## **DATS6401**

## **Visualization of Complex Data**

**Dr. Reza Jafari**

**Final Term Project**

**Nusrat Nawshin**

**NN**

**03/29/2022**

Table of Content

Table of Figures and Tables

Abstract:

The objective of this project is to apply course learning objectives to visualize complex data using Python and an interactive wed based dashboard.

Introduction:

Description of the dataset:

Preprocessing Dataset:

Outlier detection and removal:

Outliers are data points that are far from other data points or unusual values in a dataset. Outliers are problematic for many statistical analyses because they can cause tests to either miss significant findings or distort real results. Unfortunately, there are no strict statistical rules for definitively identifying outliers. Finding outliers depends on subject-area knowledge and an understanding of the data collection process. Visualizing using boxplot is one of the graphical ways of detecting outliers. After visualizing the distribution using boxplot (Figure 1), here we can see that if we remove the outliers from price column it’ll have only zero prices but considering this dataset these values aren’t outliers as there are very few apps those are paid in play store. On the other side, App Rating column doesn’t have any outliers but its imbalanced.

|  |  |
| --- | --- |
| Shape, rectangle  Description automatically generated | Chart  Description automatically generated |
| Figure 1: Box-Plot of App Price and App Rating | |

After removing outliers from the other three numeric columns (figure 2), there is still some outliers present. But removing more data will remove too much data.

|  |  |
| --- | --- |
| Chart  Description automatically generated with medium confidence | A picture containing shape  Description automatically generated |
| Chart  Description automatically generated | |
| Figure 2: Box-plot of before and after removing outliers (Maximum Installs, Minimum Installs, Rating Count) | |

After removing the outliers, the dataset now consists of 17,82,652 number of rows and 22 columns.

Principal Component Analysis (PCA):

Principal component analysis, or PCA, is a statistical procedure that summarize the information content in large data tables by means of a smaller set of “summary indices” that can be more easily visualized and analyzed [1]. To perform PCA analysis on this dataset only the 5 numeric columns are used and using python SKlearn package PCA() function with ‘mle’ as n\_component argument it reduces the feature space from 5 to 4. But from the cumulative explained variance vs number of component graph (figure 3) it can be observed that even with only 3 features we can get almost 90% explained variance. So, I am removing one more feature and the reduced feature space is now with 3 features and with almost 90% explained variance.

|  |  |
| --- | --- |
| Chart, line chart  Description automatically generated | Chart, line chart  Description automatically generated |
| Figure 3: PCA-Cumulative Explained Variance vs Number of Components | |

From the correlation matrix of the original feature space and the PCA reduced feature space (figure 4), we can see that there were 2 medium correlated features and PCA removed that on the reduced feature space.

|  |  |
| --- | --- |
| Table  Description automatically generated with low confidence | A screenshot of a computer  Description automatically generated with medium confidence |
| Figure 4: PCA-Correlation matrix of original feature space and reduced feature space | |

Data transformation:

As the column ‘Install’ was as string, I am converting it to numeric by removing the ‘+’ or ‘,’.

Normality Test:

There are two primary methods to find normal distribution on a dataset. Graphical method and statistical test. Histogram and QQ plot are two popular and effective method to identify normal distribution. Shapiro-test, Kolmogorov-Smirnov-test, D’Agostino-Pearson Test are the three popular statistical methods to identify normal distribution. All three test considers null hypothesis on normal distribution. Now the dataset has 6 numeric columns, Install, Rating, Rating Count, Minimum Installs, Maximum Installs, and price. Test results of all normality tests indicated the distribution of these 5 columns aren’t normal as the p value were significant (reject the null hypothesis) (table 1).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table 1: Normality Test | | | | |
| Column Name | Test Name | Test Statistics | P-Value | Comment |
| Minimum Installs | Shapiro-test | 0.029 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Kolmogorov-Smirnov | 1 | 0 | Sample doesn’t look Gaussian (reject H0) |
| D’Agostino-Pearson | 162034.7 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Maximum Installs | Shapiro | 0.036 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Kolmogorov-Smirnov | 1 | 0 | Sample doesn’t look Gaussian (reject H0) |
| D’Agostino-Pearson | 153416.5 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Rating | Shapiro | 0.898 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Kolmogorov-Smirnov | 0.985 | 0 | Sample doesn’t look Gaussian (reject H0) |
| D’Agostino-Pearson | 15099.6 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Rating Count | Shapiro | 0.036 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Kolmogorov-Smirnov | 1 | 0 | Sample doesn’t look Gaussian (reject H0) |
| D’Agostino-Pearson | 173807.9 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Price | Shapiro | 0.026 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Kolmogorov-Smirnov | 0.5 | 0 | Sample doesn’t look Gaussian (reject H0) |
| D’Agostino-Pearson | 134385.4 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Installs | Shapiro | 0.029 | 0 | Sample doesn’t look Gaussian (reject H0) |
| Kolmogorov-Smirnov | 1 | 0 | Sample doesn’t look Gaussian (reject H0) |
| D’Agostino-Pearson | 162034.7 | 0 | Sample doesn’t look Gaussian (reject H0) |

In the graphical tests, both the QQ-plot (figure 5) and histogram (figure 6) are indicating non normality in the features.

A picture containing shoji

Description automatically generated

Figure 5: QQ-Plot of numeric columns

A picture containing shoji, crossword puzzle, clipart

Description automatically generated

Figure 6: Histogram of numeric columns

As the columns aren’t normally distributed, we could perform a normal transformation on these datasets, but by doing so the columns original data would be manipulated and become meaningless, for example rating and price would include negative values. For this concern I am not performing normal distribution transformation on this dataset.

Heatmap & Pearson correlation coefficient matrix:

Correlation heatmaps are a type of plot that visualize the strength of relationships between numerical variables. Pearson correlation coefficient is a statistical measure of the linear relationship between two variables [2]. Which measures from -1 to 1. Correlation value near -1 indicates strong negative correlation and near 1 means strong positive correlation. Values near 0 indicates no correlation between the features.

Graphical user interface, application

Description automatically generated

Figure 7: Correlation matrix of features

From the correlation plot above (figure 7), we can notice that minimum installs and maximum installs both have strong positive correlation with installs. Whereas rating count has medium positive correlation with installs, minimum and maximum installs. Only price and free have low negative correlation. All other columns seem to have neutral to no correlation to each other.

Statistics:

|  |
| --- |
| Table 2: Statistics of the Numeric Columns |
|  | |

If we now look into the distribution (figure 8) of rating and price column we can see that the majority of rating is around 0 or 4-5 and those are free apps mostly. Also, majority of the app prices are around 0 and they are not ad supported.

|  |  |
| --- | --- |
| Chart, line chart  Description automatically generated |  |
| Figure 8.1: Kernel Density Estimation of app rating and price | |
| Figure 8.2: Bivariate Distribution of App price higher than 0 | |

Now for better observation and faster code execution, lets look into the price distribution of app with price more than 0$ and their ratings. Here from the plot (figure 8.2) it’s clearly noticeable that majority app prices aren’t much higher than few cents and they have rating of 0 or in between 3 to 5.

Data Visualization:

a. Line-plot

Chart, line chart

Description automatically generated

Figure 9: Line plot of App Installs vs Rating

Here in the figure 9 of app installs vs rating by free apps, we can observe that free apps are having higher installations and the rating is around 3 and 4.

Chart, line chart

Description automatically generated

Figure 9: App rating vs released year by content rating

Here in figure 9, I have compared apps different content rating categories by rating and released year. We can observe that the rating kept decreasing over the years and adults only apps had a peak rating in 2013 but a huge drop in 2015. Also, many apps were kept unrated on 2012 and 2015.

b. Bar-plot:

A picture containing chart

Description automatically generated

Figure 10: Count of app categories

As we can observe from the distribution of apps on different categories there are only a few apps which are not free and education app category has the highest number of apps in google play store. Business and music & audio are the second highest apps.  
c. Count-plot

Chart

Description automatically generated

Figure 10: Count of app categories by ad supported

As we found out the top three app categories are from education, business, and music & audio, from figure 10 we can see that most of the business and education apps doesn’t support adds but music & audio category have high ad supported app counts.

Chart, bar chart

Description automatically generated

Figure 11: Count of app categories by add supported

In terms of content rating of apps, most of the apps are for everyone and among them majority supports ad. (figure 11)

d. Cat-plot

Chart, bar chart

Description automatically generated

Figure 12: Count of apps over the years by ad supported

Here in figure 12, we can see that release of apps were increasing significantly over the years and in 2020 there were almost 250k apps and majority of them weren’t ad supported.

e. Pie-chart

|  |  |
| --- | --- |
| Chart, pie chart  Description automatically generated | Chart, pie chart  Description automatically generated |
| Figure 13: Pie chart of percentage of free apps and content rating | |

Here we can see only 2% of apps were not free and 88.5% of apps were for everyone.

f. Displot

Chart, histogram

Description automatically generated

Figure 14: Distribution of app rating

From figure 14 we can say that majority of the apps had 0 ratings and they were free. There is more than 40% probability that app rating is around zero. Other than that, there is less than 5% of probability that app rating is around 4-5.  
g. Pair plot

Chart, line chart

Description automatically generated

Figure 15: Pair plot of all the features

Text, whiteboard

Description automatically generated

Figure 16: Pair plot of correlation between features excluding the installs

i. Hist-plot

Chart, histogram

Description automatically generated

Figure 17: Histogram of app installs by free apps

Here in figure 17, we can see majority of apps have more than 100 to 1000 installs and they are mostly free apps.

k. Kernel density estimate

Chart

Description automatically generated

Figure 18: Kernel density estimation plot of ratings by in-app purchases

Majority of app ratings are near zero and they don’t have in app purchase option. (figure 18)

l. Scatter plot and regression line

Chart

Description automatically generated

Figure 19: Scatter plot with linear regression

There is a linearly positive relationship between rating and installs. High ratings result into higher installations (figure 19).

m. Multivariate Box plot

Chart, bar chart

Description automatically generated

Figure 20: Multivariate box plot of rating by per app categories

From the boxplot of categories (figure 20) and ratings we can say that overall, the distribution of ratings in each category are not balanced and there are no outliers. Also, only news & magazines category has a balanced rating distribution.

Chart, box and whisker chart

Description automatically generated

Figure 21: Multivariate box plot of rating by free apps

In this box plot of rating vs editors’ choice, we can see some outliers in the free apps with editors’ choice. Overall app with editor’s choice has only high ratings. (Figure 21)

o. Violin plot

A picture containing text, clock

Description automatically generated

Figure 22: Violin plot of numeric columns

Here we can see the distribution of the six numeric columns in violin plot from. Maximum installs, price and rating counts seem to have high amounts of data in only around zero (figure 22).

Subplots:

Square

Description automatically generated with medium confidence

Figure 23: Top 5 apps, categories, minimum android, and rating distribution

Here I have focused on the top 5 of apps, categories, and minimum android. In terms of categories google play service, messenger and WhatsApp are the top 3 apps. Most apps are from education category and android 4.1 is the most apps minimum requirement.

Dashboard:

To visualize the google play store market in dashboard I am using python dash and plotly packages. As the original dataset has more than 2 million rows, implementing dashboard with this huge data lowers drown the performance of it. So, for dashboard implementation I have reduced the dataset by only taking installs count more than 100k and rating count more than 4500. Farther, I have dropped Scrapped time and Last updated column as it had high missing values. The final dataset for dash has 50775 rows × 22 columns.

I have used 5 tabs in the dashboard. In the first tab I am showing some statistics of the user preferred filtered apps.

Graphical user interface, text

Description automatically generated

Figure 24: Dashboard tab 1 - Statistics

Graphical user interface, application

Description automatically generated

Figure 25: Outputs of tab 1

In this tab (figure 24) user can filter the apps based on the released date range (DataPickerRange), category (dropdown menu), free, ad supported, in-app purchase and editor’s choice (radio button) and content rating (check box). There is a reset filter button to reset all the filters. In the output field (figure 25), there is a bar chart of top 5 apps and the count of their installs. Beside that there is a graph field to show the filtered apps statistics. In the app statistics output user can select the column from a dropdown menu of the numeric features (figure 26).

Graphical user interface

Description automatically generated with low confidence

Figure 26: Statistics column selection

This tab also shows year wise count of apps and distribution of ratings of the user filtered apps.

In the second tab, I have compared two apps side by side. Here user can select two apps from dropdown menu and in the output section there are four graphs and a comparison table.

Graphical user interface, text, application

Description automatically generated

Figure 27: Dashboard tab 2 – app comparison

User can also category wise filter the apps in the two apps dropdown menu.

|  |  |
| --- | --- |
|  | |
|  |
| Figure 28: Output of tab 2 app comparison |

On the comparison table user can also directly visit the developer website and read the privacy policy of the app.

In the third tab (figure 29), user can filter apps by category, free, ad supported, in-app purchase, editor’s choice, release date, content rating and app rating. In the output there is a list of apps from that filtered apps and user can also download the filtered data in a CSV file.

|  |
| --- |
|  |
|  |

Figure 29: Dashboard tab 3 – App details

Also, by clicking on the table row, user can see that specific app details in a table format bellow. Here also user can directly access the links of developer website and privacy policy.

In the next tab I have visualized the dataset using different types of graphs. User can select the type of graphs from the drop-down menu. There is line, bar, cat, count, pie chart, displot, histogram, heatmap, scatter, box, and violin plot options.

Graphical user interface, application

Description automatically generated

Figure 30: Tab 4 – Graph Visualization

Line Plot:

Graphical user interface, chart, line chart

Description automatically generated

Figure 31: line plot outputs

These line plots indicates that year 2017 had the highest popular app releases but highest popular app installation was on year 2012. However, in 2020 there was a drop of maximum install of popular apps.

Bar Plot:

Graphical user interface, application

Description automatically generated

Figure 32: Bar plot outputs

Count Plot:

In the count plot section (figure 33), I have shown the count of minimum android version, categories, app releases over the years and content rating. Most of the popular apps requires minimum android version 4.1 and up. Tools category has the highest number of apps and most of the apps are from content rating of ‘everyone’.

Chart, bar chart

Description automatically generated

Figure 33: Count plot outputs

Catplot:

Chart, bar chart

Description automatically generated

Figure 34: Cat plot outputs

In the catplot, I have basically showed the count of four Boolean type of columns as plotly package doesn’t have any in built catplot like seaborn. From these graphs (figure 34), we can say most popular apps are free in play store, most of them are ad supported, most of them are not from editors’ choice and most apps doesn’t have in-app purchase option.

Pie Chart:

Chart, pie chart

Description automatically generated

Figure 35: Pie chart outputs

From these pie charts, we can see there are 22.7% of popular apps are from tools category, 75% of popular apps are for everyone, 12.4% of popular apps were released in 2017. Only 0.4% of app are not free in the play store.

Displot:

In this plot section (figure 36), I have visualized the distribution of popular app prices, installs, rating and rating counts over the years.

Chart, bar chart

Description automatically generated

Figure 36: Distribution plot outputs

Heatmap:

A screenshot of a computer

Description automatically generated with medium confidence

Figure 37: Heatmap output

Here is a person correlation matrix plotted as an heatmap.

Histogram and Distplot:

From these distribution we can see that the majority of the popular apps have 4 rating and and app releases increased gradually till 2017 from 2010 but after that there is a decrease in the number of popular apps.

Graphical user interface, chart, histogram

Description automatically generated

Figure 38: Histogram and Dist plot outputs

Scatter Plot:

Graphical user interface, application, Word

Description automatically generated

Figure 39: Scatter plot outputs

These scatterplots indicated that price and rating have a positive linear relationship but installs and price have a constant relationship.

Multivariate Box Plot:

A picture containing chart

Description automatically generated

Figure 40: Box plot outputs

Violin Plot:

Timeline

Description automatically generated

Figure 41: Violin plot outputs

Both these box plots (figure 40) and violin plots (figure 41) are indicating there are many outliers in the distinct categories and content rating. Overall, the distributions aren’t normal.

In the last tab on the dashboard, I have performed a normality test of the numeric columns. User can select the column as well as the type of test they want to perform. There is QQ plot to visually check the normality of the column (figure 43.1) and three statistical tests Shapiro walk, K-S test and D’Agostino test.

Graphical user interface, text, application, email

Description automatically generated

Figure 42: Dashboard Tab 5 – Normality Test

Chart, line chart

Description automatically generated

Figure 43.1: Output of normality test tab

The output of the statistical tests are shown in a table format.

Graphical user interface, text, application

Description automatically generated

Figure 43.2: Output of normality test tab

Recommendations:

Appendix:

References:

[1] [https://www.sartorius.com/en/knowledge/science-snippets/what-is-principal-component-analysis-pca-and-how-it-is-used-507186#:~:text=Principal%20component%20analysis%2C%20or%20PCA,more%20easily%20visualized%20and%20analyzed.](https://www.sartorius.com/en/knowledge/science-snippets/what-is-principal-component-analysis-pca-and-how-it-is-used-507186%23:~:text=Principal%20component%20analysis%2C%20or%20PCA,more%20easily%20visualized%20and%20analyzed.)

[2] <https://vitalflux.com/correlation-heatmap-with-seaborn-pandas/>