

CS-GY 9233: Programming for Big Data Analytics **People’s Oscar (**<https://architgyl.github.io/>**)**

short lineSpring 2018, May 2nd, 2018

Under the Guidance of: By:

Professor Juan C Rodriguez Archit Goyal (ag6288)

Avaiyang Garg (ag6026)

Rajeev Joshi (rj1234)

**TABLE OF CONTENTS**

**S.No. TITLE PAGE NO.**

**1. Introduction 2**

**2. Objective 3**

**3. Technologies and Platform 4**

**4. Architecture 5**

**5. Data Preprocessing 6**

**6. Sentiment Analysis 9**

**7. Machine Learning 12  
  
 References 19**

**Chapter 1**

**INTRODUCTION**

Big data is informational collections that are so voluminous and complex that conventional information handling application programming are deficient to manage them. Enormous information challenges incorporate catching information, information stockpiling, information investigation, look, sharing, exchange, representation, questioning, refreshing, data security and information source.

The big data is made up of 3 V’s :

1. **Volume**: Big data first and foremost has to be “big,” and size in this case is measured as volume.
2. **Velocity**: The rapidly increasing speed at which new data is being created by technological advances, and the corresponding need for that data to be digested and analyzed in near real-time.
3. **Variety**: With increasing volume and velocity comes increasing variety. This third “V” describes the huge diversity of data types that healthcare organizations see every day.

In this project we are going to analyze the relationship between the datasets from the IMDB and the user Tweets during the particular year and then predict the winner in different genre movies. By analyzing the user tweet we are studying the behaviour and responses of the users about the particular movies, and with the IMDB dataset we are extracting the ratings of the movies and the user reviews and then clubbing these to predict with some degree, about who will win. We will also be creating a machine learning model to predict whether the movies will be a hit or not in the coming future.

**Chapter 2**

**OBJECTIVE**

We wanted to showcase the power of spark and its support for diverse use cases, so we built a Movie rating model using Twitter sentiment analysis as well as a Movie Rating Prediction Model using machine learning, for both of which we used IMDB datasets.  
  
Also all of the technologies we used are scalable and can be implemented on a Distributed Computing Cluster.

* Utilize the IMDB data set to generate Meaningful and Interesting Insights
* Create a movie rating model based on average IMDB ratings and a sentiment analysis score of user tweets
* Create an accurate Machine Learning model to predict average movie ratings based on some key features
* Make the system scalable by using big data technologies for data processing and Google cloud to host the system.

**Chapter 3**

**TECHNOLOGIES AND PLATFORM**

The Technologies used for this model are:

* Spark
* Zeppelin
* Jupyter
* Twitter API
* Google Cloud Engine
* Sentiment Analysis (Text Blob)
* Python
* HTML5
* CSS3
* JavaScript

We are using zeppelin to read the IMDB datasets and write scripts using Spark to extract the data as per the requirement.

We are using Jupyter to create the machine learning model for the movie rating predictor model.

WE are using Twitter API to extract tweets then perform sentiment analysis on them using the text blob. Then host the whole system on google cloud.

**Chapter 4**

**ARCHITECTURE**

We are taking the raw data from IMDB Datasets and filtered user tweets and passing them onto the Data preprocessing step, which takes care of cleaning the data and generates relevant data for other two workflows.

Movie rating model uses text blob to do sentiment analysis on filtered user tweets which is then converted into a normalized score based on a scoring scheme for each movie.

Movie Rating Prediction Model uses Spark ML to build a Linear Regression model to predict the average rating of a movie based on some key features.

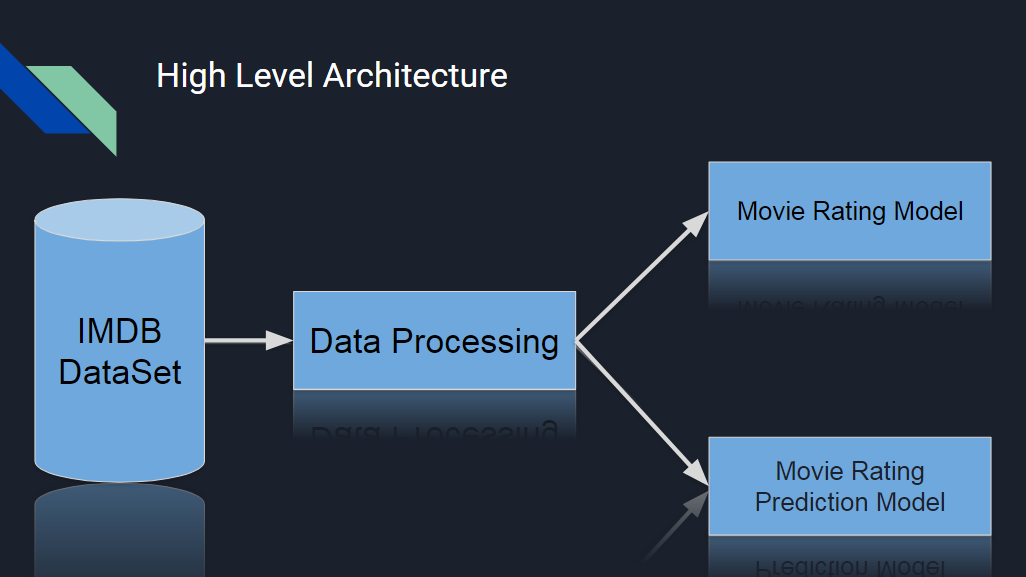
****

Figure: High Level Architecture

**Chapter 5**

**DATA PREPROCESSING**

The first step for our model is to utilize IMDB dataset and process it. Data preprocessing is done through a series of steps, namely:

* Cleaning
* Normalization
* Transformation
* Feature extraction
* Selection

Cleaning deals with removing unwanted data, and also to modify and delete the corrupted data. Since the data is large, there are chances some of the data might be corrupted so we need to remove the data. Along with that, here we removed the NULL values and duplicates.

Once the data is cleaned, then it is normalised, aggregated and generalised. This process is referred to as data transformation.

Data Selection involves the reduction of a number of values of a continuous attribute and selecting the columns and data required for the model.

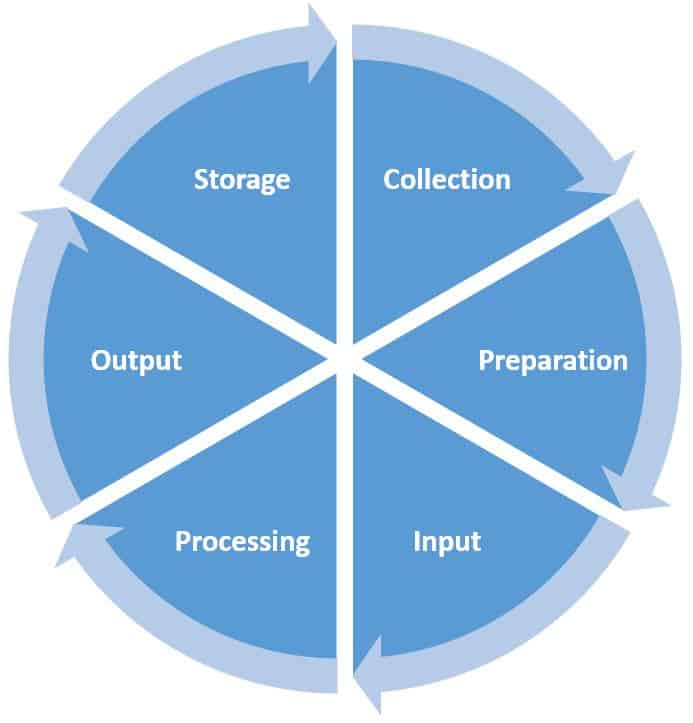


Figure: Stages of Preprocessing

For our model, we used the IMDB dataset and the steps followed to extract the required data are:

1. Read the IMDB dataset.
2. Filter the data and extract only the movies. (Since the data contains series, and other sitcoms as well)
3. Now filter the movies on the basis of year. (Here we are taking 2000-2017)
4. Now read the data for directors.
5. Extract and flatten only the directors and then merge with the movies data set extracted above.
6. Now read the data for writers.
7. Extract and flatten only the writers and then merge with the movies data set extracted above.
8. Now arrange the dataset in descending order according to the movie year.

**Chapter 6**

**SENTIMENT ANALYSIS**

We are ranking the top 10 movies for every year based on the tweets collected and performing sentiment analysis on them. Each movie received a score and then the scores were normalised and finally, the movies are ranked based on the scores received.

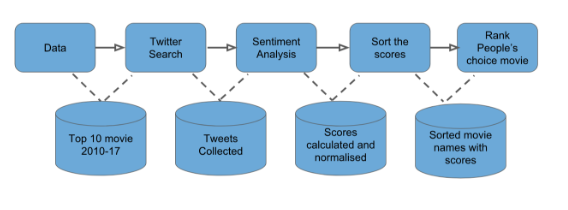
Data Processing is the initial step where we filter the movies for every year from 2010 till 2017. Next filter applied is getting movies with votes more than 10000 and then get the top 10 movies for every year. Then, we perform the sentiment analysis on the list of movies and rank them based on scores obtained.

All the sentiment analysis and script running is done on the Google Cloud Platform.

**Technologies**

* Twitter API
* Google Cloud Engine
* Sentiment Analysis (Text Blob)
* Python

**Architecture**

****

* Data Processing is done to get top ten movies from every year 2010-2017 using the IMDB dataset.
* Using the movie names as keywords, Twitter Search is done on the Google cloud platform and collected.
* While collecting the tweets, every tweets passes through a sentiment analysis where +1 is for positive tweet, -1 for negative tweet and nothing for a neutral tweet.
* Cumulative scores are calculated and then normalised.
* Based on the scores, the movies are ranked which is based on the choice of people.

**Reasons why Twitter Data for used for the analysis**

* Twitter is a popular microblogging website which has over a 240+ million active users.
* 500 million tweets are generated everyday on twitter.
* Twitter audience consists of people from all classes which includes common man and celebrities.
* Every tweet consists of short text messages of 140 characters so tweet analysis is easy.
* Tweets are smaller in length and unambiguous which makes it easy for analysis.

**Reason for Sentiment Analysis :**

* This is help us get the public opinion whether a movie review is positive or negative.
* Based on the sample tweets collected, it is easy to judge if a movie will win Oscar’s or not.
* Make a judgement if a movie should actually be nominated or not.

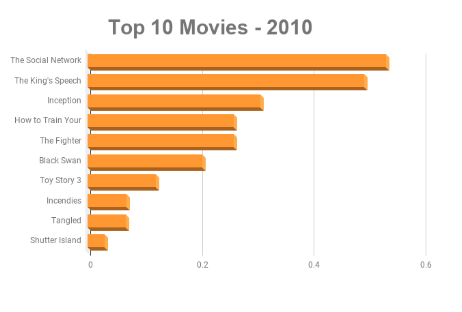
**Score Calculation and Ranking :**

* For every positive tweet +1
* For every negative tweet -1
* For every neutral tweet +0
* Total sum is normalised by adjusting values measured on different scales to a notionally common scale.
* Movie with the highest score is ranked first and movie with the least score is ranked last.

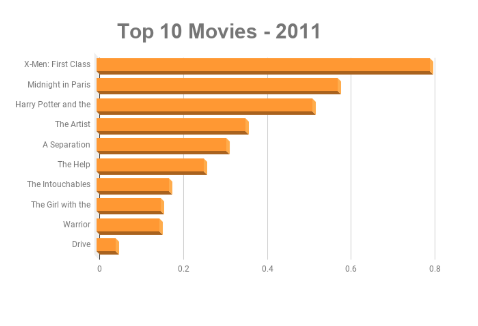
**Results :**

After performing the analysis based on the above diagram, top 10 movies from 2010 till 2017 were ranked separately for each year and following graphs were plotted for few years.

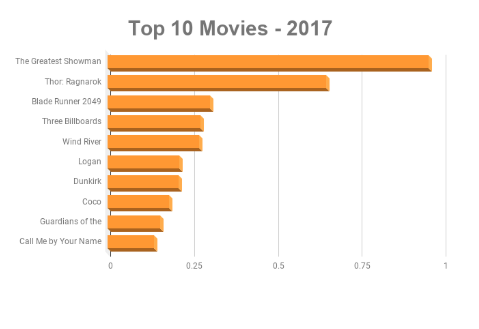
2010:



2011:



2017:



**Chapter 7**

**MACHINE LEARNING**

We built a Machine Learning model using Spark ML library to predict the average rating of a movie based on some key features:

* Director name
* Writer Name
* Run Time of the Movie
* Genre of the Movie
* Year of Release

*Technologies*

* Spark ML
* Jupyter
* Matplotlib
* plotly

Since we are trying to solve a classification model, we chose Linear Regression model to predict the ratings due to various following reasons :

* It’s a well known model for doing machine learning and gives simple representations which are easy to comprehend and draw conclusions out of.
* Gives a Linear relationship between a dependent variable and one or more independent variables, which fits our requirements perfectly as we are trying to predict movie ratings (dependent variable) using several features (independent variables).
* A scatter plot of our features and label shows that there is a close relationship between two variables

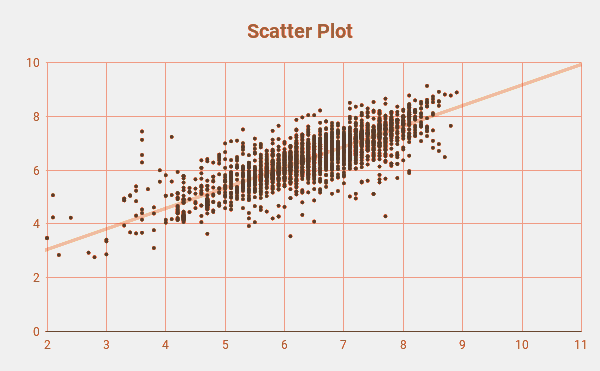


Figure: Scatter Plot

**ETL Architecture**

*Steps*

1. Extract the relevant data from the raw datasets i.e. perform data pre processing steps.
2. Transform the extracted data into a form suitable for ML model, since our data contains a lot columns with string values we have to do some transformations to convert them into numeric columns suitable for a Machine Learning Model.
3. Split the transformed data into Train and Test dataset (generally 70 - 30 , but we did 80 -20).
4. Build the model by feeding the train dataset into ML Algorithm.
5. Test or evaluate the model built using Test dataset

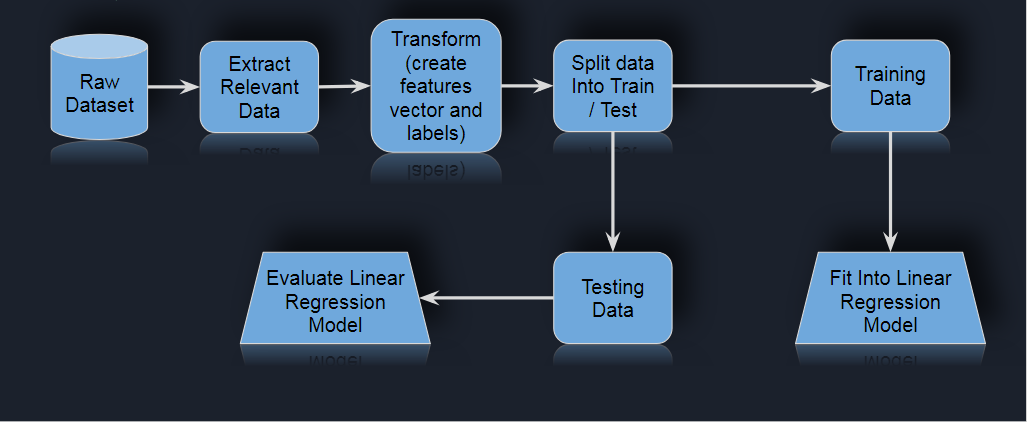


Figure: ETL Architecture

**Transformation Pipeline**

*Steps*

1. Feed the extracted relevant data into the string indexer to get an indexed dataset (or rather the indexed columns)
2. Indexed data is passed through the OneHotEncoder method which takes the indexed columns and makes numeric vectors of them suitable for machine learning process
3. Then we call the vectorAssembler on this encoded data to gather all the vectors into a single vector and make a new column of features of this assembled vector.
4. Then we can input the (column 1) feature [vector] and (column 2) Label [doubles] into the ML algorithm

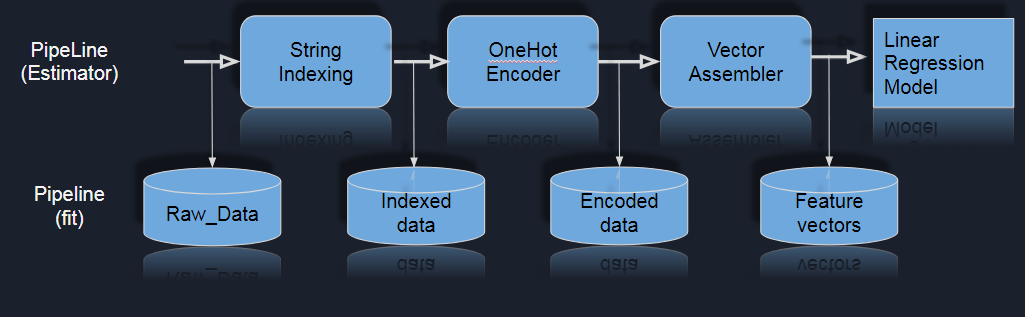


Figure: Transformation Pipeline

**Model Summary**

Some of the model metrics for evaluating the accuracy of the model.

|  |  |
| --- | --- |
| Mean Squared Error | 0.051903 |
| Mean Absolute Error | 0.135875 |
| Root Mean Squared Error | 0.227823 |
| r2 | 0.955105 |
| Degree Of Freedom | 2261.000000 |
| Intercept | 6.53804863727 |
| numIterations | 101 |

**Summary Interpretation**

**MAE** (**Mean Absolute Error**) It is a measure of difference between two continuous variables

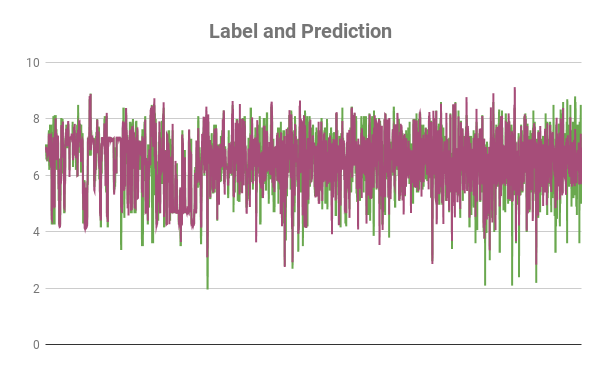
**MSE** (**Mean Squared Error**) It is a quality measure for the estimator

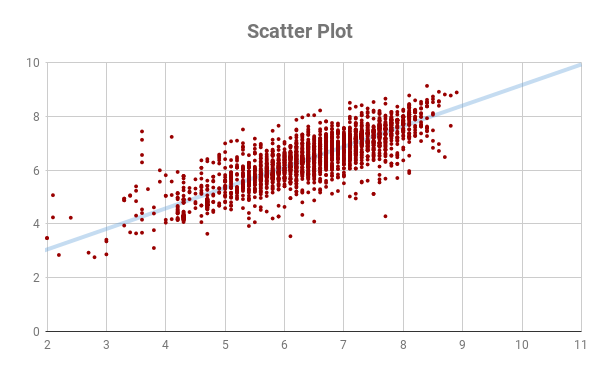
**RMSE** It is a measure of the average deviation of the estimates from the observed values (closer to zero is better)

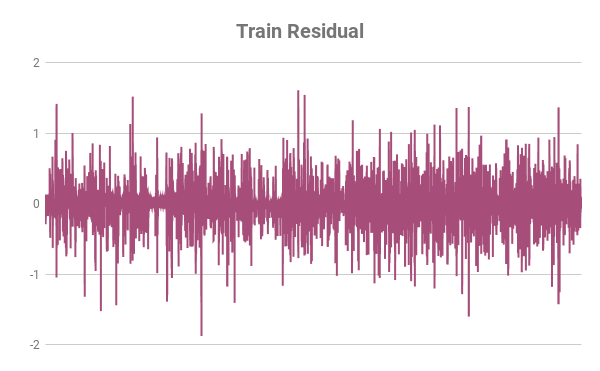
**r2** (**R-squared**) It is a statistical measure of how close the data are to the fitted regression line

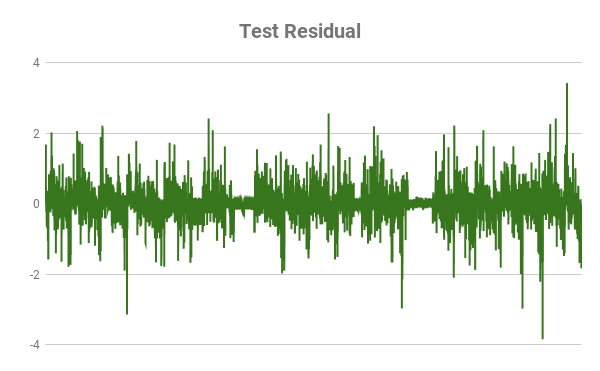
**DOF** (**Degree Of Freedom**) It is the number of independent pieces of information that go into the estimate of a parameter

**Plots**

****

****

****

****

**REFERENCES**

**[1]** <https://spark.apache.org/docs/1.2.2/ml-guide.html>

**[2]**<https://towardsdatascience.com/regression-analysis-model-used-in-machine-learning-318f7656108a>

**[3]** <https://spark.apache.org/docs/2.2.0/ml-classification-regression.html>

**[4]** <https://planningtank.com/computer-applications/data-processing-cycle>

**The link to the website is :** <https://architgyl.github.io/>