E-Health Methods and Applications: Project Report Part I GROUP 13

DATASET CREATION

The aim of our work is to create a tool that collects and analyzes existing serious games for children in the Play Store. The starting point was to create a starting database with as many apps as possible, that will be then filtered to obtain a dataset of serious games only.

A first database was generated thanks to the use of Selenium WebDriver and the Play Store Scraper library, two tools available in Python. The first one drives the browser natively, as a user would, in the Play Store looking for apps related to learning disorders and educational games automatically through the search bar, extracting the *appIDs.* This feature was then used by the Play Store Scraper library to find the rest of the information needed (as *description, category* and so on).

Then, another Google dataset was downloaded from Kaggle and it was filtered by category, preserving *puzzle, education, educational* and *trivia* to start cleaning it as much as possible.
The next step was to enrich the Google dataset with some missing, useful information such as *description* and *genreID* which were yet not available on Kaggle.
In the end, the information coming from the two different sources were gathered into a CSV file.

	Title	App ID	Category	Rating	Installs	Content Rating	Descriptions	GenreID
0	Block Fill: Puzzle Game	com.mailbeegames.casual.brain.line.block.puzzl...	Puzzle	4.4	10,000+	Everyone	Block Fill is a very fun and challenging game....	GAME_PUZZLE
1	Calculus Tutorial 1: Introduction	com.RaySemiSoft.CalculusT1	Education	0.0	100+	Everyone	In this tutorial, we first define what we mean...	EDUCATION
2	Chemistry Mission	academy.crow.chemistrymission	Educational	0.0	50+	Everyone	3 Game Modes: Organic Chemistry Inorganic Ch...	GAME_EDUCATIONAL
3	RACE ACADEMY	co.davos.snqkw	Education	0.0	100+	Everyone	RACE ACADEMY is an online platform for managin...	EDUCATION
4	Jenifa spells 2	com.jenifaspells2.game	Puzzle	0.0	500+	Everyone	Jenifa spells 2 is an educational and entertai...	GAME_PUZZLE

Figure 1 View of the head of the database

RELATED SERIOUS GAME KEYWORDS

We created a list of keywords associated with serious games for kids. This was done by analyzing the literature and going through the description of some apps considered as a good example of serious games that we were looking for. So, we created a txt file with a list of 173 words that seemed relevant.

In order to rate the apps in the dataset, we analyzed their description using our dictionary. We defined four variables:

- *number of words found:* a variable that counts the number of keywords from the dictionary found in the description.
- *number of words description:* a variable that counts the number of all words in the description for each entry.
- *percentage of words:* a variable that assigns a percentage according to the formula (number of words found/number of words description) *100.
- *word list:* a storage variable of the words found and their occurrences.

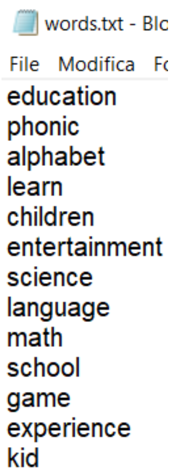


Figure 2 Example of words.txt file

word	score
art	1.000000
serious	0.774059
play	0.686192
age	0.686192

Figure 3 Example of normalized weight of words

To give appropriate weight to each of the words present in our txt file, we developed an algorithm to find their presence in the literature and verify their relevance. We went through and scanned the abstract of 250 papers published on the PubMed website. The weight of each word was defined based on the number of occurrences in the literature and the values obtained were normalized between 0 and 1, with the help of the MinMax Scaler function.

Then, an *AppScore* was computed for each app, using the following criteria:

$$S = \sum_{i=0}^{n} w_i * X_i$$

n: number of words found, w_i: weight of the i-th word, X_i : number of occurrences of the i-th word The higher the score obtained, the more likely the app is deemed to be a serious game.

Category	Rating	Installs	Content Rating	Descriptions	GenreID	numberOfWordsFounded	numberOfWordsDescription	percentageOfWords	wordsList	AppScore
Puzzle	4.4	10,000+	Everyone	Block Fill is a very fun and challenging game....	GAME_PUZZLE	12.0	127.0	9.45	{'game': 3, 'puzzle': 1, 'fun': 2, 'play': 3, ...	3.2887

Figure 4 Added variables in the database

ML APPROACH FOR CLASSIFICATION

In order to find what are the important features able to differentiate among the two classes (serious game or not), we decided to use a Machine Learning approach. Therefore, starting from our dataset, we randomly selected a significant number of apps (over 550) and then we manually classified them (1 = serious game, 0 = others). This new dataset was considered to be an adequate reference in which 110 apps were identified as serious games.

The reference dataset was split into training and test sets (70% - 30%). The features that we decided to consider are: *ContentRating, GenreID, Category, numberOfWordsFound, numberOfWordsDescription, percentageOfWords, wordsList.* The first three were transformed into dummy variables, while *wordList* was substituted with the number of the items present in the list itself. As a Classification Algorithm, we decided to adopt the Decision Tree, due to its simplicity and clarity of the results.
We applied a "GridSearch" to test different parameters (such as *minimum number of splits, max depth* of the tree and so on) and select the best ones to achieve our goal.

The trained model was applied to the complete dataset to identify the serious games and then we selected the apps that were predicted as *1* to compose our final dataset.

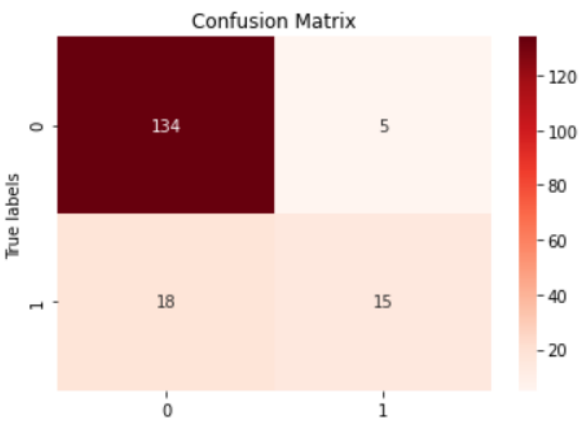
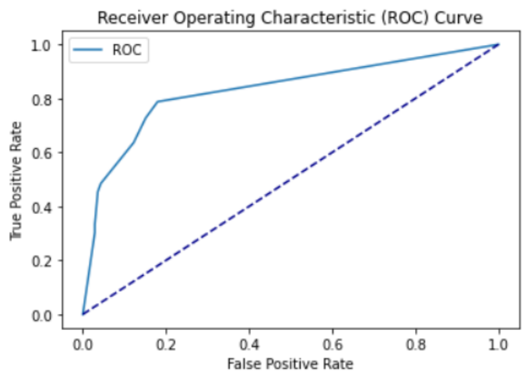


Figure 6 Confusion matrix of our model



AUC: 0.8295

Figure 5 ROC curve of our model

	precision	recall	f1-score	support
0	0.88	0.96	0.92	139
1	0.75	0.45	0.57	33
accuracy			0.87	172
macro avg	0.82	0.71	0.74	172
weighted avg	0.86	0.87	0.85	172

Figure 7 Performance's metrics of our model

CONCLUSIONS

Our results could be easily improved by enlarging the dimension of the reference dataset, to have more data to train the model. Also, another step further can be done by using Natural Language Processing to analyze the information from the app description, which we couldn't use as a feature in our machine learning approach. However, instead of manually selecting parameters by trial and error, we preferred to use the Decision Tree algorithm, to have a more reliable tool in order to maximize the information gain. In the end, starting from a dataset of over 2 million apps, then becoming 296967 after the first filtering, we finally retrieved a dataset of 9596 serious game applications.