Is this book worth a read? Using Machine Learning to Predict Book Review Scores

The Problem/Decision

Have you ever had the urge to pick up a book and have a good read or the desire to escape reality and indulge yourself in a good dose of fiction? Or have you ever picked up a book thinking it was good, only to be disappointed by it? If so, this model may help cure these maladies! Given a book or books that you want to read, our machine learning model will give your book a rating from zero to five, with five being the best, on how good the book is. The objective of our model is to help those debating whether or not to invest time in reading a book by giving them a rating based on a model that was trained on over eight thousand examples.

The Dataset

The dataset that our model was trained on was taken from Kaggle (https://www.kaggle.com/jealousleopard/goodreadsbooks?fbclid=IwAR3b7q25whb_L84fUYEE7epy_6AZDR4vdZpk8iVqBzehKxLOm9S_VZNNV6Y), the first example is given below in figure 1:

In [3]:
```
# Figure 1 - first row of the dataset
df.iloc[[0]]
```

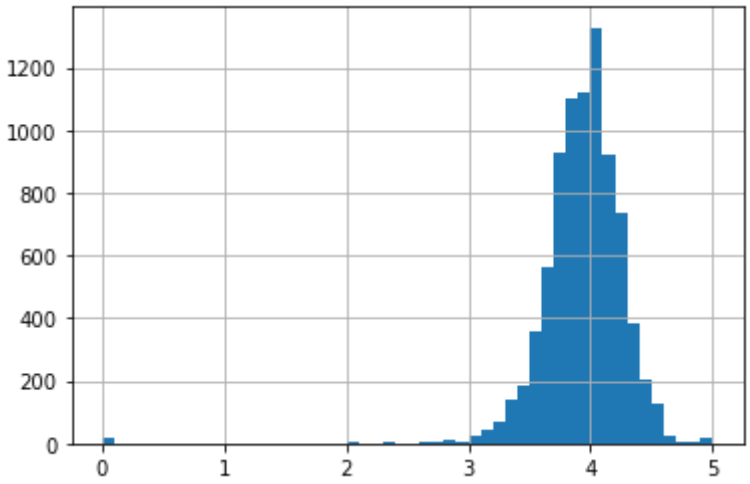
Out[3]:

	bookID	title	authors	average_rating	isbn	isbn13	language_code
0	1	Harry Potter and the Half-Blood Prince (Harry ...	J.K. Rowling/Mary GrandPré	4.57	0439785960	9780439785969	eng

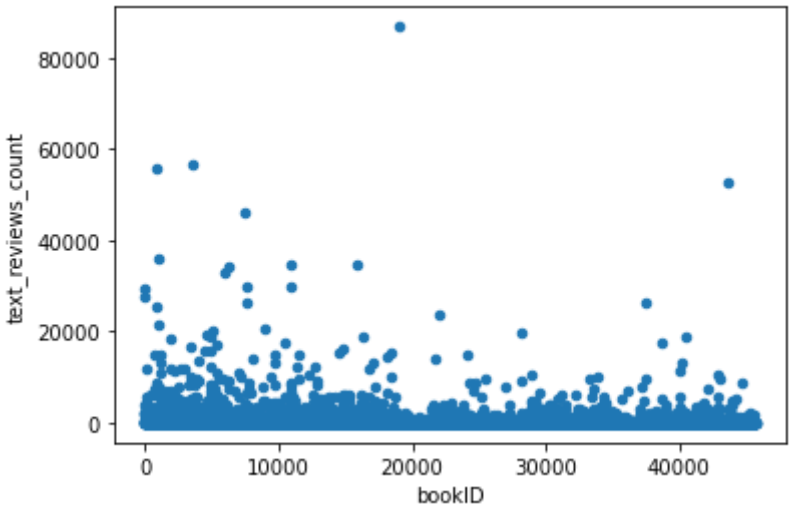
In this dataset, there are 11,123 examples with twelve columns. After splitting the dataset into a training set and test set, we ended up with about eight thousand examples in the training set and about three thousand examples in the test set. Since our target value, 'average_rating', is on a scale ranging from zero to five, no outliers are present because there can be no values outside of zero or five (which can be seen below in figure 2). However, looking at figure 3, there is one outlier in the 'text_reviews_count' features because it received over 80,000 reviews, a staggering amount compared to the amount of reviews the other books received.

In [4]:
```
# Figure 2 - histogram of 'average_rating'
df_train["average_rating"].hist(bins=50)
```

Out[4]: <AxesSubplot:>



In [5]:
```
# Figure 3 - scatter plot of 'text_reviews_count'
df_train.plot.scatter(x='bookID', y='text_reviews_count');
```



The Model

Similar to how the treatment of water involves multiple steps before arriving at your tap, machine learning models also requires multiple steps. To begin, we must preprocess the data to ensure the scores that the model outputs is not cheating (violating the Golden Rule) by peeking at the test set by using preprocessing pipelines and column transformers. The preprocessing pipeline takes in functions like transformers, which pre-process data and "transforms" the data into a useful format that models can score themselves on. In our model, we have two pipelines, one for categorical data, representing columns that can be categorized, and another pipeline for numeric data for columns that are numeric. We then use the column transformer, which takes in these pipelines and transformer functions, such that when it runs, each of the pipelines and transformers are sequentially run so that it never gains any insight to the test set. Once this column transformer is built, it can be passed into a model (regressor or classifier) so that the respective model can run on the data and be scored upon. In our case, we use regression models such as DummyRegressor, Ridge, and

LGBMRegressor as models to score the data since we are predicting the 'average_rating' which is a numerical value.

The Results

After fitting and scoring all of our models, the results showed that the training accuracy of the DummyRegressor model is around -0.0011. It is important to know that the training score is the score on the training data and the test score is the score on the test data; the farther apart these two values are means there is likely more underfitting or overfitting. A negative value for accuracy is very bad and the DummyRegressor can be used as a baseline for the rest of the results. The optimal accuracy for a model would be as close to 1.0 as possible.

The training score for the Ridge model was ~0.0971 and the test score for the Ridge model was ~0.0965. These scores appear to be quite low but higher than the baseline DummyRegressor; since the maximum score is 1.0, then the Ridge model is not considered to be that accurate in predicting the 'average_rating' for a book. For the LGBMRegressor model, the training score was ~0.1545 and the test score was ~0.2236. These scores are higher than both the baseline DummyRegressor model and the Ridge model. However, these scores are still relatively low considering the accuracy is still not even 0.25 (25%) in the highest score. By taking into account the low quality of these scores, it seems that none of the models can accurately predict a book's 'average_rating' but the most effective model of the three tested was the LGBMRegressor model.

Caveats

Firstly, the results may be problematic because of the precision of the 'average_rating' value. The 'average_rating' value is precise up to the hundredths place so therefore if the true 'average_rating' is 4.57 and the Ridge model predicts 4.56 the prediction would be interpreted as incorrect even though a 0.01 difference could be considered 'close enough' based on the context of this prediction.

Secondly, the LGBMRegressor has underfitting (test score > training score) which is an issue when it comes to interpreting scores for the model. This means the LGBMRegressor model needs to be more complex and fine-tuned so the gap between test and training scores is minimized. This is a problem as the LGBMRegressor model has the highest training score in our analysis so it would be useful to eliminate the underfitting in this model so it can be of more use.

Thirdly, the results may be problematic as the provided features in the dataset such as 'title' and 'ratings_count' may not be as useful as originally thought out to be. Some new, possibly useful features like annual sales of a book were not in this dataset however, these new features could help increase the accuracy of the 'average_rating' prediction if they had higher feature importances. Ultimately, the models used to predict the 'average_rating' of a book do not perform greatly and more advanced machine learning techniques may be needed to parse the dataset features on a deeper level.