6.1

	λ	Train Accuracy	Validation Accuracy	Test Accuracy
0	0.0	0.966182	0.967130	0.961871
1	2.0	0.972925	0.974505	0.969665
2	4.0	0.973609	0.974505	0.970718
3	6.0	0.966709	0.967341	0.967348
4	8.0	0.955041	0.958281	0.955551
5	10.0	0.941056	0.943321	0.943965

Best λ according to validation set: 2.0

Test accuracy of the best model: 0.9696650516115441

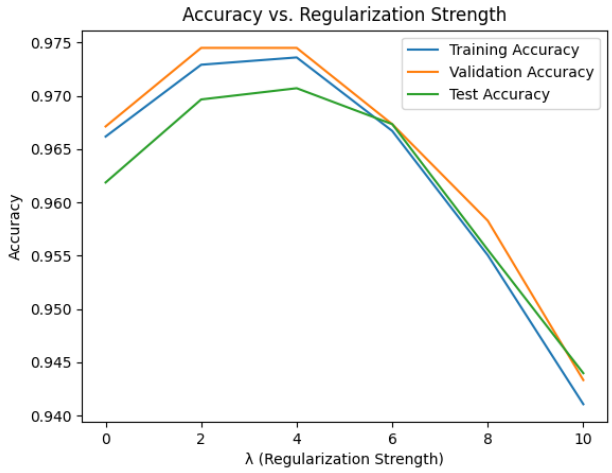
Worse λ according to validation set: 10.0

Test accuracy of the worse model: 0.9439646092268801

But if we will look on the table we can see that another best model is **λ** = 4. according to the validation set, but **λ = 4.** have better test Accuracy and better train Accuracay then **λ** = 2, that mean that our choosing model method(out from the validation set) isn't good enough to decide who is the better, because we might choose **λ** = 2 instead of **λ** = 4.

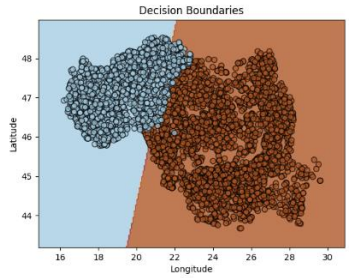
6.2

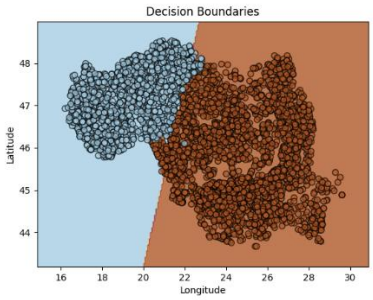
1.



We can see here that **λ** = 4 is the best model and not **λ** = 2 but we can clearly see here that **λ** = 2 and **λ** = 4 score the same thing in the validation set.

2. worse model **λ** = 10 best model **λ** = 4



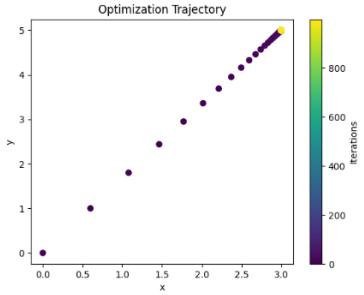


A smaller value of λ allows the algorithm to fit the training data more closely and results in less regularization a larger value of λ encourages the model to generalize better by penalizing large coefficients. This helps prevent overfitting, making the model more robust and better suited to handle unseen data.

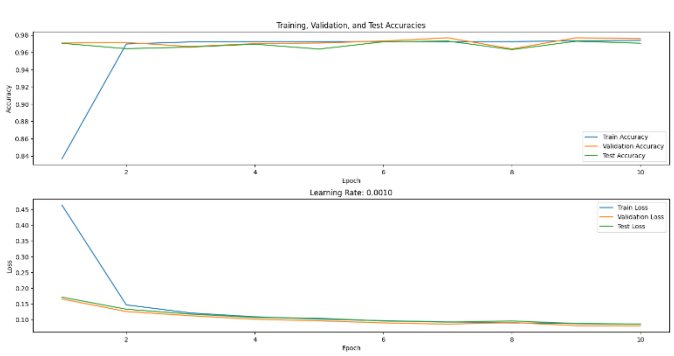
7

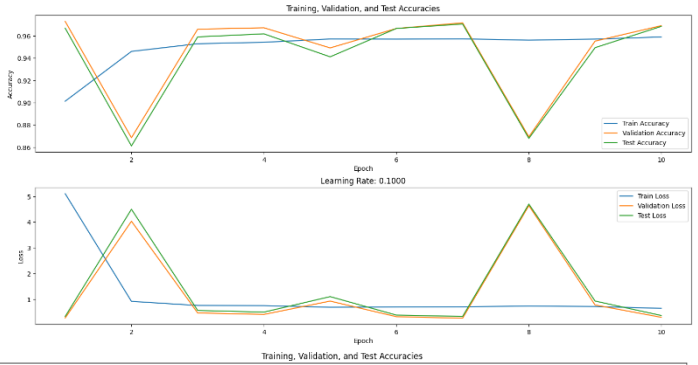
1.

The point we achieved is [3. 5.] after 1000 iterations

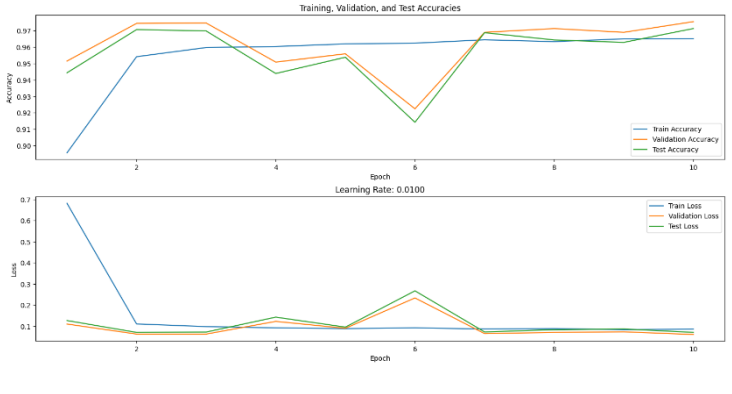


9



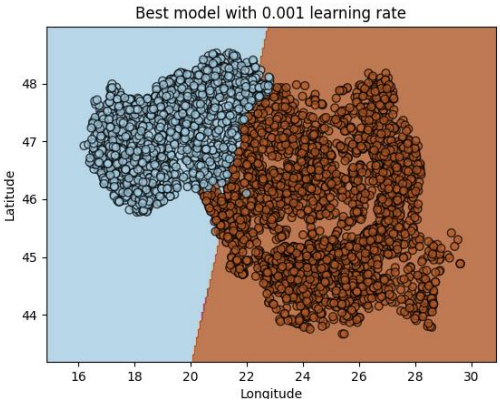


We can see that if we decrees our training rate we can achieve a better model, but we can also see that there a spikes in our graph, those spikes indicate that in our learning process , we did a

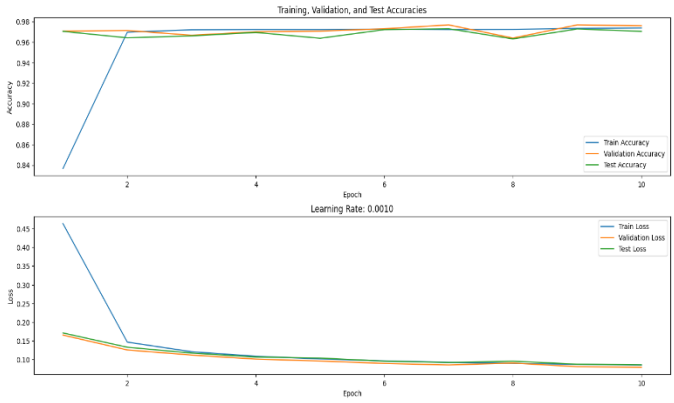


"big" step and skipped the minimum, what led to the spike. In our model with the small learning rate the risk of skipping the min is much smaller

9.3.1 We can see that the best model(according to the validation accuracy) in the 10^{th} epoch is the model with learning rate of 0.1 or 0.001 with 0.97%, we will visualize 0.001



9.3.2 On one hand, it is evident that the training, validation, and test losses closely align, starting from epoch number 2. This alignment suggests that our model demonstrates effective generalization from the training set. On the other hand, despite their visual similarity, there are instances where an increase in training set accuracy does not correspond to a similar improvement in validation and test accuracies. This observation suggests that our model struggles to generalize well overall. One would expect that enhancements in the training set would reflect in both the validation and test sets, but this is not consistently observed. Consequently, it can be concluded that our model lacks effective generalization.

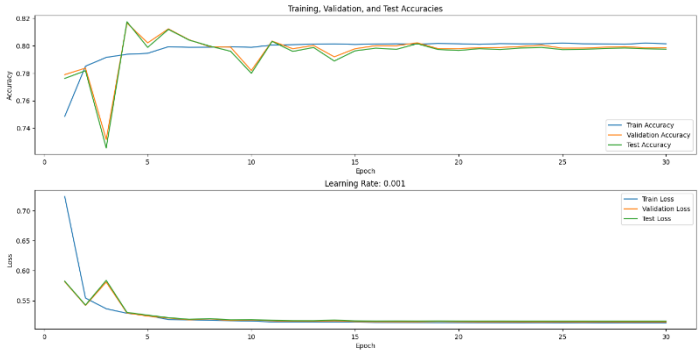


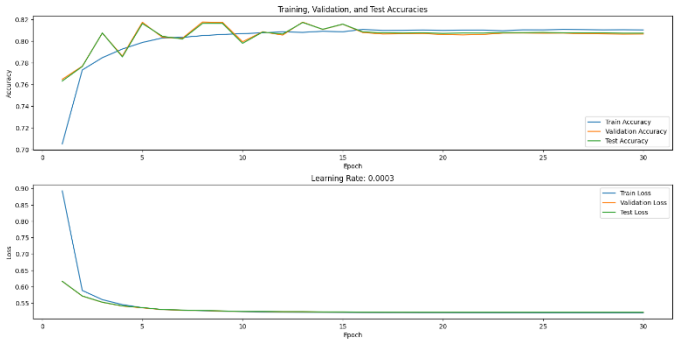
9.3.3

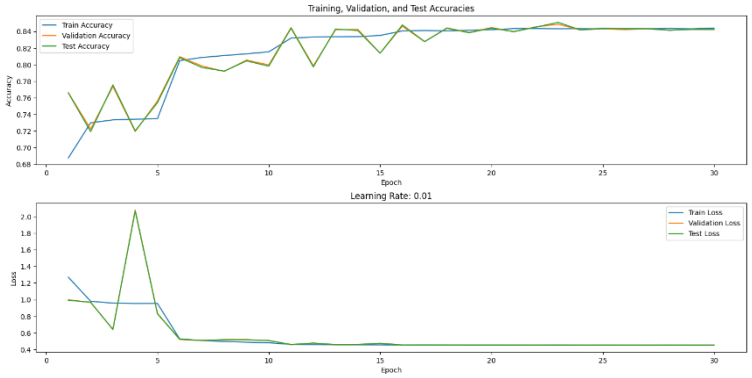
We observed that the Riged Regression model and the Logistic Regression model yields identical accuracy scores for the test, validation, and training datasets. This consistency arises from the fact that our Ridge Regression model provides us with an optimal model based on its λ parameter, and the selected λ aligns closely with the one yielding the best results (while not exhaustively checking all possible λ values). This λ parameter plays a crucial role in normalizing our weights. Interestingly, our LR model exhibits proficiency in autonomously learning and converging towards the optimal weights, ultimately arriving at the same weights as those obtained by our Ridge Regression model. An important distinction lies in how our models normalize weights—Ridge Regression utilizes the predefined λ, while LR achieves it through iterative learning and gradient descent. One may also suggest that our LG model is a little better than the RR model, although they have the same loss, this is due the fact we don't use L2 in LR , what make him less sensitive to the anomaly's what will make him a bit better model.

9.4

Let's view the different models:



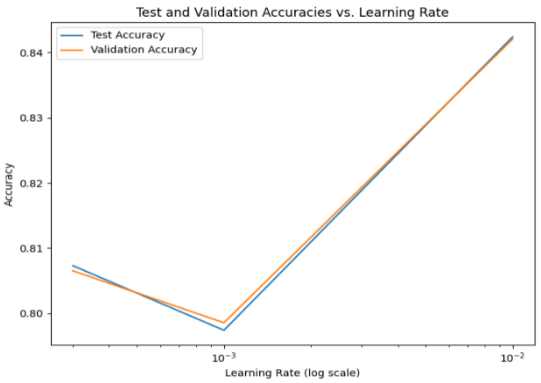




in our model, we employ multiclass sets and incorporate a decay parameter. Our most effective model utilizes a learning rate of 0.01. This improved performance is likely due to the use of the decay parameter, which gradually reduces the learning rate. Despite this reduction, the model retains the ability to make significant adjustments early in the training process. This capability helps the model approach the minimum value of the weight vector (W) more efficiently.

9.4.1

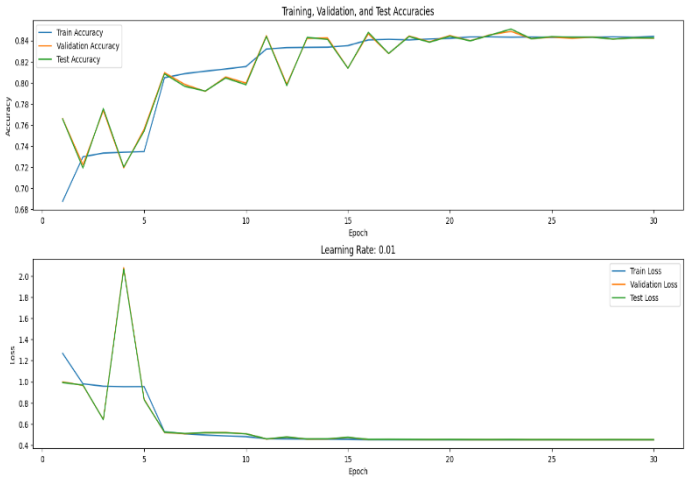
we found that the most effective model utilized a learning rate of 0.01. This model achieved the highest test accuracy of 0.842, outperforming other models with different learning rates. The superior performance of this model can be attributed to the use of decay parameters. At the beginning of the training process, our model took larger steps towards the minimum of the loss function due to the higher initial learning rate. This approach allowed the model to quickly converge towards the global minimum of the function. As the training progressed, the learning rate gradually decreased due to the decay parameter.



This reduction in learning rate allowed the model to make smaller, more precise adjustments, thereby fine-tuning its approach towards the global minimum. In contrast, models with a lower initial learning rate took smaller steps from the outset. While this approach allowed for precision, it also increased the likelihood of the models getting stuck in local minima, resulting in suboptimal performance. Therefore, the use of a decay parameter in conjunction with an initially high learning rate proved to be an effective strategy for our machine learning model. It combined the benefits of

rapid convergence in the early stages of training with the precision of slower, more deliberate adjustments in the later stages.

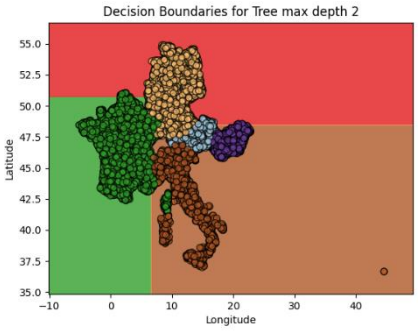
9.4.2



We can see that the best model accuracy on the validation set after 30 epoch is 0.842.

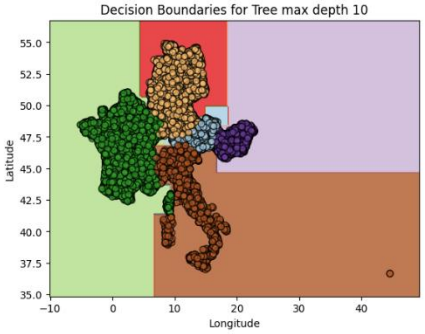
Upon evaluating our machine learning model, we found that the overall accuracy isn't as high as we would like. However, the model appears to generalize well, as evidenced by the nearly identical accuracies on the training, validation, and test sets. This consistency across different data sets suggests that our model is not overfitting and is capable of making reliable predictions on unseen data. The accuracy metric, being binary, is sensitive to even minor errors. This sensitivity is reflected in the occasional spikes observed in the test set accuracy. In other words, even a small misclassification can lead to a significant drop in accuracy because it is an all-or-nothing measure: a prediction is either correct (1) or incorrect (0). On the other hand, the loss metric, which is continuous rather than binary, provides a more nuanced view of the model's performance. We observed a significant spike in the loss at the beginning of the training process, indicating a large initial error. However, as the training progressed, the model was able to minimize this error, resulting in a much smaller loss. This trend suggests that our model is learning effectively from the training data and improving its predictions over time.

9.4.3



with a tree depth of 2, the training accuracy is 0.751, the test accuracy is 0.75, and the validation accuracy is 0.75. These results suggest that, in general, decision trees may not perform well for predictions on our dataset. This observation might indicate that our data closely aligns with the axes, and consequently, a decision tree might not be a suitable model for this particular dataset.

9.4.4



with a tree depth of 10, the training accuracy is 0.997, the test accuracy is 0.997, and the validation accuracy is 0.996. This suggests that the decision tree model performs exceptionally well, contrary to what was indicated in section 9.4.3. The model's predictive performance exceeds that of the model in Q2. Interestingly, even when the data does not align perfectly with the axis, the decision tree demonstrates remarkable precision. This could be attributed to the tree's high depth, allowing it to partition the data into smaller squares and make predictions with high accuracy, and handle well with anomaly's in contract to our model ,all this despite the data not aligning precisely with the axis.