**Write Up for the Project 2: Movielens Case Study**

**Objective:** The GroupLens Research Project has one of the projects where ratings of movies are needed to be analysed and the features affecting them need to be determined and also and prediction model needs to be built.

**Domain:** Entertainment

**Analysis Tasks to be performed:**

1. Import the 3 datasets(movies.dat, ratings.dat, users.dat): First we import the pandas, numpy and matplotlib libraries of python.

**import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

Then we import the datasets by:

**movies = pd.read\_csv('movies.dat', delimiter='::', names= ['movieid', 'title', 'genres'], encoding= 'latin-1', engine= 'python')**

**ratings = pd.read\_csv('ratings.dat', delimiter='::', names=['userid','movieid','rating','timestamp'], encoding= 'latin-1',engine='python')**

**users = pd.read\_csv('users.dat', delimiter='::', names=['userid','gender','age', 'occupation', 'zip\_code'], encoding='latin-1', engine= 'python')**

Here the list *‘name’* has the columns of the 3 datasets respectively.

1. Creation of new dataset [master\_data] with the following columns : movieid, title, userid, age, gender, occupation and ratings. This done by Merging two tables at a time and the primary Keys chosen are movieid and userid.

**df= pd.merge(users.drop('zip\_code',axis=1), ratings.drop('timestamp',axis=1),on= 'userid')**

**master\_data= pd.merge(df,movies,on='movieid')**

The two tables users and ratings are merged into a dataset ‘df’, ‘on’ userid and the columns zip\_code and timestamp is dropped. Then the master\_data is created by merging ‘df’ with movies table ‘on’ movieid.

1. Exploration of dataset with the help of tables and graphs.
   1. User age distribution explored.

**Table form: master\_data.age.value\_counts(normalize=True).to\_frame()**

**Graphical form: fig= plt.figure(figsize =(15,6))**

**plt.hist(master\_data['age']);**

Using a histogram to represent the age distribution in the dataset.

* 1. Next we get the user rating of the movie ‘Toy Story’, by grouping the dataset by ‘title’ & ‘rating’ and then sorting them on the basis of rating and storing it in ‘m\_rating’ dataset. And finally the ‘loc’ function gives the rating of the movie.

**m\_rating= master\_data[['title','rating']].groupby('title').mean().sort\_values('rating', ascending= False)**

**m\_rating.loc[['Toy Story (1995)']]**

* 1. To get the top 25 movies according to the rating, ‘head’ function is used and 25 is passed in the parameter option.

**m\_rating.head(25)**

First movie being ‘*Ulysses (Ulisse) (1954)*’ with rating ‘5’, and 25th movie being ‘*Bells, The*

*(1926)*’ with rating ‘4.5’.

* 1. The project demands us to explore the movies reviewed by the user with id ‘2696’.

**master\_data[master\_data['userid']==2696]**

This user reviewed 20 movies in the dataset.

1. Next we use the ‘genres’ column to find unique genres by applying lambda function and splitting the genres wherever ‘|’ is encountered in the genres column.

**movies['column\_genres']= movies['genres'].apply(lambda x : str(x).split('|'))**

**movies.head()**

* 1. Then after splitting we create an empty list ‘genres’ and then pass the different genres items in this empty list then ‘set’ function is used to get the unique values and the list is stored back in genres list and then it is printed.

**genres=[]**

**for i in movies['column\_genres']:**

**genres.extend(i)**

**genres= set(genres)**

**print(genres)**

* 1. To check weather a movie belongs to a genre or not we create a new column with every genre with one-hot encoding. So we apply lamba function to the ‘genres’ column and it returns 1 if the movie is present in the genre and 0 if it is not.

**for i in genres:**

**print(i)**

**master\_data[i]=master\_data['genres'].apply(lambda x : 1 if i in x else 0)**

**master\_data.head(1)**

* 1. Now to determine the features affecting the rating of the movies we use ‘corr’ function on the ratings column. So now all the features correlating to rating can be studied.

**master\_data.corr()[['rating']]**

For example ‘Horror’, is negatively related to rating and ‘Drama’ is positively related to rating of the movies.

* 1. To build a prediction model for the rating of the movies, we import the following:

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.linear\_model import LinearRegression**

Then the features affecting the rating of the movies are the Features (X), and the

rating is the Target or Label(Y) :

**x= master\_data[['age','Fantasy', 'Documentary', 'Musical', 'Film-Noir', 'Animation',**

**'Sci-Fi', 'Thriller', 'Drama', "Children's", 'Mystery', 'Western',**

**'War', 'Comedy', 'Crime', 'Romance', 'Adventure', 'Horror', 'Action']]**

**y= master\_data['rating']**

Then the dataset is divided into training and testing, with 20% being the test data :

**x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size=.2)**

Then we use the Linear Regression model and after training on x\_train & y\_train, prediction is made on x\_test :

**model=LinearRegression()**

**model.fit(x\_train,y\_train)**

**model.predict(x\_test)**

**Conclusion :** The project explores the various features affecting the rating of movies in the dataset. Exploratory data analysis and feature engineer is used to study the tables. Python libraries are used in the process on Jupyter notebook.