**Report Summary**

After spending lot of time researching and investing on Kaggle, I felt Movie Dataset is quite interesting problem to tackle on. I have chosen the Movie dataset to predict the movie revenue based on certain features as the movie industry is quite volatile and it is hard to predict which movie will have real financial gain or loss. The idea behind the regression model is that it takes the multiple input features and predict certain real value which is in this case will be movie revenue that means how much this movie is going to make the turnover.

This dataset is inspired from below kaggle, as it simplified version of movie dataset.

<https://www.kaggle.com/kevinmariogerard/tmdb-movie-dataset/data>

Thought is to try and learn how much value machine learning can provide by building various machine learning models , now this my current assignment it is regression models using the kaggle example dataset provided such that I can learn various regression models as described in the assignment and also see whether we can predict and see new insights on movie industry when and what kind of situation one can predict revenue for the movies.

I was using below fields as an input and predict ‘revenue’ as an output

popularity

budget

runtime

genres

vote\_count

vote\_average

release\_year

As I have taken dataset from Kaggle, it has already 10866 records in the training dataset, which is good dataset to train and split for the test data also so I can see the right prediction giving by the various algorithms.

**Dataset loading:**

Removed certain variable which are not required as they do not add value for this task, such as id, imdb\_id, original\_title, cast, homepage, director, keywords, overview, production\_companies, release\_date, release\_year, budget\_adj and revenue\_adj

Out of them ‘genres’ and ‘release\_year’ are the two categorical and others are normalized real-valued variables.

What I observed is that ‘genres’ has pretty long feature mapping result as we can see below, which is 2096 categorical features out of single ‘genres’

|  |
| --- |
| Feature vector length for categorical features: 2096 Feature vector length for numerical features: 6 Total feature vector length: 2102 |

Technology:

Python 3: It is the main language used for machine learning algorithm implementation. Also used Numpy, Matplotlib as other libraries to work with collections and later one is for draw the graphs.

Spark ML: Installed and spark my machine and used Spark ML. This is main Machine learning library for getting the algorithms and implement, change various parameters. It allows Regression and Classification algorithms.

Jupyter Notebook: This is very good for visualizing all my work I am doing with the machine learning implementation. I can do everything here, imports, code, outputs and all the plotting graph output. It really a lot to speed up my work to understand and implement.

I also used various references to go through the examples, tutorials and especially I used Spark ML APIs and examples.

I first started understanding the Bike dataset implementation reference document to see the various examples provided and explained lot of information about how to build model, how to see error rate, tuning the model for performance. And I have built the Bike dataset implementation by referring to the examples and the documentation. This really helped a lot to understand also which made much easier for me to implement and find a new dataset on Kaggle and also implemented it with the help of Bike dataset implementation.

I also used some of the helper functions from the assignment Bike implementation example, such as get\_mapping, extract\_features, extract\_label, extract\_features\_dt, squared\_log\_error, abs\_error and squared\_error.

I have also used the data splitting for the model building which is 70 - 30 rule , where 70% split is the training data used to train the model and 30% data is the test data, which is used to test the train model. As per my understanding this test data acts as a validation data where algorithm usage it to do cross validation and prediction test.

I have built 3 algorithms Decision tree, Linear regression and Gradient boosted tree. Each one has set of tasks. Each algorithm has specific parameters to tune the model performance also also building the model so that we can see on plot how each algorithm works.

In Decision Tree algorithm, I have learned and implemented concept of Max depth and Max bin. Most important and interesting task is to build a plot to depict the behaviour of the model by changing those parameters and looking at the error rate how well is the algorithm model performing. Looking at error rate is the one of the way to optimize and see what would the best parameters would work for the given specific dataset and the algorithm. Decision Tree Log is the one shows the error rate values of various log functions described in the assignment such as Root Squared Error, Absolute Error and Mean Squared Error.

In Gradient boosted tree, I have learned and implemented concept of iterations, Max depth and Max bin. This is same as Decision tree with extra parameter iterations. This parameter is also quite good one to understand how the algorithm model gets impacted and how is the error rate on each selected value in the iteration list.

In Linear regression, I have learned and implemented concept of Intercept, Iterations, Step size, L1 Regularization and L2 Regularization. This one took some time to understand and implement it as quite new parameter concepts. There was no difference in error rate calculated for L1 and L2 Regularization for given various list of input parameter values, error rate was always same.

**Execution**

\* Installed Python 3 on my machine.

\* Installed Java 8 on my computer.

\* Install Spark Downloaded the latest version Spark

from http://spark.apache.org/downloads.html

Version: spark-2.3.0-bin-hadoop2.7

\* Install Jupyter Notebook

\* Configure following 3 System environment variables

export PYSPARK\_PYTHON = python3

export PYSPARK\_DRIVER\_PYTHON = jupyter

export PYSPARK\_DRIVER\_PYTHON\_OPTS = 'notebook'

\* In the command prompt terminal

1. Go to Spark installed location

2. Run pyspark

\* Opens Jupyter Home page in browser