Google Play Store Dataset

IS 537 Final Project Presentation

Cleaners United: Niken Shah (niken2), Shubhangi Singhal (ss100)

Contents

Introduction & Overview

- Work on Google Play Store Dataset, perform data cleaning tasks to prepare for exploratory Data Analysis.
- Familiarize with data cleaning tools like OpenRefine
- Explore alternative options like SQL and Python
- Work on data visualizations using tools like Tableau or PowerBI

Application	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Version	Android Version
slither.io	GAME	4.4	5234162	Varies with device	100,000,000+	Free	0	Everyone	Action	14-Nov-17	Varies with device	2.3 and up
Temple Run 2	GAME	4.3	8118609	62M	500,000,000+	Free	0	Everyone	Action	5-Jul-18	1.49.1	4.0 and up
Helix Jump	GAME	4.2	1497361	33M	100,000,000+	Free	0	Everyone	Action	9-Apr-18	1.0.6	4.1 and up
Zombie Hunter King	GAME	4.3	10306	50M	1,000,000+	Free	0	Mature 17+	Action	1-Aug-18	1.0.8	2.3 and up
Stick the Buddy	GAME	4.3	1000417	Varies with device	50,000,000+	Free	0	Teen	Action	5-Jul-18	Varies with device	4.4 and up
Zombie Catchers	GAME	4.7	990491	75M	10,000,000+	Free	0	Everyone	Action	24-May-18	1.0.27	4.1 and up
Sniper 3D Gun Shooter: Free Shoo	GAME	4.6	7671249	Varies with device	100,000,000+	Free	0	Mature 17+	Action	2-Aug-18	Varies with device	Varies with device
Miraculous Ladybug & Cat Noir -	GAME	4.5	183846	99M	10,000,000+	Free	0	Everyone	Action	30-Jul-18	1.0.6	4.4 and up
Garena Free Fire	GAME	4.5	5465624	53M	100,000,000+	Free	0	Teen	Action	3-Aug-18	1.21.0	4.0.3 and up
Slowmasters	GAME	4.7	1534466	Varies with device	50,000,000+	Free	0	Teen	Action	23-Jul-18	2.12.5	4.1 and up
Talking Tom Gold Run	GAME	4.6	2698348	78M	100,000,000+	Free	0	Everyone	Action	31-Jul-18	2.8.2.59	4.1 and up
Tomb of the Mask	GAME	4.1	55380	39M	5,000,000+	Free	0	Everyone	Action	24-Jul-18	1.2.1	5.0 and up
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Use Cases

- Our use case is to create an interactive dashboard with customizable filters for the end users
- **Already good enough use cases** -
  - Top applications based on highest ratings received basis the category of the application, price, type and genre of the application.
  - Top applications based on highest reviews received basis the category of the application, price, type and genre of the application.
  - Top play Store applications in particular year.
- **Never good enough use cases** -
  - Which application current versions have had the most users
  - How to compare the applications based on the current version.
- **Middle of the road use cases** -
  - Top n genres based on count of the applications
  - Top n categories based on the average user ratings.
  - Top n applications downloaded based on the count of the reviews received.
  - Top n download categories

Data Cleaning Methodology - Python

- Utilized
  - Pandas and Math library
  - Dataframe Utility functions e.g. head, columns, count, drop_duplicates etc
  - String utility functions e.g. str.strip, str typecasting, and len etc.
  - List and array data structure to implement iterative data cleaning algorithms
  - Time taken for code execution - 1.327 seconds
- Pros
  - Lesser processing time
  - Bigger user community as compared to OpenRefine
  - Presence of data cleaning libraries e.g. Pandas and Numpy
  - Code Readability
- Cons
  - Absence of drag and drop user interface
  - Absence of some data cleaning menus and features present in OpenRefines
  - Speciality is not data cleaning as it is a programming language

Data Cleaning Methodology - OpenRefine

- Important Cleaning Steps:
  - Standardizing column data types
  - Using Columnar transformations
  - Using RegEx for Pattern matching/replacement
  - Clustering options in Facets to group similar cells
- Pros:
  - Ideal for preliminary analysis of data
  - Easy to learn and interpret cleaning options
- Cons:
  - Not suitable for performing complex cleaning functions
  - Regular Expressions knowledge is needed

SQLite & Relational Schema

- We used https://sqliteonline.com/ - an online SQLite3 server for loading our dataset.
- SQL code for creating relational schema:

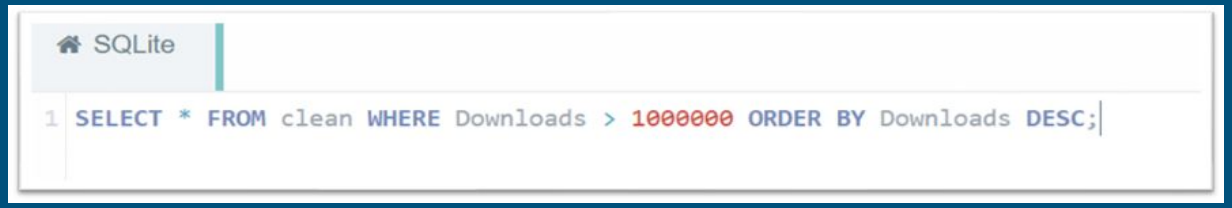
  DROP TABLE IF EXISTS `clean`; CREATE TABLE `clean` (`ID` mediumint(9), `Application` varchar(194) DEFAULT NULL, `Category` varchar(19) DEFAULT NULL, `Rating` varchar(3) DEFAULT NULL, `Reviews` int(11) DEFAULT NULL,`Size` varchar(18) DEFAULT NULL,`Downloads` bigint(20) DEFAULT NULL,`Type` varchar(4) DEFAULT NULL,`Price` varchar(8) DEFAULT NULL,`Content Rating` varchar(15) DEFAULT NULL,`Genres` varchar(37) DEFAULT NULL,`Last Updated` date(8) DEFAULT NULL,`Current Version` varchar(50) DEFAULT NULL, `Android Version` varchar(18) DEFAULT NULL) ;

- Sample Insert Query:

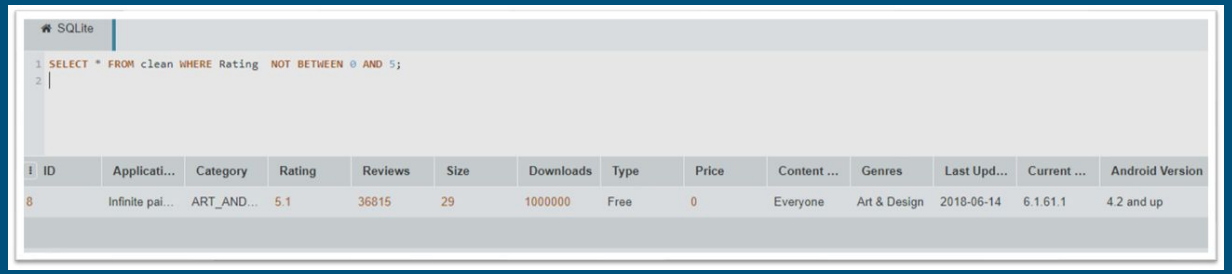
  INSERT INTO `clean` (`ID`,`Application`,`Category`,`Rating`,`Reviews`,`Size`,`Downloads`,`Type`,`Price`,`Content Rating`,`Genres`,`Last Updated`,`Current Version`,`Android Version`) VALUES ('0','Photo editor & candy camera & grid & scrapbook','ART_AND_DESIGN','4.1','159','19','10000','Free','0','Everyone','Art & Design','2018-01-07','1.0.0','4.0.3 and up');

SQLite & Relational Schema

- Sample SQL Query for Data Profiling

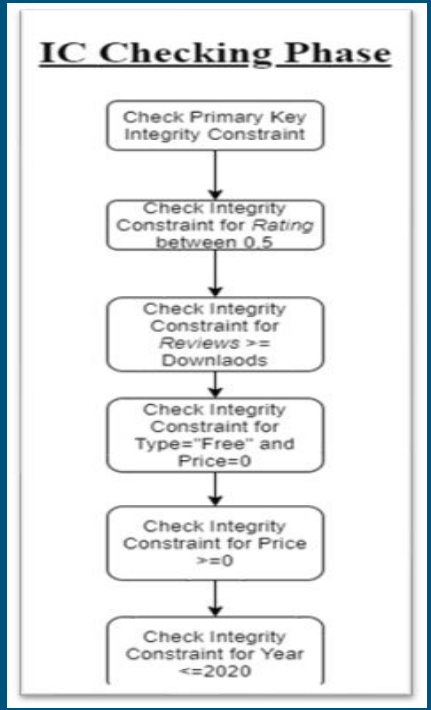


- Output of Integrity Constraint Check in SQLite

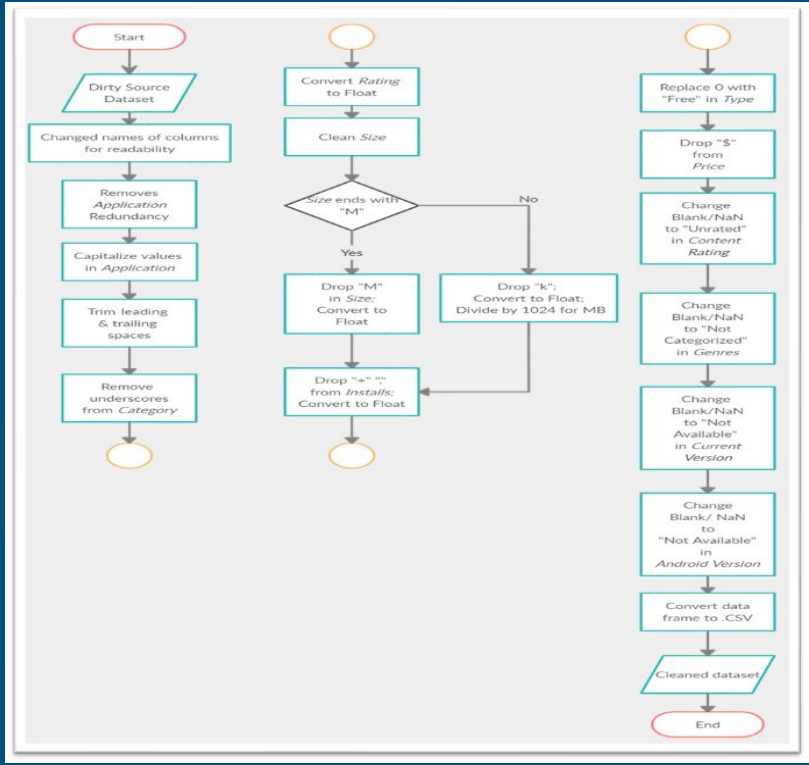


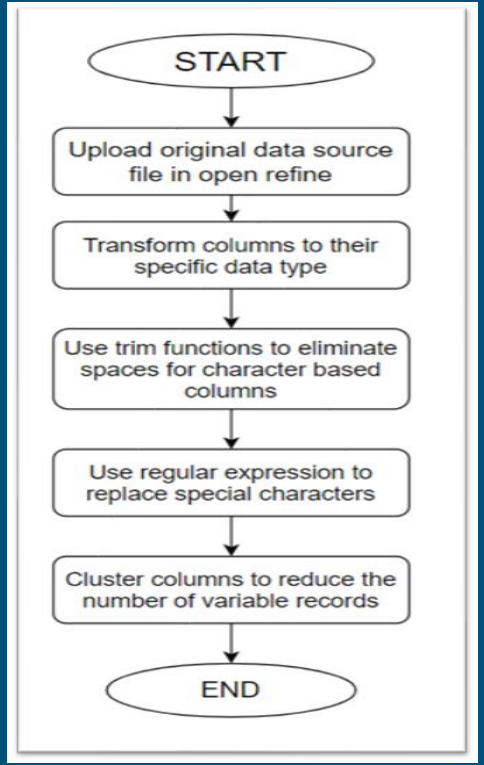
Integrity Constraint Checks

- **Rating should be between 0 and 5 inclusive:**
  SELECT * from clean WHERE Rating NOT BETWEEN 0 AND 5;
- **Primary Key ID should uniquely determine all other attributes:**
  SELECT a.* FROM clean a
  JOIN (SELECT *, COUNT(*)
  FROM clean b
  GROUP BY ID
  HAVING count(*) > 1) b
  ON a.ID = b.ID ORDER BY a.ID;
- **Price should be non-negative:**
  SELECT * FROM clean WHERE Price < 0;
- **Count of reviews should be greater than or equal to the count of downloads:**
  SELECT * FROM clean WHERE Reviews > Downloads;
- **Applications which are free must have the corresponding price as 0:**
  SELECT * FROM clean WHERE Type = 'Free' and Price > 0;

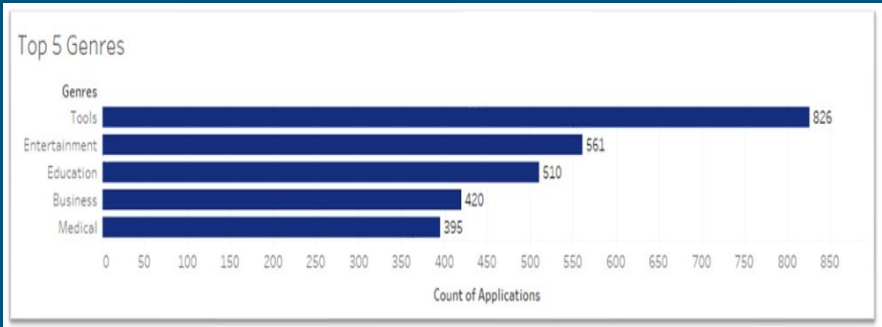


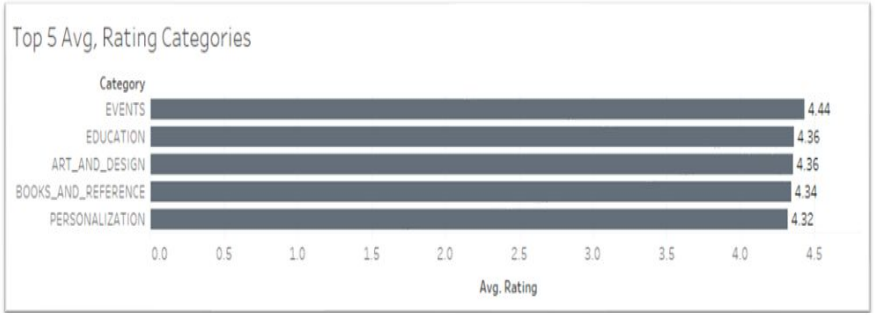
Workflows

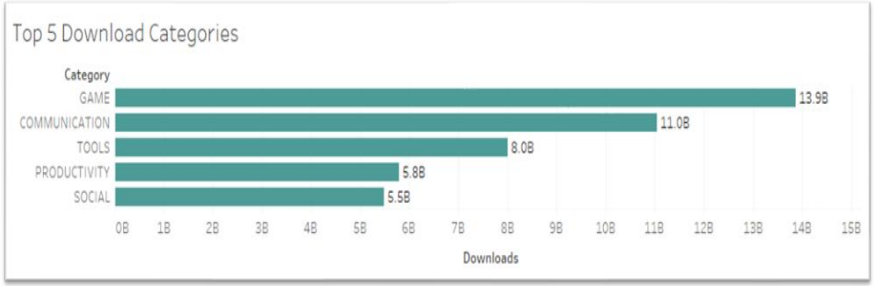

Start
Dirty Source Dataset
Changed names of columns for readability
Removes Application Redundancy
Capitalize values in Application
Trim leading & trailing spaces
Remove underscores from Category
Convert Rating to Float
Clean Size
Size ends with "M"
No
Yes
Drop "M" in Size; Convert to Float
Drop "k"; Convert to Float; Divide by 1024 for MB
Drop "+" "," from Installs; Convert to Float
Replace 0 with "Free" in Type
Drop "$" from Price
Change Blank/NaN to "Unrated" in Content Rating
Change Blank/NaN to "Not Categorized" in Genres
Change Blank/NaN to "Not Available" in Current Version
Change Blank/ NaN to "Not Available" in Android Version
Convert data frame to .CSV
Cleaned dataset
End


START
Upload original data source file in open refine
Transform columns to their specific data type
Use trim functions to eliminate spaces for character based columns
Use regular expression to replace special characters
Cluster columns to reduce the number of variable records
END

Data Visualizations

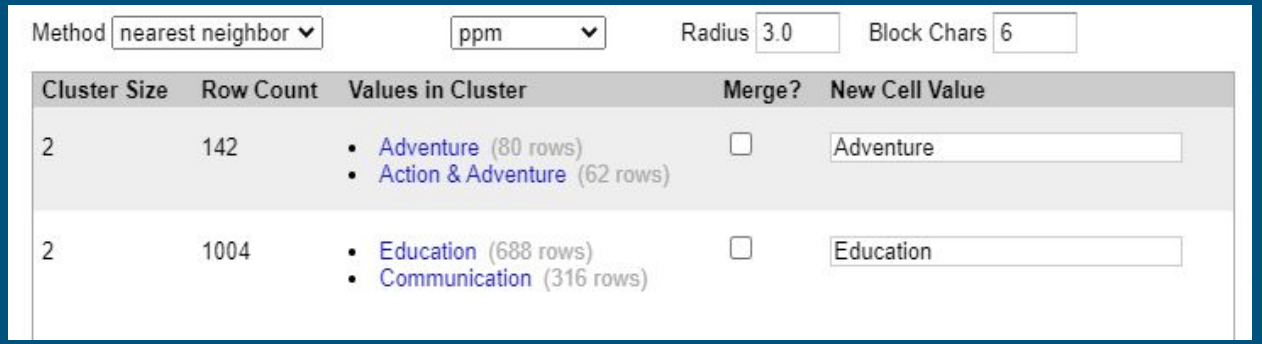






Results

Grouped similar Genres:



Implemented integrity constraints so that records in the future are standardized:

```
SELECT a.*
FROM clean a
JOIN (SELECT *, COUNT(*)
FROM clean b
GROUP BY ID
HAVING COUNT(*) > 1 ) b
ON a.ID = b.ID
ORDER BY a.ID;
```

ID	Applicati...	Category	Rating	Reviews	Size	Downloads	Type	Price	Content ...	Genres	Last Upd...	Current ...	Android Version
0	Photo edit...	ART_AND...	4.1	159	19	10000	Free	0	Everyone	Art & Design	2018-01-07	1.0.0	4.0.3 and up
0	Photo editor	ART_AND...	2.1	59	198	10000	Free	0	Everyone	Art & Design	2018-01-07	1.0.0	4.0.3 and up

Results

- Some results of data cleaning in Python -
  - Formatted the data values across columns for standardisation e.g. changed text to float for *Size*
  - Renamed field names for better audience readability and understanding such as *App* to *Application*
  - Formulated code to handle NaN values, e.g. updated to "*Unrated*" in *Content Rating*
  - Applied sorting function to fetch top Google Play Store applications using *Reviews*
  - Rectified incorrect column data type e.g. for *Reviews*
  - Handled blank values and replaced with "Not Categorized"/ "Not Available" for *Genres* & *Android Version*
  - Converted App *Size* from KiloBytes to MegaBytes by dividing kB values by 1024.

```
0                     19
1                     14
2                    8.7
3                     25
4                    2.8
                    ...
10836                 53
10837                3.6
10838                9.5
10839    Varies with device
10840                 19
Name: Size, Length: 9660, dtype: object
Wall time: 15 ms
```

App Size from KB to MB

Conclusion

- **Key Takeaways**
  - It is crucial to perform data profiling before starting the data cleaning process in order to avoid identifying new loopholes in the dataset
  - The data might be collected from multiple sources and there may be discrepancies in the data.
  - Data cleaning is important because the clean data eases visualizing and exploratory data analysis.
  - There is always scope for data cleaning no matter how good the data looks on first glance.
- **Challenges Faced**
  - Manually updated one anomaly record in MS Excel to improve execution and space complexity.
  - Poor execution time running Genre Clustering in Python.
  - Scoping and identifying data cleaning problems in the dataset basis the use cases required research of Google Play Store Applications.

Thank You!

Any Questions?