Laptop Price Prediction

Presented by: Yash Sahu

Table of Contents

Introduction & Problem Statement

Introduction

- Laptop prices vary based on multiple factors like brand, specifications, and market trends.
- Predicting laptop prices helps buyers make informed decisions and assists businesses in pricing strategies.

Problem Statement

- Manually determining a laptop's fair price is complex due to diverse specifications.
- The goal is to build a machine learning model that predicts laptop prices accurately based on key features.

Dataset Description

Overview

- The dataset contains various features influencing laptop prices.
- Key attributes include Brand, Processor, RAM, Storage, Screen Size, GPU, Operating System, and Price.

Data Insights

- There are total 1275 rows and 23 columns.
- There is no any null values in dataset.

Dataset Description

```
df.head()
```

	Company	Product	TypeName	Inches	Ram	OS	Weight	Price_euros	Screen	ScreenW	...	RetinaDisplay	CPU_company	CPU_freq	CPU_model	PrimaryStorage	SecondaryStorage	PrimaryStorageType	SecondaryStorageType	GPU_company	GPU_model
0	Apple	MacBook Pro	Ultrabook	13.3	8	macOS	1.37	1339.69	Standard	2560	...	Yes	Intel	2.3	Core i5	128	0	SSD	No	Intel	Iris Plus Graphics 640
1	Apple	Macbook Air	Ultrabook	13.3	8	macOS	1.34	898.94	Standard	1440	...	No	Intel	1.8	Core i5	128	0	Flash Storage	No	Intel	HD Graphics 6000
2	HP	250 G6	Notebook	15.6	8	No OS	1.86	575.00	Full HD	1920	...	No	Intel	2.5	Core i5 7200U	256	0	SSD	No	Intel	HD Graphics 620
3	Apple	MacBook Pro	Ultrabook	15.4	16	macOS	1.83	2537.45	Standard	2880	...	Yes	Intel	2.7	Core i7	512	0	SSD	No	AMD	Radeon Pro 455
4	Apple	MacBook Pro	Ultrabook	13.3	8	macOS	1.37	1803.60	Standard	2560	...	Yes	Intel	3.1	Core i5	256	0	SSD	No	Intel	Iris Plus Graphics 650

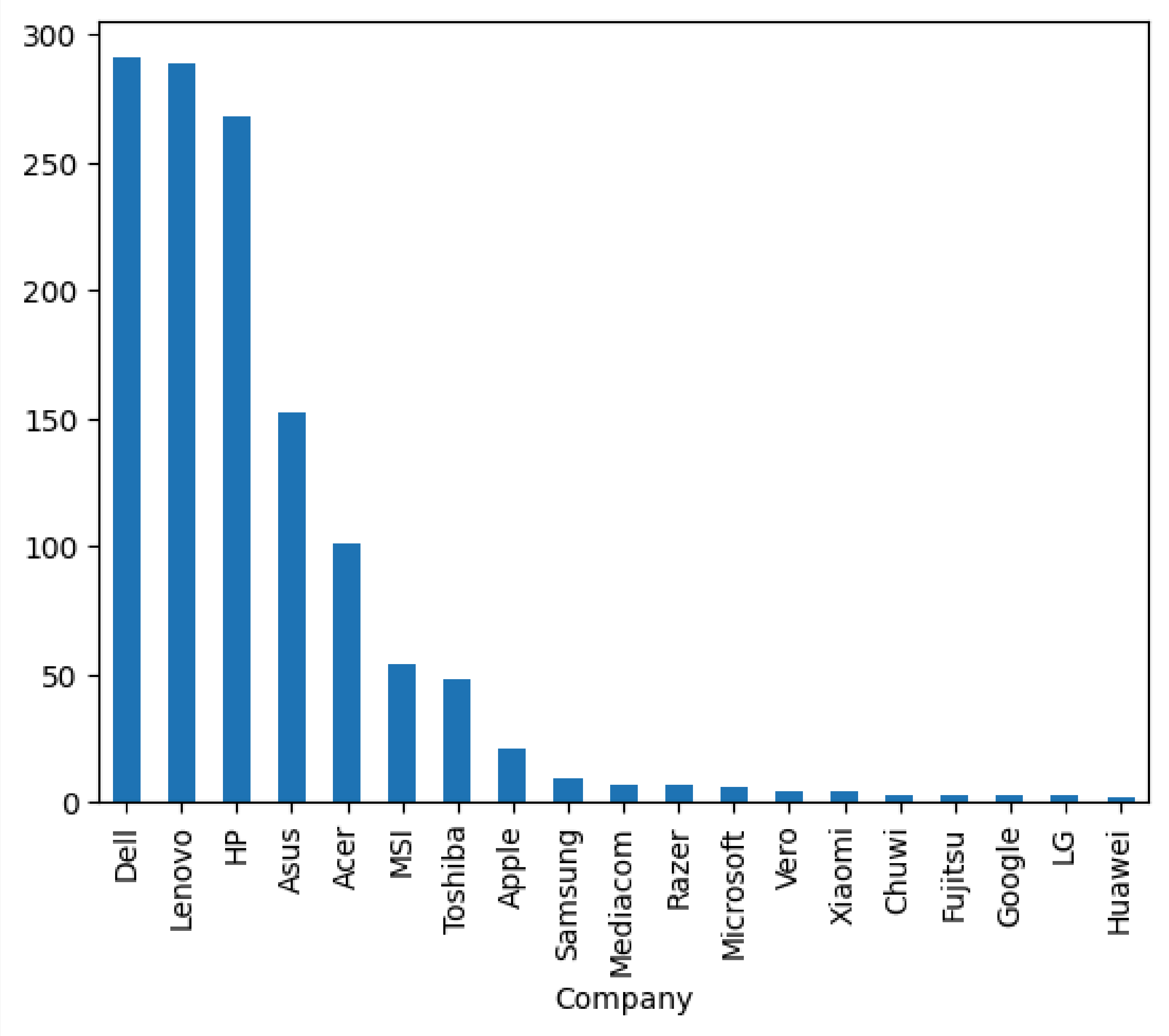
5 rows × 23 columns

This is the output of df.head() first five rows of dataset.

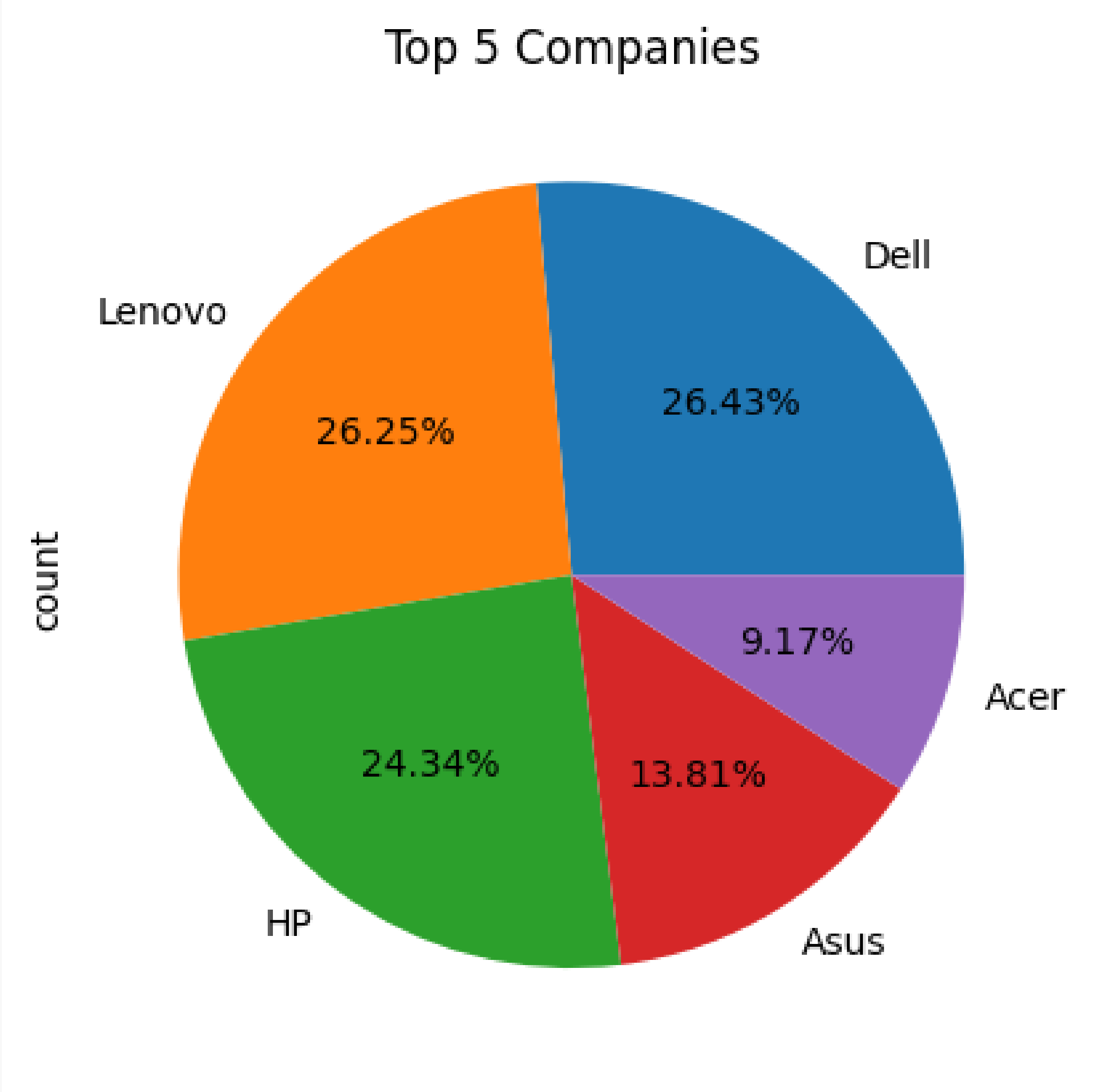
```
[ ]  df.columns

Index(['Company', 'Product', 'TypeName', 'Inches', 'Ram', 'OS', 'Weight',
       'Price_euros', 'Screen', 'ScreenW', 'ScreenH', 'Touchscreen',
       'IPSpanel', 'RetinaDisplay', 'CPU_company', 'CPU_freq', 'CPU_model',
       'PrimaryStorage', 'SecondaryStorage', 'PrimaryStorageType',
       'SecondaryStorageType', 'GPU_company', 'GPU_model'],
      dtype='object')
```

List of columns in dataset.

Data Visualization

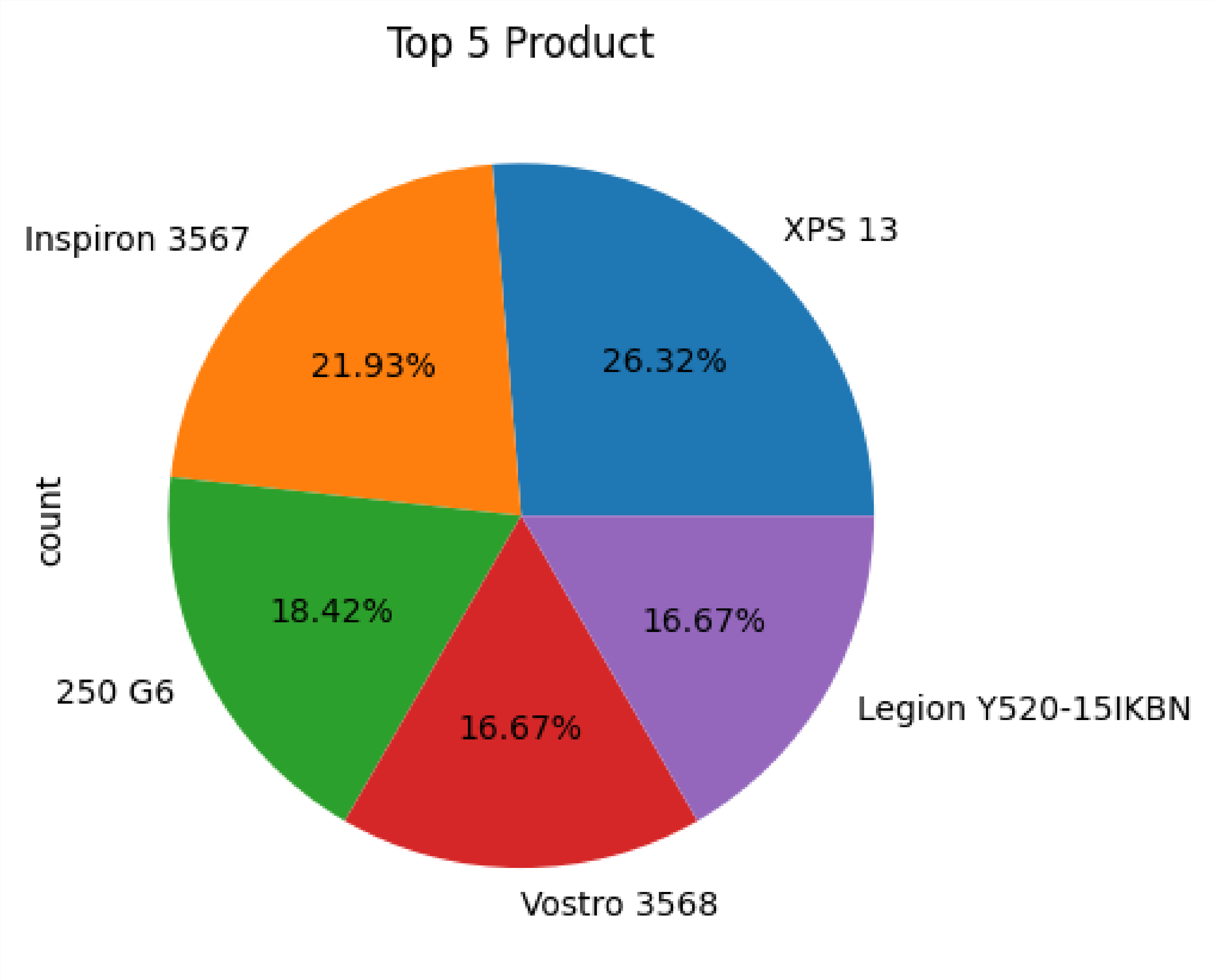


Laptop Company Distribution

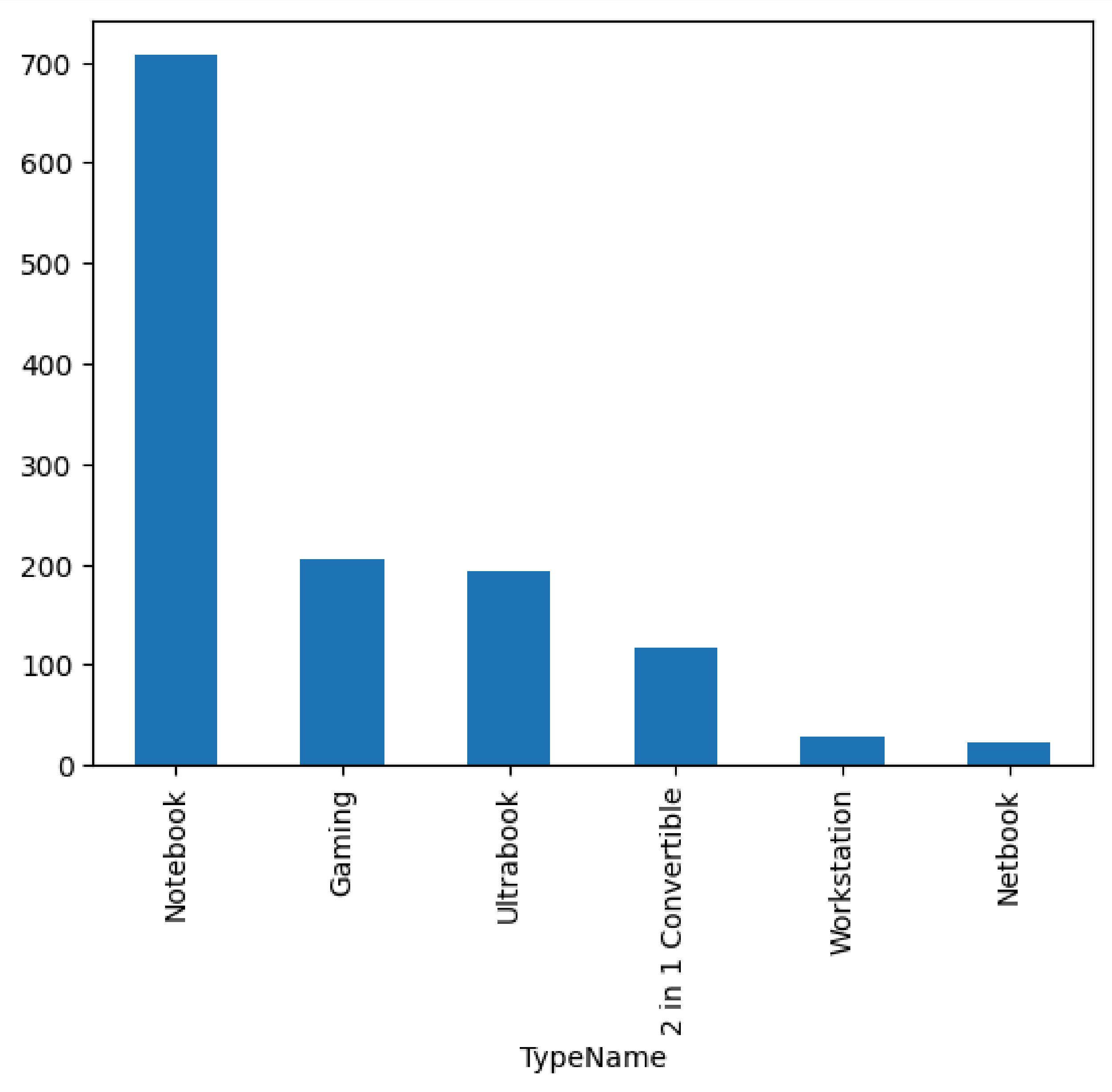


Top 5 Companies

Data Visualization

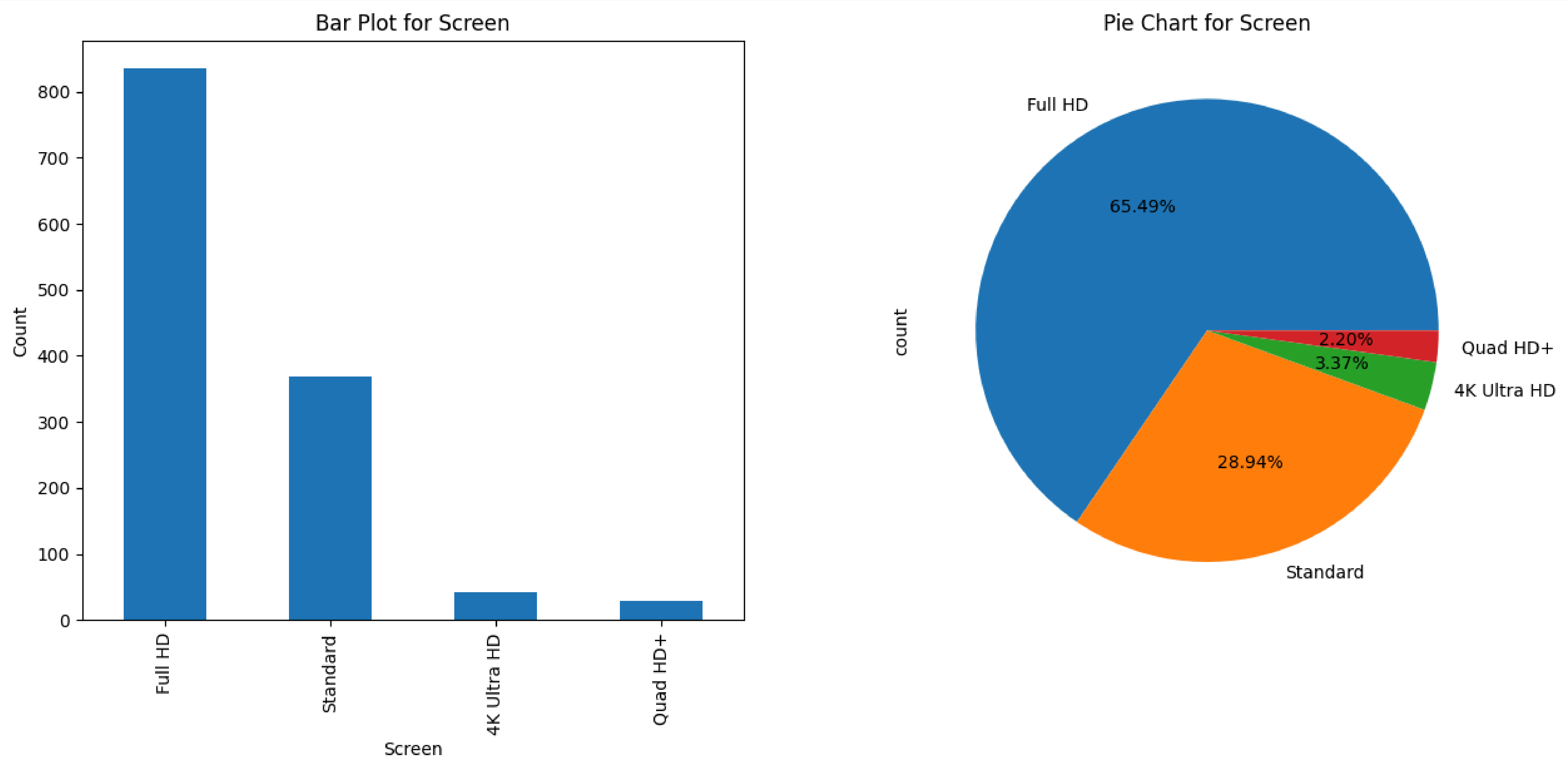


Top 5 Products



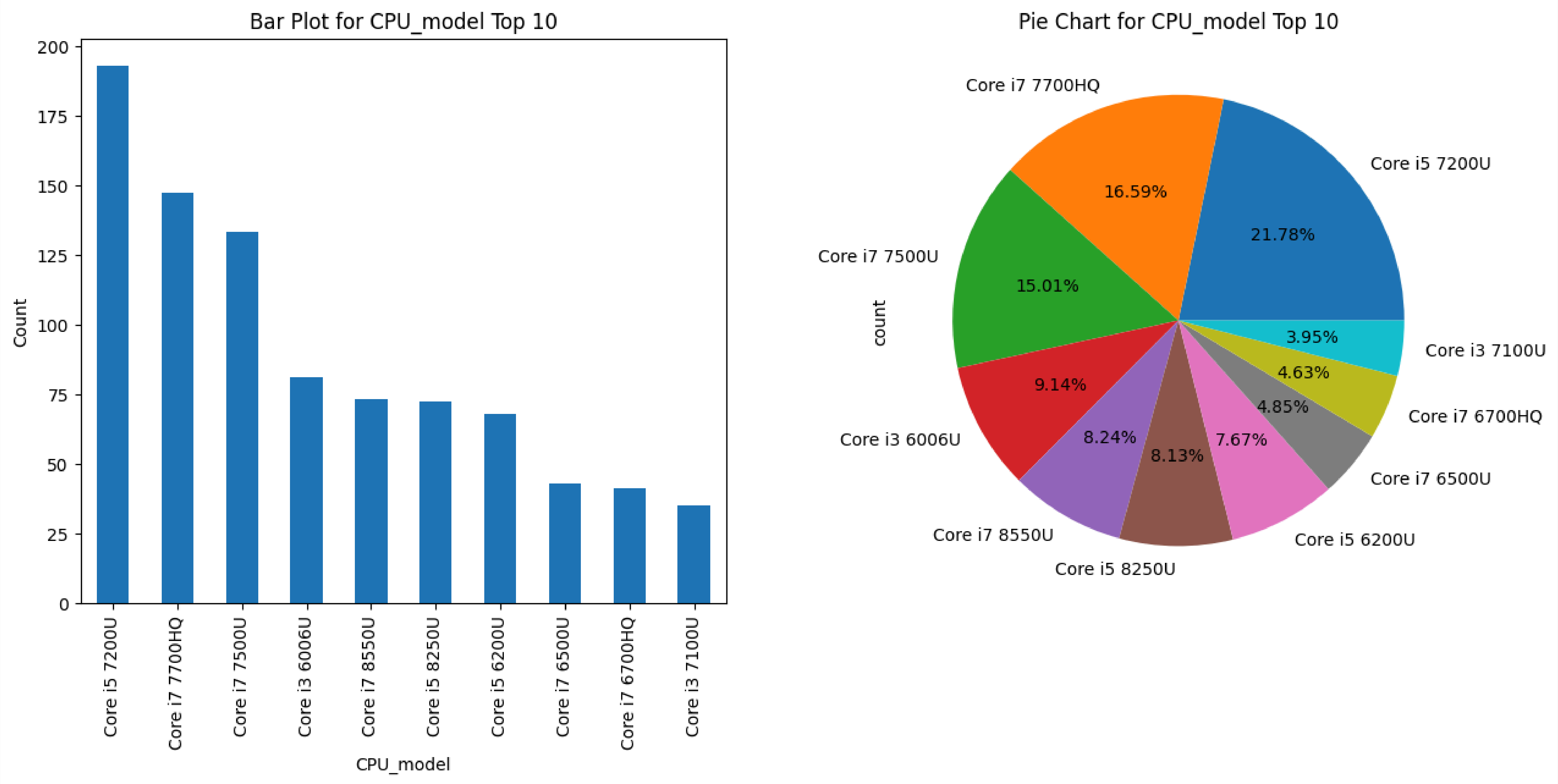
Barchart for Laptop Type

Data Visualization



Plots for Screen feature

Data Visualization



Plots for CPU Model feature

Techniques Used

Introduction

experimented with multiple regression algorithms to predict laptop prices and evaluated their performance. The models tested include:

- Linear Regression – Basic model, but it struggled with complex relationships.
- Decision Tree Regressor – Performed better but prone to overfitting.
- Random Forest Regressor – Achieved the best performance.
- XGBoost Regressor – Showed good results but slightly lower than Random Forest.

Best Model Performance:

- After testing, Random Forest Regressor provided the highest accuracy:
- **R^2 Score on Training Data: 0.98**
- **R^2 Score on Test Data: 0.87**

Techniques Used

Model Evaluation Metrics

- R^2 Score – Measures model accuracy.
- Mean Squared Error (MSE) – Determines prediction errors.

Model Training & Evaluation

Model Training Process:

1. **Data Splitting:**
- Dataset was split into 80% training data and 20% test data
2. **Feature Engineering & Preprocessing:**
- One-Hot Encoding for categorical variables (e.g., Brand, OS).
- Feature Scaling applied where necessary.
3. **Model Training:**
- Tested multiple algorithms (Linear Regression, Decision Tree, XGBoost, etc.).
- Random Forest Regressor performed the best.

Performance Analysis

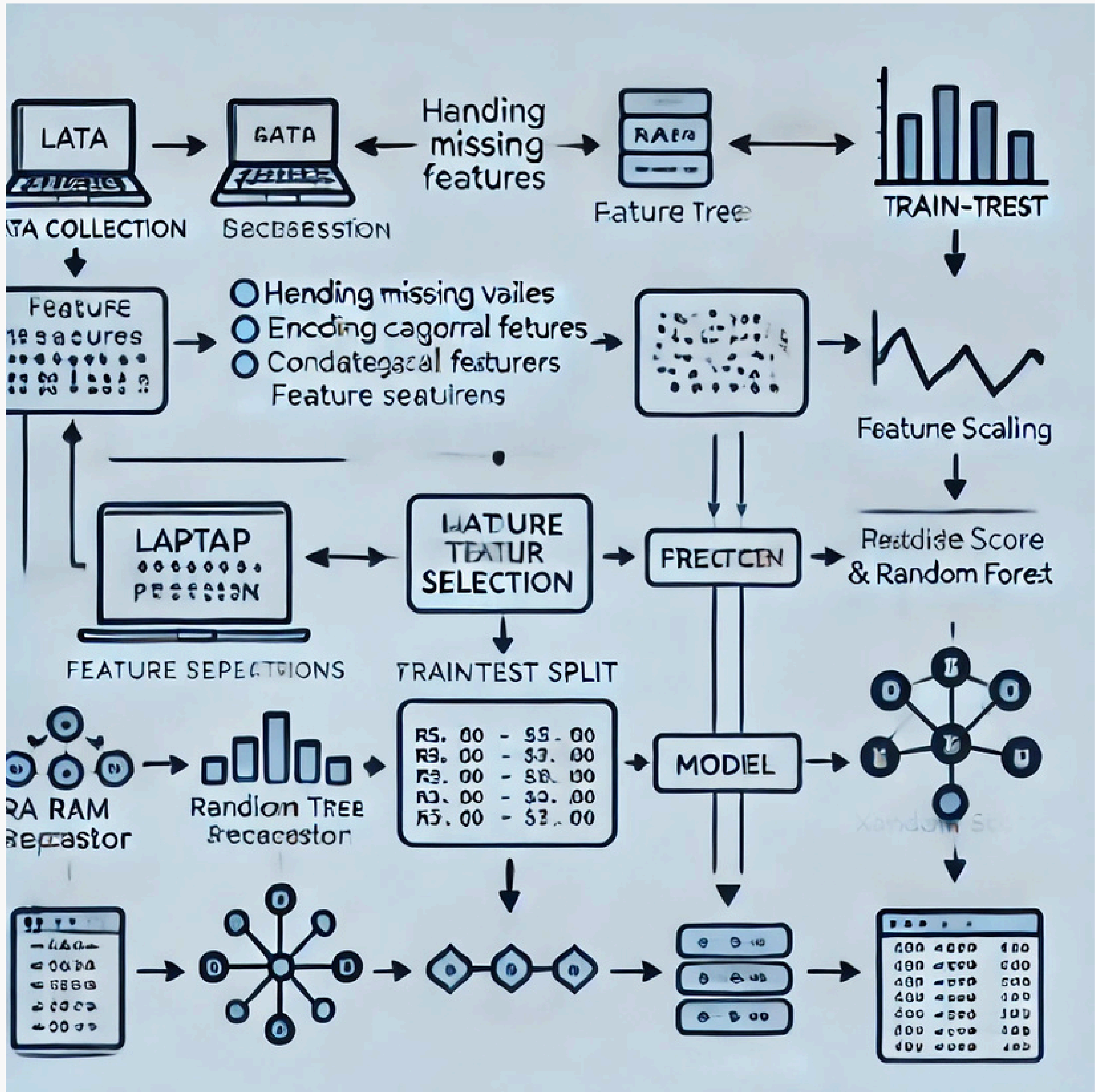
- **High training accuracy (0.98)** suggests a well-fitted model.
- **Test accuracy (0.87)** indicates good generalization.

Model Training & Evaluation

Workflow Steps

1. Data Collection – Gather laptop specifications & price dataset.
2. Data Preprocessing – Handle missing values, apply encoding, and feature scaling.
3. Feature Selection – Choose key features affecting price (RAM, Processor, Storage, etc.).
4. Train-Test Split – Split data into training (80%) and testing (20%).
5. Model Selection & Training – Train multiple models (Linear Regression, Decision Tree, Random Forest, XGBoost).
6. Model Evaluation – Compare R^2 score & MSE, select the best model.
7. Prediction & Output – Use the trained model to predict laptop prices.
8. Deployment (if applicable) – Integrate model into a web app (Flask/Streamlit).

Model Training & Evaluation



Workflow Diagram

Web Application

Web Application

Framework Used: Streamlit

- Developed an interactive web application for laptop price prediction using Streamlit.
- Simple and lightweight UI for real-time predictions.

Workflow of Web App

1. User enters laptop details in the input form.
2. Model processes the input and makes a prediction.
3. Predicted price is displayed instantly.

Output & Results

Web Application

- The **Random Forest Regressor** was the best-performing model.
- **R^2 Score:**
- Training Data: 0.98
- Test Data: 0.87

Observations

- Model predicts prices with high accuracy.
- Some slight variations due to feature importance & dataset limitations.
- Further tuning could improve generalization.

Thank You!

OCD Patient Dataset: Demographics & Clinical Data

Presented by: Yash Sahu

Table of Contents

Introduction & Problem Statement

Introduction

- OCD is a chronic disorder characterized by persistent obsessions and compulsions.
- The dataset of 1,500 patients includes demographics, symptom duration, Y-BOCS scores, comorbidities, and treatments to analyze OCD patterns.

Problem Statement

- Identifying risk factors and patterns in OCD.
- Understanding the impact of comorbidities like depression and anxiety.
- Evaluating treatment effectiveness based on medications.

Dataset Description

Overview

- The dataset contains 1,500 OCD patients, covering demographics, symptom duration, Y-BOCS scores, comorbidities, and medications.
- Key attributes include age, gender, ethnicity, marital status, obsession/compulsion types, and treatment history.

Data Insights

- Total Records: 1,500 rows and 17 columns.
- There are 248 null values in Previous Diagnoses feature and 386 null values in Medications feature.

Dataset Description

```python
df.head()
```

✓ 0.0s Python

	Patient ID	Age	Gender	Ethnicity	Marital Status	Education Level	OCD Diagnosis Date	Duration of Symptoms (months)	Previous Diagnoses	Family History of OCD	Obsession Type	Compulsion Type	Y-BOCS Score (Obsessions)	Y-BOCS Score (Compulsions)	Depressi Diagno
0	1018	32	Female	African	Single	Some College	2016-07-15	203	MDD	No	Harm-related	Checking	17	10	
1	2406	69	Male	African	Divorced	Some College	2017-04-28	180	NaN	Yes	Harm-related	Washing	21	25	
2	1188	57	Male	Hispanic	Divorced	College Degree	2018-02-02	173	MDD	No	Contamination	Checking	3	4	
3	6200	27	Female	Hispanic	Married	College Degree	2014-08-25	126	PTSD	Yes	Symmetry	Washing	14	28	
						High	2022-02-								

This is the output of df.head() first five rows of dataset.

```python
### list of columns
df.columns
```

✓ 0.0s

```
Index(['Patient ID', 'Age', 'Gender', 'Ethnicity', 'Marital Status',
       'Education Level', 'OCD Diagnosis Date',
       'Duration of Symptoms (months)', 'Previous Diagnoses',
       'Family History of OCD', 'Obsession Type', 'Compulsion Type',
       'Y-BOCS Score (Obsessions)', 'Y-BOCS Score (Compulsions)',
       'Depression Diagnosis', 'Anxiety Diagnosis', 'Medications'],
      dtype='object')
```

Techniques Used

Introduction

- **Exploratory Data Analysis (EDA):** Identified trends in OCD severity, comorbidities, and treatment effectiveness.
- **Machine Learning Models:** Tested classification models for predicting OCD severity and treatment outcomes.

Best Model Performance:

- After testing, AdaBoostClassifier provided the highest accuracy:
- **Accuracy Score: 0.54**

Techniques Used

Model Evaluation Metrics

- **Data Preprocessing:** Handled missing values, performed encoding, and feature scaling.
- **Model Performance:** Evaluated accuracy using metrics like Accuracy Score, Confusion Matrix, F1-score, Precision and Recall scores for classification tasks.

Model Training & Evaluation

Model Training Process:

1. **Data Splitting:**

- Dataset was split into 80% training data and 20% test data

2. **Feature Engineering & Preprocessing:**

- One-Hot Encoding applied to categorical variables (e.g., Gender, Ethnicity, Obsession Type).
- Feature Scaling performed where necessary (e.g., Y-BOCS Scores, Duration of Symptoms).

3. **Model Training:**

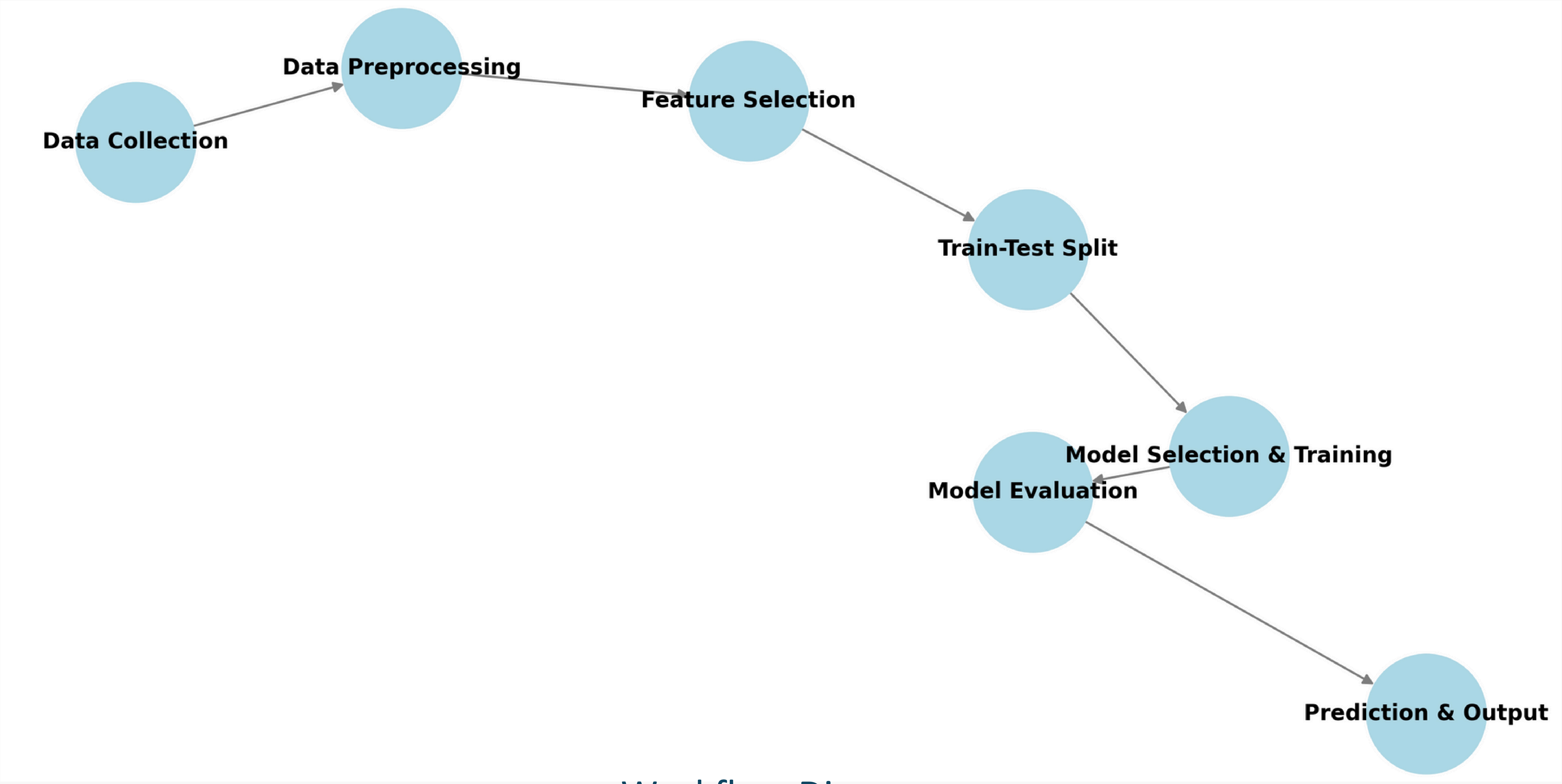
- Tested multiple algorithms (Logistic Regression, Decision Tree, Random Forest, XGBoost etc).
- Ada Boost Classifier performed the best in predicting OCD severity and treatment outcomes.

Model Training & Evaluation

Workflow Steps

1. Data Collection – Gather OCD patient demographics and clinical data.
2. Data Preprocessing – Handle missing values, apply encoding for categorical variables, and scale numerical features.
3. Feature Selection – Identify key features affecting OCD severity (e.g., Y-BOCS scores, comorbidities, medication use).
4. Train-Test Split – Split dataset into 80% training and 20% testing for model evaluation.
5. Model Selection & Training – Train multiple models (Logistic Regression, Decision Tree, Random Forest, XGBoost etc).
6. Model Evaluation – Compare accuracy, F1-score, and confusion matrix to select the best model.
7. Prediction & Output – Use the trained model to predict OCD severity and treatment effectiveness.

Model Training & Evaluation



Workflow Diagram

Output & Results

Web Application

- The **Ada Boost Classifier** was the best-performing model.
- **Accuracy:**
- Before Hyperparameter Tuning: 0.51
- After Hyperparameter Tuning: 0.54

Observations

- Model predicts with high accuracy.
- Some slight variations due to feature importance & dataset limitations.
- Further tuning could improve generalization.

Thank You!

Netflix Data Cleaning, Analysis and Visualization

Presented by: Yash Sahu

Table of Contents

Introduction & Problem Statement

Introduction

- The project aims to analyze and visualize Netflix data to uncover trends in content distribution, genres, and popularity.

Problem Statement

- How is Netflix’s content distributed by year, genre, and country?
- What are the most common content types and trends over time?

Dataset Description

Overview

- The dataset contains information on Netflix movies and TV shows, including:
- Title, Genre, Release Year, Country, Duration and Ratings.

Data Insights

- Total Records: (8790 rows and 10 columns)
- Missing Values: (No missing data)

Dataset Description

```
df.head()
```

[3] ✓ 0.0s Python

	show_id	type	title	director	country	date_added	release_year	rating	duration	listed_in
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	United States	9/25/2021	2020	PG-13	90 min	Documentaries
1	s3	TV Show	Ganglands	Julien Leclercq	France	9/24/2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...
2	s6	TV Show	Midnight Mass	Mike Flanagan	United States	9/24/2021	2021	TV-MA	1 Season	TV Dramas, TV Horror, TV Mysteries
3	s14	Movie	Confessions of an Invisible Girl	Bruno Garotti	Brazil	9/22/2021	2021	TV-PG	91 min	Children & Family Movies, Comedies
4	s8	Movie	Sankofa	Haile Gerima	United States	9/24/2021	1993	TV-MA	125 min	Dramas, Independent Movies, International Movies

```
df.columns
```

[4] ✓ 0.0s Python

```
Index(['show_id', 'type', 'title', 'director', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in'],
      dtype='object')
```

This is the output of df.head() first five rows of dataset.

List of columns in dataset.

Data Preprocessing

Steps:

- Handling Missing Values: Replaced or dropped missing values in relevant columns.
- Date Formatting: Converted release dates into a standardized format for analysis using pandas.to_datetime()
- Feature Engineering: Extracted new features (e.g., Year and Month of Release).

```
### Extracting year, month, day from date_added
df['date_added_year'] = df['date_added'].dt.year
df['date_added_month'] = df['date_added'].dt.month
df['date_added_month_name'] = df['date_added'].dt.month_name()
df['date_added_day'] = df['date_added'].dt.day
```

✓ 0.0s

Exploratory Data Analysis (EDA) & Insights

Key Questions Explored:

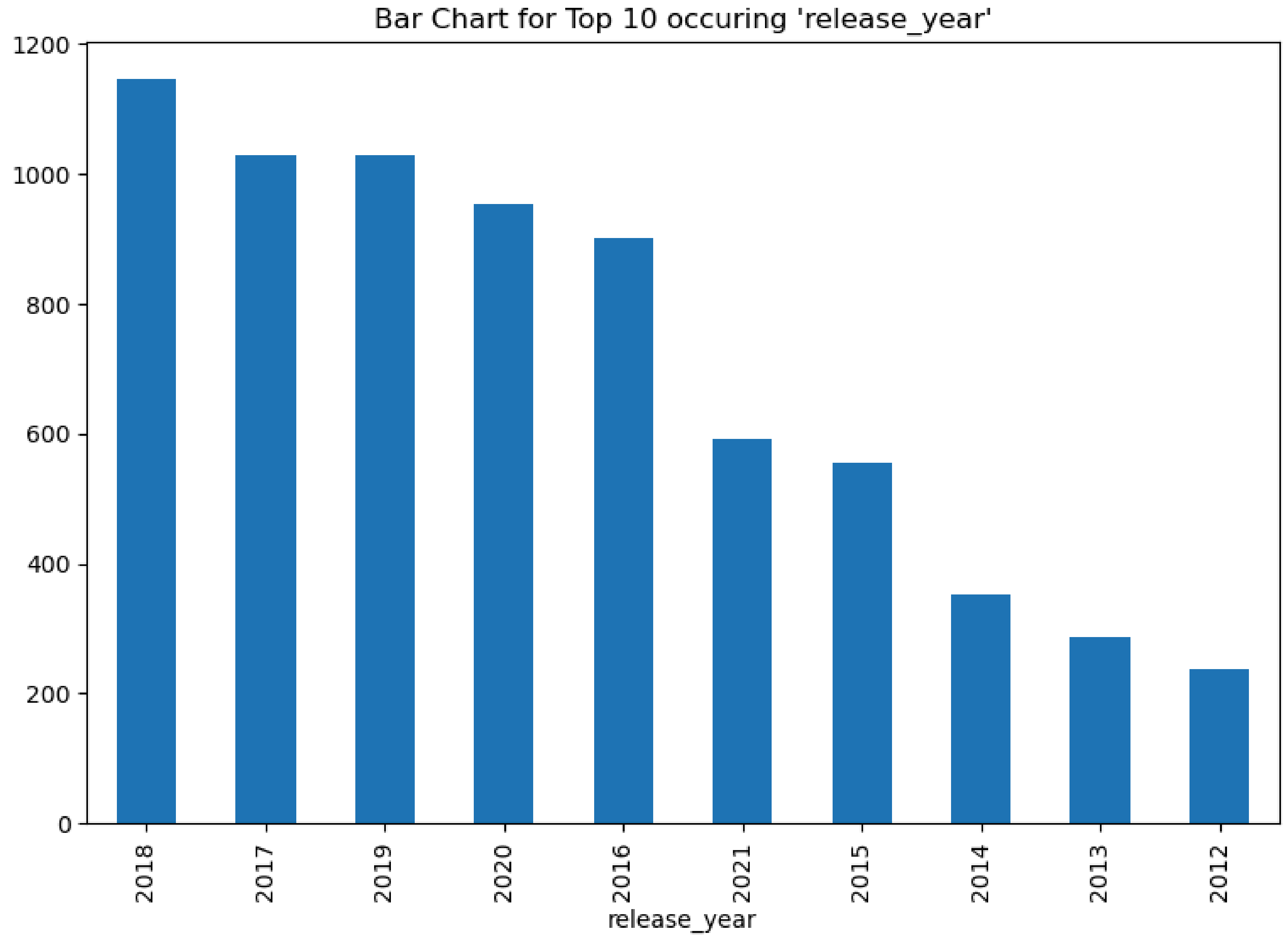
- What is the distribution of movies vs. TV shows?
- What are the most popular genres on Netflix?
- Which countries contribute the most content?
- How has content production changed over the years?

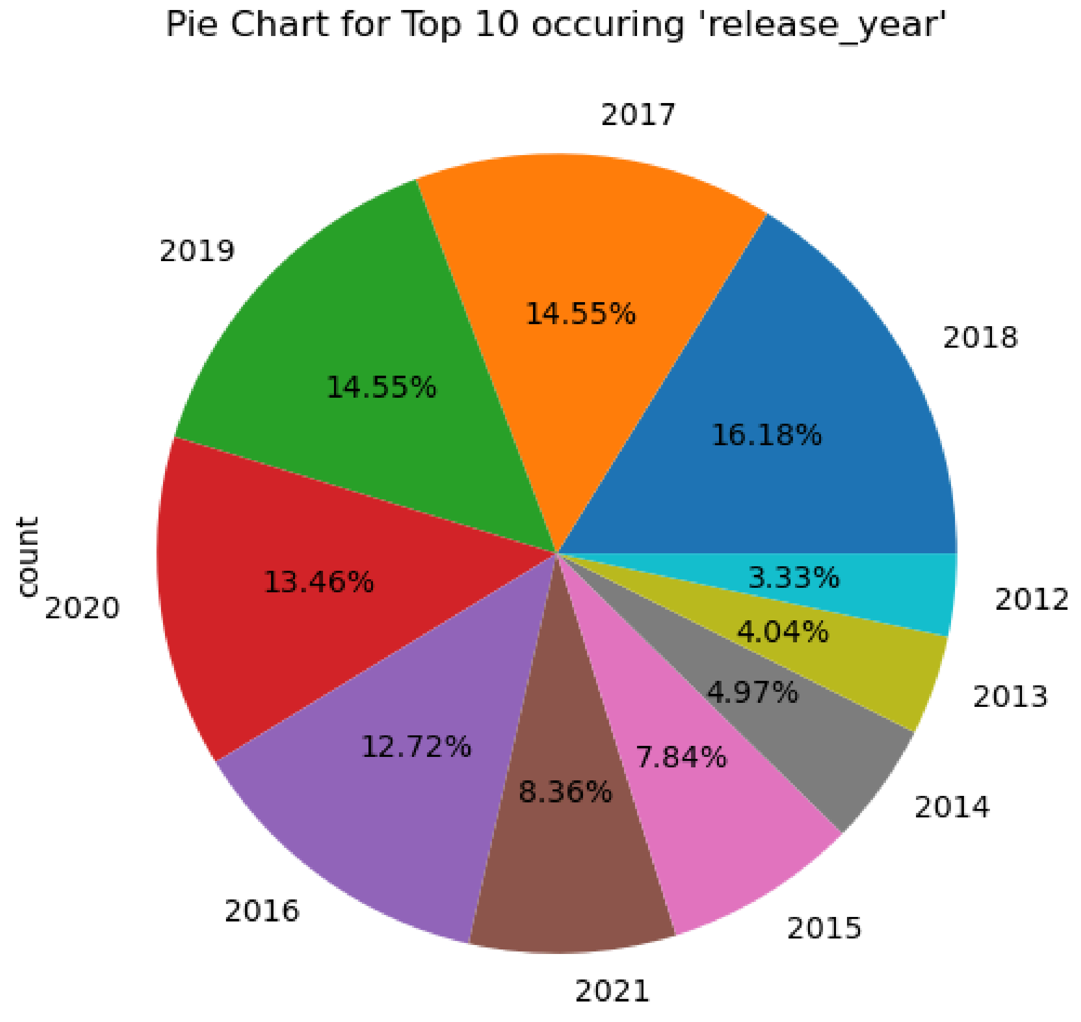
```
df['type'].value_counts()
```

0.0s

```
type
Movie      6126
TV Show    2664
Name: count, dtype: int64
```

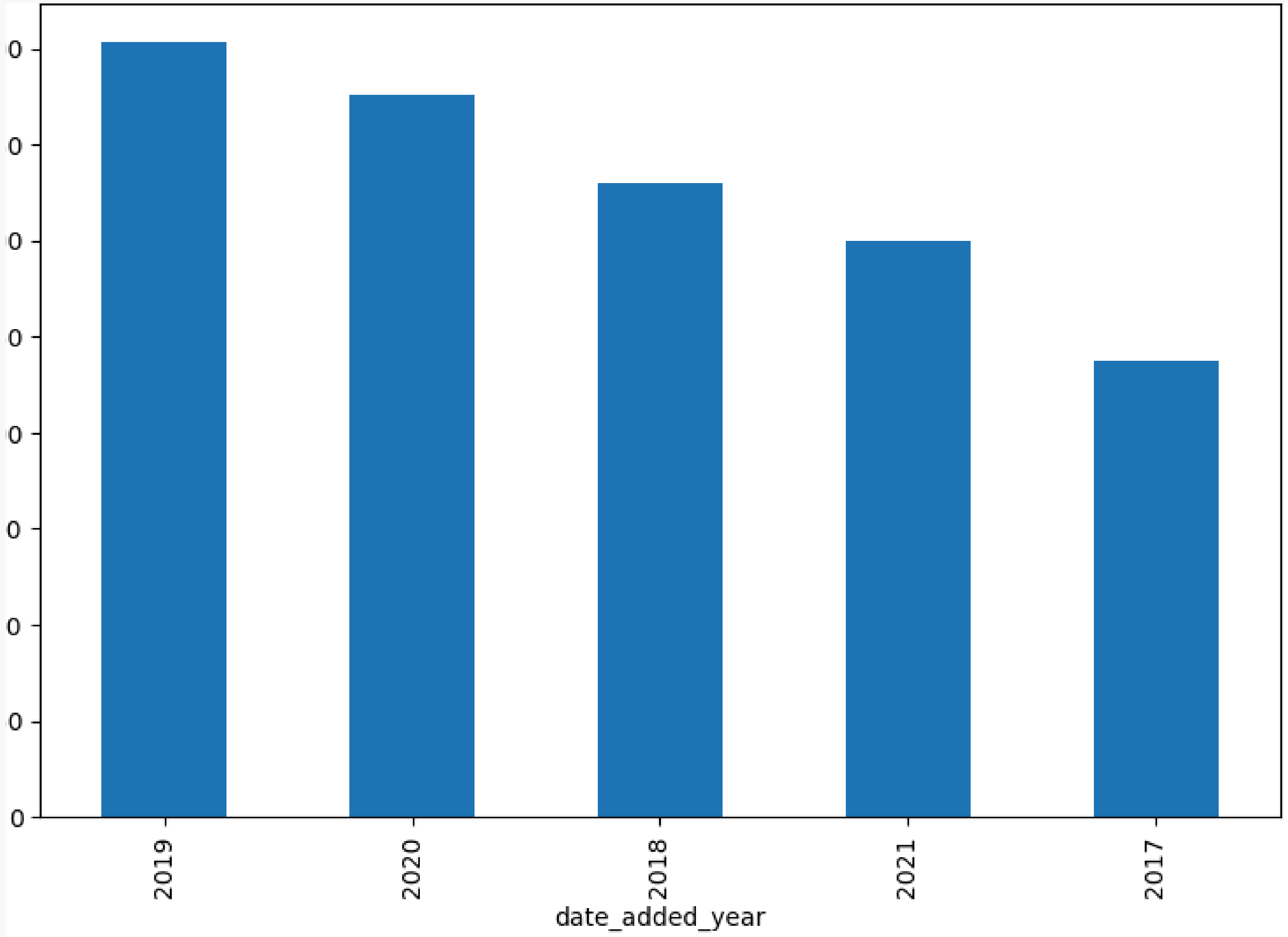
Data Visualization

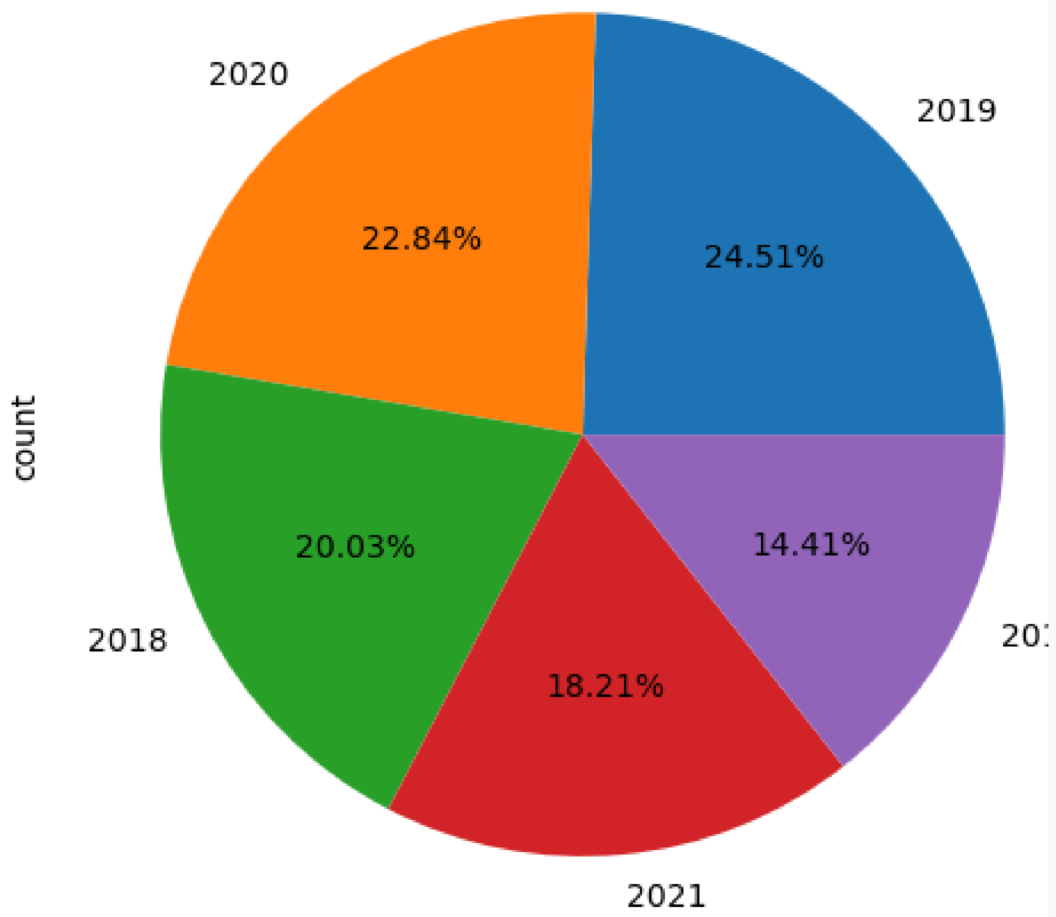




Bar chart and Pie chart for Top 10 years in ‘release_year’

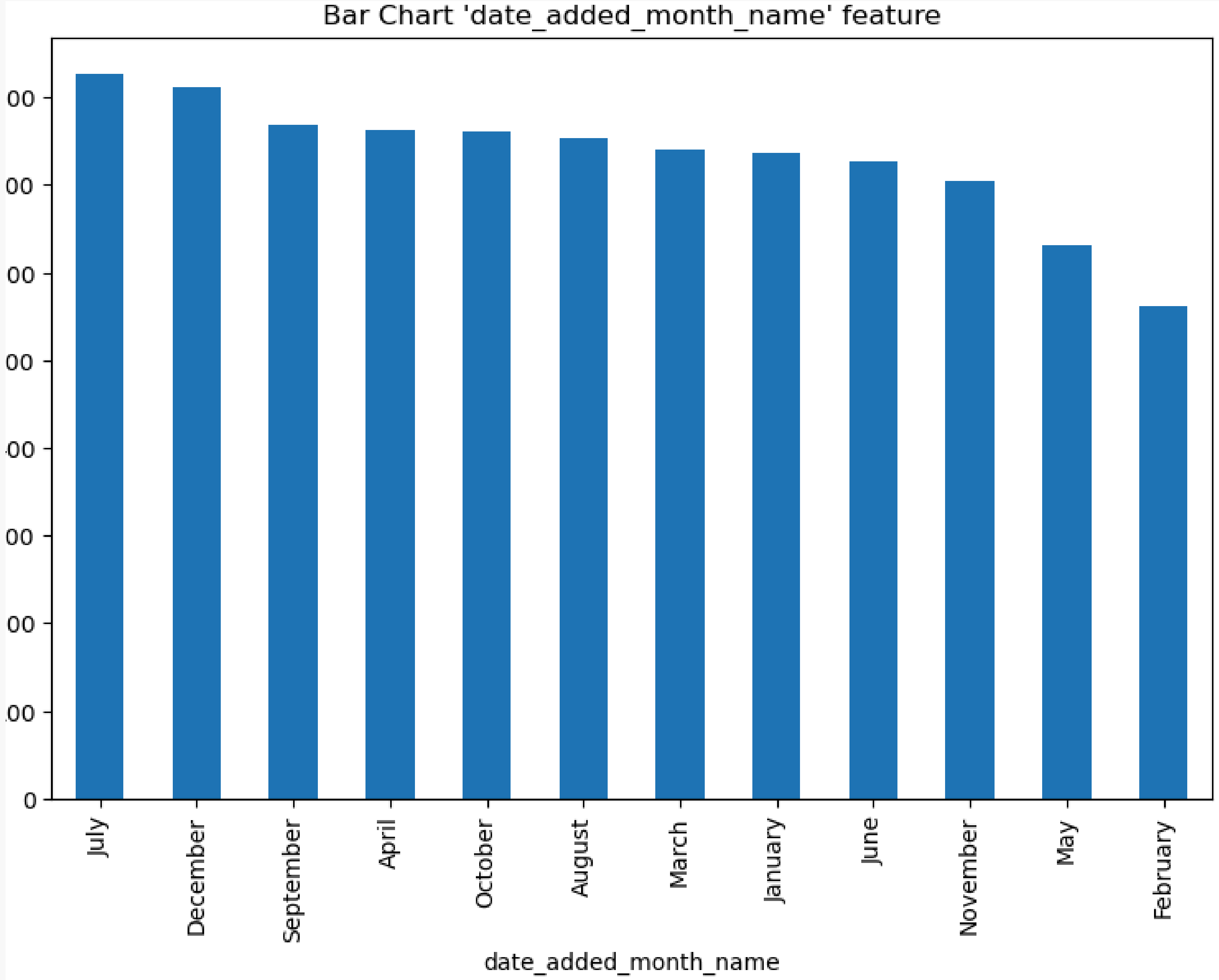
Data Visualization

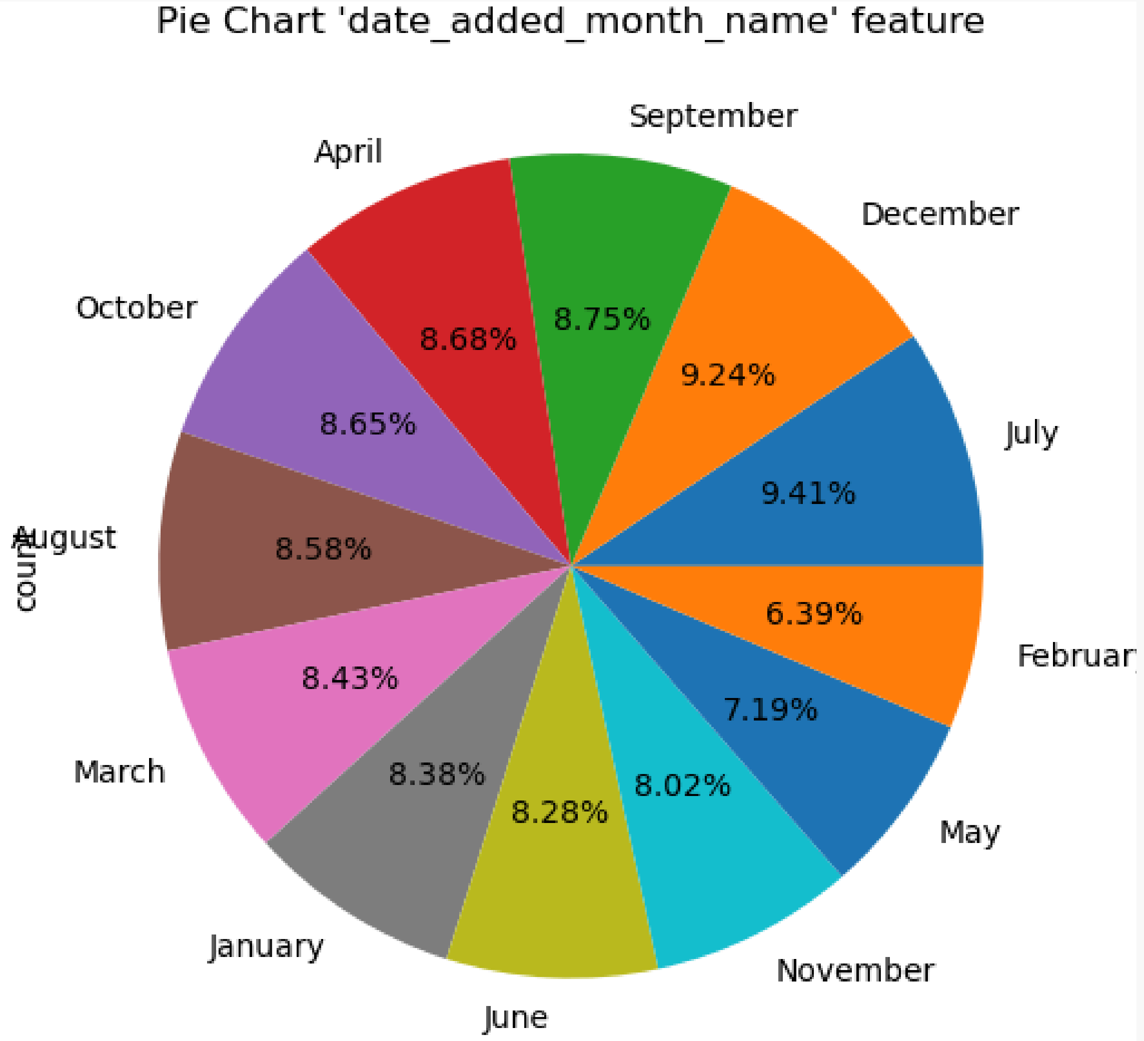




Distribution of year in date_added feature

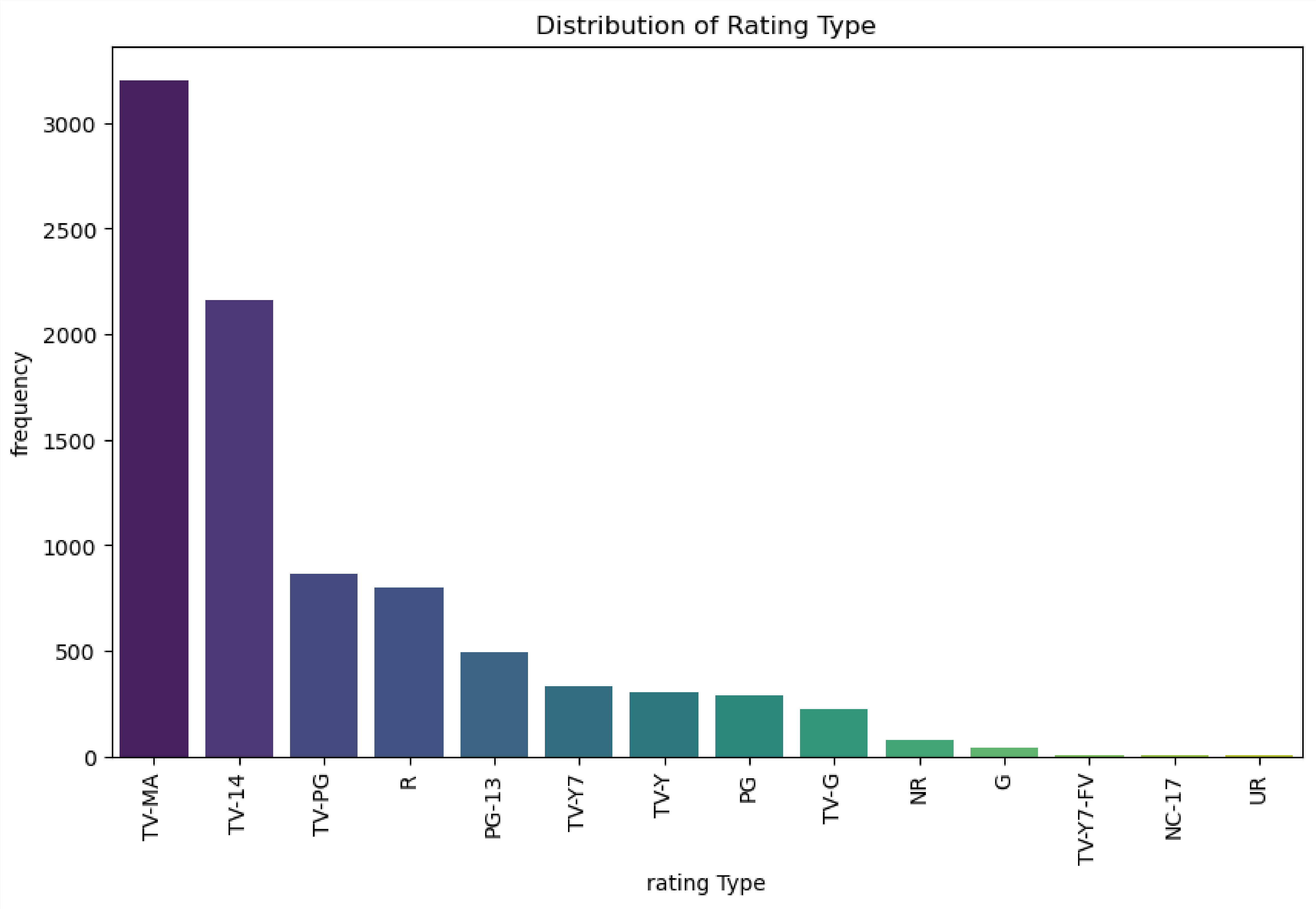
Data Visualization





Distribution of month in date_added feature

Data Visualization



Conclusion

Conclusion:

- This analysis provided insights into Netflix's content trends, genre distribution, and country-wise contributions.
- Visualizations helped in understanding data patterns and trends over time.

Thank You!