Cyber Labs

Machine Learning Algorithm Library

Peddi Hrushikesh

IIT (ISM) Dhanbad

Project Report

Project Overview: -

The project includes 5 machine learning models fine-tuned for accuracy and efficiency namely Linear Regression, Polynomial Regression, Logistic Regression, K – Nearest Neighbours and Neural Networks.

Linear Regression: -

- The basic model of machine-learning which fits a straight line into the data has been arguably the hardest to implement for me as it is my first project in Python.
- The model uses train sets split into train and test sets for the model to train and test its accuracy on.
- The model's test and train data are Z-score normalized.
- The model ran through 4000 iterations before the cost started to converge.
- The model achieved a R2 score of 0.8411 on the test set.
- The biggest hurdle during the building of the set was deciding which of the parameters must be updated when, where and how to store the updated values.
- One time I initialised y_hat outside the gradient descent function and didn't update it in the gradient descent function which led to me having the same cost in every iteration.
- The Hyper parameters used are
    - Alpha = 0.001
    - Number of iterations = 4000

Polynomial Regression: -

- After implementing linear regression, polynomial regression has been kind of a breeze as it is very similar to that of linear regression.
- The model took about half a day to complete.
- The main problem I faced was to generate the degree terms, but my mentor gave me intuition about it and finally solved it.
- The model uses train sets split into train and test sets for the model to train and test its accuracy on.
- I thought to use zscore normalization for the data, but I couldn't fit the data well, thus I decided not to use the normalization.
- In this model L1 regularisation is used.
- The model ran through multiple iterations before the cost converged at around 1,00,000
- RMSE was used to assess the performance of the model.
- RMSE of train set is 2457.48 and RMSE of test set is 2878.81
- The cost function was regularized, and lambda was set to 4 and any further increase led to increase in RMSE value of both train and test sets.

- The Hyper parameters used are: -
  - Alpha = 7.0e-38
  - Number of iterations = 1,00,000
  - Lambda = 4

Logistic Regression: -

- Logistic regression steals the second spot for being the hardest to implement after linear regression due to previous experience.
- The model took about 3 days to complete.
- The problem that I faced the most while implementing Logistic Regression is keeping track the size of different arrays which often led to multiple errors during different computations.
- And it took me time to learn multiclass regression and to implement it.
- The inputs of the images were printed and normalized by dividing with 255.
- This model is the first one in which I implemented formatting in strings.
- The model was run using the binary cross entropy loss function and One-Hot encoding.
- The model is very computationally expensive and takes about 2 hours to train for 10,000 iterations
- The variables were very difficult to keep track of.
- The model is finally trained, and the weights and bias are output with train and test accuracy averaging around 95 %
- The hyper parameters used are
  - Alpha = 1.0e-4
  - Number of iterations = 1000

K – Nearest Neighbours: -

- The algorithm was the easiest to implement among all the other algorithms and it took around 2 hours to complete.
- Though the algorithm gets daunting and computationally expensive for larger data sets. It is a clean algorithm that can be used for both regression and classification.
- The data set used for the algorithm is linear_train and K is set to 4 which gives the best regression values.
- The model was not normalized.
- The algorithm has been run on 100 random examples of the linear_train set, and the mean absolute error turned out to be around 53.
- The implementation went very smooth, and no apparent problems have been faced.

Neural Networks: -

- The implementation of neural networks wasn't as smooth as expected as implementation of backward propagation for every single element is hard to keep track of.
- The biggest problem faced during the neural network implementation is the different arrays and their sizes which get very muffled in the head.
- Even though I was understanding the course well, I couldn't implement without deep learning algorithms like tensor.
- The neural network has been used for classification based on the classification_train data set.
- It is the first model in which I included classes and I had a lot of fun playing around with them.
- I tried a lot of stuff around with neural networks and none of them seemed to properly work.
- Finally, I tried different sigmoid function which uses hyperbolic tan function, this step turned well for my model.
- Hyperparameters used
  - Alpha = 1.0e-10
  - Num_iterations = 10000