THE BUSINESS GAME

Lorenzo Baglini, Edoardo Borriello, Giacomo Flores

Overview

Our project aims to develop a marketing strategy based on a clustering process regarding a huge dataset containing customers and their detailed orders. To do so we took into consideration the outputs of the clusters and developed a different strategy for each of them, in order to optimize sales. Our second task concerned on a recommendation system in which we associated to each customer one or more items that they could like and buy. Lastly, we were also able to do the shipping time analysis, to try to predict the actual dispatch time of every order.

Clustering

The first task of our project was the clustering process, and we started our work knowing that the output should have been useful for the creation of a marketing strategy.

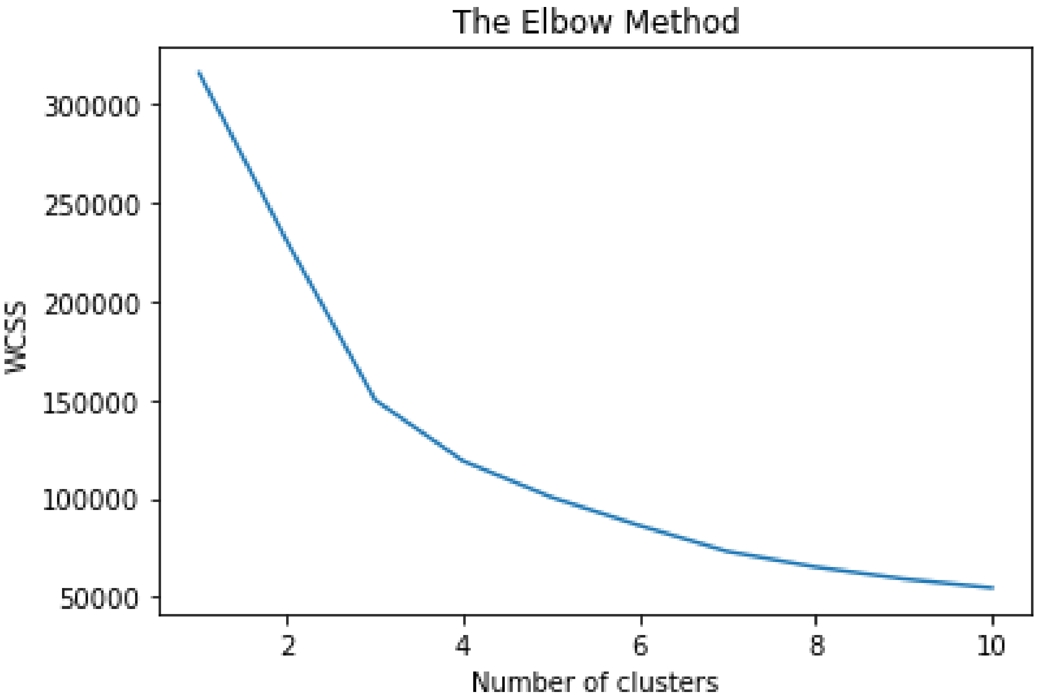
First, we imported the necessary libraries as Numpy, Matplotlib.pyplot and Pandas. Numpy is one of the most used packages for scientific computing in Python because offers a range of powerful Mathematical functions. Matplotlib.pyplot is a collection of functions that make matplotlib work like MATLAB. Each Pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. Last library, Pandas, is an important tool for Data Science and Machine learning and is used for data cleaning and analysis. Pandas is the best tool for handling this real-world messy data.

After that, we imported the dataset and replaced the few N/A values with "other" for the "product category name" variable and “0” for the "product photo quantity" one, in order to clean our data and for avoiding any possible bias.

Then we chose "price", "shipping cost" and "review score" as the variables for conducting our investigation, while, for the target variable, to which we applied a reshape to make it conform to X, we chose the "price". We selected these variables because we felt that they are the variables that most affect the actual purchasing conditions and it seemed clear to us to say that price and shipping conditions may influence a potential customer's purchasing decisions. On the other hand, reviews from customers who have already purchased products are based on experiences and thus are valuable indications for future sales.

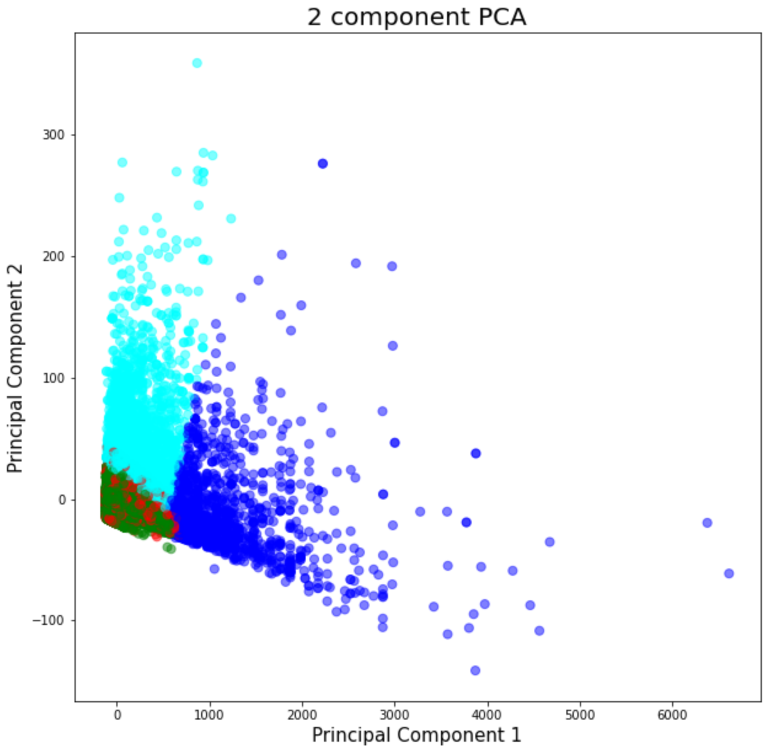
Then we applied the feature scaling to standardize the values between 0 and 1 in order to make a more accurate analysis.

At this step, we had to decide the best number of clusters for our analysis, so for this reason we set the K-Means model and we plotted the Elbow Method by which we understood that the best number of clusters was 4 and we also were able to compute the centroids. K-means clustering is one of the simplest and popular unsupervised machine learning algorithms, while the elbow method is a heuristic method used in determining the optimal number of clusters in a data set. This method consists of plotting the explained variation as a function of the number of clusters and picking the elbow of the curve as the number of clusters to use. We named our 4 clusters with numbers, from 0 to 3.



Moving on, we predicted the clusters, found their centroids and added a summary column with the clusters to our dataset.

Now comes the part of the PCA, as we needed to reduce the size of the variables to 2 in order to be able to plot, and we also choose the colors for each cluster. Subsequently, we set the code to show the graph, also setting the size of the figure and the names of the axes. Finally, we created a graph depicting only the centroids of each cluster.



Cluster 0 is the one with more observations inside and is characterized by a rating value of 4 and 5, price from 0.85 cents to 760 euros and a shipping cost from 0 to 44.88.

Cluster 1, on the other hand, differs for low rating values ranging from 1 to 3 while the price and shipping cost remain more or less unchanged, the first ranging from 0.85 to 829 and the second from 0 to 58.9.

Cluster 2 has all the rating values inside it, from 1 to 5, but it differs, in particular, for the value of the shipping cost which goes from 29.71 to 409.68, even the price is a little higher starting from 9.9 it goes up to 1149.

Last cluster, Cluster 3, has all the rating values from 1 to 5 but the difference is particularly in the price as it starts from 709.9 and goes up to 6735 while the shipping cost varies from 0.21 to 375.28.

For our market analysis we identified and classified our four clusters in:

Satisfied Customers: We think that we should continue to offer the same services, make some discount campaigns to avoid dropout, advertising and recommendation of products similar to the ones they already purchased and act promptly in case of negative reviews or experiences. In the end measure the customer satisfaction regularly could help improve the sales.

Unsatisfied Customers: It's useful to analyze negative comments and reviews to identify the problems. Make some discount campaigns, think about advertising and found recommendation of products that other costumers bought instead of the one they did not like in order to attract the costumer again on the website. It is also very important to ask for feedback on what went bad to show interest in their problems and provide instant response with live chat.

Average Customers: Continue with similar strategies for products and categories they already purchased and liked, but try to focus and analyze negative feedbacks and reviews to understand their preferences. Try to reinforce brand image to retain the costumer and provide instant response with live and bot chat to avoid website abandonment. Is also important to search and identify one-time customers to avoid overspending.

High spenders: This category is very similar to the average customers but with a focus on the quality of services offered. Most of these people, in fact, will be one-time customers, but offering a good service there will be a better chance of them coming back or of good word of mouth.

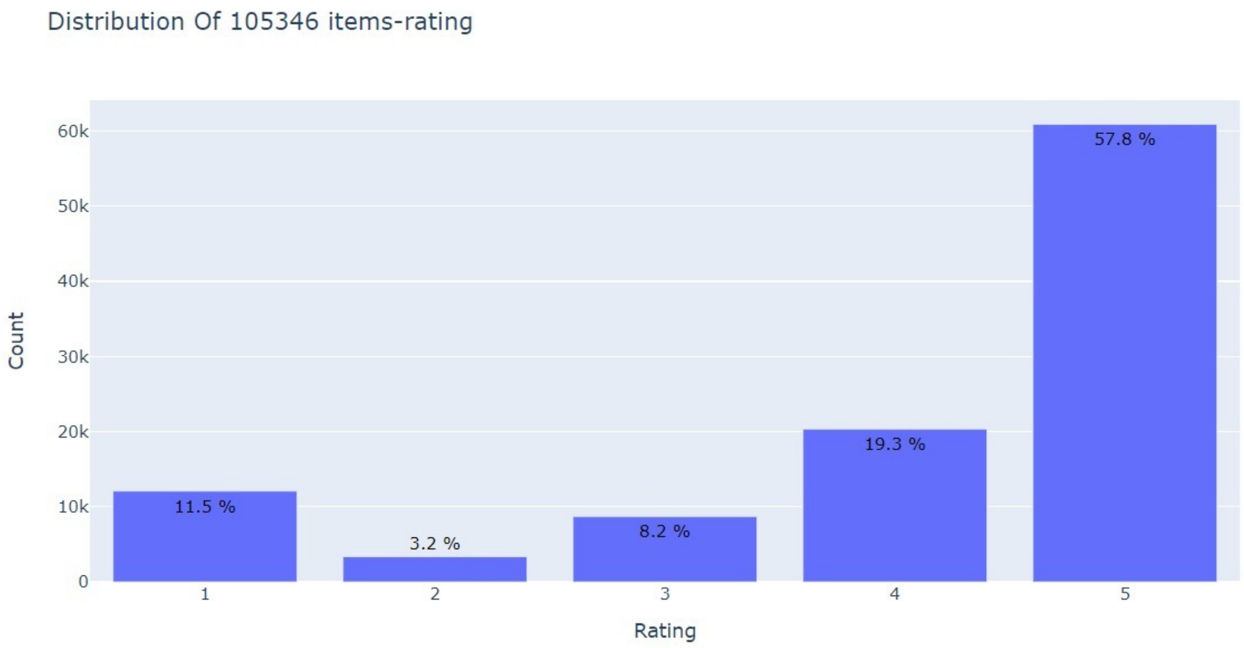
Recommendation System

The second task of the project was a recommendation system.

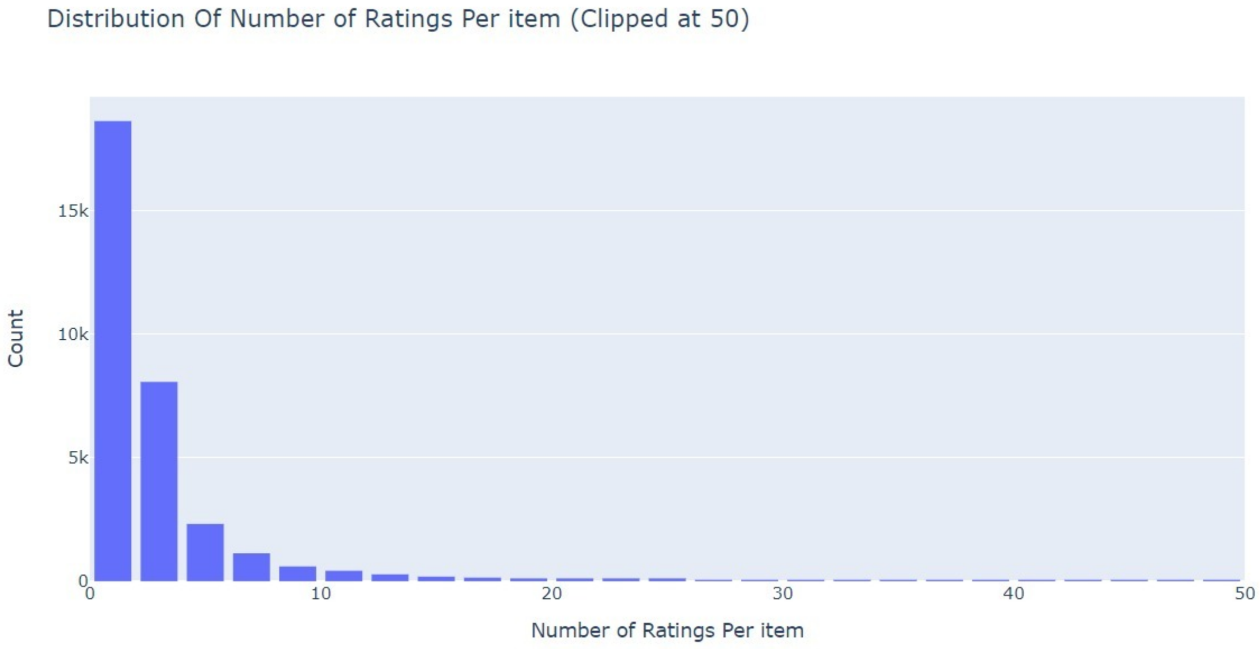
Obviously, also in this case, we imported the libraries and our dataset, and then we created a new one containing only the columns: "product id", "review score" and "customer unique id".

We then set up the first part with some plots to understand the situation well and help in subsequent analysis.

First, we found a graph depicting the distribution of ratings for the total of items which immediately makes us understand that, within the dataset, the predominant rating is 5, present in 57.8% of cases.

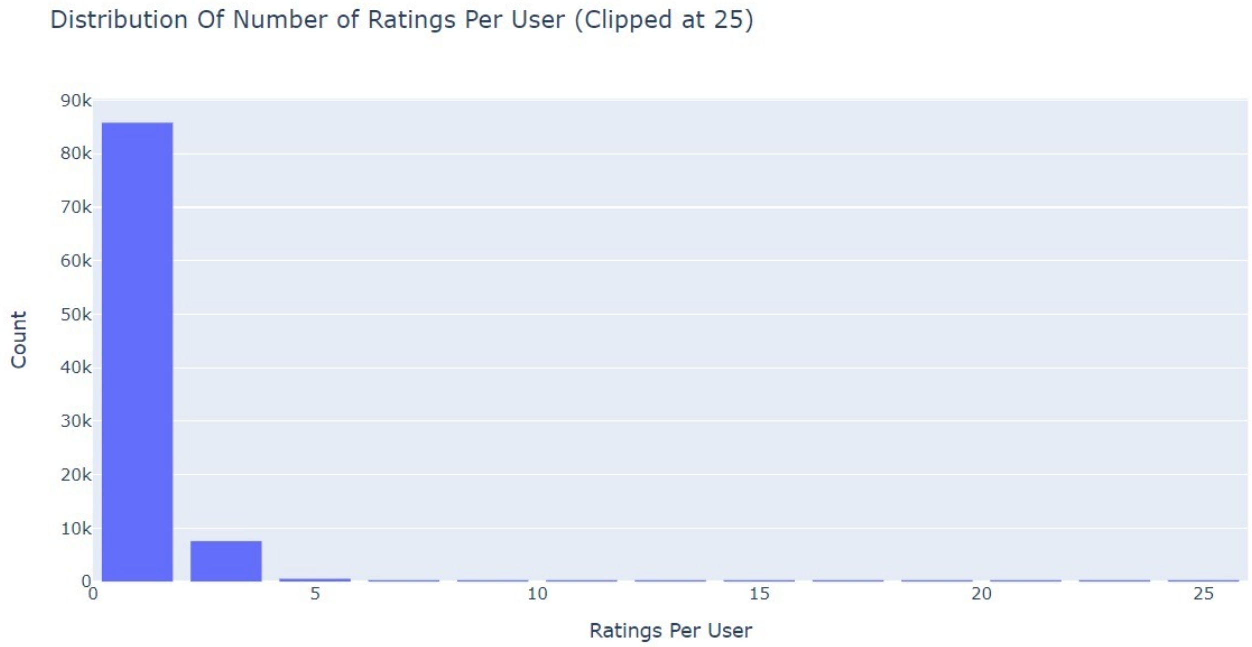


Using the second graph, instead, we have decided to show the distribution and the number of ratings for each object, noticing that, in the majority of cases, the items have only one vote (18,639), falling by more than half in the column relating to 2/3 votes (8073) going more and more tending to 0.



After that, we grouped the dataset to see which items had been voted on multiple times and we found the first one at 493 times.

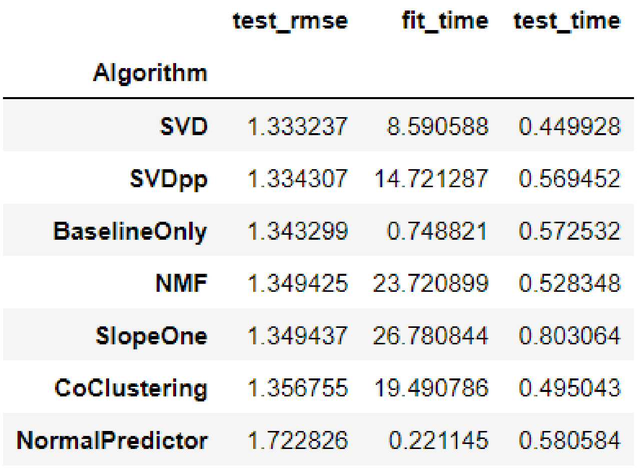
The third and last graph, represents the number of votes per user. In this case, we observe that the trend is only 1 vote per user (85,802) and drastically decreasing in the second data relative to 2/3 times (7,577). We have grouped the dataset to see which user had voted several times and found the first at 54.



But now the real process of getting a good recommendation system begins. To do this we have chosen to use the Surprise library, created specifically to solve analysis of this kind through which we have imported all types of analysis model useful to validate and process our work.

We set up the reader and imported the dataset again, always taking the same columns as before.

After that we computed 7 different algorithms (SVD, SVDpp, BaselineOnly, NMF, SlopeOne, CoClustering, NormalPredictor) and choose the most fitted with our data: Singular Value Decomposition (SVD). In fact, through it we calculated the cross validation using "RMSE" as a measure.

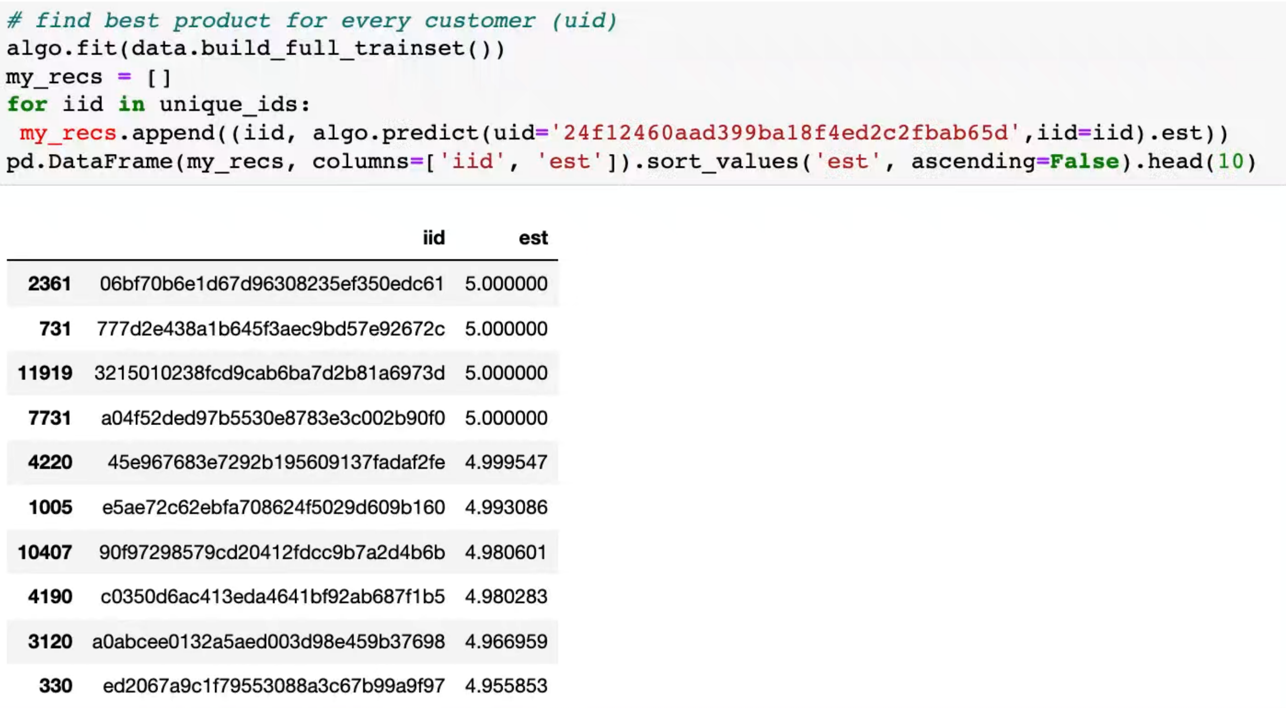


	test_rmse	fit_time	test_time
Algorithm			
SVD	1.333237	8.590588	0.449928
SVDpp	1.334307	14.721287	0.569452
BaselineOnly	1.343299	0.748821	0.572532
NMF	1.349425	23.720899	0.528348
SlopeOne	1.349437	26.780844	0.803064
CoClustering	1.356755	19.490786	0.495043
NormalPredictor	1.722826	0.221145	0.580584

After having imported, again from Surprise, "accuracy" and "train test split", we have found the accuracy of the prediction (RMSE) equivalent to 1.3371.

Then, we decided to define two new functions to create a table representing the ID of the customer (uid), associated to the number of rated items (lu), and the ID of the items associated to the number of users who rated that object (Ui) and, related to this, we printed the best and worst predictions.

Then we took the list of the IDs of the items, and we set up the script in order to receive the best product as output based on any customer ID entered. Finally, we set up a plot to understand how many and how a given item was rated.



```python
# find best product for every customer (uid)
algo.fit(data.build_full_trainset())
my_recs = []
for iid in unique_ids:
 my_recs.append((iid, algo.predict(uid='24f12460aad399ba18f4ed2c2fbab65d',iid=iid).est))
pd.DataFrame(my_recs, columns=['iid', 'est']).sort_values('est', ascending=False).head(10)
```

	iid	est
2361	06bf70b6e1d67d96308235ef350edc61	5.000000
731	777d2e438a1b645f3aec9bd57e92672c	5.000000
11919	3215010238fcd9cab6ba7d2b81a6973d	5.000000
7731	a04f52ded97b5530e8783e3c002b90f0	5.000000
4220	45e967683e7292b195609137fadaf2fe	4.999547
1005	e5ae72c62ebfa708624f5029d609b160	4.993086
10407	90f97298579cd20412fdcc9b7a2d4b6b	4.980601
4190	c0350d6ac413eda4641bf92ab687f1b5	4.980283
3120	a0abcee0132a5aed003d98e459b37698	4.966959
330	ed2067a9c1f79553088a3c67b99a9f97	4.955853

(it’s a video)

Shipping time analysis

Obviously also in this case we have, we have imported the libraries and the dataset, checked for null values and deleted them, if strictly necessary.

We have chosen the target variable and the variables to study on and split into training and test data. Through sklearn.linear_model, we imported and computed the linear regression, and, through sklearn.preprocessing, we did the same for the polynomial regression. The variables chosen to conduct the analysis are those concerning the size of the package (computed with the measurement we had on the dataset) to be shipped and the distance to be traveled (computed with the geographical coordinates), in order to be able to show correlations with actual times. Unfortunately, the predicted times and the actual times recorded in some of the cases do not match, so we can say that the variables examined do not fully account for the shipping times of the various items purchased online. We found confirmation that shipping times are not perfectly predicted and do not follow a precise algorithm since, in various cases, orders placed at the same time, concerning the same product, and from the same city, were experiencing variation in shipping times of even more than 10 days.

We have given more weight to the results acquired by the polynomial regression because they are clearly more precise, follows a more reliable trend and provides more information than the results obtained by linear regression.

After that we have printed the predictions of both the regressions, and we have filled the values of the polynomial in the dataset.