Journal of Research in Science and Engineering (JRSE)
ISSN: 1656-1996 Volume-6, Issue-11, November 2024
DOI: 10.53469/jrse.2024.06(11).14

A Material Type Prediction Model Based on Machine Learning Technology

Seetaram Nagaraju Naik

Abstract: ***In materials science, traditional experimental and computational approaches require the investment of enormous amounts of time and resources, and the experimental conditions limit the use of these methods. Sometimes, traditional approaches may not yield satisfactory results for the desired purpose. Therefore, it is essential to develop a new approach to accelerate experimental progress and avoid unnecessary waste of time and resources.***

Keywords: Supervised Learning, Classification Problems, Feature Engineering, Scaling, Python, Material Type

1.Introduction

As a data-driven method, machine learning provides reliable and accurate performance in solving problems in materials science. Research concerning the prediction of various properties of materials by machine learning has continued.

In other words, machines can learn from past data and situations, and improve algorithms to make decisions when encountering different and even unknown situations.

The important feature variables used to determine the 'Material Types' are as follows:

- 'Usage Class'
- 'Checkout Type'
- 'Title'
- 'Creator'
- 'Subjects'
- 'Publisher'
- 'Publication Year'

Using these data, we will try to use analytical techniques and ML algorithms to determine the 'Material Type' broadly divided into classes **{BOOK, SOUNDDISC, VIDEOCASS, VIDEODISC, SOUNDCASS, MIXED, MUSIC, CR}.**

2.Materials and Methods

The materials used are usually divided into training and testing types. The training dataset was used for data preparation, and data modelling. The test dataset was used to obtain the best evaluation metrics used for business trade-offs.

Methodologies:

Any classification problem follows the steps below. Each step is important because the final results are dependent on the processes when they are managed well.

Define the problem: Clearly, articulate the problem you want to solve or the goal you want to achieve with ML. Determine if it is a classification, regression, clustering, or any other type of problem.

Gather and explore the data: Collect the relevant dataset for your problem domain. The data were explored to determine their structure, quality, and relationships. Descriptive statistics, visualization, and data pre-processing tasks, such as handling missing values, outliers, and feature scaling, were performed.

Split the dataset: Divide the dataset into two or three parts, typically training, validation, and test sets. The training set is used to train the ML algorithm, the validation set helps in hyper parameter tuning, and the test set is used to evaluate the final model's performance.

Feature engineering: Extract or create meaningful features from the dataset that can improve the ML model's performance. This might involve feature selection, dimensionality reduction techniques such as PCA, or the creation of new features through transformations or domain knowledge.

Select an appropriate algorithm: Based on the problem type, dataset size, complexity, and other factors, choose a suitable ML algorithm. Consider algorithms such as decision trees, random forests, support vector machines, neural networks, or ensemble methods such as gradient boosting or stacking.

Model training: The training dataset is fed into the chosen algorithm, after which the underlying patterns and relationships are learned. The algorithm's hyper parameters (e.g., learning rate and regularization strength) are adjusted to optimize the model's performance. The validation set is used to fine-tune the hyper parameters through techniques such as grid searching or random searching.

Test the model: Finally, evaluate the model's performance on the test set, which provides an unbiased assessment of its generalization capabilities. The model's performance on the test set was consistent with that on the validation set.

Evaluation of the model: Once the model is trained, its performance is assessed using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, or means squared error. The model's performance on the validation set was compared with different hyper parameter configurations to choose the best-performing model.

Iterate and improve: If the model's performance is not satisfactory, revisit the previous steps. To explore alternative

algorithms, perform more feature engineering, collect additional data, or refine the existing approach to improve the model's performance. Iterate until you achieve the desired results.

Deploy the model: Once you are satisfied with the model's performance, deploy it in a production environment to make predictions on new, unseen data. The model's performance is monitored over time and retrained or updated as needed.

3.Algorithms used

TF-IDF with scikit-learn

The text analysis method is called the *term frequency–inverse document frequency method*, often abbreviated *tf-idf*. Tf-IDF is a method that tries to identify the most distinctively frequent or significant words in a document. We specifically learned how to calculate tf-idf scores using word frequencies per page-or “extracted features”.

In this lesson, we will learn how to calculate tf-idf scores using a collection of plain text (.txt) files and the Python library scikit-learn, which has a quick and nifty module called TfidfVectorizer.

```
tfidf = TfidfVectorizer(max_features = 500,
                        ngram_range = (1,3),
                        stop_words = "english")
train_subjects = tfidf.fit_transform(train_data["subjects"].tolist())
```

Code snippets: TF-IDF Vectorizer

XGBoost algorithm

XGBoost is an optimized gradient boosting library that excels in handling tabular data and is widely used for classification and regression tasks. This approach implements the gradient boosting algorithm, which combines multiple weak models to create a more accurate ensemble model.

```
###! pip install xgboost

from xgboost import XGBClassifier

xgb = XGBClassifier()
xgb.fit(X_train,y_train)
xgbpred = xgb.predict(X_test)
print(metrics.accuracy_score(y_test,xgbpred))

    0.8847976495118947
```

Code snippets: XGBoost Classifier

Deep Learning algorithm

Deep learning is a branch of machine learning that is based on artificial neural networks. It is capable of learning complex patterns and relationships within data. In deep learning, we do not need to explicitly program everything. It has become increasingly popular in recent years due to advances in processing power and the availability of large datasets. ANNs are based on artificial neural networks (ANNs), also known as deep neural networks (DNNs). These neural networks are inspired by the structure and function of the human brain’s biological neurons, and they are designed to learn from large amounts of data.

1. Deep learning is a subfield of machine learning that involves the use of neural networks to model and solve complex problems. Neural networks are constructed after the structure and function of the human brain and consist of layers of interconnected nodes that process and transform data.

2. The key characteristic of deep learning is the use of deep neural networks, which have multiple layers of interconnected nodes. These networks can learn complex representations of data by discovering hierarchical patterns and features in the data. Deep learning algorithms can automatically learn and improve from data without the need for manual feature engineering.

```
model.summary()

Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, None, 32)          6400032

 bidirectional_1 (Bidirecti  (None, 64)                16640
 onal)

 dense_4 (Dense)             (None, 128)               8320

 dropout_2 (Dropout)         (None, 128)               0

 dense_5 (Dense)             (None, 256)               33024

 dropout_3 (Dropout)         (None, 256)               0

 dense_6 (Dense)             (None, 128)               32896

 dense_7 (Dense)             (None, 8)                 1032

=================================================================
Total params: 6491944 (24.76 MB)
Trainable params: 6491944 (24.76 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Code sniffets: Deep Learning

LSTM – Long short-term memory

LSTM is wellsuited for sequence prediction tasks and excels in capturing long-term dependencies. Its applications extend to tasks involving time series and sequences. The strength of LSTM lies in its ability to grasp the order dependence crucial for solving intricate problems, such as machine translation and speech recognition. This article provides an in-depth introduction to LSTM, covering the LSTM model, architecture, working principles, and critical role they play in various applications.

```
import tensorflow as tf

es_callback = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=3)

model.fit(train, epochs=41, validation_data = val, callbacks=[es_callback])
```

Code sniffets: LSTM – Long short term memory

4.Plots and Figures

Figure 1 shows the sizes of the trains and tests used for model training and evaluation. During this process, we also decided that the dataset should be balanced in nature.

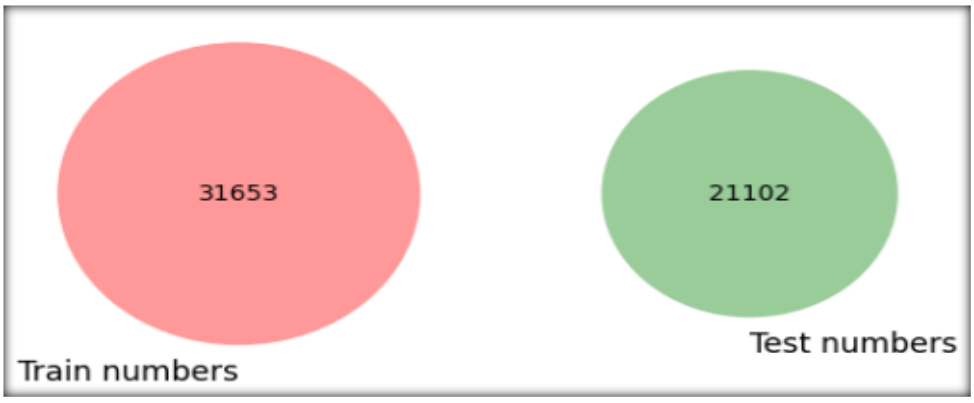

Figure 1: Venn diagram visualizing the train and test Datasets

Figure 2explains the categorical variables used for modelling. As far as definitions are concerned, we can also categorize categorical variables as **binary**, i.e., categorical variables with only two categories. Some even talk about a type called "**cyclic**" for categorical variables. Cyclic variables are present in "cycles", for example, days in a week: Sunday, Monday, Tuesday, Wednesday, Thursday, Friday and Saturday. After Saturday, we have Sunday again. This is a cycle. Another example would be hours in a day if we consider them to be categories.

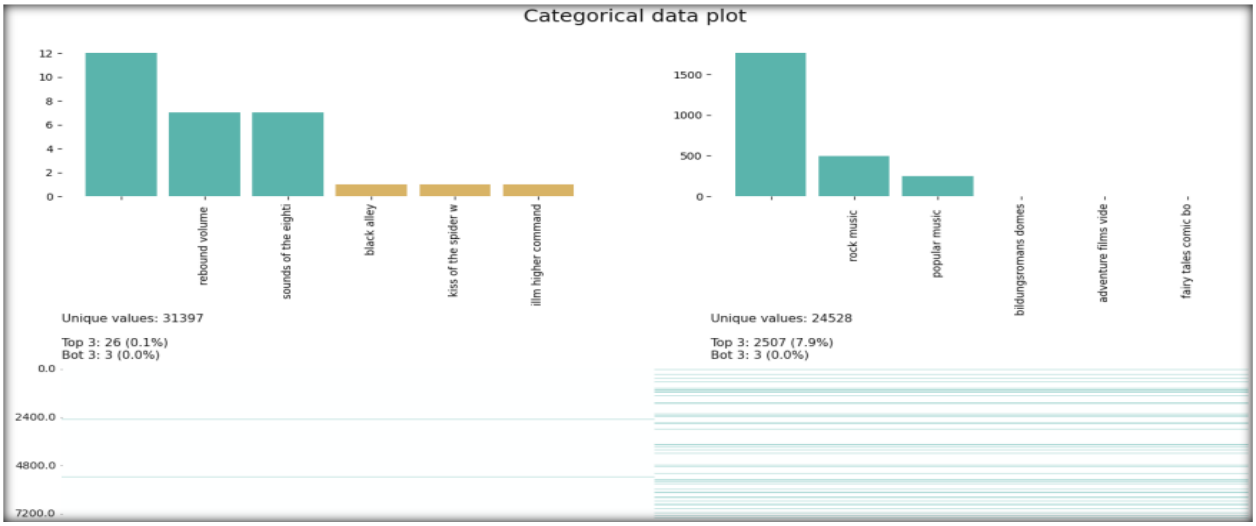

Figure 2: Categorical data plot explaining the categorical variables used as independent data

Figure 3 explains the numerical distribution of the data along an axis.

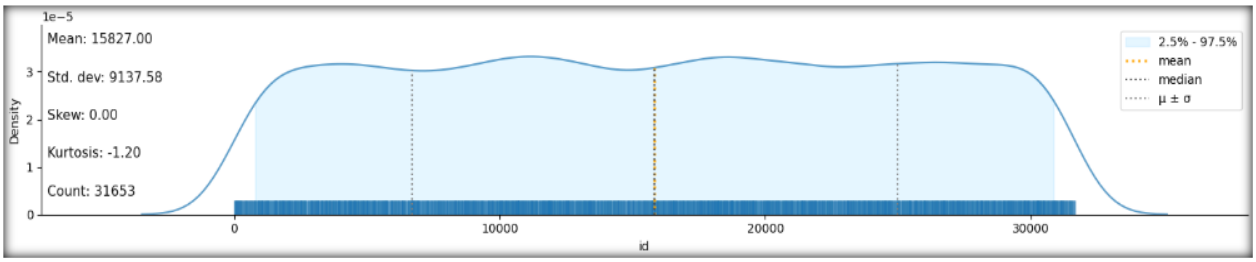

Figure 3: Numerical variables distributed across an axis

Figure 4 explains the hyper parameters used to find the best fit among the parameter spaces for the model to be trained. Bayesian optimization is one such technique that highlights the optimized parameters in the best and easiest ways. It also indicates the total amount of time required for training the model using training data.

iter	target	colsam...	gamma	max_depth	min_ch...	subsample
1	0.7158	0.6293	9.017	6.815	8.96	0.9135
2	0.7168	0.7273	9.475	9.777	1.724	0.9527
3	0.7172	0.9785	6.255	8.997	1.545	0.6176
4	0.7156	0.7794	5.176	3.523	8.818	0.6412
5	0.716	0.876	9.844	8.295	2.638	0.745
6	0.7155	0.7789	8.788	5.659	5.765	0.6004
7	0.7161	0.6412	7.975	5.588	2.092	0.7848
8	0.7182	0.7557	6.66	8.121	9.383	0.8889
9	0.7196	0.915	4.882	8.633	1.914	0.879
10	0.7194	0.879	4.248	8.435	3.525	0.7445
11	0.7188	0.874	5.917	9.242	2.438	0.926
12	0.7161	0.6753	8.648	4.995	2.547	0.8598
13	0.7191	0.6	3.677	10.0	2.361	1.0
14	0.8785	1.0	2.728	7.442	1.543	1.0
15	0.8785	1.0	2.105	6.947	1.219	1.0
16	0.8759	1.0	1.283	7.673	2.211	1.0

It takes 37.373470834891 minutes

Figure 4: Model training executed using bayesian optimization as a part of hyper parameter tuning

5.Results

The final results predicted from the dataset are as follows:

Table 1: Final Results

Material Types	Count
BOOK	18692
VIDEOCASS	1351
VIDEODISC	578
SOUNDDISC	311
MIXED	135
SOUNDCASS	32
MUSIC	3

6.Metrics

Before selecting the best result, we performed a comparative analysis using various algorithms. The evaluation used is the foundation metric computed based on a series of algorithms.

A brief overview of the metrics:

Precision – TP/(TP + FP)
Recall – TP/ (TP + FN)
F1 – Score - 2 * (P * R)/(P+R)

Table 2: Comparative Metrics

AlgorithmUsed	Precision	Recall	F1-Score	Accuracy
XGBoost Classifier	1.00	0.88	0.94	88.48
XGBoost with Hyper parameter	1.00	0.89	0.94	88.58
Random Forest Classifier with Optuna	1.00	0.90	0.95	89.75

7.Data availability statement

Available online at https://assessment.hackerearth.com/challenges/new/hiring/ericsson-ml-challenge-2019/

Kaggle Set available online at https://www.kaggle.com/datasets/saranyashalya/ericsson-ml-challenge-materialtype-prediction

8.Code availability

The codes for analysing the data from mapping the feature variables (Usage Class, Title, Subjects, Publisher etc) to target variable (Material Type) using both Train and Test Data are available on GitHub:

https://github.com/DebmalyaRay9989/codefilesml

9.Discussion

The focus was on the use of machine learning (ML) algorithms in the field of material sciences to predict the correct material type. The analysis was performed using various algorithms from foundation to deep learning models to derive results which satisfactory to be considered a significant contribution to implement ML models in this field.

While comparing the results, we were able to understand the importance of combining the numerical columns with vector data used as a part of text columns to create dense matrix before feeding it into the classification-type foundation models.

Deep Learning emphasizes less on feature engineering skills and more on model definition. In this case, we used a Bi-Directional LSTM-type Neural Network with 32types neurons. We also considered 'Relu' as the activation type function and dropout layer for reducing overfitting scenarios. The model was trained for 41 epochs.

For our foundation model, we considered random forest and xgboost as the main classification algorithms. Each time, the classification algorithm has been optimized using hyper parameter tuners like optuna.

Authors Contributions

The problem descriptions were formulated as a part of ML Hackathon and data source were free licensed in Kaggle. D. Ray performed feature engineering and text analysis and executed comparative results on various classification algorithms. The author has also successfully analysed the results with expected baseline accuracy and contributed to the writing of the paper.

Informed consent statement: Not applicable.

Conflicts of interest: The authors declare no conflicts of interest.

Abbreviations:

The following abbreviations are used in this manuscript:

Table 3: Comparative analysis

ML	Machine Learning
PCA	Principal component analysis
RF	Random Forest
XAI	Explainability AI
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative

References

[1] "Study of the Prediction of Different Particle Material on Conveying Capacity of Differential Vertical Screw Conveyor Using Similarity Theory," 2019 12th International Conference on Intelligent Computation Technology and Automation (ICICTA), Xiangtan, China, 2019, pp. 715-718, doi: 10.1109/ICICTA49267.2019.00156.

[2] PREDICTION OF MATERIAL PRICE USING PREDICTION TOOLS- A COMPARISON International Journal of Scientific & Engineering Research Volume 8, Issue 5, May-2017 160 ISSN 2229-5518

[3] Application of Machine Learning in Material Synthesis and Property Prediction. *Materials* 2023, *16*, 5977. https://doi.org/10.3390/ma16175977

[4] Smart Materials Prediction: Applying Machine Learning to Lithium Solid-State Electrolyte. Materials 2022, 15, 1157. https://doi.org/10.3390/ma15031157

[5] Materials Prediction via Classification Learning online **https://rdcu.be/dvEOP**

[6] "Prediction of compressive strength of self-compacting concrete using least square support vector machine and relevance vector machine." *KSCE J. Civ. Eng.*, 18(6), 1753–1758.

[7] A Data-Driven Pipeline to Learn Structure–Property Insights from Scanning Electron Microscopy Images. ACS Materials Letters **2023**, *5* (11) , 3117-3125. https://doi.org/10.1021/acsmaterialslett.3c00909

[8] Patterns Lead the Way to Far-from-Equilibrium Materials. ACS Physical Chemistry Au **2023**, Article ASAP.

[9] Reformulating Reactivity Design for Data-Efficient Machine Learning. ACS Catalysis **2023**, *13* (20) , 13506-13515. https://doi.org/10.1021/acscatal.3c02513

[10] Cutting conditions and work material state identification through acoustic emission methods. CIRP Annals, 41(1), 89–92, https://doi.org/10.1016/S0007-8506(07)61159-7.

[11] *Data mining: Practical machine learning tools and techniques* (3rd ed.).

[12] Influence of small deviations in steel chemical composition on hard enability. In Proceedings of materials science and technology 2014 (Vol. 1, pp. 493–499).

[13] Audio-based tool condition monitoring in milling of the workpiece material with the hardness variation using support vector machines and convolutional neural networks. *Journal of Manufacturing Science and Engineering, 140* (11), 111006.

[14] Fundamental modelling concepts. *Effective Communication of IT Systems, England, 2005,* 51