**Dataset Preprocessing**

For normalization I used a zscore for all feature and experimented with scaling certain values from 0-1. I found the best results come from a flat zscore even though it introduced negative values into the data. Accuracy with zscore was about ~0.1 error points better than my other method.

I worked on preparing my data by adding new features and splitting the data to make it more suitable for modeling. First, I added new columns, such as fractional representations of the pickup and dropoff times (e.g., pickup\_month, pickup\_day, and pickup\_hour), to capture more granular information about the trip. I also created a pickup\_day\_of\_week feature to account for weekly patterns.

**Model Selection**

features\_model1 = ["pickup\_longitude", "pickup\_latitude", "dropoff\_longitude", "dropoff\_latitude"]

model1 = Sequential()

model1.add(LinearLayer(model1\_x\_train.shape[1], 4))

model1.add(ReLU())

model1.add(LinearLayer(4, 4))

model1.add(ReLU())

model1.add(LinearLayer(4, 1))

epochs\_model1 = 1000

learning\_rate\_model1 = 0.01

patience\_model1 = 3

Epoch 999, Loss: 0.5671, Val Loss: 0.5455

Saving model at epoch 999 with loss 0.5671

Test Loss (MSE): 0.6195

A graph with numbers and lines

AI-generated content may be incorrect.

model1 = Sequential()

model1.add(LinearLayer(model1\_x\_train.shape[1], 4))

model1.add(ReLU())

model1.add(LinearLayer(4, 4))

model1.add(ReLU())

model1.add(LinearLayer(4, 4))

model1.add(ReLU())

model1.add(LinearLayer(4, 1))

epochs\_model1 = 1000

learning\_rate\_model1 = 0.01

loss\_function\_model1 = MeanSquaredErrorLoss()

patience\_model1 = 3

Epoch 999, Loss: 0.5726, Val Loss: 0.5501

Saving model at epoch 999 with loss 0.5726

Test Loss (MSE): 0.6108

A graph with numbers and lines

AI-generated content may be incorrect.

features\_model2 = ["pickup\_longitude", "pickup\_latitude", "dropoff\_longitude", "dropoff\_latitude", "pickup\_month", "pickup\_day", "pickup\_hour", "dropoff\_month", "dropoff\_day", "dropoff\_hour"]

model2 = Sequential()

model2.add(LinearLayer(model2\_x\_train.shape[1], 4))

model2.add(ReLU())

model2.add(LinearLayer(4, 10))

model2.add(ReLU())

model2.add(LinearLayer(10, 1))

epochs\_model2 = 1000

learning\_rate\_model2 = 0.01

patience\_model2 = 3

Epoch 999, Loss: 0.6014, Val Loss: 0.5672

Saving model at epoch 999 with loss 0.6014

Test Loss (MSE): 0.6322

A graph with numbers and lines

AI-generated content may be incorrect.

model2 = Sequential()

model2.add(LinearLayer(model2\_x\_train.shape[1], 10))

model2.add(ReLU())

model2.add(LinearLayer(10, 10))

model2.add(ReLU())

model2.add(LinearLayer(10, 10))

model2.add(ReLU())

model2.add(LinearLayer(10, 1))

epochs\_model2 = 1000

learning\_rate\_model2 = 0.01

loss\_function\_model2 = MeanSquaredErrorLoss()

patience\_model2 = 3

Saving model at epoch 998 with loss 0.4638

Epoch 999, Loss: 0.4638, Val Loss: 0.4090

Saving model at epoch 999 with loss 0.4638

Test Loss (MSE): 0.4648

A graph with numbers and lines

AI-generated content may be incorrect.

features\_model3 = ["pickup\_longitude", "pickup\_latitude", "dropoff\_longitude", "dropoff\_latitude", "pickup\_month", "pickup\_day", "pickup\_hour", "dropoff\_month", "dropoff\_day", "dropoff\_hour","pickup\_day\_of\_week", "dropoff\_day\_of\_week"]

model3 = Sequential()

model3.add(LinearLayer(model3\_x\_train.shape[1], 12))

model3.add(ReLU())

model3.add(LinearLayer(12, 12))

model3.add(ReLU())

model3.add(LinearLayer(12, 1))

epochs\_model3 = 1000

learning\_rate\_model3 = 0.01

patience\_model3 = 3

Epoch 999, Loss: 0.5260, Val Loss: 0.4445

Saving model at epoch 999 with loss 0.5260

Test Loss (MSE): 0.7539

A graph with numbers and lines

AI-generated content may be incorrect.

model3 = Sequential()

model3.add(LinearLayer(model3\_x\_train.shape[1], 12))

model3.add(ReLU())

model3.add(LinearLayer(12, 12))

model3.add(ReLU())

model3.add(LinearLayer(12, 12))

model3.add(ReLU())

model3.add(LinearLayer(12, 1))

epochs\_model3 = 1000

learning\_rate\_model3 = 0.01

loss\_function\_model3 = MeanSquaredErrorLoss()

best\_loss\_model3 = float("inf")

patience\_model3 = 3

Epoch 999, Loss: 0.4608, Val Loss: 0.4300

Saving model at epoch 999 with loss 0.4608

Test Loss (MSE): 0.5177

A graph with numbers and lines

AI-generated content may be incorrect.