Market Basket Analysis: frequent itemsets for the IMDB dataset

Algorithms for Massive Datasets – MSc Data Science and Economics, University of Milan
Maria Pia Chiatante, 944032 – maria.chiatante1@studenti.unimi.it

Abstract

This report is going to present the implementation of a system able to find frequent itemsets, also known as *Market Basket Analysis*, for the IMDB dataset, where movies will be considered as baskets while actors as items. The practical implementation has been carried out using the Python language on a Google Colab Notebook, available here: https://github.com/mariapiachiatante/AMD-DSE-MBA-Project/blob/main/AMD_DSE_Project_MP_Chiatante_MBA.ipynb.

First of all, there will be a brief exploratory analysis about the dataset and the variables involved, trying to display some interesting insights.

Then we will deal with the application of the Market Basket Analysis, specifically using the Frequent Pattern – Growth algorithm and explaining why we have decided to choose this model rather than any other, such as the A Priori algorithm. Here we will also discuss about the results obtained with this approach.

Description of the dataset and exploratory analysis

Description of the dataset

The **IMDB dataset** is actually a sort of database containing five different tables, with information about several movies, their ratings, the actors and characters involved.

The *title.akas* table contains the title of each movie in different languages, according to the region where they are localized, with an attribute telling us whether each single title is the original one or not.
The value "\N" is used to denote that a particular field is missing or null for that title.
The table has the following structure:

titleId	ordering	title	region	language	types	attributes	isOriginalTitle
tt0000001	1	Carmencita - span...	HU	\N	imdbDisplay	\N	0
tt0000001	2	Καρμενσίτα	GR	\N	\N	\N	0
tt0000001	3	Карменсита	RU	\N	\N	\N	0
tt0000001	4	Carmencita	US	\N	\N	\N	0
tt0000001	5	Carmencita	\N	\N	original	\N	1
tt0000002	1	Le clown et ses c...	\N	\N	original	\N	1
tt0000002	2	A bohóc és kutyái	HU	\N	imdbDisplay	\N	0
tt0000002	3	Le clown et ses c...	FR	\N	\N	\N	0
tt0000002	4	Clovnul si cainii...	RO	\N	imdbDisplay	\N	0
tt0000002	5	Клоун и его собаки	RU	\N	\N	\N	0
tt0000002	6	The Clown and His...	US	\N	\N	literal English t...	0
tt0000003	1	Sarmanul Pierrot	RO	\N	imdbDisplay	\N	0
tt0000003	2	Szegény Pierrot	HU	\N	imdbDisplay	\N	0
tt0000003	3	Бедный Пьеро	RU	\N	\N	\N	0
tt0000003	4	Pauvre Pierrot	\N	\N	original	\N	1
tt0000003	5	Poor Pierrot	GB	\N	imdbDisplay	\N	0
tt0000003	6	Pauvre Pierrot	FR	\N	\N	\N	0
tt0000004	1	Un bon bock	\N	\N	original	\N	1
tt0000004	2	Un țap de bere	RO	\N	\N	\N	0
tt0000004	3	Un bon bock	FR	\N	\N	\N	0

only showing top 20 rows

The *title.basics* table contains different information about each title, such as its type (short, movie, TV series, etc.), its duration in minutes and the genres associated to it.

The “primaryTitle” field represents the more popular title or the title used by the filmmakers on promotional materials at the point of release, which might be equal to the original title as well.

Tv Series have both a start and an end year, while other kinds of movies only have a start year, which represents the year of release of that movie.

The table has the following structure:

tconst	titleType	primaryTitle	originalTitle	isAdult	startYear	endYear	runtimeMinutes	genres
tt0000001	short	Carmencita	Carmencita	0	1894	\N	1	Documentary,Short
tt0000002	short	Le clown et ses c...	Le clown et ses c...	0	1892	\N	5	Animation,Short
tt0000003	short	Pauvre Pierrot	Pauvre Pierrot	0	1892	\N	4	Animation,Comedy,...
tt0000004	short	Un bon bock	Un bon bock	0	1892	\N	\N	Animation,Short
tt0000005	short	Blacksmith Scene	Blacksmith Scene	0	1893	\N	1	Comedy,Short

only showing top 5 rows

The *title.principals* table relates each title to the people that had a part in it, and specifies the category and job each person was attributed. Moreover, the “characters” field represents the name of the character played by a person, if applicable.

The table has the following structure:

tconst	ordering	nconst	category	job	characters
tt0000001	1	nm1588970	self	\N	["Herself"]
tt0000001	2	nm0005690	director	\N	\N
tt0000001	3	nm0374658	cinematographer	director of photo...	\N
tt0000002	1	nm0721526	director	\N	\N
tt0000002	2	nm1335271	composer	\N	\N

only showing top 5 rows

The *title.ratings* table reports, for each title, the weighted average of all the individual user ratings and the number of votes received.

The table has the following structure:

tconst	averageRating	numVotes
tt0000001	5.6	1550
tt0000002	6.1	186
tt0000003	6.5	1207
tt0000004	6.2	113
tt0000005	6.1	1934

only showing top 5 rows

The *name.basics* table contains information about the people that took part in the movies, such as actors, directors, composers, etc.

For each person, there is the primary name, i.e. the name by which the person is most often credited, the birth and death year (if applicable), the top-3 professions and the titles for which that person is commonly known.

The table has the following structure:

nconst	primaryName	birthYear	deathYear	primaryProfession	knownForTitles
nm0000001	Fred Astaire	1899	1987	soundtrack,actor,...	tt0050419,tt00531...
nm0000002	Lauren Bacall	1924	2014	actress,soundtrack	tt0071877,tt01170...
nm0000003	Brigitte Bardot	1934	\N	actress,soundtrac...	tt0054452,tt00491...
nm0000004	John Belushi	1949	1982	actor,writer,soun...	tt0077975,tt00725...
nm0000005	Ingmar Bergman	1918	2007	writer,director,a...	tt0069467,tt00509...

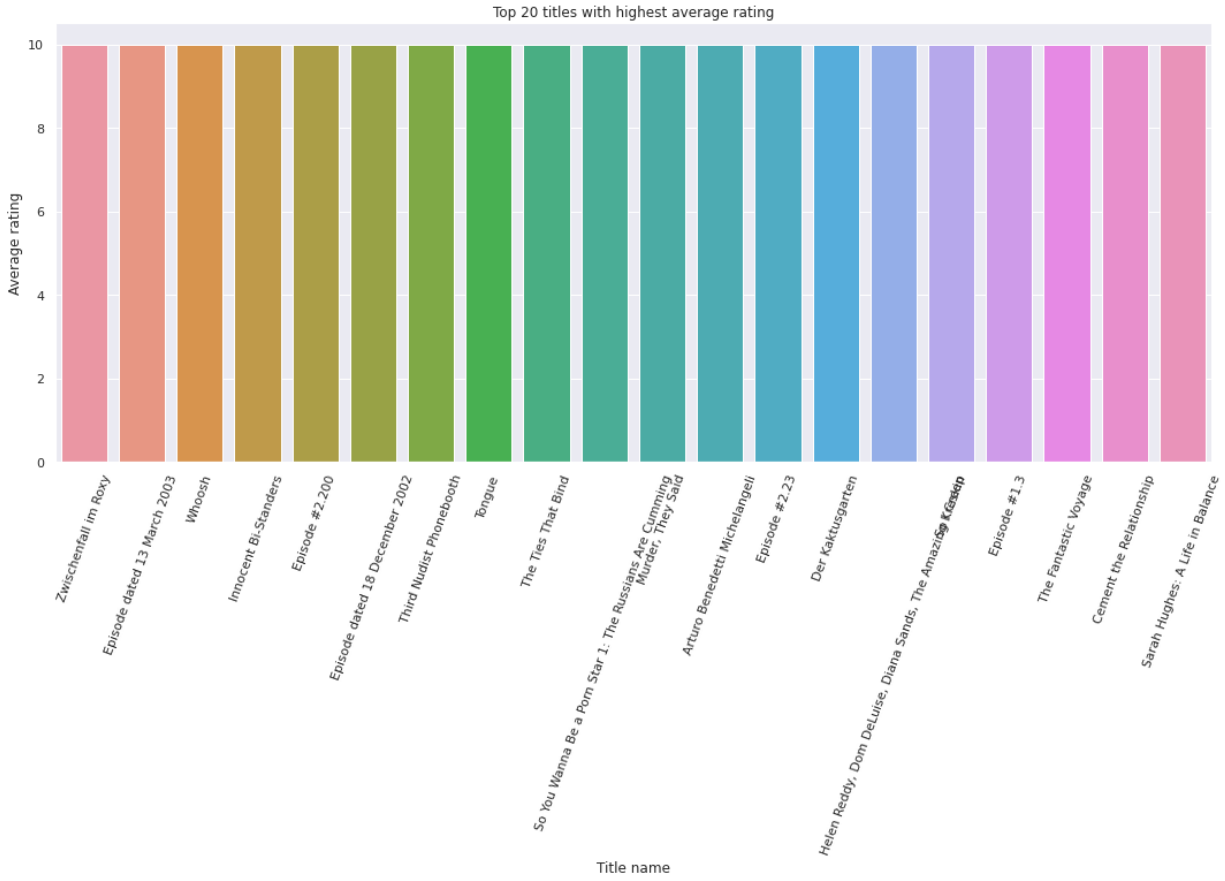
only showing top 5 rows

Before performing some exploratory analysis, we make sure that we do not consider **duplicates**, if they exist, using the Spark function “dropDuplicates()”.

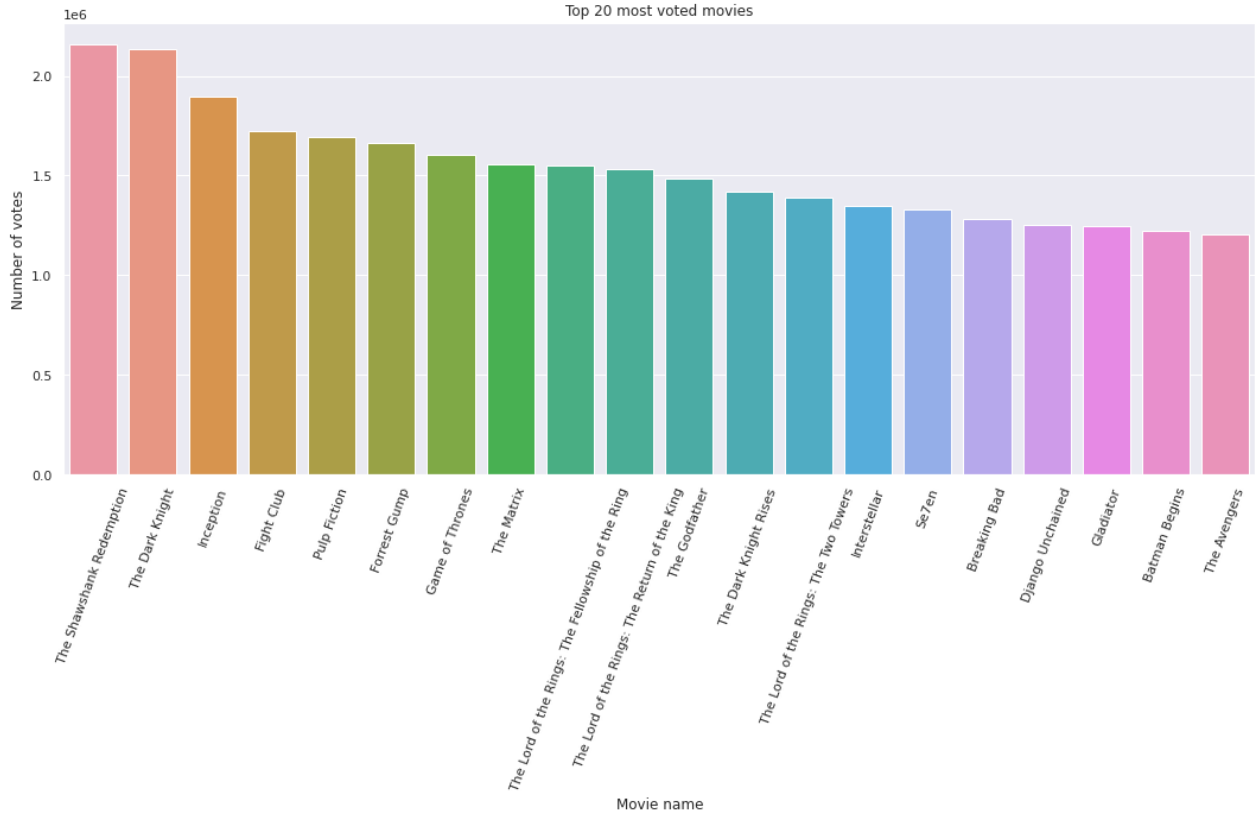
Exploratory analysis

We thought about some questions that one may be interested in answering when dealing with these data and we tried to answer these questions querying the dataset using the SQL language and providing some graphic visuals.

For instance, we might be interested in knowing which are the **titles** with the **highest average rating**. We selected only the top 20 titles just to limit the memory and runtime costs. However, this visual is not really informative because in the dataset there are lots of movies with the maximum average rating, which is equal to 10.

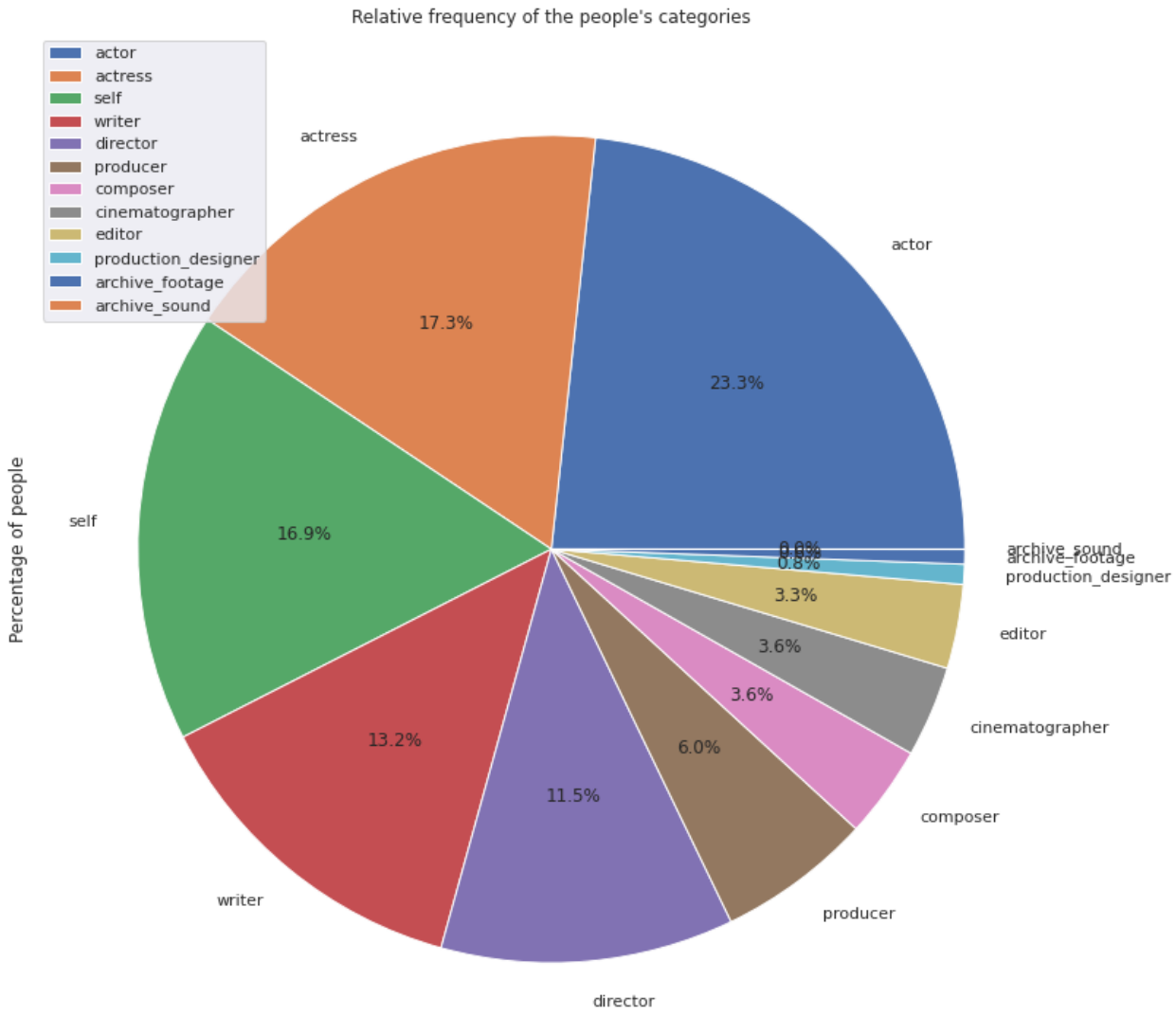


Therefore, it would be more insightful to display the titles that received the **highest number of votes** from users.



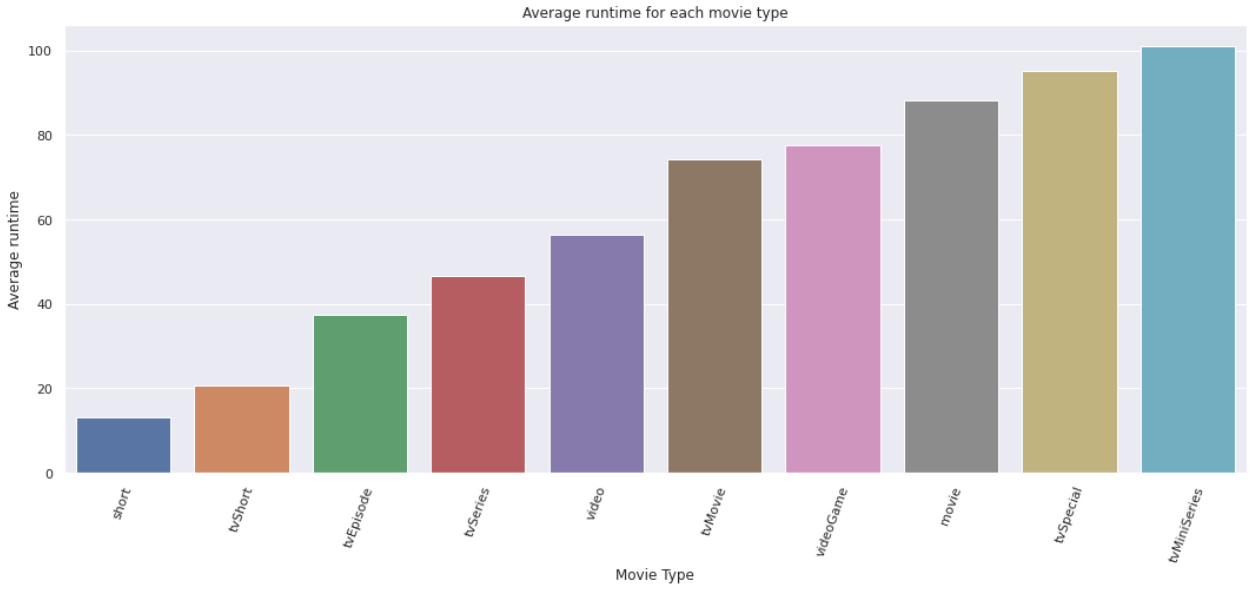
As we could expect, the titles with the highest number of votes are very well-known masterpieces, such as “Inception”, “Pulp Fiction”, “The Matrix” and so on, but the list is not limited to movies since some popular TV Series are also plotted in this graph, like “Game of Thrones” and “Breaking Bad”.

Another interesting question could be knowing how the **people** associated to the titles are distributed among the different **job categories**.

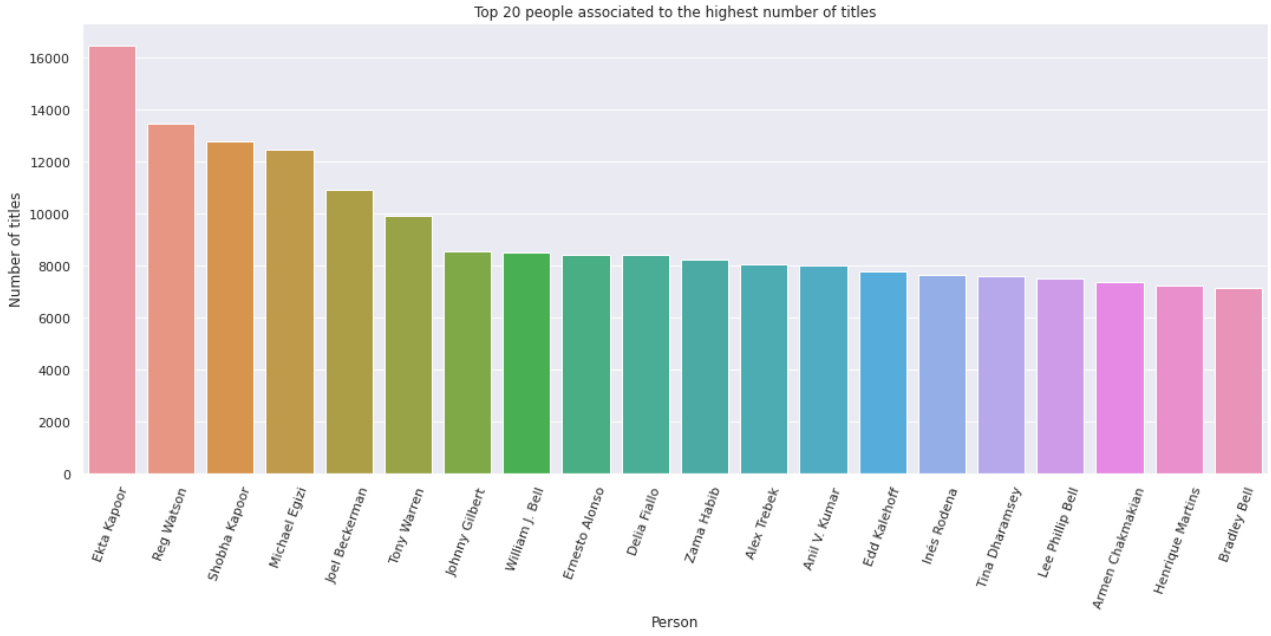


The actors’ category in general seems to be the more frequent, since actors and actresses together represent the 40.6% of people, followed by self, writers and directors. The other job categories are less frequent because they represent less than 10% of people.

Moreover, we can show the **average runtime** associated to each **movie type** and, as we would expect, short and TV short are the types with lowest duration. However, something that is not really trivial is the fact that TV Specials and TV Miniseries, on average, last more than normal or TV Movies.

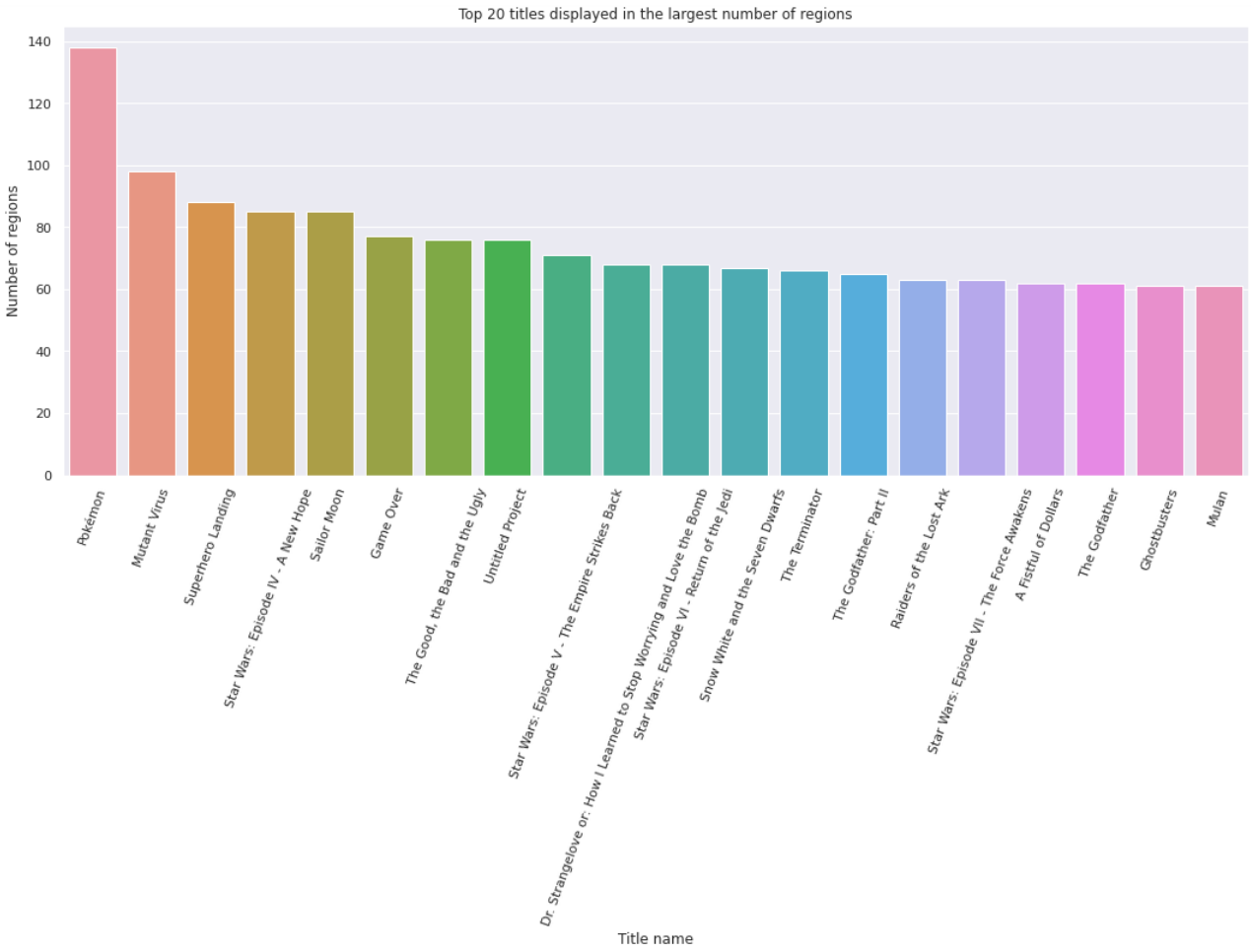


Next, we visualize the top 20 **people** associated to the **largest number of titles**.



It is interesting to note that the names of Ekta Kapoor and Shobha Kapoor appear in the first three places and, after a quick research, we found that they are daughter and father and that they run together a production house in Mumbai, so we guess they should be very popular in the Indian movies industry.

Finally, we plotted the top 20 **titles** displayed in the **largest number of regions** and, surprisingly, we found that “Pokémon” has been displayed in 138 different countries, which means that it has been distributed almost all over the world. Also “The GodFather” and many “Star Wars” episodes seem to be very international movies.



Market basket analysis

Now we are ready to perform the Market Basket Analysis (MBA) in order to find **frequent itemsets** for the IMDB dataset, that is the main goal of this report.

First of all, it is important to specify that we did not use the whole dataset for the purposes of the MBA, but just the portions of it that are necessary: the **titles**, that we considered as **baskets**, and the **actors**, that we considered as **items**. Since, as we saw during the exploratory analysis, the people associated to the movies belong to different job categories but we are only interested in actors, we filtered only for the categories “actress” and “actor”. This also allowed us to reduce the dimension of the dataset and speed up the process, improving the performance.

For the practical application of the MBA we choose the **Frequent Pattern – Growth algorithm**, provided by the Spark MLlib library, because it is generally known for being an improvement to the Apriori algorithm, mainly for two reasons:

- It is able to generate frequent patterns without the need to generate candidate itemsets, which can be large in number if the itemset in the dataset is huge;

- It reduces performance costs since there is no need to scan the dataset for each iteration in order to check the support of each generated itemset (it scans only twice).

We implemented the FP - Growth algorithm choosing values for the **hyperparameters** that we think are reasonable for this dataset, but there is not a correct configuration: it is all a matter of trial and error. We set the **minimum support** to 0.001 and the **minimum confidence** to 0. In particular, the minimum support creates a threshold in order to distinguish between frequent and infrequent occurrences of an itemset: the higher the threshold, the less will be the number of the found itemsets, which improves the performance and increases the statistical significance of the results. However, if the minimum support is too high, the algorithm may find zero itemsets, which would be meaningless and not useful. Therefore, the value of the minimum support is usually set low.

An **iteration** of the algorithm on the filtered dataset, with the parameters configured as stated above, takes around 7 minutes, which seems to be reasonable for a task like this.

The output of the algorithm provides us the frequent itemsets, which was our main goal, but also the generated **associations rules**.

antecedent	consequent	confidence	lift	support
[Vandana Vithlani...]	[Mat Solar]	1.0	882.2312614259597	0.001000351236736...
[Vandana Vithlani...]	[Nani Widjaja]	1.0	878.6172052799272	0.001000351236736...
[Vandana Vithlani...]	[Hari Teja]	1.0	777.7284448025786	0.001000351236736...
[Vandana Vithlani...]	[Sameera Sherief]	1.0	590.6738066095471	0.001000351236736...
[Vandana Vithlani...]	[Dilip Joshi]	1.0	699.6455237404857	0.001000351236736...
[Mat Solar, Andi ...]	[Hari Teja]	1.0	777.7284448025786	0.001028843892366144
[Mat Solar, Andi ...]	[Rupal Patel]	0.986404833836858	845.8813645764689	0.001014856588693...
[Mat Solar, Andi ...]	[Kavita Lad]	1.0	602.0966936993138	0.001028843892366144
[Mat Solar, Andi ...]	[Pallavi Ramisetty]	0.9843907351460222	622.6052072898229	0.001012784395556...
[Mat Solar, Andi ...]	[Ravi Kiran]	1.0	593.2151198524892	0.001028843892366144
[Mat Solar, Andi ...]	[Vandana Vithlani]	0.986404833836858	881.1101118286123	0.001014856588693...
[Mat Solar, Andi ...]	[Devoleena Bhatta...]	0.9869083585095669	883.1946761311565	0.001015374636977665
[María Bouzas, Ma...]	[Subhalekha Sudha...]	1.0	772.1288	0.001030398037218661
[María Bouzas, Ma...]	[Ravi Kiran]	1.0	593.2151198524892	0.001030398037218661
[María Bouzas, Ma...]	[Rohini Hattangadi]	1.0	589.4112977099237	0.001030398037218661
[María Bouzas, Ma...]	[Pallavi Ramisetty]	0.9854198089492208	623.2560735420963	0.001015374636977665
[María Bouzas, Ma...]	[Hari Teja]	1.0	777.7284448025786	0.001030398037218661
[María Bouzas, Ma...]	[Vandana Vithlani]	1.0	893.2540490513651	0.001030398037218661
[María Bouzas, Ma...]	[Devoleena Bhatta...]	0.9984917043740573	893.5607342469816	0.001028843892366144
[Jui Gadkari, And...]	[Harshada Khanvil...]	1.0	992.9639917695474	0.001003977574725...

only showing top 20 rows

The first association rule might be interpreted as follows: if Vandana Vithlani (and other actors) is in a movie then, with confidence 1, we can say that Mat Solar will also be in that movie.

It could be interesting to display the most frequent association rules, ordering them by the confidence (Figure 1).

antecedent (if)	consequent (then)	confidence
[Jui Gadkari, And...	[Astad Kale]	1.0
[Jui Gadkari, Har...	[Sameera Sherief]	1.0
[Jui Gadkari, And...	[Mat Solar]	1.0
[Vandana Vithlani...	[Nani Widjaja]	1.0
[Jui Gadkari, And...	[Nani Widjaja]	1.0
[Vandana Vithlani...	[Sameera Sherief]	1.0
[Jui Gadkari, And...	[Subhalekha Sudha...	1.0
[Mat Solar, Andi ...	[Hari Teja]	1.0
[Jui Gadkari, And...	[Sameera Sherief]	1.0
[Jui Gadkari, Har...	[Ravi Kiran]	1.0
[Jui Gadkari, And...	[Mounica]	1.0
[María Bouzas, Ma...	[Subhalekha Sudha...	1.0
[Devoleena Bhatta...	[Citra Kirana]	1.0
[María Bouzas, Ma...	[Rohini Hattangadi]	1.0
[Devoleena Bhatta...	[Subhalekha Sudha...	1.0
[María Bouzas, Ma...	[Vandana Vithlani]	1.0
[Devoleena Bhatta...	[Ravi Kiran]	1.0
[Vandana Vithlani...	[Mat Solar]	1.0
[Devoleena Bhatta...	[Sameera Sherief]	1.0
[Vandana Vithlani...	[Dilip Joshi]	1.0

Figure 1

items	freq
[Ravi Kiran, Moun...	3253
[Kavita Lad, Same...	3178
[Kavita Lad, Ravi...	3175
[Kavita Lad, Ravi...	3175
[Kavita Lad, Ravi...	3175
[Ravi Kiran, Moun...	3175
[Ravi Kiran, Moun...	3175
[Kavita Lad, Moun...	3175
[Kavita Lad, Moun...	3175
[Kavita Lad, Ravi...	3175
[Kavita Lad, Ravi...	3175
[Kavita Lad, Moun...	3175
[Kavita Lad, Ravi...	3175
[Ravi Kiran, Same...	3175
[Kavita Lad, Ravi...	3175
[Mounica, Sameera...	3175
[Pallavi Ramisett...	3046
[Pallavi Ramisett...	3046
[Pallavi Ramisett...	3046
[Pallavi Ramisett...	3046

Figure 2

Surprisingly (or not), we can see that the top 20 association rules mainly contain Indian actors.

We can also display the most frequent baskets of actors (Figure 2), containing at least two actors: we observe that Kavita Lad and Pallavi Ramisetty are almost always present in the top 20 baskets and they are both Indian actresses, again.

Finally, the transform method provides us a sort of **prediction**: given a title and the actors playing in it, it tells us the actors that would have been likely to find in that title. However, the result we got is not really interpretable, since we obtained empty predictions.

primaryTitle	actors	prediction
"Akatuski" shinkô	[Chie Nakamura, K...	[]
"Aki no Koraku! R...	[Kôji Yusa, Jûrôt...	[]
"Akubokutosen! Es...	[Kôji Yusa, Aya H...	[]
"Alte Ratsklause"...	[Christian Eckert...	[]
"American Idol" J...	[Adam Meir]	[]
"Antidote" by Tra...	[Daren Jackson]	[]
"Ask A Black Dude...	[Victor Dean, Ton...	[]
"Bad Things" - MG...	[Daren Jackson]	[]
"Bad and Boujee" ...	[Todd Nathanson]	[]
"Birthday Song" -...	[Daren Jackson]	[]
"Blank Space" by ...	[Malinda Kathleen...	[]
"Bohemian Rhapsod...	[Malinda Kathleen...	[]
"Bottled Up" Revi...	[Alana Jordan]	[]
"Breaking Bad" Blues	[Joe Gannon]	[]
"Captain America:...	[Joe Gannon]	[]
"Captain Philips"...	[Joe Gannon]	[]
"Catfish: Untold ...	[Miranda De Meo, ...	[]
"Chojugiga! Toki ...	[Kôji Yusa, Jûrôt...	[]
"Closer" by the C...	[Todd Nathanson]	[]
"Cool Kids' Table...	[Taylor Black]	[]

only showing top 20 rows