

# Capstone Project - 1

## Team: Reality

### Team Members

AYUSH JAIN  
DHONGARI PAVAN  
NADEEHA A  
ROMALY DAS  
SREENIVASAN KV  
YASH PATIL

# World of Android & Malwares!

*"Obviously, you will always see more **malware targeting Android** because Android is used more than any smartphone platform by a pretty substantial difference."* - Sundar Pichai

A new malicious app is released every **7.5 seconds**, **10,000** new samples everyday!



Apple CEO Tim Cook says "**Android has 47 times**" more Malwares than iOS.

# Malware attack trends

**Information Extraction:** The malware in this category also endangers the device then steal your personal information such as IMEI number, user details and many more.

**Automatic Calls and SMS:** This malware group increase the billing of the user. This took the user's phone access like contact books and make automatic calls and send SMS to other numbers.

**Root Exploits:** This malware seek to gain system root rights in order to control the system and modify the system configuration with another application details.

**Dynamically Downloaded Code:** This method enables the installed application to download malicious code and use it on mobile devices without the user's knowledge.

# Our Problem

- To identify whether an application is **malware(1)** or **benign(0)**
- Based on data collected over 3 years during installation and runtime of an application.



# Digging up Malware Android Dataset..

- The data was collected from different app markets such as google play store and has 30k records.
- The dataset consists of four textual columns:
  - App :- Name of the App
  - Package :- OBB/Data package installed in root folder
  - Category :- App Category (eg. Entertainment, Adventure, puzzle, Action, Antivirus, etc.)
  - Description :- App Description

# Digging up Malware Android Dataset..

- The dataset consists of some numeric columns:

Rating :- Rating out of 5

Number of ratings :- No. of Ratings given by users

Price :- Price of the App

Related apps :- Apps related to installed App

Dangerous (D) permissions count :- No. of Dangerous  
Permissions allowed by user

Safe (S) permissions count :- No. of Safe Permissions allowed by  
user

# Digging up Malware Android Dataset..

- The rest of the columns are **binary columns** specifying certain kind of permissions:

**Default Permissions**

**Development Tools Permissions**

**Hardware Controls**

**Network Communications**

**Phone Calls**



# Digging up Malware Android Dataset..

- The rest of the columns are binary columns specifying certain kind

of permissions:

Service That cost you money

Storage

Systems tools

Personal info: Your accounts, Your Location, Your messages,

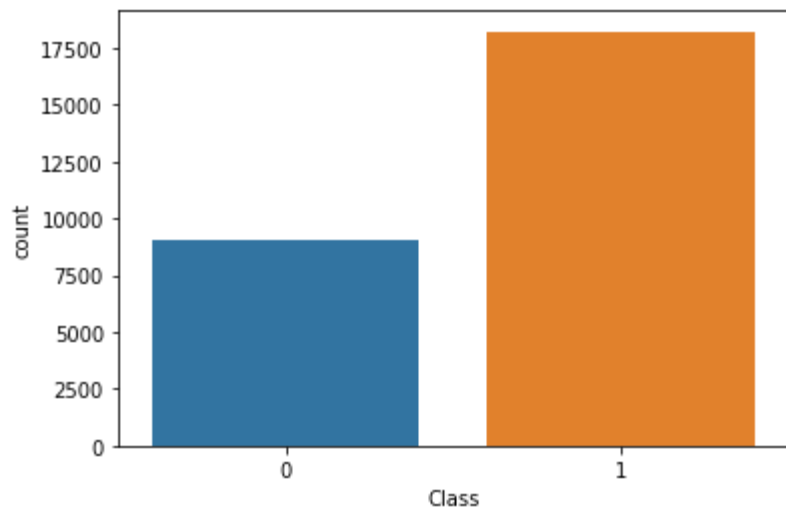
Other personal info



# Digging up Malware Android Dataset..

- The dataset has 2.6k duplicate records across all columns
- It has over 720 null records in related apps and 202 null records in dangerous permissions counts.

# Digging up Malware Android Dataset..



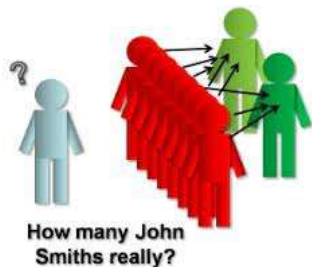
Class Distribution for Target Classes

*The dependent variable is a class count with 67% count of malware apps and 33% count of benign apps.*

# Where is my broom..??



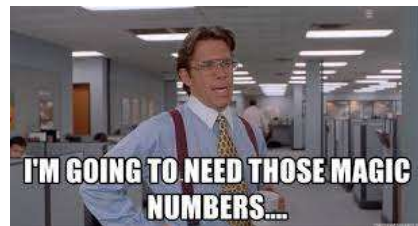
Handling Duplicates



Handling Null Values



Handling Numerical Columns



Handling Outliers



# UNCLEAN DATA

- There are 2689 Duplicates (Class 0: 921, Class 1: 1768)
- Dropped Duplicates, Shape of data(after dropping): 27310,184



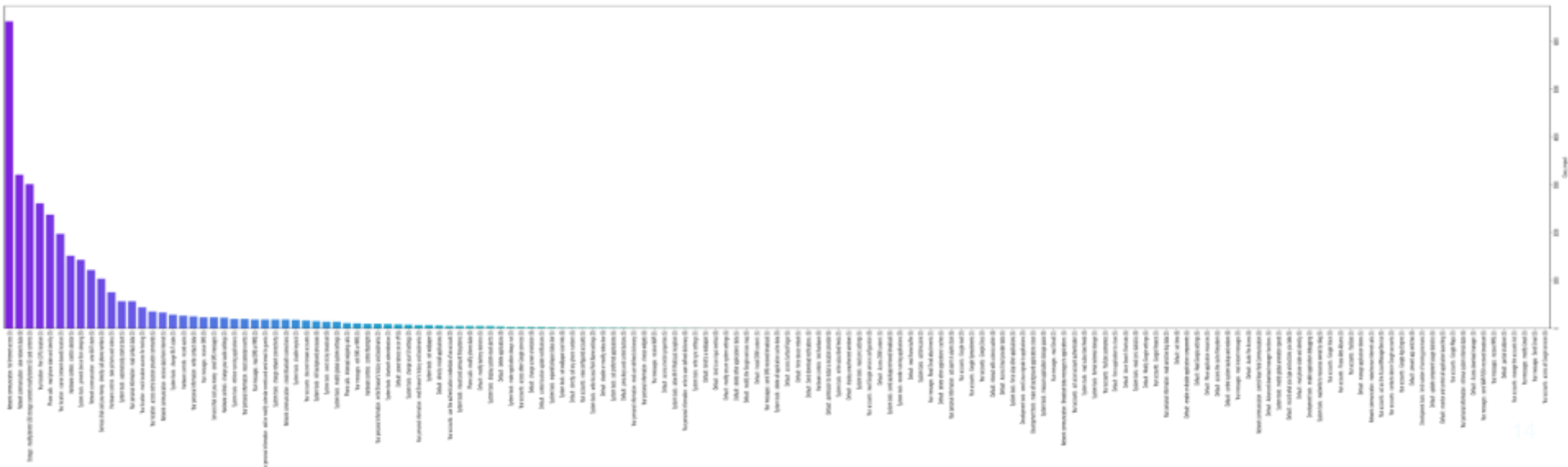
# Handling Nulls!



- **Related Apps:** There are 720 null values in this column.
- We have used **Datawig imputer package** to impute values for Related apps.
- **Dangerous permission count :** There are 201 null values in this column. We had imputed them with the mean value of 3.

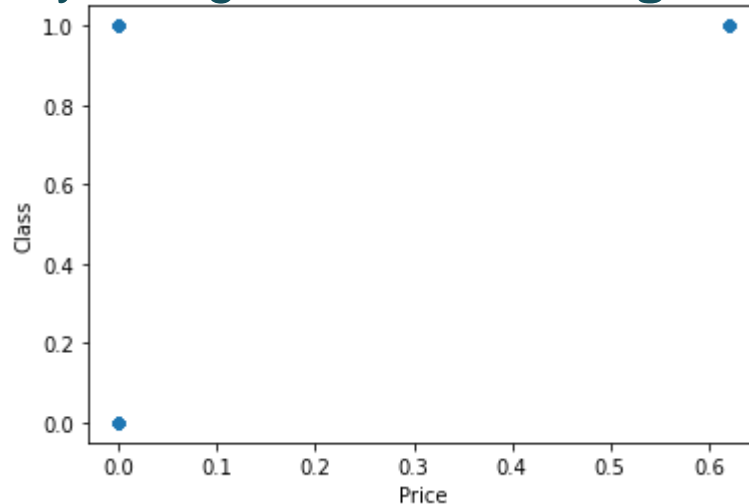
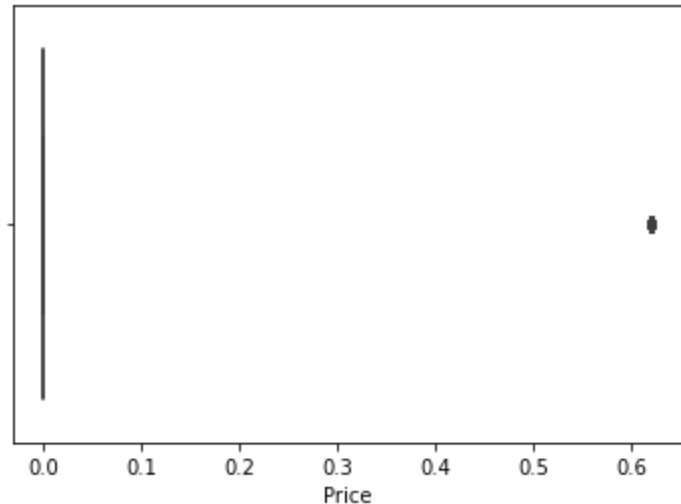
# Handling Numerical Columns

- There were 22 columns in which all the values are 0. So removed them as they are not impacting Target Class.



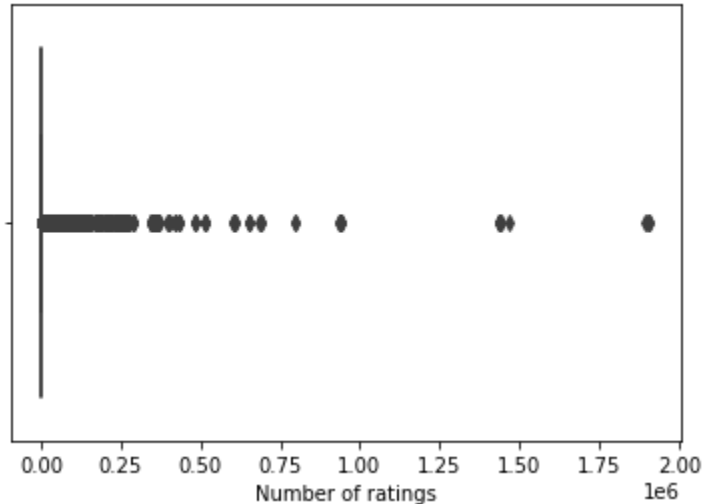
# Handling Outliers

- We have outliers in **Price column** and **Number of Ratings column**.
- We handled outliers in Price by doing **Mean Encoding**.



# Handling Outliers

- Number of ratings column



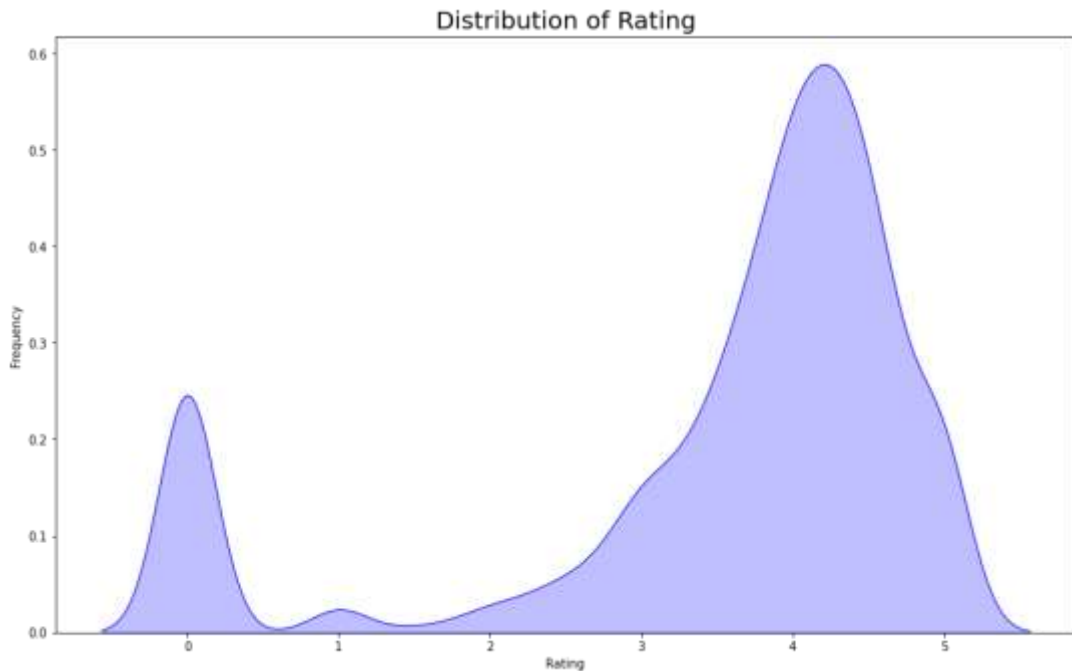
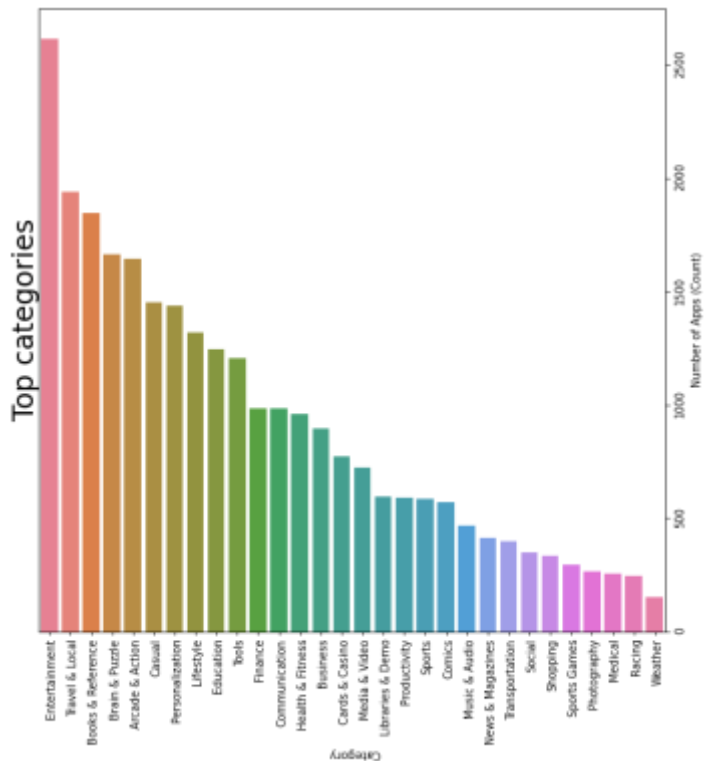


# Mean Encoding

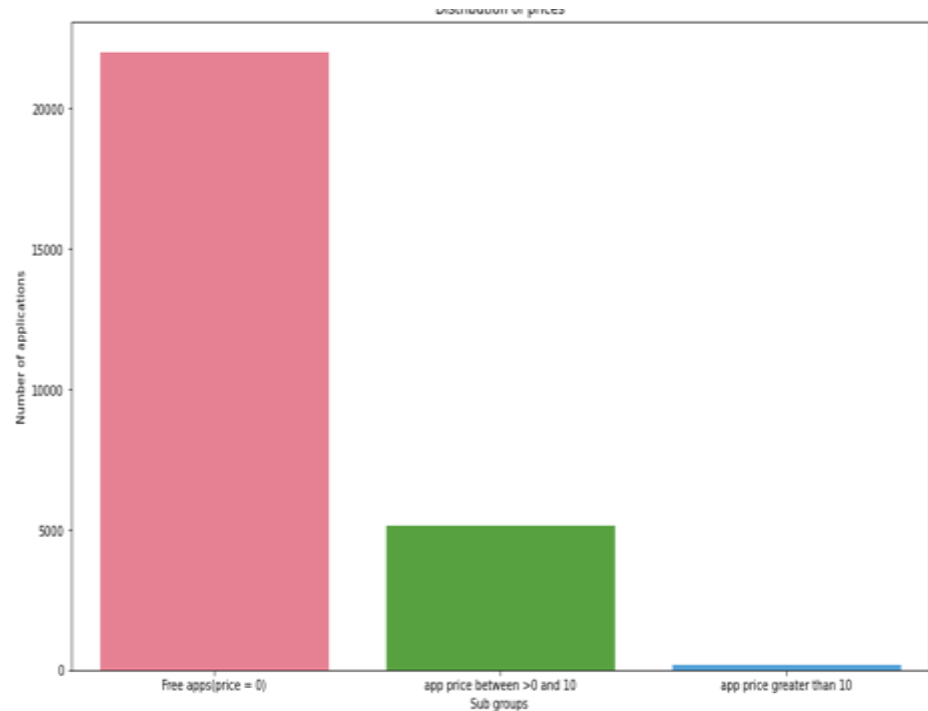
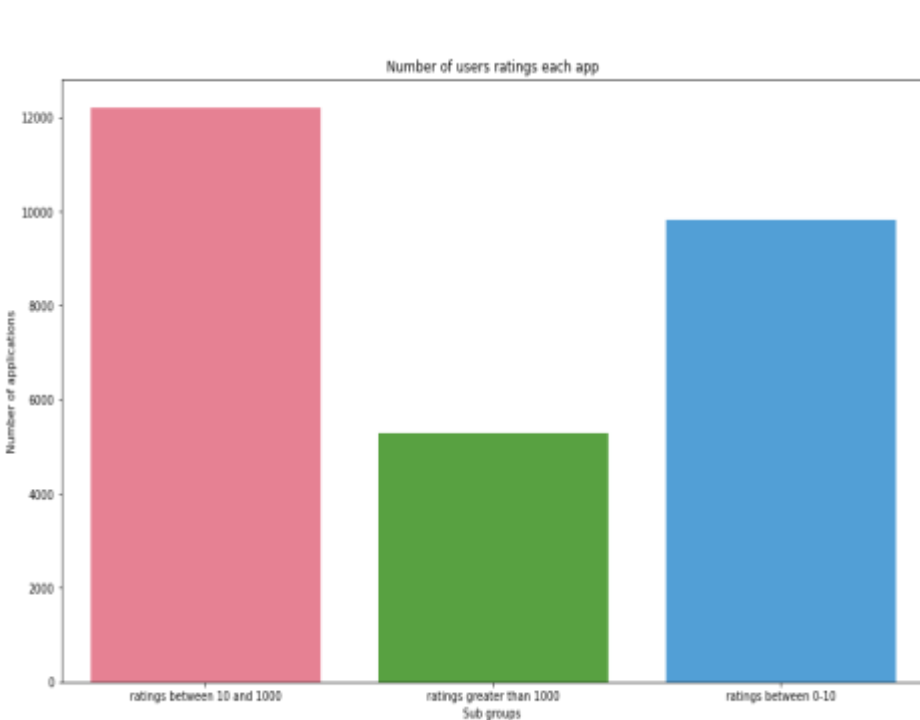
Category	
Arcade & Action	0.607453
Books & Reference	0.725370
Brain & Puzzle	0.635878
Business	0.448677
Cards & Casino	0.321226
Casual	0.487485
Comics	0.133333
Communication	0.477788
Education	0.609962
Entertainment	0.779625
Finance	0.493780
Health & Fitness	0.487476
Libraries & Demo	0.168576
Lifestyle	0.613937
Media & Video	0.475703
Medical	0.988889
Music & Audio	0.844485
News & Magazines	0.925000
Personalization	0.678454
Photography	0.885522
Productivity	0.834532
Racing	0.722222
Shopping	0.920200
Social	0.827068
Sports	0.963608

- We have “Category” column which is Categorical so we applied Mean encoding to convert each category into the respective mean.

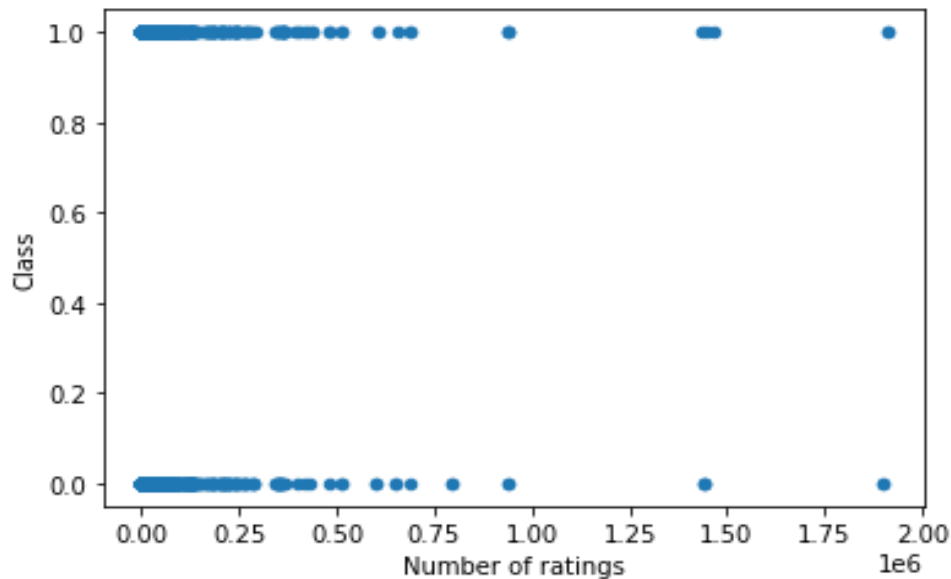
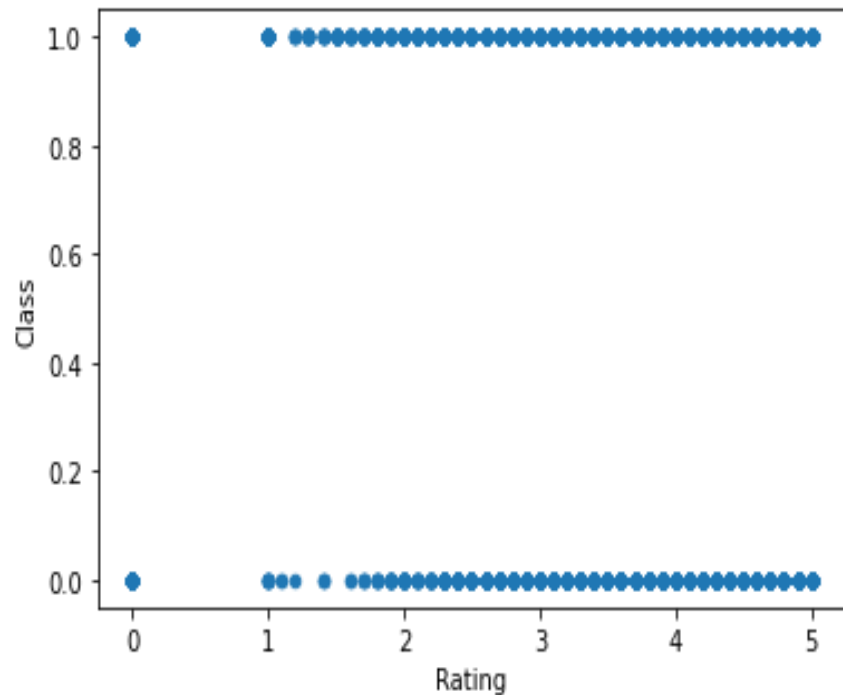
# Exploratory Data Analysis



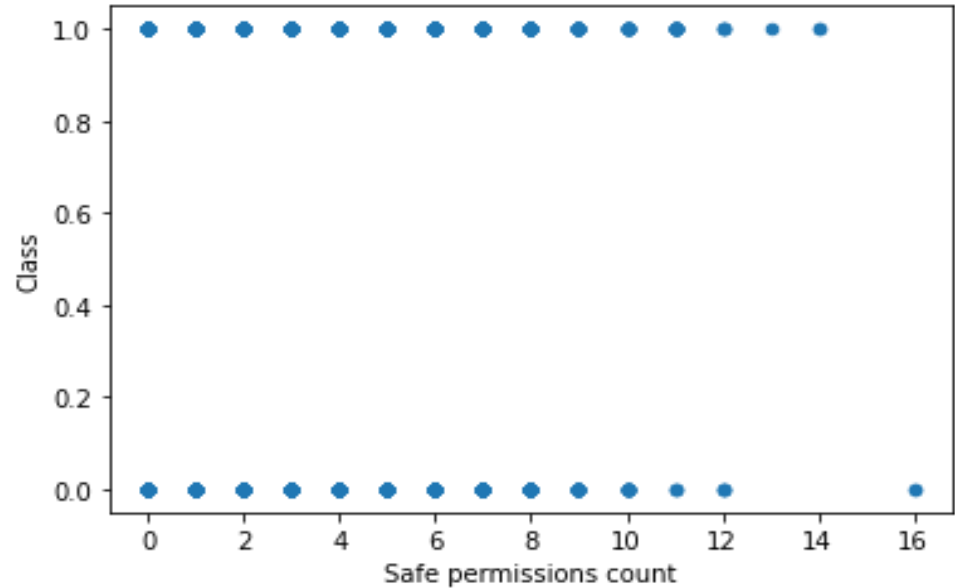
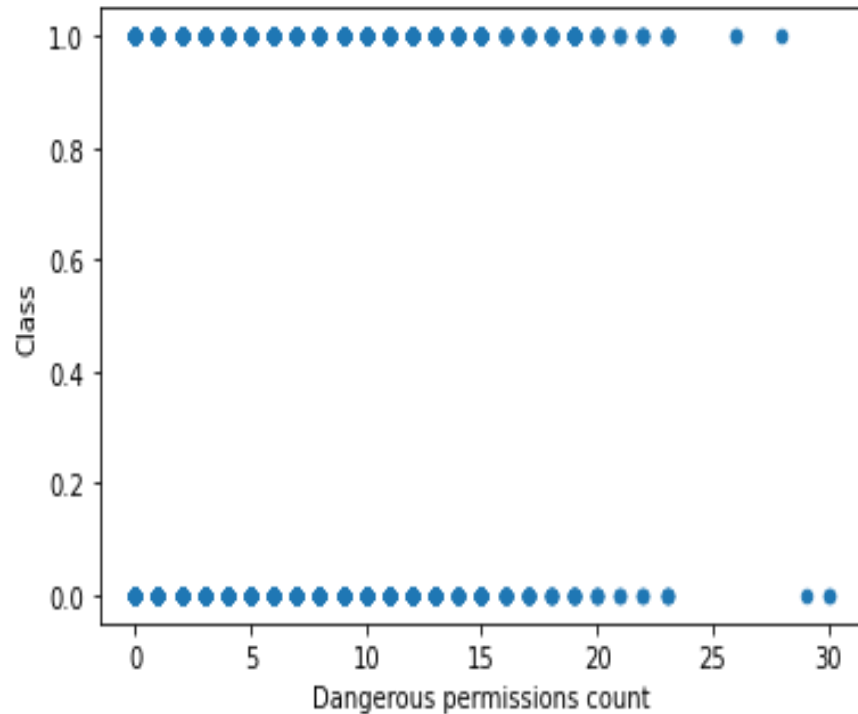
# EDA (Continued..)



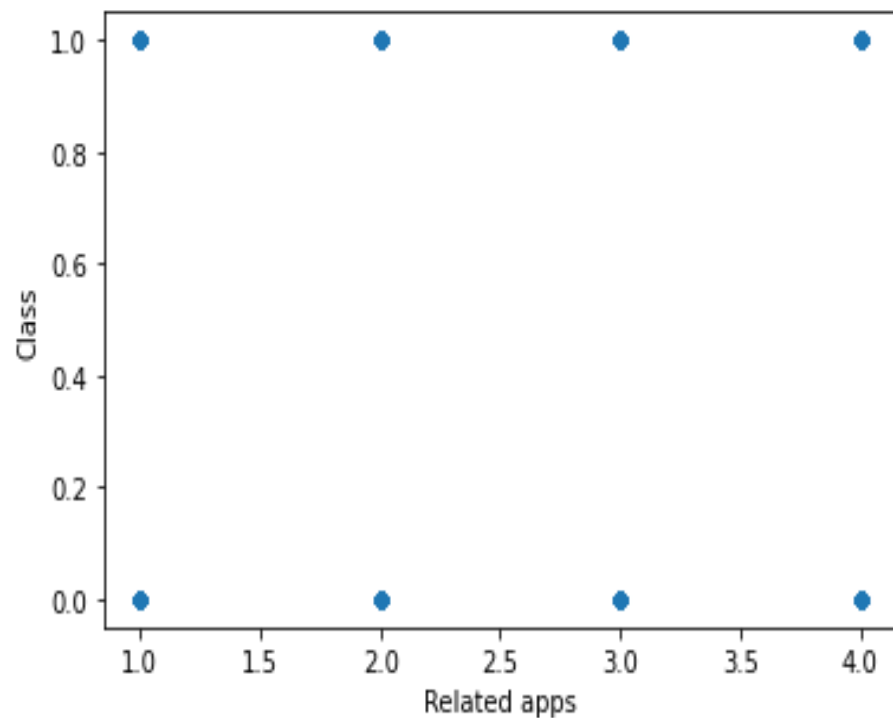
# EDA (Continued..)



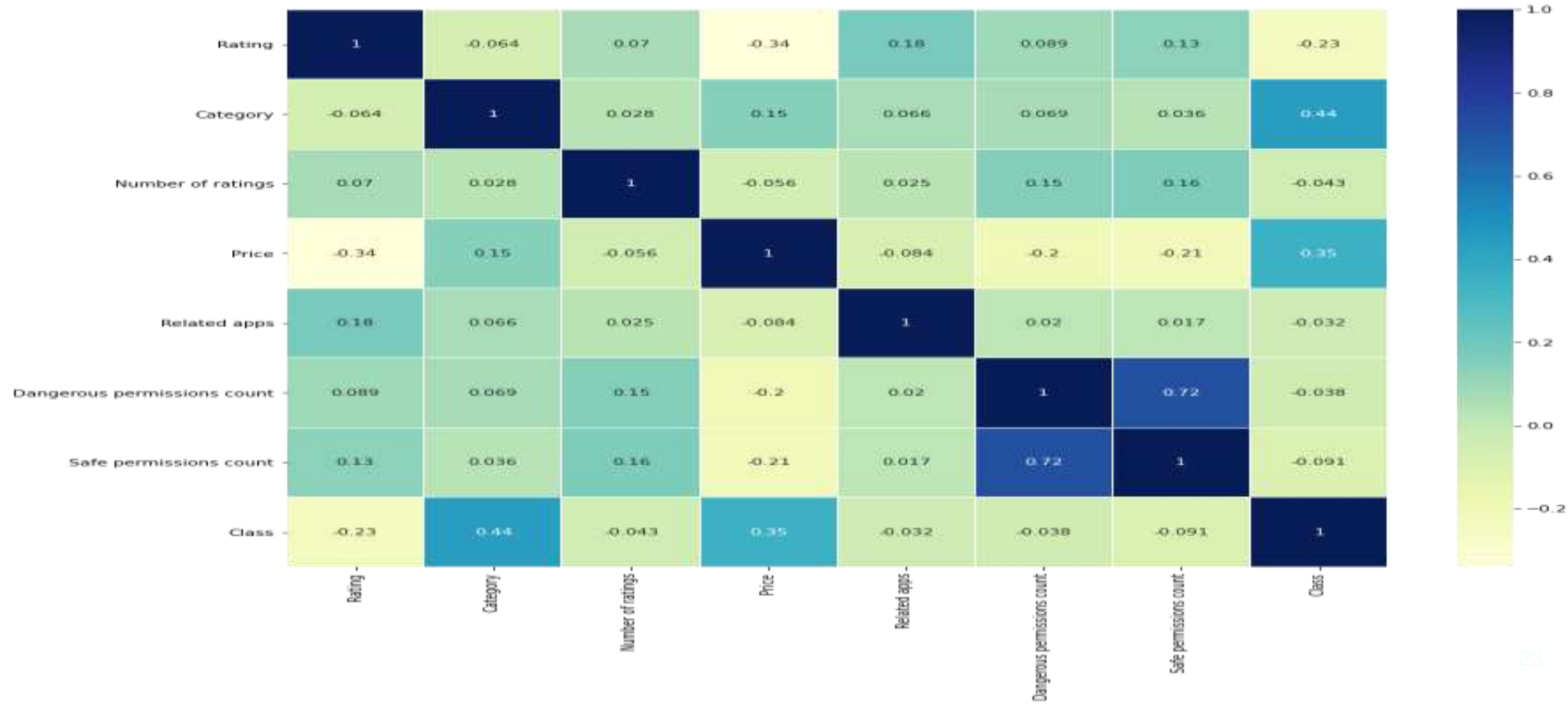
# EDA (Continued..)



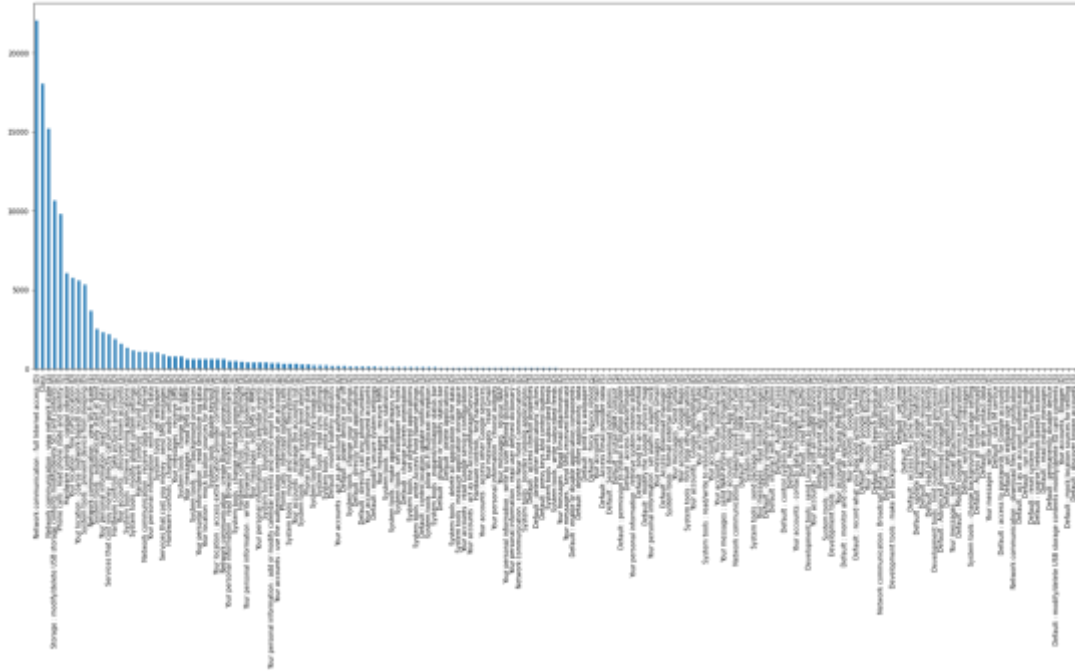
# EDA (Continued..)



# Correlation plot



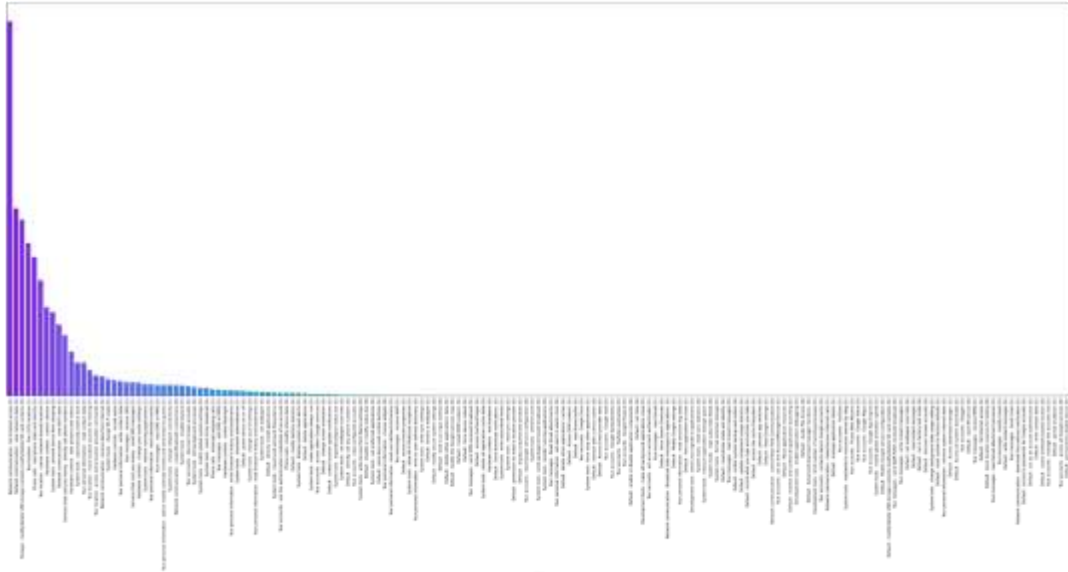
# Number of Apps per Permission



It can be seen from the graph that the permission count is huge overall but only **30%** of permissions are required by majority of apps.



# Impact on Target Variable per Permission



It can be seen from graph there are few permissions out of total **173** permissions which has impacted our target variable to a **malware** or **benign**.

permissions out of total 173 permissions which has impacted our target variable to a malware or benign.

# Playing around with Text Data!



## Description

Minecraft Pocket Edition starts with a random stage. You'll find yourself on a chaotic land in the middle of the ocean, surrounded by mountains, valleys, trees, and animals. In survival mode, the target becomes more vital as the sun sets.

[com.mojang.minecraftpe.demo](http://com.mojang.minecraftpe.demo)

*Can Text Columns be Significant?*



# Let's Clean them all!

```
df.isnull().sum().sort_values(ascending=False)
```

```
Related apps          720
Dangerous permissions count  201
Description            3
App                    1
Default : read phone state and identity (S)  0
```

**App Column: 1 Missing Value**  
**Description: 3 missing values**



	App	Package	Category	Description	Rating
18470	Comic Books	com.eddie.comic_reader	0.125000	NaN	3.6
21129	Stop Watch	dxp.nandalky.stopwatch	0.609977	NaN	3.5
26148	Pedometer ***NEW***	com.lexapps.pedometer	0.468815	NaN	3.0

```
df.at[18470,'Description'] = 'Comic Reader is a next-gen comic reader'
df.at[21129,'Description'] = 'Stopwatch and Timer is a simple, easy and'
df.at[26148,'Description'] = 'The best pedometer app and step counter'
```

**Description Column Text Cleaning!**

# Text Preprocessing!

## 1. CLEANING

- Description: Removed HTML tags
- Package: Separated words from APKs
- All Columns: Only characters selected by regex
- All words to lowercase
- Merged text columns

## 2. STOPWORDS

- Removed Stop words
- Normal english words & problem specific (app, android)

## 3. TOKENIZATION

- Splitted sentences to tokens
- Used word\_tokenize from nltk

## 4. STEMMING

- Transformed words to roots
- Used Snowball Stemmer

*Everybody stand back, I know regex expressions!*

# Time to Model..

Vectorization

Dimensionality Reduction

Classification Model

Evaluation & Insights

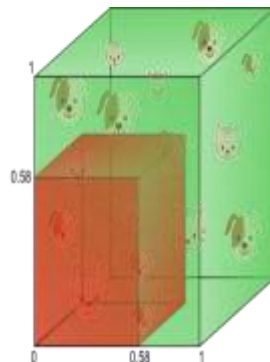
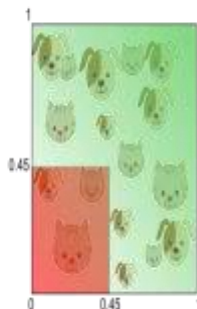
TFIDF Vectorizer

$$\text{tf-idf} = \text{tf} \times \text{idf} \quad (1)$$

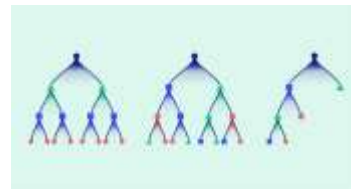
$$\text{idf}(t) = \log \frac{n+1}{\text{df}(d,t)+1} + 1 \quad (2)$$



PCA



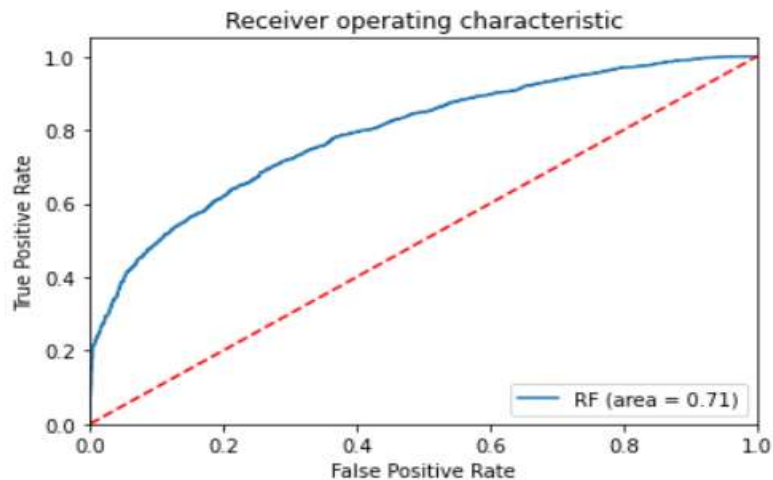
XGBoost via Bayes Search



What could we infer?



# Evaluation & Insights: Surprised!



ROC Curve

	precision	recall	f1-score	support
0	0.56	0.72	0.63	1864
1	0.83	0.71	0.76	3598
accuracy			0.71	5462
macro avg	0.69	0.71	0.70	5462
weighted avg	0.74	0.71	0.72	5462

Classification Report

fingers crossed



*Definitely not bad for a text column based Prediction! Let's see...!*

# Should we or shouldn't we?

## -A Dilemma

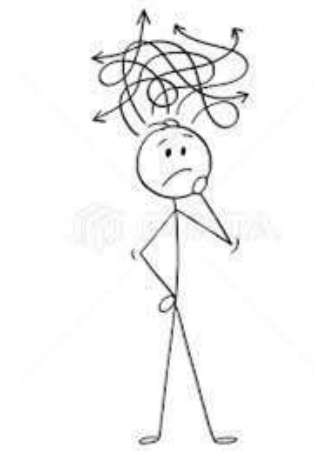
Class Derived\_Prob\_Text

0	0.651585
0	0.756145
0	0.365056
0	0.838322
0	0.921213
...	...
1	0.097495
0	0.810635
1	0.208205

Target Class and Derived  
text Column probability for  
the class

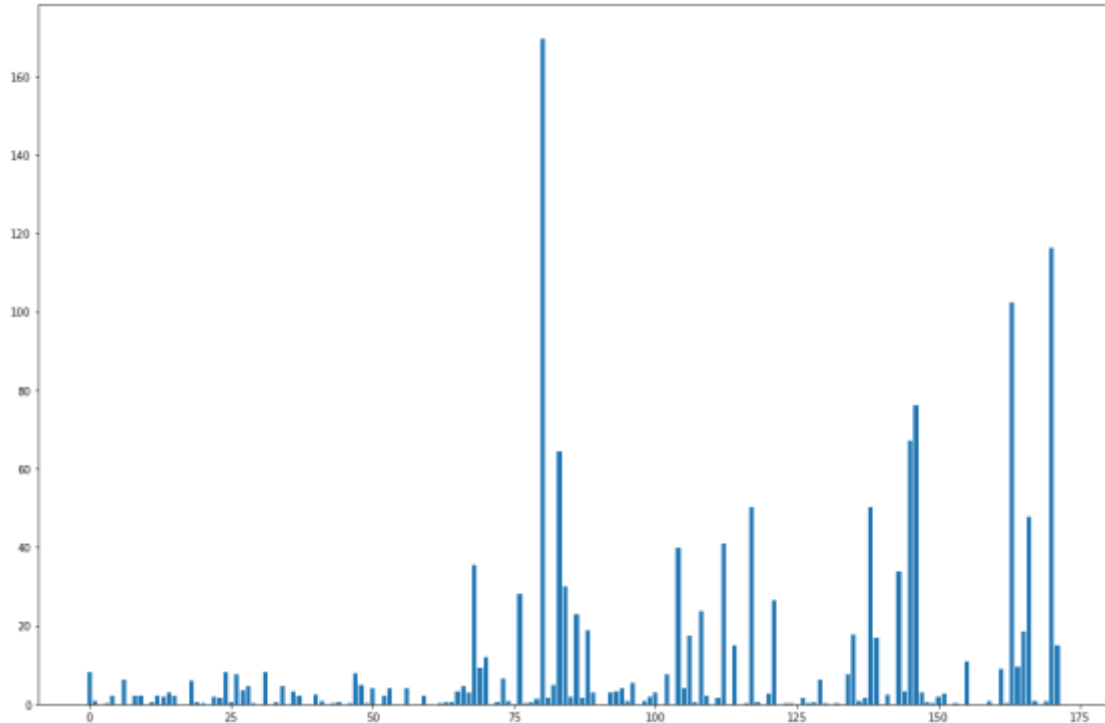
How about we have a derived column from NLP  
Model?

*Let's use a probability score derived from the Best  
NLP Model for a class!*



How about a  
Hybrid  
Model???

# Mathematics to the rescue: 1. Chi-Square Feature Selection

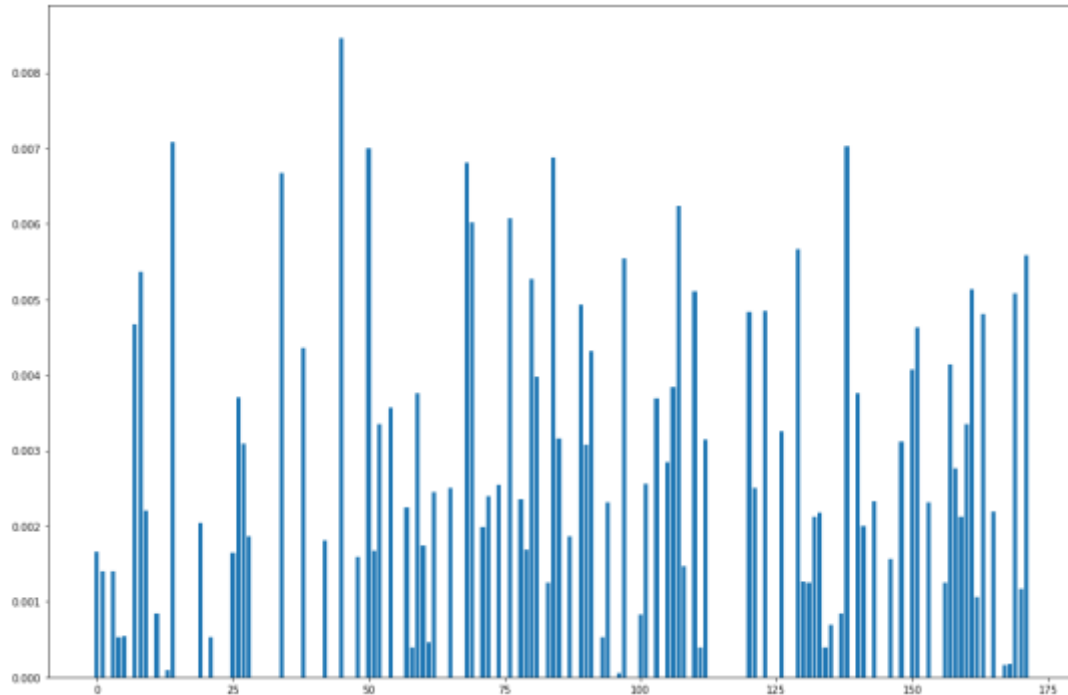


Using Chi-Square Test for Using Chi-Square test for binary categorical variable we did feature selection for permission columns.

binary categorical variable we did feature selection for permission columns.



## 2. Mutual Information Feature Selection

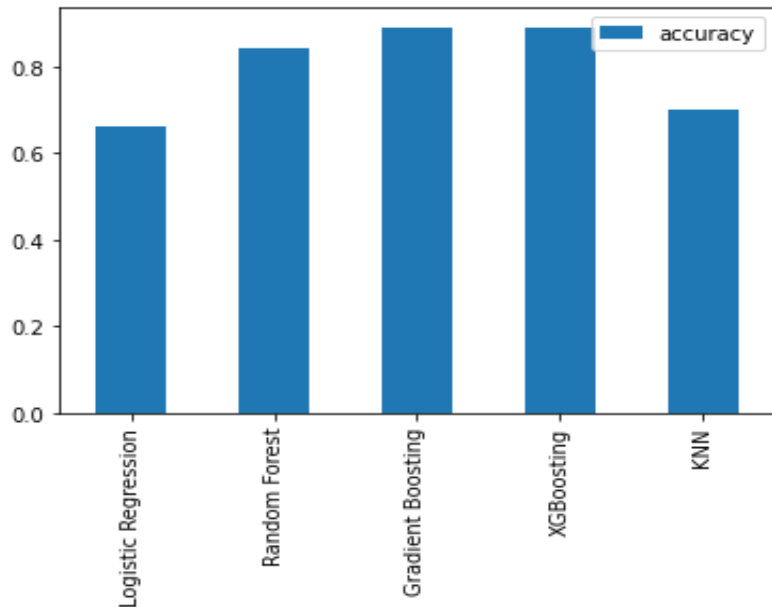


Feature selection for  
**permission columns**  
using **mutual**  
**information**.

# Finally!

- **Text Columns ----> Probability Derived Column**
- **Permissions Columns:** Removed columns with **0** impact on Target
- **Category ----> Mean Encoded**
- **Related App -> Count** of Related Apps
- **Price -> Imputed for mean price** (null values)
- **Number of Ratings**
- **Ratings**

# Let's start Modelling!



*Accuracy Comparison for Different Models*

**5 Models:** XGBoost, Random Forest, GBM, Logistic Regression, KNN with and without Text Derived Columns



# Best of all worlds!

**XGBoost:** Hyper parameter tuning using Random Search CV

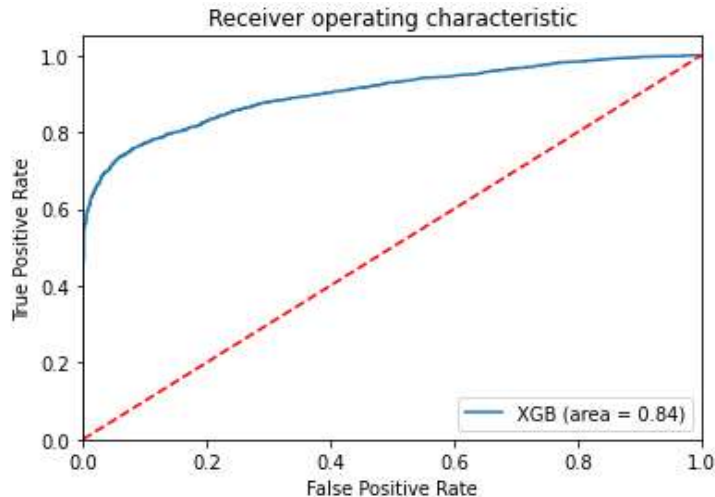


```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,  
              colsample_bynode=1, colsample_bytree=1, gamma=0,  
              learning_rate=0.16777466758976015, max_delta_step=0, max_depth=2,  
              min_child_weight=1, missing=None, n_estimators=160, n_jobs=1,  
              nthread=None, objective='binary:logistic', random_state=0,  
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,  
              silent=None, subsample=1, verbosity=1)
```

# Evaluation of Best Model Without Text Column

	precision	recall	f1-score	support
0	0.65	0.93	0.76	1832
1	0.95	0.75	0.84	3630
accuracy			0.81	5462
macro avg	0.80	0.84	0.80	5462
weighted avg	0.85	0.81	0.81	5462

*Classification Report of XGB Without Text derived Column*



*ROC Curve of XGB Without Text Derived Column*

*So, does adding a new  
derived Column like **Text**  
**derived probability score**  
really make a difference?*

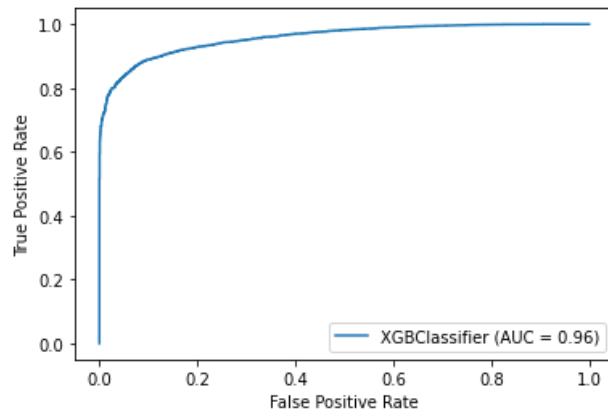


# Surprise! Surprise!



	0	1
0	1936	340
1	404	4148

Confusion Matrix



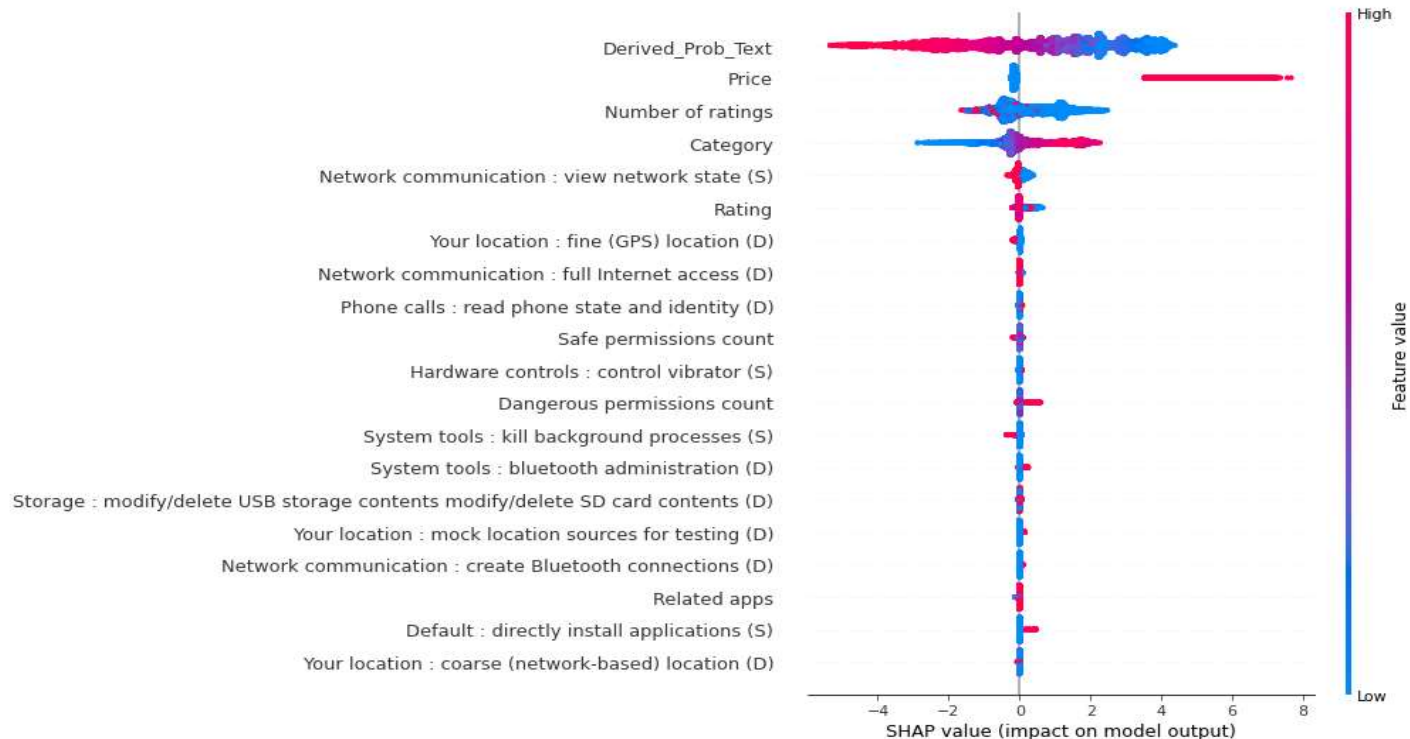
ROC Curve for XGBoost with Text Derived Column

	precision	recall	f1-score	support
0	0.83	0.85	0.84	2276
1	0.92	0.91	0.92	4552
accuracy			0.89	6828
macro avg	0.88	0.88	0.88	6828
weighted avg	0.89	0.89	0.89	6828

Classification Report

*Adding a **Text Derived Column** from NLP increased the overall F1 score from **81%** to **89%**!*

# What's Important?





# If only we had more time: Future Scope!

- **Individual probability** for each Text Column via NLP Modelling
- **Handling Outliers** well
- Deep Learning Using Transformers (BERT, RoBERTa etc.)
- Improve overall Accuracy of the model
- Deploy the Model on an App
- More Research on the Use Case and domain



# Conclusion

- Performed **EDA** and **Cleaned Dataset**
- Univariate & multivariate **analysis**
- Visualised Data, inferred **insights**
- **NLP Model** for Text Column
- **Hybrid Model** : Power of NLP with XGBoost!
- Classified **92%** of Malware Apps & **82%** of Benign Apps!
- Identified Future Scopes



# Suggestions

***“Torture the data, and it will confess to anything.”***

*-Ronald Coase, Nobel Prize winner*



# Together **Everyone Achieves More!**



Time for Q&A!!

