Android Application Success Prediction and Review Analysis

Rituraj Saha [1] , Ritam Barik [1] , Pranav Jain [1] , Bhaswati Sahoo [2]
School of Computer Engineering
Kalinga Institute of Industrial Technology, Deemed to be University, Bhubaneswar - 24, Orissa, India
saharituraj@gmail.com, er.ritambarik@gmail.com, pranavjain594@gmail.com, bhaswati.sahoofcs@kiit.ac.in

ABSTRACT. Machine Learning plays an important role in computer technology and artificial intelligence. Using machine learning can reduce human effort in prediction, learning, and recognition. In this paper, data analysis was accomplished on a smaller scale to dive deeper into the Google play store data records that become accrued, discovering relationships with precise functions including categorization of app such as effect on installations , good way of using them to discover apps that are more likely to be successful. These extracted features were considered along with the current sentiment of users. The accomplishment of an app is anticipated soon after it's miles launched into the Google play store. In this paper the performance of several regressors and clustering algorithms are analyzed and compared on a standard dataset[9] depicting various attributes of Google play store apps and also evaluating the performance of a classifier algorithm on Google play store app review dataset[9].

Keywords: App review, Bag of Words Model, Clustering Models, Hierarchical Clustering, K-Means Clustering, Random Forest Regressor, Logistic Regression, Multiple linear Regressor, Natural Language Processing, Nearest Neighbor Regressor, Rating prediction, Regression Models, Support Vector Regressor.

1. INTRODUCTION

Mobile applications have significant growth in this era and thus has a great impact on digital technology. Having said that, with the ever-growing mobile app marketplace there's additionally a first-rate upward thrust of mobile app developers, ultimately ensuing in sky-high revenue with the aid of the worldwide mobile app industry, with sizeable opposition from everywhere in the globe, it's miles imperative for a developer to know that he's proceeding in the correct direction.

The research companies adeven, distimo and localytics found more than 60% of applications in the app store the at were not downloaded in 2012 and 80% of paying and android apps got less than 100 downloads in 2011, and nearly 80% of all app users across all industries dropped an app within 90 days in 2017, with 25% of them using it. App creators, therefore consequently should understand their customers' needs to make sure their applications get a better response as soon as deployed into an app store. App store offers a brand new channel of communication between android developer and their users. Moreover, evidence has also shown that ratings and installs are often highly linked and that android app rating is one of the main determinants in consumer app download decision.

A Machine Learning approach was used as it eliminates most of the guesswork and helps to focus on other tasks like leverage past and continuous data, create predictivemodels for maximizing asset lifetime, operational efficiency and optimize the periodic maintenance operations.

This dataset[9] used consist of more than 10k apps each having 13 attributes. For the EDA part, the dataset[9] was analyzed and several charts were created to visualize the relationship between different attributes in order to conclude the correlation between them. Additionally, an analysis on the features App Name, Content Rating and Type was done to reach some valid conclusion.

This paper tries to find ways to help developers reach the right people with the right apps by answering the problem statement stated in the previous section. On the Google Play Store data set, data analysis was performed on various apps. It has been found that certain category apps have considerably more installs than others. Moreover, text reviews/comments on different user apps were studied and discovered how most people express happiness and frustration in writing. The data analysis features and the basic attributes of apps were used to train five different machine learning algorithms such as KNN, Random Forest, Linear Regressor, SVM, Logistic Regressor Classification to predict the previously formulated success. Hence, from the analysis, it can be seen that determining the success rate of an app will play a very important role for developers and can bring certain changes that might affect the lifetime of their app. Our success model will help developers predict whether their app will be successful in the long run given the current state of the apps that have slow growth rates or are new to the market.

Android App Industry is growing significantly and this increases competition among those who build applications in the developing field. This is a kind of ultimatum or warning to each individual developer as a larger number of competitions would increase the likelihood of weakening their app's store value. Considering that most Play Store applications are free, the revenue model is quite unclear and uncertain as to how in- app advertising, subscriptions and in-app purchase contribute to an app's success. The standard of an app is therefore usually determined by the number of installs and user ratings it has received over its lifetime rather than the revenue it generates.On the other hand we have to consider that even if an app is released and has achieved a good rating along with a strong number of downloads, holding their place in the Play store forever is not enough because there will be many other applications that will provide the end users with even better functionality.Because of the growth and competitiveness between the industry, we chose to carry out research related to this field, so we did a thorough analysis of our Google Play Store data and generated our own established view of success, which we believe will be a major contribution to the developers. As a result, they will be able to know their success rate and determine which function needs to be maintained or updated in depending upon the current state of their application. Our original motive for this study was to find something significant throughout the exploration that could make a difference to the developers or end users.

2. LITERATURE REVIEW

Tuckerman [8], has scraped data from the Google play store to extract large number of features using a web crawler 'scrapy' and used the data to train three models to predict the success of an application. To predict success metrics of an app, revenue should be the key feature but as it is not found publicly, the author used the number of installations and average user rating, trained Linear model to categorize whether an app will be successful or not and additionally used linear regression to predict the average rating of a system. The principal component analysis was performed in order to focus on variation and create strong patterns of the dataset by using inputs of GLM and Linear regression models. Using these models, the author concluded that about 35% of the total successful applications has the word 'photo' in the description and about 31% has the word 'share', using these models, the developer can be referred to the genres of application which are mostly liked by end users. Furthermore, in Luiz et al [6], the authors observed that how store-ratings after reaching a certain value does not affect the overall store rating even after users rate it, they also noticed that when an app is updated their rating varies version wise, but it is not observable in the store-rating. Therefore, they came up with the idea of version rating which they could calculate based on the calculation of store rating which was available in App Store. This idea was proposed in order to help developers gain incentive to develop apps even after the store rating reaches a threshold value and they recommended that every App store owner should display the current version rating. Their approach of calculating version rating to make developers work for the updates is very efficient.

Aralikatte et al[1], tried to represent the review-rating mismatch, through establishing multiple systems that can automatically detect the inconsistency between these two, to prove this mismatch they implemented two machine learning approaches, it included various classifiers like Decision table, Decision tree, Naive Bayes Classifier, Decision stump and few other algorithms, and the other approach focused on deep learning techniques. They also hosted several surveys in order to learn the opinion of users and developers regarding this mismatch. The outcome of the survey was quite expected, both the Developer and end users of Android app agreed that rating of an app should match with its corresponding review and they also asserted to have an automated system to detect the mismatch between rating and review if there is any.

Islam et al[4], states that the numeric rating as in the stars given by users have a huge difference than compared to the reviews given by them thus a rating system has been proposed by the author which will remove the ambiguity created by the mismatch of the rating and respective review by the same user. It has been seen that how the users are dependent on others' opinion while taking any decision. He proposed a system with a purpose to begin with behavior sentiment analysis on the client reviews and will generate a numeric rating from the polarity. This paper shows a strong relationship between the app rating and the reviews of the users, from where the idea of picking up the idea of using the reviews of the app in the work and a thorough research on the reviews that are explained in later sections of this paper was done.

Fu et al [2], helped to get the idea of removing the inconsistent reviews that will basically reduce noise from the dataset and give better performance in sentiment analysis, and therefore will generate polarity value more accurately. Fu, Lin, and Li proposed WisCom, a system that is able to analyze at least ten million of user ratings and comments in the app markets in three different levels. The first feature of their system is to identify the Inconsistency in reviews; secondly, they looked for the reasons why people do not like a particular app and how the reviews change over time. They extended their analysis to provide a valuable insight into the complete app market providing users with major concerns. More specifically it analysis review on three levels that are firstly on the single review (micro-level) done using Linear regression model, secondly on review of every app (Meso level) by doing LDA as well as they performed topic analysis on different time segments to judge how review changes over time.

Pang et al [7], worked on movie reviews; they converted their extracted rating into one of the three categories: neutral, positive and negative. From the words they initially found accuracy for the overall negative and positive sentiment and later made a more compressed file of words using seven words for positive and negative respectively. They used three algorithms: SVM, Naive Bayes, and Maximum Entropy, where SVM performed the best and Naive Bayes the worst. Their work is quite different as that of this paper, as movie review are generally different from app review Fu et al[2].

Jong et al [5] worked with yelp dataset consisting of one hundred fifty thousand reviews and their corresponding ratings for restaurants. According to his findings Grover[3], he stated text review holds the more quantitative value that a star rating thus combining the star rating with a list of restaurant text review will provide a rich quantitative estimation of the service satisfaction rating. The average rating of his dataset was 3.6, so he placed all ratings above this to positive sentiment and below this to negative sentiment.

3. METHODOLOGY

3.1 Overview

This paper proposes feature-oriented rating prediction, analyzes the relations between different attributes of the app using clustering and predicts the sentiment of user reviews.
Basic work-flow of this project-

1. Collect dataset
2. Prepare the dataset
3. Choose required algorithm
4. Fit a model
5. Compare different models
6. Choose an optimum model
7. Analyze the fit and upgrade until satisfied
8. Use fitted model for predictions

3.2 Data Collection

The dataset[9] that has been used was downloaded from Kaggle which had about 10,000 apps along with another dataset[9] having 64,000 reviews.

3.3 Data Description

The attributes of play store app dataset[9] and app review dataset[9] are briefly explained in the following Table 1 and Table 2.

Table 1: Description of the App review dataset

Attribute	Description
App	Name of the app
Review	Comment text entered by a user
Sentiment	Polarity of the review

Table 2: Description of the App dataset attributes

Attribute	Description
App	Name of the app
Category	Category of the app
Rating	Average rating of the app
Size	Size of app in KB,MB
Installs	No. of Installs of the app
Type	Free or Paid
Price	If Paid
Content Rating	Content for audience
Genre	Genre of the app
Last Updated	Date of the last app update
Current Version	Latest app version
Android Version	Supported android versions

3.4 Data Processing

Since there were some null values present in the given dataset[9], it was replaced by the mean of their respective attribute. Also, the raw data extracted needed to be pre-processed to turn it into some valuable information. To be able to perform EDA and run algorithms on the dataset[9], installs, total number of ratings, rating distribution were converted to integers. Sizes of app having megabytes were converted to kilobytes to have consistent units. Apps with varying sizes (those that vary with device) were set to the average size of the rest of the apps. Later the categorical attributes like categories, genre, content rating and type were label encoded, for instance free apps were labeled as 1 and paid apps were labeled as 0.

Further, to find out the sentiment of the users, the review text was processed using Bag of Words model. Sentiment of each of the reviews were label encoded from Positive, Neutral and Negative to 0,1 and 2 respectively.

3.5 Data Analysis

Here the analysis the dataset[9] was done to summarize the main features with visual representation to see what the data can tell us beyond the formal modeling.

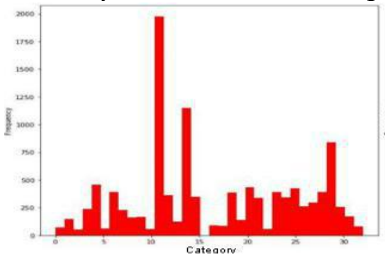


Fig 1: Frequency vs Category

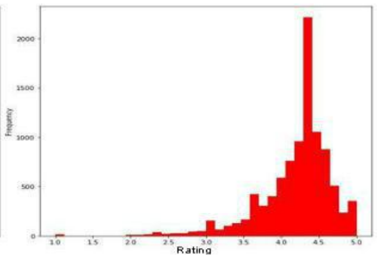


Fig 2: Frequency vs Rating

Category vs Frequency and Rating vs Frequency. The Category column in the dataset[9] has 30 different types of categories. The left figure shows the histogram of the number of categories and their respective frequency. And the right figure shows the histogram of the app rating which ranges from 0 to 5 along with their respective number of downloads. It clearly shows that the apps having the highest rating are mostly downloaded.

3.6 Algorithms

Regression Models

K Nearest Neighbor

KNN is a simple regression model that contains all the possible cases and predicts the new value based on various similarity measures such as distance functions like Euclidean distance, Manhattan distance and Minkowski distance.

$$\text{Euclidean: } \sqrt{\sum_{i=l}^{k}(x_i - y_i)^2} \quad \text{Manhattan: } \sum_{i=l}^{k}|x_i - y_i| \quad \text{Minkowski: } \left(\sum_{i=l}^{k}(|x_i - y_i|)^q\right)^{1/q}$$

Equation 1: Distance Functions

The three distance measurements mentioned above are valid only for continuous values. A big K value is generally more accurate because it decreases the overall noise. For most datasets, the best possible K value is 10 or more.

Random Forest

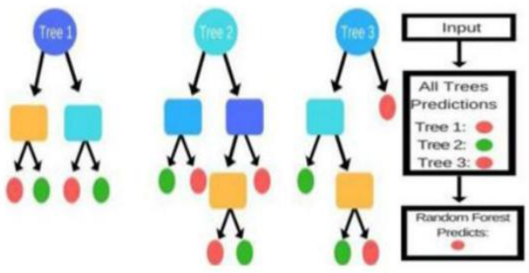


Fig 3: Random Forest

Random Forest Model is an ensemble method that can perform both regression as well as classification functions. The fundamental concept is to incorporate various decision trees instead of focusing on any individual decision tree to determine the final outcome.

Steps:

1. Select K variables randomly from the training data.
2. Construct the decision tree linked to those K points.
3. Choose the N no. Of trees to be constructed and repeat the above steps.
4. Make each of your N trees determine the value of Y for a new variable and allocate this new variable the average over all the Y values predicted.

Multiple Linear Regression (MLR)

MLR is a technique for modelling the linear connection between a dependent (target) variable and one or more interdependent (predictors) variables.

$$\text{Observed data: } y = b_0 + b_1x_1 + b_2x_2 + \ldots + b_px_p + \varepsilon$$
$$\text{Predicted data: } y' = b_0 + b_1x_1 + b_2x_2 + \ldots + b_px_p$$
$$\text{Error: } \varepsilon = y - y'$$

Equation 2: MLR equations

MLR works on ordinary least squares (OLS), in such a way that the model is fitted to minimize the sum of squares of measured and predicted values.

Support Vector Regression (SVR)

Support Vector Machine (SVM) is used as a technique of regression, preserving all the primary characteristics of the algorithm (maximal margin). With only a few slight differences, SVR uses the same classification principles as the SVM.

$$y = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*).\langle x_i, x\rangle + b$$

Equation 3: Linear SVR

$$y = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*).K\langle x_i, x\rangle + b$$

Equation 4: Non-Linear SVR

Clustering Models

K- Means Clustering

K-Means algorithm aims to separate n items into k no. of clusters such that each item is the member of the cluster having the nearest mean. This technique generates precisely k distinct clusters of maximum possible difference

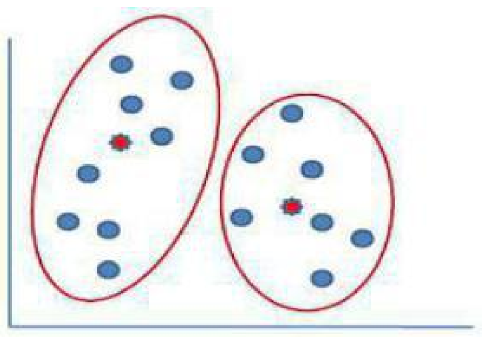


Fig 4: Clusters

Algorithm:

1. Clusters the data into *k* groups where *k* is predefined.
2. Choose k cluster centers randomly.
3. Assign objects to their nearest cluster centroid using the Euclidean distance function.
4. Find the mean of objects in each of the clusters. Then go to step 2 until the similar points are assigned in consecutive rounds to each cluster.

Hierarchical Clustering

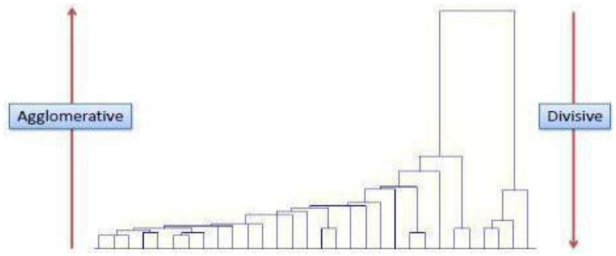


Fig 5: Hierarchical Clustering (Dendogram)

Hierarchical clustering includes the development of clusters from top to bottom with a specified order. Hierarchical clustering has two forms, Divisive and Agglomerative.Every observation was assigned to its own cluster in the agglomerate technique.
Then consider the similarity (for e.g., distance) between each cluster and be a member of the 2 clusters that are most similar.

Natural Language Processing

Natural Language Processing (NLP) is a computer program's capability to recognize human speech as it is expressed. NLP is an AI component.

Bag of words model
It operates on numbers whenever an algorithm is implemented in NLP. Bag of Words model is therefore used to preprocess the text by translating it into a bag of words that contains a list of the total occurrences of the most commonly used words.

Classification Model
Logistic Regression
Logistic regression estimates the probability of an outcome which can have values 0 or 1. The estimation is based on the use of one or more predictors (numerical or categorical). A logistic regression classification model generates a logistic curve that is limited to zero to one value.

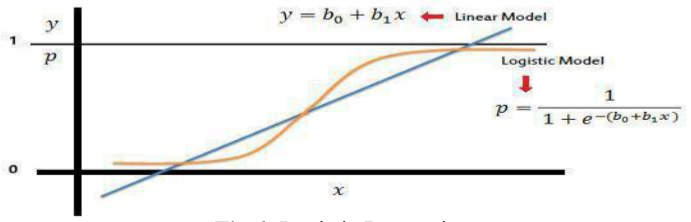


Fig 6: Logistic Regression

In logistic regression classification model, the constant (*b₀*) moves the curve left & right and also the constant (*b₁*) defines the slope of the curve.

$$\frac{p}{1-p} = \exp\,(b_0 + b_1 x)$$

Equation 5: LR equation

4. COMPARISONS AND RESULTS

The result shows the mean squared error MSE, mean squared log error MSL and R2 score of many regression models used for calculating the ratings of 10,000+ apps. Moreover, it also shows the confusion matrix and accuracy score on classification models in- order to classify the sentiments of users from their reviews given in the app review dataset[9].According to our data set for rating prediction ,K-Nearest Neighbour gave good competition to the other classification algorithms Random forest and support vector machine.This algorithm is quite robust in the sense that if a new data point is added in the data set, the overall algorithm will not be significantly affected because new data must affect one category in which the most neighbor occurs, but it is very difficult for it to affect all categories.Logistic Regressor classification gives better result in classifying the feeling of the users as the data set is linearly separable also this algorithm is easier to implement and very efficient to train.

From this project, a minimum MSE of 16.91% in predicting the rating of the apps by using KNN regressor model and a maximum accuracy score of 91.2% in classifying the sentiment of the users by using Logistic Regressor classification model was achieved. We have also used various clustering models in-order to analyze the relationship between ‘category’ vs ‘no. of installs’ and also for ‘rating’ vs ‘no. of ‘installs’.

The various regression models that have been used in- order to predict the rating of the apps depending on their features are shown below-

Table 3: Comparison between various models

Regression Model	**MSE**	**R2 Score**	**MSL**
KNN	0.1691	0.1126	0.0073
Random Forest	0.1760	0.0768	0.0076
MLR	0.1900	-0.0016	0.0082
SVM	0.1875	0.016	0.0082

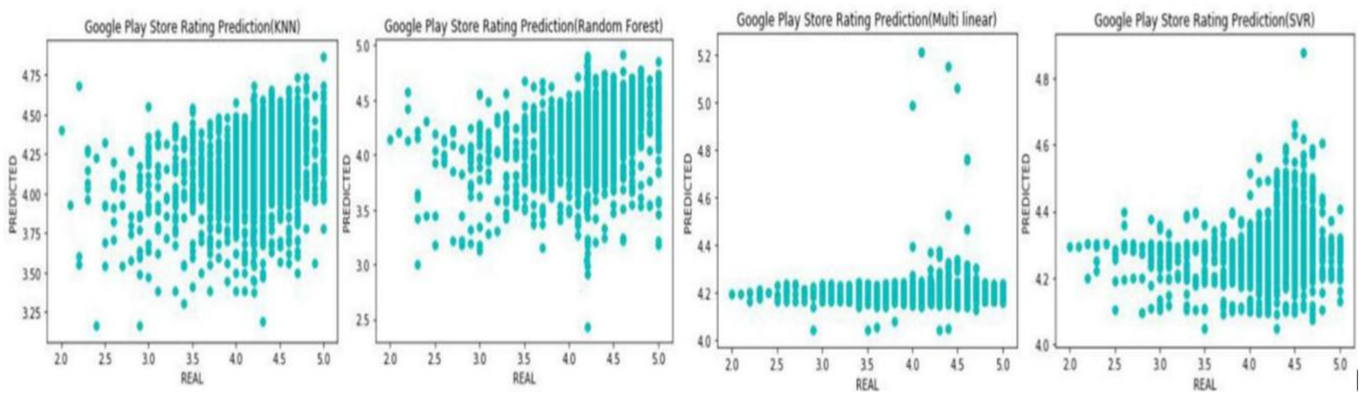


Fig 7: Visual representation of various regression models

K Means Clustering
Two clustering models that has been used in- order to study the relations between various attributes of an app are shown below-

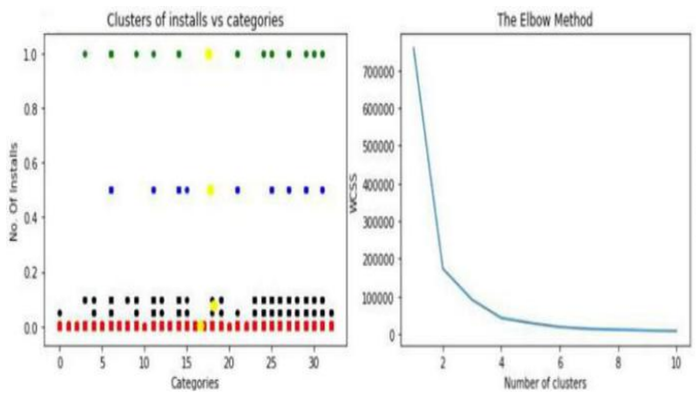


Fig 8: No. of Installs vs Category

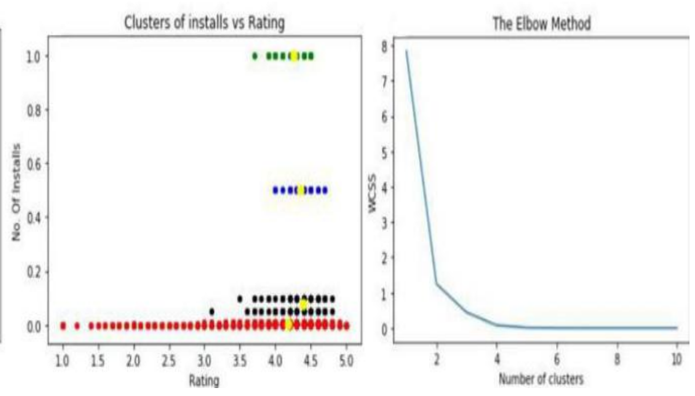


Fig 9: No. of Installs vs Rating

Hierarchical Clustering

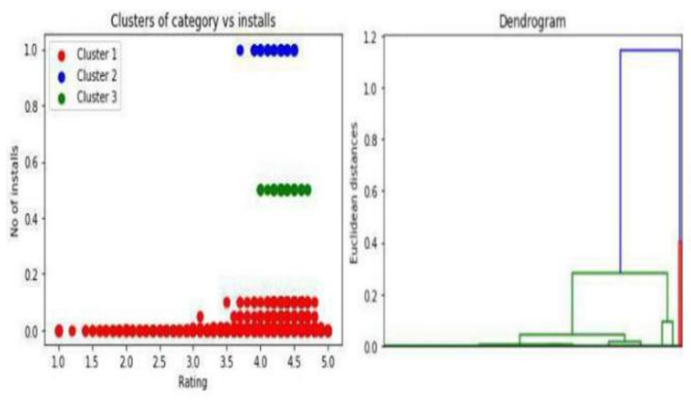


Fig 10: No. of Installs vs Rating

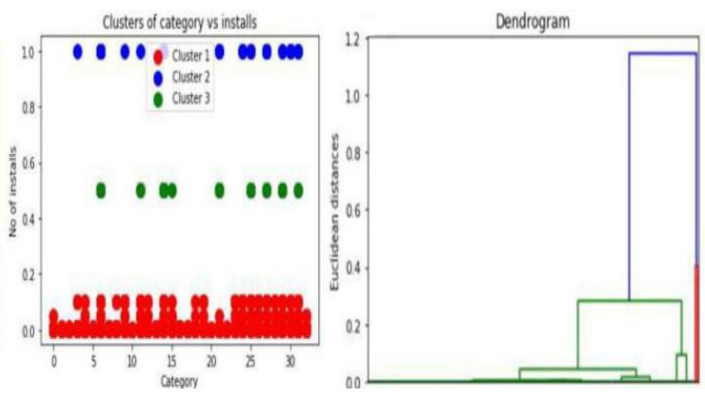
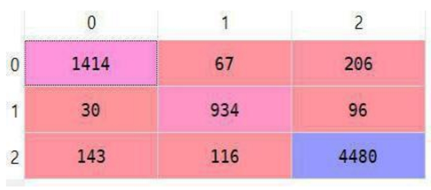


Fig 11: No. of Installs vs Category

At first, simple logistic regression model was fit with the training data and then the confusion matrix was made to compare the actual outcomes to the predicted outcomes.

Table 5: Classification

Accuracy Score	R2 Score	Precision	Recall	F score
0.91210	0.67095	0.91225	0.91210	0.91190



	0	1	2
0	1414	67	206
1	30	934	96
2	143	116	4480

Table 4: Confusion Matrix

Table 4 shows the confusion matrix of the logistic regression model with class labels. Here, the model predicted success correctly 6828 out of 7486 times. From the classification report in Table 5, these values are reflected in the precision column. The model has predicted success with a precision of 0.91.

5. CONCLUSION

In this paper, an approach was suggested to predict the ranking of play store apps based mainly on the technical information gathered from the summary of the apps This information is based on the various user ratings in the play store. An NLP algorithm was also used to retrieve the users' feedbacks from the current dataset [9] of app reviews.

REFERENCES

[1] Aralikatte, R., Sridhara, G., Gantayat, N., and Mani, S. (2018). Fault in your stars: an analysis of android app reviews. In Proceedings of the ACM India Joint International Conference on Data Science and Management of Data, pages 57–66. ACM.
[2] Fu, B., Lin, J., Li, L., Faloutsos, C., Hong, J., and Sadeh, N. (2013). Why people hate your app: Making sense of user feedback in a mobile app store. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 1276–1284. ACM.
[3] Grover, S. (2015). 3 apps that failed (and what they teach us about app marketing).
[4] Islam, M. R. (2014). Numeric rating of apps on google play store by sentiment analysis on user reviews.In 2014 International Conference on Electrical Engineering and Information Communication Technology, pages 1–4.
[5] Jong, J. (2011). Predicting rating with sentiment analysis. [online] http://cs229.stanford.edu/proj2011/Jong-PredictingRatin gwithSentimentAnalysis.pdf. References 65
[6] Luiz, W., Viegas, F., Alencar, R., Mourão, F., Salles, T., Carvalho, D., Gonçalves, M. A., and Rocha, L.(2018). A feature-oriented sentiment rating for mobile app reviews. In Proceedings of the 2018 World Wide Web Conference on World Wide Web, pages 1909–1918. International World Wide Web Conferences Steering Committee.
[7] Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10, pages 79–86. Association for Computational Linguistics.
[8] Tuckerman, C. (2014). Predicting mobile application success.
[9] Datasets Link: https://www.kaggle.com/lava18/google-play-store-apps.