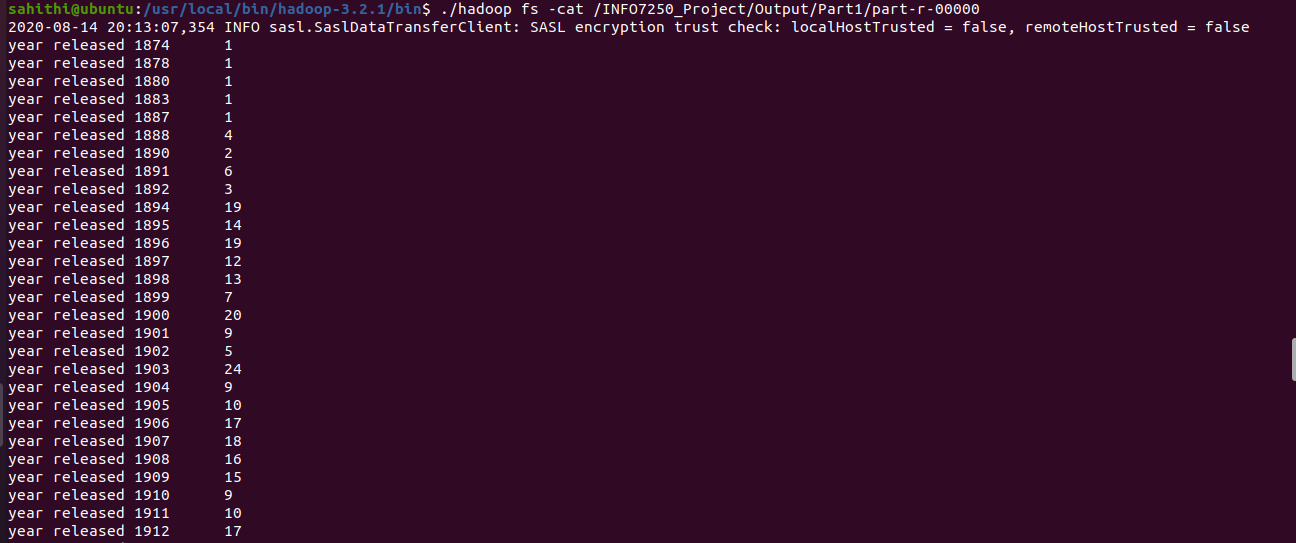
Final Project

Part 1:

**Find number of movies released per year**

./hadoop jar /home/sahithi/Downloads/eclipse-installer/Part1.jar info7250.bigData.Project.Part1.Driver /INFO7250\_Project/ml-25m/movies.csv /INFO7250\_Project/Output/Part1

./hadoop fs -cat /INFO7250\_Project/Output/Part1/part-r-00000

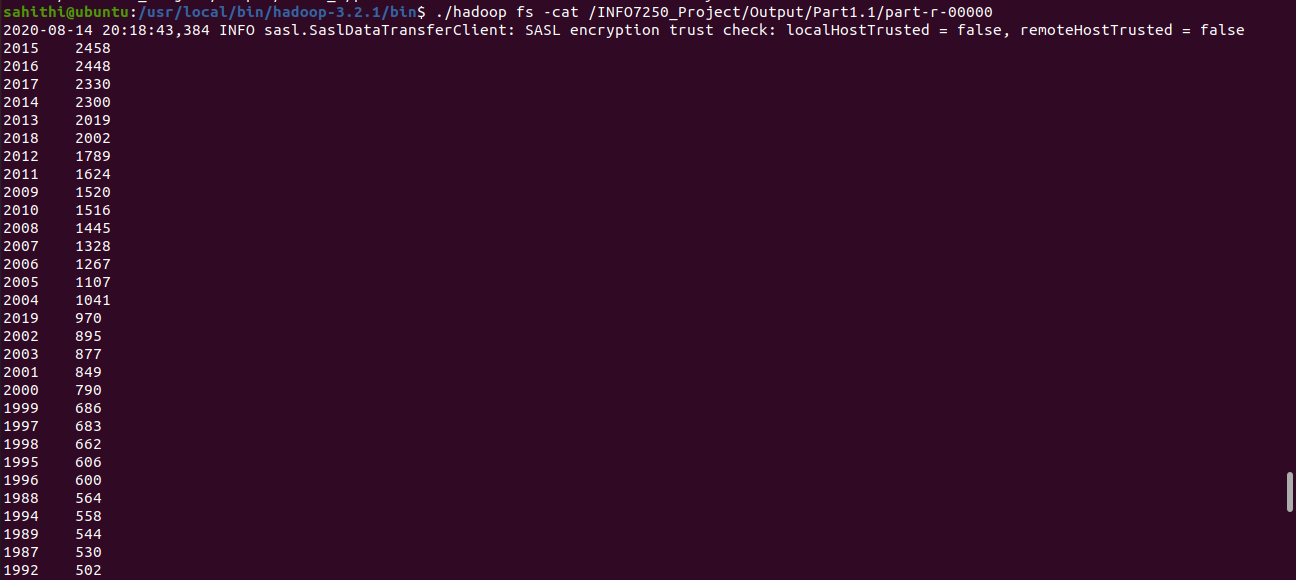


Part 2:

**Find the years when most movies were released.**

./hadoop jar /home/sahithi/Downloads/eclipse-installer/Part1\_1.jar info7250.bigData.Project.Part1\_1.Driver /INFO7250\_Project/ml-25m/movies.csv /INFO7250\_Project/Output/Part1.1

./hadoop fs -cat /INFO7250\_Project/Output/Part1.1/part-r-00000

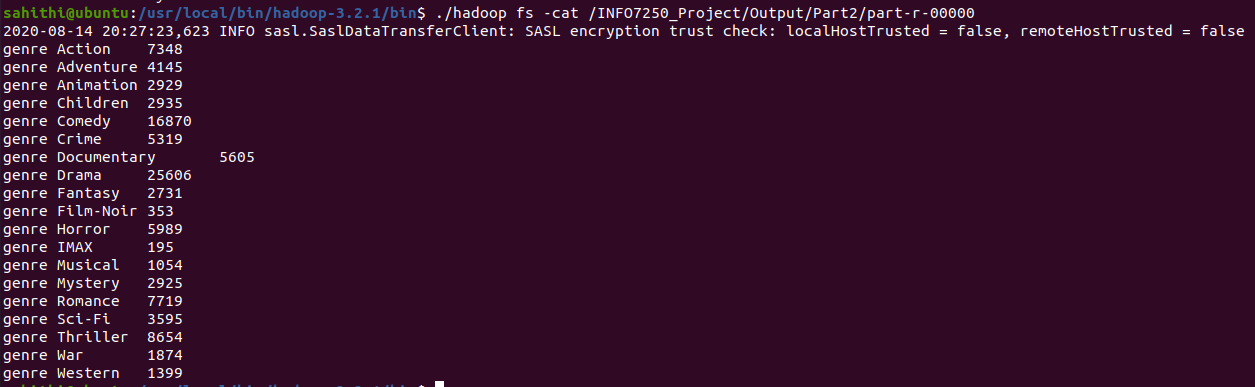


Part 3:

**Find total number of movies per genre**

./hadoop jar /home/sahithi/Downloads/eclipse-installer/Part2.jar info7250.bigData.Project.Part2.Driver /INFO7250\_Project/ml-25m/movies.csv /INFO7250\_Project/Output/Part2

./hadoop fs -cat /INFO7250\_Project/Output/Part2/part-r-00000



Part 4:

**Find number of tags for each movieId**

./hadoop jar /home/sahithi/Downloads/eclipse-installer/Part3.jar info7250.bigData.Project.Part3.Driver /INFO7250\_Project/ml-25m/tags.csv /INFO7250\_Project/Output/Part3

./hadoop fs -head /INFO7250\_Project/Output/Part3/part-r-00000

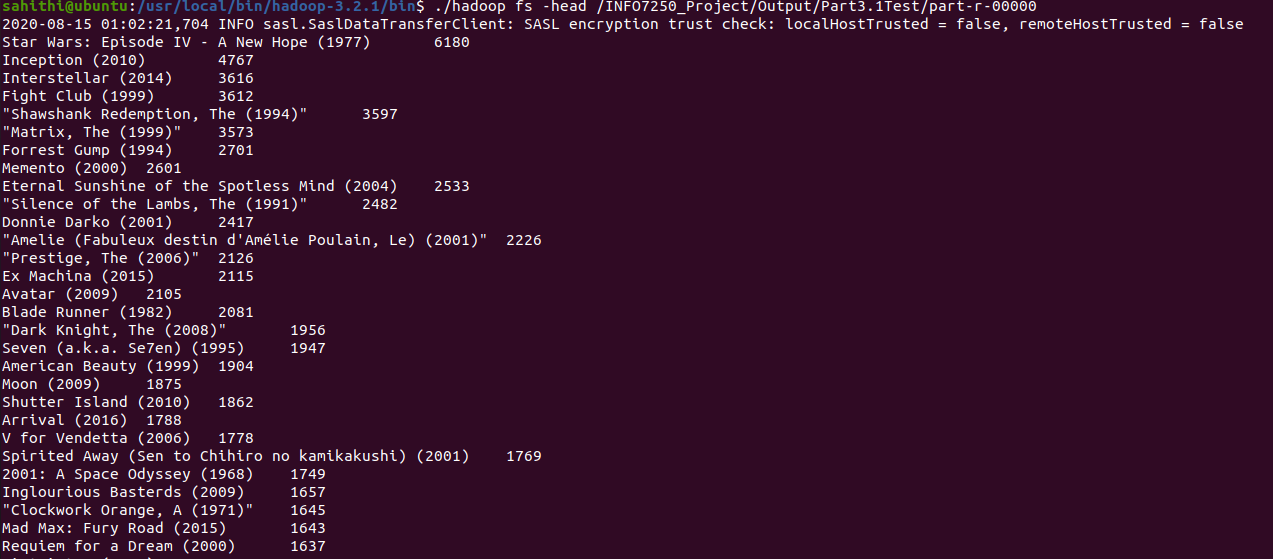


Part 5:

**Find movies that are most tagged by the users**

./hadoop jar /home/sahithi/Downloads/eclipse-installer/Part3\_1Test.jar info7250.bigData.Project.Part3\_1.Driver /INFO7250\_Project/ml-25m/tags.csv /INFO7250\_Project/ml-25m/movies.csv /INFO7250\_Project/Output/Part3.1Test

./hadoop fs -head /INFO7250\_Project/Output/Part3.1Test/part-r-00000

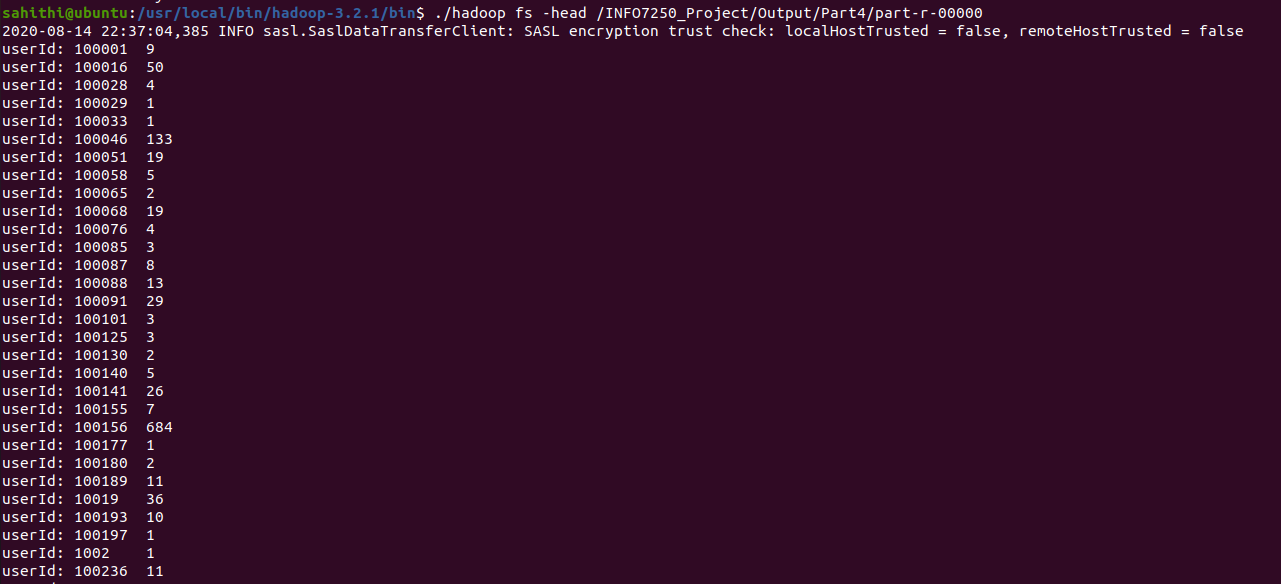


Part 6:

**Find the number of times a user tagged movies** (Since are usernames are not available in the dataset, performing the analysis using userId’s)

./hadoop jar /home/sahithi/Downloads/eclipse-installer/Part4.jar info7250.bigData.Project.Part4.Driver /INFO7250\_Project/ml-25m/tags.csv /INFO7250\_Project/Output/Part4

./hadoop fs -head /INFO7250\_Project/Output/Part4/part-r-00000



Part 7:

**Find probable non-spam users based on the user tag relevance**

genomeScore = LOAD '/home/sahithi/Desktop/ml-25m/genome-scores.csv' USING PigStorage(',') AS (movieId:int, tagId:int, relevance:float);

genomeTag = LOAD '/home/sahithi/Desktop/ml-25m/genome-tags.csv' USING PigStorage(',') AS (tagId:int, tag:chararray);

genome = JOIN genomeTag by tagId, genomeScore by tagId;

genomeFilter = FILTER genome by relevance >= 0.7;

tagData = LOAD '/home/sahithi/Desktop/ml-25m/tags.csv' Using PigStorage(',') As (userId:int,movieId:int,tag:chararray,timestamp:double);

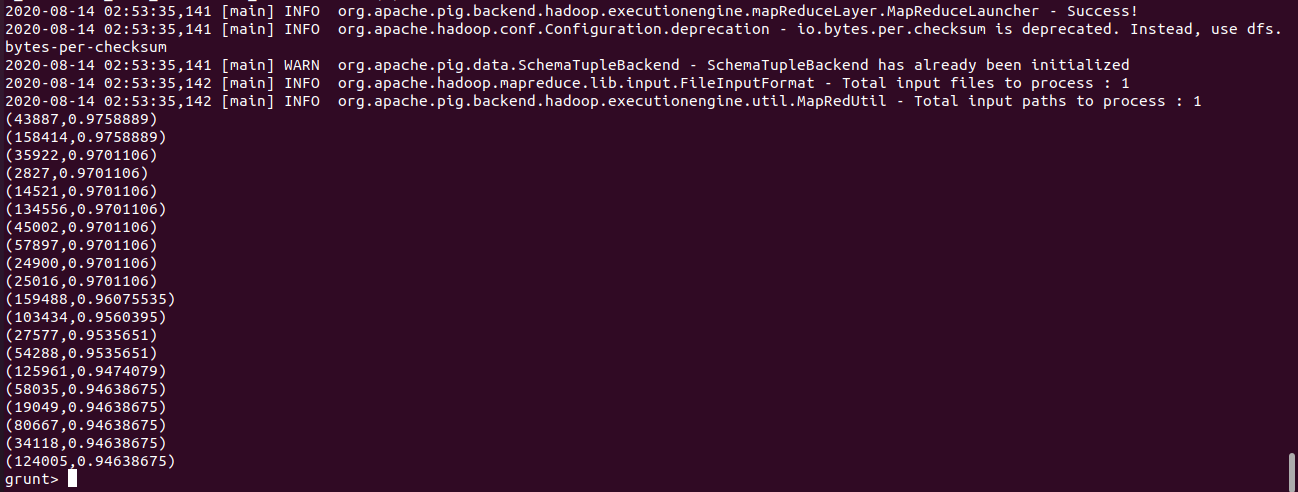
genomeTagCombine = JOIN genomeFilter by tag, tagData by tag;

genomeRelevantUser = FOREACH genomeTagCombine GENERATE userId, relevance;

genomeTagCombineGroup = GROUP genomeRelevantUser by userId;

genomeTagAvg = FOREACH genomeTagCombineGroup GENERATE group,AVG(genomeRelevantUser.relevance) AS avgRelevance:float;

genomeTagSort = ORDER genomeTagAvg by avgRelevance DESC;

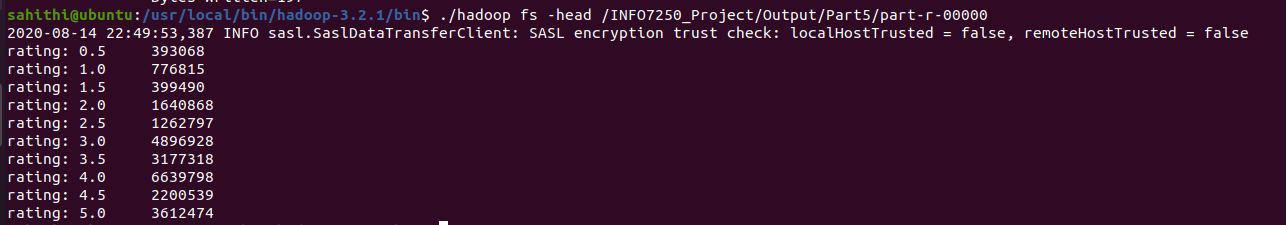


Part 8:

**Number of movies for each rating value**

/hadoop jar /home/sahithi/Downloads/eclipse-installer/Part5.jar info7250.bigData.Project.Part5.Driver /INFO7250\_Project/ml-25m/ratings.csv /INFO7250\_Project/Output/Part5

./hadoop fs -head /INFO7250\_Project/Output/Part5/part-r-00000



Part 9:

**Provide an IMDB link for each movie**

movieData = LOAD '/home/sahithi/Desktop/movies.txt' Using PigStorage() As(MovieId:int, Title:chararray, Genre:chararray);

linksData = LOAD '/home/sahithi/Desktop/ml-25m/links.csv' Using PigStorage(',') As(MovieId:int, ImdbId:int, TmdbId:int);

movie\_Links = JOIN movieData by MovieId, linksData by MovieId;

movieLinkAdded = FOREACH movie\_Links GENERATE \*, CONCAT('http:://www.imdb.com/title/tt',(chararray)ImdbId) AS IMDBLink:chararray;

outLinks = FOREACH movieLinkAdded GENERATE Title,IMDBLink;

lm = LIMIT outLinks 25;

dump lm;



Part 10:

**Find the average rating for each movie along with its corresponding user provided tags.**

ratingData = LOAD '/home/sahithi/Desktop/ml-25m/ratings.csv' USING PigStorage(',') AS (userId:int,movieId:int,rating:float,timestamp:double);

groupRating = Group ratingData By movieId;

ratingAvg = FOREACH groupRating GENERATE group, AVG(ratingData.rating) AS average:float;

tagData = LOAD '/home/sahithi/Desktop/ml-25m/tags.csv' Using PigStorage(',') As (userId:int,movieId:int,tag:chararray,timestamp:double);

groupTag = GROUP tagData by movieId;

tagConcat = FOREACH groupTag GENERATE group, BagToTuple(tagData.tag) as tagCollection;

movieData = LOAD '/home/sahithi/Desktop/movies.txt' Using PigStorage() As(MovieId:int, Title:chararray, Genre:chararray);

tagMovieRatingJoin = JOIN ratingAvg by group,tagConcat by group,movieData by MovieId;

tagMovieRating = FOREACH tagMovieRatingJoin GENERATE Title,average,tagCollection;

lmi = LIMIT tagMovieRating 1;

dump lmi



Part 11:

**Find the list of movies released in 2011 in “Action” Genre**

./hadoop jar /home/sahithi/Downloads/eclipse-installer/Part8.jar info7250.bigData.Project.Part8.Driver /INFO7250\_Project/ml-25m/movies.csv /INFO7250\_Project/Output/Part8

./hadoop fs -head /INFO7250\_Project/Output/Part8/part-m-00000



Part 12:

**Find the top movies based on user tags relevance**

genomeData = LOAD '/home/sahithi/Desktop/ml-25m/genome-scores.csv' Using PigStorage(',') As(movieId:int, tagId:int, relevance:float);

groupGenome = GROUP genomeData by movieId;

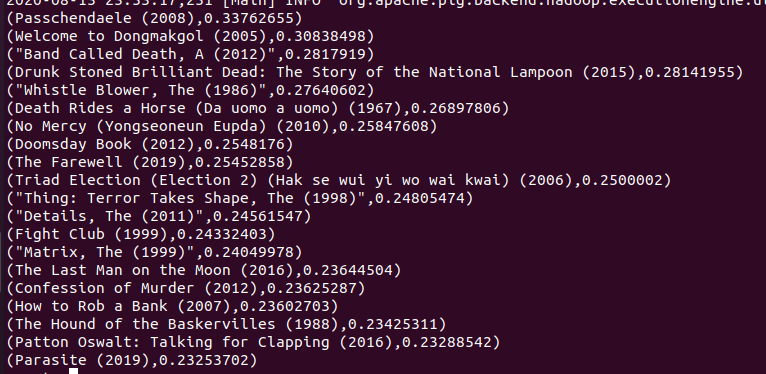
GenomeAvg = FOREACH groupGenome GENERATE group, AVG(genomeData.relevance) AS average:float;

movieData = LOAD '/home/sahithi/Desktop/movies.txt' Using PigStorage() As(MovieId:int, Title:chararray, Genre:chararray);

movieGenome = JOIN movieData by MovieId, GenomeAvg by group;

movieRelevance = FOREACH movieGenome GENERATE Title, average;

orderMovieRelevance = ORDER movieRelevance BY average DESC;



Part 13:

**Find top 5 movies of all time per genre based on average user rating**

ratingData = LOAD '/home/sahithi/Desktop/ml-25m/ratings.csv' USING PigStorage(',') AS (userId:int,movieId:int,rating:float,timestamp:double);

grpByRating = GROUP ratingData by movieId;

grpByRatingAvg = FOREACH grpByRating GENERATE group, AVG(ratingData.rating) AS average:float;

movieData = LOAD '/home/sahithi/Desktop/movies.txt' Using PigStorage() As(MovieId:int, Title:chararray, Genre:chararray);

movierating = JOIN movieData by MovieId , grpByRatingAvg by group;

movieGenreAvgRating = FOREACH movierating GENERATE Title,Genre,average;

movieGenreFlatten = FOREACH movieGenreAvgRating GENERATE Title,average, flatten(TOKENIZE(Genre, '|'));

movieGenreGroup = GROUP movieGenreFlatten by token;

movieGenreGroupDesc = FOREACH movieGenreGroup {

movieGenreOrder = ORDER movieGenreFlatten BY average DESC;

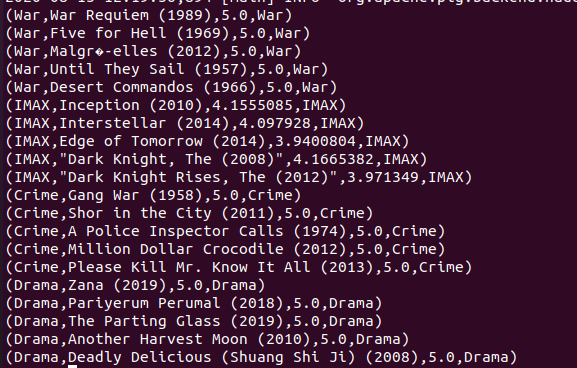
top = LIMIT movieGenreOrder 5;

GENERATE group, flatten(top);

}

lmi = LIMIT movieGenreGroupDesc 20;

dump lmi;



Part 14:

**Genres liked by user based on their given rating**

ratingData = LOAD '/home/sahithi/Desktop/ml-25m/ratings.csv' USING PigStorage(',') AS (userId:int,movieId:int,rating:float,timestamp:double);

movieData = LOAD '/home/sahithi/Desktop/movies.txt' Using PigStorage() As(MovieId:int, Title:chararray, Genre:chararray);

movieRating = JOIN movieData by MovieId, ratingData by movieId;

userRatingGenre = FOREACH movieRating GENERATE userId, rating,flatten(TOKENIZE(Genre, '|'));

userGenreGroup = Group userRatingGenre by (userId,token);

userGenreAvg = FOREACH userGenreGroup GENERATE group, AVG(userRatingGenre.rating) as average:float;

userGenreOrder = ORDER userGenreAvg BY average DESC;

lmi = LIMIT userGenreDesc 10;

dump lmi;

