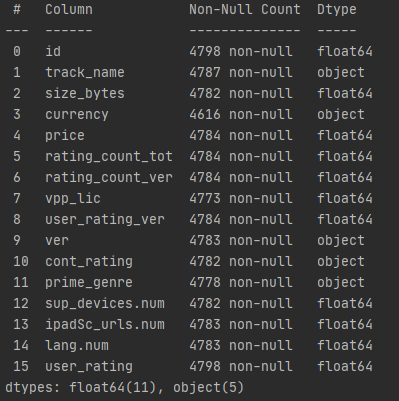
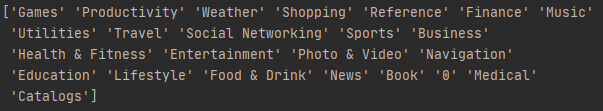
**Milestone 1 Report**

* **Dataset Analysis:**
* Rows count = 4801 entries
* Columns count = 16 columns



* Necessary categorical features:

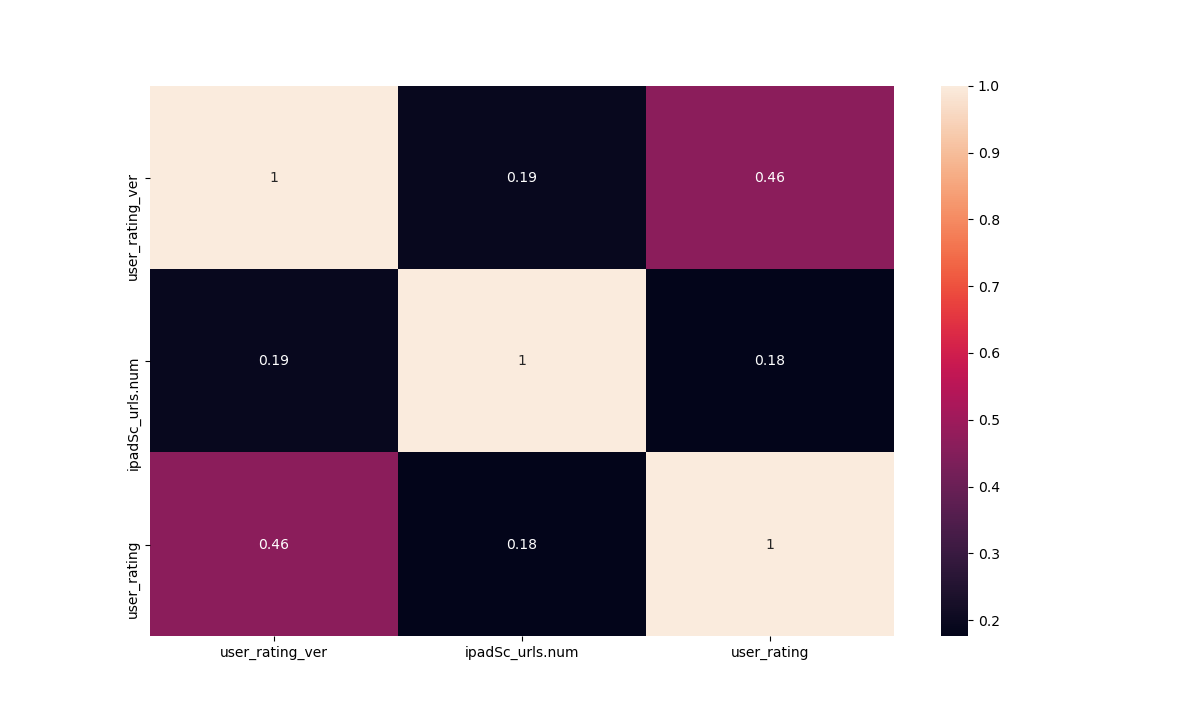
1. ‘prime\_genre’ with unique categories:



1. ‘cont\_rating’ with unique categories:



* Data correlation:



As shown in the correlation heatmap, the effect of each feature on the others is somehow poor, but the most effective features on **‘user\_rating’** is:

1. **‘user\_rating\_ver’**
2. **‘ipadSc\_urls.num’**

So, some features will not be needed to be used in the model like:

1. Id
2. Track\_name
3. Currency (It is all the same value)
4. Version

As they do not strongly affect the **‘user\_rating’.**

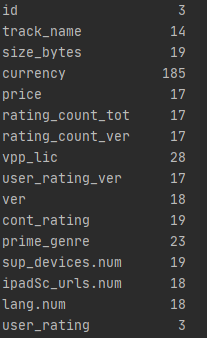
* **Pre-processing techniques:**

1. **Discarding unnecessary features:**



Usually there are some data that are not useful to the machine learning model. It does not have an effect on the desired prediction, so it must be dropped in order to have an efficient model.

1. **Checking out the missing values:**



The concept of missing values is important to understand in order to successfully manage data. If the missing values are not handled properly, then the model may end up drawing an inaccurate inference about the data. Due to improper handling, the result obtained by the model will differ from ones where the missing values are present. It can be handled by a lot of ways. Here, we will delete a particular row if it has a null value for a particular feature.

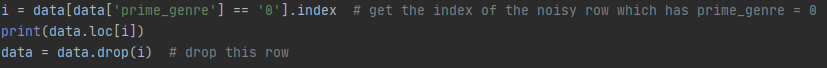
And this way is suitable as the dataset has a large number of samples so it will not be highly affected.



And for the feature ‘user\_rating\_ver’ as it has a good effect on the ‘user\_rating’, its null values will be replaced with the median of the column, they will not be dropped from the data.



1. **Cleaning data with unreasonable values:**

In the categorical feature **‘prime\_genre’**, a category with value **‘0’** was found and it does not have a meaning among the rest of the categories. So, the records with this value should be dropped.

1. **Process the categorical data:**

Since machine learning models are based on mathematical equations and you can intuitively understand that it would cause some problem if we can keep the categorical data in the equations because we want only want numbers in the equations.

**So, we need to encode the categorical variables with numeric values:**

* **cont\_rating:** will be encoded using label encoder. Label encoder is an object which is I use to help us in transferring Categorical data into numerical data. Next, I fitted this object to the column **‘cont\_rating’** of our matrix X and all this return it encoded. It encodes target labels with values between **0** and **n-1** classes.

****

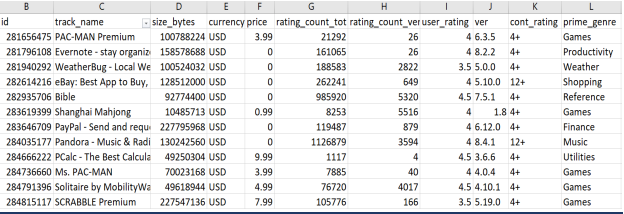
* **prime\_genre**: will be encoded using One-Hot Encoder. It’s one that takes the value 0 or 1 to indicate the absence or presence of some categorical effect that may be expected to shift the outcome. Instead of having one column, we are going to have n columns (n = #classes).

****

**make\_column\_transformer** is a function in **sklearn.compose** that’s used to perform some operation on a specific columns in the given matrix. It takes the operation object and the column name as parameters. **(remainder = ‘passthrough’)** means that you only edit the given column and keep the rest as they are. Then the returned object will be activated through fit\_transform that takes the data matrix and returns it after performing the operation.

1. **Feature Scaling:**

It is a method to limit the range of variables so that they can be compared on common grounds. As we see in the dataset below, there are features like **‘size\_bytes’** and ‘**rating\_count\_tot’** that have large different scaled ranges. If feature scaling is not done, then a machine learning algorithm tends to weigh greater values, higher and consider smaller values as the lower values, regardless of the unit of the values.



The used technique for features scaling in our model is **Standardization**. It is a very effective technique which re-scales a feature value so that it has distribution with 0 mean value and variance equals to 1. By importing **StandardScaler** from **sklearn.preprocessing**:



1. **Splitting data into training and testing sets:**

Generally, we split the dataset into 70:30 ratio. It means that 70 percent data take in train and 30 percent data take in test. However, this Splitting can vary according to the dataset shape and size.



**x\_train:** is the training part of the matrix of features.

**x\_test:** is the test part of the matrix of features.

**y\_train:** is the training part of the label values.

**y\_test:** is the test part of the label values.