**Machine Learning with Python**

**GoodReads ML Report**

Prepared by

**Caique Dias**

[**caique.miranda-dias@edu.dsti.institute**](mailto:caique.miranda-dias@edu.dsti.institute)

December 15, 2022

**Instructions for Project Setup**

This report is to explain step by step how to use the project files. You can extract the files to a folder.

Graphical user interface, application

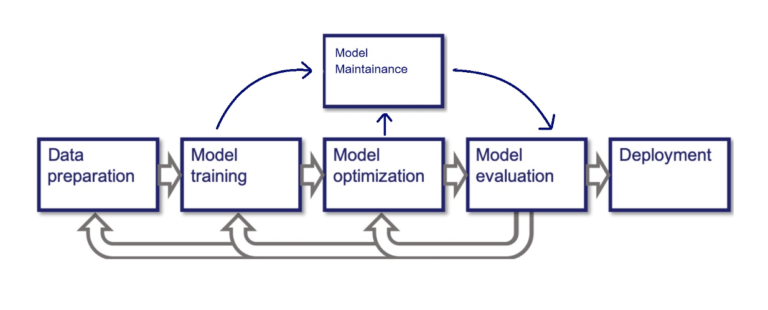
Description automatically generated

* **books.csv :** It’s the dataset that will be used for this project.
* **GoodReads ML Project Report.docx :** It’s this report.
* **Project\_Description\_ML\_DA\_2022.pdf :** It’s the description and requirements of the project
* **ProjectGoodReads.ipynb :** It’s the entire code of the project in Jupyter Notebook. It’s highly recommended to run it with Anaconda distribution of Python versions 3.

You can either upload the files using Jupyter notebook or place these files in the current working directory and then run the notebooks with a Python interpreter.

**Methodology**

In this machine learning project, I will try to follow exactly the methodology below.



* **Data preparation :** It’s the stage at which the dataset will be analyzed and I’ll try to understand it. It’s necessary to arouse my curiosity and ask intelligent questions for queries. This step will compose most of the work, because after analyzing them, I will apply transformations to prepare them for model training.
* **Model training :** in this step, I will import several machine learning models from the scikit learn library. I will use part of the dataset to train these models to fit them.
* **Model optimization :** This step and the one below work together. Because the model evaluation is the main indication that the model is not optimized. Numerous tests and parameter changes were made to improve the model.
* **Model evaluation :** The evaluation is super important to say if all the dataset preparation and the chosen models were appropriate. If the result of the evaluation is bad, we will go back to previous steps.
* **Model maintenance :** It’s the debug step.
* **Deployment :** After the entire process that is carried out in a development environment, this project will be put into production.

**Imported Libraries**

I will start by explaining the code by the libraries used in Python.

* **numpy :** It’s a library which supports the processing of large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions for operating on these matrices.
* **pandas :** In particular, it offers structures and operations for manipulating numeric tables and time series. It’s a fast, powerful, flexible and easy to use open source data analysis and manipulation tool.
* **matplotlib :** It’s a comprehensive library for creating static, animated, and interactive visualizations.
* **seaborn :** It’s a data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
* **sklearn-metrics :** sklearn is a library and metrics is its module. This module implements several loss, score, and utility functions to measure models performance.
* **sklearn-preprocessing-onehotencoder :** preprocessing is a module that offers functions to process the data. onehotencoder transforms each categorical feature with n\_categories possible values into n\_categories binary features, with one of them 1, and all others 0.
* **sklearn-preprocessing-standardscaler :** standardscaler is a function to numerical columns and it standardize features by removing the mean and scaling to unit variance.
* **sklearn-compose-make\_column\_selector :** Create a callable to select columns to be used with ColumnTransformer.
* **sklearn-compose-columntransformer :** Applies transformers to columns of an array or pandas DataFrame.
* **sklearn-compose-pipeline:** The purpose of the pipeline is to assemble several steps that can be cross-validated together while setting different parameters.

**Data Preparation**

\* *Data Analysis*

We'll start by importing the .csv file into Jupyter.

In a first analysis, I realized that one of the columns does not have a practical name, so I changed it to make the job easier.

The project description already provides us with the necessary information to understand what each column is in the data set, but we have to know the type of data that each columns have.

Text

Description automatically generated

There are no nulls on the dataset.

"publication\_date" may be valuable for numerical distribution analysis, especially when years will be extracted.

After that, the analysis is divided between the columns of type numerical and the columns of type object.

For the columns of type numerical, the first question that arises is whether there is any kind of correlation between them.

**Graphical user interface, application

Description automatically generated**

The strongest correlation is between text reviews counts and rating counts, which makes sense. But overall, the result says that there is not much correlation between them and the target column (<< 1) except the number of pages.

Another question that comes up is how the target column is distributed.

**Histogram

Description automatically generated**

To help visualization we will create a column that shows the ranges of the target column.

**A picture containing icon

Description automatically generated**

We can see that there is a very large concentration between 3 and 5 and that results below 2.5 are almost non-existent.

I plot a histogram for all the numerical columns to check their distribution.

**A picture containing shoji, crossword puzzle, clipart

Description automatically generated**

The histograms help us to get an idea that there are many outliers, but I will use the box plot chart to analyze this better.

Chart, box and whisker chart

Description automatically generated

Now looking at the object columns, I'll start with language\_code.

**Chart, pie chart

Description automatically generated**

**A picture containing chart

Description automatically generated**

Looking at the distribution we can see that books of a language other than eng and spa are very few, so I will put them in an 'others' label.

\* *Queries*

What are the top 10 best rated books ?

Text

Description automatically generated

What are the top 10 best rated books with a higher total ratings count than the average ?

Text

Description automatically generated

What are the 10 authors who have more books with average\_rate and ratings\_count longer than average ?

Chart, bar chart

Description automatically generated

What is the top 10 most rated books ?

Chart

Description automatically generated

Top 10 authors who own more books.

Chart, funnel chart

Description automatically generated

What is the top 10 longest books ?

Chart, funnel chart

Description automatically generated

What is the top 15 publishers by number of publications ?

Chart

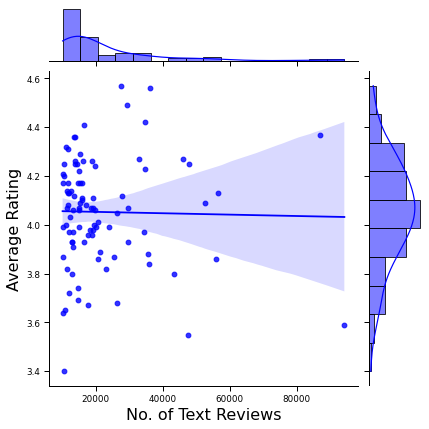
Description automatically generated

What is the bottom 5 poorly rated authors (average\_rate < 3)?

Chart

Description automatically generated

Did the books with more text reviews receive higher ratings?



Did the books with more reviews receive higher ratings?

Chart, scatter chart

Description automatically generated

This graph complements the correlation graph, showing that the relationship between them is almost nonexistent.

Does a second author ever appear as the first author?



\* *Data Cleaning*

The 'title' has many unique results, and it is not possible to observe if it really has relevance in the rating score, so it will be discarded.

I check for duplications and apply get\_dummies to transform the categorical language\_code column into n\_categories binary features, with one of them 1, and all others 0.

I split the data into the training set that is a subset to train a model and test set that is a subset to test the trained model. It’s necessary to check if the average of the samples does not vary too much. The parameter test\_size = 0.20 indicates the percentage of rows that will be used to test the model.

I will also separate the target column from the dataset. X will be the feature columns and y will be the target column.

\* *Data Processing*

The purpose of the pipeline is to assemble several steps that can be cross validated together while setting different parameters. To process the data, I will use pipeline and column transformation.

First, I created two functions to retrieve the numeric and categorical columns because the transformation will depend on this information. For the numerical columns we will use StandardScaler and for the categorical columns we will use OneHotEncoder.

StandardScaler standardize features by removing the mean and scaling to unit variance.The standard score of a sample x is calculated as:

z = (x - u) / s

Where u is the mean of the training samples or zero if with\_mean=False, and s is the standard deviation of the training samples or one if with\_std=False.

OneHotEncoder encode categorical features as a one-hot numeric array.

The input to this transformer should be an array-like of integers or strings, denoting the values taken on by categorical (discrete) features. The features are encoded using a one-hot (aka ‘one-of-K’ or ‘dummy’) encoding scheme. This creates a binary column for each category and returns a sparse matrix or dense array (depending on the sparse\_output parameter). the output of the processing is a sparse matrix. A sparse matrix is a matrix that is comprised of mostly zero values. This is due to the OneHotEncoder.

**Model Training**

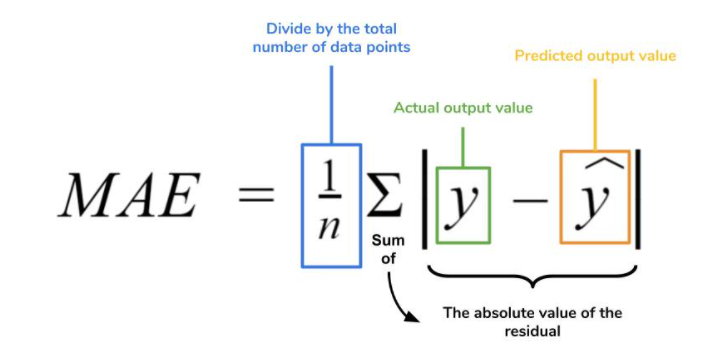
Here we train the model on the prepared or the training data. I will use the following machine learning models:

* **LinearRegression :** LinearRegression fits a linear model with coefficients w = (w1, …, wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation. Since the target column is a continuous value, supervised regression models will be more appropriate in this case.
* **RandomForestRegressor :** A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.
* **SuportVectorRegressor :** Support Vector Regression is a supervised learning algorithm that is used to predict discrete values. Support Vector Regression uses the same principle as the SVMs. The basic idea behind SVR is to find the best fit line. In SVR, the best fit line is the hyperplane that has the maximum number of points.
* **Ridge :** Ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where the independent variables are highly correlated, that’s not this case, but I will se the outputs.

**Model Evaluation**

I will evaluate the models with the following indicators:

* **Mean Absolute Error :** Mean Absolute Error is a model evaluation metric used with regression models. The mean absolute error of a model with respect to a test set is the mean of the absolute values of the individual prediction errors on over all instances in the test set.



* **Residual sum of squares :** The residual sum of squares (RSS) measures the level of variance in the error term, or residuals, of a regression model. The smaller the residual sum of squares, the better your model fits your data; the greater the residual sum of squares, the poorer your model fits your data.

A picture containing letter

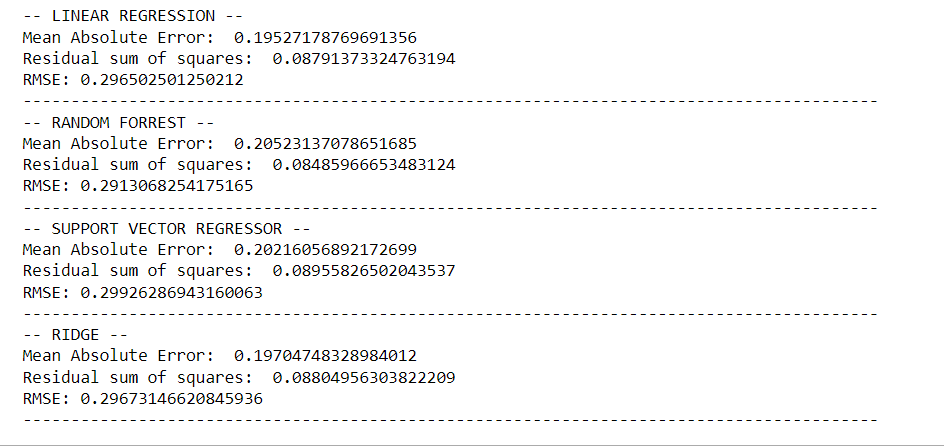
Description automatically generated

* **RMSE :** The root-mean-square deviation (RMSD) or root-mean-square error (RMSE) is a frequently used measure of the differences between values (sample or population values) predicted by a model or an estimator and the values observed. RMSD is always non-negative, and a value of 0 (almost never achieved in practice) would indicate a perfect fit to the data. In general, a lower RMSD is better than a higher one. However, comparisons across different types of data would be invalid because the measure is dependent on the scale of the numbers used.



**Results**

We can observe that the results were quite similar. The best model is the Random Forest model for presenting the lowest values. The fit took a little while to execute, a sign that there is still a lot of optimization work to be done. The data processing, the choice and parameters of the model directly influence this value.



**Conclusion**

We can conclude with this exercise how powerful machine learning algorithms can be. No wonder that many companies work very hard to master them. Prediction is something that can be very valuable financially. Many skills must be honed to reach a very high level as a professional in this area, such as statistical analysis, notions of probability, computing, and mathematics. The models are complex, and a thorough study must be done to understand their parameters and their influence on the outcome.