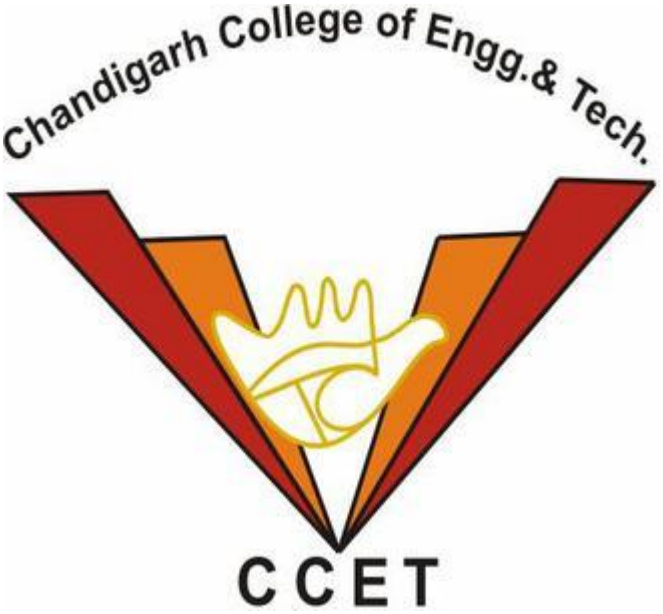
CHANDIGARH COLLEGE

OF

ENGINEERING AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



DATA MINING AND ANALYSIS

ASSIGNMENT (DATA ANALYSIS OF GOOGLE APPS RATING)

SUBMITTED TO:

DR. ANKIT GUPTA

SUBMITTED BY:

ABOLI (CO17305)

SIDDHARTH (CO17358)

CSE 6TH SEM/3RD YEAR

LIST OF CONTENTS

ACKNOWLEDGEMENT

We would like to put forth my utmost thanks to all those who had helped us in guiding us to get the most out of this golden opportunity to make the data mining assignment on topic Data Analysis of Google Apps' Rating at Chandigarh College of Engineering & Technology Sec-26, Chandigarh.

We pay our special thanks to Dr. Ankit Gupta (Assistant Professor, CSE Department), who gave us a chance to work with him and gave us excellent knowledge. Also, we put our special regards to the management, team members and friends who have always been so supporting and ready to help.

We also put forth our special thanks to all the concerned persons as well as Dr. Sunil K. Singh, (HOD, CSE), Dr. Ankit Gupta(Training In-charge, CSE) and Dr. Manpreet Singh, Principal C.C.E.T. (Degree Wing), Chandigarh, who have enabled us to have an opportunity to work at the prestigious organization.

ABSTRACT

In this report, we are performing data analysis on the dataset containing the details of Google Apps' rating.

We've performed data analysis on the said data and have given detailed description of the methods and techniques used along with the results and conclusions.

The tools used to build the models and representations have been mentioned.

Initially, basic analysis on the dataset has been performed and visualizations have been presented accordingly. Moving further, based on the patterns present Association Rule Mining, Naïve Bayes Classifier and One R Training have been performed.

Predictive Models have been built using the Regression, KNN and Tree Models. Later on, Correlation Analysis, Clustering and Discretization have been done to depict the relations and patterns present among the data.

TOOLS USED

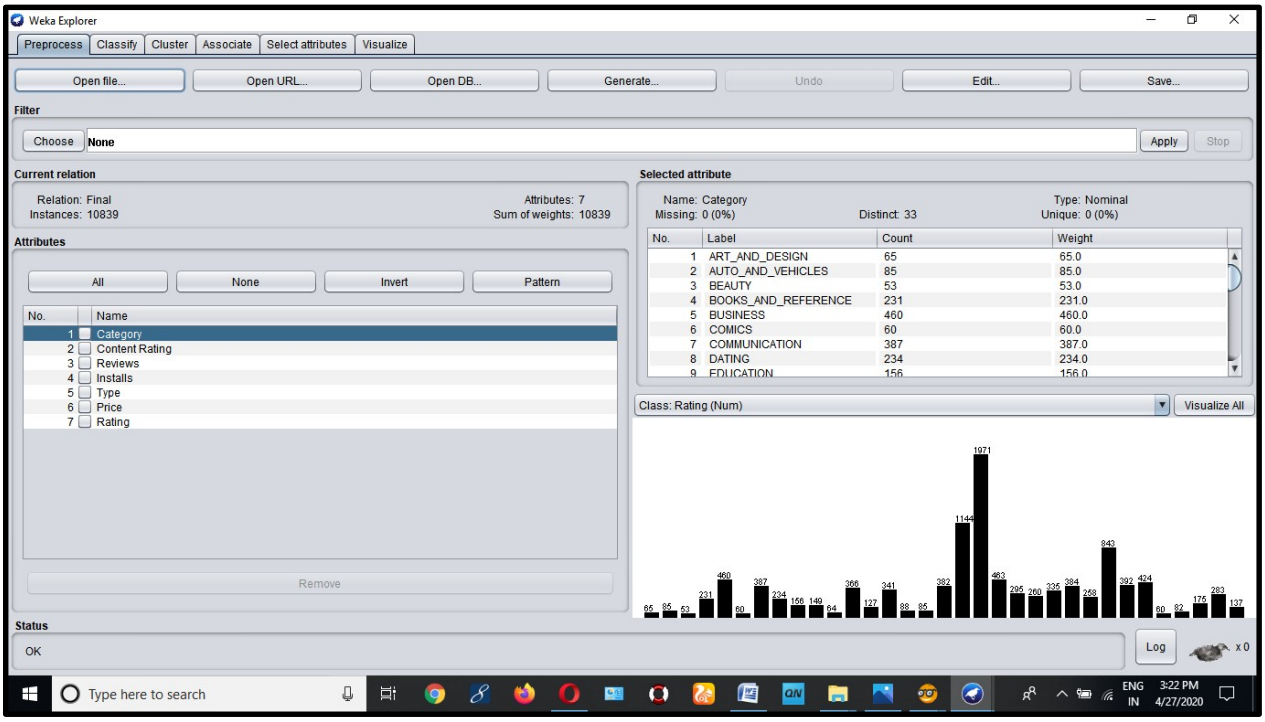
For performing the Data Analysis, building various Models and Visualization we have used the following:

- WEKA
- ORANGE
- PYTHON
- MICROSOFT EXCEL

WEKA

It is open source software provides tools for data preprocessing, implementation of several Machine Learning algorithms, and visualization tools so that you can develop machine learning techniques and apply them to real-world data mining problems.

IMPORTING THE DATASET IN WEKA



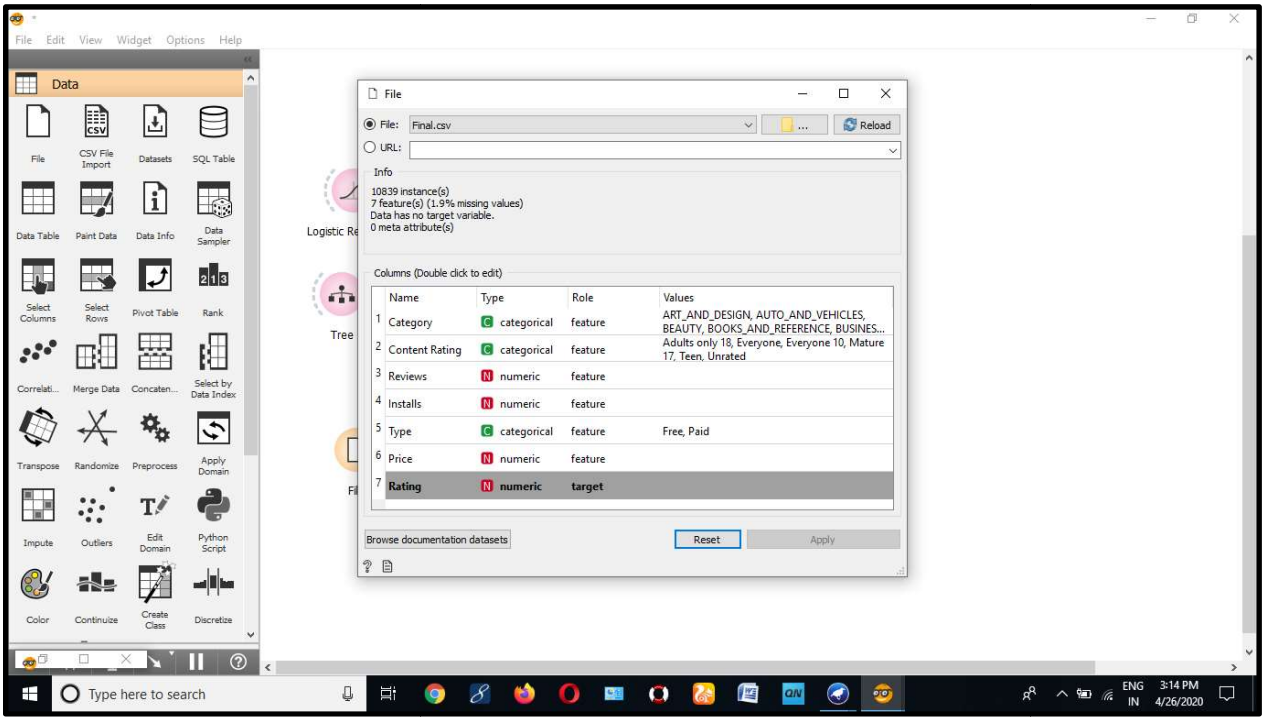
ORANGE

Orange is a C++ core object and routines library that incorporates a huge variety of standard and non-standard machine learning and data mining algorithms. It is an open-source data visualization, data mining, and machine learning tool. Orange is a scriptable environment for quick prototyping of the latest algorithms and testing patterns. It is a group of python-based modules that exist in the core library. It implements some functionalities for which execution time is not essential, and that is done in Python.

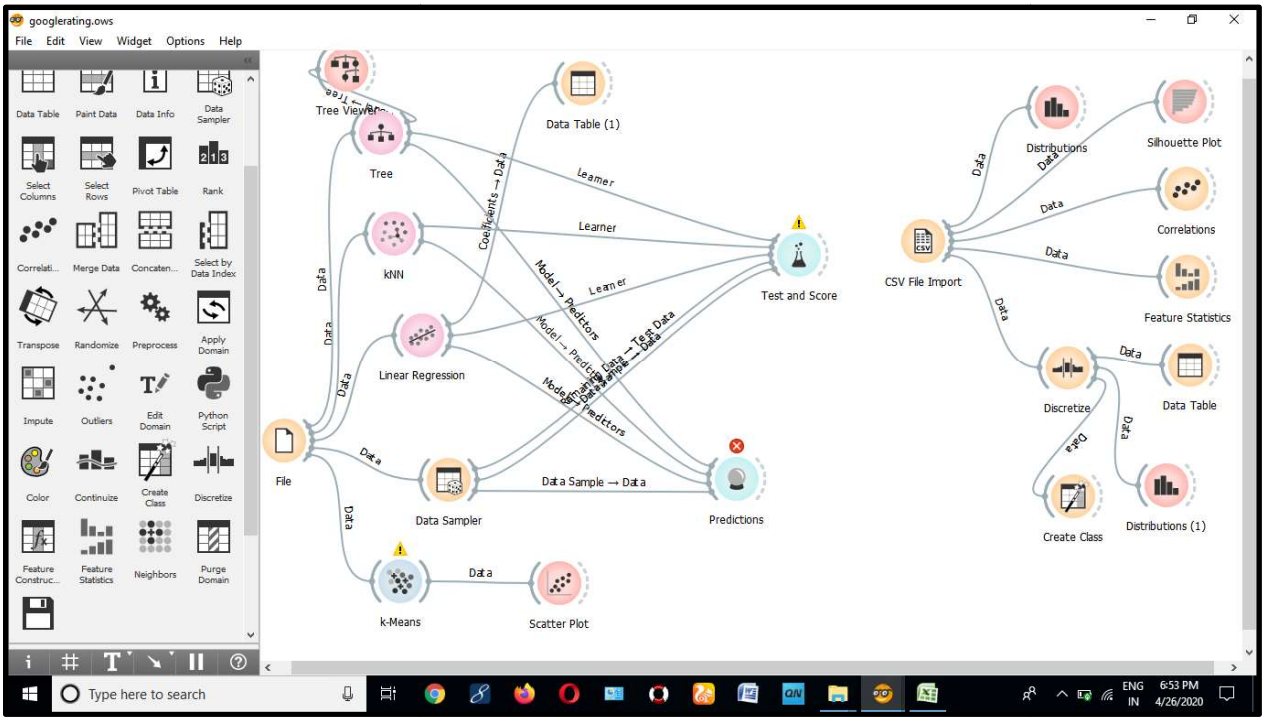
Orange is a set of graphical widgets that utilizes strategies from the core library and orange modules and gives a decent user interface. The widget supports digital-based communication and can be gathered together into an application by a visual programming tool called an orange canvas.

The objective of Orange is to provide a platform for experiment-based selection, predictive modeling, and recommendation systems. It is primarily used in bioinformatics, genomic research, biomedicine, and teaching. In education, it is used for providing better teaching methods for data mining and machine learning to students of biology, biomedicine, and informatics.

IMPORTING DATASET IN ORANGE



OUR WORKFLOW IN ORANGE



DATASETS

Attributes in the dataset:

- Category: It is a categorical feature and represents under which category the app falls. For example, Art and Design.
- Rating: It is the Google rating of the App.
- Reviews: It represents the number of reviews
- Installs: Number of installations for that particular app
- Type: It tells whether the app is freely available for use or paid
- Price: The price of App
- Content Rating: The audience for whom the app is meant, for example, teens, adults, etc.

Attributes and its type

	Name	Type	Role
1	Category	C categorical	feature
2	Rating	N numeric	feature
3	Reviews	N numeric	feature
4	Installs	N numeric	feature
5	Type	C categorical	feature
6	Price	N numeric	feature
7	Content Rating	C categorical	feature

CONTINUOUS

	Category	Rating	Reviews	Installs	Type	Price	Content Rating
1	ART_AND_DESIGN	4.1	159	10000	Free	0.00	Everyone
2	ART_AND_DESIGN	3.9	967	500000	Free	0.00	Everyone
3	ART_AND_DESIGN	4.7	87510	5000000	Free	0.00	Everyone
4	ART_AND_DESIGN	4.5	215644	50000000	Free	0.00	Teen
5	ART_AND_DESIGN	4.3	967	100000	Free	0.00	Everyone
6	ART_AND_DESIGN	4.4	167	50000	Free	0.00	Everyone

NOMINAL

No.	1: Category Nominal	2: Rating Numeric	3: Reviews Numeric	4: **Installs** Nominal	5: Type Nominal	6: Price Numeric	7: Content Rating Nominal
1	ART_AN...	4.1	159.0	LOW	Free	0.0	Everyone
2	ART_AN...	3.9	967.0	MEDIUM	Free	0.0	Everyone
3	ART_AN...	4.7	87510.0	HIGH	Free	0.0	Everyone
4	ART_AN...	4.5	215644.0	VERY ...	Free	0.0	Teen
5	ART_AN...	4.3	967.0	MEDIUM	Free	0.0	Everyone
6	ART_AN...	4.4	167.0	MEDIUM	Free	0.0	Everyone

INTERVALS	CATEGORY
<750	VERY LOW
750-30,000	LOW
30,000-750,000	MEDIUM
750,000-7,500,000	HIGH
>7,500,000	VERY HIGH

NOTE: We have used equal frequency discretization to discretize Installs Variable.

STATISTICS

PRICE

Statistic	Value
Minimum	0
Maximum	400
Mean	1.027
StdDev	15.95

RATING

Statistic	Value
Minimum	1
Maximum	5
Mean	4.192
StdDev	0.515

REVIEWS

Statistic	Value
Minimum	0
Maximum	78158306
Mean	444193.873
StdDev	2927892.561

TYPE

No.	Label	Count
1	Free	10039
2	Paid	800

CONTENT RATING

No.	Label	Count
1	Everyone	8715
2	Teen	1208
3	Everyone 10	412
4	Mature 17	499
5	Adults only 18	3
6	Unrated	2

INSTALLS NUMERIC

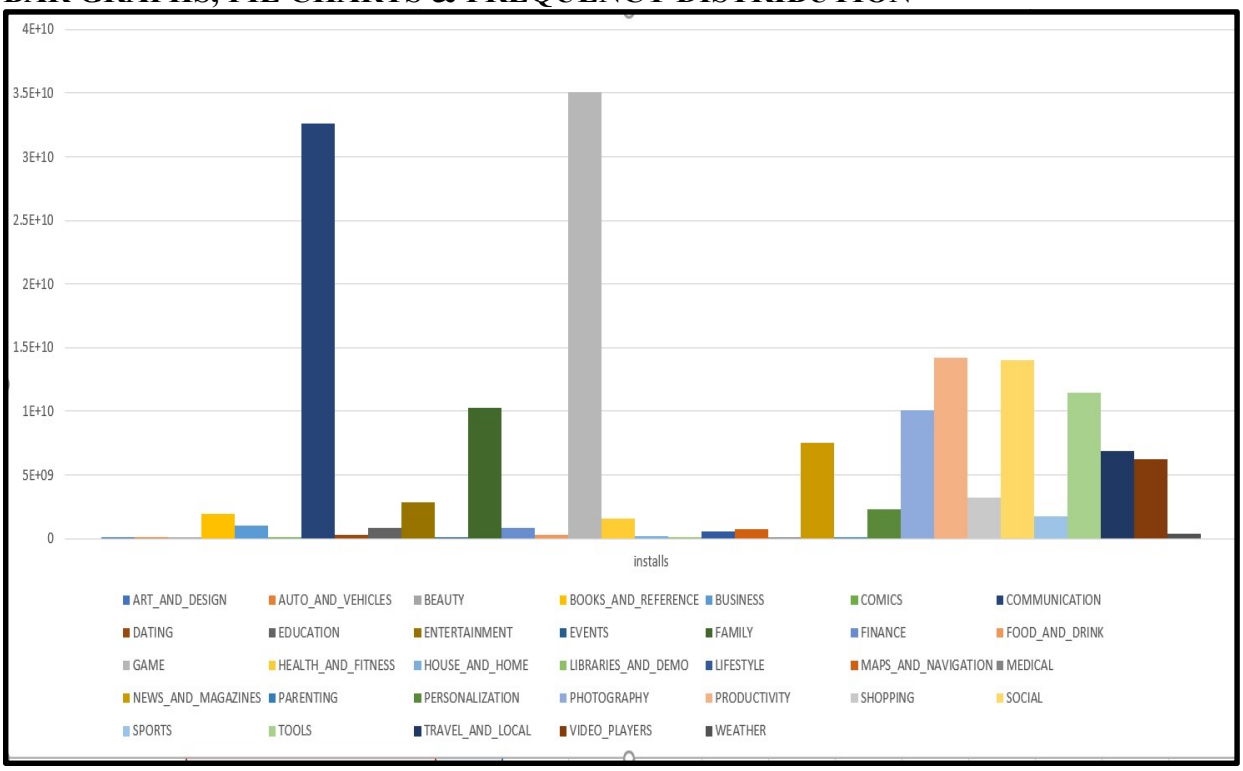
Statistic	Value
Minimum	0
Maximum	1000000000
Mean	15465765.614
StdDev	85033154.289

INSTALLS NOMINAL

No.	Label	Count
1	LOW	2438
2	MEDIUM	2187
3	HIGH	2331
4	VERY HIGH	2080
5	VERY LOW	1803

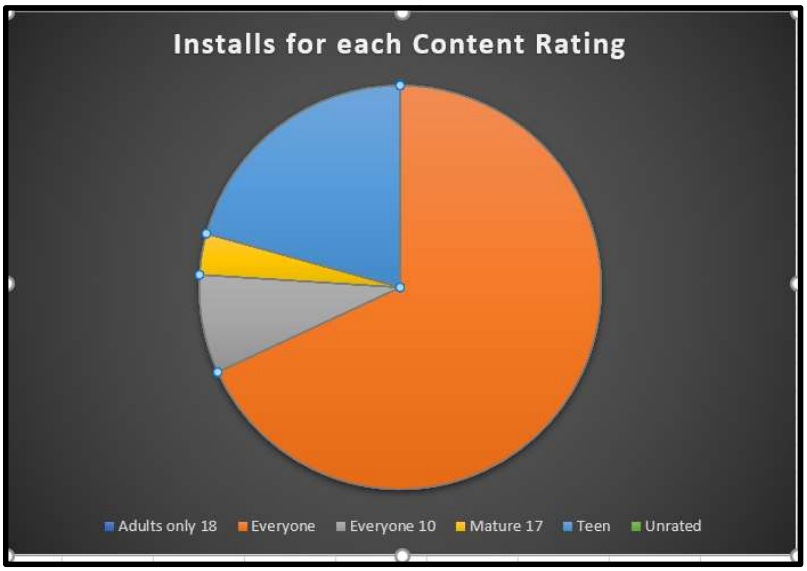
ANALYSING THE DATASET

BAR GRAPHS, PIE CHARTS & FREQUENCY DISTRIBUTION



DESCRIPTION: It shows a bar graph for the number of installs for each category.

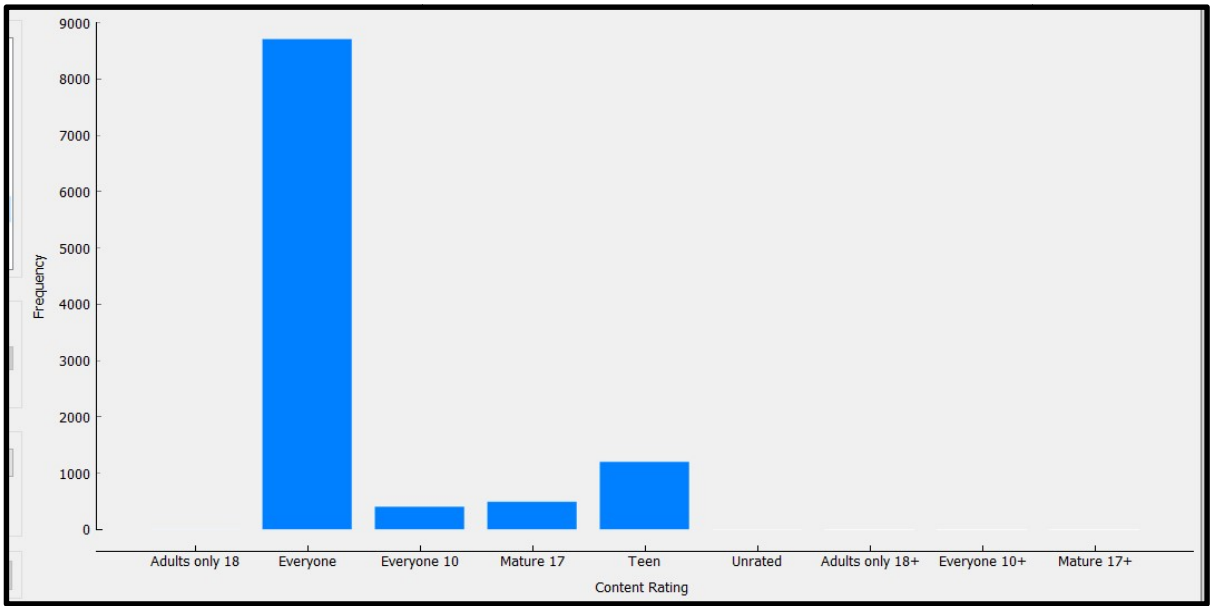
INFERENCE: We observe APPS with category Game were installed the most followed by communication, social productivity, tools.



DESCRIPTION: Pie chart showing installs for each category.

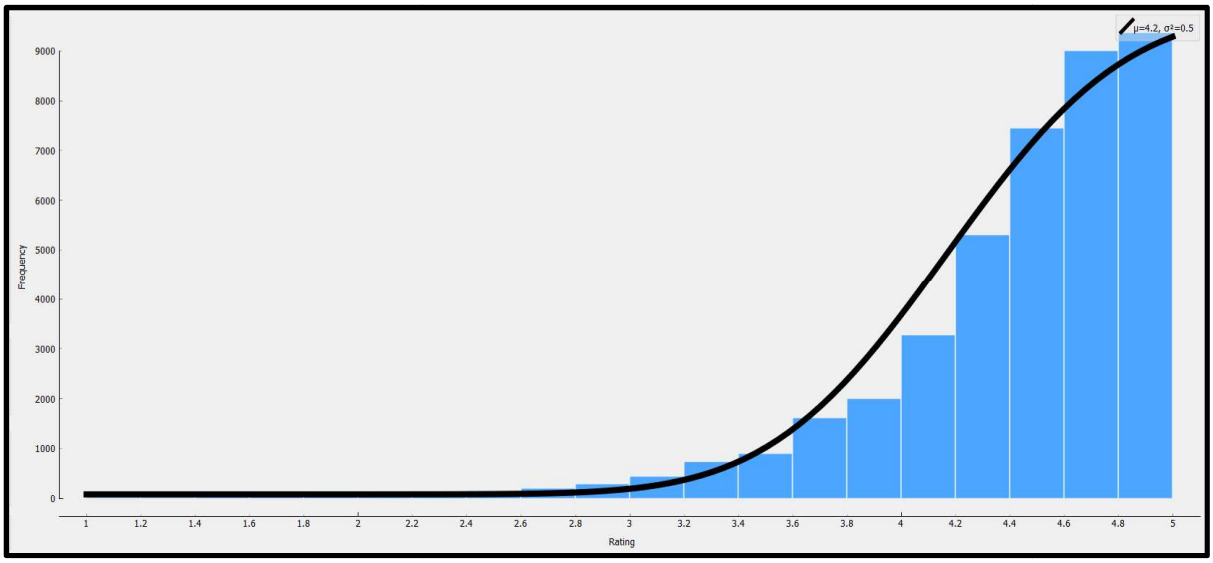
INFERENCE: Content rating ‘everyone’ is most installed followed by Teen.

Content Rating:



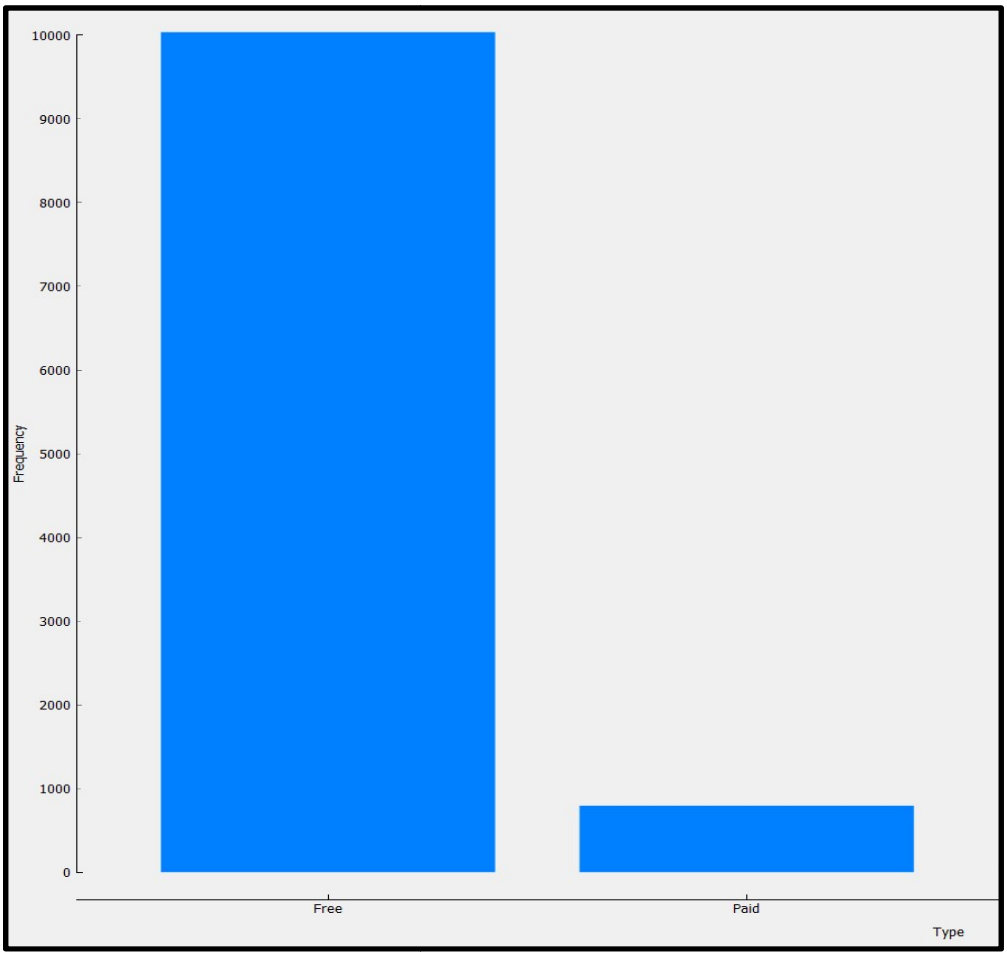
Most of the apps have a content rating of everyone hence skewed towards content rating of everyone.

App Rating:



This shows cumulative distributions of App rating attribute.

Type (Fee/Paid)



Most of the apps in our data are of type free. Our data is skewed towards type of free.

ASSOCIATION RULE MINING

Association Rule Mining is one of the ways to find patterns in data. It finds:

- features (dimensions) which occur together
- features (dimensions) which are "correlated"

Market Basket Analysis is a popular application of Association Rules.

The measures of effectiveness of the rule are as Follows:

- Support
- Confidence
- Lift
- Others: Affinity, Leverage

PERFORMING ASSOCIATION RULE MINING

No.	1: Category Nominal	2: Installs Nominal	3: Type Nominal	4: **Content Rating** Nominal
1	ART_AND_DESIGN	LOW	Free	Everyone
2	ART_AND_DESIGN	LOW	Free	Everyone
3	ART_AND_DESIGN	LOW	Free	Everyone
4	ART_AND_DESIGN	LOW	Free	Everyone
5	ART_AND_DESIGN	LOW	Free	Everyone
6	ART_AND_DESIGN	LOW	Free	Everyone
7	ART_AND_DESIGN	LOW	Free	Teen
8	ART_AND_DESIGN	LOW	Free	Everyone
9	ART_AND_DESIGN	LOW	Free	Everyone
10	ART_AND_DESIGN	LOW	Free	Everyone

DATASET: It has attributes category installs type content rating which are all nominal.

We perform association rule mining using APRIORI Algorithm

APRIORI ALGORITHM

APRIORI PROPERTY: - All subsets of a frequent item set must be frequent(Apriori property).

If an item set is infrequent, all its supersets will be infrequent.

ALGORITHM

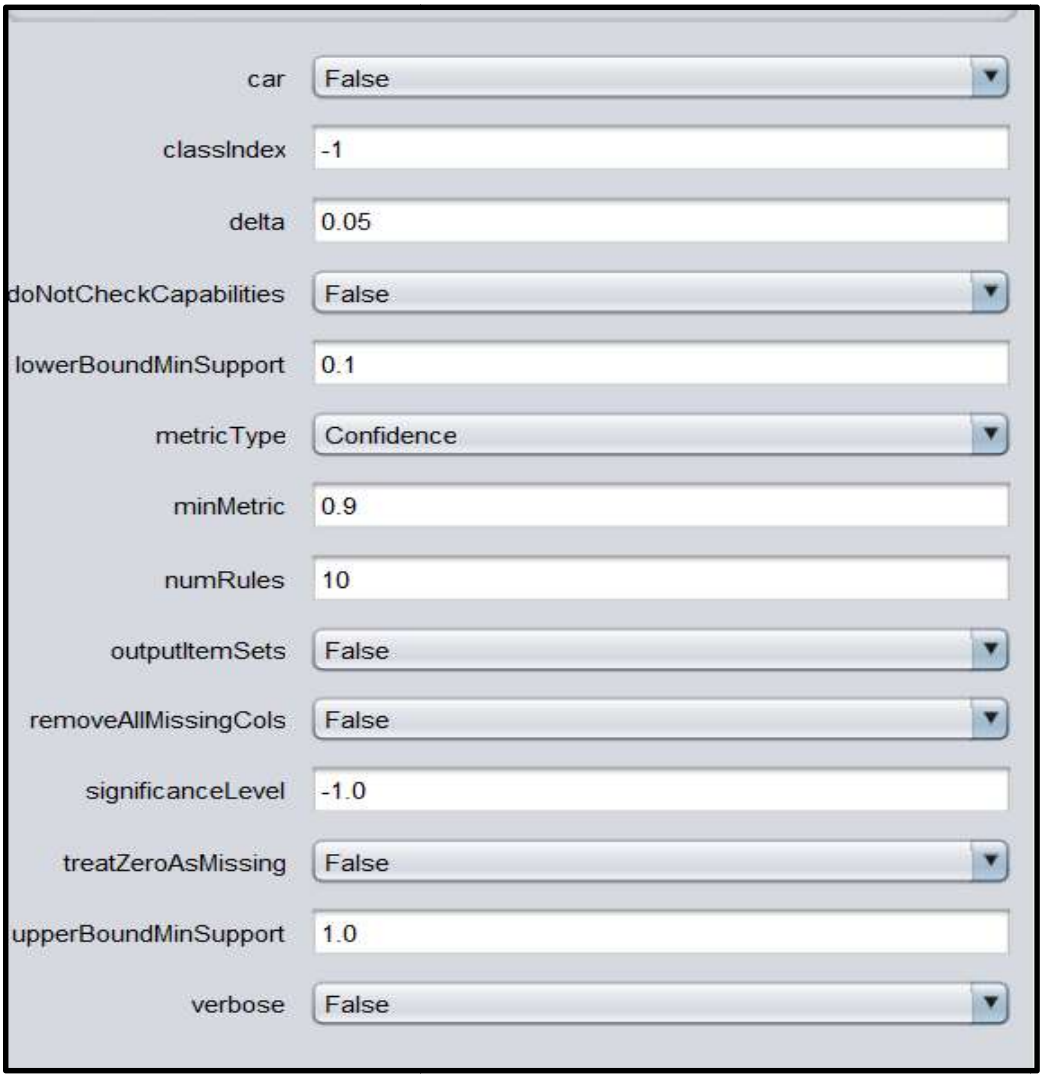
Step 1, Start with item sets containing just a single item.

Step 2. Determine the support for item sets. Keep the item sets that meet your minimum support threshold, and remove item sets that do not.

Step 3. Using the item sets you have kept from Step 1, generate all the possible item set configurations.

Step 4. Repeat Steps 1 & 2 until there are no more new item sets.

```
Apriori
=======

Minimum support: 0.1 (1084 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 18
```



PARAMETERS:

MINIMUM SUPPORT is set to 10%

MAXIMUM SUPPORT is set to 100%

METRIC TYPE is set to confidence

MINIMUM CONFIDENCE is set to 90% represented by min metric

MINIMUM RULES is set to 10

DELTA VALUE is set to 5% (It represents how in each iteration support value is decreased .we go from maximum support to minimum support and in each iteration we decreases by delta value until we get required number of rules)

NUMBER OF CYCLES This is the output parameter and its value comes out to be 18 , it means Apriori algorithm performs 18 iterations.

ITEMSETS

It shows the item sets with large support values

L(1) represents Item set with 1 attribute value.

L(2) represents Item set with 2 attribute values.

L(3) represents Item set with 3 attribute values.

NOTE: We did not get L(4) as there were no item sets which have 4 attribute values and also satisfies minimum support conditions at the same time.

```
Generated sets of large itemsets:

Size of set of large itemsets L(1): 10

Large Itemsets L(1):
Category=GAME 1144
Category=FAMILY 1971
Installs=LOW 2438
Installs=MEDIUM 2187
Installs=HIGH 2331
Installs=VERY HIGH 2080
Installs=VERY LOW 1803
Type=Free 10039
Content Rating=Everyone 8715
Content Rating=Teen 1208
```

```
Size of set of large itemsets L(2): 14

Large Itemsets L(2):
Category=FAMILY Type=Free 1780
Category=FAMILY Content Rating=Everyone 1529
Installs=LOW Type=Free 2093
Installs=LOW Content Rating=Everyone 2141
Installs=MEDIUM Type=Free 2042
Installs=MEDIUM Content Rating=Everyone 1732
Installs=HIGH Type=Free 2307
Installs=HIGH Content Rating=Everyone 1780
Installs=VERY HIGH Type=Free 2077
Installs=VERY HIGH Content Rating=Everyone 1481
Installs=VERY LOW Type=Free 1520
Installs=VERY LOW Content Rating=Everyone 1581
Type=Free Content Rating=Everyone 8020
Type=Free Content Rating=Teen 1156
```

```
Size of set of large itemsets L(3): 6

Large Itemsets L(3):
Category=FAMILY Type=Free Content Rating=Everyone 1382
Installs=LOW Type=Free Content Rating=Everyone 1833
Installs=MEDIUM Type=Free Content Rating=Everyone 1623
Installs=HIGH Type=Free Content Rating=Everyone 1763
Installs=VERY HIGH Type=Free Content Rating=Everyone 1481
Installs=VERY LOW Type=Free Content Rating=Everyone 1320
```

CONCLUSION AND RESULTS

```
Best rules found:

 1. Installs=VERY HIGH Content Rating=Everyone 1481 ==> Type=Free 1481    <conf:(1)>
 2. Installs=VERY HIGH 2080 ==> Type=Free 2077    <conf:(1)>
 3. Installs=HIGH Content Rating=Everyone 1780 ==> Type=Free 1763    <conf:(0.99)>
 4. Installs=HIGH 2331 ==> Type=Free 2307    <conf:(0.99)>
 5. Content Rating=Teen 1208 ==> Type=Free 1156    <conf:(0.96)>
 6. Installs=MEDIUM Content Rating=Everyone 1732 ==> Type=Free 1623    <conf:(0.94)>
 7. Installs=MEDIUM 2187 ==> Type=Free 2042    <conf:(0.93)>
 8. Content Rating=Everyone 8715 ==> Type=Free 8020    <conf:(0.92)>
 9. Category=FAMILY Content Rating=Everyone 1529 ==> Type=Free 1382    <conf:(0.9)>
10. Category=FAMILY 1971 ==> Type=Free 1780    <conf:(0.9)>
```

ASSOCIATION RULES AND INFERENCES

Above are the 10 Best association rules.

RULE NO. 8 has the maximum support of about 80% so we can easily infer from that if the Content rating is Everyone then the type of app will be Free.

And similarly we can use other 9 rules also and infer from them.

ONE-R TRAINING

THEORY

OneR, short for "One Rule" is a simple classification algorithm that generates one rule for each predictor in the data, then selects the rule with the smallest total error as its "one rule". To create a rule for a predictor, we construct a frequency table for each predictor against the target.

ALGORITHM

For each predictor,

For each value of that predictor, make a rule as follows;

Count how often each value of target (class) appears

Find the most frequent class

Make the rule assign that class to this value of the predictor

Calculate the total error of the rules of each predictor

Choose the predictor with the smallest total error.

EVALUATION METHOD :- Stratified cross validation with 10 folds

BUCKET SIZE	ACCURACY
1	76.1
2	77.6
3	78.5
4	78.5
5	78.6
6	78.8
7	79.2
8	79.2
9	79.2
10	79.1
11	79.0
12	79.0
13	79.1
14	79.5
15	79.6
16	79.6

17	79.6
18	79.7
50	79.86

We have found that the Best bucket size is 50 as it gives 79.86% accuracy.

Following are the rules made by One-R:

```
Reviews:
        < 9.5    -> VERY LOW
        < 295.5 -> LOW
        < 6817.0         -> MEDIUM
        < 7580.5         -> HIGH
        < 8390.5         -> MEDIUM
        < 104527.5       -> HIGH
        >= 104527.5      -> VERY HIGH
(8679/10839 instances correct)
```

CONFUSION MATRIX

```
=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
 1785  354    1    0  298 |    a = LOW
  170 1710  299    1    7 |    b = MEDIUM
    3  247 1796  284    1 |    c = HIGH
    1    7  269 1803    0 |    d = VERY HIGH
  241    0    0    0 1562 |    e = VERY LOW
```

NAIVE BAYES CLASSIFIER

A Naive Bayes classifier is a probabilistic machine learning model that's used for classification tasks. The crux of the classifier is based on the Bayes theorem.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Two assumptions are made:-

- Predictors are Independent
- All the predictors have an equal effect on the outcome

TABLE

Our target class variable is Install. Following table shows the Conditional probabilities and prior probabilities.

Attribute	Class MEDIUM (0.3)	HIGH (0.41)	LOW (0.29)
Category			
ART_AND_DESIGN	43.0	14.0	11.0
AUTO_AND_VEHICLES	40.0	19.0	29.0
BEAUTY	33.0	11.0	12.0
BOOKS_AND_REFERENCE	75.0	62.0	97.0
BUSINESS	108.0	98.0	257.0
COMICS	35.0	19.0	9.0
COMMUNICATION	64.0	213.0	113.0
DATING	96.0	59.0	82.0
EDUCATION	47.0	109.0	3.0
ENTERTAINMENT	18.0	133.0	1.0
EVENTS	22.0	7.0	38.0
FINANCE	143.0	112.0	114.0
FOOD_AND_DRINK	44.0	64.0	22.0
HEALTH_AND_FITNESS	100.0	157.0	87.0
HOUSE_AND_HOME	35.0	44.0	12.0
LIBRARIES_AND_DEMO	47.0	15.0	26.0
LIFESTYLE	132.0	90.0	163.0
GAME	279.0	738.0	130.0
FAMILY	698.0	652.0	624.0
MEDICAL	167.0	27.0	272.0
SOCIAL	64.0	157.0	77.0
SHOPPING	57.0	162.0	44.0
PHOTOGRAPHY	58.0	227.0	53.0
SPORTS	106.0	175.0	106.0
TRAVEL_AND_LOCAL	64.0	133.0	64.0
TOOLS	276.0	290.0	280.0
PERSONALIZATION	107.0	127.0	161.0
PRODUCTIVITY	86.0	198.0	143.0
PARENTING	41.0	16.0	6.0

Rating			
mean	4.098	4.2947	4.1745
std. dev.	0.5413	0.3197	0.8114
weight sum	3165	4409	1792
precision	0.1053	0.1053	0.1053
Reviews			
mean	2194.5097	1089122.0847	0
std. dev.	7718.9602	4512351.8429	2171.0641
weight sum	3241	4411	3187
precision	13026.3843	13026.3843	13026.3843
Type			
Free	2968.0	4385.0	2689.0
Paid	275.0	28.0	500.0
[total]	3243.0	4413.0	3189.0
Price			
mean	1.2274	0.0179	2.1585
std. dev.	17.9993	0.7326	23.082
weight sum	3241	4411	3187
precision	4.3956	4.3956	4.3956
Content Rating			
Everyone	2641.0	3262.0	2815.0
Teen	310.0	662.0	239.0
Everyone 10	118.0	258.0	39.0
Mature 17	173.0	232.0	97.0
Adults only 18	3.0	2.0	1.0
Unrated	2.0	1.0	2.0
[total]	3247.0	4417.0	3193.0

RESULT AND ACCURACY

Stratified cross validation using 10 folds is used for testing.

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        7884               72.7373 %
Incorrectly Classified Instances      2955               27.2627 %
```

CONFUSION MATRIX

```
=== Confusion Matrix ===

    a    b    c   <-- classified as
 1957   54 1230 |    a = MEDIUM
  820 3521   70 |    b = HIGH
  781    0 2406 |    c = LOW
```

PREDICTIVE MODELS

Predictive modeling, also called predictive analytics, is a mathematical process that seeks to predict future events or outcomes by analyzing patterns that are likely to forecast future results.

LINEAR REGRESSION

Regression algorithms fall under the family of Supervised Machine Learning algorithms which is a subset of machine learning algorithms. One of the main features of supervised learning algorithms is that they model dependencies and relationships between the target output and input features to predict the value for new data. Regression algorithms predict the output values based on input features from the data fed in the system. The go-to methodology is the algorithm builds a model on the features of training data and using the model to predict value for new data.

Regression is a data mining technique used to predict a range of numeric values (also called *continuous values*), given a particular dataset. For example, regression might be used to predict the cost of a product or service, given other variables.

PREDICTION USING REGRESSION:

	Linear Regression	Rating	Category	Content Rating	Reviews	Installs	Type	Price
1	4.2	3.8	PRODUCTIVITY	Everyone	8226	10000000	Free	0.00
2	4.2	4.0	SPORTS	Teen	28895	5000000	Free	0.00
3	4.2	3.1	FAMILY	Everyone	52	5000	Free	0.00
4	4.3	3.5	FAMILY	Everyone	13	100	Paid	0.99
5	4.2	4.5	COMMUNICAT...	Everyone	5150801	100000000	Free	0.00
6	4.2	4.5	FAMILY	Everyone 10	96658	1000000	Free	0.00
7	4.2	?	PHOTOGRAPHY	Everyone	0	5	Free	0.00
8	4.1	4.5	VIDEO_PLAYERS	Everyone	1013867	50000000	Free	0.00
9	4.1	4.2	FINANCE	Everyone	36746	5000000	Free	0.00
10	4.1	3.2	VIDEO_PLAYERS	Everyone	39	10000	Free	0.00
11	4.2	3.8	PHOTOGRAPHY	Everyone	6	1000	Free	0.00
12	4.3	4.6	FAMILY	Everyone	190086	1000000	Paid	2.99
13	4.2	4.5	SOCIAL	Everyone	228737	10000000	Free	0.00
14	4.3	4.2	ART_AND_DESI...	Everyone	1015	100000	Free	0.00
15	4.3	4.2	COMMUNICAT...	Everyone	10790092	500000000	Free	0.00
16	4.1	2.2	VIDEO_PLAYERS	Everyone	32	5000	Free	0.00
17	4.2	4.7	FAMILY	Everyone	8600	500000	Free	0.00
18	4.0	4.7	DATING	Mature 17	6	100	Free	0.00
19	4.1	5.0	LIFESTYLE	Everyone	2	100	Free	0.00
20	4.1	4.3	MAPS_AND_NA...	Everyone	70556	10000000	Free	0.00
21	4.3	4.3	GAME	Everyone	1295606	100000000	Free	0.00
22	4.3	4.7	BEAUTY	Everyone	900	5000	Free	0.00
23	4.1	4.3	ENTERTAINMENT	Mature 17	88185	10000000	Free	0.00
24	4.3	4.6	HEALTH_AND_...	Everyone	524299	10000000	Free	0.00
25	4.1	4.2	COMMUNICAT...	Everyone	104990	5000000	Free	0.00

INTERCEPTS IN REGRESSION MODEL

After building the regression model we get a certain set of intercept values corresponding to the independent variables on the basis of which the target or dependent variable is predicted.

The intercepts or coefficients:

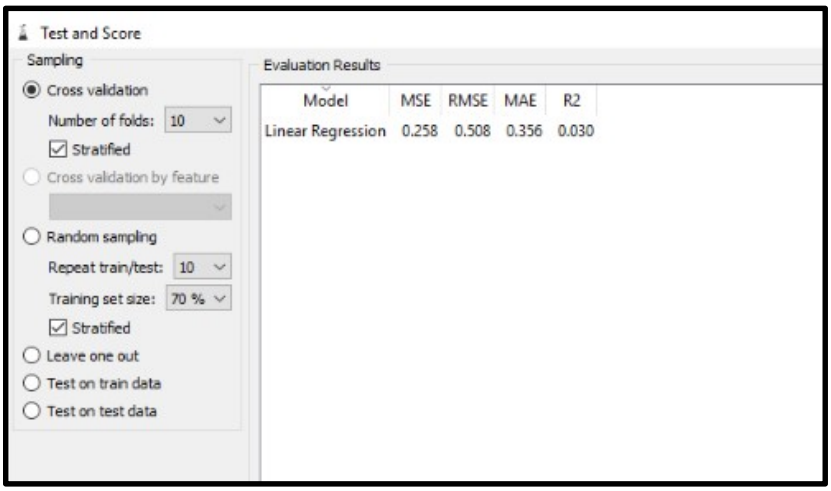
	name	coef
18	Category=HOUSE_AND_HOME	0.00505962
19	Category=LIBRARIES_AND_DEMO	-0.013012
20	Category=LIFESTYLE	-0.0978033
21	Category=MAPS_AND_NAVIGATION	-0.146116
22	Category=MEDICAL	-0.0235053
23	Category=NEWS_AND_MAGAZINES	-0.0751596
24	Category=PARENTING	0.104721
25	Category=PERSONALIZATION	0.120394
26	Category=PHOTOGRAPHY	-0.0141798
27	Category=PRODUCTIVITY	0.0075917
28	Category=SHOPPING	0.0591398
29	Category=SOCIAL	0.0314017
30	Category=SPORTS	0.0208075
31	Category=TOOLS	-0.156702
32	Category=TRAVEL_AND_LOCAL	-0.0915482
33	Category=VIDEO_PLAYERS	-0.142795
34	Category=WEATHER	0.0414406
35	Content Rating=Adults only 18	0.093305
36	Content Rating=Everyone	-0.039985
37	Content Rating=Everyone 10	-0.0131092
38	Content Rating=Mature 17	-0.0401679
39	Content Rating=Teen	-0.0255212
40	Content Rating=Unrated	0.0254782
41	Reviews	7.96802e-09
42	Installs	1.30186e-10
43	Type=Free	-0.0463822
44	Type=Paid	0.0463822
45	Price	-0.000902201

REGRESSION MODEL EVALUATION METRICS

The MSE, MAE, RMSE, and R-Squared metrics are mainly used to evaluate the prediction error rates and model performance in regression analysis.

- **MAE** (Mean absolute error) represents the difference between the original and predicted values extracted by averaging the absolute difference over the data set.
- **MSE** (Mean Squared Error) represents the difference between the original and predicted values extracted by squaring the average difference over the data set.

- **RMSE** (Root Mean Squared Error) is the error rate by the square root of MSE.
- **R-squared** (Coefficient of determination) represents the coefficient of how well the values fit compared to the original values. The value from 0 to 1 interpreted as percentages.



KNN ALGORITHM

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.

The KNN algorithm hinges on this assumption being true enough for the algorithm to be useful. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics calculating the distance between points on a graph.

There are other ways of calculating distance, and one way might be preferable depending on the problem we are solving. However, the straight-line distance (also called the Euclidean distance) is a popular and familiar choice.

The KNN Algorithm

1. Load the data
2. Initialize K to your chosen number of neighbors
3. for each example in the data

3.1 Calculate the distance between the query example and the current example from the data.

3.2 Add the distance and the index of the example to an ordered collection

4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances

5. Pick the first K entries from the sorted collection

6. Get the labels of the selected K entries

7. If regression, return the mean of the K labels

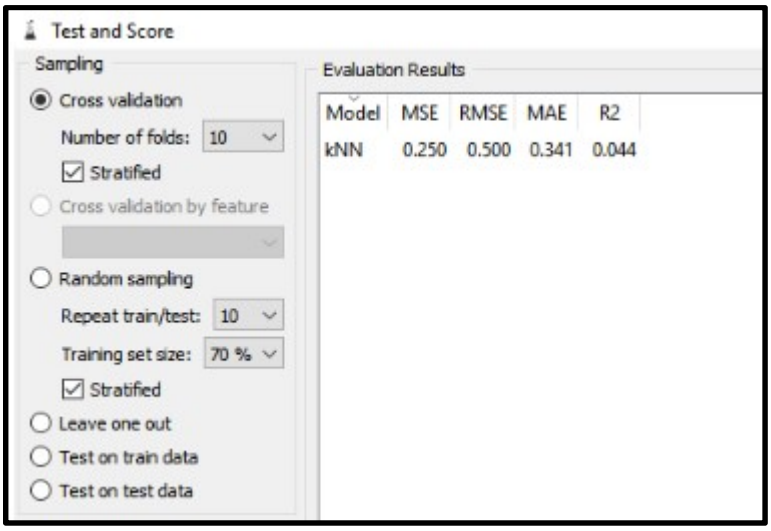
8. If classification, return the mode of the K labels

PREDICTION USING THE KNN ALGORITHM

	kNN	Rating	Category	Content Rating	Reviews	Installs	Type	Price
1	4.0	4.1	ART_AND_DESI...	Everyone	159	10000	Free	0.00
2	4.1	3.9	ART_AND_DESI...	Everyone	967	500000	Free	0.00
3	4.3	4.7	ART_AND_DESI...	Everyone	87510	5000000	Free	0.00
4	4.1	4.5	ART_AND_DESI...	Teen	215644	50000000	Free	0.00
5	3.8	4.3	ART_AND_DESI...	Everyone	967	100000	Free	0.00
6	3.8	4.4	ART_AND_DESI...	Everyone	167	50000	Free	0.00
7	3.9	3.8	ART_AND_DESI...	Everyone	178	50000	Free	0.00
8	4.5	4.1	ART_AND_DESI...	Everyone	36815	1000000	Free	0.00
9	4.3	4.4	ART_AND_DESI...	Everyone	13791	1000000	Free	0.00
10	3.8	4.7	ART_AND_DESI...	Everyone	121	10000	Free	0.00
11	4.3	4.4	ART_AND_DESI...	Everyone	13880	1000000	Free	0.00
12	4.2	4.4	ART_AND_DESI...	Everyone	8788	1000000	Free	0.00
13	4.1	4.2	ART_AND_DESI...	Teen	44829	10000000	Free	0.00
14	4.3	4.6	ART_AND_DESI...	Everyone	4326	100000	Free	0.00
15	4.2	4.4	ART_AND_DESI...	Everyone	1518	100000	Free	0.00
16	3.8	3.2	ART_AND_DESI...	Everyone	55	5000	Free	0.00
17	4.2	4.7	ART_AND_DESI...	Everyone	3632	500000	Free	0.00
18	4.2	4.5	ART_AND_DESI...	Everyone	27	10000	Free	0.00
19	4.6	4.3	ART_AND_DESI...	Everyone	194216	5000000	Free	0.00
20	4.3	4.6	ART_AND_DESI...	Everyone	224399	10000000	Free	0.00
21	4.0	4.0	ART_AND_DESI...	Everyone	450	100000	Free	0.00
22	4.0	4.1	ART_AND_DESI...	Everyone	654	100000	Free	0.00
23	4.2	4.7	ART_AND_DESI...	Everyone	7699	500000	Free	0.00
24	4.0	?	ART_AND_DESI...	Everyone	61	100000	Free	0.00
25	3.9	4.7	ART_AND_DESI...	Everyone	118	50000	Free	0.00
26	4.0	4.8	ART_AND_DESI...	Everyone	192	10000	Free	0.00

MODEL EVALUATION METRICS

The MSE, MAE, RMSE, and R-Squared metrics are mainly used to evaluate the prediction error rates and model performance in regression analysis.



TREE

A prediction or decision tree is a tree shaped diagram that shows statistical probability or determines a course of action. It shows the steps to take and why one choice may lead to another. Therefore, it is a suitable decision-making tool for research analysis or for planning the strategy to reach a goal.

A prediction tree has three main parts, a root, leaf nodes and branches. The root node is the target value that we are seeking to reach. The leaf nodes contain the information about criteria. The branches connect the nodes and show the route through the leaves to the target value.

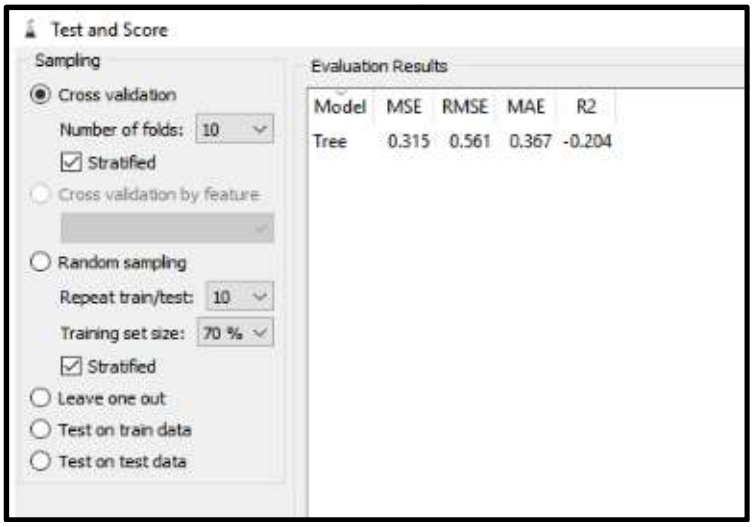
There are only a few steps required to make a prediction tree:

1. Select the target value or starting point, which will be the root node.
2. Highlight the columns that you would like to involve.
3. The branches will be created automatically and show the strength of the connections between the nodes.

PREDICTION USING TREE

	Tree	Rating	Category	Content Rating	Reviews	Installs	Type	Price
1	4.6	4.1	ART_AND_DESI...	Everyone	159	10000	Free	0.00
2	4.2	3.9	ART_AND_DESI...	Everyone	967	500000	Free	0.00
3	4.7	4.7	ART_AND_DESI...	Everyone	87510	5000000	Free	0.00
4	4.4	4.5	ART_AND_DESI...	Teen	215644	50000000	Free	0.00
5	4.2	4.3	ART_AND_DESI...	Everyone	967	100000	Free	0.00
6	4.1	4.4	ART_AND_DESI...	Everyone	167	50000	Free	0.00
7	4.1	3.8	ART_AND_DESI...	Everyone	178	50000	Free	0.00
8	4.1	4.1	ART_AND_DESI...	Everyone	36815	1000000	Free	0.00
9	4.5	4.4	ART_AND_DESI...	Everyone	13791	1000000	Free	0.00
10	4.7	4.7	ART_AND_DESI...	Everyone	121	10000	Free	0.00
11	4.5	4.4	ART_AND_DESI...	Everyone	13880	1000000	Free	0.00
12	4.6	4.4	ART_AND_DESI...	Everyone	8788	1000000	Free	0.00
13	4.1	4.2	ART_AND_DESI...	Teen	44829	10000000	Free	0.00
14	4.6	4.6	ART_AND_DESI...	Everyone	4326	100000	Free	0.00
15	4.1	4.4	ART_AND_DESI...	Everyone	1518	100000	Free	0.00
16	3.7	3.2	ART_AND_DESI...	Everyone	55	5000	Free	0.00
17	4.6	4.7	ART_AND_DESI...	Everyone	3632	500000	Free	0.00
18	4.2	4.5	ART_AND_DESI...	Everyone	27	10000	Free	0.00
19	4.4	4.3	ART_AND_DESI...	Everyone	194216	5000000	Free	0.00
20	4.5	4.6	ART_AND_DESI...	Everyone	224399	10000000	Free	0.00
21	3.7	4.0	ART_AND_DESI...	Everyone	450	100000	Free	0.00
22	4.0	4.1	ART_AND_DESI...	Everyone	654	100000	Free	0.00
23	4.7	4.7	ART_AND_DESI...	Everyone	7699	500000	Free	0.00
24	4.2	?	ART_AND_DESI...	Everyone	61	100000	Free	0.00
25	4.7	4.7	ART_AND_DESI...	Everyone	118	50000	Free	0.00
26	4.6	4.8	ART_AND_DESI...	Everyone	192	10000	Free	0.00

MODEL EVALUATION METRICS

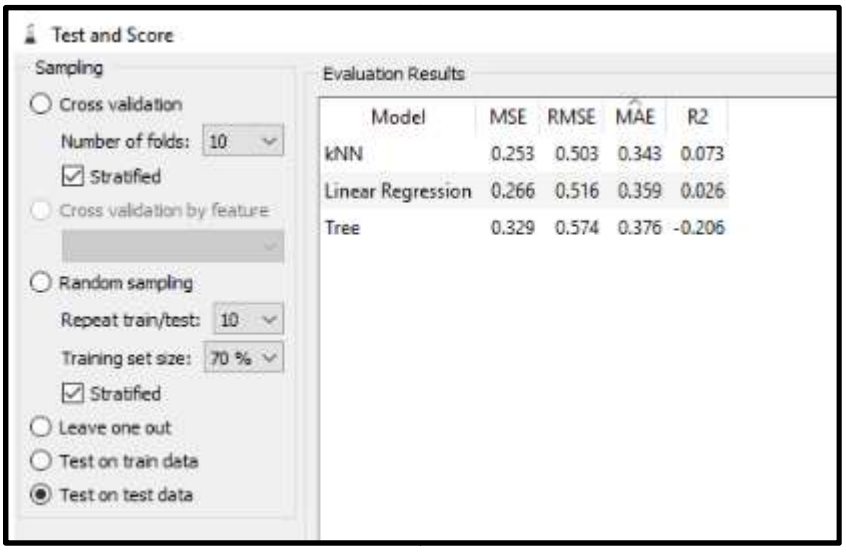


COMPARING THE MODELS

PREDICTED VALUES

	Linear Regression	kNN	Tree	Rating	Category	Content Rating	Reviews	Installs	Type	Price
1	4.2	3.6	3.9	3.8	PRODUCTIVITY	Everyone	8226	10000000	Free	0.00
2	4.2	4.1	4.0	4.0	SPORTS	Teen	28895	5000000	Free	0.00
3	4.2	4.1	3.5	3.1	FAMILY	Everyone	52	5000	Free	0.00
4	4.3	4.5	4.5	3.5	FAMILY	Everyone	13	100	Paid	0.99
5	4.2	4.4	4.5	4.5	COMMUNICAT...	Everyone	5150801	100000000	Free	0.00
6	4.2	4.3	4.5	4.5	FAMILY	Everyone 10	96658	1000000	Free	0.00
7	4.2	4.5	5.0	?	PHOTOGRAPHY	Everyone	0	5	Free	0.00
8	4.1	4.4	4.4	4.5	VIDEO_PLAYERS	Everyone	1013867	50000000	Free	0.00
9	4.1	4.1	4.2	4.2	FINANCE	Everyone	36746	5000000	Free	0.00
10	4.1	3.9	3.2	3.2	VIDEO_PLAYERS	Everyone	39	10000	Free	0.00
11	4.2	4.4	3.7	3.8	PHOTOGRAPHY	Everyone	6	1000	Free	0.00
12	4.3	4.6	4.7	4.6	FAMILY	Everyone	190086	1000000	Paid	2.99
13	4.2	4.3	4.4	4.5	SOCIAL	Everyone	228737	10000000	Free	0.00
14	4.3	3.6	4.2	4.2	ART_AND_DESI...	Everyone	1015	100000	Free	0.00
15	4.3	4.3	4.2	4.2	COMMUNICAT...	Everyone	10790092	500000000	Free	0.00
16	4.1	3.7	3.3	2.2	VIDEO_PLAYERS	Everyone	32	5000	Free	0.00
17	4.2	4.2	4.7	4.7	FAMILY	Everyone	8600	500000	Free	0.00
18	4.0	4.3	4.7	4.7	DATING	Mature 17	6	100	Free	0.00
19	4.1	4.5	4.3	5.0	LIFESTYLE	Everyone	2	100	Free	0.00
20	4.1	4.1	4.3	4.3	MAPS_AND_NA...	Everyone	70556	10000000	Free	0.00
21	4.3	4.3	4.3	4.3	GAME	Everyone	1295606	100000000	Free	0.00
22	4.3	4.6	4.7	4.7	BEAUTY	Everyone	900	5000	Free	0.00

MODEL EVALUATION METRICS



CORRELATION

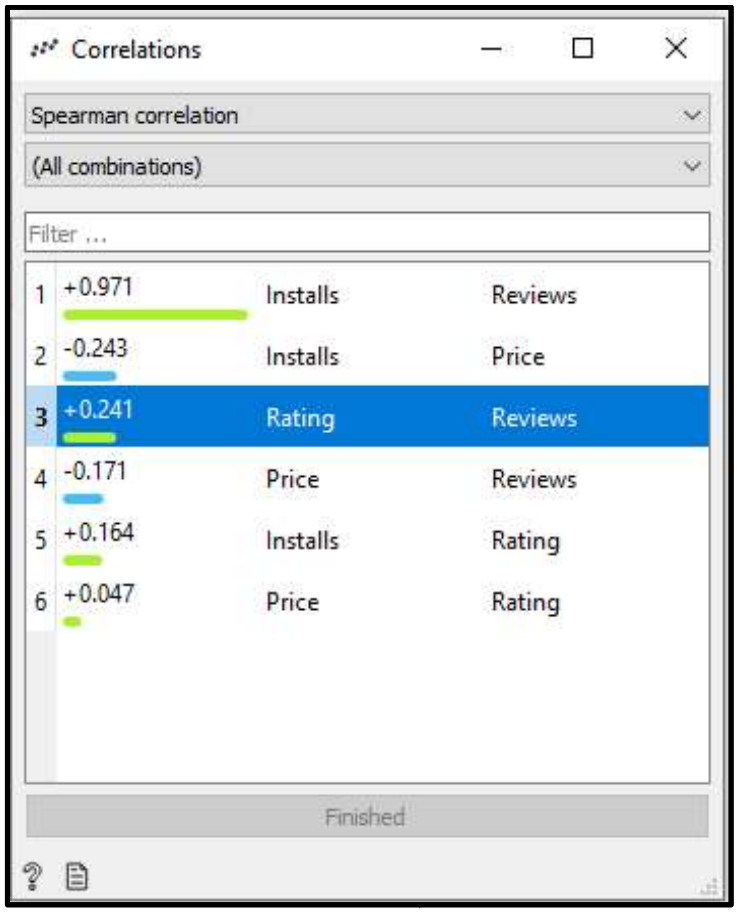
Correlation is a statistical technique that can show whether and how strongly pairs of variables are related.

SPEARMAN CORRELATION

In statistics, Spearman's rank correlation coefficient or Spearman's ρ, often denoted by the Greek letter (rho) , is a nonparametric measure of rank correlation (statistical dependence between the rankings of two variables). It assesses how well the relationship between two variables can be described using a monotonic function.

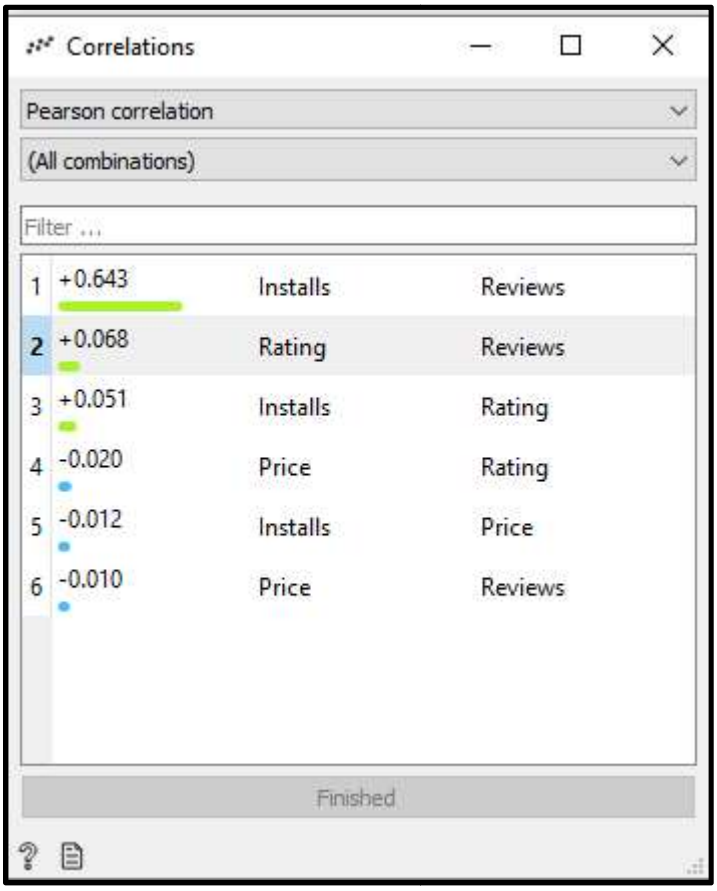
The Spearman correlation between two variables is equal to the Pearson correlation between the rank values of those two variables; while Pearson's correlation assesses linear relationships, Spearman's correlation assesses monotonic relationships (whether linear or not). If there are no repeated data values, a perfect Spearman correlation of +1 or −1 occurs when each of the variables is a perfect monotone function of the other.

Intuitively, the Spearman correlation between two variables will be high when observations have a similar (or identical for a correlation of 1) rank (i.e. relative position label of the observations within the variable: 1st, 2nd, 3rd, etc.) between the two variables, and low when observations have a dissimilar (or fully opposed for a correlation of −1) rank between the two variables.



PEARSON CORRELATION

In statistics, the Pearson correlation coefficient (PCC), also referred to as Pearson's *r*, the Pearson product-moment correlation coefficient (PPMCC) or the bivariate correlation, is a statistic that measures linear correlation between two variables *X* and *Y*. It has a value between +1 and −1, where 1 is total positive linear correlation, 0 is no linear correlation, and −1 is total negative linear correlation (that the value lies between -1 and 1 is a consequence of the Cauchy–Schwarz inequality).



INFERENCE:

Here, we can see in both Pearson and Spearman correlation, the attributes Installs and Reviews are highly related, having coefficients 0.643 and 0.971 respectively.

DISCRETIZATION

Discretization is the process of putting values into buckets so that there are a limited number of possible states. The buckets themselves are treated as ordered and discrete values. You can discretize both numeric and string columns. There are several methods that you can use to discretize data.

Performing discretization on the numeric attributes using frequency distribution(5 intervals):

Discretize

Default method: Equal-frequency discretization

Thresholds

Reviews: 16.50, 415.50, 9294.50, 99424.50
Installs: 750.00, 30000.00, 750000.00, 7500000.00 (equal frequency k=5)
Price: 0.49, 1.50, 3.00, 5.74
Rating: 3.95, 4.25, 4.45, 4.65

Write a comment...

Data Table

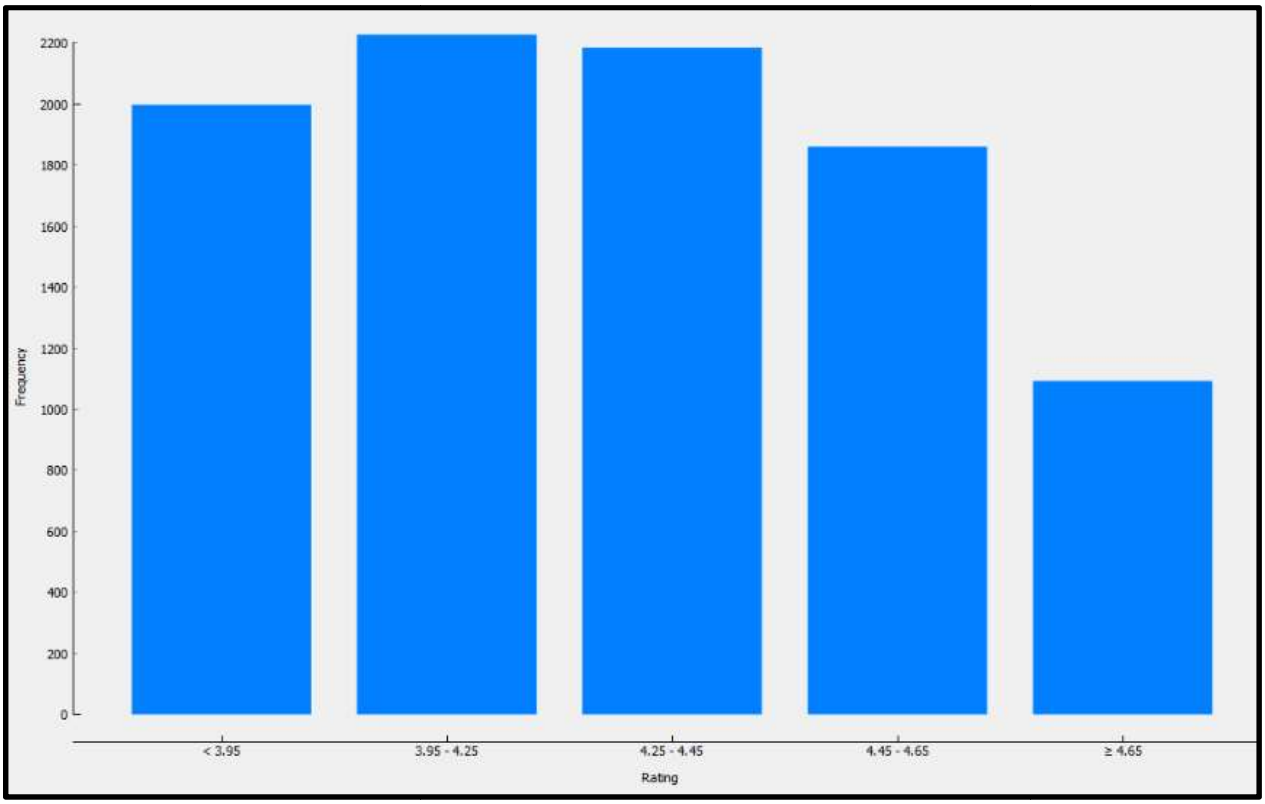
Data instances: 10839
Features: 7
Meta attributes: None

	Category	Content Rating	Reviews	Installs	Type	Price	Rating
1	ART_AND_DESIGN	Everyone	16.5 - 415.5	750 - 30000	Free	< 0.495	3.95 - 4.25
2	ART_AND_DESIGN	Everyone	415.5 - 9294.5	30000 - 750000	Free	< 0.495	< 3.95
3	ART_AND_DESIGN	Everyone	9294.5 - 99424.5	750000 - 7.5e+06	Free	< 0.495	≥ 4.65
4	ART_AND_DESIGN	Teen	≥ 99424.5	≥ 7.5e+06	Free	< 0.495	4.45 - 4.65
5	ART_AND_DESIGN	Everyone	415.5 - 9294.5	30000 - 750000	Free	< 0.495	4.25 - 4.45
6	ART_AND_DESIGN	Everyone	16.5 - 415.5	30000 - 750000	Free	< 0.495	4.25 - 4.45
7	ART_AND_DESIGN	Everyone	16.5 - 415.5	30000 - 750000	Free	< 0.495	< 3.95
8	ART_AND_DESIGN	Everyone	9294.5 - 99424.5	750000 - 7.5e+06	Free	< 0.495	3.95 - 4.25
9	ART_AND_DESIGN	Everyone	9294.5 - 99424.5	750000 - 7.5e+06	Free	< 0.495	4.25 - 4.45

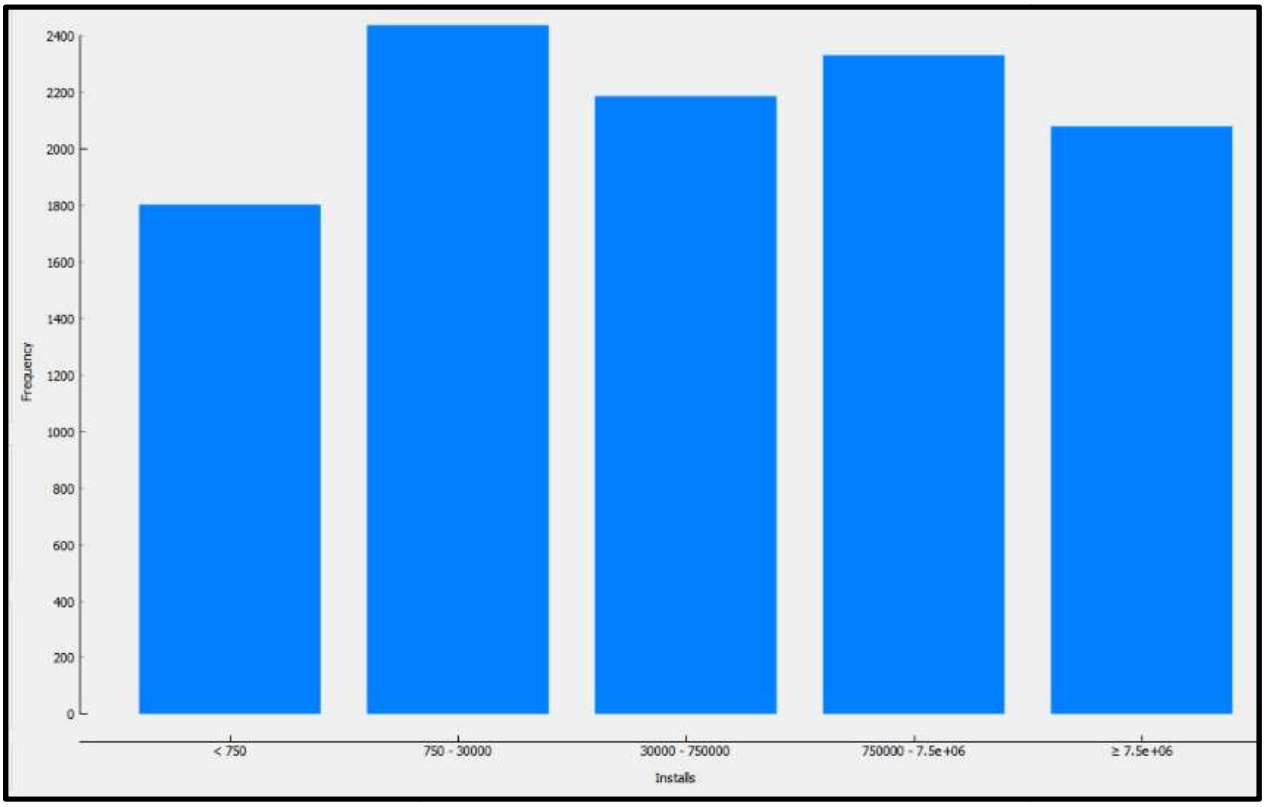
	Category	Content Rating	Reviews	Installs	Type	Price	Rating
1	ART_AND_DESI...	Everyone	16.5 - 415.5	750 - 30000	Free	< 0.495	3.95 - 4.25
2	ART_AND_DESI...	Everyone	415.5 - 9294.5	30000 - 750000	Free	< 0.495	< 3.95
3	ART_AND_DESI...	Everyone	9294.5 - 99424.5	750000 - 7.5e+06	Free	< 0.495	≥ 4.65
4	ART_AND_DESI...	Teen	≥ 99424.5	≥ 7.5e+06	Free	< 0.495	4.45 - 4.65
5	ART_AND_DESI...	Everyone	415.5 - 9294.5	30000 - 750000	Free	< 0.495	4.25 - 4.45
6	ART_AND_DESI...	Everyone	16.5 - 415.5	30000 - 750000	Free	< 0.495	4.25 - 4.45
7	ART_AND_DESI...	Everyone	16.5 - 415.5	30000 - 750000	Free	< 0.495	< 3.95
8	ART_AND_DESI...	Everyone	9294.5 - 99424.5	750000 - 7.5e+06	Free	< 0.495	3.95 - 4.25
9	ART_AND_DESI...	Everyone	9294.5 - 99424.5	750000 - 7.5e+06	Free	< 0.495	4.25 - 4.45
10	ART_AND_DESI...	Everyone	16.5 - 415.5	750 - 30000	Free	< 0.495	≥ 4.65
11	ART_AND_DESI...	Everyone	9294.5 - 99424.5	750000 - 7.5e+06	Free	< 0.495	4.25 - 4.45
12	ART_AND_DESI...	Everyone	415.5 - 9294.5	750000 - 7.5e+06	Free	< 0.495	4.25 - 4.45

STATISTICAL DISTRIBUTION OF DISCRETIZED DATA

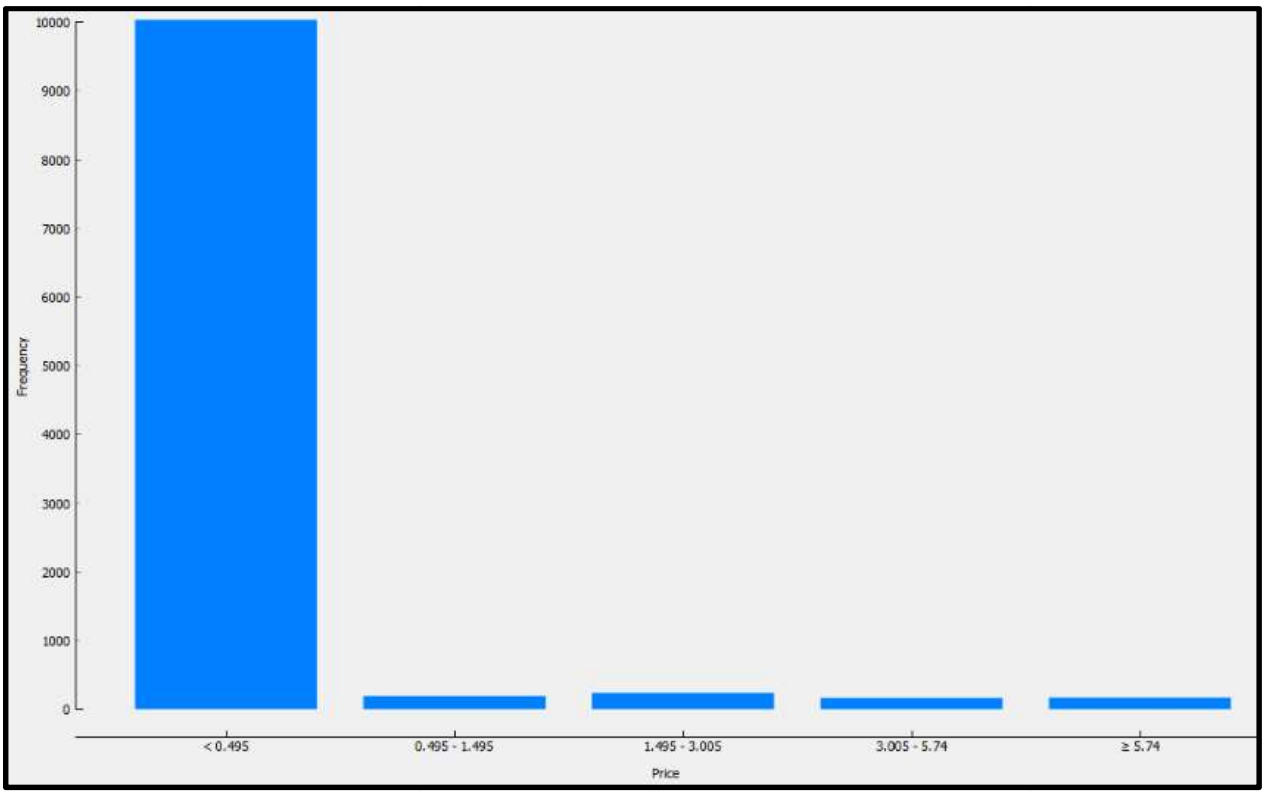
RATING:



INSTALLS:



PRICE OF APPS:



CLUSTERING

EM CLUSTERING

This operator performs clustering using the Expectation Maximization algorithm. Clustering is concerned with grouping objects together that are similar to each other and dissimilar to the objects belonging to other clusters. But the Expectation Maximization algorithm extends this basic approach to clustering in some important ways.

The general purpose of clustering is to detect clusters in examples and to assign those examples to the clusters. A typical application for this type of analysis is a marketing research study in which a number of consumer behavior related variables are measured for a large sample of respondents. The purpose of the study is to detect 'market segments', i.e., groups of respondents that are somehow more similar to each other (to all other members of the same cluster) when compared to respondents that belong to other clusters. In addition to identifying such clusters, it is usually equally of interest to determine how the clusters are different, i.e., determine the specific variables or dimensions that vary and how they vary in regard to members in different clusters.
The EM (expectation maximization) technique is similar to the K-Means technique. The basic operation of K-Means clustering algorithms is relatively simple: Given a fixed number of k clusters, assign observations to those clusters so that the means across clusters (for all variables) are as different from each other as possible. The EM algorithm extends this basic approach to clustering in two important ways:

Instead of assigning examples to clusters to maximize the differences in means for continuous variables, the EM clustering algorithm computes probabilities of cluster memberships based on one or more probability distributions. The goal of the clustering algorithm then is to maximize the overall probability or likelihood of the data, given the (final) clusters.

Expectation Maximization algorithm: The basic approach and logic of this clustering method is as follows. Suppose you measure a single continuous variable in a large sample of observations. Further, suppose that the sample consists of two clusters of observations with different means (and perhaps different standard deviations); within each sample, the distribution of values for the continuous variable follows the normal distribution. The goal of EM clustering is to estimate the means and standard deviations for each cluster so as to maximize the likelihood of the observed data (distribution). Put another way, the EM algorithm attempts to approximate the observed distributions of values based on mixtures of different distributions in different clusters. The results of EM clustering are different from those computed by k-means clustering. The latter will assign observations to clusters to maximize the distances between clusters. The EM algorithm does not compute actual assignments of observations to clusters, but classification probabilities. In other words, each observation belongs to each cluster with a certain probability. Of course, as a final result you can usually review an actual assignment of observations to clusters, based on the (largest) classification probability.

RESULTS OF EM CLUSTERING

Clusterer output

Content Rating				
Everyone	2739.6376	1367.2214	4223.9737	387.1673
Teen	951.1995	11.8196	106.8099	142.171
Everyone	1.8733	1.0088	1.1177	1.0002
Everyone 10	312.6693	7.8244	36.516	58.9903
Mature 17	440.3214	9.1737	16.0623	37.4425
Adults only 18	3.974	1.0022	1.0238	1.0001
Unrated	2.6394	1.3168	1.0431	1.0007
[total]	4452.3145	1399.3669	4386.5465	628.7721
Reviews				
mean	58632.3887	15185.5526	110032.2291	6515157.2232
std. dev.	142611.2958	54010.3044	294979.7176	10466316.8217
Installs				
mean	2066969.3449	925389.5567	4346724.4312	222139181.6479
std. dev.	3796800.2776	2838073.6588	11421358.8014	282231010.1753
Type				
Free	4445.7498	1073.607	3901.2725	622.3708
Paid	1.5647	320.7599	480.274	1.4013
[total]	4447.3145	1394.3669	4381.5465	623.7721
Price				
mean	0.0001	7.082	0.2911	0.0009
std. dev.	0.0112	43.9908	0.9686	0.0405
Rating				
mean	4.1442	3.7953	4.3369	4.3972
std. dev.	0.4541	0.8031	0.264	0.1743

Clusterer output

Category				
ART_AND_DESIGN	18.777	3.5335	45.6409	1.0486
AUTO_AND_VEHICLES	21.1076	14.4382	52.4517	1.0026
BEAUTY	39.6458	2.0702	14.2827	1.0014
BOOKS_AND_REFERENCE	76.7691	18.7442	131.4702	8.0165
BUSINESS	67.361	69.4229	324.1016	3.1144
COMICS	58.066	1.2315	3.6852	1.0173
COMMUNICATION	127.3882	24.8784	161.2953	77.4381
DATING	227.8658	3.9088	5.2082	1.0171
EDUCATION	54.5866	3.1535	97.2582	5.0018
ENTERTAINMENT	128.9987	1.7247	8.3734	13.9033
EVENTS	39.1996	3.185	24.6099	1.0055
FINANCE	24.4061	78.6881	264.1187	2.7871
FOOD_AND_DRINK	46.2943	14.8372	68.859	1.0096
HEALTH_AND_FITNESS	144.1343	29.2656	168.549	3.0511
HOUSE_AND_HOME	19.1598	9.9819	61.8547	1.0036
LIBRARIES_AND_DEMO	15.1881	9.5326	63.2773	1.002
LIFESTYLE	176.5885	65.7896	141.6077	2.0142
GAME	667.2332	26.308	241.6446	212.8142
FAMILY	1036.9085	52.8826	842.3718	42.8372
MEDICAL	107.0875	115.7603	243.1501	1.002
SOCIAL	241.7488	3.6822	15.1833	38.3857
SHOPPING	92.3964	11.1629	136.1865	24.2541
PHOTOGRAPHY	71.9493	36.2684	188.2734	42.5088
SPORTS	195.1353	20.4452	165.7412	6.6783
TRAVEL_AND_LOCAL	44.1436	43.6393	158.5968	15.6203
TOOLS	129.9042	639.0761	33.4441	44.5756
PERSONALIZATION	148.9901	12.8873	219.0244	15.0982

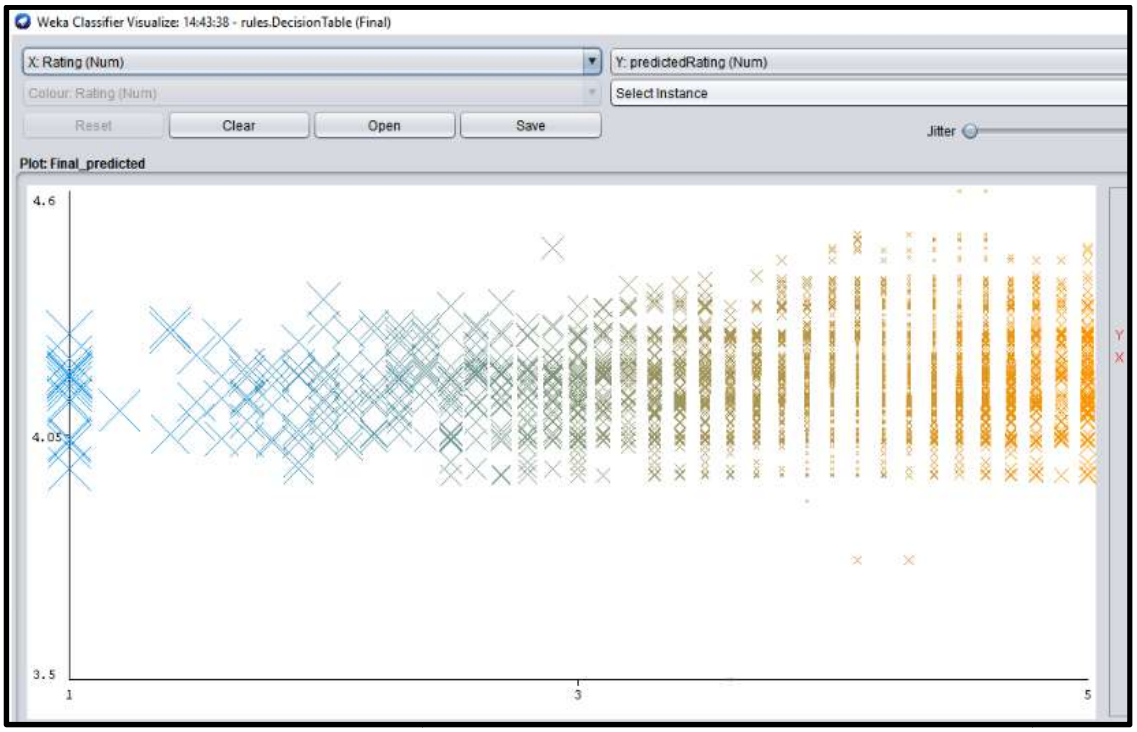
ANALYSIS USING DIFFERENT MODELS

DECISION TABLE

A decision table consists of a hierarchical table in which each entry in a higher level table gets broken down by the values of a pair of additional attributes to form another table. The structure is similar to dimensional stacking.

```
Decision Table:

Number of training instances: 9366
Number of Rules : 85
Non matches covered by Majority class.
        Best first.
        Start set: no attributes
        Search direction: forward
        Stale search after 5 node expansions
        Total number of subsets evaluated: 29
        Merit of best subset found:    0.508
Evaluation (for feature selection): CV (leave one out)
Feature set: 1,3,4,6,7

Time taken to build model: 0.53 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient                  0.1706
Mean absolute error                      0.3559
Root mean squared error                  0.5077
```

PLOTTING RATING Vs PREDICTED RATING



ADDITIVE REGRESSION MODEL

In statistics, an additive model (AM) is a nonparametric regression method. It is an essential part of the ACE algorithm. The *AM* uses a one-dimensional smoother to build a restricted class of nonparametric regression models. Because of this, it is less affected by the curse of dimensionality than e.g. a *p*-dimensional smoother. Furthermore, the *AM* is more flexible than a standard linear model, while being more interpretable than a general regression surface at the cost of approximation errors. Problems with *AM* include model selection, overfitting, and multicollinearity.

Here, we've built a 5 fold model.

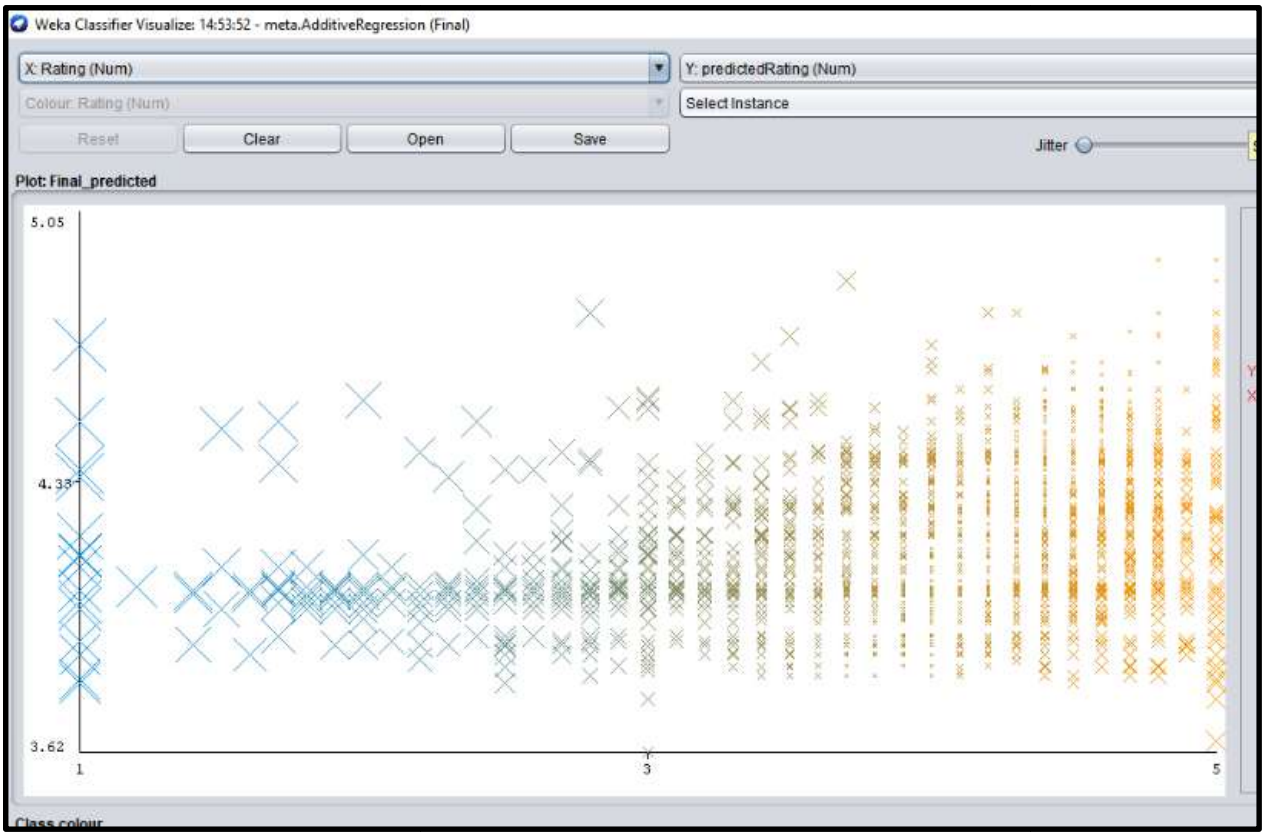
The model mainly uses the attributes Installs, Reviews and Price.

```
Model number 1

Decision Stump

Classifications

Installs <= 300.0 : 0.3319370683244167
Installs > 300.0 : -0.01659685341622249
Installs is missing : -1.5741732069048525E-15
```

```
Model number 2

Decision Stump

Classifications

Reviews <= 178828.5 : -0.02247052638200817
Reviews > 178828.5 : 0.0999605091583578
Reviews is missing : 6.430367662647148E-17
```

```
Model number 3

Decision Stump

Classifications

Price <= 0.495 : -0.009548926027288468
Price > 0.495 : 0.12868174038938857
Price is missing : 6.676036808337478E-16
```

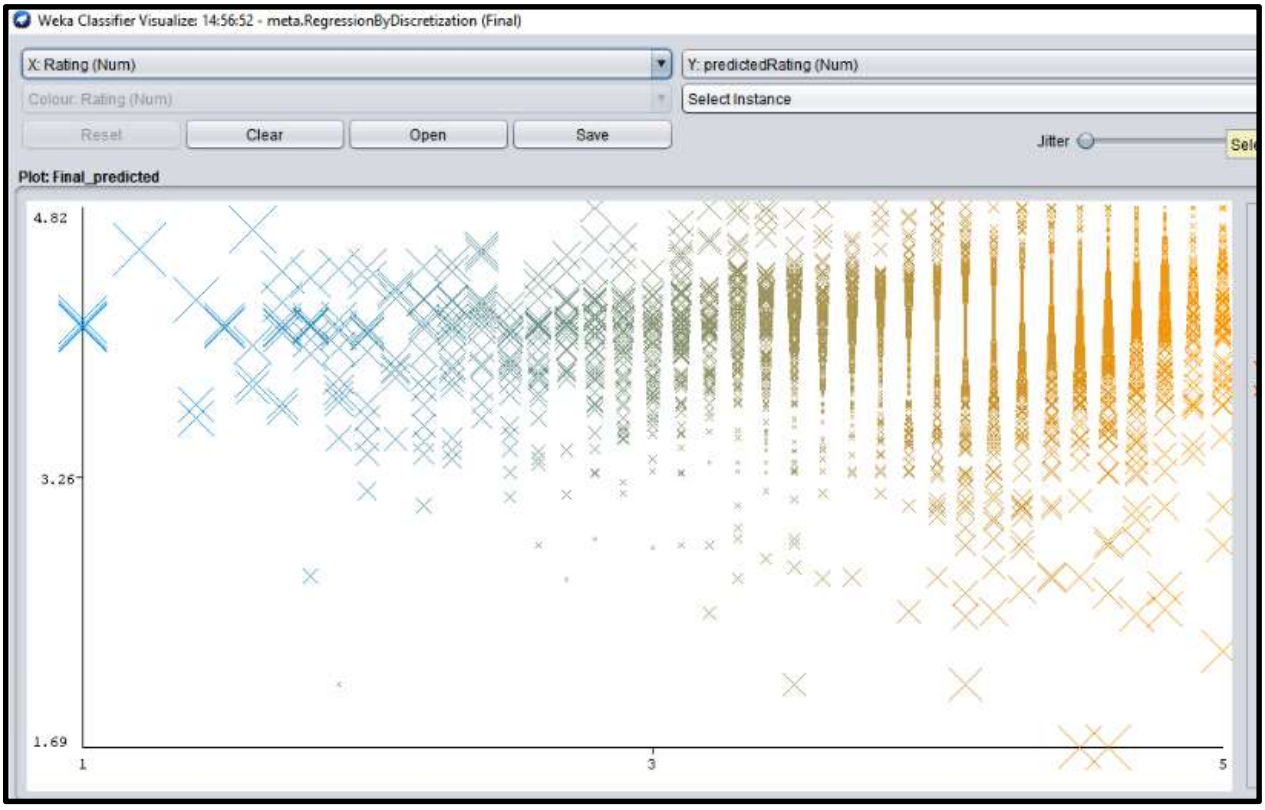
PLOTTING RATING Vs PREDICTED RATING



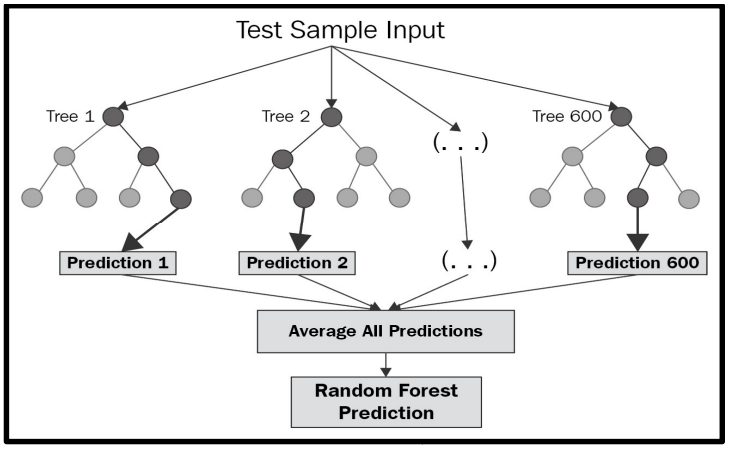
REGRESSION BY DISCRETIZATION

A regression scheme that employs any classifier on a copy of the data that has the class attribute (equal- width) discretized. The predicted value is the expected value of the mean class value for each discretized interval (based on the predicted probabilities for each interval).

PLOTTING RATING Vs PREDICTED RATING



RANDOM FOREST MODEL



Random forest is a Supervised Learning algorithm which uses ensemble learning method for classification and regression.

Random forest is a bagging technique and not a boosting technique. The trees in random forests run in parallel. There is no interaction between these trees while building the trees.

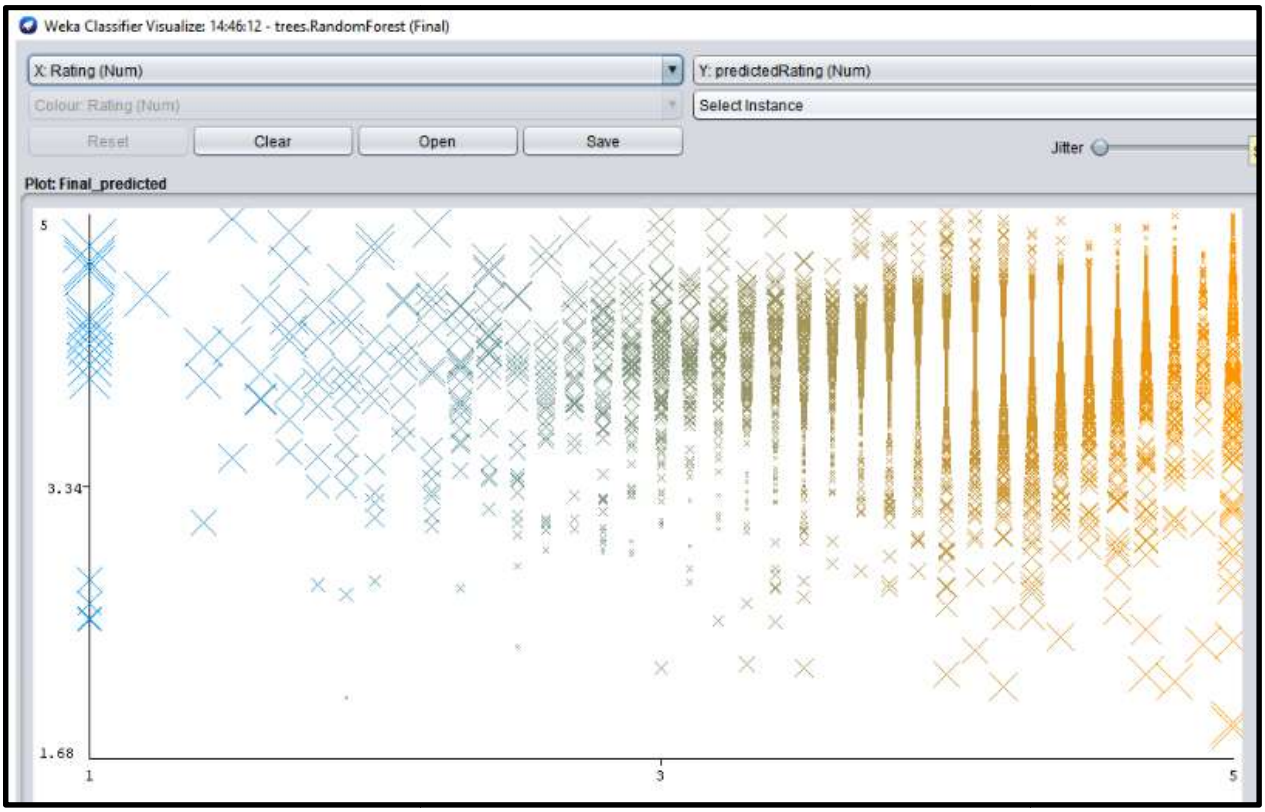
It operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

A random forest is a meta-estimator (i.e. it combines the result of multiple predictions) which aggregates many decision trees, with some helpful modifications:

1. The number of features that can be split on at each node is limited to some percentage of the total (which is known as the hyper parameter). This ensures that the ensemble model does not rely too heavily on any individual feature, and makes fair use of all potentially predictive features.

2. Each tree draws a random sample from the original data set when generating its splits, adding a further element of randomness that prevents overfitting.

PLOTTING RATING Vs PREDICTED RATING



DATA ANALYSIS OF GOOGLE APPS RATINGS USING PYTHON

IMPORTING AND REVIEWING DATA

IMPORTING THE REQUIRED LIBRARIES

```
In [79]: import pandas as pd
         import numpy as np
         import seaborn as sns
         import matplotlib.pyplot as plt
         %matplotlib inline
```

READING DATA

```
In [80]: google_data = pd.read_csv('googleplaystore.csv')
```

INSPECTING THE DATA

```
In [81]: google_data.head()        #Inspecting the first 5 rows
```

Out[81]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up

INSPECTING SHAPE OF THE DATA

```
In [82]: google_data.shape
Out[82]: (10841, 13)
```

STATISTICS SUMMARY

```
In [83]: google_data.describe()       # Summary Statistics
```

Out[83]:

	Rating
count	9367.000000
mean	4.193338
std	0.537431
min	1.000000
25%	4.000000
50%	4.300000
75%	4.500000
max	19.000000

DATA INFORMATION

```
In [86]: google_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10841 entries, 0 to 10840
Data columns (total 13 columns):
App                10841 non-null object
Category           10841 non-null object
Rating             9367 non-null float64
Reviews            10841 non-null object
Size               10841 non-null object
Installs           10841 non-null object
Type               10840 non-null object
Price              10841 non-null object
Content Rating     10840 non-null object
Genres             10841 non-null object
Last Updated       10841 non-null object
Current Ver        10833 non-null object
Android Ver        10838 non-null object
dtypes: float64(1), object(12)
memory usage: 1.1+ MB
```

DATA CLEANING

COUNTING MISSING VALUES

```
In [87]: google_data.isnull()
```

Out[87]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False	False	False	False	False	False	False
7	False	False	False	False	False	False	False	False	False	False	False	False	False
8	False	False	False	False	False	False	False	False	False	False	False	False	False
9	False	False	False	False	False	False	False	False	False	False	False	False	False
10	False	False	False	False	False	False	False	False	False	False	False	False	False
11	False	False	False	False	False	False	False	False	False	False	False	False	False
12	False	False	False	False	False	False	False	False	False	False	False	False	False
13	False	False	False	False	False	False	False	False	False	False	False	False	False
14	False	False	False	False	False	False	False	False	False	False	False	False	False
15	False	False	False	False	False	False	False	False	False	False	False	True	False

10828	False	False	False	False	False	False	False	False	False	False	False	False	False
10829	False	False	False	False	False	False	False	False	False	False	False	False	False
10830	False	False	False	False	False	False	False	False	False	False	False	False	False
10831	False	False	True	False	False	False	False	False	False	False	False	False	False
10832	False	False	False	False	False	False	False	False	False	False	False	False	False
10833	False	False	False	False	False	False	False	False	False	False	False	False	False
10834	False	False	False	False	False	False	False	False	False	False	False	False	False
10835	False	False	True	False	False	False	False	False	False	False	False	False	False
10836	False	False	False	False	False	False	False	False	False	False	False	False	False
10837	False	False	False	False	False	False	False	False	False	False	False	False	False
10838	False	False	True	False	False	False	False	False	False	False	False	False	False
10839	False	False	False	False	False	False	False	False	False	False	False	False	False
10840	False	False	False	False	False	False	False	False	False	False	False	False	False

10841 rows × 13 columns

```
In [88]: # Count the number of missing values in each column
         google_data.isnull().sum()

Out[88]: App                  0
         Category             0
         Rating            1474
         Reviews              0
         Size                 0
         Installs             0
         Type                 1
         Price                0
         Content Rating       1
         Genres               0
         Last Updated         0
         Current Ver          8
         Android Ver          3
         dtype: int64
```

OUTLIERS

```
In [89]: google_data[google_data.Rating > 5]
```

Out[89]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
10472	Life Made WI-Fi Touchscreen Photo Frame	1.9	19.0	3.0M	1,000+	Free	0	Everyone	NaN	February 11, 2018	1.0.19	4.0 and up	NaN

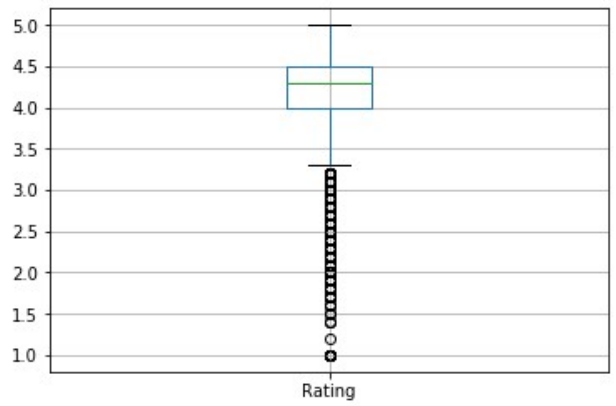
```
In [90]: google_data.drop([10472],inplace=True)

In [91]: google_data[10470:10475]
```

Out[91]:

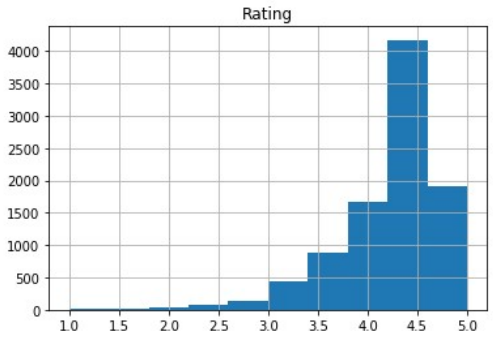
	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
10470	Jazz Wi-Fi	COMMUNICATION	3.4	49	4.0M	10,000+	Free	0	Everyone	Communication	February 10, 2017	0.1	2.3 and up
10471	Xposed Wi-Fi-Pwd	PERSONALIZATION	3.5	1042	404k	100,000+	Free	0	Everyone	Personalization	August 5, 2014	3.0.0	4.0.3 and up
10473	osmino Wi-Fi: free WiFi	TOOLS	4.2	134203	4.1M	10,000,000+	Free	0	Everyone	Tools	August 7, 2018	6.06.14	4.4 and up
10474	Sat-Fi Voice	COMMUNICATION	3.4	37	14M	1,000+	Free	0	Everyone	Communication	November 21, 2014	2.2.1.5	2.2 and up
10475	Wi-Fi Visualizer	TOOLS	3.9	132	2.6M	50,000+	Free	0	Everyone	Tools	May 17, 2017	0.0.9	2.3 and up

AFTER DROPPING DATA

```
In [92]: google_data.boxplot()

Out[92]: <matplotlib.axes._subplots.AxesSubplot at 0x21b4c1ccf60>
```



```
In [93]: google_data.hist()

Out[93]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x0000021B4C2378D0>]],
               dtype=object)
```



DROPPING EMPTY TUPLES

```
In [56]: threshold = len(google_data)* 0.1
         threshold

Out[56]: 1084.0
```

```
In [57]: google_data.dropna(thresh=threshold, axis=1, inplace=True)

In [58]: print(google_data.isnull().sum())

App                0
Category           0
Rating          1474
Reviews            0
Size               0
Installs           0
Type               1
Price              0
Content Rating     0
Genres             0
Last Updated       0
Current Ver        8
Android Ver        2
dtype: int64
```

DATA IMPUTATION AND MANIPULATION

Fill the null values with appropriate values using aggregate functions such as mean, median or mode.

```
In [59]: #Define a function impute_median
         def impute_median(series):
             return series.fillna(series.median())

In [60]: google_data.Rating = google_data['Rating'].transform(impute_median)

In [61]: #count the number of null values in each column
         google_data.isnull().sum()

Out[61]: App              0
         Category         0
         Rating           0
         Reviews          0
         Size             0
         Installs         0
         Type             1
         Price            0
         Content Rating   0
         Genres           0
         Last Updated     0
         Current Ver      8
         Android Ver      2
         dtype: int64
```

HANDLING MISSING VALUES

```
In [62]: # modes of categorical values
         print(google_data['Type'].mode())
         print(google_data['Current Ver'].mode())
         print(google_data['Android Ver'].mode())

         0    Free
         dtype: object
         0    Varies with device
         dtype: object
         0    4.1 and up
         dtype: object

In [63]: # Fill the missing categorical values with mode
         google_data['Type'].fillna(str(google_data['Type'].mode().values[0]), inplace=True)
         google_data['Current Ver'].fillna(str(google_data['Current Ver'].mode().values[0]), inplace=True)
         google_data['Android Ver'].fillna(str(google_data['Android Ver'].mode().values[0]), inplace=True)
```

```
In [64]: #count the number of null values in each column
         google_data.isnull().sum()

Out[64]: App              0
         Category         0
         Rating           0
         Reviews          0
         Size             0
         Installs         0
         Type             0
         Price            0
         Content Rating   0
         Genres           0
         Last Updated     0
         Current Ver      0
         Android Ver      0
         dtype: int64
```

HANDLING NUMERIC VALUES

```
In [65]: ### Let's convert Price, Reviews and Ratings into Numerical Values
         google_data['Price'] = google_data['Price'].apply(lambda x: str(x).replace('$', '') if '$' in str(x) else str(x))
         google_data['Price'] = google_data['Price'].apply(lambda x: float(x))
         google_data['Reviews'] = pd.to_numeric(google_data['Reviews'], errors='coerce')

In [66]: google_data['Installs'] = google_data['Installs'].apply(lambda x: str(x).replace('+', '') if '+' in str(x) else str(x))
         google_data['Installs'] = google_data['Installs'].apply(lambda x: str(x).replace(',', '') if ',' in str(x) else str(x))
         google_data['Installs'] = google_data['Installs'].apply(lambda x: float(x))
```

```
In [67]: google_data.head(7)
```

Out[67]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10000.0	Free	0.0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500000.0	Free	0.0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND_DESIGN	4.7	87510	8.7M	5000000.0	Free	0.0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50000000.0	Free	0.0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100000.0	Free	0.0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up
5	Paper flowers instructions	ART_AND_DESIGN	4.4	167	5.6M	50000.0	Free	0.0	Everyone	Art & Design	March 26, 2017	1	2.3 and up
6	Smoke Effect Photo Maker - Smoke Editor	ART_AND_DESIGN	3.8	178	19M	50000.0	Free	0.0	Everyone	Art & Design	April 26, 2018	1.1	4.0.3 and up
7	Infinite Painter	ART_AND_DESIGN	4.1	36815	29M	1000000.0	Free	0.0	Everyone	Art & Design	June 14, 2018	6.1.61.1	4.2 and up

ANALYSING NUMERIC ATTRIBUTES

```
In [68]: google_data.describe()
```

Out[68]:

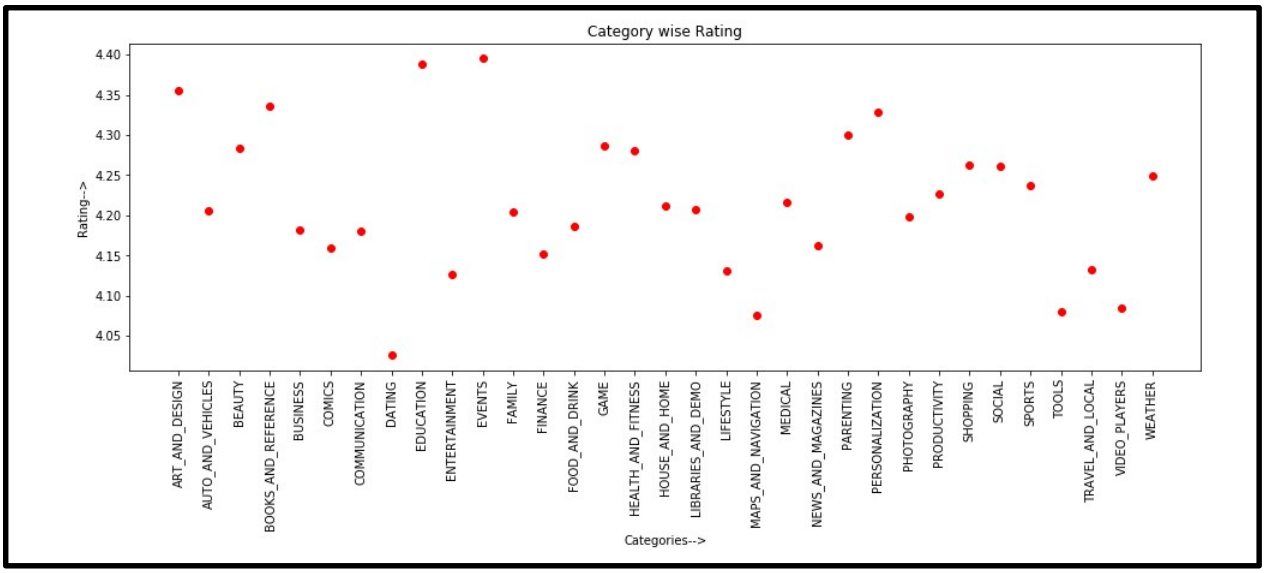
	Rating	Reviews	Installs	Price
count	10840.000000	1.084000e+04	1.084000e+04	10840.000000
mean	4.206476	4.441529e+05	1.546434e+07	1.027368
std	0.480342	2.927761e+06	8.502936e+07	15.949703
min	1.000000	0.000000e+00	0.000000e+00	0.000000
25%	4.100000	3.800000e+01	1.000000e+03	0.000000
50%	4.300000	2.094000e+03	1.000000e+05	0.000000
75%	4.500000	5.477550e+04	5.000000e+06	0.000000
max	5.000000	7.815831e+07	1.000000e+09	400.000000

DATA VISUALIZATION

```
In [69]: grp = google_data.groupby('Category')
         x = grp['Rating'].agg(np.mean)
         y = grp['Price'].agg(np.sum)
         z = grp['Reviews'].agg(np.mean)
```

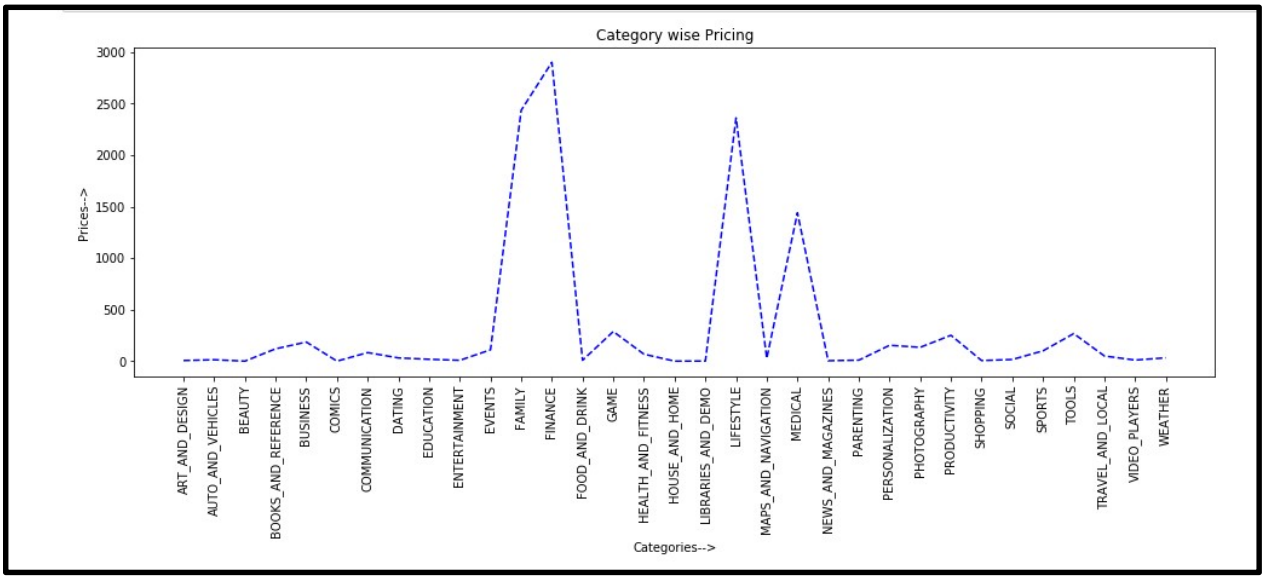
CATEGORY Vs RATING

```
In [71]: plt.figure(figsize=(16,5))
         plt.plot(x,'ro', color='r')
         plt.xticks(rotation=90)
         plt.title('Category wise Rating')
         plt.xlabel('Categories-->')
         plt.ylabel('Rating-->')
         plt.show()
```



CATEGORY Vs PRICE

```
In [72]: plt.figure(figsize=(16,5))
         plt.plot(y,'r--', color='b')
         plt.xticks(rotation=90)
         plt.title('Category wise Pricing')
         plt.xlabel('Categories-->')
         plt.ylabel('Prices-->')
         plt.show()
```



CATEGORY Vs REVIEWS

```
In [73]: plt.figure(figsize=(16,5))
         plt.plot(z,'bs', color='g')
         plt.xticks(rotation=90)
         plt.title('Category wise Reviews')
         plt.xlabel('Categories-->')
         plt.ylabel('Reviews-->')
         plt.show()
```