Project Title	**Unlocking YouTube Channel Performance Secrets**
Tools	Visual Studio code / jupyter notebook
Domain	Data Analyst
Project Difficulties level	intermediate

Dataset : Dataset is available in the given link. You can download it at your convenience.

[Click here to download data set]

About Dataset

This dataset provides an in-depth look at YouTube video analytics, capturing key metrics related to video performance, audience engagement, revenue generation, and viewer behavior. Sourced from real video data, it highlights how variables like video duration, upload time, and ad impressions contribute to monetization and audience retention. This dataset is ideal for data analysts, content creators, and marketers aiming to uncover trends in viewer engagement, optimize content strategies, and maximize ad revenue. Inspired by the evolving landscape of digital content, it serves as a resource for understanding the impact of YouTube metrics on channel growth and content reach.

Video Details: Columns like Video Duration, Video Publish Time, Days Since Publish, Day of Week.

Revenue Metrics: Includes Revenue per 1000 Views (USD), Estimated Revenue (USD), Ad Impressions, and various ad revenue sources (e.g., AdSense, DoubleClick).

Engagement Metrics: Metrics such as Views, Likes, Dislikes, Shares, Comments, Average View Duration, Average View Percentage (%), and Video Thumbnail CTR (%).

Audience Data: Data on New Subscribers, Unsubscribes, Unique Viewers, Returning Viewers, and New Viewers.

Monetization & Transaction Metrics: Details on Monetized Playbacks, Playback-Based CPM, YouTube Premium Revenue, and transactions like Orders and Total Sales Volume (USD).

Example: You can get the basic idea how you can create a project from here

Major Machine Learning Project: Unlocking YouTube Channel Performance Secrets Analysis

This project aims to analyze YouTube channel performance by leveraging extensive metrics and using Machine Learning techniques to uncover patterns, trends, and actionable insights. We'll focus on **Exploratory Data Analysis (EDA)**, **data visualization**, and developing a **predictive model** to estimate revenue or subscribers based on the provided dataset.

Step-by-Step Workflow

---

1. Import Libraries

python

code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

---

2. Load and Explore the Dataset

python

code

```
# Load the dataset
data = pd.read_csv("youtube_channel_data.csv")

# Display basic information about the dataset
print(data.info())

# Check for null values
print(data.isnull().sum())

# Preview the dataset
data.head()
```

---

3. Data Cleaning

Handle Missing Values:

python

code

```
# Fill or drop null values
data = data.dropna()  # Drop rows with missing values (for
simplicity)
```

-

Convert Duration:

python

code

```
# Convert 'Video Duration' into seconds
import isodate
data['Video Duration'] = data['Video Duration'].apply(lambda x:
isodate.parse_duration(x).total_seconds())
```

-

---

4. Exploratory Data Analysis (EDA)

Analyze relationships:

python

code

```
# Pairplot to visualize relationships
sns.pairplot(data[['Revenue per 1000 Views (USD)', 'Views',
'Subscribers', 'Estimated Revenue (USD)']])
plt.show()
```

-

Correlation Heatmap:

python

code

```
plt.figure(figsize=(12, 8))
sns.heatmap(data.corr(), annot=True, fmt=".2f",
```

```
cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()
```

-

Top Performers by Revenue:

python

code

```
top_videos = data.sort_values(by='Estimated Revenue (USD)',
ascending=False).head(10)
print(top_videos[['ID', 'Estimated Revenue (USD)', 'Views',
'Subscribers']])
```

-

---

5. Feature Engineering

Create new features:

python

code

```
# Create revenue per view
data['Revenue per View'] = data['Estimated Revenue (USD)'] /
data['Views']

# Create engagement rate
data['Engagement Rate'] = (data['Likes'] + data['Shares'] +
data['Comments']) / data['Views'] * 100
```

-

---

6. Data Visualization

Revenue Distribution:

python

code

```
plt.figure(figsize=(10, 6))
sns.histplot(data['Estimated Revenue (USD)'], bins=50,
kde=True, color='green')
plt.title("Revenue Distribution")
plt.xlabel("Revenue (USD)")
plt.ylabel("Frequency")
plt.show()
```

-

Revenue vs Views:

python

code

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x=data['Views'], y=data['Estimated Revenue
(USD)'], alpha=0.7)
plt.title("Revenue vs Views")
plt.xlabel("Views")
plt.ylabel("Revenue (USD)")
plt.show()
```

-

---

7. Predictive Model: Estimate Revenue

Prepare Data:

python

code

```
# Select features and target
features = ['Views', 'Subscribers', 'Likes', 'Shares',
'Comments', 'Engagement Rate']
target = 'Estimated Revenue (USD)'

X = data[features]
y = data[target]

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

-

Train Random Forest Regressor:

python

code

```
model = RandomForestRegressor(n_estimators=100,
random_state=42)
model.fit(X_train, y_train)
```

```
# Predict on test data
y_pred = model.predict(X_test)
```

-

Evaluate the Model:

python

code

```
# Calculate performance metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared: {r2:.2f}")
```

-

---

8. Insights and Recommendations

Use visualizations and feature importance to derive insights:

python

code

```
# Feature Importance
importances = model.feature_importances_
feature_importance_df = pd.DataFrame({'Feature': features,
'Importance': importances})
feature_importance_df =
```

```
feature_importance_df.sort_values(by='Importance',
ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature',
data=feature_importance_df)
plt.title("Feature Importance")
plt.show()
```

-

---

9. Deployment and Presentation

- Summarize findings:
  - Highlight top revenue drivers (e.g., views, engagement rate).
  - Identify underperforming areas (e.g., low CPM or low engagement).

Export model:

python

code

```
import joblib
joblib.dump(model, 'youtube_revenue_predictor.pkl')
```

-

Example: You can get the basic idea how you can create a project from here

Sample code with output

In the world of YouTube, where every click and view can translate into revenue, understanding the nuances of channel performance is crucial. This dataset offers a treasure trove of insights into YouTube channel analytics, from video duration to revenue streams. Let's dive into the data and see what stories it has to tell. If you find this notebook useful, please consider upvoting it.

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
# Load the dataset
df =
pd.read_csv('/kaggle/input/youtube-channel-performance-analytic
s/youtube_channel_real_performance_analytics.csv')
df.head()
```

Out[2]:

	ID	Video Duration	Video Publish Time	Days Since Publish	Day	Month	Year	Day of Week	Revenue per 1000 Views (USD)	Monetized Playbacks (Estimate)	...	Watched (Not Skipped) (%)	Feed Impressions	Average View Percentage (%)	Average View Duration	Views	Watch Time (hours)	Subscribers	Estimated Revenue (USD)	Impressions	Video Thumbnail CTR (%)
0	0	20	20	0	2	6	20	Thurs	0.0	723.	..	0.	0.0	40.	81.	23	53	51.	0.56	4111	27.6

		1.0	16-06-0200:00:00				16	day	24	0	.	0		38	0	531.0	3.16336	0	1	8.0	6
1	1	391.0	2016-06-1000:00:00	8	10	6	2016	Friday	0.056	727.0	...	0.0	0.0	39.85	156.0	11478.0	500.5628	33.0	0.648	41627.0	5.85

2	2	133.0	2016-06-14 00:00:00	4	14	6	2016	Tuesday	0.014	76.0	...	0.0	0.0	30.88	41.0	6153.0	70.72877	8.0	0.089	38713.0	7.07
3	3	14.0	2016-06-29 00:0	15	29	6	2016	Wednesday	0.004	18.0	...	0.0	0.0	103.05	14.0	4398.0	17.6251	2.0	0.017	35245.0	5.60

			0:00																		
4	4	45.0	2016-07-01 00:00:00	2	1	7	2016	Friday	0.000	0.0	...	0.0	0.0	55.70	25.0	14659.0	104.3341	28.0	0.000	46218.0	8.62

5 rows × 70 columns

Data Overview

Let's take a look at the basic information about the dataset to understand its structure and contents.

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 364 entries, 0 to 363
Data columns (total 70 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   ID                                  364 non-null    int64
 1   Video Duration                      364 non-null
float64
 2   Video Publish Time                  364 non-null    object
 3   Days Since Publish                  364 non-null    int64
 4   Day                                 364 non-null    int64
 5   Month                               364 non-null    int64
 6   Year                                364 non-null    int64
 7   Day of Week                         364 non-null    object
 8   Revenue per 1000 Views (USD)        364 non-null
float64
 9   Monetized Playbacks (Estimate)      364 non-null
float64
 10  Playback-Based CPM (USD)            364 non-null
float64
 11  CPM (USD)                           364 non-null
float64
 12  Ad Impressions                      364 non-null
```

```
float64
 13  Estimated AdSense Revenue (USD)     364 non-null
float64
 14  DoubleClick Revenue (USD)           364 non-null
float64
 15  YouTube Ads Revenue (USD)           364 non-null
float64
 16  Watch Page Ads Revenue (USD)        364 non-null
float64
 17  YouTube Premium (USD)               364 non-null
float64
 18  Transaction Revenue (USD)           364 non-null
float64
 19  Transactions                        364 non-null
float64
 20  Revenue from Transactions (USD)     364 non-null
float64
 21  Reactions                           364 non-null
float64
 22  Chat Messages Count                 364 non-null
float64
 23  Reminders Set                       364 non-null
float64
 24  Stream Hours                        364 non-null
float64
```

```
 25  Remix Views                           364 non-null
float64
 26  Remix Count                           364 non-null
float64
 27  Subscribers from Posts                364 non-null
float64
 28  New Comments                          364 non-null
float64
 29  Shares                                364 non-null
float64
 30  Like Rate (%)                         364 non-null
float64
 31  Dislikes                              364 non-null
float64
 32  Likes                                 364 non-null
float64
 33  Unsubscribes                          364 non-null
float64
 34  New Subscribers                       364 non-null
float64
 35  Returned Items (USD)                  364 non-null
float64
 36  Unconfirmed Commissions (USD)         364 non-null
float64
 37  Approved Commissions (USD)            364 non-null
```

```
float64
 38  Orders                                  364 non-null
float64
 39  Total Sales Volume (USD)                364 non-null
float64
 40  End Screen Click-Through Rate (%)       364 non-null
float64
 41  End Screen Impressions                  364 non-null
float64
 42  End Screen Clicks                       364 non-null
float64
 43  Teaser Click-Through Rate (%)           364 non-null
float64
 44  Teaser Impressions                      364 non-null
float64
 45  Teaser Clicks                           364 non-null
float64
 46  Card Click-Through Rate (%)             364 non-null
float64
 47  Card Impressions                        364 non-null
float64
 48  Card Clicks                             364 non-null
float64
 49  Views per Playlist Start                364 non-null
float64
```

```
 50  Playlist Views                         364 non-null
float64
 51  Playlist Watch Time (hours)            364 non-null
float64
 52  Clip Watch Time (hours)                364 non-null
float64
 53  Clip Views                             364 non-null
float64
 54  YouTube Premium Watch Time (hours)  364 non-null
float64
 55  YouTube Premium Views                  364 non-null
float64
 56  Returning Viewers                      364 non-null
float64
 57  New Viewers                            364 non-null
float64
 58  Average Views per User                 364 non-null
float64
 59  Unique Viewers                         364 non-null
float64
 60  Watched (Not Skipped) (%)              364 non-null
float64
 61  Feed Impressions                       364 non-null
float64
 62  Average View Percentage (%)            364 non-null
```

```
float64
 63  Average View Duration                  364 non-null
float64
 64  Views                                  364 non-null
float64
 65  Watch Time (hours)                     364 non-null
float64
 66  Subscribers                            364 non-null
float64
 67  Estimated Revenue (USD)                364 non-null
float64
 68  Impressions                            364 non-null
float64
 69  Video Thumbnail CTR (%)                364 non-null
float64
dtypes: float64(63), int64(5), object(2)
memory usage: 199.2+ KB
```

Data Cleaning and Preprocessing

Before diving into analysis, let's ensure the data is clean and ready for exploration.

In [4]:

```
# Check for missing values
df.isnull().sum()
```

Out[4]:

```
ID                         0
Video Duration             0
Video Publish Time         0
Days Since Publish         0
Day                        0
                          ..
Watch Time (hours)         0
Subscribers                0
Estimated Revenue (USD)    0
Impressions                0
Video Thumbnail CTR (%)    0

Length: 70, dtype: int64
```

In [5]:

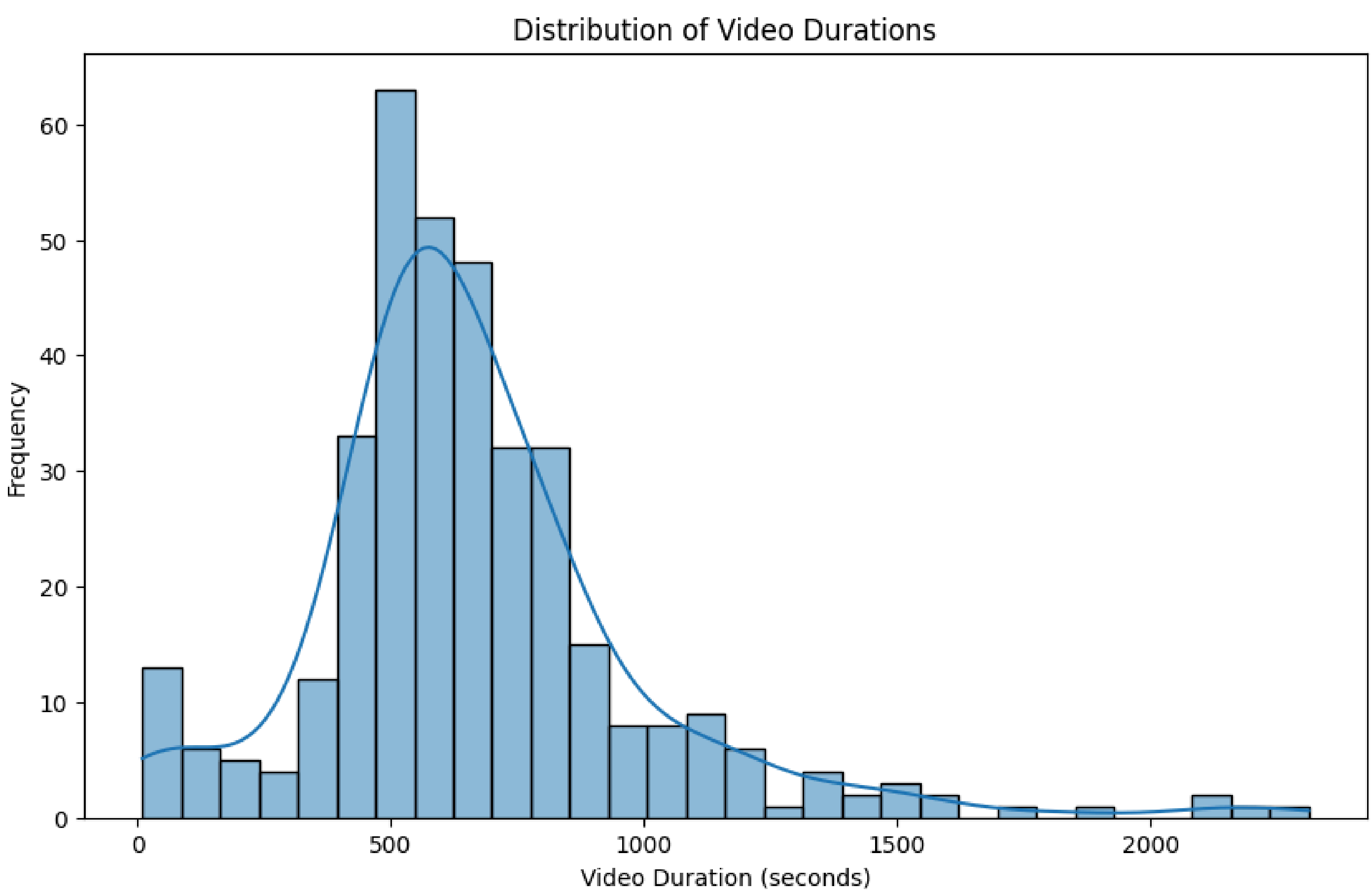
```
# Convert 'Video Publish Time' to datetime format
df['Video Publish Time'] = pd.to_datetime(df['Video Publish Time'])
```

Exploratory Data Analysis

Let's explore the data to uncover patterns and insights.

In [6]:

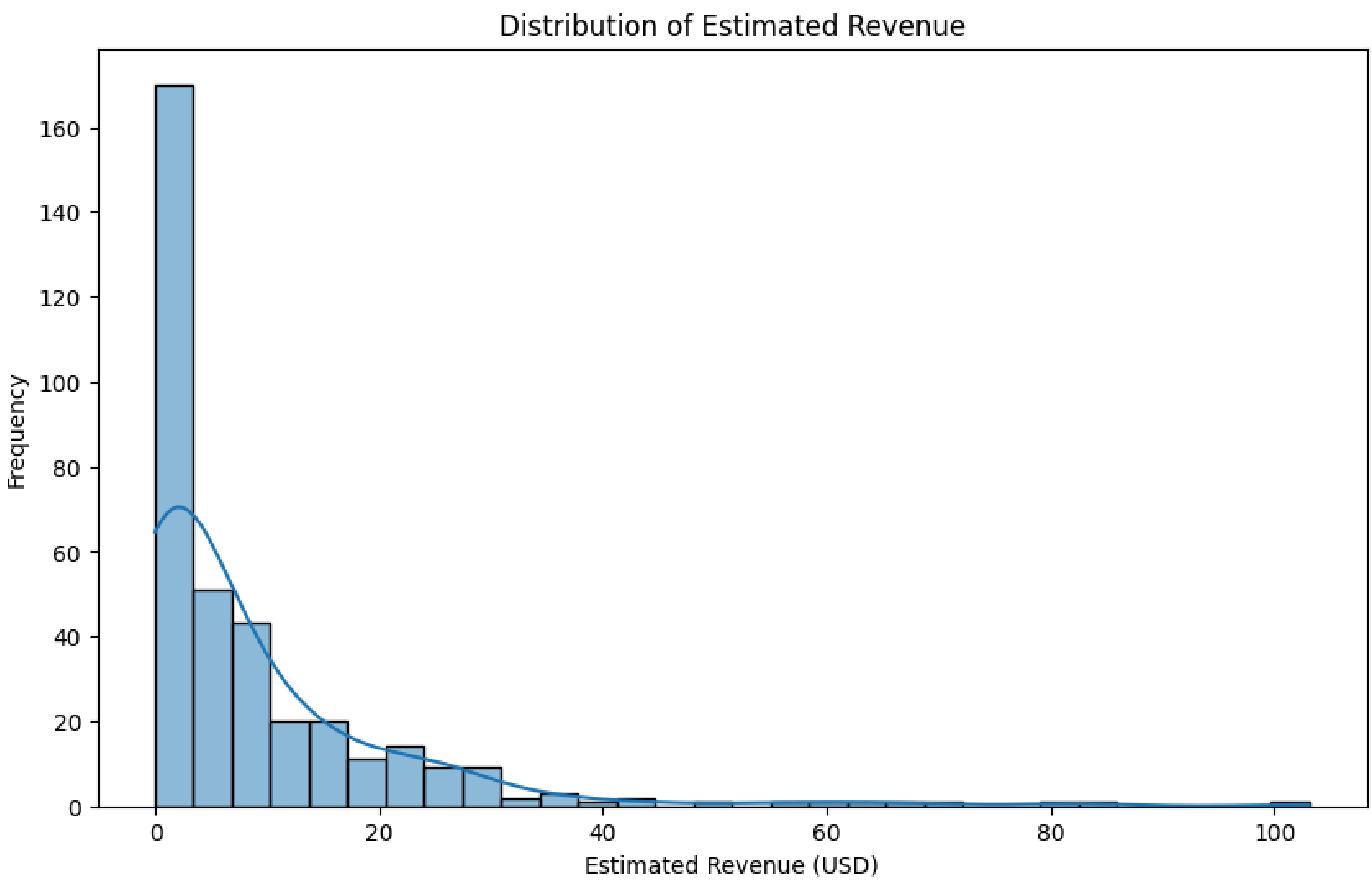
```
# Distribution of video durations
plt.figure(figsize=(10, 6))
sns.histplot(df['Video Duration'], bins=30, kde=True)
plt.title('Distribution of Video Durations')
plt.xlabel('Video Duration (seconds)')
plt.ylabel('Frequency')
plt.show()
```



In [7]:

```
# Revenue distribution
```

```python
plt.figure(figsize=(10, 6))
sns.histplot(df['Estimated Revenue (USD)'], bins=30, kde=True)
plt.title('Distribution of Estimated Revenue')
plt.xlabel('Estimated Revenue (USD)')
plt.ylabel('Frequency')
plt.show()
```



Correlation Analysis

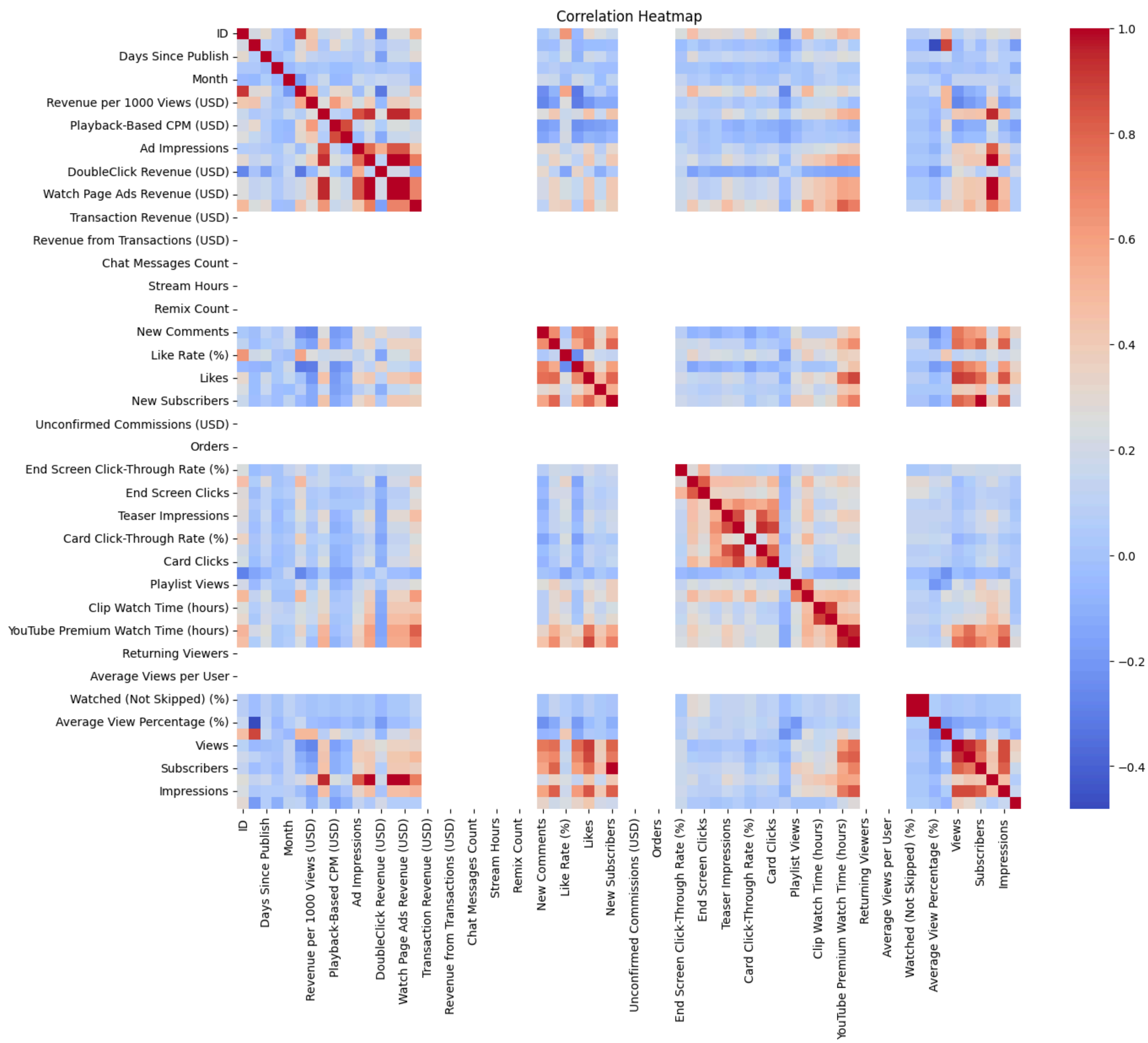
Let's examine the correlation between numeric features to identify potential relationships.

In [8]:

```
# Select only numeric columns
numeric_df = df.select_dtypes(include=[np.number])

# Compute the correlation matrix
corr = numeric_df.corr()

# Plot the heatmap
plt.figure(figsize=(15, 12))
sns.heatmap(corr, cmap='coolwarm', annot=False, fmt='.2f')
plt.title('Correlation Heatmap')
plt.show()
```



Predictive Modeling

Given the richness of this dataset, let's attempt to predict the 'Estimated Revenue (USD)' using other features.

In [9]:

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
```

```python
from sklearn.metrics import mean_squared_error

# Define features and target variable
X = numeric_df.drop(columns=['Estimated Revenue (USD)'])
y = numeric_df['Estimated Revenue (USD)']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Initialize and train the model
model = RandomForestRegressor(n_estimators=100,
random_state=42)
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Calculate the prediction accuracy
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
rmse
```

Out[9]:

```
0.45593992214488205
```

linkcode

Discussion

In this notebook, we explored a comprehensive YouTube channel performance dataset. We visualized key metrics, examined correlations, and built a predictive model for estimating revenue. The Random Forest model provided a reasonable prediction accuracy, but there's always room for improvement. Future analysis could explore feature engineering, hyperparameter tuning, or even different modeling approaches to enhance prediction performance. If you found this analysis insightful, please consider upvoting this notebook.

Reference link