google-play-store-data-analysis

September 14, 2024

[139]:
```
import pandas as pd
import numpy as np
import plotly.express as px
import  plotly.io as pio
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
from nltk.sentiment import SentimentIntensityAnalyzer
import nltk
import webbrowser
import os
import warnings
warnings.filterwarnings('ignore')
nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\ayaan\AppData\Roaming\nltk_data…
[nltk_data]   Package vader_lexicon is already up-to-date!
```

[139]:
```
True
```

LOADING THE DATASETS

[140]:
```
apps_df = pd.read_csv('Play Store Data.csv')
```

[141]:
```
reviews_df = pd.read_csv('User Reviews.csv')
```

VIEWING THE DATA

[142]:
```
apps_df.head()
```

[142]:
```
                                                App        Category  Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                                 Coloring book moana  ART_AND_DESIGN     3.9
2  U Launcher Lite – FREE Live Cool Themes, Hide …  ART_AND_DESIGN     4.7
3                                Sketch - Draw & Paint  ART_AND_DESIGN     4.5
4              Pixel Draw - Number Art Coloring Book  ART_AND_DESIGN     4.3

  Reviews  Size     Installs  Type Price Content Rating  \
```

```
0     159   19M      10,000+  Free     0       Everyone
1     967   14M     500,000+  Free     0       Everyone
2   87510  8.7M   5,000,000+  Free     0       Everyone
3  215644   25M  50,000,000+  Free     0           Teen
4     967  2.8M     100,000+  Free     0       Everyone

                      Genres      Last Updated         Current Ver  \
0               Art & Design   January 7, 2018               1.0.0
1  Art & Design;Pretend Play  January 15, 2018               2.0.0
2               Art & Design    August 1, 2018               1.2.4
3               Art & Design      June 8, 2018  Varies with device
4    Art & Design;Creativity     June 20, 2018                 1.1

    Android Ver
0  4.0.3 and up
1  4.0.3 and up
2  4.0.3 and up
3    4.2 and up
4    4.4 and up
```

[143]:
```
reviews_df.head()
```

[143]:
```
                     App                                Translated_Review  \
0  10 Best Foods for You  I like eat delicious food. That's I'm cooking …
1  10 Best Foods for You    This help eating healthy exercise regular basis
2  10 Best Foods for You                                               NaN
3  10 Best Foods for You         Works great especially going grocery store
4  10 Best Foods for You                                       Best idea us

  Sentiment  Sentiment_Polarity  Sentiment_Subjectivity
0  Positive                1.00                0.533333
1  Positive                0.25                0.288462
2       NaN                 NaN                     NaN
3  Positive                0.40                0.875000
4  Positive                1.00                0.300000
```

DATA CLEANING

[144]:
```
apps_df = apps_df.dropna(subset=['Rating'])
for column in apps_df.columns:
    apps_df[column].fillna(apps_df[column].mode()[0], inplace=True)
apps_df.drop_duplicates(inplace=True)
apps_df = apps_df[apps_df['Rating'] <= 5]
reviews_df.dropna(subset=['Translated_Review'], inplace=True)
```

DATA TRANSFORMATION

-> CONVERTING THE INSTALLS COLUMN TO NUMERIC STATE BY REMOV-

ING COMMAS AND SYMBOLS

-> CONVERTING THE PRICE COLUMN TO NUMERIC STATE BY REMOVING $ SYMBOL

[145]:
```
apps_df['Reviews'] = apps_df['Reviews'].astype(int)
apps_df['Installs'] = apps_df['Installs'].str.replace(',', '').str.replace('+',␣
↪'').astype(int)
apps_df['Price'] = apps_df['Price'].str.replace('$', '').astype(float)
```

COMBINING THE TWO DATASET

[146]:
```
merged_df = pd.merge(apps_df, reviews_df, on='App', how='inner')
```

[147]:
```
merged_df.head()
```

[147]:

	App	Category	Rating	Reviews	Size	Installs	Type	\
0	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500000	Free	
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500000	Free	
2	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500000	Free	
3	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500000	Free	
4	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500000	Free	

	Price	Content Rating	Genres	Last Updated	\
0	0.0	Everyone	Art & Design;Pretend Play	January 15, 2018	
1	0.0	Everyone	Art & Design;Pretend Play	January 15, 2018	
2	0.0	Everyone	Art & Design;Pretend Play	January 15, 2018	
3	0.0	Everyone	Art & Design;Pretend Play	January 15, 2018	
4	0.0	Everyone	Art & Design;Pretend Play	January 15, 2018	

	Current Ver	Android Ver	\
0	2.0.0	4.0.3 and up	
1	2.0.0	4.0.3 and up	
2	2.0.0	4.0.3 and up	
3	2.0.0	4.0.3 and up	
4	2.0.0	4.0.3 and up	

	Translated_Review	Sentiment	\
0	A kid's excessive ads. The types ads allowed a…	Negative	
1	It bad >:(	Negative	
2	like	Neutral	
3	I love colors inspyering	Positive	
4	I hate	Negative	

	Sentiment_Polarity	Sentiment_Subjectivity
0	-0.250	1.000000
1	-0.725	0.833333
2	0.000	0.000000

```
3               0.500                0.600000
4              -0.800                0.900000
```

[148]:
```
def convert_size(size):
  if 'M' in size:
    return float(size.replace('M', ''))
  elif 'k' in size:
    return float(size.replace('k', '')) / 1024
  else:
    return np.nan

apps_df['Size'] = apps_df['Size'].apply(convert_size)
```

[149]:
```
apps_df.head()
```

[149]:
```
                                                App        Category  Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                                 Coloring book moana  ART_AND_DESIGN     3.9
2  U Launcher Lite - FREE Live Cool Themes, Hide …  ART_AND_DESIGN     4.7
3                               Sketch - Draw & Paint  ART_AND_DESIGN     4.5
4              Pixel Draw - Number Art Coloring Book  ART_AND_DESIGN     4.3

   Reviews  Size  Installs  Type  Price Content Rating  \
0      159  19.0     10000  Free    0.0       Everyone
1      967  14.0    500000  Free    0.0       Everyone
2    87510   8.7   5000000  Free    0.0       Everyone
3   215644  25.0  50000000  Free    0.0           Teen
4      967   2.8    100000  Free    0.0       Everyone

                      Genres      Last Updated         Current Ver  \
0               Art & Design   January 7, 2018               1.0.0
1  Art & Design;Pretend Play  January 15, 2018               2.0.0
2               Art & Design    August 1, 2018               1.2.4
3               Art & Design      June 8, 2018  Varies with device
4    Art & Design;Creativity     June 20, 2018                 1.1

    Android Ver
0  4.0.3 and up
1  4.0.3 and up
2  4.0.3 and up
3    4.2 and up
4    4.4 and up
```

[150]:
```
# LOGARITHMIC CONVERSION
apps_df['log_installs'] = np.log(apps_df['Installs'])
apps_df['log_reviews'] = np.log(apps_df['Reviews'])
```

[151]:
```
apps_df.dtypes
```

[151]:
```
App                object
Category           object
Rating            float64
Reviews             int32
Size              float64
Installs            int32
Type               object
Price             float64
Content Rating     object
Genres             object
Last Updated       object
Current Ver        object
Android Ver        object
log_installs      float64
log_reviews       float64
dtype: object
```

[152]:
```
def rating_group(rating):
  if rating >= 4:
    return 'Top Rated App'
  elif rating >= 3:
    return 'Above Average'
  elif rating >= 2:
    return 'Average'
  else:
    return 'Below Average'

apps_df['Rating_Group'] = apps_df['Rating'].apply(rating_group)
```

[153]:
```
# Deriving a Metric : Revenue Column
apps_df['Revenue'] = apps_df['Price'] * apps_df['Installs']
```

[154]:
```
sia = SentimentIntensityAnalyzer()
```

[155]:
```
# POLARITY SCORE IN SIA
# - POSITIVE
# - NEGATIVE
# - COMPOUND (-1 means very negative & +1 means very positive )
```

ADDING THE SENTIMENTS SCORES

[156]:
```
reviews_df['Sentiment_Score'] = reviews_df['Translated_Review'].apply(lambda x:␣
 ↪sia.polarity_scores(x)['compound'])
```

[157]:
```
reviews_df.head()
```

[157]:
```
                    App                                Translated_Review  \
0  10 Best Foods for You  I like eat delicious food. That's I'm cooking …
1  10 Best Foods for You    This help eating healthy exercise regular basis
3  10 Best Foods for You         Works great especially going grocery store
4  10 Best Foods for You                                    Best idea us
5  10 Best Foods for You                                        Best way

  Sentiment  Sentiment_Polarity  Sentiment_Subjectivity  Sentiment_Score
0  Positive                1.00                0.533333           0.9531
1  Positive                0.25                0.288462           0.6597
3  Positive                0.40                0.875000           0.6249
4  Positive                1.00                0.300000           0.6369
5  Positive                1.00                0.300000           0.6369
```

[158]:
```
apps_df['last_updated'] = pd.to_datetime(apps_df['Last␣
↪Updated'],errors='coerce')
```

EXTRACTING THE YEAR FROM DATE-TIME COLUMN

[159]:
```
apps_df['Year'] = apps_df['last_updated'].dt.year
```

[160]:
```
apps_df.head(3)
```

[160]:
```
                                                 App        Category  Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN     4.1
1                                Coloring book moana  ART_AND_DESIGN     3.9
2  U Launcher Lite - FREE Live Cool Themes, Hide …  ART_AND_DESIGN     4.7

   Reviews  Size  Installs  Type  Price Content Rating  \
0      159  19.0     10000  Free    0.0       Everyone
1      967  14.0    500000  Free    0.0       Everyone
2    87510   8.7   5000000  Free    0.0       Everyone

                      Genres      Last Updated Current Ver   Android Ver  \
0               Art & Design   January 7, 2018       1.0.0  4.0.3 and up
1  Art & Design;Pretend Play  January 15, 2018       2.0.0  4.0.3 and up
2               Art & Design    August 1, 2018       1.2.4  4.0.3 and up

   log_installs  log_reviews   Rating_Group  Revenue last_updated  Year
0      9.210340     5.068904  Top Rated App      0.0   2018-01-07  2018
1     13.122363     6.874198  Above Average      0.0   2018-01-15  2018
2     15.424948    11.379508  Top Rated App      0.0   2018-08-01  2018
```

[161]:
```
html_files_path = "./"
if not os.path.exists(html_files_path):
    os.makedirs(html_files_path)
```

[162]:
```
plot_containers = ""
```

[163]:
```
def save_plot_as_html(fig, filename, insight):
    global plot_containers
    filepath = os.path.join(html_files_path, filename)
    html_content = pio.to_html(fig, full_html=False, include_plotlyjs='inline')
    # Append the plot and its insight to plot_containers
    plot_containers += f"""
    <div class="plot-container" id="{filename}"␣
 ↪onclick="openPlot('{filename}')">
        <div class="plot">{html_content}</div>
        <div class="insights">{insight}</div>
    </div>
    """
    fig.write_html(filepath, full_html=False, include_plotlyjs='inline')
```

[164]:
```
# Define your plots
plot_width = 400
plot_height = 300
plot_bg_color = 'black'
text_color = 'white'
title_font = {'size': 16}
axis_font = {'size': 12}
```

Figure - 1 (category_analysis)

[165]:
```
# Category Analysis Plot
category_counts = apps_df['Category'].value_counts().nlargest(10)
fig1 = px.bar(
    x = category_counts.index,
    y = category_counts.values,
    labels = {'x': 'Category', 'y': 'Count'},
    title = 'Top Categories on Play Store',
    color = category_counts.index,
    color_discrete_sequence=px.colors.sequential.Plasma,
    width = plot_width,
    height = plot_height
)
fig1.update_layout(
    plot_bgcolor = plot_bg_color,
    paper_bgcolor = plot_bg_color,
    font_color = text_color,
    title_font = title_font,
    xaxis = dict(title_font = axis_font),
    yaxis = dict(title_font = axis_font),
    margin = dict(l=10, r=10, t=30, b=10)
)
```

```
fig1.update_traces(marker=dict(line=dict(color=text_color, width=1)))
save_plot_as_html(fig1, "category_analysis.html", "The top categories on the␣
 ↪Google Play Store are dominated by tools, entertainment, and productivity␣
 ↪apps. This suggests users are looking for apps that either provide utility␣
 ↪or offer leisure activities.")
```

Figure - 2 (type_count)

[166]:
```
type_counts = apps_df['Type'].value_counts()
fig2 = px.pie(
    values = type_counts.values,
    names = type_counts.index,
    title = 'App Type Distribution',
    color_discrete_sequence=px.colors.sequential.RdBu,
    width = plot_width,
    height = plot_height
)
fig2.update_layout(
    plot_bgcolor = 'black',
    paper_bgcolor = 'black',
    font_color = 'white',
    title_font = title_font,
    margin = dict(l=10, r=10, t=30, b=10)
)
# fig2.update_traces(marker=dict(line=dict(color=text_color, width=1)))
save_plot_as_html(fig2, "type_graph_2.html", "Most Apps on the playstore are␣
 ↪free, which indicates the strategy to attract users first and monetize␣
 ↪through ads or in-app purchases")
```

Figure - 3 (Rating Distribution)

[167]:
```
type_counts = apps_df['Type'].value_counts()
fig3 = px.histogram(
    apps_df,
    x = 'Rating',
    nbins=20,
    title = 'Rating Distribution',
    color_discrete_sequence=['#636EFA'],
    width = plot_width,
    height = plot_height
)
fig3.update_layout(
    plot_bgcolor = 'black',
    paper_bgcolor = 'black',
    font_color = 'white',
    title_font = title_font,
    xaxis = dict(title_font = axis_font),
```

```
    yaxis = dict(title_font = axis_font),
    margin = dict(l=10, r=10, t=30, b=10)
)
# fig3.update_traces(marker=dict(line=dict(color=text_color, width=1)))
save_plot_as_html(fig3, "Rating_Graph_3.html", "Ratings are skewed towards␣
 ↪higher values, suggesting that most apps are rated favourably by users.")
```

Figure - 4 (SENTIMENT COUNT)

[168]:
```
sentiment_counts = reviews_df['Sentiment_Score'].value_counts()
fig4 = px.bar(
    x=sentiment_counts.index,
    y=sentiment_counts.values,
    labels = {'x': 'Sentiment', 'y': 'Count'},
    title = 'Sentiment Distribution',
    color = sentiment_counts.index,
    color_discrete_sequence=px.colors.sequential.RdPu_r,
    width = plot_width,
    height = plot_height
)
fig4.update_layout(
    plot_bgcolor = plot_bg_color,
    paper_bgcolor = plot_bg_color,
    font_color = text_color,
    title_font = title_font,
    xaxis = dict(title_font = axis_font),
    yaxis = dict(title_font = axis_font),
    margin = dict(l=10, r=10, t=30, b=10)
)
fig4.update_traces(marker=dict(line=dict(color=text_color, width=1)))
save_plot_as_html(fig4, "sentiment_count.html", "Sentiments in reviews shows a␣
 ↪positive and negative feedback, with a slight lean towards positive␣
 ↪sentiments.")
```

Figure - 5 (Installs by Category)

[169]:
```
installs_by_category = apps_df.groupby('Category')['Installs'].sum().
 ↪nlargest(10)
fig5 = px.bar(
    x = installs_by_category.index,
    y = installs_by_category.values,
    orientation='h',
    labels = {'x': 'Installs', 'y': 'Category'},
    title = 'Install by Category',
    color = installs_by_category.index,
    color_discrete_sequence=px.colors.sequential.Greens,
    width = plot_width,
```

```
    height = plot_height
)
fig5.update_layout(
    plot_bgcolor = plot_bg_color,
    paper_bgcolor = plot_bg_color,
    font_color = text_color,
    title_font = title_font,
    xaxis = dict(title_font = axis_font),
    yaxis = dict(title_font = axis_font),
    margin = dict(l=10, r=10, t=30, b=10)
)
fig5.update_traces(marker=dict(line=dict(color=text_color, width=1)))
save_plot_as_html(fig5, "Install_by_Category.html", "The categories with the␣
 ↪most installs are social and communication apps, reflecting their broad␣
 ↪appeal and daily usage.")
```

Figure - 6 (Number of Updates over year)

[170]:
```
apps_df['Last Updated'] = pd.to_datetime(apps_df['Last Updated'],␣
 ↪errors='coerce')
updates_per_year = apps_df['Last Updated'].dt.year.value_counts().sort_index()

fig6 = px.line(
    x=updates_per_year.index,
    y=updates_per_year.values,
    labels={'x': 'Year', 'y': 'Number of Updates'},
    title='Number of Updates Over the Years',
    color_discrete_sequence=['#AB63FA'],
    width=plot_width,
    height=plot_height
)
fig6.update_layout(
    plot_bgcolor = plot_bg_color,
    paper_bgcolor = plot_bg_color,
    font_color = text_color,
    title_font = title_font,
    xaxis = dict(title_font=axis_font),
    yaxis = dict(title_font=axis_font),
    margin = dict(l=10, r=10, t=30, b=10)
)

save_plot_as_html(
    fig6,
    "updates_per_year.html",
    "Updates have been increasing over the years, showing that developers are␣
 ↪actively maintaining and improving their apps."
)
```

Figure - 7 (Revenue by Category Plot)

[171]:
```
revenue_by_category = apps_df.groupby('Category')['Revenue'].sum().nlargest(10)
fig7 = px.bar(
    x=revenue_by_category.index,
    y=revenue_by_category.values,
    labels={'x': 'Category', 'y': 'Revenue'},
    title='Revenue by Category',
    color=revenue_by_category.index,
    color_discrete_sequence=px.colors.sequential.Greens,
    width=plot_width,
    height=plot_height
)
fig7.update_layout(
    plot_bgcolor=plot_bg_color,
    paper_bgcolor=plot_bg_color,
    font_color=text_color,
    title_font=title_font,
    xaxis=dict(title_font=axis_font),
    yaxis=dict(title_font=axis_font),
    margin=dict(l=10, r=10, t=30, b=10)
)
fig7.update_traces(marker=dict(line=dict(color=text_color, width=1)))
save_plot_as_html(fig7, "revenue_by_category.html", "Categories such as␣
 ↪Business and Productivity lead in revenue generation, indicating their␣
 ↪monetization potential.")
```

Figure - 8 (Genre Count Plot)

[172]:
```
genre_counts = apps_df['Genres'].str.split(';', expand=True).stack().
 ↪value_counts().nlargest(10)
fig8 = px.bar(
    x=genre_counts.index,
    y=genre_counts.values,
    labels={'x': 'Genre', 'y': 'Count'},
    title='Top Genres',
    color=genre_counts.index,
    color_discrete_sequence=px.colors.sequential.OrRd,
    width=plot_width,
    height=plot_height
)
fig8.update_layout(
    plot_bgcolor=plot_bg_color,
    paper_bgcolor=plot_bg_color,
    font_color=text_color,
    title_font=title_font,
    xaxis=dict(title_font=axis_font),
    yaxis=dict(title_font=axis_font),
```

```
    margin=dict(l=10, r=10, t=30, b=10)
)
fig8.update_traces(marker=dict(line=dict(color=text_color, width=1)))
save_plot_as_html(fig8, "genres_counts.html", "Action and Casual genres are the␣
 ↪most common, reflecting users' preference for engaging and easy-to-play␣
 ↪games.")
```

Figure - 9 (Update on Rating)

[173]:
```
fig9 = px.scatter(
    apps_df,
    x='Last Updated',
    y='Rating',
    color='Type',
    title='Impact of Last Update on Rating',
    color_discrete_sequence=px.colors.qualitative.Vivid,
    width=plot_width,
    height=plot_height
)
fig9.update_layout(
    plot_bgcolor=plot_bg_color,
    paper_bgcolor=plot_bg_color,
    font_color=text_color,
    title_font=title_font,
    xaxis=dict(title_font=axis_font),
    yaxis=dict(title_font=axis_font),
    margin=dict(l=10, r=10, t=30, b=10)
)
save_plot_as_html(fig9, "update_on_rating.html", "The scatter plot shows a weak␣
 ↪correlation between the last update date and ratings, suggesting that more␣
 ↪frequent updates don't always result in better ratings.")
```

Figure - 10 (Ratings of Paid vs Free Apps)

[174]:
```
fig10 = px.box(
    apps_df,
    x='Type',
    y='Rating',
    color='Type',
    title='Ratings for Paid vs Free Apps',
    color_discrete_sequence=px.colors.qualitative.Pastel,
    width=plot_width,
    height=plot_height
)
fig10.update_layout(
    plot_bgcolor=plot_bg_color,
    paper_bgcolor=plot_bg_color,
```

```
    font_color=text_color,
    title_font=title_font,
    xaxis=dict(title_font=axis_font),
    yaxis=dict(title_font=axis_font),
    margin=dict(l=10, r=10, t=30, b=10)
)
save_plot_as_html(fig10, "ratings_paid_free.html", "Paid apps generally have␣
 ↪higher ratings compared to free apps, suggesting that users expect higher␣
 ↪quality from apps they pay for.")
```

[175]:
```
# Split plot_containers to handle the last plot properly
plot_containers_split = plot_containers.split('</div>')
if len(plot_containers_split) > 1:
    final_plot = plot_containers_split[-2] + '</div>'
else:
    final_plot = plot_containers  # Use plot_containers as default if splitting␣
 ↪isn't sufficient
```

HTML template for the dashboard

[176]:
```
dashboard_html = """
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Google Play Store Reviews Analytics</title>
    <style>
        body {{
            font-family: Arial, sans-serif;
            background-color: #333;
            color: #fff;
            margin: 0;
            padding: 0;
        }}
        .header {{
            display: flex;
            align-items: center;
            justify-content: center;
            padding: 20px;
            background-color: #444;
        }}
        .header img {{
            margin: 0 10px;
            height: 50px;
        }}
        .container {{
```

```
            display: flex;
            flex-wrap: wrap;
            justify-content: center;
            padding: 20px;
        }}
        .plot-container {{
            border: 2px solid #555;
            margin: 10px;
            padding: 10px;
            width: {plot_width}px;
            height: {plot_height}px;
            overflow: hidden;
            position: relative;
            cursor: pointer;
        }}
        .insights {{
            display: none;
            position: absolute;
            right: 10px;
            top: 10px;
            background-color: rgba(0, 0, 0, 0.7);
            padding: 5px;
            border-radius: 5px;
            color: #fff;
        }}
        .plot-container:hover .insights {{
            display: block;
        }}
    </style>
    <script>
        function openPlot(filename) {{
            window.open(filename, '_blank');
        }}
    </script>
</head>
<body>
    <div class="header">
        <img src="https://logos-world.net/wp-content/uploads/2020/09/
 ↪Google-Logo-700x394.png" alt="Google Logo">
        <h1>Google Play Store Reviews Analytics</h1>
        <img src="https://upload.wikimedia.org/wikipedia/commons/thumb/7/78/
 ↪Google_Play_Store_badge_EN.svg/1024px-Google_Play_Store_badge_EN.svg.png"␣
 ↪alt="Google Play Store Logo">
    </div>
    <div class="container">
        {plots}
    </div>
```

```
</body>
</html>
"""
```

[177]:
```
# Use these containers to fill in your dashboard HTML
final_html = dashboard_html.format(plots=plot_containers,␣
 ↪plot_width=plot_width, plot_height=plot_height)

# Save the final dashboard to an HTML file
dashboard_path = os.path.join(html_files_path, "dashboard.html")
with open(dashboard_path, "w", encoding="utf-8") as f:
    f.write(final_html)

# Automatically open the generated HTML file in a web browser
webbrowser.open('file://' + os.path.realpath(dashboard_path))
```

[177]: True

Author: Ayan Jawaid

Email: ayaan.jawaid786@gmail.com