**Data Cleaning With Google Playstore Dataset**

**Data Cleaning With Google Playstore Dataset**

In this session, the focus is on data cleaning and preparation using the Google Play Store dataset, which contains over 10,000 records and 20 columns. The objective is to prepare the data for exploratory data analysis (EDA) and feature engineering.

**Overview of the Dataset**

The Google Play Store dataset is unique and different from previously used datasets. It contains a variety of columns such as app name, category, rating, reviews, size, installs, type, price, content rating, genres, Android version, and more. The dataset is not clean and contains special characters and missing values.

**Initial Data Exploration**

First, the dataset is read using pandas' **read\_csv** function. The shape of the dataset is checked, revealing over 10,000 records and 13 columns. General information is obtained using **df.info()**, which shows that most features are of object type, except for the rating, which is a float.

**Checking for Missing Values**

By using **df.isnull().sum()**, it is observed that there are missing values in several columns: ratings, type, current version, and Android version. These observations are important for further data cleaning.

**Handling the Reviews Column**

The reviews column appears to be numeric but is stored as a string. By checking if all values are numeric using **df['Reviews'].str.isnumeric().sum()**, it is found that one value is not numeric: '3.0M'. This value needs to be handled before converting the column to integers.

To identify the problematic record, filter for non-numeric values in the reviews column.

**python Code Sample**

df[~df['Reviews'].str.isnumeric()]

The record with '3.0M' in reviews is either dropped or converted to 3,000,000. After handling this, the reviews column can be safely converted to integer type.

**Creating a Copy for Data Cleaning**

To avoid modifying the original dataset, create a copy for cleaning.

**python Code Sample**

df\_copy = df.copy()

Drop the problematic index (e.g., 10472) from the reviews column.

**python Code Sample**

df\_copy = df\_copy.drop(index=10472)

Now, convert the reviews column to integer type.

**python Code Sample**

df\_copy['Reviews'] = df\_copy['Reviews'].astype(int)

**Cleaning the Size Column**

The size column contains values like '8.0M' (million), '19k' (thousand), and 'Varies with device'. The goal is to standardize these values into a numeric format (preferably in kilobytes).

Replace 'M' with '000' to convert millions to thousands, and remove 'k'. Replace 'Varies with device' with NaN.

**python Code Sample**

import numpy as np

df\_copy['Size'] = df\_copy['Size'].str.replace('M', '000')

df\_copy['Size'] = df\_copy['Size'].str.replace('k', '')

df\_copy['Size'] = df\_copy['Size'].replace('Varies with device', np.nan)

df\_copy['Size'] = df\_copy['Size'].astype(float)

**Cleaning Installs and Price Columns**

The installs and price columns contain special characters such as '+', ',', and '$'. These need to be removed to convert the columns to numeric types.

**python Code Sample**

characters\_to\_remove = ['+', ',', '$']

columns\_to\_clean = ['Installs', 'Price']

for item in characters\_to\_remove:

for column in columns\_to\_clean:

df\_copy[column] = df\_copy[column].str.replace(item, '')

Convert the installs column to integer and the price column to float.

**python Code Sample**

df\_copy['Installs'] = df\_copy['Installs'].astype(int)

df\_copy['Price'] = df\_copy['Price'].astype(float)

**Handling the Last Updated Column**

The last updated column contains date strings. Convert this column to pandas datetime format, and extract day, month, and year into separate columns.

**python Code Sample**

df\_copy['Last Updated'] = pd.to\_datetime(df\_copy['Last Updated'])

df\_copy['Day'] = df\_copy['Last Updated'].dt.day

df\_copy['Month'] = df\_copy['Last Updated'].dt.month

df\_copy['Year'] = df\_copy['Last Updated'].dt.year

**Finalizing and Saving the Cleaned Dataset**

After cleaning, the dataset contains properly formatted numerical and categorical features. Save the cleaned dataset to a CSV file for further analysis.

**python Code Sample**

df\_copy.to\_csv('data/Google\_cleaned.csv', index=False)

**Conclusion**

The data cleaning process is essential for preparing real-world datasets for analysis. The next steps will involve exploratory data analysis and feature engineering using the cleaned data.

**Key Takeaways**

* Data cleaning is a crucial first step when working with real-world datasets, as they often contain inconsistencies and special characters.
* Converting string-based numerical columns to appropriate numeric types is essential for analysis.
* Handling missing values and special cases, such as 'varies with device', ensures data integrity.
* Saving the cleaned dataset allows for efficient further analysis, including EDA and feature engineering.