Chapter 4 – Implementation

Implementation is defined as the process of transforming the theoretical concepts and ideas into practical and useful form. Where the plan and process are carried out in action. The necessary steps are described and action is taken according to the design and plan made. Execution may include analysis of data, processing and providing desired result. Defining the goal of the project, conducting proper research and analysing the result and meeting the scheduled deadline, such key strategies and execution in performed in the implementation section. (Lutkevich & Ehrens, 2022). The system design and the working mechanism is shown below in figure 4.1.

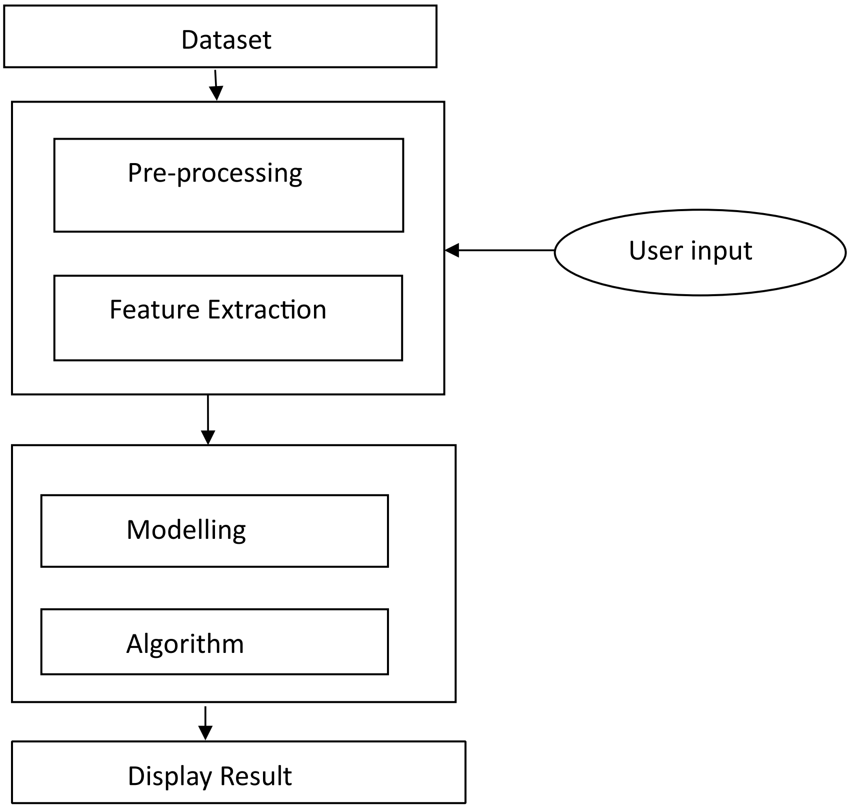
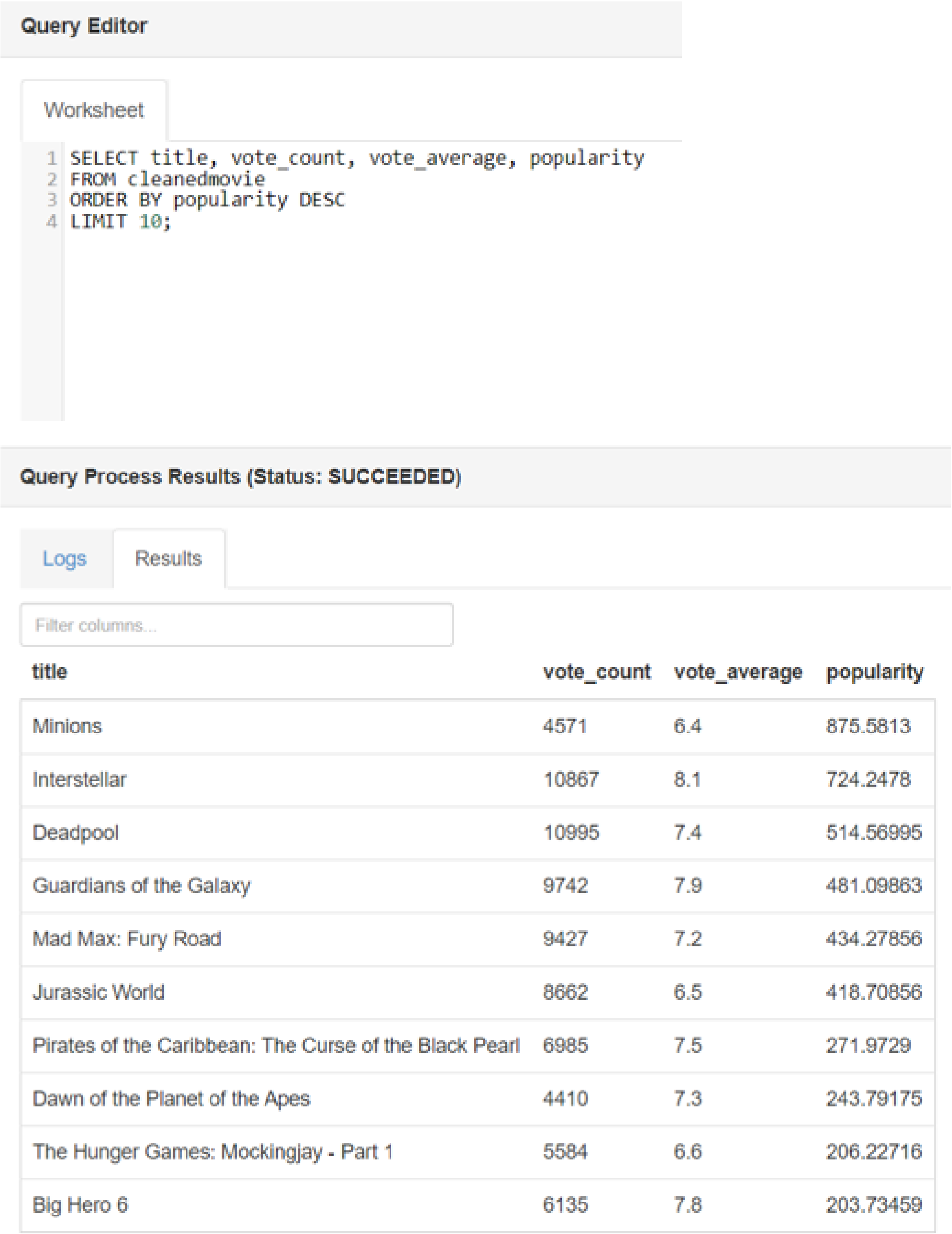


Figure 4.1: System Design Architecture

4.1 Phases in query analysis and recommendation

The process of recommendation cannot be completed in single step. It takes several steps, which involves collection of data and analysis of data, knowing which attributes are suitable for the recommendation. Cleaning and pre-processing of data is an important step, where data needs to be in proper order cleaned and ready to use, so that it makes extraction and modelling process easy. Extracting the features, converting the data into vectors, creating a model and implementation of algorithm will provide the result according to the user query.

4.2 Analysis and Results



```
SELECT title, vote_count, vote_average, popularity
FROM cleanedmovie
ORDER BY popularity DESC
LIMIT 10;
```

Query Process Results (Status: SUCCEEDED)

title	vote_count	vote_average	popularity
Minions	4571	6.4	875.5813
Interstellar	10867	8.1	724.2478
Deadpool	10995	7.4	514.56995
Guardians of the Galaxy	9742	7.9	481.09863
Mad Max: Fury Road	9427	7.2	434.27856
Jurassic World	8662	6.5	418.70856
Pirates of the Caribbean: The Curse of the Black Pearl	6985	7.5	271.9729
Dawn of the Planet of the Apes	4410	7.3	243.79175
The Hunger Games: Mockingjay - Part 1	5584	6.6	206.22716
Big Hero 6	6135	7.8	203.73459

Figure 4.2: Analysis of movies based on popularity

The above analysis is performed using Hive query language, ranked based on popularity along with the total vote count and average rating which provides the top 10 popular movies. It can be observed that result based on popularity is given but the total number of votes can vary and shows that popularity is not dependent of the average rating given, which basically means that high popularity doesn’t mean high rating and vice versa, advertisement and famous actors may make movie popular but if the story, content and plot of movie is good than it may be highly rated. The above figure 4.2 shows the query and result of the query.

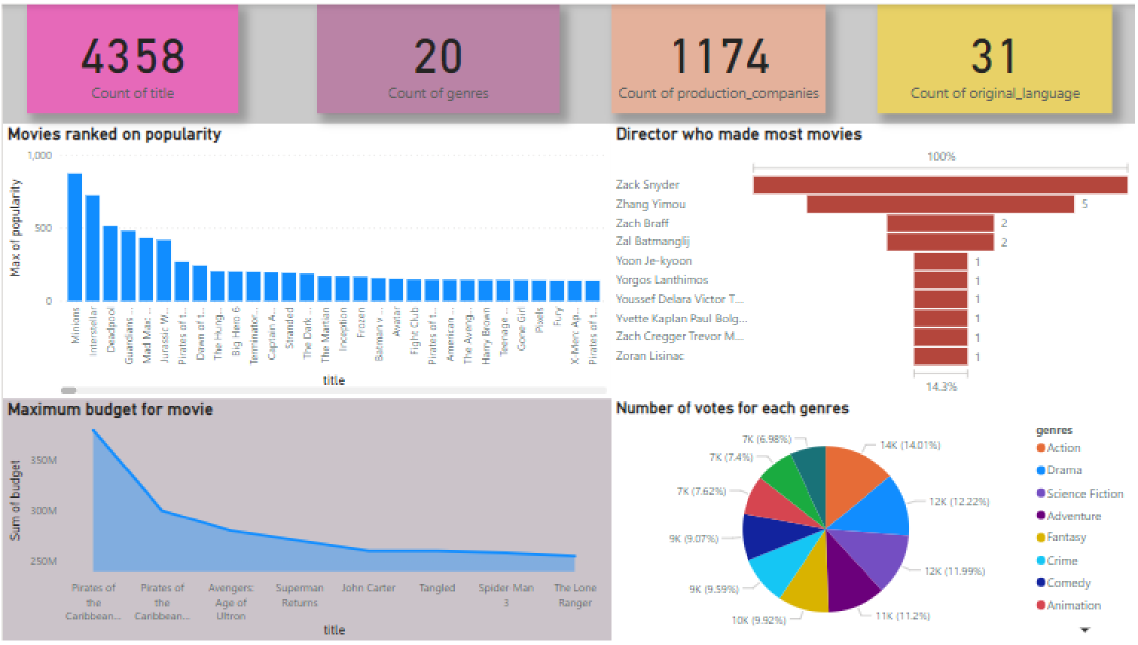


Figure 4.3: Visualization of movies data

The above graphs and charts provide the visualization aspect of movie data as shown in figure 4.3, where different features related to movies are computed and plotted. Starting with basic information like total number of movies, genres, production companies, and original language. On the top left corner down form the cards shows the statistics where the movies are shown in bar chart according to the popularities, secondly on the top right corner it is providing information related to director's and how many movies they have made with the help of funnel diagram. Thirdly on bottom left corner the most expensive movies are ranked based on their budget with the help of area chart. Finally, the pie chart on bottom right corner provides the information regarding the number of genres on which the movies are made.

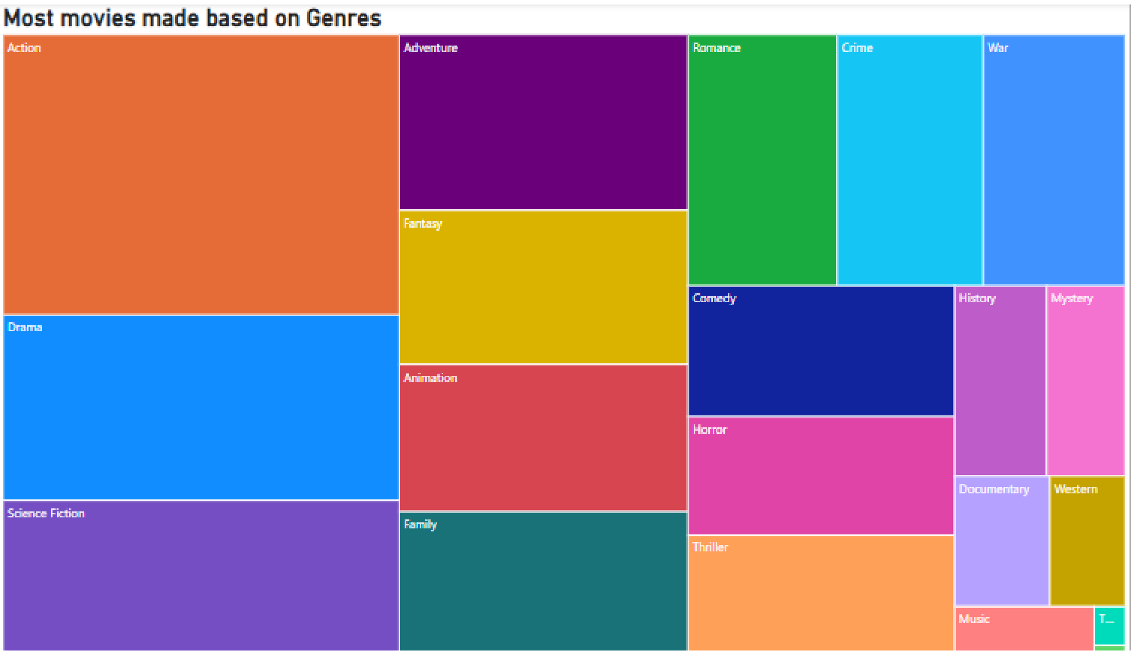


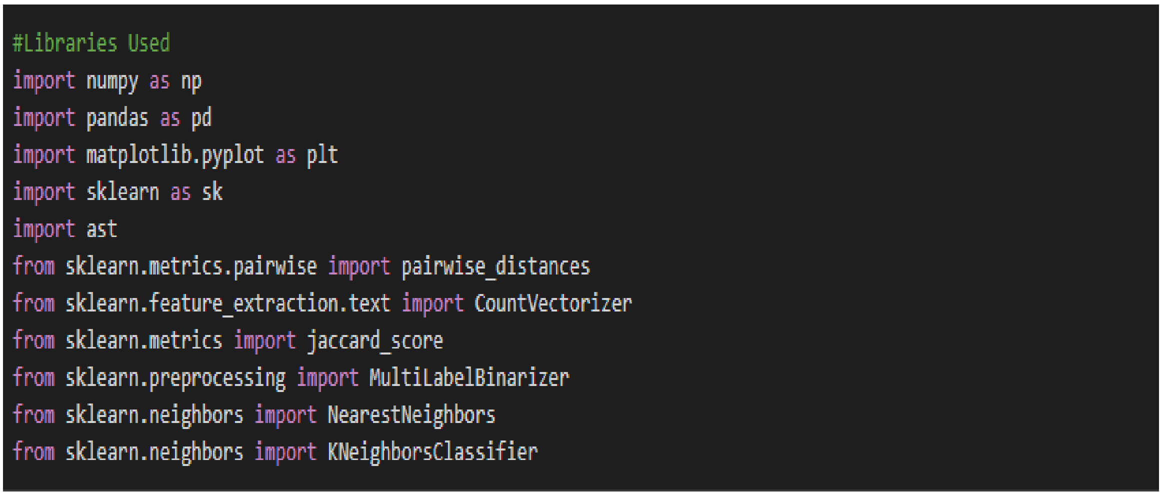
Figure 4.4: Tree map showing top genres

The tree map is able to provide the information related to genres on movies, showing that the number of movies are mostly made on action genre, followed by drama, science fiction, adventure, fantasy, animation and so on. Which is shown above in figure 4.4.

Overall, Visualization provides in depth understanding of data, able to show the huge data in diagrams making it easier for user to understand and able to make good decision. Movies and movies related attributes and their appropriate relationship can be evaluated for better understating of data in visual form.

Libraries used in machine learning

In machine learning libraries can be defined as the built-in or pre-written functions and the codes that enables users to pre-process the data, train and extract features from the data, which helps in implementation and evaluation of results. It reduces the effort and time for implementing the code from the scratch. Libraries makes sure that the code are running in efficient manner, error-free as these codes and functions in libraries are written by experts. The libraries used in machine learning for this project are shown below in figure 4.5.

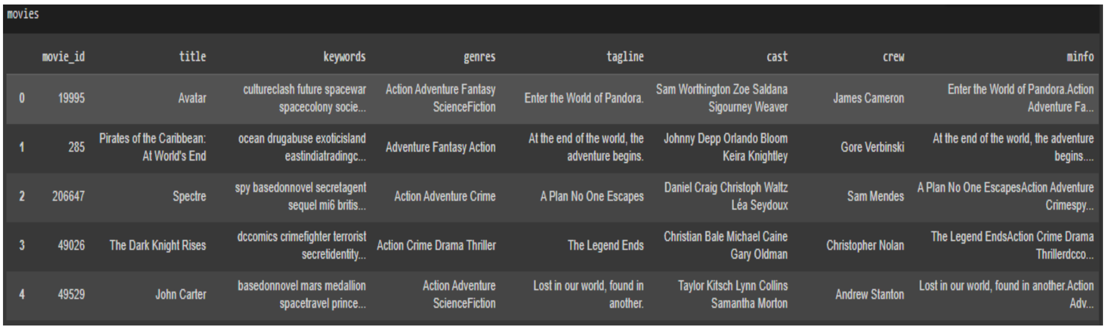


```
#Libraries Used
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn as sk
import ast
from sklearn.metrics.pairwise import pairwise_distances
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import jaccard_score
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.neighbors import NearestNeighbors
from sklearn.neighbors import KNeighborsClassifier
```

Figure 4.5: Libraries used

Cleaning and Pre-processing of Data

It involved the task of merging the two datasets. As a part of data cleaning process first thing was figuring out the null values or empty spaces in a dataset. As required for this dataset there were some taglines missing form the dataset, which were replaced by the movie names and the movies who didn't provide the information regarding their movie genres were filled with "Unknow" term and same action was performed for some empty keywords. As for the pre-processing of data only the relevant information from each data has been extracted, which involved extracting names of movies, keywords and movie genres, picking top 3 casts from the movie and just using director name as crew ignoring all information which are less relevant form entire dataset. The cleaned and pre-processed data has been shown in figure 4.6



movies

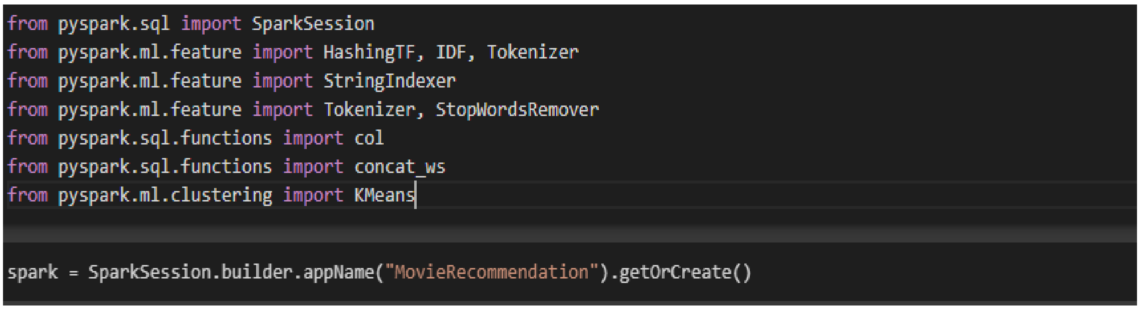
	movie_id	title	keywords	genres	tagline	cast	crew	minfo
0	19995	Avatar	cultureclash future spacewar spacecolony socie...	Action Adventure Fantasy ScienceFiction	Enter the World of Pandora.	Sam Worthington Zoe Saldana Sigourney Weaver	James Cameron	Enter the World of Pandora.Action Adventure Fa...
1	285	Pirates of the Caribbean: At World's End	ocean drugabuse exoticisland eastindiatradingc...	Adventure Fantasy Action	At the end of the world, the adventure begins.	Johnny Depp Orlando Bloom Keira Knightley	Gore Verbinski	At the end of the world, the adventure begins....
2	206647	Spectre	spy basedonnovel secretagent sequel mi6 britis...	Action Adventure Crime	A Plan No One Escapes	Daniel Craig Christoph Waltz Léa Seydoux	Sam Mendes	A Plan No One EscapesAction Adventure Crimespy...
3	49026	The Dark Knight Rises	dccomics crimefighter terrorist secretidentity...	Action Crime Drama Thriller	The Legend Ends	Christian Bale Michael Caine Gary Oldman	Christopher Nolan	The Legend EndsAction Crime Drama Thrillerdcco...
4	49529	John Carter	basedonnovel mars medallion spacetravel prince...	Action Adventure ScienceFiction	Lost in our world, found in another.	Taylor Kitsch Lynn Collins Samantha Morton	Andrew Stanton	Lost in our world, found in another.Action Adv...

Figure 4.6: Pre-processed data

Implementation of pySpark

Movie recommendation system using big data technology can bring efficiency in processing of data as it has high processing power and can handle large data sets. In this project pySpark with K-means clustering is implemented to create a system. Where pySpark is used to process the large dataset and K-mean clustering is used to cluster the similar datapoints. As it is a movie recommendation system k -mean has been used to cluster the features in the movies like genres, casts, crew, keywords and the movies has been clustered based on those features which is further used for recommendation. Tokenization of words, removing the stop words, counting the frequency of words.

Spark Session object is created, which acts as entry points form where the project is created and with the help of this object different operations can be carried out, which may include writing and reading of data, performing analysis on data, transformation, aggregation, and it is able to run machine learning algorithm. pySpark comes with inbuilt libraries enabling machine learning and data processing to be carried out in convenient manner. pySpark SQL and pySpark MLlib are some of the libraries which is made accessible by installing pySpark. The libraries used for this project and the Spark Session build has been shown below in figure 4.7.

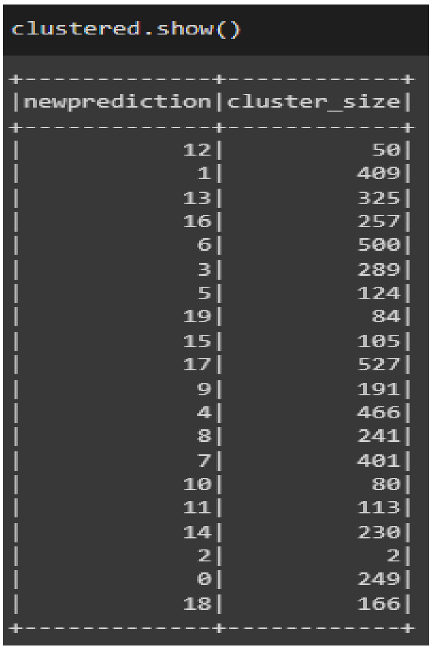


```
from pyspark.sql import SparkSession
from pyspark.ml.feature import HashingTF, IDF, Tokenizer
from pyspark.ml.feature import StringIndexer
from pyspark.ml.feature import Tokenizer, StopWordsRemover
from pyspark.sql.functions import col
from pyspark.sql.functions import concat_ws
from pyspark.ml.clustering import KMeans
```

```
spark = SparkSession.builder.appName("MovieRecommendation").getOrCreate()
```

Figure 4.7: Spark Session and Libraries for pySpark

Based on the frequency of words and features provided the clusters are form grasping the similarities between the movies. The predictions obtained is further used for recommendation. The size of the clusters formed and the predictions are shown below in figure 4.8.



```
clustered.show()

+-------------+------------+
|newprediction|cluster_size|
+-------------+------------+
|           12|          50|
|            1|         409|
|           13|         325|
|           16|         257|
|            6|         500|
|            3|         289|
|            5|         124|
|           19|          84|
|           15|         105|
|           17|         527|
|            9|         191|
|            4|         466|
|            8|         241|
|            7|         401|
|           10|          80|
|           11|         113|
|           14|         230|
|            2|           2|
|            0|         249|
|           18|         166|
+-------------+------------+
```

Figure 4.8: Formation of movie clusters

4.3 Description of Algorithm Used

4.3.1 K-means Clustering algorithm

When it comes to unsupervised learning k-means clustering algorithm is one the most used and popular algorithm, where data are grouped or clustered based on their similarities also defined as proximity of cluster centres known as centroid. The purpose of this k-means clustering is to partition the provided dataset into different (K) clusters. The number of clusters that is needed to be formed is defined by user. It basically works in iterative manner by assigning each data or data points to each cluster based on the similarities. Its implementation and computation are quite easy and efficient respectively.

The algorithm works as described below:

1. Choosing the required number of (K) clusters
2. Selecting data points from the given dataset and form initial centroid for clusters
3. Based on the nearest centroid assign each data points based on Euclidean distance
4. Based on the mean of datapoints, recalculate the centroids to assign all of the datapoints
5. Repeating the 3rd and 4th step until optimum clusters are formed or the maximum iteration has been carried out.
6. Output of number of (K) clusters based on the similarities of data.

The goal of K-means clustering algorithm is to minimize the distance between data points based on provided centroid. It has been widely used in the field of data science for segmentation, detection, recommendation and so on. But it sure does have some limitations at its initial phase while the clusters are being formed the clusters centroids doesn’t perform well giving non-regular and overlapping clusters.

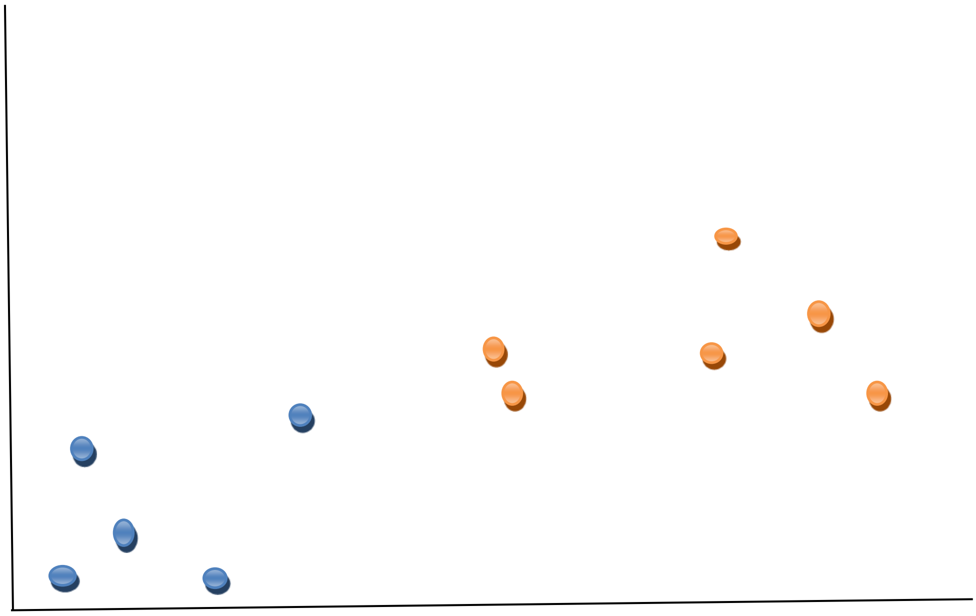
4.3.2 K-Nearest Neighbour algorithm

There are different algorithms that can be used to evaluate similarities between the movies such that we can provide list of similar movies in this recommendation project. Here K Nearest Neighbour (KNN) algorithm is used because it is very simple, easy to implement, low calculation time and fulfils the requirement. Brief description about the algorithm is given below.

The K-nearest neighbours (KNN) is a popular supervised machine learning algorithm. It can be used for solving different kinds of classification and regression problems related to machine learning. It works basically on distance format. The distance points close to each other tends to have more similar characteristics. The distance type may vary e.g., Manhattan or Euclidean etc. Selection of k values, which can be defined as the number of nearest neighbours should be considered as it plays vital role in making the recommendation. The K value may vary depending upon the type of data and the complexity of problem that is going to be solved. (Chatterjee, 2022).

Overall KNN is one of the most powerful problem-solving algorithms, when provided with great data, it can provide excellent results and has huge application in variety of domains.

Let us consider there are two variables, which are then plotted it may look like something that has been shown below figure 4.9.



Let’s classify the data points considering 'X' into "Blue" class or "orange" class. Let’s say the value of k is 3. Now, the calculation is carried out by measuring the distance form point x to all the nearest data points. As it has found the 3 data points from the x points. By encircling the data points the top three nearest datapoints are shown in figure below.

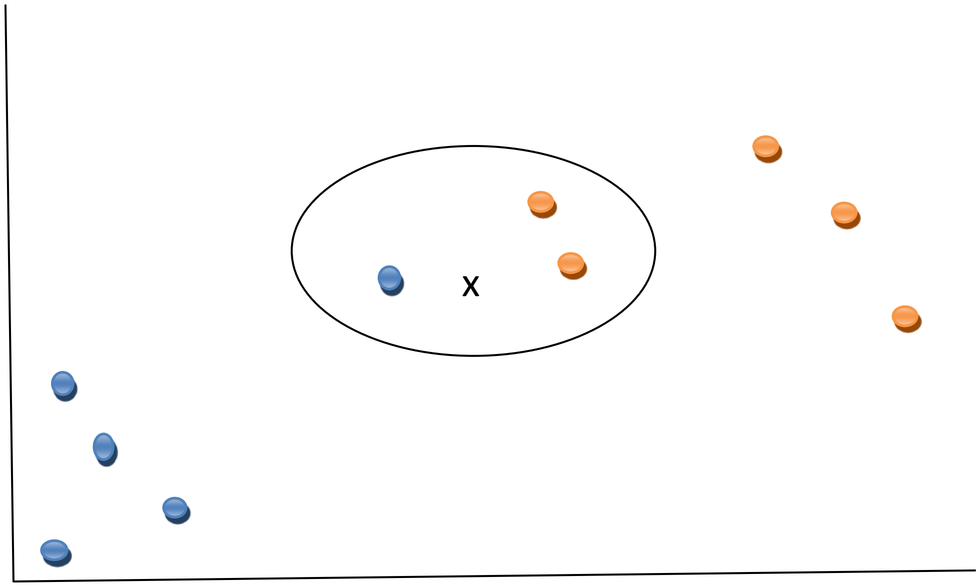


Figure 4.9: Working Mechanism of KNN Algorithm

Finally, after assigning the nearest datapoints to the specific class, where it belongs further predictions can be made. It is clearly visible that the data point had one blue and two orange data selected as k values. Based on the k values new set of data points will be formed for performing further classification and other process.

Finding K-nearest neighbours

Once the k value has been selected or the similarity metric has been decided, finding the k nearest data points for that value by calculating the distance between the datapoints, the data points having the smallest distance will be selected and will be considered the most similar point for that specific data point, which will be further used in the following process to come.

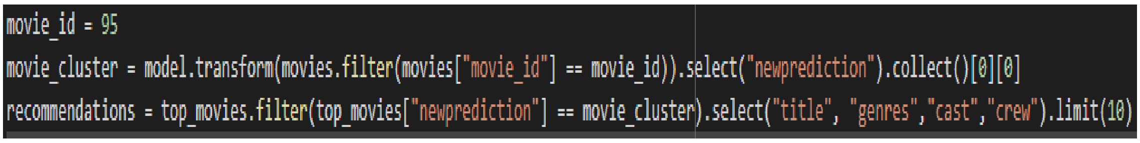
There are some things that needs to be kept in mind while selecting K values:

- Decrease in the value of K may make it less stable e.g., if K=1
- Likewise, If the value of k increases if may provide better and stable result but there will always be a limit where the K value might be too high and may start generating the error. That's the point where it must be known that the k value must be stable and should be stopped, It can be obtained through trial and error method.

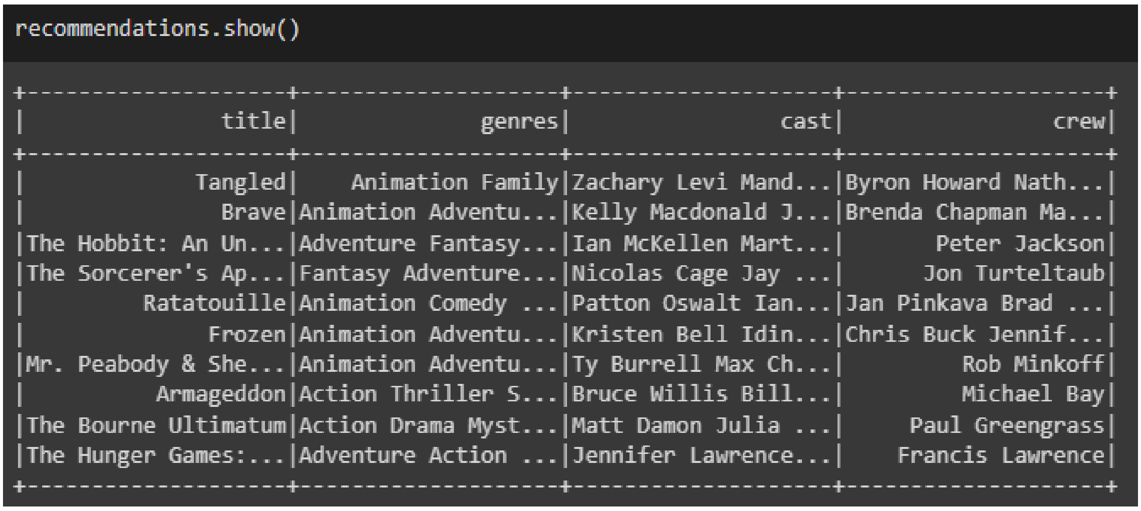
Finally, the Prediction can be made by aggregating the k nearest neighbour labels and selecting the label with higher frequency, selection to prediction and counting the frequency of data makes this algorithm best for classification and regression and best to solve the problems related to such.

Movie Recommendation using big data technology

As pySpark supports the pythonic libraries, it was also able to incorporate the k- mean clustering algorithm, which was able to form the clusters of movies, their genres, actors and directors based on the given features along with keywords and taglines for the movies. The system takes movie id as an input and provides the relevant recommendation of that movie based on the features provided. Not just the title of movies, it also provides the corresponding genres, actors name and the director's name of all the relevant recommendations made. Which provides the information related to movies along with recommendations. Processing speed, able to handle large amount of data, and ease of use makes the big data technology an excellent choice for movie recommendation system and more. The figure 4.10 shows the results produced by using pySpark along with K-mean clustering algorithm.



```
movie_id = 95
movie_cluster = model.transform(movies.filter(movies["movie_id"] == movie_id)).select("newprediction").collect()[0][0]
recommendations = top_movies.filter(top_movies["newprediction"] == movie_cluster).select("title", "genres","cast","crew").limit(10)
```



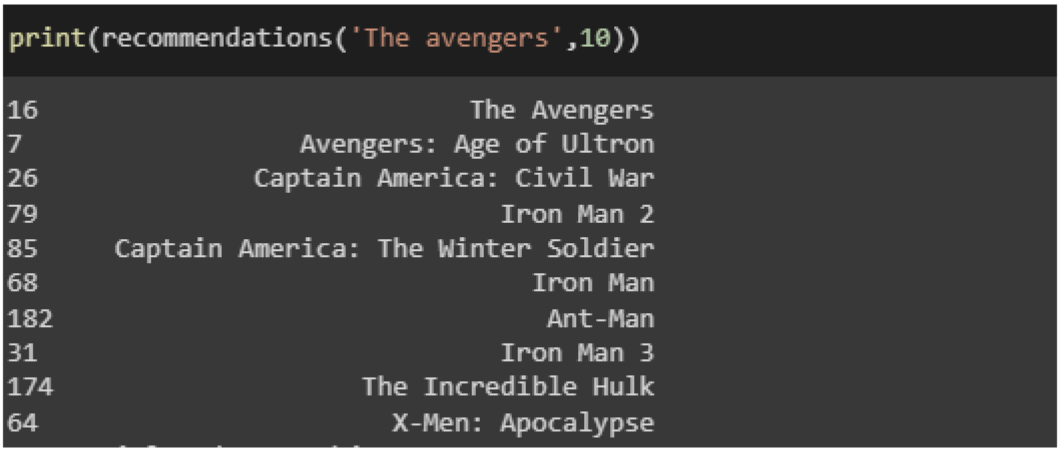
```
recommendations.show()

+--------------------+--------------------+--------------------+--------------------+
|               title|              genres|                cast|                crew|
+--------------------+--------------------+--------------------+--------------------+
|             Tangled|    Animation Family|Zachary Levi Mand...|Byron Howard Nath...|
|               Brave|Animation Adventu...|Kelly Macdonald J...|Brenda Chapman Ma...|
|The Hobbit: An Un...|Adventure Fantasy...|Ian McKellen Mart...|       Peter Jackson|
|The Sorcerer's Ap...|Fantasy Adventure...|Nicolas Cage Jay ...|      Jon Turteltaub|
|         Ratatouille|Animation Comedy ...|Patton Oswalt Ian...|Jan Pinkava Brad ...|
|              Frozen|Animation Adventu...|Kristen Bell Idin...|Chris Buck Jennif...|
|Mr. Peabody & She...|Animation Adventu...|Ty Burrell Max Ch...|         Rob Minkoff|
|          Armageddon|Action Thriller S...|Bruce Willis Bill...|         Michael Bay|
|The Bourne Ultimatum|Action Drama Myst...|Matt Damon Julia ...|     Paul Greengrass|
|The Hunger Games:...|Adventure Action ...|Jennifer Lawrence...|    Francis Lawrence|
+--------------------+--------------------+--------------------+--------------------+
```

Figure 4.10: Result of movie recommendation system using big data technology

Results for movie recommendation system using Jaccard Similarity

- Movie recommendation based on title

The system was able to provide the recommendation based on movie title, not just the similarities between the words but also the able to process the different features provided in the model, able to recognize the cast in the movies, the specific genres, directors of movies, keywords and taglines related to movies. For example, "The avengers" is the movie for which I want to get the recommendation for then the recommendations provided by the systems for top 10 movies is shown below in figure 4.11.

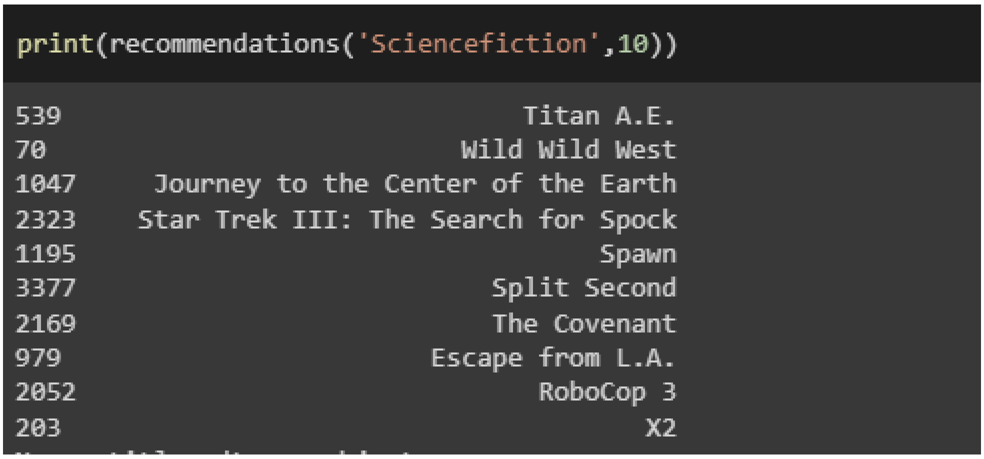


```
print(recommendations('The avengers',10))

16                              The Avengers
7                    Avengers: Age of Ultron
26                Captain America: Civil War
79                                Iron Man 2
85       Captain America: The Winter Soldier
68                                  Iron Man
182                                 Ant-Man
31                                Iron Man 3
174                      The Incredible Hulk
64                        X-Men: Apocalypse
```

Figure 4.11: Movie recommendation based on movie title

- Movie recommendation based on genres

Along with the title the system is able to recommend the movies based on genres, there are limited number of genres in movies, the hassle for the user can be reduced to provide recommendation based on their genre preferences. The system is designed to combine the different information regarding the movies by applying different techniques and is able to analyse the data for recommendation even in presence of large data. If user wants the result on Sci-Fi(science-fiction) movies the top 10 movies are shown below in figure 4.12.



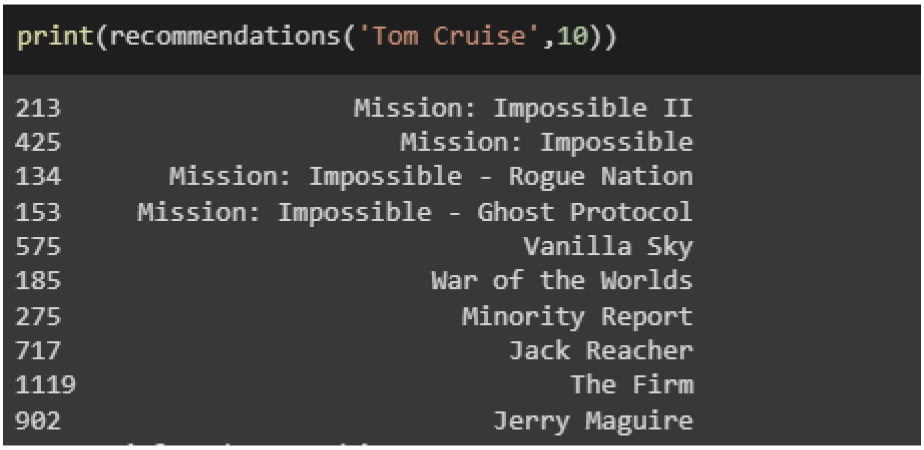
```
print(recommendations('Sciencefiction',10))
```

```
539                             Titan A.E.
70                          Wild Wild West
1047     Journey to the Center of the Earth
2323    Star Trek III: The Search for Spock
1195                                  Spawn
3377                           Split Second
2169                           The Covenant
979                        Escape from L.A.
2052                             RoboCop 3
203                                      X2
```

Figure 4.12: Movie recommendation based on movie genre

- Movie recommendation based on cast/actors

In this part of recommendation, the system is able to provide the recommendation of movies based on top three actors, everyone has their own favourite actors making it more relevant to users, that they can search the names of movies by their actor's name. If user searches the actor name "Tom Cruise" the results the user will get for the movies are shown below in the figure 4.13.



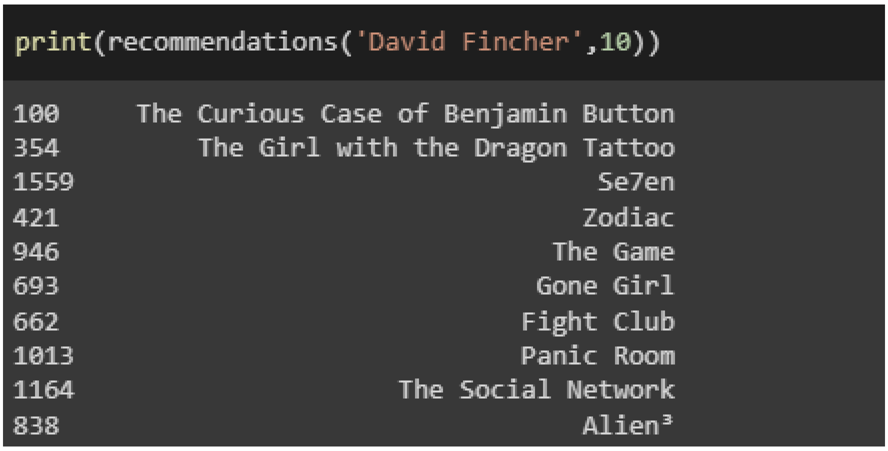
```
print(recommendations('Tom Cruise',10))
```

```
213                    Mission: Impossible II
425                       Mission: Impossible
134      Mission: Impossible - Rogue Nation
153     Mission: Impossible - Ghost Protocol
575                              Vanilla Sky
185                        War of the Worlds
275                          Minority Report
717                              Jack Reacher
1119                                 The Firm
902                             Jerry Maguire
```

Figure 4.13: Movie recommendation based on movie actors/cast

- Movie recommendation based on Crew (Director)

It is very rare that people search for the movies based on the directors or crew, but whoever tends to search the movies based on director can be movie enthusiast and specifically likes the content produced by that specific content creator. The system being able to provide the recommendation based on specific director helps that kind of viewer to learn and explore more about his/her contents. For example, if user search for “David Fincher” the users are likely get the recommendation Which are shown below in the figure 4.14.



```
print(recommendations('David Fincher',10))
```

```
100     The Curious Case of Benjamin Button
354         The Girl with the Dragon Tattoo
1559                                  Se7en
421                                  Zodiac
946                                The Game
693                               Gone Girl
662                              Fight Club
1013                             Panic Room
1164                     The Social Network
838                                  Alien³
```

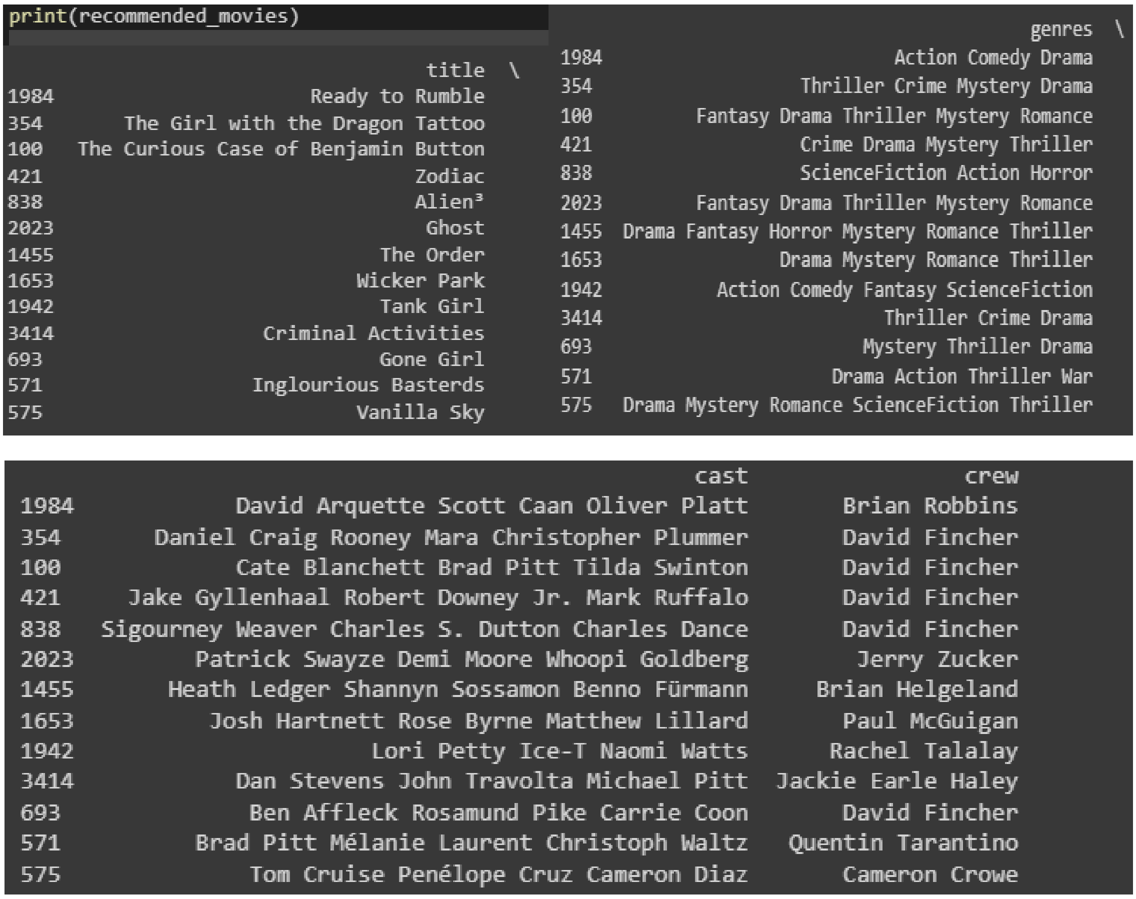
Figure 4.14: Movie recommendation based on movie director

Overall Movie recommendation system using KNN

With the combination of Jaccard similarity and K-nearest neighbour algorithms, it was possible to create the system, where the recommendation for each features described above like title, genres, cast and crew were able produce the recommendation based on movie id which was provide as an input. It is important to note that while Jaccard similarity can work with sparse data and provide better result, KNN needs clean and pre-processed data fit in the model, so the data was cleaned and well processed before feeding it to the model. KNN being used and popular algorithm it is good with small dataset and is able to provide excellent result with proper cleaned data. Able to combine Jaccard metric while fitting it to the KNN model it was able to provide better result and was able to handle large amount of dataset.

It is very important that the recommendation provided is accurate and efficient, as per this system, the thing that makes the movie recommendation system accurate is, if the user searches something and gets what he wants, or able to provide the recommendation based on their viewing history then the system is accurate. And the system is effective and efficient if it can satisfy the user with the given result.

In the system, movie id is provided as an input, each movie would be recognized as unique identifier, which is going to serve the purpose for retrieving the meaningful information for the movie content. The cast and the crew will have the information regarding actors and directors in the movie. Genre would provide the information regarding the characteristics, theme and plot of the movie. And as a recommendation system it would be helpful in providing the valuable information to the user, while search and know their interest while searching the movie and would provide them with more opportunities to explore new genres of movie that they would like. The outputs are shown below in figure 4.15.



```
print(recommended_movies)
                                      title  \
1984                        Ready to Rumble
354         The Girl with the Dragon Tattoo
100     The Curious Case of Benjamin Button
421                                  Zodiac
838                                  Alien³
2023                                  Ghost
1455                              The Order
1653                            Wicker Park
1942                              Tank Girl
3414                    Criminal Activities
693                               Gone Girl
571                    Inglourious Basterds
575                             Vanilla Sky
```

```
                                             genres  \
1984                             Action Comedy Drama
354                     Thriller Crime Mystery Drama
100          Fantasy Drama Thriller Mystery Romance
421                     Crime Drama Mystery Thriller
838                     ScienceFiction Action Horror
2023         Fantasy Drama Thriller Mystery Romance
1455  Drama Fantasy Horror Mystery Romance Thriller
1653                  Drama Mystery Romance Thriller
1942          Action Comedy Fantasy ScienceFiction
3414                            Thriller Crime Drama
693                           Mystery Thriller Drama
571                       Drama Action Thriller War
575   Drama Mystery Romance ScienceFiction Thriller
```

```
                                               cast                 crew
1984           David Arquette Scott Caan Oliver Platt        Brian Robbins
354      Daniel Craig Rooney Mara Christopher Plummer        David Fincher
100            Cate Blanchett Brad Pitt Tilda Swinton        David Fincher
421     Jake Gyllenhaal Robert Downey Jr. Mark Ruffalo        David Fincher
838   Sigourney Weaver Charles S. Dutton Charles Dance        David Fincher
2023        Patrick Swayze Demi Moore Whoopi Goldberg         Jerry Zucker
1455     Heath Ledger Shannyn Sossamon Benno Fürmann       Brian Helgeland
1653         Josh Hartnett Rose Byrne Matthew Lillard        Paul McGuigan
1942                      Lori Petty Ice-T Naomi Watts       Rachel Talalay
3414           Dan Stevens John Travolta Michael Pitt    Jackie Earle Haley
693              Ben Affleck Rosamund Pike Carrie Coon        David Fincher
571         Brad Pitt Mélanie Laurent Christoph Waltz    Quentin Tarantino
575              Tom Cruise Penélope Cruz Cameron Diaz        Cameron Crowe
```

Figure 4.15: Movie recommendation using KNN algorithm