Data Analysis on Book's Details

Reyhaneh Saffar_99222064

1. Data Exploration

This dataset is a collection of book's data.

Loading the dataset: I used pandas to read the xlsx file and store it in a data frame. The dataset has 5699 rows and 9 columns.

Here is a description of each column:

Title: The title of the book.

Author: The author of the book.

Edition: The edition of the book.

Reviews: The number of reviews the book has received.

Ratings: The average rating of the book.

Synopsis: A brief summary of the book.

Genre: The genre of the book.

BookCategory: The category of the book.

Price: The price of the book.

The data types of all columns except for the Price column are object. The Price column is of type float64.

2. Data Processing

In the Data Processing phase of the statistical analysis, an exploration of the dataset was conducted to gain insights into its characteristics. The **nunique()** function was applied to the DataFrame, revealing the unique values present in each column. The "**Title**" column, for instance, encompasses 5130 unique titles, reflecting the diversity of books in the dataset. Similarly, the "**Author**" and "**Edition**" columns exhibit 3438 and 3183 unique entries, respectively, highlighting the variety of authors and editions included.
The "**Reviews**" column has 36 distinct values, indicating the presence of a discrete set of review categories. The "**Ratings**" column, with 333 unique values, signifies a broad range of rating scores associated with the books.
The "**Synopsis**" column, being diverse with 5114 unique entries, suggests a wide array of book synopses. The "**Genre**" column, with 335 unique values, underscores the rich spectrum of literary genres covered.
The "**BookCategory**" column, encompassing 11 unique categories, provides insights into the classification of books, while the "**Price**" column displays 1538 unique price points.

- **Null Values:**

  In the data processing phase, handling null values is a crucial step to ensure the integrity of the dataset. The process involves creating a duplicate of the original dataset and identifying columns containing null values. A summary is generated, presenting the number of null values for each relevant column and their respective percentages relative to the total dataset size.

  Upon examination, it is observed that the dataset currently exhibits no null values. While the absence of empty cells is apparent, it is essential to exercise caution and recognize that the lack of null values does not necessarily imply a complete absence of missing information. Some cells might be populated with potentially irrelevant or partial data. Vigilance in identifying and addressing such scenarios is essential for a comprehensive and accurate statistical analysis.

- **Encoding Categorical Columns:**
  In the data processing phase, the categorical columns undergo a specialized encoding process tailored to the unique characteristics of each feature. Conventional methods like one-hot or binary encoding are set aside due to the substantial number of distinct values within these columns. Each column demands a thoughtful selection of a new encoding method.

  Rating Column:

  Focusing on the 'Ratings' column, it contains data in the form of customer review counts. The objective is to distill numerical ratings from this column and introduce a new column in the processed dataframe ('encoded_df'). By parsing and extracting the numerical component from each entry in the 'Ratings' column, a new column named 'Encoded_Ratings' is created. The unique numerical ratings are then explored to offer insights into the range and diversity of these extracted values.
  Finally, after incorporating the new 'Encoded_Ratings' column, the original 'Ratings' column is dropped from the dataframe to streamline and organize the encoded data. The resulting 'encoded_df' provides a transformed representation of the initial dataset, facilitating further statistical analysis without compromising the integrity of the original information.
  -Rating column befor transforming:
  array(['8 customer reviews', '14 customer reviews', '6 customer reviews','13 customer reviews', '1 customer review', '72 customer reviews','16 customer reviews', …,'111 customer reviews']
  -Ratings column after transforming (Encoded_Ratings):
  array(['8', '14', '6', '13', '1', '72', '16', '111', '132', '17', '4', '3', '5',..., '27', '267']

Reviews Column:

During the data processing stage, the 'Reviews' column undergoes a transformation to distill numerical ratings from the original textual representations. The entries in this column consist of ratings given in the format 'X.X out of 5 stars'. The objective is to extract the numerical component and create a new column, 'Encoded_Reviews', in the processed dataframe ('encoded_df').

The unique values within the 'Reviews' column are explored to provide insights into the diversity of ratings present in the dataset. Subsequently, a new column, 'Encoded_Reviews', is created by extracting the numerical portion from each entry in the 'Reviews' column. This new column serves as a numerical representation of the original ratings.

Following the incorporation of the 'Encoded_Reviews' column, the original 'Reviews' column is dropped from the dataframe to maintain a more concise and efficient representation of the encoded data. The resulting 'encoded_df' now includes the transformed 'Encoded_Reviews' column, facilitating further statistical analysis without losing the essence of the initial information.

- Reviews column befor transforming:

array(['4.0 out of 5 stars', '3.9 out of 5 stars', '4.8 out of 5 stars', '4.1 out of 5 stars', '5.0 out of 5 stars',…, '4.7 out of 5 stars', '4.2 out of 5 stars']

- Reviews column after transforming (Encoded_ Reviews):

array(['4.0', '3.9', '4.8', '4.1', '5.0',…, '2.0', '3.7', '3.2']

Genre column:

The data processing involves a decision to eliminate the 'Genre' column from the dataset. This choice is grounded in the presence of the 'BookCategory' feature, which serves as a broader and more encompassing category encompassing the 'Genre' information. Essentially, each genre falls under a specific book category, rendering the 'Genre' column redundant for analytical purposes.

A snapshot of the unique values within the 'BookCategory' column reveals a diverse range of high-level categories, providing a comprehensive overview of the dataset's content.

The subsequent examination of unique values within the 'Genre' column reaffirms its nature as a subset of 'BookCategory,' with each book category comprising various genres.

To quantify this relationship, a grouping operation is performed, showcasing the number of unique genres within each book category. The results illustrate that each book category is associated with a distinct count of genres, solidifying the understanding that 'Genre' is intricately nested within 'BookCategory.'

In light of this relationship, the 'Genre' column is dropped from the dataframe, ensuring that the analytical focus remains on the higher-level categorization provided by 'BookCategory.' The resulting 'encoded_df' reflects a streamlined representation of the dataset, free of redundant information and poised for further statistical analysis.

Edition column:

In the data processing phase, attention is directed towards the 'Edition' column, recognizing that the month of publication may not wield substantial influence on book prices. Instead, the critical factors are identified as the year of publication and the format of the book. To enhance the dataset, the 'Edition' values undergo a transformation by splitting them into two distinct sections: one denoting the book format and the other specifying the year of publication.

A new 'Format' column is introduced to encapsulate information about the book format, and a 'Publication_Date' column is created to isolate and process the year of publication. The publication dates are extracted using a regular expression, converting them to a standardized datetime format for consistency. The unique values within the 'Edition' column are explored to understand the diversity of formats, and the number of unique values for both formats and publication dates is quantified.

The simplification of the 'Edition' column, achieved by excluding the month information, results in a substantial reduction in the number of unique values. With this transformation, the original 'Edition' column is rendered redundant and subsequently dropped from the dataframe. The resulting 'encoded_df' now encapsulates the essential information related to book formats and publication dates, setting the stage for further analysis without compromising the integrity of the dataset.

In the continuation of the data processing for the 'Edition' section, an exploration of the unique values in the 'Format' and 'Publication_Date' columns reveals a diverse range of book formats and publication years. While there were no initially empty cells in the 'Edition' column, the presence of missing values, denoted by <NA> in the 'Publication_Date' column, poses a challenge. To address this issue, a systematic approach is adopted to handle missing values by leveraging the forward-fill method.

Subsequently, dummy variables are created for the 'Format' column using the one-hot encoding technique. This results in a set of binary columns, each representing a distinct book format. These dummy variables are then concatenated with the original dataframe, enriching it with format-related information. To streamline the dataframe and eliminate the redundant 'Format' column, the original column is dropped.
This comprehensive data processing strategy ensures that the dataset is now equipped with a nuanced representation of book formats and publication years. The forward-fill method addresses missing values, and the one-hot encoding technique enhances the dataset's suitability for subsequent statistical analysis without compromising the integrity of the original information.

Book Category Column:

For the 'Book Category' column, a streamlined approach to encoding is applied, given the limited number of unique values. One-hot encoding is deemed appropriate for this feature due to its efficiency, especially given the relatively small number of resulting columns. The unique values within the 'BookCategory' column are explored to comprehend the diversity of book categories present in the dataset.
To implement one-hot encoding, dummy variables are generated for the 'BookCategory' column using the Pandas library. Each unique book category is transformed into a binary column, and these dummy variables are then integrated with the original dataframe. The resulting enriched dataframe now contains additional columns corresponding to each unique book category.
Following this encoding process, the original 'BookCategory' column is dropped to declutter the dataframe and retain only the newly created binary columns representing book categories. The resulting 'encoded_df' provides a more structured and informative representation of the book categories, facilitating subsequent statistical analysis without sacrificing the integrity of the original dataset.

Synopsis column:

In the text data preprocessing stage, the 'Synopsis' column is earmarked for utilization in training a machine learning model. The chosen approach involves vectorization, a pivotal step in converting textual information into a numerical representation compatible with machine learning algorithms. Among the available methods for text vectorization, the decision is made to implement Word Embeddings, an advanced technique that represents words as dense vectors in a high-dimensional space. This method is selected for its capacity to capture semantic meaning effectively.

An initial exploration of the 'Synopsis' column reveals a varied range of lengths for the text data. To ensure consistency, the subsequent processing involves the application of a neural network model using the Keras library. The Tokenizer class is employed to convert the text data into sequences of integers, a fundamental step in preparing the data for subsequent modeling. These sequences are then padded to a fixed length for uniformity in the dataset

The 'word_embeding' function encapsulates this process, taking into account the maximum number of words and the maximum length for padding. The result is a set of unique tokens, and the sequences of integers corresponding to the 'Synopsis' column are appended to the dataframe as 'Synopsis_sequences'.

The integrity of the data is safeguarded by confirming the absence of null values. Finally, to streamline the dataframe for further analysis, the original 'Synopsis' column is dropped. This comprehensive preprocessing ensures that the 'Synopsis' data is now in a suitable format for training machine learning models, paving the way for subsequent analysis without compromising the essential information contained within the dataset.

Title column:

In the process of enhancing the 'Title' column, various methods for text vectorization are considered, each offering distinct advantages. The chosen method is Word Embeddings.

To prepare the 'Title' data for Word Embeddings, the lengths of the text entries are analyzed, revealing a range from 2 to 199 characters. Subsequently, a custom word embedding function is implemented, utilizing the Keras Tokenizer to convert the text data into sequences of numerical values. The unique tokens found during this process provide insight into the richness of the vocabulary in the 'Title' column.

The final output of the Word Embeddings application on the 'Title' column results in the creation of a new column named 'Title_sequences,' containing the numerical representations of the original text. To streamline the dataframe and eliminate the redundant 'Title' column, the original column is dropped. This transformation equips the dataset with a numerical representation of the 'Title' information, enabling more advanced and meaningful analysis without sacrificing the inherent meaning of the text.

Author column:

In the data processing workflow for the 'Author' column, a sophisticated method known as Word Embeddings is employed. The initial exploration of the 'Author' column involves extracting author names and analyzing the lengths of these names. This reveals a range of unique lengths, from a minimum of 2 characters to a maximum of 91 characters, showcasing the diversity within the author names.

The heart of the process lies in the implementation of the Word Embeddings technique.The resulting 'Author_sequences' column in the dataframe now holds the transformed sequences, representing the authors in a vectorized form. This enriched representation enhances the dataset's suitability for advanced statistical analysis, enabling the extraction of nuanced insights from the 'Author' information. The original 'Author' column is subsequently dropped from the dataframe.

3. Conversion:

In the conversion section of the data processing, the primary objective is to address challenges associated with array values in specific columns ('Title_sequences,' 'Author_sequences,' and 'Synopsis_sequences') that could potentially hinder model application. To overcome this, a specialized conversion function is employed, dynamically creating new columns for each element in the sequences. This process ensures a uniform and expanded representation of the array data, facilitating seamless model application.

The initial shape of the dataframe reveals the presence of 36 columns, and after applying the conversion function to the specified columns, the dataframe expands significantly to 2203 columns. This expansion captures individual elements within the sequences, allowing for a more detailed and nuanced analysis.

To streamline the dataframe for model compatibility, the original sequence columns are then dropped. Consequently, the final shape of the dataframe is refined to 2200 columns. This transformation not only addresses the challenge posed by array values in the specified columns but also optimizes the dataset for subsequent model application, ensuring a more structured and conducive format for analysis.

4. Model:

- **Linear Regression:**
  In the model section, a Linear Regression model is applied to predict the 'Price' variable based on the features extracted from the preprocessed dataset. The features and target variable are separated, and the dataset is divided into training and testing sets to assess the model's performance.
  A Linear Regression model is instantiated and trained using the training set. Predictions are then generated on the test set, and the model's performance is evaluated using common regression metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared. The output metrics provide insights into the model's predictive accuracy and overall performance. However, the results displayed indicate some anomalies, such as exceptionally high values for Mean Absolute Error, Mean Squared Error, and a negative R-squared. These anomalies suggest potential issues with the model, such as overfitting or data-related challenges.
  The output of the Linear Regression model evaluation reveals metrics that assess the model's performance on predicting the 'Price' variable:

  Mean Absolute Error (MAE):
  The MAE represents the average absolute difference between the predicted and actual values. In this case, the MAE is exceptionally high, indicating a significant average discrepancy between the predicted and actual prices.
  Mean Squared Error (MSE):
  The MSE measures the average of the squared differences between predicted and actual values. The MSE value is extremely large, suggesting substantial variance in the squared errors, emphasizing potential issues with the model's predictions.
  R-squared (R^2):
  R-squared is a measure of how well the model explains the variance in the target variable. A negative R-squared value is unusual and typically indicates a poorly performing model. In this case, the extremely negative R-squared suggests a considerable discrepancy between the model's predictions and the actual 'Price' values.

The output metrics indicate significant challenges with the Linear Regression model applied to predict book prices. The exceptionally high errors (MAE and MSE) and the negative R-squared suggest that the model might not be capturing the underlying patterns in the data effectively.

- **Random Forest Regressor:**
  In the model section employing a Random Forest Regressor, the process begins with the extraction of features and the target variable from the preprocessed dataset. Features are designated as X, excluding the 'Price' column, while the 'Price' column itself becomes the target variable (y). Subsequently, the dataset is split into training and testing sets using a specified test size and a fixed random state for reproducibility.
  A Random Forest Regressor model is instantiated and trained using the training set. The model's predictive capability is then assessed by making predictions on the test set, and the results are evaluated using three metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared. These metrics provide insights into the model's accuracy, precision, and overall performance.
  The output of the model evaluation is displayed, showcasing the computed values for Mean Absolute Error, Mean Squared Error, and R-squared. In the specific case presented here, the Mean Absolute Error is approximately 293.51, the Mean Squared Error is approximately 291089.59, and the R-squared value is about 0.1884. These metrics collectively offer a comprehensive understanding of the model's ability to predict the target variable based on the provided features. A lower MAE and MSE and a higher R-squared indicate better model performance, reflecting its effectiveness in capturing the variance in the target variable.