PROJECT REPORT

Digital Marketing Analytics

- **Submitted by:**
  Rakshit Varma
  Department of Operational Research
  University of Delhi

INDEX

OBJECTIVE

To analyse campaign data for finding optimal age gender demographics based on Click-through rate (CTR) for a new campaign.

DATA VARIABLES

1. Campaign Data

campaign_name	title	date_start	age_group	gender
impressions	clicks	spend	interests	region

(Contained 80 unique movie titles)

2. Movies Data

title	runtime	rated	release_date	Year	plot
genre	language	country	writer	production	type

• Director	• Actors
• major_wins	• major_wins
• major_nominations	• major_nominations
• other_wins	• other_wins
• other_nominations	• other_nominations
• director_credits	• director_credits
• actor_credits	• actor_credits
	• instagram_followers

(Contained 5000 unique movie titles)

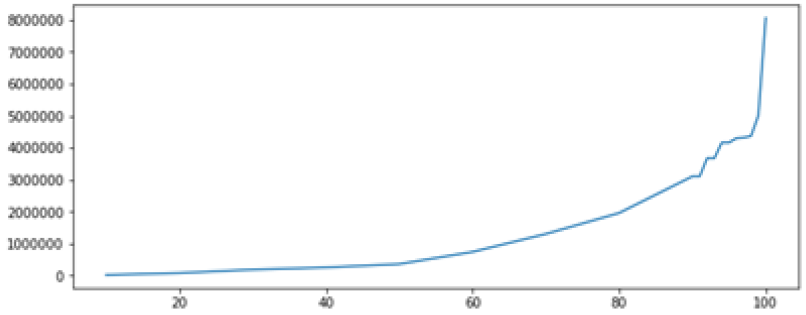
DATA PREPARATION

1. Those campaigns which weren't run on movies were removed. (3 campaigns)

2. The campaign data was then brought down to **3 Dimensions: Title, Age and Gender**. (Using the SUMIFS() function in Excel putting in the above three levels as criteria)

3. **Outlier treatment:** Observations in the clicks and impressions columns which were beyond the 99th percentile was replaced by the 99th percentile.

```
[22348.0, 81886.0, 188810.0, 256532.0, 363119.0, 743383.0, 1304724.0, 1959783.0, 3105314.0, 3105314.0, 3673535.0, 3673535.0, 41
62263.0, 4162263.0, 4300027.0, 4318781.399999984, 4367007.0, 5011866.0, 8061913.0]
[10, 20, 30, 40, 50, 60, 70, 80, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100]
```



4. **Added attributes** for the campaign data on movie level and compiled movie data for the database of 77 movies from campaign data + 5000 movies dataset obtained from IMDb.
   - For the campaign data movie properties such as genres, cast, director as such were obtained using The Open Movie Database (OMDb) API.
   - For the 4917 movies data these properties were already present in the dataset.
   - Dropping out the following variables from our datasets:

campaign_name	date_start	release_date	director	actor
plot	writer	production	type	

   - To obtain cast and director level information their IMDb pages were scraped in R.
     Prior to scraping a unique imdb_code was required to link to a specific person's page on IMDb website. This was obtained by using merge function in Python on two datasets one containing list of all cast and director names present in our dataset and other was an exhaustive list of all people obtained from IMDb as given below:

     name.basics.tsv.gz – Contains the following information for names:
     1. nconst (string) - alphanumeric unique identifier of the name/person
     2. primaryName (string)– name by which the person is most often credited

5. From the scraped pages of Cast and Directors, relevant information was obtained which included the following:
    a. Awards and Nominations
    b. Credits

    Hence format of obtained information was:

	Name	ID	Awards	Actorcredit	Director credit
0	Brigitte Bardot	nm0000003	Nominatedfor1BAFTAFilmAward.Another2win&1nomin...	(50credit)	NaN
1	John Belushi	nm0000004	Won1PrimetimeEmmy.Another1win&5nomination.	(13credit)	NaN
2	Ingmar Bergman	nm0000005	Nominatedfor9Oscars.Another75win&44nomination.	(15credit)	(70credit)
3	Humphrey Bogart	nm0000007	Won1Oscar.Another6win&5nomination.	(84credit)	NaN
4	Marlon Brando	nm0000008	Won2Oscars.Another28win&33nomination.	(46credit)	(1credit)

    With the help of **Regular Expressions**, awards and nominations count were obtained for a person following which the data obtained was:

	Name	Awardstsring	MWINS	MNOMS	WINS	NOMS	ActCreds	DirCreds
0	Brigitte Bardot	Nominatedfor1BAFTAFilmAward.Another2win&1nomin...	0	1	2	1	50	0
1	John Belushi	Won1PrimetimeEmmy.Another1win&5nomination.	1	0	1	5	13	0
2	Ingmar Bergman	Nominatedfor9Oscars.Another75win&44nomination.	0	9	75	44	15	70
3	Humphrey Bogart	Won1Oscar.Another6win&5nomination.	1	0	6	5	84	0
4	Marlon Brando	Won2Oscars.Another28win&33nomination.	2	0	28	33	46	1

    These counts were then used to create a comparative score for each identity whose formula is given by:

$$\boldsymbol{Actor(Director\ Score)}$$
$$= \frac{30 \times MWINS + 20 \times MNOMS + 10 \times WINS + 5 \times NOMS}{ActCreds + DirCreds}$$

6. **Instagram Follower Count** for each of the cast member was obtained with the help of an Instagram search query and then filtering out the results based on maximum followers for all profiles related to the specific actor/actress in R.

7. The Cast level information was then compiled to movie level for which the following attributes were made

$$\boldsymbol{Cast\ Score} = \sum Actor\ Score$$
$$\boldsymbol{Total\ Follower\ Count} = \sum Actor\ Follower\ Count$$

8. **Normalizing numerical variables** which included:
    a. Runtime
    b. Cast Score
    c. Director Score
    d. Total Follower Count

    By using the following formula:

$$Normalised\ variable = \frac{\text{Xi} - \text{Mean(Xi)}}{\sqrt{Variance(Xi)}} + 3$$

9. The final data structure containing movies from both the datasets looked like this:

Title	Total.Fol	Cast.Score	Dir.Score	Runtime	Year	Rating	language	country	Action	...	Family	Western	Musical	Film.Noir	History	War	Sport
Avatar	0.941728	0.001794	0.100922	178	2009	PG-13	English	USA	1	...	0	0	0	0	0	0	0
Pirates of the Caribbean: At World's End	0.850448	0.015611	0.037363	169	2007	PG-13	English	USA	1	...	0	0	0	0	0	0	0
Spectre	0.179707	0.006289	0.144156	148	2015	PG-13	English	UK	1	...	0	0	0	0	0	0	0
The Dark Knight Rises	0.789324	0.025887	0.484762	164	2012	PG-13	English	USA	1	...	0	0	0	0	0	0	0

Having 34 columns(variables) out of which we have 25 different genres as separate variables whose list is given below:

Action	Documentary	Adventure	Drama	Animation
Comedy	Mystery	Fantasy	Crime	Biography
Sci.Fi	Horror	Romance	Thriller	Game Show
Family	Western	Musical	Film.Noir	History
War	Sport	Reality.Tv	Short	News

Moreover, the data was further split into 3 parts:

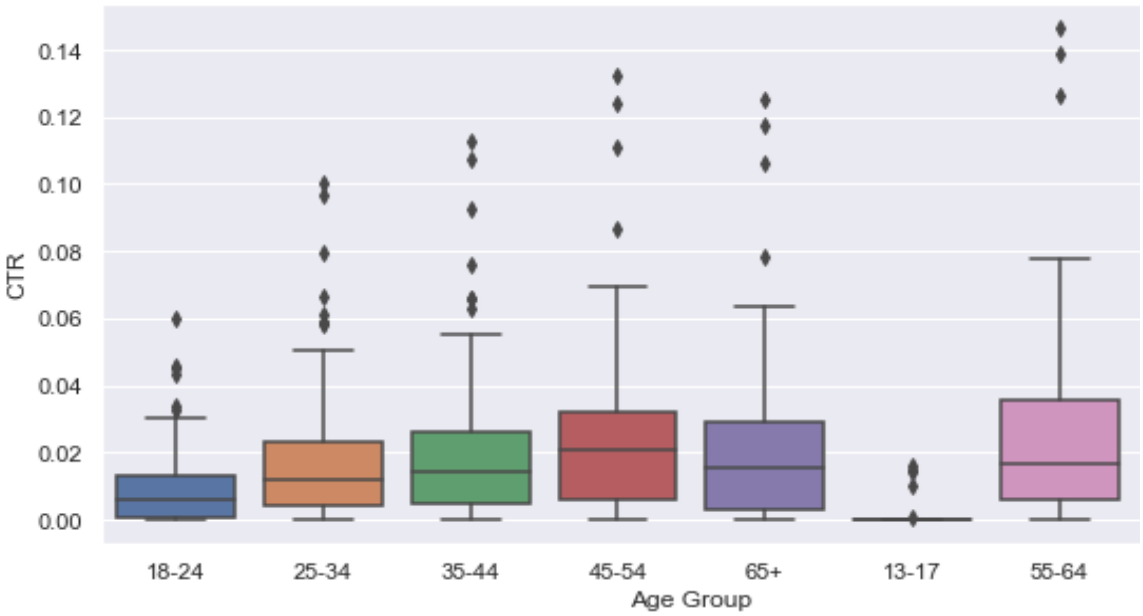
- **Training set:** 5000 movies from IMDb dataset.
- **Validation set:** 3 randomly chosen movies from the campaign dataset.
- **Test set:** Remaining 74 movies from the campaign dataset.

EXPLORATORY DATA ANALYSIS

Using Python's **seaborn** library and **Tableau**, our EDA objective was to understand how variables in this dataset relate to the CTR for campaigns and obtain a visual understanding of those patterns in data.

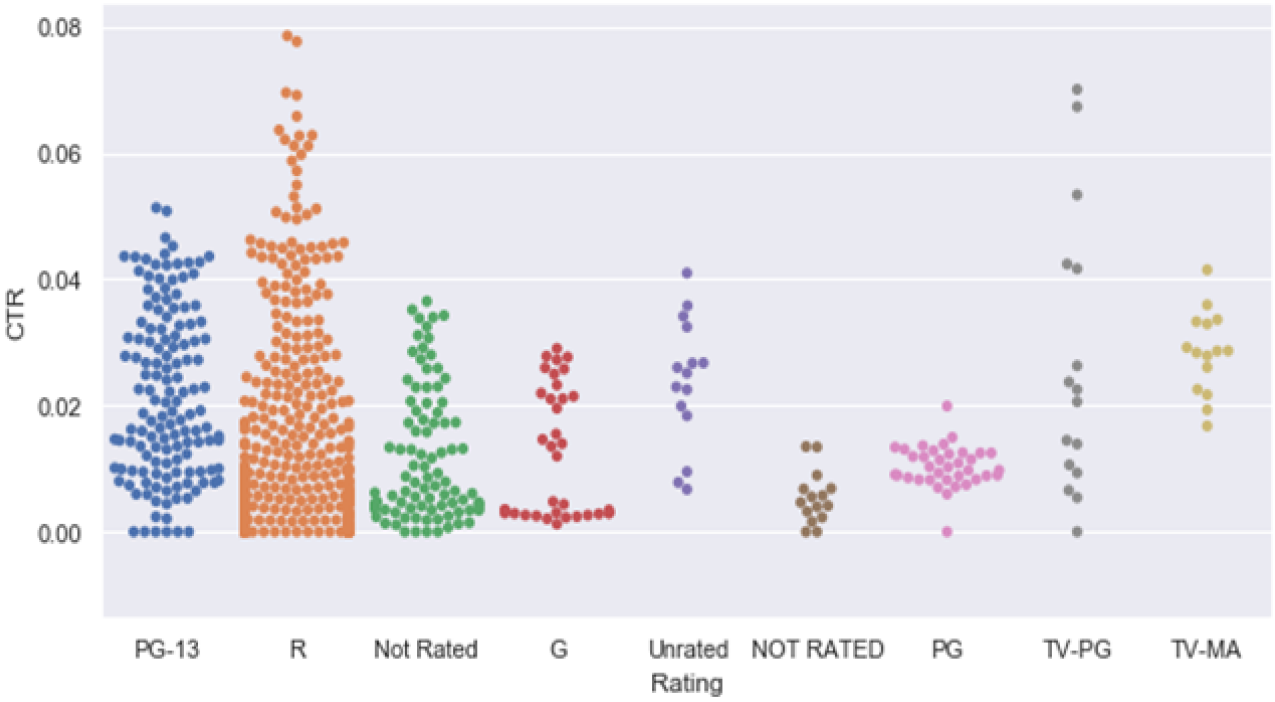
The **box-and-whisker plot** that we've implemented below are commonly used for visualizing relationships between numerical variables and categorical variables. In our case we're studying the variation between CTR (numerical) and other categorical variables such as Age Group, Rating and Genre.

- **Age Group vs CTR**



From the above plots we infer that the CTR is not distributed uniformly across different age groups and it is the highest in the middle age groups and tends to decrease as we move towards the extreme ends on either side, among all our campaigns. The dots outside the box represent outliers in the data. The line between the box represents the median value among the data points.

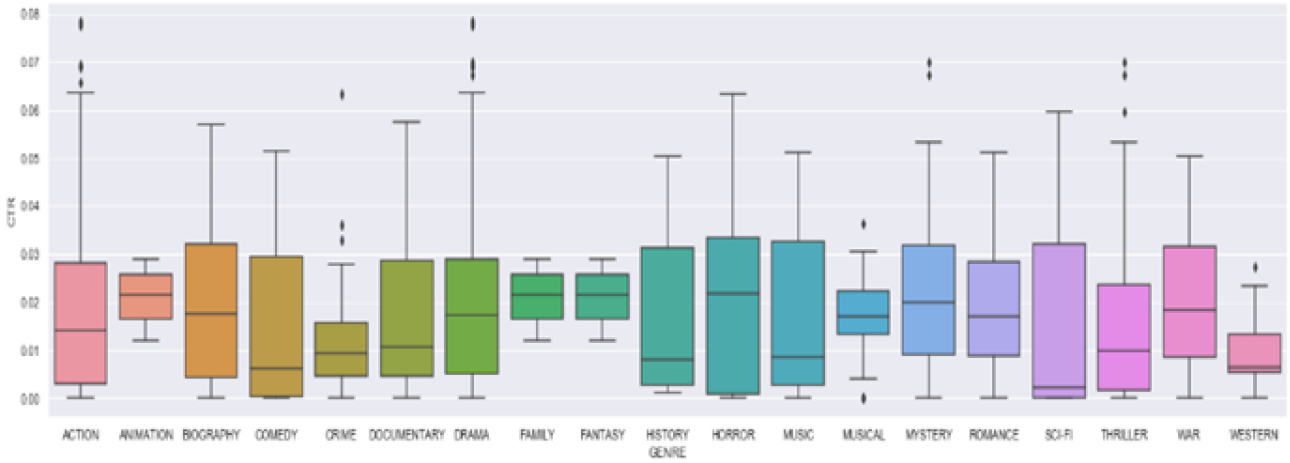
- **Movie Rated vs CTR**



The plot above signifies that in our campaign dataset we have many campaigns which correspond to movies Rated R followed by PG-13.

It also shows that CTR achieves a comparatively higher values among those campaigns which were run movies rated R as well as TV-PG.

- **Genre vs CTR**



In the above plot we see quite a varied distribution of CTR across different genres hence it made sense to club the data together based on this factor among others to reduce complexity in analysis.

- **CTR Distribution across Age Group, Genre and Gender**

		Age Group						
Gender	Genre1	13-17	18-24	25-34	35-44	45-54	55-64	65+
female	Action	0.00035	0.00086	0.02312	0.02939	0.03388	0.05004	0.04564
	Adventure		0.01954	0.01216	0.01295	0.01415	0.02765	0.02496
	Animation		0.01954	0.02193	0.02721	0.02900	0.02765	0.02496
	Biography	0.00000	0.01398	0.02076	0.02766	0.03355	0.03839	0.03325
	Comedy	0.00309	0.01246	0.01595	0.02051	0.02712	0.02993	0.02416
	Crime	0.00035	0.01105	0.01425	0.01795	0.02193	0.02520	0.01766
	Documentary		0.00822	0.01871	0.02516	0.03001	0.03083	0.02784
	Drama	0.00309	0.01256	0.01683	0.02081	0.02811	0.03282	0.03040
	Family		0.01954	0.02193	0.02721	0.02900	0.02765	0.02496
	History			0.01254	0.02068	0.02929	0.01587	0.01219
	Horror	0.00035	0.00256	0.02187	0.03086	0.03860	0.04054	0.04059
	Musical		0.01621	0.01549	0.01665	0.02415	0.03067	
	Mystery		0.00967	0.02184	0.02799	0.02982	0.01780	0.01606
	Romance		0.00597	0.02040	0.02617	0.03135	0.03366	0.03313
	Sci-Fi		0.04578	0.04557	0.05135			
	Thriller		0.00781	0.00827	0.01185	0.01552	0.02512	
male	Action		0.00481	0.01177	0.01584	0.02137	0.04605	0.04507
	Adventure		0.00834	0.00826	0.00990	0.01126	0.02779	0.02322
	Animation		0.02106	0.02101	0.02145	0.02577	0.02779	0.02322
	Biography	0.00000	0.01414	0.02313	0.02935	0.03537	0.04279	0.03095
	Comedy	0.01586	0.00885	0.01433	0.01907	0.02763	0.03561	0.02489
	Crime		0.00562	0.00919	0.01222	0.01544	0.02501	0.01921
	Documentary		0.01123	0.03299	0.03546	0.03192	0.02760	0.02276
	Drama	0.01448	0.01264	0.01444	0.01779	0.02264	0.02770	0.02186
	Family		0.02106	0.02101	0.02145	0.02577	0.02779	0.02322
	History			0.01036	0.01676	0.02189	0.02541	0.02363
	Horror	0.00000	0.01652	0.01959	0.02384	0.02566	0.03549	0.03570
	Musical		0.01452	0.01486	0.01428	0.02219	0.02581	
	Mystery	0.01447	0.01569	0.01365	0.01907	0.01713	0.01577	0.01524
	Romance		0.00745	0.00819	0.01086	0.01440	0.01603	0.01620
	Sci-Fi		0.03245	0.03138	0.02917			
	Thriller		0.00499	0.00985	0.01219	0.01740	0.04095	

In the heatmap given above we see the CTR distribution among different age groups and genres for male and female respectively.

Collectively females are shown to give better response to our campaigns than males.

While their values are varying across genres. For example, males give better response to movies with Horror as their Genre in the age group 13-24 as compared to Females.

While Females showed better CTR values for campaigns with Romance as a Genre than males in the higher age groups of 25 and above.

REGRESSION TECHNIQUES

Based on the results obtained in Exploratory Data Analysis, we moved on to Regression techniques with the objective of finding expected CTR for a specific set of demographics and then analysing the same for finding the most optimal value.

• MULTIPLE LINEAR REGRESSION

Analysis done in Python, libraries used: pandas, sklearn, statsmodels.

- **Response Variable:** CTR

- **Explanatory Variables:** Age-Gender, Genre, Cast and Director Scores, Rating, Runtime, Instagram Followers.

- **Observations:**
  - R^2 - 78.4%
  - Adjusted R^2 - 76.2%.
    However, the p-values associated with the individual regressors showed that nearly all of them were **insignificant.**

• DECISION TREE REGRESSION

Analysis done in R, packages used: dplyr, rpart, rpart.plot

- **Response Variable:** CTR

- **Explanatory Variables:** Age-Gender, Genre, Cast and Director Scores, Rating, Runtime, Instagram Followers.

- **Observations:**
  - No clear split of CTR
  - Cast scores emerged as primary point of split
  - Genre did not appear as very significant – counterintuitive
  - Age – gender appeared to have **no** effect.

❖ CONCUSIONS

The result from both the techniques were inconclusive, since no clarity was observed regarding the correlation between variables in Multiple Linear Regression and the Decision Tree model as well which gave primary split(emphasis) on less important factors without emphasizing much on the main variables affecting CTR.

Hence, we moved on to Unsupervised Learning: Clustering Techniques.

Unsupervised learning is a branch of machine learning that learns from test data that has not been labelled, classified or categorized. Instead of responding to feedback, unsupervised learning identifies commonalities in the data and reacts based on the presence or absence of such commonalities in each new piece of data.

CLUSTERING TECHNIQUES

Clustering is a technique that involves the grouping of data points. Given a set of data points, we can use a clustering algorithm to classify each point into a specific group. In theory, data points that are in the same group should have similar properties and/or features, while those in different groups should have highly dissimilar properties and/or features. The similarity between points is usually quantified by a distance metric based on some sort of feature variable set.

• HEIRARCHICAL CLUSTERING

The analysis was done in R, packages used: dplyr, cluster (to perform different types of hierarchical clustering) and functions used daisy(), diana(), hclust(), clusplot()

The Clustering algorithm requires 3 main steps listed as below:

3. Calculating the Dissimilarity Matrix
4. Choosing the clustering method
5. Assessing the clusters

Dissimilarity matrix is a mathematical expression of how different, or distant, the points in a data set are from each other. For our case to calculate the matrix we'll be using Gower distance which helps calculate the distance between categorical variables.

- daisy: Dissimilarity matrix calculation
  Computes all the pairwise dissimilarities (distances) between observations in the data set. In our case of mixed data types, metric = "gower" is used.

  Function:
  ```
  daisy (x, metric = c ("euclidean", "manhattan","gower"))
  ```

  where,
  - x: numeric matrix or data frame, of dimension n x p, say. Dissimilarities will be computed between the rows of x.
  - metric: character string specifying the metric to be used.

1. AGGLOMERATIVE CLUSTERING

Agglomerates individual data points stepwise to lesser numbers of clusters. Works on continuous and discrete variables.

- hclust: Hierarchical cluster analysis on a set of dissimilarities and methods for analyzing it.

  Function:
  ```
  hclust (d, method = "complete")
  ```

  where,
  - d: a dissimilarity structure as produced by dist.
  - method: the agglomeration method to be used. This should be one of "ward.D", "ward.D2", "single", "complete", "average", "mcquitty", "median" or "centroid".

2. DIVISIVE CLUSTERING

Stepwise splits the dataset to smaller clusters. Not effective in case of multivariable hierarchies.

- diana: Computes a divisive hierarchical clustering of the dataset returning an object of class diana.

Function:

```
diana (x, diss = inherits (x, "dist"), metric = "euclidean")
```

where,

- x: data matrix or data frame, or dissimilarity matrix or object, depending on the value of the diss argument.
- diss: logical flag: if TRUE (default for dist or dissimilarity objects), then x will be considered as a dissimilarity matrix. If FALSE, then x will be considered as a matrix of observations by variables.
- metric: character string specifying the metric to be used for calculating dissimilarities between observations. If x is already a dissimilarity matrix, then this argument will be ignored.

• K-MEANS CLUSTERING

K-Means starts by randomly defining *k* centroids. From there, it works in iterative (repetitive) steps to perform two tasks:

1. Assign each data point to the closest corresponding centroid.
2. For each centroid, calculate the mean of the values of all the points belonging to it. The mean value becomes the new value of the centroid.

The analysis was done in R by using the kmeans() function.
Works on Euclidean distance, hence, was not appropriate for our data.

- kmeans: Performs k-means clustering on a data matrix.

  Function:

  ```
  kmeans (x, centers)
  ```

  where,

  - x: numeric matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns).
  - centers: either the number of clusters, say k, or a set of initial (distinct) cluster centres.

- # K-PROTOTYPE CLUSTERING

For mixed data (both numeric and categorical variables), we use **k-prototypes** which is basically **combining k-means and k-modes clustering algorithms**. For numeric variables, it runs **Euclidean distance**. For categorical variables, k-modes use **Simple Matching**. In k-modes, modes act as centroids (i.e. cluster center). It minimizes the sum of distances between each object in the cluster and centroid. The algorithm has mainly the following steps:

1. Cluster prototypes are computed as cluster means for numeric variables and modes for categorical variables. Randomly select k objects as the initial cluster centers.
2. Calculate the distances between each object and the cluster prototype; assign the object to the cluster whose center has the shortest distance to the object; repeat this step until all objects are assigned to clusters. Update the prototype of the cluster after each allocation.
   Where,
   *dist(x,y) = **euclid**(numeric_variables) + λ* **simple matching**(categorical_variables)*

Package Used: clustMixType

Function used: kproto: Computes k-prototypes clustering for mixed-type data.

```
kproto(x, k, lambda = NULL)
```

where,

- x: Data frame with both numerics and factors.
- k: Either the number of clusters, a vector specifying indices of initial prototypes, or a data frame of prototypes of the same columns as x.
- lambda: Parameter > 0 to trade-off between Euclidean distance of numeric variables and simple matching coefficient between categorical variables.

❖ CONCLUSIONS

1. **Hierarchical Clustering – Agglomerative and Divisive** did not result in well defined clusters. There was a lot of overlaps between data points in all clusters. Each cluster had similar properties to some extent with no clear distinction between them, which was in contradiction to our objective.

2. **K-Means Clustering** works best for Euclidean distances; however, we computed the distance using Gower metric. Clusters obtained had noticeable properties, it was an improvement to those obtained through Hierarchical Clustering.

3. **K-Prototype Clustering** is specifically applied while working for mixed data types. And it was found to be more effective in creating better clusters for the schema in this analysis among all the techniques that we applied.
   We got distinct set of clusters which were homogenous within and heterogeneous among themselves.

CLUSTER PROPERTIES

Optimum number of clusters obtained through k-prototype was 4 and we observed the following set of properties for them.

• GENRES

	1	2	3	4
Drama	1,907	132	219	336
Romance	585	70	394	58
Biography	252	9	17	15
Comedy	428	106	1,233	105
Documentary	59	7	47	8
Family	98	86	359	3
Fantasy	161	175	232	42
Animation	29	41	168	4
Action	158	559	118	318
Adventure	126	539	235	23
Sci.Fi	149	240	128	99
Crime	186	42	101	560
Thriller	337	229	31	814
Horror	215	44	100	206
Mystery	201	47	35	217

Train Set

	1	2	3	4
Drama	31.00	2.00	2.00	4.00
Romance	6.00	1.00	2.00	1.00
Biography	8.00	0.00	0.00	0.00
Comedy	4.00	0.00	9.00	1.00
Documentary	5.00	0.00	7.00	0.00
Family	0.00	0.00	1.00	0.00
Fantasy	0.00	0.00	1.00	0.00
Animation	0.00	0.00	1.00	0.00
Action	3.00	2.00	0.00	2.00
Adventure	0.00	2.00	1.00	0.00
Sci.Fi	1.00	0.00	0.00	1.00
Crime	5.00	2.00	1.00	7.00
Thriller	2.00	2.00	3.00	10.00
Horror	1.00	0.00	2.00	3.00
Mystery	3.00	1.00	1.00	3.00
War	2.00	1.00	0.00	0.00

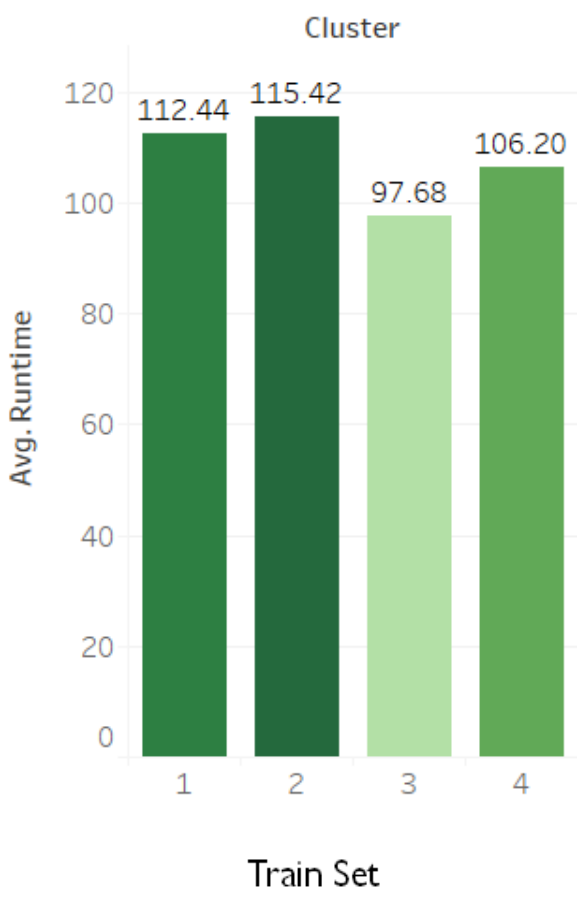
Test Set

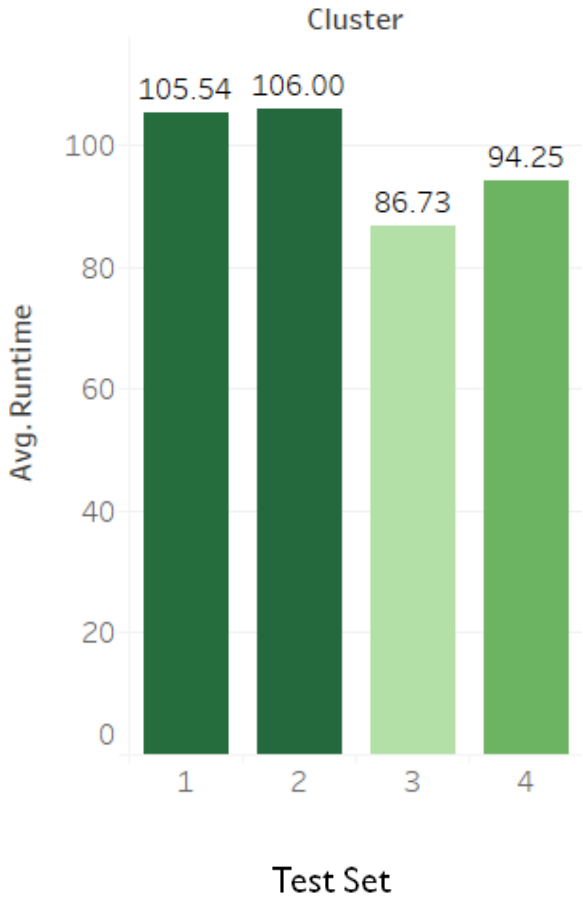
The above heatmap shows the count values for top 15 genres for each cluster in our dataset. Each cluster was shown to have datapoints in majority related to some specific genre which are correlated. The table below shows the most prominent ones occuring in each cluster:

Cluster 1	Cluster 2	Cluster 3	Cluster 4
Drama	Action	Comedy	Crime
Romance	Adventure	Documentary	Thriller
Biography	Sci-Fi	Animation	Horror
		Family	Mystery

- **RUNTIME**

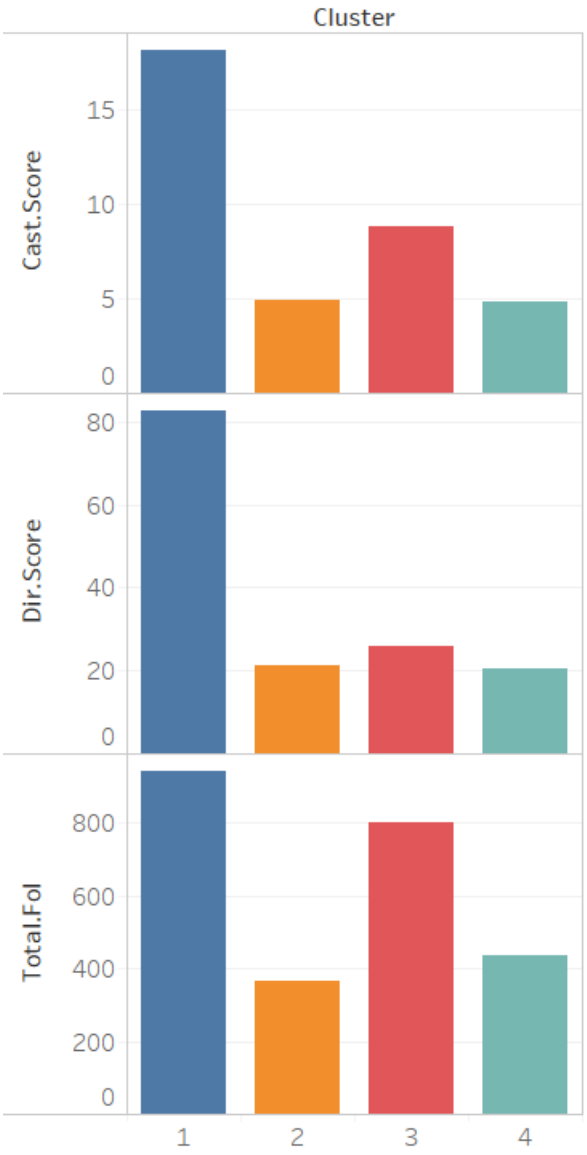
Different set of properties were obtained in clusters with Runtime.
Cluster 3 was shown to have the lowest average Runtime while Cluster 2 had the highest. Cluster 1 had a slightly lesser value than the second one. These results are depicted in the bar graph as shown below:





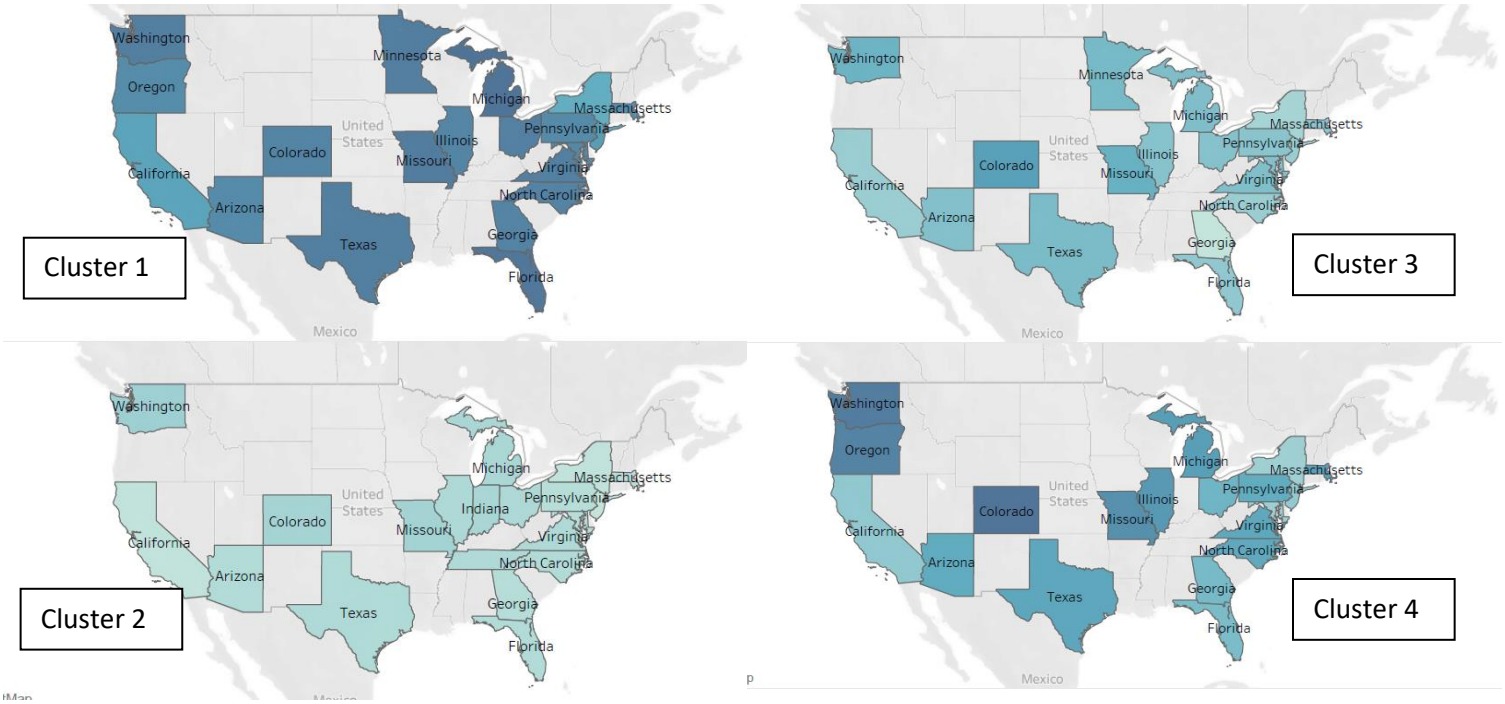
- # CAST SCORES, DIRECTOR SCORES AND INSTAGRAM FOLLOWERS

1. The clusters showed a linear decreasing trend for Cast Scores with Cluster1 having the highest value and Cluster 4 having the lowest
2. The Director Score had the highest value for Cluster1 after which it started decreasing with Cluster 3 having the lowest value and it rose again for cluster 4.
3. In terms of Total Instagram Followers Count, Cluster 3 ranked first and Cluster 1 having the lowest value.
4. The above-mentioned conclusions are presented in a Bar Graph as below:



- **REGIONS**

With respect to regions we got a higher CTR value for regions which comparatively much less impressions, hence ranking them based on CTR would be inconclusive.
There was a high degree of overlap between regions in all clusters as well and CTR was relatively moderate in regions with highest number of impressions.



The above heatmap of United States shows the CTR distribution of top 20 regions in each cluster. The distribution of impressions and CTR in these regions is shown below, these are sorted on impressions:

Cluster 1		
Region	SumImp	CTR
CALIFORNIA	31614965	0.020097
NEW YORK	22362609	0.01825
TEXAS	15098924	0.027275
FLORIDA	8604163	0.028966
ILLINOIS	7976490	0.026217
NEW JERSEY	7048445	0.021429
PENNSYLVANIA	6717142	0.02508
GEORGIA	4992882	0.026258
OHIO	4974351	0.026917
MASSACHUSETTS	4768129	0.024563
MICHIGAN	4350825	0.02939
NORTH CAROLINA	4315876	0.026446
WASHINGTON	4009254	0.027327
VIRGINIA	3762134	0.0271
ARIZONA	3552074	0.024985
MARYLAND	3094687	0.025468
COLORADO	3041184	0.026157
OREGON	2911831	0.024606
MISSOURI	2910327	0.027828
MINNESOTA	2783410	0.027401

Cluster 2		
Region	SumImp	CTR
CALIFORNIA	1344228	0.006845
TEXAS	957833	0.008817
FLORIDA	694984	0.00855
NEW YORK	678556	0.006882
PENNSYLVANIA	384527	0.00796
ILLINOIS	356090	0.009138
GEORGIA	327156	0.008085
NORTH CAROLINA	308068	0.008605
OHIO	305863	0.009105
NEW JERSEY	302266	0.006508
MICHIGAN	259784	0.0095
VIRGINIA	256624	0.008888
MASSACHUSETTS	221194	0.007749
ARIZONA	212929	0.008599
WASHINGTON	209383	0.011037
TENNESSEE	203831	0.008924
INDIANA	188290	0.009724
MARYLAND	177818	0.008109
MISSOURI	174672	0.010379
COLORADO	160388	0.01015

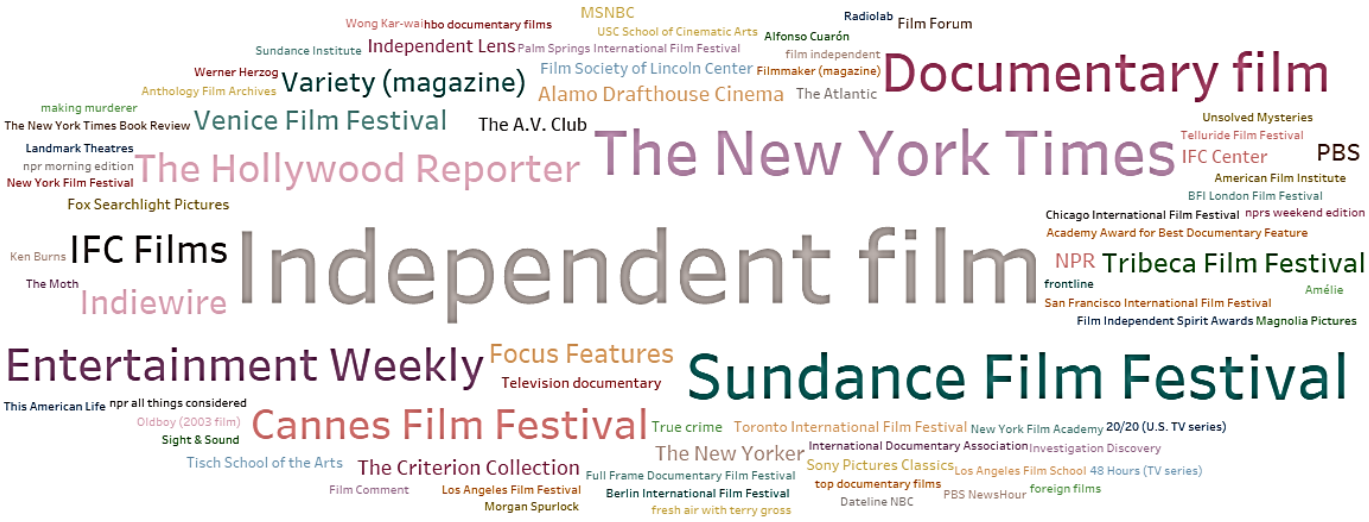
Cluster 3		
Region	SumImp	CTR
CALIFORNIA	13450053	0.010871
NEW YORK	8908219	0.010156
TEXAS	6068874	0.012647
ILLINOIS	4609340	0.012021
FLORIDA	4447787	0.011391
MASSACHUSETTS	3315918	0.011719
PENNSYLVANIA	3229097	0.011792
WASHINGTON	2906674	0.013418
NEW JERSEY	2649927	0.010855
GEORGIA	2541756	0.008561
OHIO	2455784	0.01207
VIRGINIA	2345599	0.012387
ARIZONA	2315353	0.012196
MINNESOTA	2191026	0.012863
NORTH CAROLINA	2074419	0.010836
MICHIGAN	1939405	0.012443
MARYLAND	1887741	0.012165
COLORADO	1611605	0.015134
WASHINGTON, DIS	1522115	0.013004
MISSOURI	1404296	0.013697

Cluster 4		
Region	SumImp	CTR
NEW YORK	4273580	0.010615
CALIFORNIA	4234800	0.011396
TEXAS	2314715	0.014568
GEORGIA	1345609	0.012851
ILLINOIS	1300188	0.015895
PENNSYLVANIA	1160363	0.014065
FLORIDA	1030494	0.012624
NEW JERSEY	949084	0.011276
OHIO	607077	0.013295
MARYLAND	567383	0.01061
ARIZONA	497433	0.014171
MICHIGAN	447163	0.01522
NORTH CAROLINA	442201	0.014407
MASSACHUSETTS	437057	0.015138
VIRGINIA	434763	0.014449
WASHINGTON	403093	0.018187
WASHINGTON, DIS	398171	0.011166
MISSOURI	389596	0.016586
COLORADO	362984	0.019034
OREGON	297529	0.017702

The regions in all clusters are almost same but their ranking is varied slightly.
Cluster 2 has a different distribution of CTR than other clusters where it has shown much less CTR comparatively, in the region with the highest impressions.
Rest of the clusters do not differ much in the distribution.

- **INTERESTS**

There were a significantly large number of unique interests corresponding to our dataset to obtain a clear conclusion with respect to clusters. Also, the major interests with high impressions had an overlap among clusters. The word cloud associated with interests with their sizes corresponding to their impressions is given below:



And the distribution of these interests among the 4 clusters is shown as:

Cluster 1			Cluster 2			Cluster 3			Cluster 4		
Interest	SumImp	CTR	Interest	SumImp	CTR	Interest	SumImp	CTR	Interest	SumImp	CTR
Independent film	97738739	0.028133	Action film	13132785	0.005362	Democratic Party (United Sta	76600497	0.022688	Independent film	10802971	0.027123
Horror film	67359380	0.050624	Drive (2011 film)	6427795	0.005553	Women`s rights	35664817	0.028268	Michael B. Jordan	5640985	0.030067
Thriller (genre)	64280019	0.053012	Dwayne Johnson	6140314	0.005939	Feminism	26937940	0.02559	Chadwick Boseman	4536149	0.031343
Feminism	33640175	0.010856	The Lord of the Rings	5228938	0.00566	Film director	24388886	0.026703	Community	3244163	0.018011
The New York Times	30807181	0.046237	The Lord of the Rings (film series)	4789397	0.005224	Michelle Obama	21875481	0.029941	Barack Obama	2944530	0.01766
Sundance Film Festival	24975008	0.031378	The Fast and the Furious	4436449	0.004673	The Daily Show	20283937	0.026436	Sundance Film Festival	2812986	0.035389
Action film	23207348	0.047863	Jason Statham	3868605	0.004531	Comedy	17995110	0.036813	Ryan Coogler	2514830	0.03185
Documentary film	23107679	0.063133	Vin Diesel	2960409	0.005217	hillary clinton	17551693	0.027975	Bernie Sanders	1876028	0.01497
IGN	21945882	0.042267	The Hobbit	1721061	0.006239	Rick and Morty	13718154	0.011336	Black Lives Matter	1742438	0.010114
Bustle	19681052	0.016537	Fast & Furious (2009 film)	1542757	0.005349	Entertainment Weekly	12163969	0.025005	Thriller (genre)	1670579	0.035471
Rob Zombie	17274521	0.028592	Pirates of the Caribbean (film series)	1013068	0.008578	Sundance Film Festival	11774342	0.040026	IFC Films	1650355	0.056936
Film director	16616988	0.024077	Mission: Impossible	1003572	0.007675	The New York Times	11728397	0.009052	Documentary film	1586976	0.02828
Elizabeth Warren	15319008	0.009191	The Fast and the Furious (2001 film)	609031	0.006543	International Women`s Day	10871336	0.02846	Elizabeth Warren	1355398	0.014817
Female Entrepreneur Associat	14850974	0.005057	Mission: Impossible (film series)	591751	0.012088	Independent film	10042352	0.069059	Spike Lee	1327462	0.035032
Psychological thriller	14586233	0.049551	Orlando Bloom	382635	0.015349	The Rachel Maddow Show	9100004	0.036951	Horror film	1292779	0.042518
Cult film	14017819	0.044867	Bourne (film series)	293921	0.007192	The Office (U.S. TV series)	8665230	0.01636	Activism	1216003	0.017433
Cannes Film Festival	13229873	0.017063	Crank (film)	293704	0.006312	Gender equality	8390045	0.032847	Psychological thriller	1101848	0.045826
Quentin Tarantino	12282308	0.040615	Die Hard	277957	0.011761	LGBT	8330547	0.047643	Indiewire	863008	0.050347
BET	9999710	0.051073	The Hobbit (film series)	251614	0.016092	Social justice	8043495	0.034637	Michelle Obama	861741	0.016129
Entertainment Weekly	9806951	0.033227	The Expendables (2010 film)	197023	0.009131	Judaism	7222863	0.016358	PBS	859710	0.027944
Theatre	8878919	0.051954	The Hobbit: An Unexpected Journey	196883	0.015806	Rock music	6979687	0.049303	Yorgos Lanthimos	854621	0.055533
The Hollywood Reporter	8801931	0.039383	The Hobbit: The Battle of the Five Armies	184030	0.015459	The Hollywood Reporter	6708251	0.027312	Film	842072	0.010111
The Washington Post	8089667	0.064529	The Hobbit: The Desolation of Smaug	178758	0.015104	Drama	6696855	0.050691	Black Mirror (TV series)	740868	0.048918
Complex (magazine)	7955958	0.031688	The Expendables (film series)	176308	0.008383	Parks and Recreation	6373387	0.037222	Quentin Tarantino	727552	0.045623
Michelle Obama	7734988	0.009569	Spectre (2015 film)	149058	0.009137	Documentary film	6031401	0.133598	Criminal Minds	705128	0.040869
Essence (magazine)	7726590	0.046638	Pirates of the Caribbean	143693	0.00666	Alternative rock	5877130	0.046954	Cannes Film Festival	686931	0.029579
IFC Films	7724797	0.068273	Jack Ryan (character)	124774	0.020301	Israel	5784425	0.013789	Ava DuVernay	643250	0.03604
Pulp Fiction	7614802	0.044648	7	108586	0.006907	Left-wing politics	5696393	0.036618	Action film	611196	0.017194
Democratic Party (United State	7612240	0.051398	Fast Five	105493	0.003195	Mother Jones (magazine)	5339630	0.041481	Steve McQueen	603726	0.028049
Motorsport	7103241	0.0181	Tom Clancy	105178	0.008471	Adult Swim	5287154	0.024271	True Detective (TV series)	600840	0.045651

CONCLUSION

• AGE & GENDER DEMOGRAPHICS

Following were the age gender demographics that we obtained for the 4 clusters respectively in our training and test set, in which the percentages denote the CTR distribution.

Age Group	Gender	Cluster 1	Cluster 2	Cluster 3	Cluster 4
13-24	male	1.49%	0.48%	0.70%	1.64%
13-24	female	1.03%	0.00%	0.86%	1.02%
13-24	unknown	1.03%	0.00%	0.73%	0.86%
25-34	male	2.55%	0.78%	1.16%	1.41%
25-34	female	1.86%	1.03%	1.15%	1.51%
25-34	unknown	1.74%	0.72%	1.00%	1.13%
35-44	male	2.72%	0.96%	1.33%	1.61%
35-44	female	2.21%	1.04%	1.36%	1.94%
35-44	unknown	2.18%	0.90%	1.24%	1.42%
45-54	male	3.01%	1.06%	0.96%	1.31%
45-54	female	2.89%	1.12%	1.64%	1.94%
45-54	unknown	2.66%	0.86%	1.40%	1.36%
55+	male	3.01%	0.00%	1.09%	1.00%
55+	female	3.20%	0.00%	1.58%	1.36%
55+	unknown	3.36%	0.00%	1.38%	1.00%

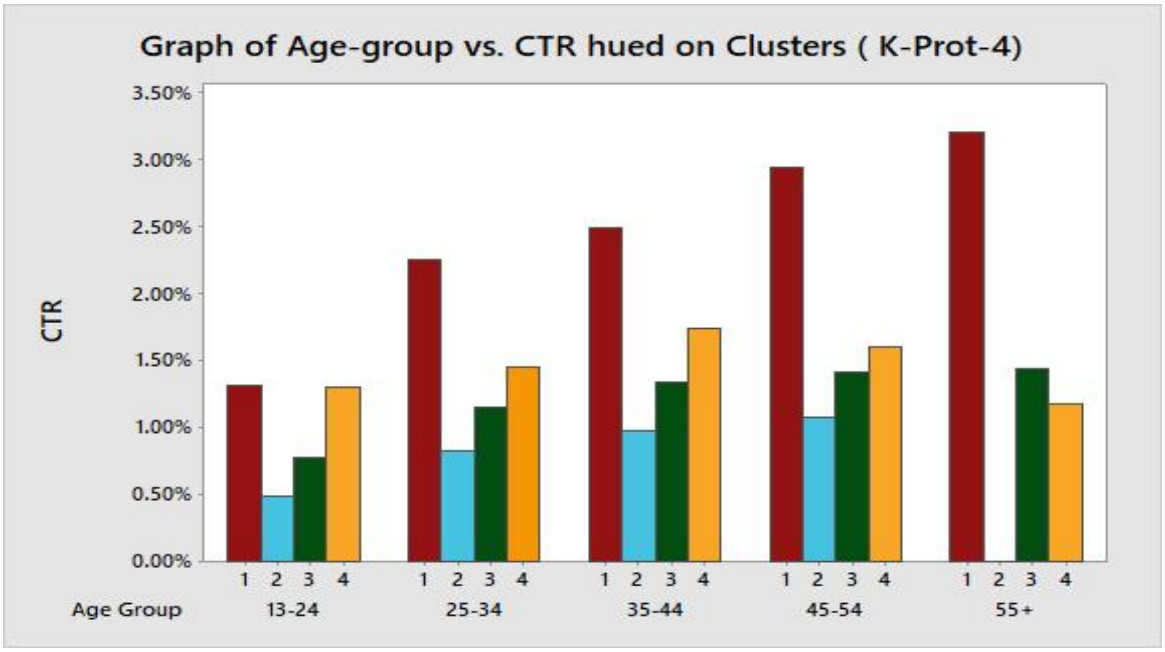
Train Set

Age Group	Gender	Cluster 1	Cluster 2	Cluster 3	Cluster 4
13-24	male	1.49%	0.48%	0.54%	1.64%
13-24	female	1.03%	0.00%	0.61%	1.02%
13-24	unknown	1.03%	0.00%	0.58%	0.86%
25-34	male	2.59%	0.78%	1.17%	1.41%
25-34	female	1.88%	1.03%	1.24%	1.51%
25-34	unknown	1.80%	0.72%	1.03%	1.13%
35-44	male	2.76%	0.96%	1.48%	1.61%
35-44	female	2.21%	1.04%	1.47%	1.94%
35-44	unknown	2.20%	0.90%	1.30%	1.42%
45-54	male	3.04%	1.06%	0.99%	1.31%
45-54	female	2.88%	1.12%	1.53%	1.94%
45-54	unknown	2.69%	0.86%	1.24%	1.36%
55+	male	3.03%	0.00%	1.24%	1.00%
55+	female	3.23%	0.00%	1.79%	1.36%
55+	unknown	3.60%	0.00%	1.24%	1.00%

Test Set

- For Cluster 1 the most responsive age groups were in the higher ranges of 45 and above.
- The middle age groups were dominant in Cluster 2.
- Cluster 3 had a mix of all age groups above 25.
- While Cluster 4 showed high response in the middle groups with females giving better response to campaigns than males.

The above matrix is depicted in a bar graph as shown below:



- **DATA VALIDATION**

1. 3 movies from the test set were removed and the clustering algorithm was run on the remaining 5000 + 74.
2. Using a predictive algorithm based on k-proto, clusters were assigned to these 4 movies.
3. Comparison between the actual age-gender target group and the ones obtained through clustering.

The 3 movies and their cluster numbers obtained:

Movie	**Cluster**
Revenge	1
All I Wish	3
The Captain	3

The validation set was found to be consistent with the actual campaign data and the result obtained from Clustering, it approximately defined the age gender cohort to be targeted for best value of CTR and when compared with the actual CTR values, the result was the same.

The CTR distribution of these movies from campaign data and the distribution of CTR among the Cluster 1 and 3 are shown as below:

- For Cluster 1 & Revenge

Age Group	Gender	Cluster 1
13-24	male	1.49%
13-24	female	1.03%
13-24	unknown	1.03%
25-34	male	2.55%
25-34	female	1.86%
25-34	unknown	1.74%
35-44	male	2.72%
35-44	female	2.21%
35-44	unknown	2.18%
45-54	male	3.01%
45-54	female	2.89%
45-54	unknown	2.66%
55+	male	3.01%
55+	female	3.20%
55+	unknown	3.36%

CTR Distribution Cluster 1

Age Group	Gender	CTR
13-17	male	1.45%
18-24	male	2.37%
18-24	female	0.66%
18-24	unknown	0.54%
25-34	male	2.25%
25-34	female	2.06%
25-34	unknown	0.93%
35-44	male	2.63%
35-44	female	1.39%
35-44	unknown	1.06%
45-54	unknown	6.73%
45-54	male	4.24%
45-54	female	4.16%
55-64	male	7.00%
55-64	female	5.33%
55-64	unknown	0.00%

CTR Distribution Revenge

- For Cluster 3, All I Wish & The Captain

Age Group	Gender	Cluster 3
13-24	male	0.70%
13-24	female	0.86%
13-24	unknown	0.73%
25-34	male	1.16%
25-34	female	1.15%
25-34	unknown	1.00%
35-44	male	1.33%
35-44	female	1.36%
35-44	unknown	1.24%
45-54	male	0.96%
45-54	female	1.64%
45-54	unknown	1.40%
55+	male	1.09%
55+	female	1.58%
55+	unknown	1.38%

CTR Distribution Cluster 3

Age Group	Gender	CTR
18-24	female	4.49%
18-24	unknown	2.33%
18-24	male	2.05%
25-34	female	4.21%
25-34	unknown	3.80%
25-34	male	2.42%
35-44	unknown	4.58%
35-44	female	4.50%
35-44	male	2.72%
45-54	female	4.95%
45-54	unknown	4.56%
45-54	male	3.04%
55-64	female	5.11%
55-64	unknown	4.31%
55-64	male	3.78%
65+	female	4.62%
65+	unknown	4.32%
65+	male	3.83%

CTR Distribution All I Wish

Age Group	Gender	CTR
25-34	unknown	3.50%
25-34	male	3.11%
25-34	female	2.79%
35-44	male	3.29%
35-44	unknown	3.25%
35-44	female	2.84%
45-54	male	4.10%
45-54	unknown	3.92%
45-54	female	3.34%
55-64	male	5.04%
55-64	unknown	4.33%
55-64	female	3.66%
65+	male	4.68%
65+	unknown	4.40%
65+	female	3.82%

CTR Distribution The Captain

BUSINESS ASPECT

Considering varied audience behaviours, reaching viewers who are most interested in a specific kind of film and most likely to see it in theatres is increasingly valuable. The aim of the study was to come up with the best demographics for a new campaign by analysing historical data of all previously run campaigns. It helps to uncover the proper demographics for movie campaigns. After following up multiple techniques to tackle this problem, the solution was ultimately obtained through Cluster Analysis.

Cluster analysis has a wide use of applications. Most importantly it is used to identify consumer segments, or competitive sets of products, for geo-demographic segmentation, etc. As observed frequently, it is rather necessary to split our data into multiple segments and perform any further analysis within each segment in order to come up with (potentially more granularized) segment-specific insights.

Different types of films appeal to different types of people, and we need to properly identify the target demographic segment for optimising marketing activities. This analysis groups similar kinds of movies based on the following properties:

- Genres
- Movie Rated
- Runtime
- Language
- Country
- Cast Scores
- Director Scores
- Instagram Followers

With this project the client shall be able to target the demographics more precisely for a new campaign. The new campaign(datapoint) can be run through the same Cluster Analysis and be assigned a Cluster number or be assigned a cluster number by a predictive function, the former being more effective. Following which the optimal Age Gender cohort for that campaign can be identified.

Enhanced efforts ensure the campaign reaches the potentially more responsive audience and funds can be optimised accordingly for maximum exposure.

APPENDIX

- OUTLIER TREATMENT

```
import os
import pandas as pd
import numpy

data = pd.read_csv("FinalCompiledData.csv")

perc = [] #percentiles in impressions
val = []
i = 10

while i < 101:
    x = numpy.percentile(data["Impression"],i)
    perc.append(i)
    val.append(x)
    if i < 81:
        i = i + 10
    else:
        i = i + 1

print(perc)
print(val)

import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize = (10,4))
sns.lineplot(x = perc, y = val)

#Delete those rows of the dataframe whose impressions are beyond the 96th percentile

crit = numpy.percentile(data["Impression"], 96) #the 96th percentile
ind = []

for i in range(0,data.shape[0]):
    if data.iloc[i,5] > crit:
        ind.append(i)

len(ind)

data = data.drop(ind, axis = 0) #removing the outlier points from the dataset.

perc1 = [] #outliers in followers
val1 = []
i = 10

while i < 101:
    x = numpy.percentile(data["Follower Count"],i)
    perc1.append(i)
    val1.append(x)
    if i < 81:
        i = i + 10
    else:
        i = i + 1

plt.figure(figsize = (10,4))
sns.lineplot(x = perc1, y = val1)

print(val1)
print(perc1)

#Replace those observations whose follower counts are beyond the 99th percentile
```

```
61.
62. crit1 = numpy.percentile(data["Follower Count"], 99) #the 99th percentile
63.
64. for i in range(0,data.shape[0]):
65.     if data.iloc[i,15] > crit1:
66.         data.iloc[i,15] = crit1
67.
68. perc1 = [] #percentile in Runtime
69. val1 = []
70. i = 10
71.
72. while i < 101:
73.     x = numpy.percentile(data["Follower Count"],i)
74.     perc1.append(i)
75.     val1.append(x)
76.     if i < 81:
77.         i = i + 10
78.     else:
79.         i = i + 1
80.
81. plt.figure(figsize = (10,4))
82. sns.lineplot(x = perc1, y = val1)
83.
84. print(val1)
85. print(perc1)
86.
87. perc1 = [] #outliers in Language Count
88. val1 = []
89. i = 10
90.
91. while i < 101:
92.     x = numpy.percentile(data["Lang Count"],i)
93.     perc1.append(i)
94.     val1.append(x)
95.     if i < 81:
96.         i = i + 10
97.     else:
98.         i = i + 1
99.
100.    plt.figure(figsize = (10,4))
101.    sns.lineplot(x = perc1, y = val1)
102.
103.    print(val1)
104.    print(perc1)
105.
106.    perc1 = [] #outliers in Country Count
107.    val1 = []
108.    i = 10
109.
110.    while i < 101:
111.        x = numpy.percentile(data["Cntry Count"],i)
112.        perc1.append(i)
113.        val1.append(x)
114.        if i < 81:
115.            i = i + 10
116.        else:
117.            i = i + 1
118.
119.    plt.figure(figsize = (10,4))
120.    sns.lineplot(x = perc1, y = val1)
121.
122.    print(val1)
123.    print(perc1)
124.
125.    perc1 = [] #outliers in Genre Count
126.    val1 = []
```

```
    i = 10

    while i < 101:
        x = numpy.percentile(data["Genre Count"],i)
        perc1.append(i)
        val1.append(x)
        if i < 81:
            i = i + 10
        else:
            i = i + 1

    plt.figure(figsize = (10,4))
    sns.lineplot(x = perc1, y = val1)

    print(val1)
    print(perc1)

    #Normalising the data

    import statistics#package to calculate meand and standard deviation

    for i in range(15,21):
        mean = statistics.mean(data.iloc[:,i])
        sd = statistics.stdev(data.iloc[:,i])
        for j in range(0, data.shape[0]):
            data.iloc[j,i] = (data.iloc[j,i] - mean)/sd
            data.iloc[j,i] = (data.iloc[j,i] +3)

    mean = statistics.mean(data.iloc[:,8])
    sd = statistics.stdev(data.iloc[:,8])
    for j in range(0, data.shape[0]):
            data.iloc[j,8] = (data.iloc[j,8] - mean)/sd
```

- QUERYING MOVIE ATTRIBUTES FOR CAMPAIGN DATA USING OMDB API

```
for t in lst:
    querystring = {"apikey":"PlzBanMe","t":"{}".format(t)}
    response = requests.request("GET", url, data=payload, headers=headers, params=querystring)

    if response.json()['Response']=='False':
        with open('omdb_not_found_response.json', 'a') as out_obj:
            out_obj.write(querystring['t'] + '\n')
    else:
        with open('omdb_response.json', 'a') as out_obj:
            out_obj.write(response.text)
```

- GETTING ACTORS UNIQUE IMDB ID

```
import os
import pandas as pd

#List of all actors in our dataset
actdf = pd.read_csv("actor.csv", encoding = 'utf_8')

#Exhaustive list of all actors
title = pd.read_csv("data.tsv", sep = '\t', encoding = 'utf_8')
```

```
#selecting only names and codes from master dataset
fetch = title.iloc[:,0:2]

result = fetch.merge(actdf,on='primaryName')

#Selecting the correct unique ID in case of multiple values
titles = []
nameid = []

data = pd.DataFrame()
for i in range (1, result.shape[0]):
    if result.iloc[i,1]!=result.iloc[i-1,1]:
        titles.append(result.iloc[i-1,1])
        nameid.append(result.iloc[i-1,0])

data["Names"] = titles
data["ID"] = nameid
```

• IMDB ACTOR & DIRECTOR PAGES SCRAPING

```
library(stringr)
library(XML)
library(rvest)
library(tidyverse)


actor_links <- read.csv("Actor_ID.csv", sep="|", encoding = "UTF-8")
html_resp <- c()
txt_ext <- c()

# IMDb Page HTML Scrape
for (i in 221:278){
  rawcode <-  read_html(as.character(actor_links[i,3]))
  html_resp <- rbind(html_resp,as.character(rawcode))
  j <- runif(1, min=4, max=12)
  msg <- paste0("Iteration: ",i+1, ", Lag: ",round(j,2)," sec")
  print(msg)
  Sys.sleep(j)
}

save(html_resp, file = 'scr_html_raw.Rdata')

# Text Extract out of HTML
for (i in 1:278){
  rawParse <- html_nodes(read_html(html_resp[i]),"div .awards-blurb")
  txtval1 <- html_text(rawParse[1])
  txtval2 <- try(html_text(rawParse[2]), silent = TRUE)
  txt_all <- paste0(txtval1,txtval2," ")
  txt_ext <-rbind(txt_ext, txt_all)
  txt_val1 <- c()
  txt_val2 <- c()
}

# Cleanup of text extracts
df <- as.data.frame(txt_ext)
df2 <- as.data.frame(txt_ext) %>%  mutate(txt_ext_cu = str_replace_all(V1, "\\n", "")) %>%
  mutate(txt_ext_cu = str_replace_all(txt_ext_cu, "Error in nodes_duplicated(nodes) :   Expect
ing an external pointer: [type=NULL].", ""))
```

```r


# Export of Data
actor_data <-  cbind(actor_links,df2$txt_ext_cu)
write.csv(actor_data,file  = 'actor_data.csv')

rm(list=ls())
######################################### Round 4 for Actor Credits  ########################
####################

#Steps: Load the files for wd: scr_html_raw, scr_html_raw_rd2_2 and scr_html_dd_2 one by one a
nd run the Text extract code below.
# export files for each of these loads

txt_ext <- c()
# Text Extract out of HTML
for (i in 1:length(html_resp)){
  a0 <- html_nodes(read_html(html_resp[i]),"div .head")
  a1 <- grepl("actress",as.character(a0)) | grepl("actor",as.character(a0))
  a2 <- paste0(html_text(a0[a1]),"")
  txt_ext <-rbind(txt_ext, a2)
  a0 <- c()
  a1 <- c()
  a2 <- c()
}

# Cleanup of text extracts

df2 <- as.data.frame(txt_ext) %>%  mutate(txt_ext_cu = str_replace_all(V1, "\\n", "")) %>%
  mutate(txt_ext_cu2 = str_replace_all(txt_ext_cu, "Hide|Show|Actress|Actor| ", ""))


# Export of Data
director_data <-  cbind(actor_links,df2$txt_ext_cu2)
write.csv(director_data,file  = 'director_data_rd1.csv')

######################################### Round 5 for Director Credits  ######################
#######################

#Steps: Load the files for wd: scr_html_raw, scr_html_raw_rd2_2 and scr_html_dd_2 one by one a
nd run the Text extract code below.
# export files for each of these loads
rm(actor_data, df2)
txt_ext <- c()
# Text Extract out of HTML
for (i in 1:length(html_resp)){
  a0 <- html_nodes(read_html(html_resp[i]),"div .head")
  a1 <- grepl("director",as.character(a0))
  a2 <- paste0(html_text(a0[a1]),"")
  txt_ext <-rbind(txt_ext, a2)
  a0 <- c()
  a1 <- c()
  a2 <- c()
}

# Cleanup of text extracts

df2 <- as.data.frame(txt_ext) %>%  mutate(txt_ext_cu = str_replace_all(V1, "\\n", "")) %>%
  mutate(txt_ext_cu2 = str_replace_all(txt_ext_cu, "Second|Unit|Director|or|Assistant|Hide|Sho
w| ", ""))


```

```
# Export of Data
actor_data <-  cbind(actor_links,df2$txt_ext_cu2)
write.csv(actor_data,file  = 'director_data_1_rd.csv')
```

- EXTRACTING AWARDS COUNT FROM SCRAPED INFORMATION

```
import os
import pandas as pd

data = pd.read_csv("Act_Award_Cred_final.csv", encoding = 'latin_1')
data = data.drop(columns="Links")
data.head()

mwins = []
mnoms = []
wins = []
noms = []
actcred = []
dircred = []

data1 = pd.DataFrame()

for i in range(0, data.shape[0]):
    mwins = data["Awards"].str.extract((r'(?:Won)(\d+)'))
    mnoms = data["Awards"].str.extract((r'(?:for)(\d+)'))
    wins = data["Awards"].str.extract((r'(\d+)(?:win)'))
    noms = data["Awards"].str.extract((r'(\d+)(?:nomination)'))
    actcred = data["Actorcredit"].str.extract((r'(\d+)(?:credit)'))
    dircred = data["Director credit"].str.extract((r'(\d+)(?:credit)'))

data1["Name"] = data["Name"]
data1["Awardstsring"] = data["Awards"]
data1["MWINS"] = mwins
data1["MNOMS"] = mnoms
data1["WINS"] = wins
data1["NOMS"] = noms
data1["ActCreds"] = actcred
data1["DirCreds"] = dircred

data1 = data1.fillna(0)
```

- EXTRACTING INSTAGRAM FOLLOWERS COUNT

```
library(stringr)
library(XML)
library(rvest)
library(tidyverse)
library(jsonlite)

actor_names <- read.csv("Actor_ID.csv", sep="|", encoding = "UTF-8")
foll_count <-  c()


for (i in 20:26){
  gg <- read_html(URLencode(paste0("https://www.instagram.com/web/search/topsearch/?query=",ac
tor_names[i,1])))
```

```
    gg2 <- fromJSON(html_text(gg), flatten = TRUE, simplifyVector = TRUE)
    gg3 <- tryCatch({as.data.frame(gg2$users)[,"user.follower_count"]}, error=function(cond) {
        return(0) })
    foll_count <- rbind(foll_count,max(gg3))
    j <- runif(1, min=0, max=0.4)
    msg <- paste0("Iteration: ",i+1, ", Lag: ",round(j,2)," sec  Actor: ",actor_names[i,1],"   "
, max(gg3))
    print(msg)
    Sys.sleep(j)
}

save(foll_count, file = 'ig_foll_.Rdata')

ig_foll <-  cbind(actor_names,foll_count)
write.csv(ig_foll, file = 'ig_followers.csv')
```

- # EXPLORATORY DATA ANALYSIS

```
import os
import numpy
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

data = pd.read_csv("Data_Visualization.csv")
data.head()

sns.set() #setting standard colour, theme, scaling, settings, etc

data.dtypes

#Age vs. CTR
plt.figure(figsize = (10,5))
sns.boxplot( x = "Age Group", y = "CTR", data = data)
plt.savefig('Age_vs_CTR.png')

#Gender vs. CTR
sns.boxplot( x = data.iloc[:,2], y = data.iloc[:,1])
plt.savefig('Gender_vs_CTR.png')


#Rating vs. CTR
plt.figure(figsize=(10,5))
sns.swarmplot( x = "Rating", y = "CTR", data = data)
plt.savefig('Rating_vs_CTR.png')

#Spend vs. CTR
plt.figure(figsize=(5,5))
sns.scatterplot( x = "Spend", y = "CTR", data = data)
plt.savefig('Spend_vs_CTR.png')

#Standardized Spend vs. CTR
plt.figure(figsize=(5,5))
sns.scatterplot( x = "Standardized Spend", y = "CTR", data = data)
plt.savefig('Spend_vs_CTR.png')

#Year vs. CTR
plt.figure(figsize=(20,5))
sns.lineplot( x = "Year", y = "CTR", data = data)
```

```
plt.savefig('Year_vs_CTR.png')

#Age and Gender vs. CTR
plt.figure(figsize=(15,6)) # this creates a figure 8 inch wide, 4 inch high
sns.boxplot( x = "Age Group", y = "CTR", hue = "Gender", data = data )
plt.savefig("AgeGender_vs_CTR.png")

#Genre vs. CTR
plt.figure(figsize=(25,6)) # this creates a figure 8 inch wide, 4 inch high
sns.boxplot(x = "GENRE", y = "CTR", data = genre)
plt.show()

#Genre and Age vs. CTR
plt.figure(figsize=(20,4)) # this creates a figure 25 inch wide, 4 inch high
sns.boxplot(x = "GENRE", y = "CTR", hue = 'Age Group', data = genre)
plt.show()

#Genre and Gender vs. CTR
plt.figure(figsize=(25,4)) # this creates a figure 25 inch wide, 4 inch high
sns.boxplot(x = "GENRE", y = "CTR", hue = 'Gender', data = genre)
plt.show()

plt.figure(figsize=(25,4)) # this creates a figure 25 inch wide, 4 inch high
sns.boxplot(x = "Gender", y = "CTR", hue = "Age Group", data = genre)
plt.show()
```

- DECISION TREE REGRESSION

```
d <- read.csv("CleanData.csv")


str(d)

require(dplyr)

d <- d %>% select(-c(Title, Impression,Clicks, Country, Language, Director))
str(d)


d<- d[complete.cases(d),]


require(rpart)


id <- sample(1:nrow(d),nrow(d)*0.8)
train <- d[id,]
test <- d[-id,]

dt <- rpart(CTR ~ . , data =train, method = 'anova' ,control =  rpart.control(cp=0.01))
printcp(dt)

dt1 <- prune(dt,cp=0.007)


preds <- as.data.frame(cbind(test$ctr,predict(dt1,test))) %>% rename(act=V1, pred=V2)

MAE(preds$act,preds$pred)
RMSE(preds$act,preds$pred)
```

```
preds$err <- preds$act-preds$pred

dt1
```

- HIERARCHICHAL CLUSTERING: AGGLOMERATIVE AND DIVISIVE

```
library(dplyr)


set.seed(40)


#dataframe
movies <- read.csv("FinalCompiledCleanData.csv")


str(movies)


require(dplyr)



movies$Follower.Count <- movies$Follower.Count + 3
movies$Cast.Score <- movies$Cast.Score + 3
movies$Director.Score <- movies$Director.Score + 3

scaled_set <- scale(movies %>% select(Runtime,Lang.Count,Cntry.Count,Genre.Count),scale=TRUE,center = TRUE)+3

cl_data <- cbind(movies %>% select(-c(Runtime,Lang.Count,Cntry.Count,Genre.Count)),scaled_set)

dat1 <- cl_data %>% select(-c(Title,Age.Group,Gender,CTR,Impression,Clicks, Country, Language, Director))


dat <- unique(dat1)



dat <- dat %>% select(-c(Title,Age.Group,Gender,CTR,Impression,Clicks, Country, Language, Director))


str(movies)

library(cluster)

##gower
gower.dist <- daisy(dat[ ,], metric = c("gower"))

class(gower.dist)


#Divisive Clustering
divisive.clust <- diana(as.matrix(gower.dist),
                        diss = TRUE, keep.diss = TRUE)
```

```r
#Plotting Denedogram
plot(divisive.clust, main = "Divisive")


#Agglomerative Clustering
aggl.clust.c <- hclust(gower.dist, method = "complete")
#Plotting Dendogram
plot(aggl.clust.c,
     main = "Agglomerative, complete linkages")


library(fpc)


cstats.table <- function(dist, tree, k){


clust.assess <- c("cluster.number","n","within.cluster.ss","average.within","average.between",
"wb.ratio","dunn2","avg.silwidth")


clust.size <- c("cluster.size")

stats.names <- c()

row.clust <- c()

output.stats <- matrix(ncol = k, nrow = length(clust.assess))

cluster.sizes <- matrix(ncol = k, nrow = k)

for(i in c(1:k))
{
  row.clust[i] <- paste("Cluster-", i, " size")
}

for(i in c(2:k))
{
  stats.names[i] <- paste("Test", i-1)

  for(j in seq_along(clust.assess))
  {
    output.stats[j, i] <- unlist(cluster.stats(d = dist, clustering = cutree(tree, k = i))[clu
st.assess])[j]

  }

  for(d in 1:k)
  {
    cluster.sizes[d, i] <- unlist(cluster.stats(d = dist, clustering = cutree(tree, k = i))[cl
ust.size])[d]
    dim(cluster.sizes[d, i]) <- c(length(cluster.sizes[i]), 1)
    cluster.sizes[d, i]

  }
}

output.stats.df <- data.frame(output.stats)

cluster.sizes <- data.frame(cluster.sizes)

```

```r

  cluster.sizes[is.na(cluster.sizes)] <- 0

  rows.all <- c(clust.assess, row.clust)


  output <- rbind(output.stats.df, cluster.sizes)[ ,-1]

  colnames(output) <- stats.names[2:k]

  rownames(output) <- rows.all

  is.num <- sapply(output, is.numeric)

  output[is.num] <- lapply(output[is.num], round, 2)

  output
  }



  #capping the maximum amout of clusters by 10


  #Obtaining information about clusters obtained through Divisive Clustering
  stats.df.divisive <- cstats.table(gower.dist, divisive.clust, 10)

  stats.df.divisive


  #Obtaining information about clusters obtained through Agglomerative Clustering
  stats.df.aggl <-cstats.table(gower.dist, aggl.clust.c, 10)

  stats.df.aggl


  #radial plot

  library(ggplot2)

  # Agglomerative clustering
  ggplot(data = data.frame(t(cstats.table(gower.dist, aggl.clust.c, 15))),
    aes(x=cluster.number, y=within.cluster.ss)) +
    geom_point()+
    geom_line()+
    ggtitle("Agglomerative clustering") +
    labs(x = "Num.of clusters", y = "Within clusters sum of squares (SS)") +
    theme(plot.title = element_text(hjust = 0.5))


  ggplot(data = data.frame(t(cstats.table(gower.dist, aggl.clust.c, 15))),
    aes(x=cluster.number, y=avg.silwidth)) +
    geom_point()+
    geom_line()+
    ggtitle("Agglomerative clustering") +
    labs(x = "Num.of clusters", y = "Average silhouette width") +
    theme(plot.title = element_text(hjust = 0.5))

```

- K-MEANS CLUSTERING

```
library(dplyr)


set.seed(40)


#dataframe
movies <- read.csv("FinalCompiledCleanData.csv")


str(movies)


require(dplyr)



movies$Follower.Count <- movies$Follower.Count + 3
movies$Cast.Score <- movies$Cast.Score + 3
movies$Director.Score <- movies$Director.Score + 3

scaled_set <- scale(movies %>% select(Runtime,Lang.Count,Cntry.Count,Genre.Count),scale=TRUE,c
enter = TRUE)+3

cl_data <- cbind(movies %>% select(-
c(Runtime,Lang.Count,Cntry.Count,Genre.Count)),scaled_set)

dat1 <- cl_data %>% select(-
c(Title,Age.Group,Gender,CTR,Impression,Clicks, Country, Language, Director))


dat <- unique(dat1)



dat <- dat %>% select(-
c(Title,Age.Group,Gender,CTR,Impression,Clicks, Country, Language, Director))


str(movies)

library(cluster)

##gower
gower.dist <- daisy(dat[ ,], metric = c("gower"))

##kmeans
km <-kmeans(gower.dist,5)
table(km$cluster)
table(km$size)
moviescl <- cbind(dat,km$cluster)
write.csv(moviescl,"moviescl.csv")
##kmeans end
```

- K-PROTOTYPE CLUSTERING

```
require(clustMixType)
require(dplyr)

set.seed(555666)

# Read data
data <- read.csv("fort5000.csv")


data2 <- data %>% mutate_if(is.integer,as.factor) %>% mutate(Total.Fol=as.integer(Total.Fol),

                                                             Title=as.character(Title))
str(data2)


dim(data %>% filter(Model=="Test") %>% na.omit())


# Select num and cats
num_vars <- data2 %>% select_if(is.numeric)
cat_vars <- data2 %>% select_if(is.factor)
titles <- data2 %>% select(Title)

str(num_vars)
str(cat_vars)


# Scale num vars
rng01 <- function(x){(x-min(x))/(max(x)-min(x))}
num_vars_sc <- apply(num_vars, 2, rng01)
summary(num_vars_sc)


# Combine these
df_cluster <- as.data.frame(cbind(num_vars_sc,cat_vars,titles))
str(df_cluster)


# Split the data in train - test ( Experiment )
train <- df_cluster %>% filter(Title !="All I Wish" & Title!="Revenge" & Title!="The Captain")

test <- df_cluster %>% filter(Title=="All I Wish" |Title=="Revenge"|Title=="The Captain")


#K-ProtoType ####

kprot <- kproto(train %>% select(-
c(Title,Cast.Score,Dir.Score)),lambda = 0.75, k=4, na.rm = F)

train$cluster <- kprot$cluster

write.csv(train %>% filter(Model=="Test") %>% select(Title,cluster),"test.csv")
train$c
train$cluster <-kprot$cluster
test <- predict(kprot,test%>% select(-c(Title,Cast.Score,Dir.Score)))
```

```
write.csv(train,"trainclus.csv")
```