УДК 004

Методы машинного обучения для анализа данных выпущенных на Google Play приложениях

Фатеенков Данила Витальевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В статье рассмотрен набор данных о приложениях, которые были опубликованы в Google Play. Описывается процесс подготовки набора для анализа с использованием язык программирования Python и соответствующих модулей (Pandas, NumPy, SkLearn и другие). На основе приведённых к подходящему формату для анализа данных используются технологии машинного обучения для визуализации и поиска зависимостей в характеристиках, заданных в наборе.
Ключевые слова: анализ данных, машинное обучение, Python, Pandas, Matplotlib.

Machine learning methods for analyzing data released on Google Play apps

Fateenkov Danila Vitalievich
Sholom-Aleichem Priamursky State University
Student

Abstract
The article considers a set of data about the applications that were published on Google Play. It describes the process of preparing a set for analysis using the Python programming language and related modules (Pandas, NumPy, SkLearn, and others). Based on the data brought to a suitable format for analysis, machine learning techniques are used to visualize and find dependencies in the characteristics given in the set.
Keywords: data analysis, machine learning, Python, Pandas, Matplotlib.

1. Введение

1.1 Актуальность

Технологии машинного обучения в настоящее время быстро развиваются и очень востребованы во многих сферах жизни человека. Большие данные в настоящий момент являются неотъемлемой частью IT-сферы.

Для работы с такими данными необходимо уметь работать с соответствующим языком программирования или ПО. Умение работать с данными может повысить эффективность при составлении прогнозов или получении необходимой информации из большого набора.

Одним из самых популярных сервисов на Android смартфонов является Google Play. Ежедневно в нём выпускаются сотни новых приложений. Для понимания, что на данный момент востребовано среди пользователей, разработчику можно изучить данные о выпущенных приложениях. Для такой задачи хорошо подходят технологии машинного обучения.

1.2 Обзор исследований

Б.А. Пернебай описал работу и реализацию принятий решений методом Дерево решений с использованием готовых библиотек (Pandas, Sklearn) языка программирования Python [1]. Е.А. Недорезова и Т.А. Самойлова рассмотрели разработку с использованием языка программирования Python (а также модуля Sklearn) модели анализа текстовой информации на основе многомерной модели алгоритма Наивного Байеса [2]. А.Д. Кубегенова, Ж.С. Иксебаева и Е.С. Кубегенов применили методы машинного обучения для анализа данных о ВИЧ-инфицированных и рассмотрели аспекты исследования данных, глубокого анализа данных [3]. О.Н. Сериков описал реализацию с использованием языка программирования Python метода семантического анализа LSA [4]. П.А. Пальмин описал применение логистической регрессии для решения задачи двоичной классификации на языке программирования Python (с использованием Sklearn) [5].

1.3 Цель исследования

Цель – провести анализ данных о приложениях, которые были опубликованы в Google Play с использованием языка программирования Python.

2. Материалы и методы

Для реализации поставленной задачи используется язык программирования Python и модули, необходимые для анализа данных: Pandas, NumPy, Matplotlib.

Также используется набор данных о приложениях, которые были опубликованы в Google Play [6].

3. Результаты и обсуждения

Необходимо сначала проверить набор данных. В этот этап входит поиск незаполненных элементов с последующим заполнением ячеек определённым значением или удалением целой записи из набора.

После подключения всех необходимых для работы библиотек (pandas, numpy, seaborn, scipy и sklearn), загружается набор данных через функцию “load_csv()”:

```
%matplotlib inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy
```

```
import scipy.stats as stats
import seaborn as sns

data = pd.read_csv("googleplaystore.csv")
data
```

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & Design
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Art & Design;Pretend Play
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
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & Design
...	...	...	...	...	...	...	...	...	...	...
10837	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3.6M	100+	Free	0	Everyone	Education
10838	Parkinson Exercices FR	MEDICAL	NaN	3	9.5M	1,000+	Free	0	Everyone	Medical
10839	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	Varies with device	1,000+	Free	0	Mature 17+	Books & Reference
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19M	10,000,000+	Free	0	Everyone	Lifestyle

Рисунок 1. Набор данных, загруженный с использованием load_csv

Набор данных содержит 10841 запись и 13 характеристик:

1. App – название приложения;
2. Category – категория приложения;
3. Rating – рейтинг приложения;
4. Reviews – количество отзывов о приложении;
5. Size – размер приложения после установки;
6. Installs – количество установок приложения за всё время существования приложения в магазине;
7. Type – модель, по которой распространяется приложение (бесплатно или платно);
8. Price – цена приложения;
9. Content Rating – возрастное ограничение, установленное на приложение;
10. Genres – жанр, к которому относится приложение;
11. Last Updated – дата последнего обновления приложения;
12. Current Ver – версия приложения на момент сбора набора данных;
13. Android Ver – версия устройства, на котором поддерживается работа приложения.

В качестве целевого параметра выбран Rating. Он является вещественным и если воспользоваться параметром “unique()”, то можно увидеть все значения, которые принимает данный параметр. Все значения представлены в виде вещественных чисел и нет необходимости их форматировать. Основные характеристики данного параметра можно увидеть, если использовать метод “describe()” (см. рис. 2).

```
data['Rating'].describe()

count    9360.000000
mean        4.191838
std         0.515263
min         1.000000
25%         4.000000
50%         4.300000
75%         4.500000
max         5.000000
Name: Rating, dtype: float64
```

Рисунок 2. Характеристики параметра “Rating”

Также необходимо проверить набор на наличие пустых ячеек в записях. Проверяется это следующим образом:

```
missing_values=data.isnull().sum().sort_values(ascending=Fal
se)
```

Функция “isnull()” находит все пустые ячейки в записях, а метод “sum()” использован для получения количества таких ячеек. Метод “sort_values” нужен для сортировки значений. Было найдено 1487 пропущенных значений из которых:

1. Rating – 1474 пустых ячеек;
2. Current Ver – 8 пропущенных значений;
3. Android Ver – 3 пропущенных значений;
4. Type – 1 пустая запись;
5. Content Rating – 1 пустая запись.

В данном случае стоит удалить записи с пропущенными значениями, так как усреднить их не получится (потому что числовое значение представлено только одно, когда все остальные представлены в виде строк):

```
data.dropna(inplace=True)
```

После удаления, в наборе осталось 9360 записей и 13 характеристик.

В наборе приложения разделены по категориям. С помощью параметра “unique()” можно узнать, какие категории представлены в наборе (см. рис. 3).

```
data['Category'].unique()

array(['ART_AND_DESIGN', 'AUTO_AND_VEHICLES', 'BEAUTY',
       'BOOKS_AND_REFERENCE', 'BUSINESS', 'COMICS', 'COMMUNICATION',
       'DATING', 'EDUCATION', 'ENTERTAINMENT', 'EVENTS', 'FINANCE',
       'FOOD_AND_DRINK', 'HEALTH_AND_FITNESS', 'HOUSE_AND_HOME',
       'LIBRARIES_AND_DEMO', 'LIFESTYLE', 'GAME', 'FAMILY', 'MEDICAL',
       'SOCIAL', 'SHOPPING', 'PHOTOGRAPHY', 'SPORTS', 'TRAVEL_AND_LOCAL',
       'TOOLS', 'PERSONALIZATION', 'PRODUCTIVITY', 'PARENTING', 'WEATHER',
       'VIDEO_PLAYERS', 'NEWS_AND_MAGAZINES', 'MAPS_AND_NAVIGATION'],
      dtype=object)
```

Рисунок 3. Категории приложений, представленные в наборе данных

Чтобы определить, приложения каких категорий появляются в наборе чаще всего, нужно построить гистограмму по категориям и количеству приложений в них. Для построения используется countplot из модуля Seaborn:

```
graph1 = sns.countplot(y="Category",data=data)
graph1.set(yticks=range(0,34))
graph1
```

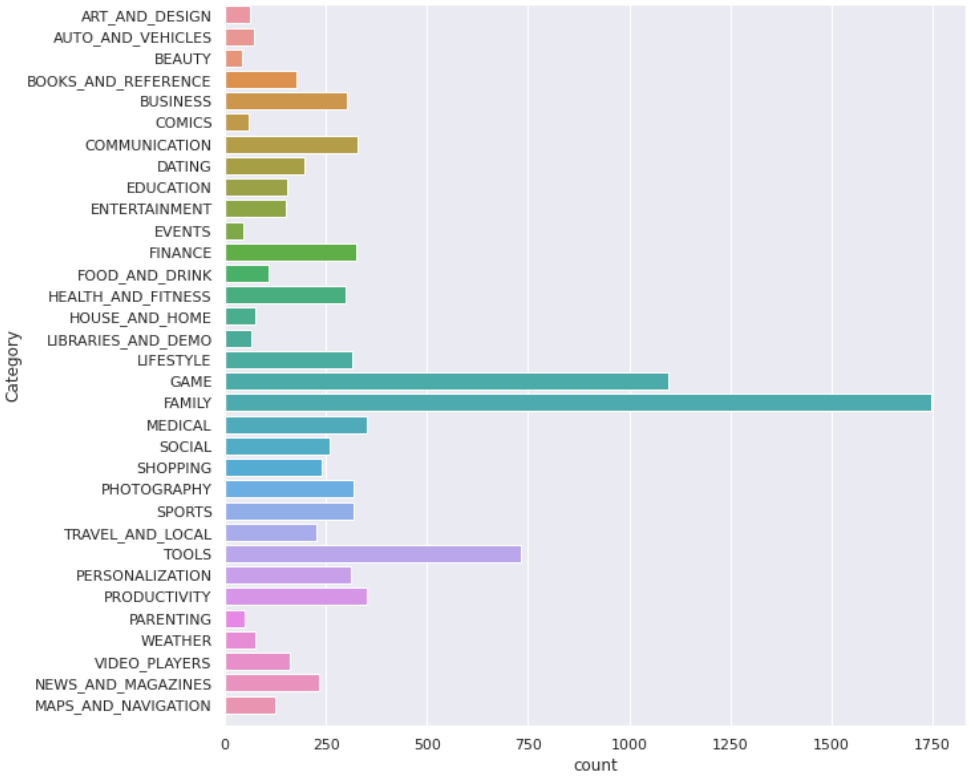

Рисунок 4. Гистограмма по категориям и количеству приложений в них

Преобладающими категориями в наборе являются “Family” (семейные приложения) и “Game” (игры). Также стоит построить диаграмму распределения параметра Rating среди всех категорий. Для этого используется catplot модуля Seaborn. В качестве параметра kind указывается “box”. По оси X указаны категории (названия необходимо повернуть на 90 градусов, чтобы они не накладывались друг на друга), а по Y значение параметра Rating:

```
graph2=
sns.catplot(x="Category",y="Rating",data=data,kind="box",hei
ght = 10)
graph2.set_xticklabels(rotation=90)
graph2
```

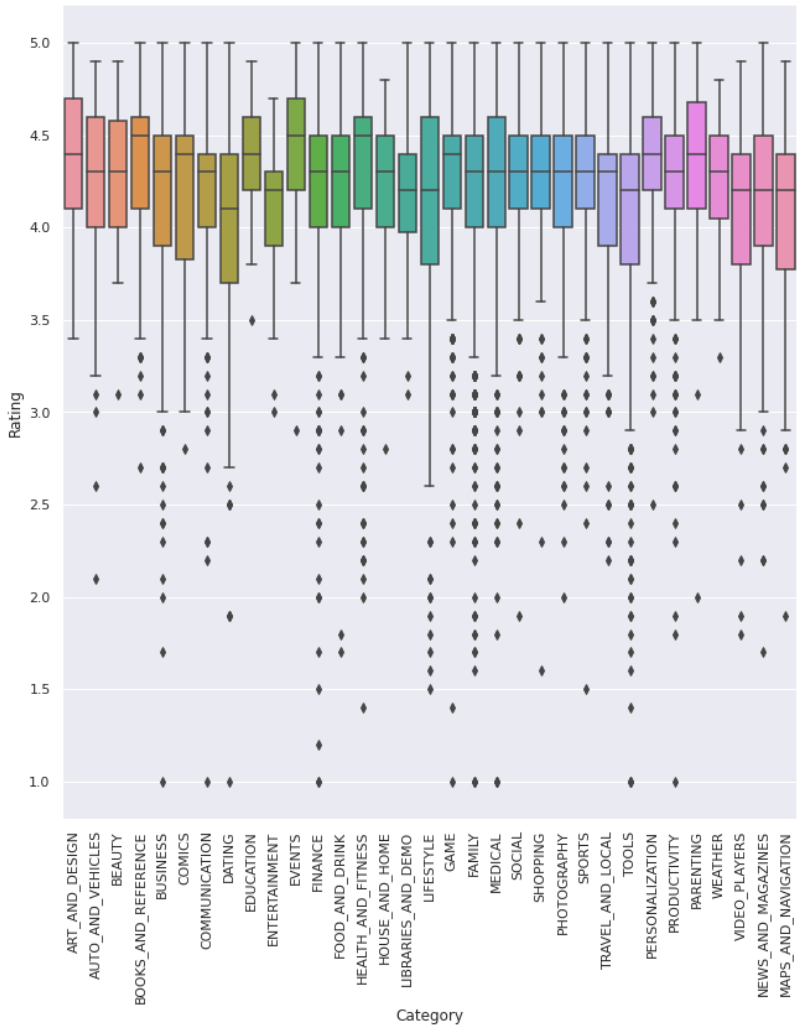

Рисунок 5. Диаграмма распределения оценок среди всех категорий

Среднее значение оценок во всех категориях варьируется от 4 до 4.5, а самое высокое значение у категорий “Events” (мероприятия) и “Health and fitness” (здоровье и фитнес).

Кроме характеристики Category есть другие, которые могут коррелировать с оценкой приложения. Стоит посмотреть зависимость оценки приложения от количества установок приложения. Количество установок представлено в строковом виде (см. рис. 6). Чтобы преобразовать данные значения в числовые, необходимо убрать лишние символы “,” и “+”:

```
data['Installs'] = [x.replace("+",'') and x.replace(",",'')
and int(x) for x in data['Installs']]
data['Installs'].unique()
```

```
array(['10,000+', '500,000+', '5,000,000+', '50,000,000+', '100,000+',
       '50,000+', '1,000,000+', '10,000,000+', '5,000+', '100,000,000+',
       '1,000,000,000+', '1,000+', '500,000,000+', '100+', '500+', '10+',
       '5+', '50+', '1+'], dtype=object)

array([     10000,     500000,    5000000,   50000000,     100000,
            50000,    1000000,   10000000,       5000,  100000000,
       1000000000,       1000,  500000000,        100,        500,
               10,          5,         50,          1])
```

Рисунок 6. Значения параметра “Installs” до и после преобразования

У популярных приложений много скачиваний, но их в выборке немного. Увидеть это, если построить соответствующую гистограмму (см. рис. 7). На гистограмме можно увидеть, что приложений, которые были скачаны более 50 миллионов раз значительно меньше, а значит их можно не отображать на диаграмме рассеяния.

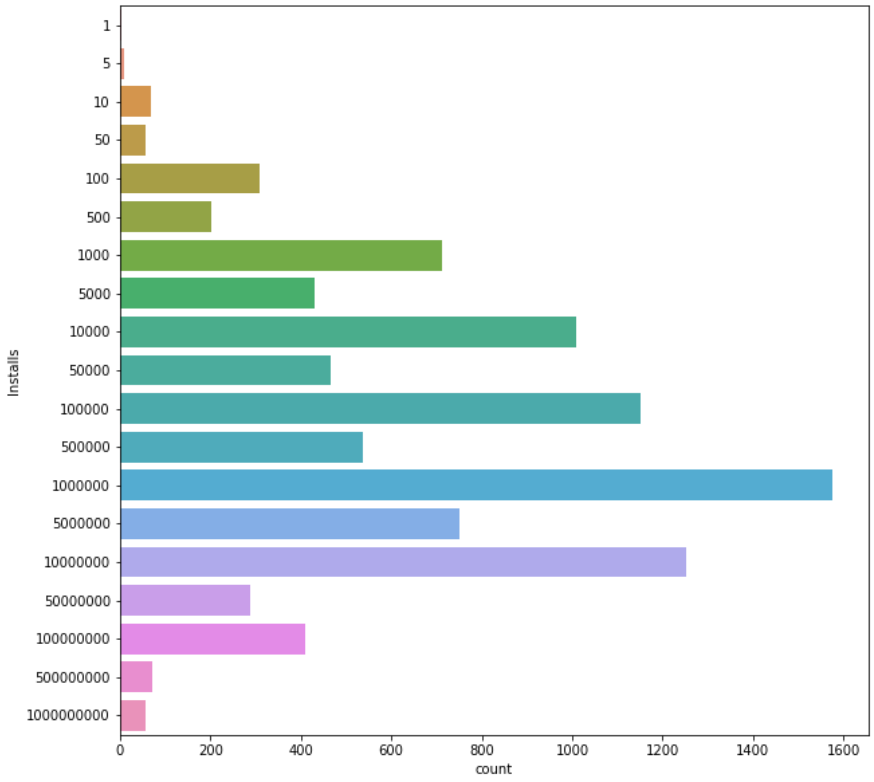

Рисунок 7. Гистограмма, показывающая количество приложений, превысивших определённый порог скачиваний

При построении диаграммы рассеяния не отображены приложения, превысивших 50 миллионов скачиваний (см. рис. 8). Диаграмму рассеяния можно построить, используя функцию “regplot” из модуля Seaborn. Корреляция между характеристиками “Installs” и “Rating” минимальная и её можно не учитывать при дальнейшей работе. Это можно установить из полученной диаграммы, а также из расчёта коэффициента корреляции, применяя функцию “corrcoef” из модуля NumPy:

```
    sns.regplot(x="Installs",                    y="Rating",
data=data[data['Installs']<50000000])
    print(np.corrcoef(data['Installs'],data['Rating']))
```

Коэффициент корреляции равен 0.051, что является низким показателем.

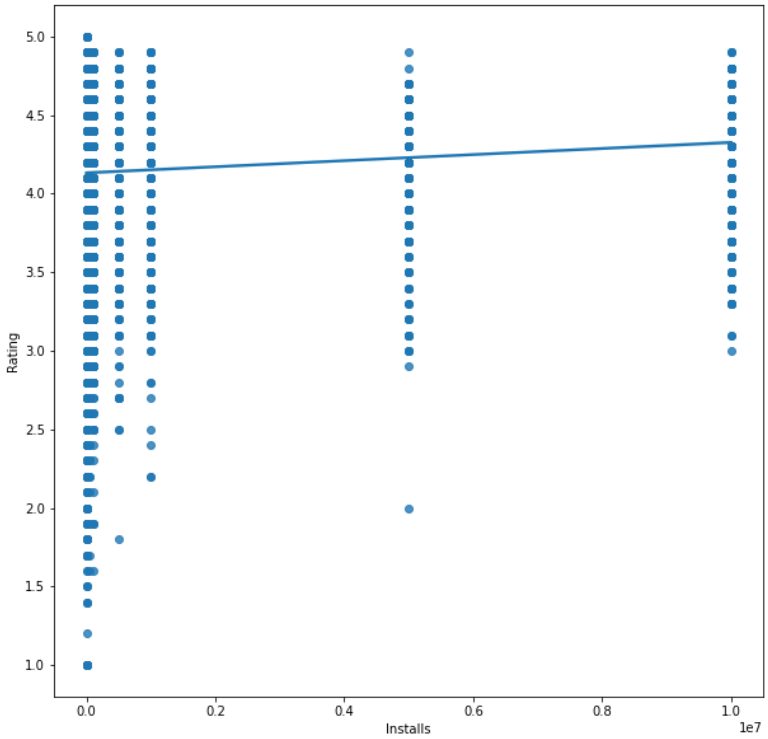

Рисунок 8. Диаграмма рассеяния для параметров “Installs” и “Rating”

Также в наборе данных представлен параметр “Reviews”. Может присутствовать корреляция между этим параметром и “Installs”. Перед построением диаграммы рассеяния, необходимо перевести значения параметра “Reviews” в числовой тип данных:

```
    plt.figure(figsize = (10,10))
    sns.regplot(x="Installs",                    y="Reviews",
data=data[data['Installs']<10000000])
    print(np.corrcoef(data['Installs'],data['Reviews']))
```

Коэффициент корреляции составляет 0.64, что может означать следующее: при возрастании значения количества скачиваний, возрастает также и количество отзывов о приложении. Но при этом параметр параметр “Rating” слабо коррелирует с “Reviews” – коэффициент равен 0.068.

Рейтинг приложения может быть связан с его способом распространения – если приложение платное, то оценок у него меньше. Для начала стоит узнать долю бесплатных приложений в наборе данных: “data['Type'].value_counts(sort = True)”. Всего представлено 8715 бесплатных приложений (или 93.1%), а платных приложений всего 645 или 6.9%.

Необходимо привести значения параметра “Price” к вещественному типу данных. Для этого достаточно удалить “$” из строки. После приведения к вещественному типу, можно построить диаграмму рассеяния (см. рис. 9).

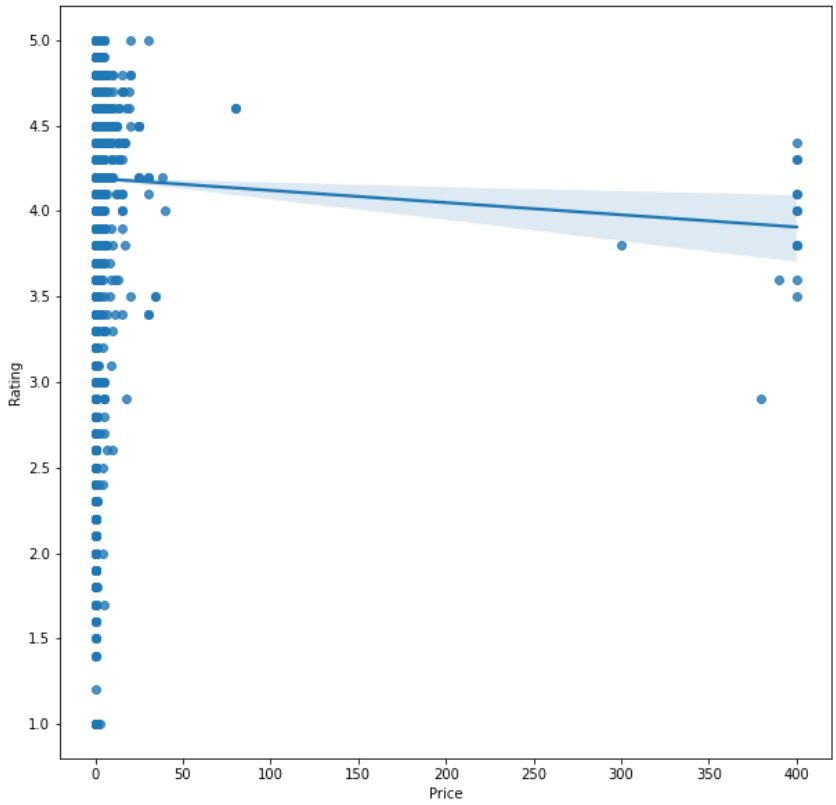

Рисунок 9. Диаграмма рассеяния для “Rating” и “Price”

Можно увидеть, что в наборе данных присутствуют приложения, цена которых достигла 400 долларов. Узнать, какие это приложения, можно следующим образом: “data[data['Price'] >= 350].head(8)”.

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
4197	most expensive app (H)	FAMILY	4.3	6	1.5M	100	Paid	399.99	Everyone	Entertainment	July 16, 2018	1.0	7.0 and up
4362	💎 I'm rich	LIFESTYLE	3.8	718	26M	10000	Paid	399.99	Everyone	Lifestyle	March 11, 2018	1.0.0	4.4 and up
4367	I'm Rich - Trump Edition	LIFESTYLE	3.6	275	7.3M	10000	Paid	400.00	Everyone	Lifestyle	May 3, 2018	1.0.1	4.1 and up
5351	I am rich	LIFESTYLE	3.8	3547	1.8M	100000	Paid	399.99	Everyone	Lifestyle	January 12, 2018	2.0	4.0.3 and up
5354	I am Rich Plus	FAMILY	4.0	856	8.7M	10000	Paid	399.99	Everyone	Entertainment	May 19, 2018	3.0	4.4 and up
5356	I Am Rich Premium	FINANCE	4.1	1867	4.7M	50000	Paid	399.99	Everyone	Finance	November 12, 2017	1.6	4.0 and up
5357	I am extremely Rich	LIFESTYLE	2.9	41	2.9M	1000	Paid	379.99	Everyone	Lifestyle	July 1, 2018	1.0	4.0 and up
5358	I am Rich!	FINANCE	3.8	93	22M	1000	Paid	399.99	Everyone	Finance	December 11, 2017	1.0	4.1 and up

Рисунок 10. Самые дорогие приложения в Google Play

При построении диаграммы рассеяния для параметров “Reviews” и “Price” можно увидеть, что количество отзывов о приложении становится меньше при увеличении цены. Это также подтверждается тем, что у дорогих приложений намного меньше скачиваний чем у бесплатных, а значит и отзывов тоже.

В наборе представлен параметр, характеризующий размер приложения. Он также может влиять на количество скачиваний и рейтинг приложения. Для

того, чтобы это проверить, нужно перевести значения параметра в числовой тип данных. Для начала нужно узнать все уникальные значения, которые принимает данный параметр: “data['Size'].unique()” (см. рис. 11).

```
array(['19M', '14M', '8.7M', '25M', '2.8M', '5.6M', '29M', '33M', '3.1M',
       '28M', '12M', '20M', '21M', '37M', '5.5M', '17M', '39M', '31M',
       '4.2M', '23M', '6.0M', '6.1M', '4.6M', '9.2M', '5.2M', '11M',
       '24M', 'Varies with device', '9.4M', '15M', '10M', '1.2M', '26M',
       '8.0M', '7.9M', '56M', '57M', '35M', '54M', '201k', '3.6M', '5.7M',
       '8.6M', '2.4M', '27M', '2.7M', '2.5M', '7.0M', '16M', '3.4M',
       '8.9M', '3.9M', '2.9M', '38M', '32M', '5.4M', '18M', '1.1M',
       '2.2M', '4.5M', '9.8M', '52M', '9.0M', '6.7M', '30M', '2.6M',
       '7.1M', '22M', '6.4M', '3.2M', '8.2M', '4.9M', '9.5M', '5.0M',
       '5.9M', '13M', '73M', '6.8M', '3.5M', '4.0M', '2.3M', '2.1M',
       '42M', '9.1M', '55M', '23k', '7.3M', '6.5M', '1.5M', '7.5M', '51M',
       '41M', '48M', '8.5M', '46M', '8.3M', '4.3M', '4.7M', '3.3M', '40M',
```

Рисунок 11. Значения параметра “Size”.

Размер приложения может быть как в мегабайтах, так и в килобайтах, а также данный параметр принимает значение “Varies with device”. Такое значение стоит заменить на среднее среди всех значение Size для дальнейшей работы. Для этого необходимо сгруппировать значения Size по удобному параметру и используя метод “transform(‘mean’)” заменить все NaN значения на среднее.

Размер приложения не сильно влияет на рейтинг приложения, количество скачиваний и количество отзывов.

После завершения преобразования данных, стоит обучить модель машинного обучения на наборе. Для прогнозирования рейтинга приложения можно использовать регрессионные модели (линейная регрессия, дерево решений, метод случайного дерева и другие).

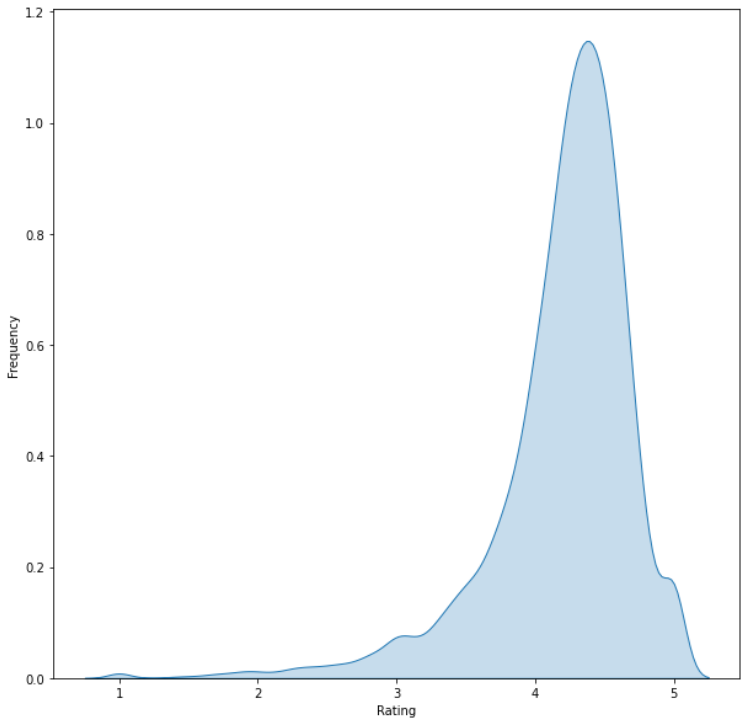

Рисунок 12. KDE Диаграмма для параметра Rating

Также перед началом обучения модели, следует заменить строковые значения в характеристике “Type”. Так как уникальных значений всего 2, то их можно заменить на 0 и 1:

```
c = {'Free':0, 'Paid':1}
data['Type'] = data['Type'].map(c)
```

Для решения поставленной задачи (определение рейтинга приложения на основе вводимых значений некоторых характеристик) подходит регрессионная модель. Первая модель для обучения – линейная регрессия (Linear Regression). Перед началом обучения необходимо разделить данные на тестовые и тренировочные:

```
from sklearn.model_selection import train_test_split
X = data.drop(['Rating','App','Genres','Last Updated', 'Current Ver', 'Content Rating', 'Android Ver','Category'],axis=1)
Y = data['Rating']
X_train, X_val, y_train, y_val = train_test_split(X, Y, test_size=0.25)
X.head(8)
```

Набор был разделён на 25% тестовых и 75% тренировочных данных. В X входят следующие параметры: Reviews, Size, Installs, Type, Price. Y является целевой характеристикой Rating.

```
LR = lm.LinearRegression()
model = LR.fit(X_train,y_train)
```

Линейная регрессия показывает плохие результаты и не подходит для работы с данным набором, так как метрика R^2 равна 0.015 (что является очень слабым результатом).

Вторая модель для обучения – метод случайного дерева (Random Tree):

```
from sklearn.ensemble import RandomForestRegressor
Tr = RandomForestRegressor()
model = Tr.fit(X_train,y_train)
```

Модель случайного дерева показала себя хорошо – метрика R^2 равна 0.85. Данная модель может подойти для дальнейшей работы с набором данных.

Третья модель, которая может подойти для решения поставленной задачи – модель повышения градиента (Gradient Boosting):

```
from sklearn.ensemble import GradientBoostingRegressor
HR = GradientBoostingRegressor()
model = HR.fit(X_train,y_train)
```

Коэффициент метрики R^2 составляет 0.22, что является низким показателем и данная модель не подходит для работы с набором данных.

Последняя модель, выбранная для обучения – дерево принятия решения (Decision Tree):

```
from sklearn.tree import DecisionTreeRegressor
DT = DecisionTreeRegressor()
model = DT.fit(X_train,y_train)
```

Дерево принятия решений подходит для работы с загруженными данными лучше всего, так как метрик R^2 равна 0.98, что является наивысшим показателем среди всех моделей, рассмотренных выше.

Таким образом, были изучен набор данных приложениях, опубликованных в Google Play. Были также изменён вид значений некоторых характеристик для дальнейшей обработки полученной информации. На основе полученных результатов применены модели машинного обучения и определена наиболее подходящая для предсказания значения характеристики Rating на основе заданных значений других характеристик.

Библиографический список

1. Пернебай Б.А. Python | Регрессия дерева решений с использованием Sklearn // Polish Journal of Science. 2021. № 38-1 (38). С. 51-56.
2. Недорезова Е.А., Самойлова Т.А. Многомерная модель классификации текстов // Системы компьютерной математики и их приложения. 2018. № 19. С. 193-198.
3. Кубегенова А.Д., Иксебаева Ж.С., Кубегенов Е.С. Многомерная модель классификации текстов // Системы компьютерной математики и их приложения. 2021. № 8-1 (10). С. 48-55.
4. Сериков О.Н. Анализ трендов YouTube при помощи визуализации и машиного обучения // Вестник молодёжной науки России. 2019. № 1. С. 54.
5. Пальмин П.А. Использование логистической регрессии для решения задачи классификации на примере Python // Научно-практические исследования. - 2019. - № 8-7 (23). - С. 51-57.
6. Google Play Store Apps | Kaggle. URL: https://www.kaggle.com/datasets/lava18/google-play-store-apps. - Дата обращения: 15.05.2022.