Task3

Asal Rahbari – 99222044

1. **Loading Dataset**: First step is always loading the dataset, in this section after loading our dataset we can use available methods to gain more information about the features of our dataset like getting the list and types of columns, comprehend some characteristics just by observing the data and so on.
2. **Handling Null Values**: So in this section I searched for any null values in the whole data frame, but there were none. There’s something that we have to pay attention here, that this simply means that there aren’t any empty cells in our table. There still might be some values that are meaningless like “Import”, in other words it may seem that this data frame doesn’t have any null values, but deep down it may have. For now, we move forward but keep this as a kind of caution in mind.
3. **Encoding Categorical Columns:**
4. Ratings: This feature is for storing the number of customers that reviewed for a specific book.

'5 customer reviews'

As you can see this feature contains a numerical part, which actually is the important part of it and it contains all of the information we need. So we can just easily extract the numerical part of the values and store them in a new column called ‘New\_Ratings’.

1. Reviews: This feature shows that what score each book got out of 5.

'4.9 out of 5 stars',

This one also has a numerical part which matters and the other part doesn’t really play a significant role here. So we extract the score from values.

1. Genre: By observing this feature and book category we can easily realize the fact that genre is somehow the subset of book category. So, it’s basically book category with more details. Since the accuracy of the model isn’t really a matter for us in this task and we still have book category to use it in our training data, considering that fact that encoding genre is not that easy because of its large number of unique values, we just drop this column.
2. Edition: There are two main factors that this feature contains: The date of publication and the format of each book.

The way that the values are represented though, isn’t the kind that we want. By thinking about the relation between these two features and the price of a book, we can clearly see that most effective features must be the format and the year of publication, the month and day wouldn’t affect the price that much to consider them in training the model.

So, here’s what we do: We extract the format and the year and merge them together.

Now we have a new set of values which is better that the previous one.

Now, if we think of it since we’re looking for converting our categorical values into numerical. We can also separate the year and format and make them two different columns. In this case, the year would be numerical already and we’ll probably have less number of unique values for the format alone than we have now for the combination of these two, so encoding it would be easier as well.

So we define a function with this purpose, but the problem is that we’re facing some errors here and these happen because of the null or meaningless values that appear in this step.

PaperbackImport

So we define the function again in a way that it fills these null values and then store values in two new columns “Format” and “Year\_of\_publish”.

Then for the format column since there are only a few unique values, we easily use one-hot coding.

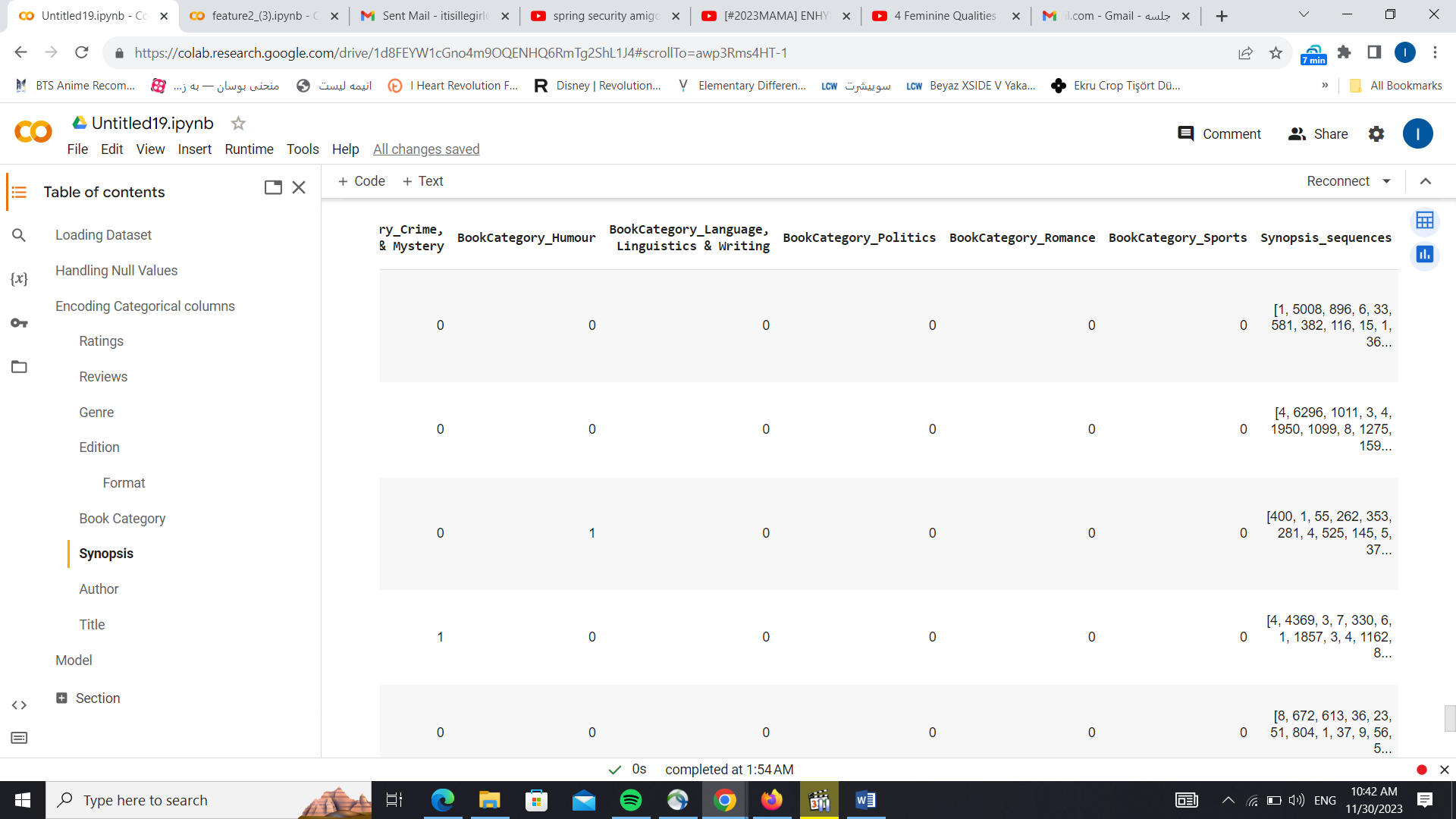
1. Book Category: This feature is also a categorical one and has only 11 unique values, for these kind of categorical columns with few unique values one-hot coding can be applied. So, for book category we also use one-hot encoding.
2. Synopsis: This feature seems a lot different than the previous ones, and the reason is that it’s text and therefore the number of unique values of Synopsis is equal to the number of books (rows of data frame).

In this case we need to find a whole new way for encoding this column, because the simple encoding methods like one-hot, frequency and others won’t work for this one.

For these kind of features we use word embedding: This method creates a vector for each input, by indexing each word in the text. It’s mostly used for encoding natural languages.

Here’s the result for synopsis:

Since the text for synopsis is fairly long , there are many words that need to be encoded and therefore 46323 unique tokens were found for this feature.



g&h) Author and Title: For these features we should use word embedding as well, because even though there are not as long as synopsis, they’re still text.

Author: 4610 unique tokens.

Title: 7512 unique tokens.

1. **Conversion:** Now we need to do something about these tokens that we have.

The reason for this step is that each row has a unique value (due to the fact that each value of these columns was unique). Here for each column we get the length of the longest vector in the column and set new columns for all of the components of the vectors.

The function I defined for this conversion works like this:

If we have a set of vectors: {[1,2], [1,3,4]}

This function will get the length of the longest vector which is 3 and then creates 3 new columns such as column1, column2, column3

And then the new data frame will be:

|  |  |  |
| --- | --- | --- |
| Column1 | Column2 | Column3 |
| 1 | 2 | 0 |
| 1 | 3 | 4 |

Now we apply this function on the features with vector values: ‘New\_Author’ , ‘New\_Title’ , ‘Synopsis\_sequences’.

1. **Model:**

For the model since our target column is continuous, we can use the regression model here.

The End.