**A MINI PROJECT REPORT**  
**ON**  
**Predicting Rating Of Mobile Application**

Submitted to Mumbai University  
In the partial fulfillment of the requirement for the award of the degree of  
**Bachelor of Engineering**

In  
**COMPUTER ENGINEERING**

By

**Mrs. Khan Heena Mohd Rahis**  
**(16CO04)**

Under the guidance of  
**Mr. Abdus Salaam**  
**Assistant Professor**



**Department of Computer Engineering**  
**Anjuman-I-Islam’s Kalsekar Technical Campus**

**Affiliated to Mumbai University**

KHANDA GOAN, NEW PANVEL, NAVI MUMBAI, MAHARASHTRA

2016-2017

**Department of Computer Engineering**  
**Anjuman-I-Islam’s Kalsekar Technical Campus**

**Affiliated to Mumbai University**

KHANDA GOAN, NEW PANVEL, NAVI MUMBAI, MAHARASHTRA

2016-2017



**DECLARATION BY THE CANDIDATE**

**Khan Heena Mohd Rahis** bearing **Roll number: 16CO04**, hereby declare that the mini project report entitled **“Predicting Rating Of Mobile Application”**, is a record of bonafide work carried out by me and the results embodied in this project have not been reproduced or copied from any source. The results of this project report have not been submitted to any other University or Institute for the award of any other Degree or Diploma.

**Khan Heena**

**( 16CO04 )**

**Department of Computer Engineering**  
**Anjuman-I-Islam’s Kalsekar Technical Campus**

**Affiliated to Mumbai University**

KHANDA GOAN, NEW PANVEL, NAVI MUMBAI, MAHARASHTRA

2016-2017



**CERTIFICATE**

This is to certify that the project report entitled **“Predicting Rating Of Mobile Application”**, submitted by **Mrs. Khan Heena** , bearing **Roll. No.: 16CO04** in the partial fulfillment of the requirements for the award of the degree of **Bachelor of Computer Engineering** is a record of bonafide work carried out by him/her for the course **Mini Project CSP605**.

**Mini Project Guide**

**(Prof . Abdus Salaam)**

**Mini Project Coordinator**

**(Prof. Muhammed Salman Shamsi)**

**Program Owner**

**(Prof. Tabrez Khan)**

**INDEX**

**CONTENTS**

CH**APTER 1: INTRODUCTION**

* 1. Introduction……………………………………………00
  2. Scope…………………………………………………..03
  3. Problem Statement……..………...................................03

**CHAPTER 2 SYSTEM SPECIFICATION**

2.1 System Requirement……………………………………26

2.2 System Features……………………………………...…26

**CHAPTER 3: SYSTEM DESIGN**

3.1 System Architecture…………...………………………. 47

3.2 Modules in the System………...………………………. 47

3.3 Use Case Diagram ………...…………………………... 47

3.4 Activity Sequence Diagram………......……………...... 47

**CHAPTER 4: IMPLEMENTATION**

4.1 Code Snippets………..…..……….…….…………........50

4.2 Screen Shots…...………………………………………..70

**CHAPTER 5: CONCLUSION**

5.1 Conclusion….…………………………………………..75

5.1 Future Scope....…………………………………………..75

**REFERENCES**...…………………………………………………………... 76

**CHAPTER 1**

**INTRODUCTION**

**1.1 INTRODUCTION**

Applications are designed for use specifically on tablets, smartphones, and other mobile devices. user generally download them through an app store specific to there mobile operating system, such as Apple’s App Store or Google Play. These apps might be games (think Candy Crush or Flappy birds), office programs (Gmail and Slack are good examples), or even shopping or service portals (the Amazon and Uber apps fit well in this category).

In recent years, there is great success of mobile devices, such as smartphones and tablet computers. The enormous popularity of mobile devices can be partially credited to mobile application. For example, apps can help users to buy products, book restaurants, watch movies, and read news simply with a mobile device connected to the Internet. Recently, the amount of mobile apps has grown in a dramatic rate than we have ever expected. For example, on Google Play (formerly known as the Android Market), there were over 1.43 million apps published and over 50 billion downloads. With such a huge number of apps, a direct challenge for the app

By 2015, most developers say, they'll be coding apps for more than just smartphones and tablets--branching out into televisions (83.5%), game consoles (71.2%), the foldable screen (68.1%), and Google Glass (67.1%).

When user asked to rate their satisfaction with a number of features the standard supports, developers were lukewarm, to say the least. Most gave the rating "neutral to dissatisfied" regarding the HTML5 user experience (62%), performance (72.4%), monetization (83.4%), fragmentation (75.4%), distribution control (60.3%), timeliness of new updates (67.9%), and security (81.8%). The only things that satisfy developers when it comes to HTML5 are cross-development capabilities (83.4%) and immediate updates (81.8%).

**1.2 SCOPE**

Without a sufficient presence on the web, chances of business success diminished significantly. As time has gone on, the extent of that web presence has broadened. Social media profiles, optimized pages, and mobile-responsive websites have all become commonplace and, some would say, necessary to conduct business in the modern age. Add now to that list the mobile business application, or the “app.”

The main takeaway here is that people want to use apps, but they’re sick of downloading them. People want things fast, simple, and easy. For example, in early 2018, Google implemented changes that take into account the loading speed of websites on mobile devices when determining ranking in their results pages. It’s far from the only ranking factor, but it heralds a serious shift toward a mobile-friendly world. Google has signaled their recognition that the mobile experience is extremely important to a vast majority of internet users.

Hybrid monetization models, such as in-app ads and in-app purchases, are quickly gaining popularity in the business world. Most studies show that in-app advertising is set to be a key driver of mobile growth over the coming years

**1.3 PROBLEM STATEMENT**

Though the future of app development is not yet certain, we can look to the past for reference and to predict what’s yet to come. It’s a good bet that in the future the field will be heavily saturated with data points and API connections — and this says nothing of app development.

Naive Bayes classification is used in this project to predict the rating. The rest of the model attempted was to build a model which would classify an application as successful or not based on its description. A naive based classifier text was chosen, and the implementation was done in Python.

**CHAPTER 2**

**SYSTEM SPECIFICATION**

**2.1 SYSTEM REQUIREMENT**

**2.1.1 HARDWARE REQUIREMENTS**

* Intel Core i3 3rd gen processor or later
* 200Mb disk space
* 1 GB RAM

**2.1.2 SOFTWARE REQUIREMENTS**

* Microsoft Windows XP or later / Ubuntu 12.0 LTS or later /MAC OS 10.1 or later
* Python Interpreter (2.7 or later)
* Tkinter module
* tkSimpleDialog, tkMessageBox, functools, pandas, pprint libraries

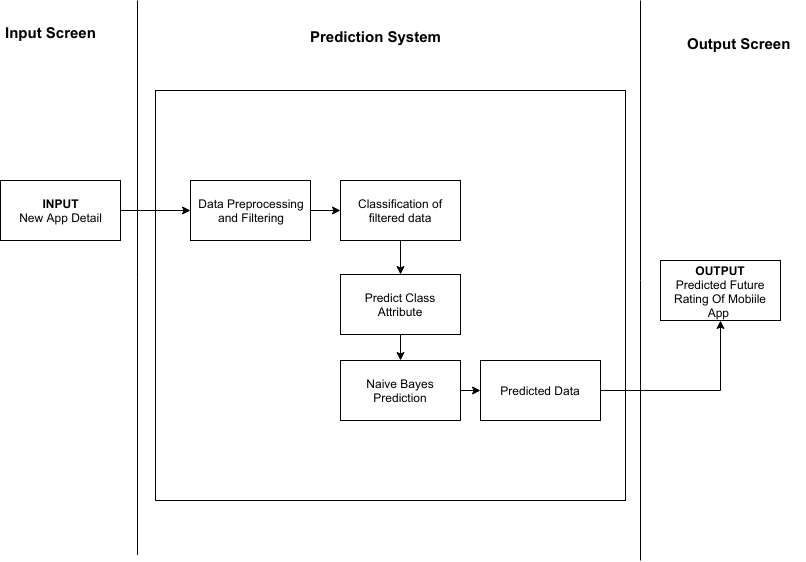
**2.2 SYSTEM FEATURES**

* Dialoug box using tkinter to take input and display the result.
* Prediction on Rating is based on prior observation on dataset of mobile app
* Naive bayes algorithm to predict the rating using probability

**CHAPTER 3**

**SYSTEM DESIGN**

**3.1 SYSTEM ARCHITECTURE**

****

**Fig 1: System Architecture for Predicting Rating of Mobile App.**

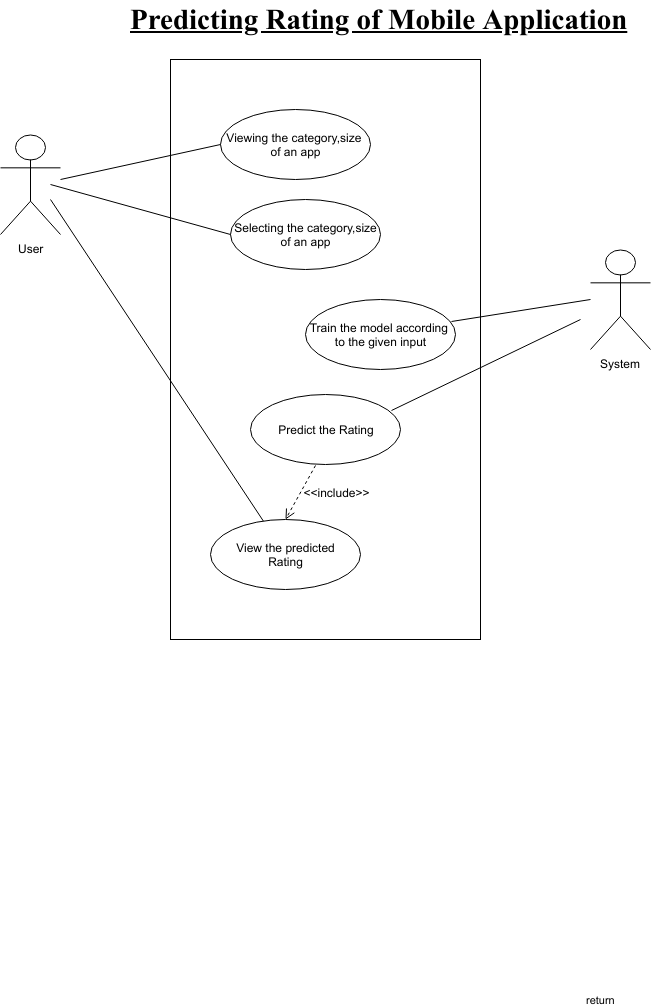
**3.2 MODULES IN THE SYSTEM**

* **Input Screen :-** In this module System will take Mobile Application Detail using Tkinter Dialoug Box such as Category (i.e Beauty, Communication,…. etc ) , Content Rating (i.e Everyone, Teen, …. etc), Type(i.e Free, Paid ), Size (I.e 12M , 24M, … etc)as a Input from the user to predict Rating of that respective app.

R dialoug box

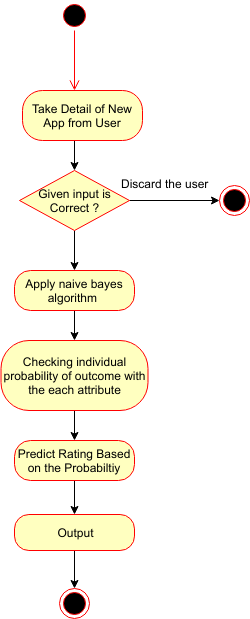
* **Prediction System :-** In this Project, Naive Bayes algorithms is used to predict the rating of mobile app By calculating the Probability of each evidence with each Class attribute.
* **Output Screen :-** In this module system will display the predicted rating as a result, in the output using tkinter dialoug box

**3.3 USE CASE DIAGRAM**

****

**Fig 2: Use Case Diagram for Predicting Rating of Mobile App.**

**3.4 ACTIVITY DIAGRAM**



**Fig 3: Activity Diagram for Predicting Rating of Mobile App.**

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 Code Snippets**

import Tkinter as tkr

import tkSimpleDialog

import tkMessageBox

from functools import reduce

import pandas as pd

import pprint

class Classifier():

data = None

class\_attr = None

priori = {}

cp = {}

hypothesis = None

def \_\_init\_\_(self,filename=None, class\_attr=None ):

self.data = pd.read\_csv(filename, sep=',', header =(0))

self.class\_attr = class\_attr

'''

probability(class) = How many times it appears in cloumn

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

count of all class attribute

'''

def calculate\_priori(self):

class\_values = list(set(self.data[self.class\_attr]))

class\_data = list(self.data[self.class\_attr])

for i in class\_values:

self.priori[i] = class\_data.count(i)/float(len(class\_data))

print ("Priori Values: ", self.priori)

'''

Here we calculate the individual probabilites

P(outcome|evidence) = P(Likelihood of Evidence) x Prior prob of outcome

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

P(Evidence)

'''

def get\_cp(self, attr, attr\_type, class\_value):

data\_attr = list(self.data[attr])

class\_data = list(self.data[self.class\_attr])

total =1

for i in range(0, len(data\_attr)):

if class\_data[i] == class\_value and data\_attr[i] == attr\_type:

total+=1

return total/float(class\_data.count(class\_value))

'''

Here we calculate Likelihood of Evidence and multiple all individual probabilities with priori

(Outcome|Multiple Evidence) = P(Evidence1|Outcome) x P(Evidence2|outcome) x ... x P(EvidenceN|outcome) x P(Outcome)

scaled by P(Multiple Evidence)

'''

def calculate\_conditional\_probabilities(self, hypothesis):

for i in self.priori:

self.cp[i] = {}

for j in hypothesis:

self.cp[i].update({ hypothesis[j]: self.get\_cp(j, hypothesis[j], i)})

print ("\nCalculated Conditional Probabilities: \n")

pprint.pprint(self.cp)

def classify(self):

root =tkr.Tk()

print ("Result: ")

a=[]

for i in self.cp:

result=reduce(lambda x, y: x\*y, self.cp[i].values())\*self.priori[i]

print (i, " ==> ", result)

a.insert(i,result)

l=max(a)

d=a.index(l)

w =tkr.Label(root,text=d+1)

w.pack()

button = tkr.Button(root, text="Predicted Rating ", command=self.classify)

button.pack()

root.mainloop()

if \_\_name\_\_ == "\_\_main\_\_":

root =tkr.Tk()

tkMessageBox.showinfo("welcome","add your detail here")

c = Classifier(filename="googleplaystore.csv", class\_attr="Rating" )

c.calculate\_priori()

category=tkSimpleDialog.askstring("category","Enter the Category(eg.BEAUTY):")

content\_rating=tkSimpleDialog.askstring("content\_rating","Enter the content\_rating(HINT:Who can use your APP):")

Generes=tkSimpleDialog.askstring("Generes","Enter the Generes(eg.Beauty):")

Type=tkSimpleDialog.askstring("Type","Enter the Type(HINT:Free or Paid):")

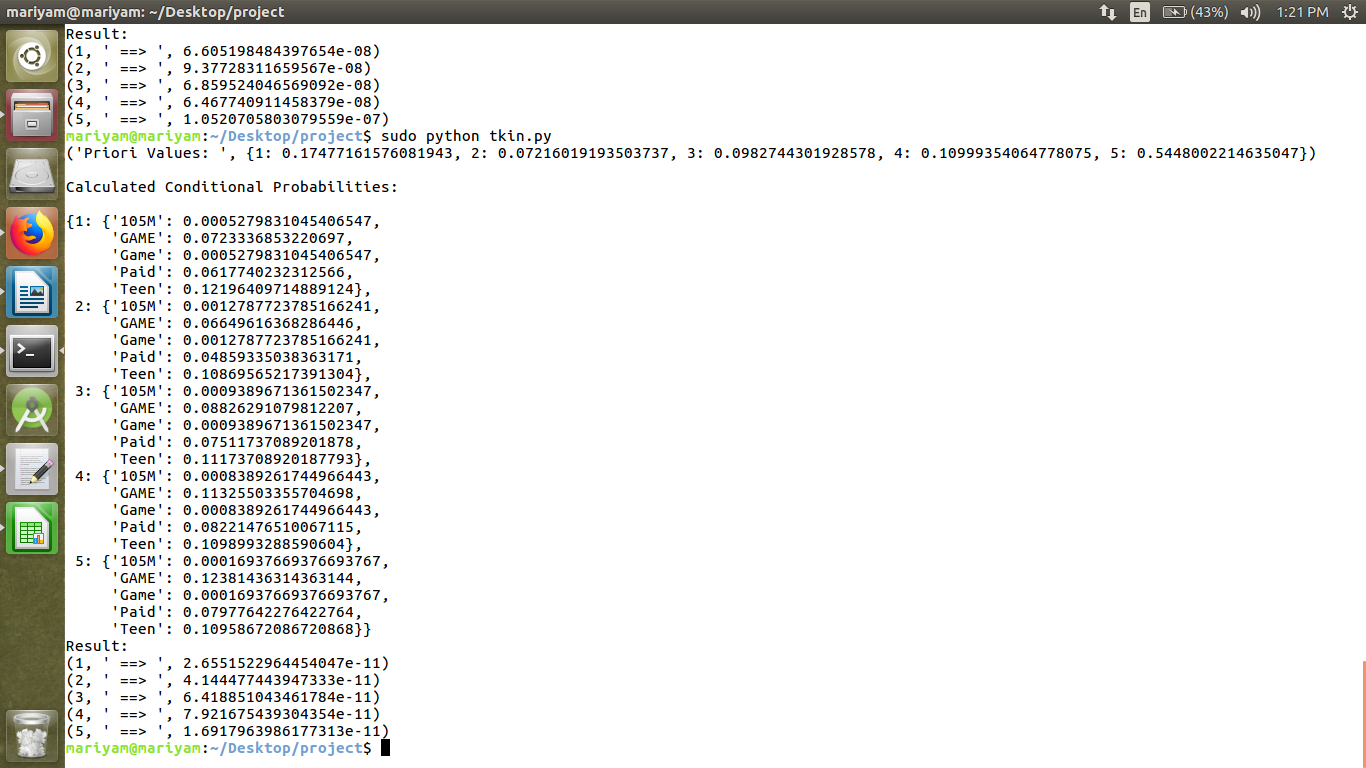
Size=tkSimpleDialog.askstring("Size","Enter the Size(eg.58M):")

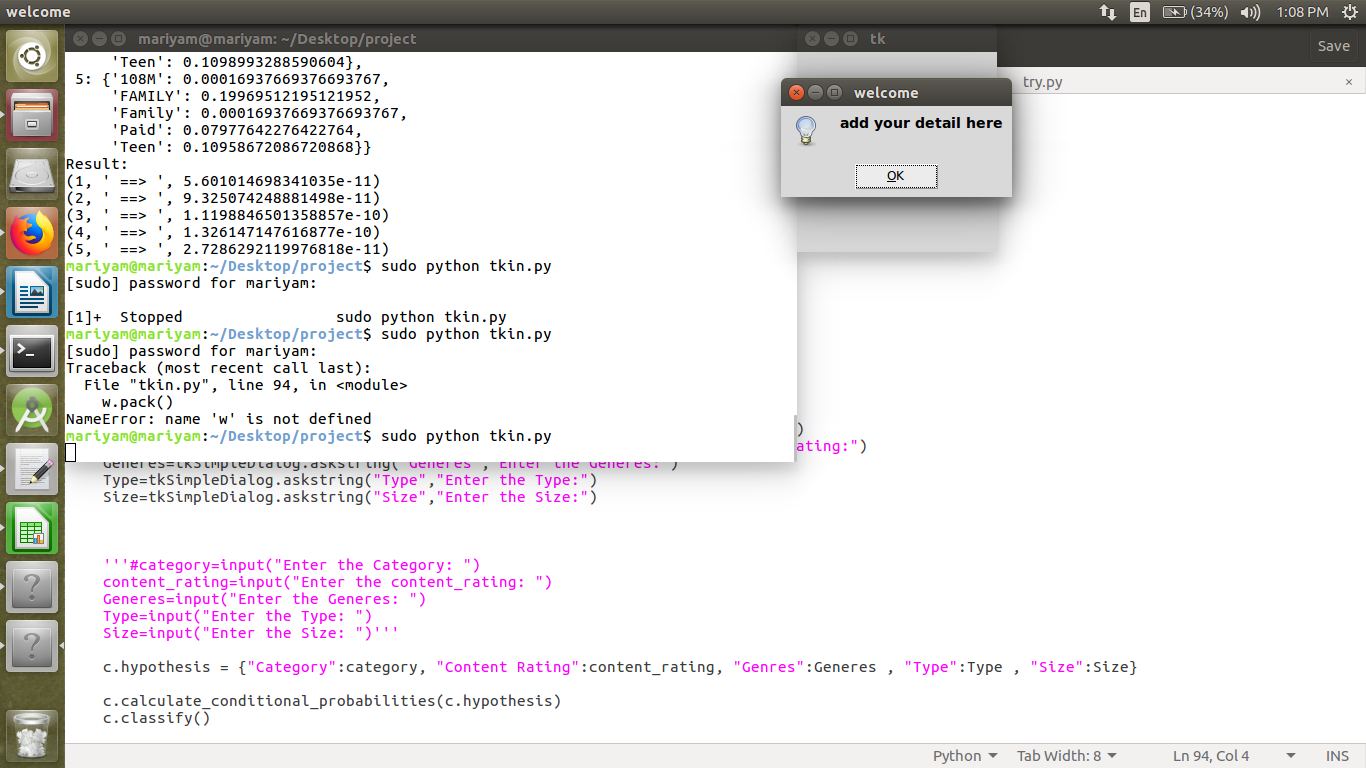
c.hypothesis = {"Category":category, "Content Rating":content\_rating, "Genres":Generes , "Type":Type , "Size":Size}

c.calculate\_conditional\_probabilities(c.hypothesis)

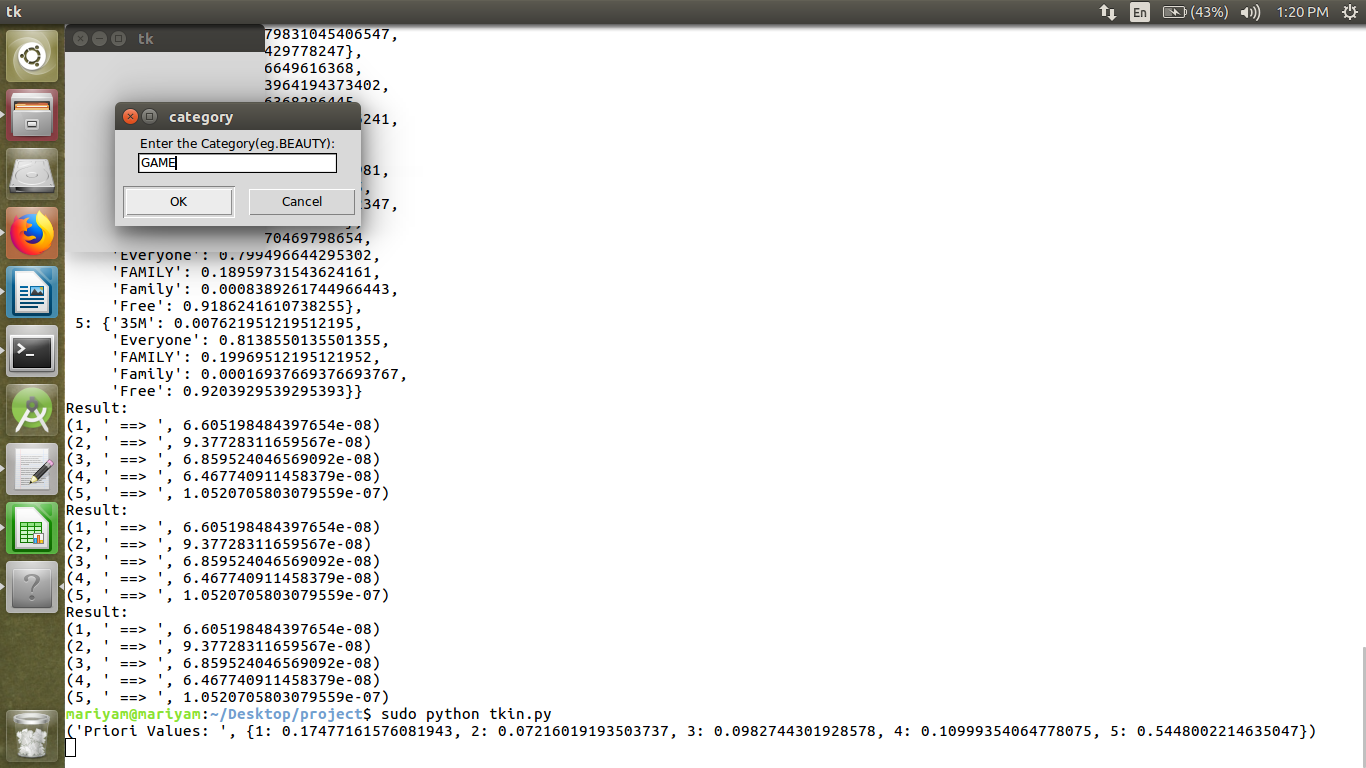
c.classify()

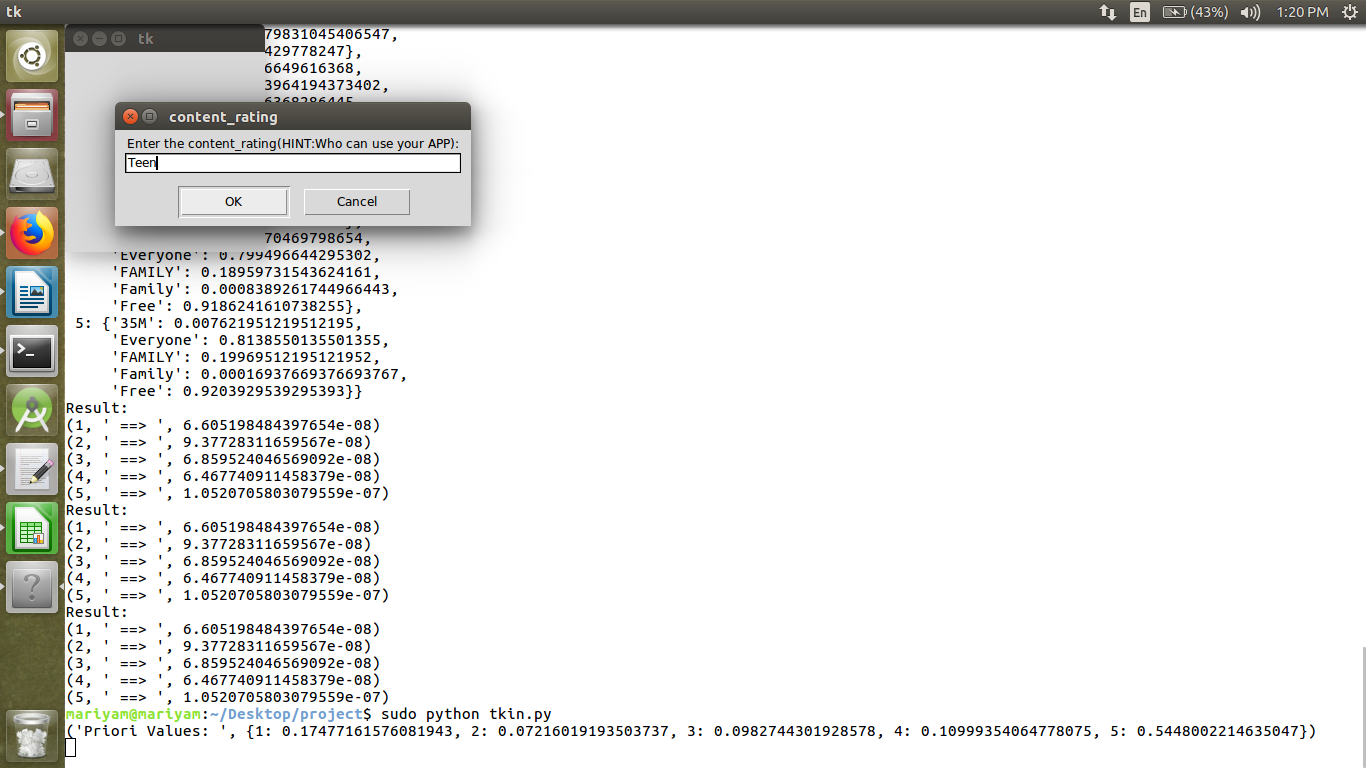
**4.2 Screen Shots**

**Fig 4.2.1:calculating probability of each evidence with individual outome.**

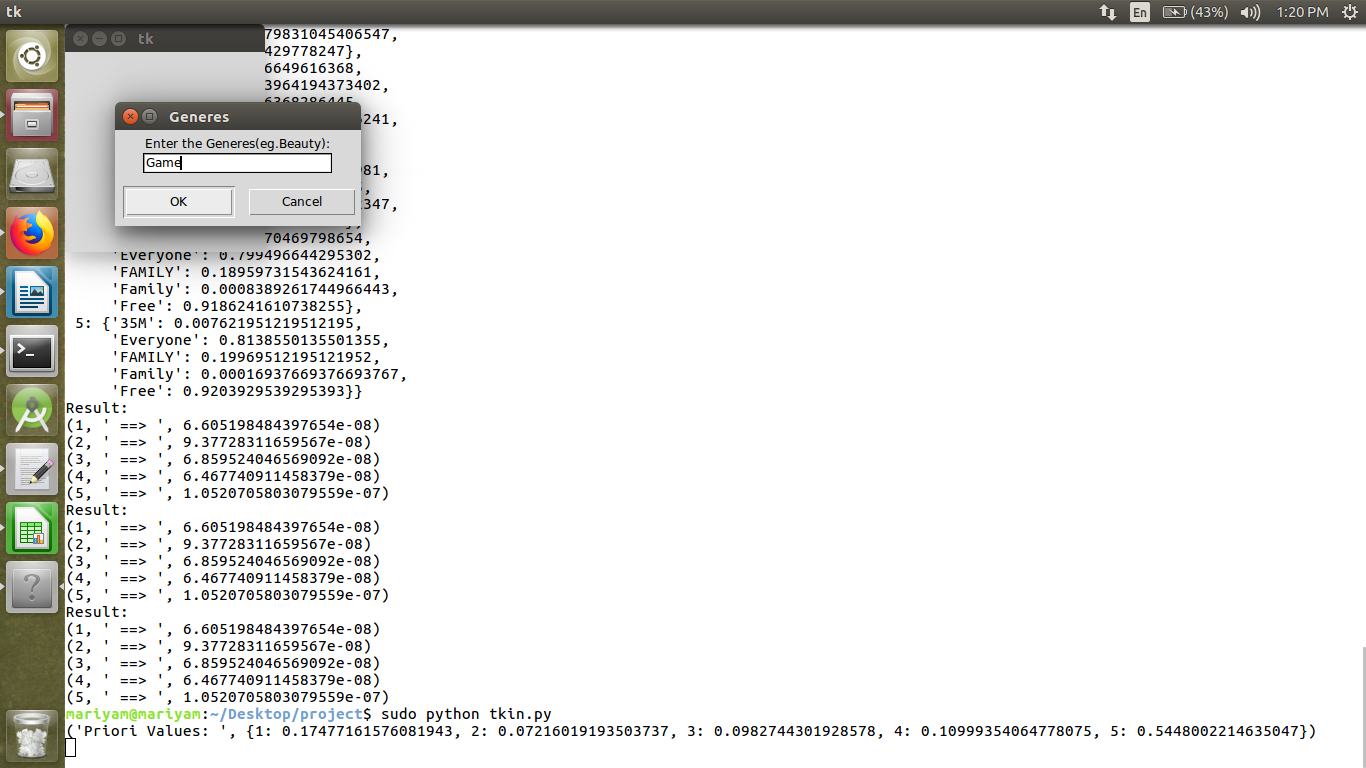


**Fig 4.2.2: displaying dialoug box to enter the detail of app for which we have to predict rating**

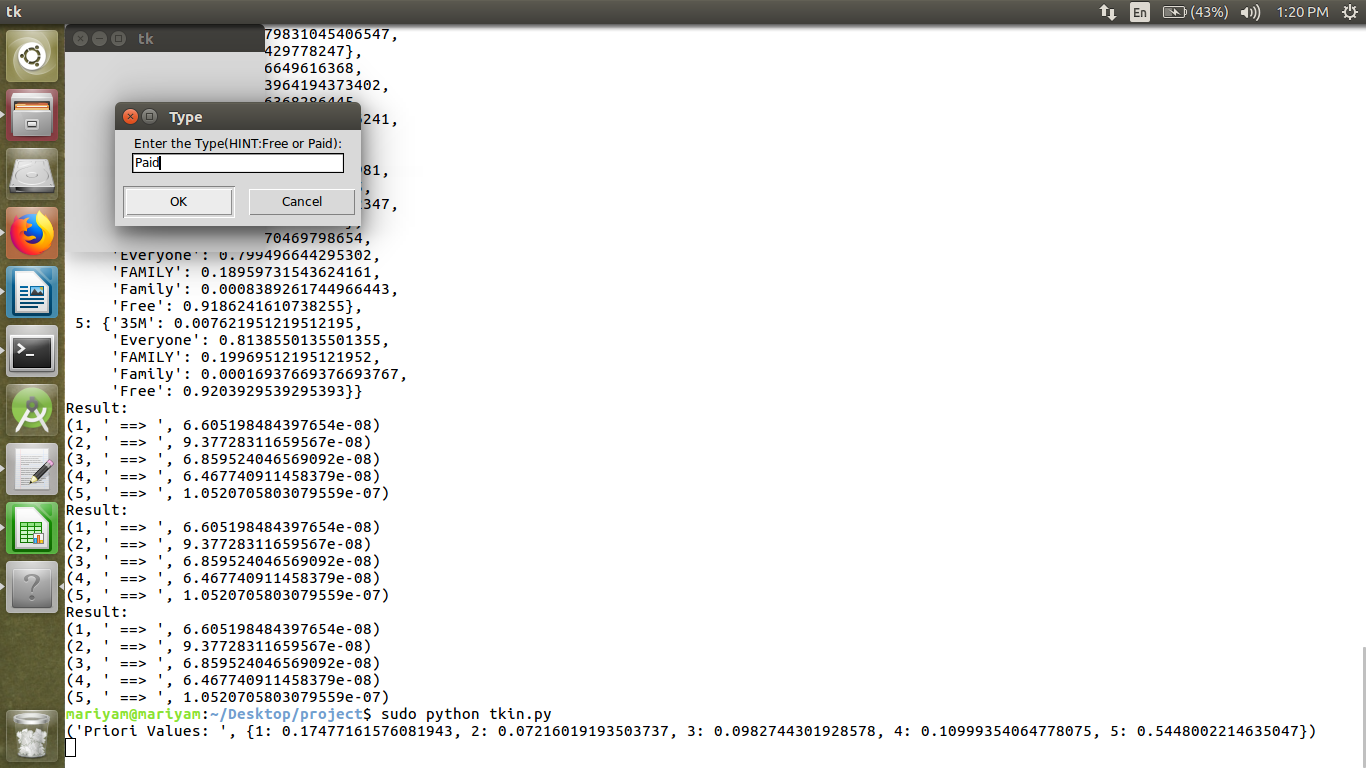
****

 **Fig 4.2.3: Entering the category of App**

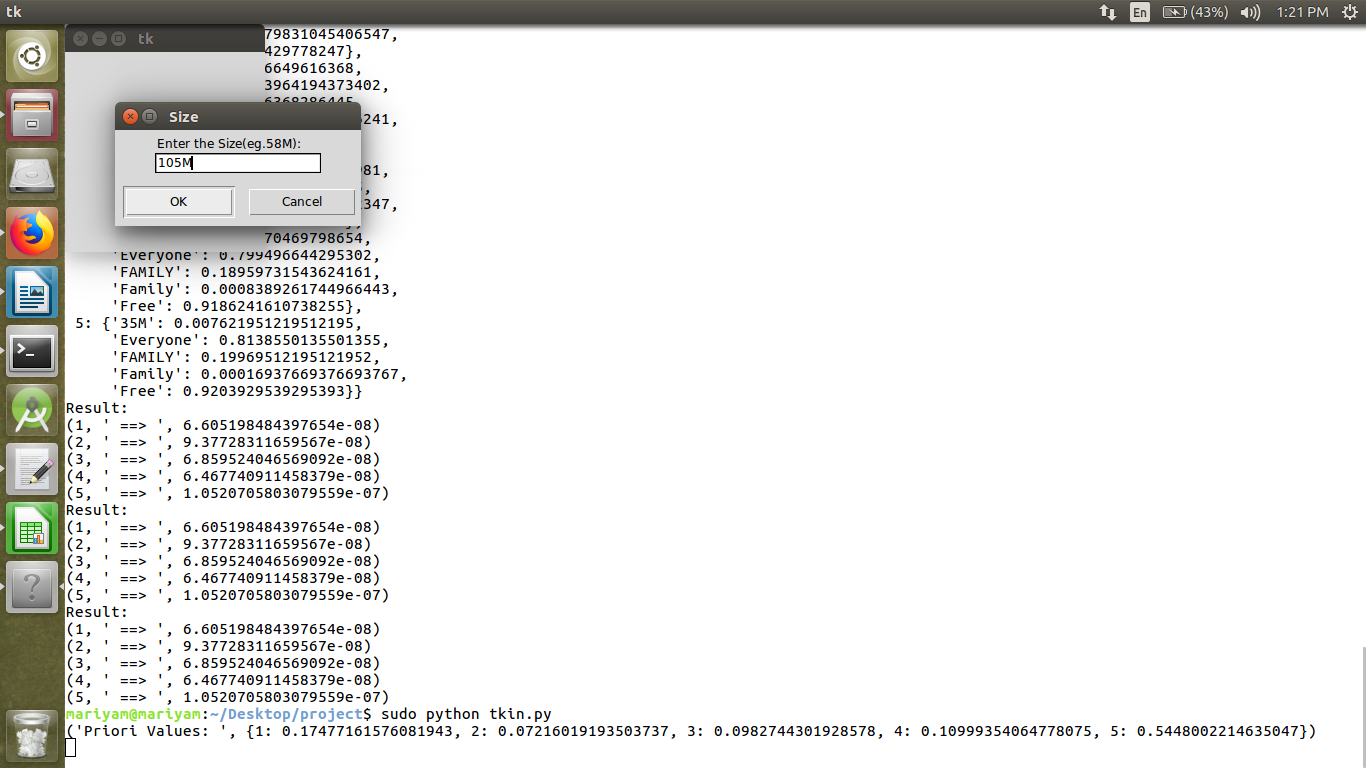
**Fig 4.2.4: Entering the content rating**



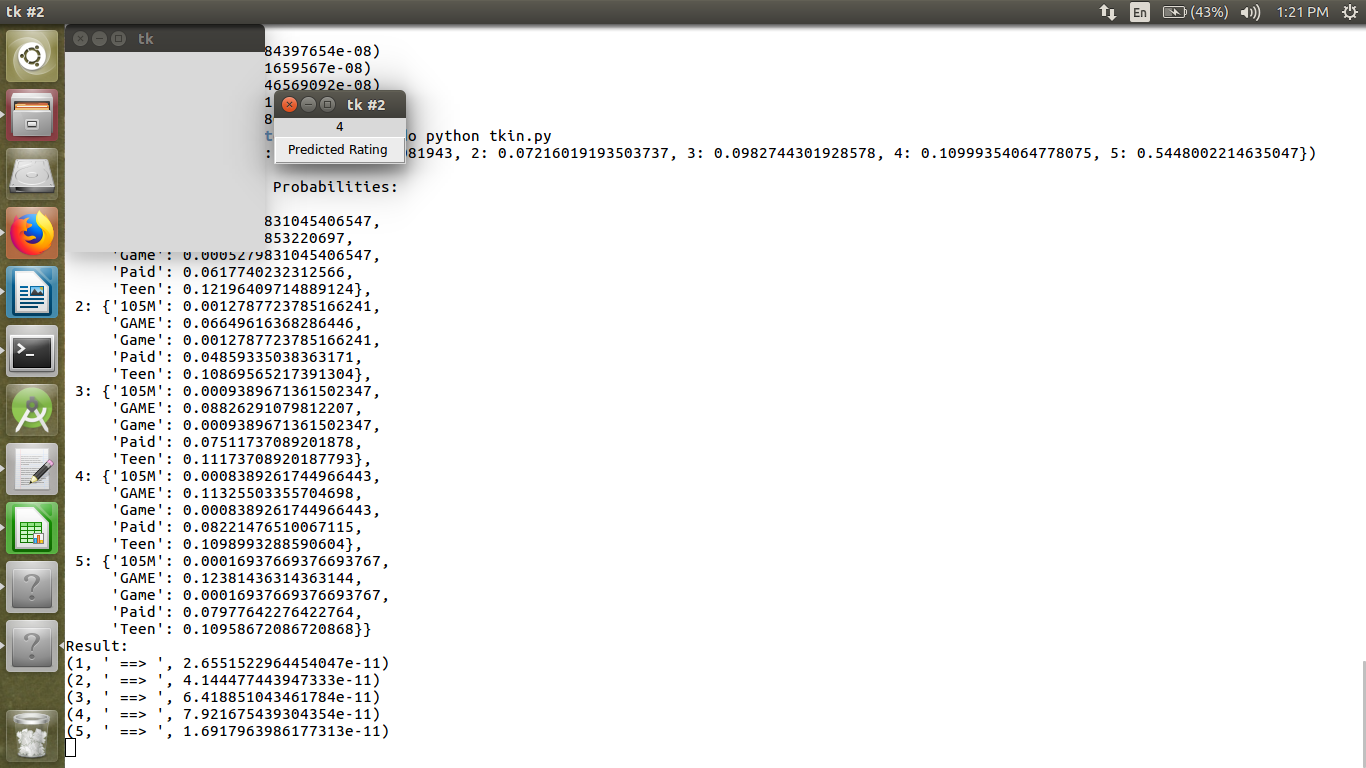
**Fig 4.2.5: Entering the Generes**

****

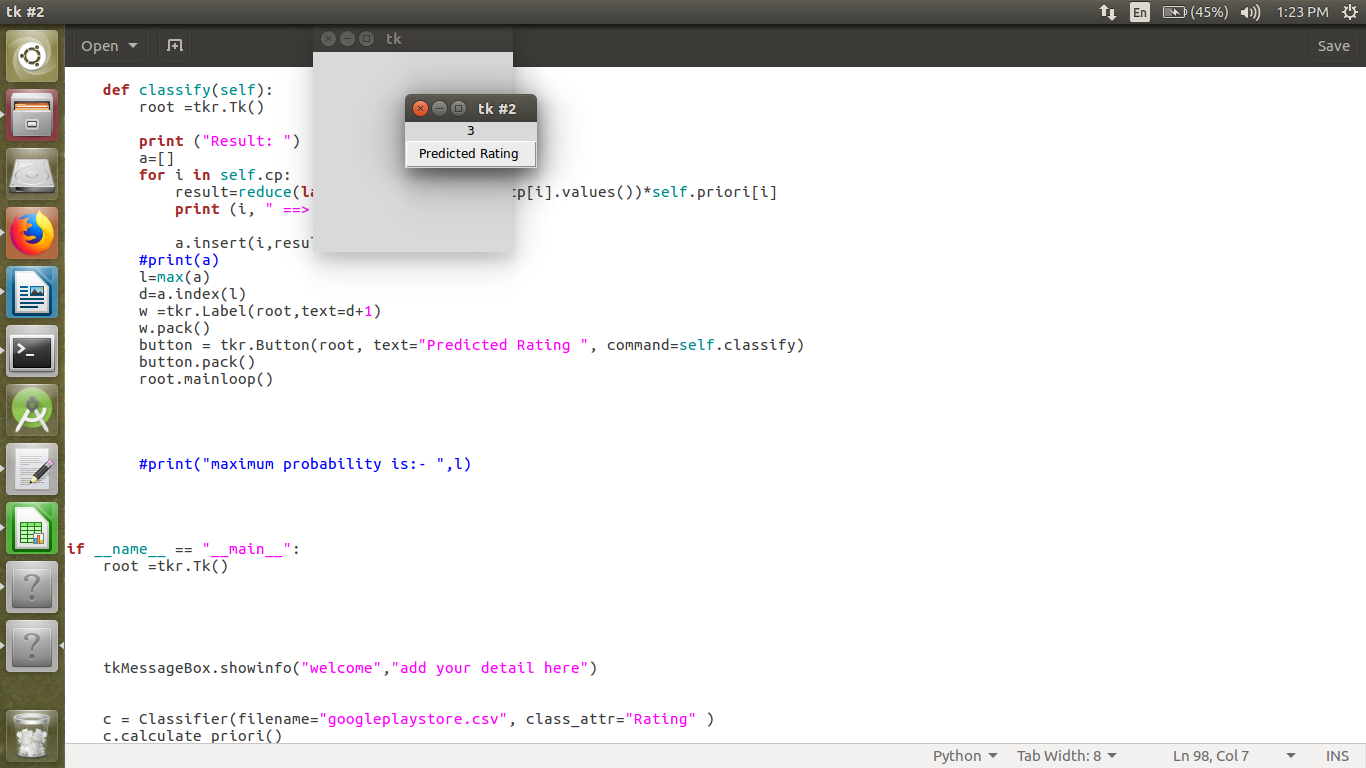
**Fig 4.2.2: Entering the type of App whether it is Free or Paid**

****

**Fig 4.2.2: Entering the size of App**

****

**Fig 4.2.2: Result of predicted rating is displayed of given App**

****

**Fig 4.2.2:ed Result of Predicted Rating of another App is Display**

**CHAPTER 5**

**CONCLUSION**

**5.1 Conclusion**

The previous work about app recommendation mainly focus on users or apps, and features of apps are generally considered as additional information to improve the quality of user preferences prediction. In this paper, we propose a naive bayes algorithm to predict user ratings on apps, which is motivated by a simple observation: a user likes an app because he likes several features of the app rather than all the features. Our contribution is to transform the user-app rating prediction problem to user-feature rating prediction problem, and utilize the prediction result to derive user ratings on apps .

**5.2 Future Scope**

In the future, We plan to explore more factors and evaluate our approach with datasets I.e We are planning to do it with sentiment analysis i.e separating positive and negative review. We can predict the rating of the app that if the review is positive then rating will be good but if the review is bad the rating will be low.

**References**

[1]"Mobile Application Rating Prediction via Feature-Oriented Matrix Factorization - IEEE Conference Publication", Ieeexplore.ieee.org, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8029770/citations#citations. [Accessed: 18- Apr- 2019].

[2]"How Naive Bayes Algorithm Works? (with example and full code) | ML+", Machine Learning Plus, 2019. [Online]. Available: https://www.machinelearningplus.com/predictive-modeling/how-naive-bayes-algorithm-works-with-example-and-full-code/. [Accessed: 18- Apr- 2019].

[3]"veeruviolenzz/Playstoredata-rating-prediction", GitHub, 2019. [Online]. Available: https://github.com/veeruviolenzz/Playstoredata-rating-prediction/blob/master/playstoredata%20and%20rating%20prediction.ipynb. [Accessed: 18- Apr- 2019].

[4]"Graphical User Interfaces with Tk — Python 3.7.3 documentation", Docs.python.org, 2019. [Online]. Available: https://docs.python.org/3/library/tk.html. [Accessed: 18- Apr- 2019].

[5] Gupta, Lavanya. “Google Play Store Apps.” Kaggle, 3 Feb. 2019, www.kaggle.com/lava18/google-play-store-apps.