binary classification procedure, we perform 10-fold cross validation in training our model using the batch gradient descent (BGD) procedure. During training, we used a learning rate of 0.01. The plots for the training losses, validation losses, training accuracies, validation accuracies, and test data confusion matrices are shown in figure 7. We tested multi-class classification using different numbers of PCA components: 10, 20, and 40. And, we attained test accuracies of 0.484, 0.524, 0.578, respectively for each model.

As more principal components are used, the average test accuracy improves. Furthermore, the spread of the error bars in all losses and accuracies are generally tighter with more principal components.

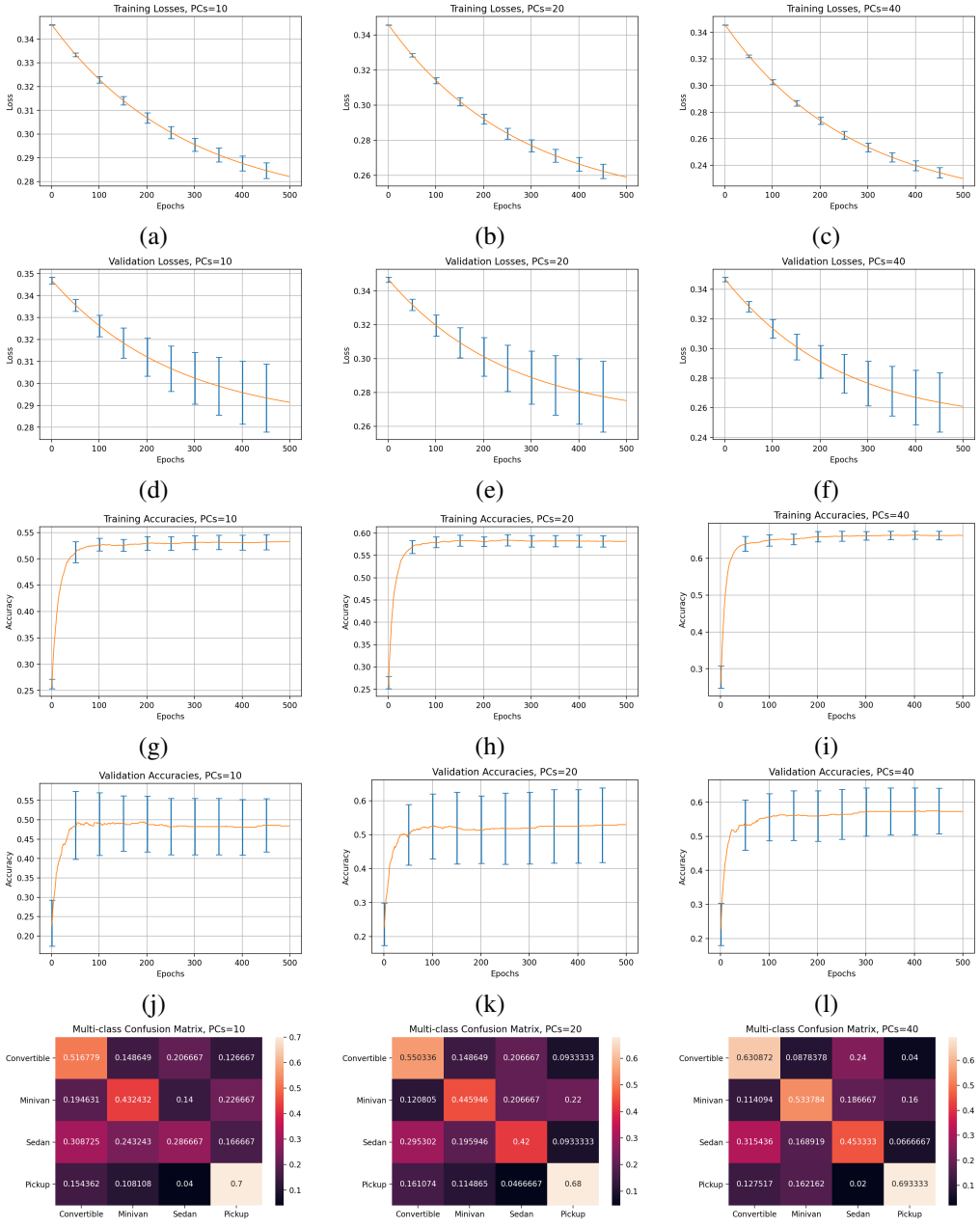
We also implemented stochastic gradient descent (SGD) for this multi-class classification problem. Using the same set of principal components (10, 20, and 40) and a learning rate of 0.01 again, we trained our new model with SGD and compared it with the model trained by BGD. In figure 8, we show plots of this comparison in training loss, validation loss, training accuray, validation accuracy. Respective to the set of principal components, we got average test accuracies of 0.484, 0.524, 0.578 for BGD and 0.481, 0.526, 0.575 for SGD.

From the results comparing BGD with SGD, they both converge to approximately the same accuracy on the test data. The training loss and validation loss were approximately the same over all epochs for both SGD and BGD. This implies that the network learned at similar rates for both gradient descent procedures.

Similarly, the training and validation accuracies between SGD and BGD were close over all epochs. However, the BGD accuracy curves are "smoother" while the SGD accuracy curves are "noisier." This is because BGD calculates a single large gradient step per epoch, while SGD takes N number of small gradient steps per epoch, where N is the number of training examples. Nonetheless, both gradient descent methods converge to approximately the same accuracy on the test data.

In both BGD and SGD, as more principal components are used, the average test accuracy improves. In both cases, the spread of the error bars in all losses and accuracies are generally tighter with more principal components.

In figure 9, we show the visualized weights corresponding to each car type in the multi-class dataset. To generate each of these images, we scale the principal components according to the class weights, and sum the weighed principal components. In this visualization, we can see that each image resembles its corresponding vehicle class. We can see that the convertible and sedan class, typically appear wider relative to their heights than the minivan and pickup classes, which are more square. Convertibles generally have smaller windshields, pickups usually have bodies further off the ground, etc. These images characterize what the typical characteristics of each class are. The closer that an input image is to one of these four points in the principal component space, the more likely the input image belongs to the corresponding class.

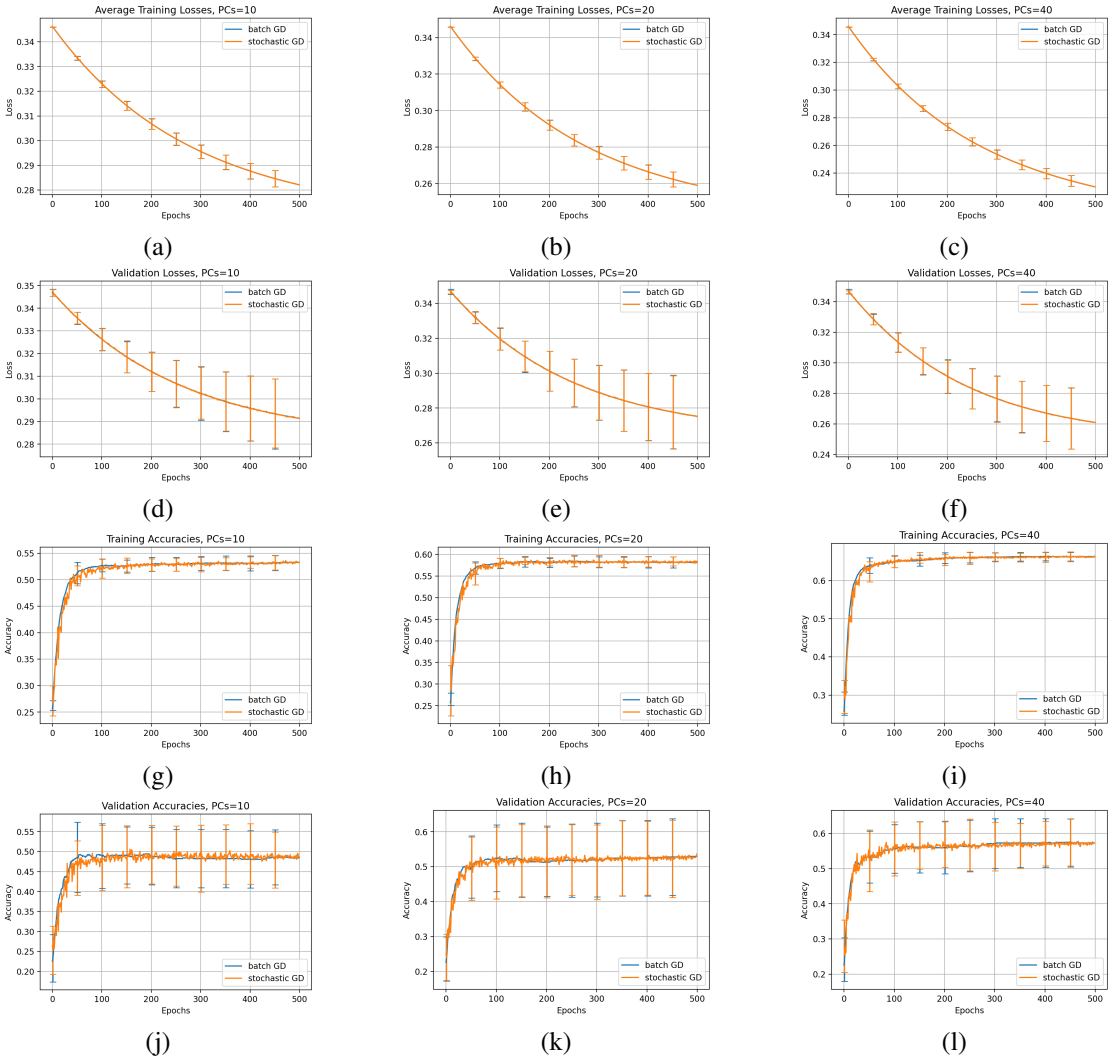


Avg. Test Accuracy, PC=10: 0.484

Avg. Test Accuracy, PC=20: 0.524

Avg. Test Accuracy, PC=40: 0.578

Figure 7: Training losses, validation losses, training accuracies, validation accuracies, and test data confusion matrices. The average test accuracy after 10-fold cross validation is listed below the corresponding confusion matrix. Each column corresponds to 10, 20, and 40 principal components used, respectively. Training was performed using batch gradient descent.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

BGD Avg. Test Accuracy, PC=10: 0.484

BGD Avg. Test Accuracy, PC=20: 0.524

BGD Avg. Test Accuracy, PC=40: 0.578

SGD Avg. Test Accuracy, PC=10: 0.481

SGD Avg. Test Accuracy, PC=20: 0.526

SGD Avg. Test Accuracy, PC=40: 0.575

Figure 8: Batch vs. Stochastic Gradient Descent on multi-class classification.

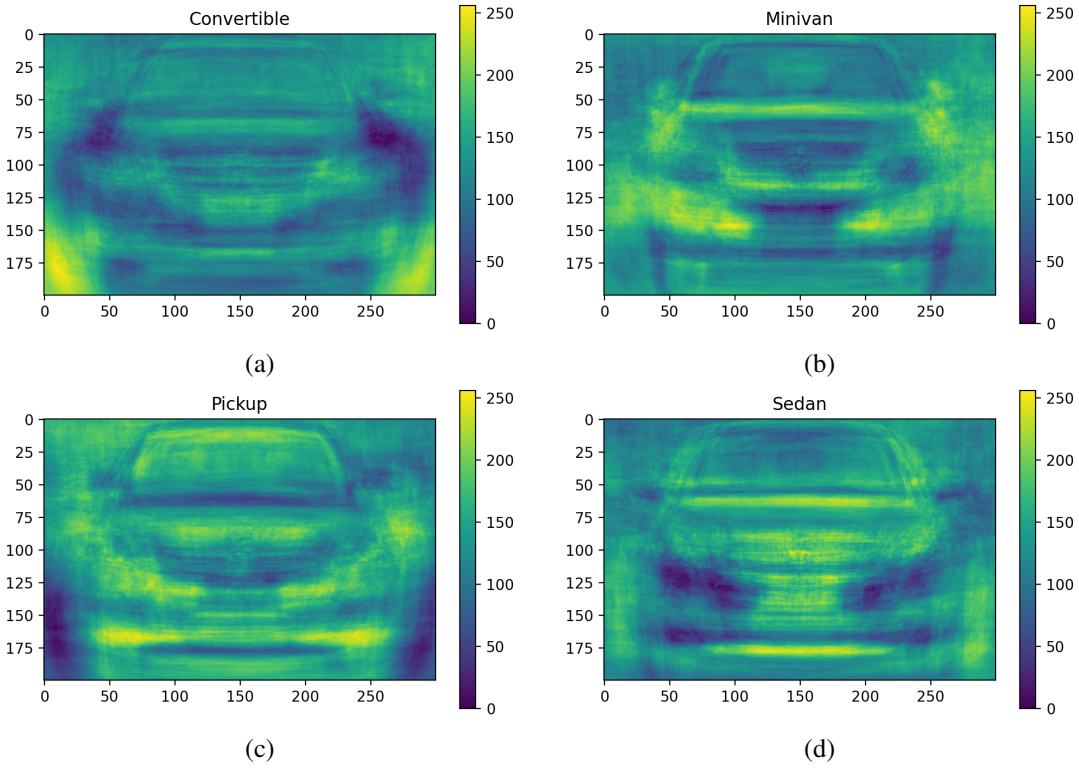


Figure 9: Visualized Weights of a model trained with 40 principal components.

3 Multi-class Classification on an Imbalanced Dataset

We used softmax regression to classify the imbalanced dataset [1]. Initially, we used the same batch gradient procedure, learning rate, and cross-validation as described in 2 and got a poorer test accuracy compared to the balanced "Aligned" dataset.

Then, we implemented a new "balanced" batch gradient update method using undersampling. During the training procedure while gradients are being calculated, we deliberately undersample the over-represented classes. Specifically, we first find the class with the minimum number of examples in the current (folded) dataset. Using that minimum number n, we randomly choose n examples each from all of the classes. All of the classes except the minimum-numbered class gets undersampled. After picking these examples, the gradient calculation and the rest of the training loop proceeds as previously.

Note that, however, we kept the cross-entropy loss and accuracy calculations the same in both models. The metrics are evaluated on the entire respective datasets. These plots are shown in figure 10.

Using this undersampling method, and even keeping the learning rate the same as before, the new model converged much more quickly and achieved a better average test accuracy than the old model. The average test accuracy for the standard model is 0.549, and for the undersampling model is 0.571.

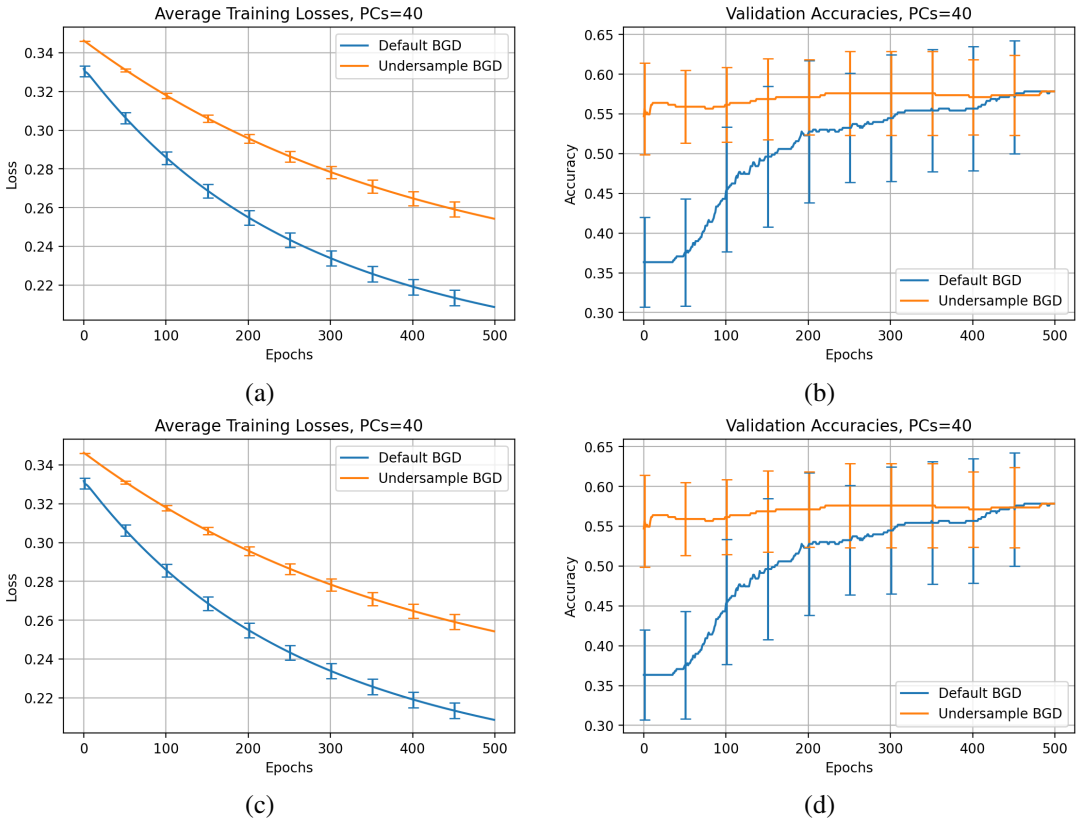


Figure 10: Standard batch gradient descent vs. undersampling batch gradient descent run on the imbalanced dataset. The average test accuracy for the standard model is 0.549, and for the undersampling model is 0.571.

4 Individual Contributions

Shubham and Allen scheduled meetings and worked on the core code simultaneously. Once the training procedure and cross validation procedures were in place, we were able to run the different experiments individually and record them into this report.

Acknowledgments

We used the code written by Divyanshu Mund to perform PCA and to load in the datasets.

References

[1] UCSD CSE 251B Staff (2021) *Programming Assignment 1.*, pp. 3-10. San Diego, CA.

[2] Yang, L., Luo, P., Loy, C. C., & Tang, X. (2015) *A Large-Scale Car Dataset for Fine-Grained Categorization and Verification.* arXiv 1506.08959 [cs.CV].