2024

EF 37 Artificial Intelligence 23/24 Final Assignment

DECISSION TREE MODELING – DRUG CLASSIFICATION

VEIT BRUNNHUBER

1. Introduction and Outline

The integration of machine learning techniques has transformed pharmaceutical research and drug development. This paper explores the application of machine learning models for drug classification, emphasizing high accuracy and interpretability.

In drug discovery, traditional experimental methods are time-consuming and resource intensive. Machine learning offers an alternative, enabling the analysis of large datasets for predicting pharmacological properties. This shift accelerates drug development, making it more efficient and cost-effective.

Motivated by the need for accurate drug classification and transparent decision-making, this study leverages machine learning to balance predictive performance and interpretability. Understanding the therapeutic properties of compounds is crucial for optimizing treatment strategies, particularly in the pharmaceutical domain were decisions impact human health.

The objectives include exploring various machine learning models (e.g., decision trees, random forests, SVM, XGBoost, LightGBM) for drug classification, hyperparameter tuning, performance evaluation using metrics like macro-averaged F1-Score and assessing interpretability.

The paper's structure comprises three main sections:
Chapter 1 (Introduction and Outline): Introduces the project and its goal. Further it provides the outline of his paper.

Chapter 2 (Methodology): Describes the methodology, covering data preprocessing, model selection, hyperparameter tuning and the implementation of it with a pipeline.

Chapter 3 (Implementation): Provides a detailed account of the implementation process, showcasing, hyperparameters, model characteristics and performance. Further strengths and weaknesses are being mentioned.

Chapter 4 (Discussion and Conclusion): Discusses performance across all models and concludes with insights for future research.

2. Methodology

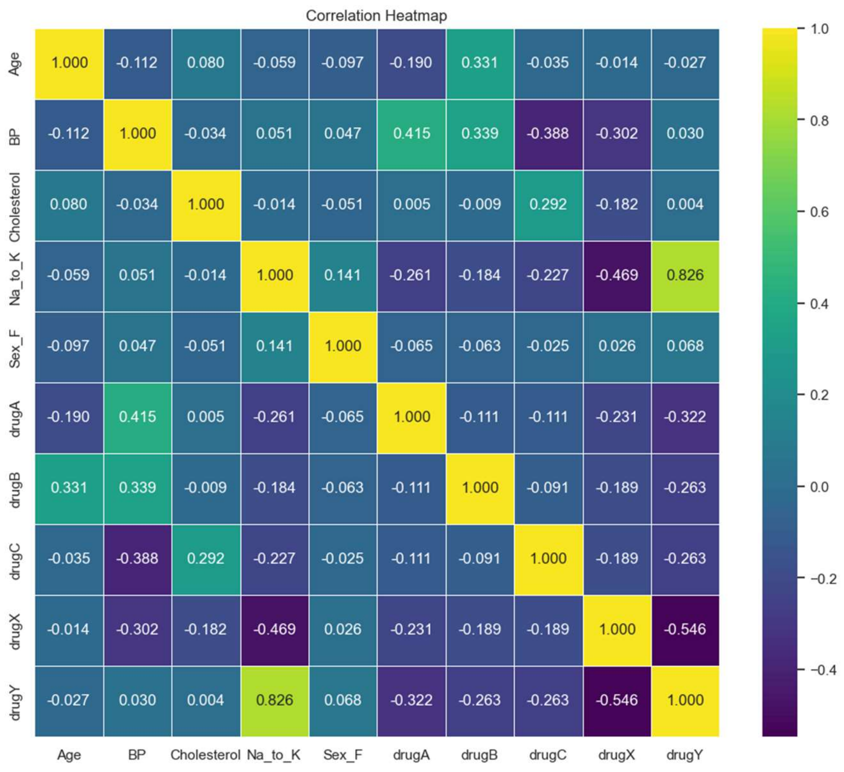
This chapter details the methodology applied to create and assess machine learning models dedicated to drug classification based on specific features. The process encompasses crucial steps such as dataset selection, model selection, hyperparameter tuning, sampling strategies, and the use of evaluation metrics.

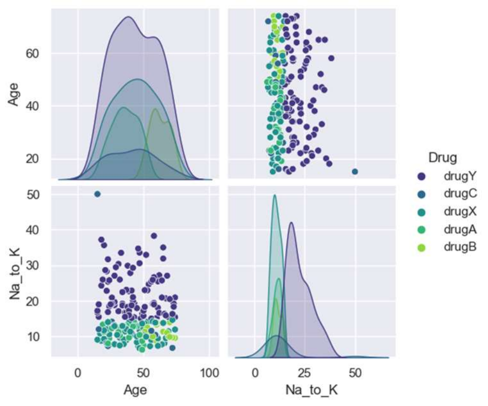
Data Selection: When designing a dataset to classify the correct medication for a patient group, it is crucial that the selected features hold relationships and domain specific related knowledge. Due to the complexity of data collection, I use in the following a dataset that is already prepared for ML use cases provided on Kaggle[1]. The set consists of numerical and categorical features. The

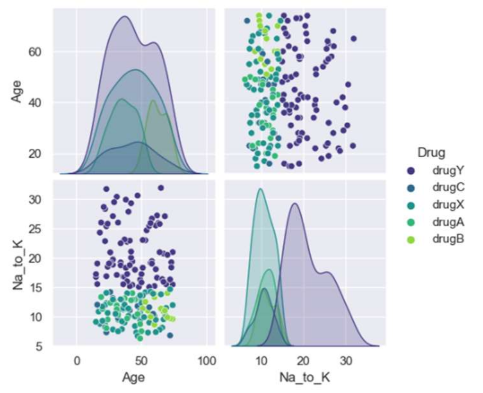
[1] Hirethanad, J. R. (n.d.). drug200.csv. Kaggle. https://www.kaggle.com/datasets/jeevanrh/drug200csv, Accessed on February 17, 2024.

task was to find a dataset that provides a good fit for decision trees. For the exam, I manually added observations to introduce null values and outliers, to showcase the preprocessing step.

Data Preprocessing: Null values were detected and removed. In some cases, filling with the mean or the closest previous value might be suitable. Since the observations are synthetic, they got removed. Further I detected outliers through calculating the Interquartile range (IRQ). The threshold of 1.5 times the IRQ sets the upper and lower decision border. All observations outside of the interval were removed. The last step was to encode all ordinal categorical features with label encoding and the nominal features with one hot encoding. The categorical target variable was label encoded even though it is not ordinal, because the values just must be unique to enable a distinct mapping for the models.

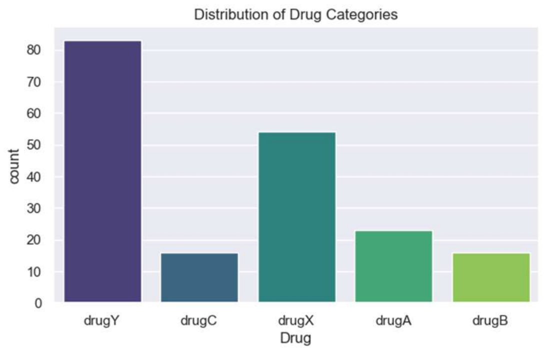






Model Selection: Since the dataset was selected to show of decision tree and other tree-based algorithms. My selection includes Decision Tree, Random Forest, XGBoost, and LightGBM as tree-based models. Further I decided to also use Support Vector Machine (SVM), to have comparison to a strong model that does not rely on trees. SVM constructs an optimal hyperplane to separate different target classes in a high dimensional space, where each dimension is constructed out of one feature of the dataset. All selected models are able to handle non-linear relationships.

Sampling: To address class imbalance in target Medications, random Under-Sampling and Synthetic Minority Over-sampling Technique (SMOTE[2]) were used in the Pipeline. I decided to under sample all target variables to 31. To establish the balance, I oversampled all target variables to 33. These numbers were first guessed as a multiple of 3 of the minority class, to find a balance of over and under sampling. The idea is not to introduce noise through losing relevant data points or due to overfit with the oversampled minority class.



Hyperparameter tuning: To ensure that each model uses the proper parameters, I first guessed most obvious parameter combinations. After that I extended the parameter list with positive and negative steps with an appropriate step size. The model gets trained with all possible parameter combinations to find the best parameter combination. These are selected using Grid Search with a Stratified K Fold cross validation with 5 splits and f1 scoring.

Training the models: To address feature imbalance in the dataset, as well as handling hyperparameter tuning, I constructed a IMBLearn Pipeline. This prevents data leakage and streamlines the training process in a concise and elegant way. The Pipeline I used pipes the data through Under-Sampling, Scaling, Over-Sampling, and the actual model itself. Each Pipe for the according model, is being Grid Searched with a parameter grid that supplies the according parameters to the associated step in the pipelines. This streamlines the process.

Evaluation metrics: The classification report serves as a central metric for evaluating machine learning models in this Examination task. It offers a comprehensive overview that highlights precision, recall, and the F1-score. The F1-score is particularly crucial, providing insights into the model's balance between precision and recall, especially in scenarios like drug classification, where false positives and false negatives carry significant implications. Recall and precision metrics play distinctive roles in model evaluation. Recall measures the model's ability to capture relevant instances within each target class, emphasizing sensitivity. Precision, on the other hand, assesses the model's accuracy in classifying positive instances, with a focus on avoiding misclassification of negatives. To inspect the model’s generalisation capability the evaluation includes f1 macro score for train and test data.

3. Implementation and Outcomes

This chapter delves into the practical application and outcomes of the outlined methodology, emphasizing the performance of machine learning models for drug classification with the selected dataset. The experimental setup involves a dataset division into training and testing sets

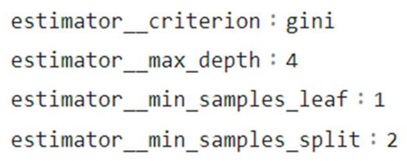
[2] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique..

with a standard approach of 25% test size. Further the split is sensitive to class balance. The stratify parameter ensures that the target variables are balanced across the splits.

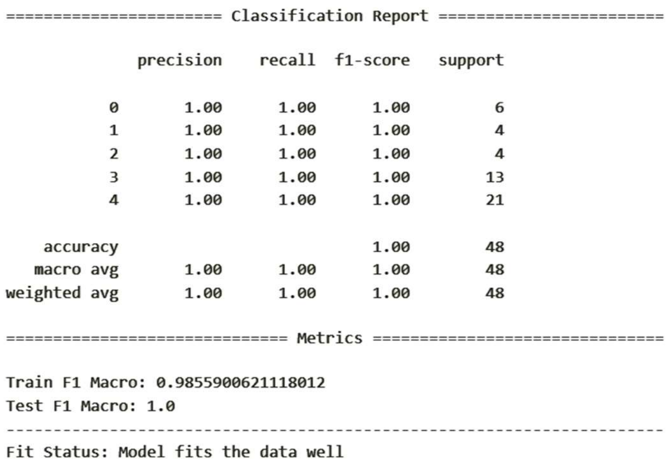
In the following all models were trained and tuned with the IMBPipeline approach outlined in the previous chapter. The whole pipeline is being grid searched. So, the optimal parameters are chosen based on the F1 Macro score. The F1 Macro score balances precision and recall across multiple classes, offering an overall performance metric for classification models.

Each model holds a figure of their best parameters and their classification report associated with the best parameters.

Decision Tree Model

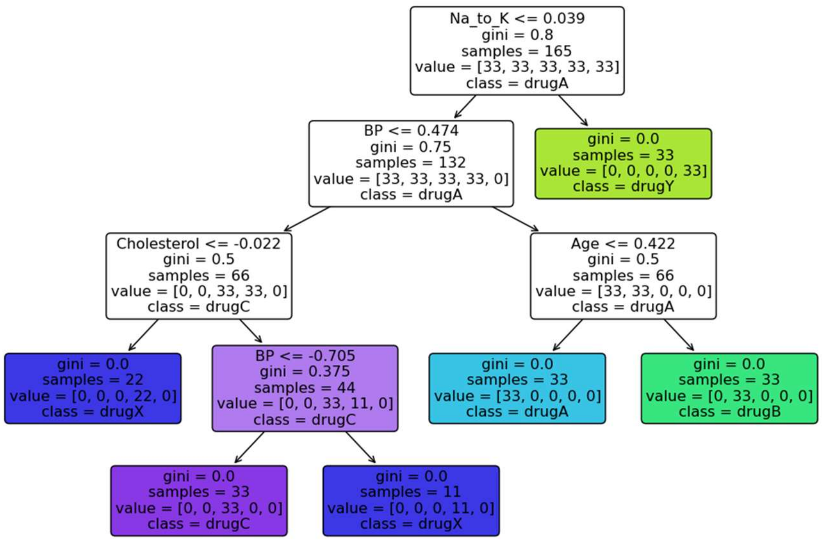


```
estimator__criterion : gini
estimator__max_depth : 4
estimator__min_samples_leaf : 1
estimator__min_samples_split : 2
```



```
====================== Classification Report =======================

              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       1.00      1.00      1.00         4
           2       1.00      1.00      1.00         4
           3       1.00      1.00      1.00        13
           4       1.00      1.00      1.00        21

    accuracy                           1.00        48
   macro avg       1.00      1.00      1.00        48
weighted avg       1.00      1.00      1.00        48

============================= Metrics ==============================

Train F1 Macro: 0.9855900621118012
Test F1 Macro: 1.0
--------------------------------------------------------------------
Fit Status: Model fits the data well
```

The Grid Search found optimal parameters, these are well balanced across depth, leaf samples and split criteria. The decision tree model[3] performed expectedly good, since the data was for it selected. It archives 100% accuracy on the test set. To inspect the generalisation capabilities, I also checked the accuracy for the train set. The result shows that the model rather leans towards underfitting than overfitting. Therefore, it provides a strong generalisation power. This might be due to the set being so small. So, this model is very capable of representing the underlying data tendencies.



One of the biggest differences between decision tree and other models is, that it provides a comprehensive insight into the workings of the underlying model.

As explanation I will take the first two levels of the tree diagram to highlight the meaning of these parameters.

At the root level, the decision is based on the feature "Na_to_K"

[3] Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and Regression Trees. CRC Press.

being less than or equal to 0.039. This splits the data into two branches. “Gini” refers to Gini impurity[4].

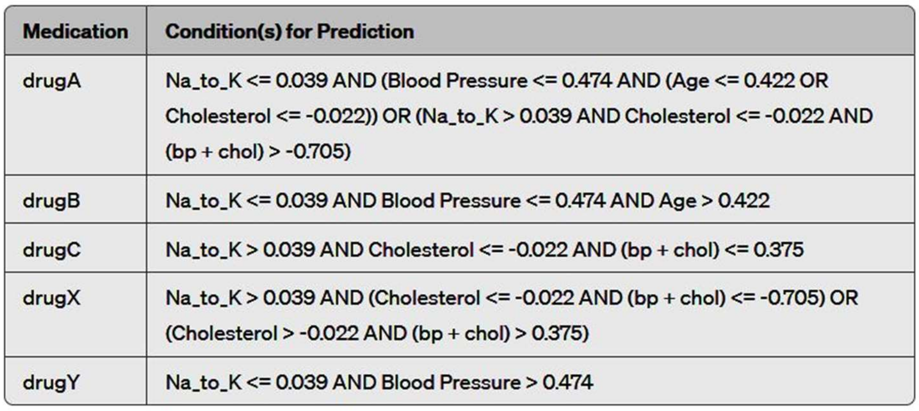
This is a measurement of classification uncertainty. In this example out of 165 samples a Gini of 80% implies that each class has 33 samples.

$$\text{Gini impurity} = 1 - \sum_{i=1}^{c} (p_i)^2$$

- C is the number of categories
- Pi represents the probability of an instance of class i.

The second level shows that after the split we already have a separated class. So, one can read out of this that to prescribe “drugY” the patient needs a “Na_to_K” value of less than or equal to 0.039. The other child node shows that there is now only an impurity of 75% left. Further it shows that the next splitting criteria is “BP” less than or equal to 0.474.

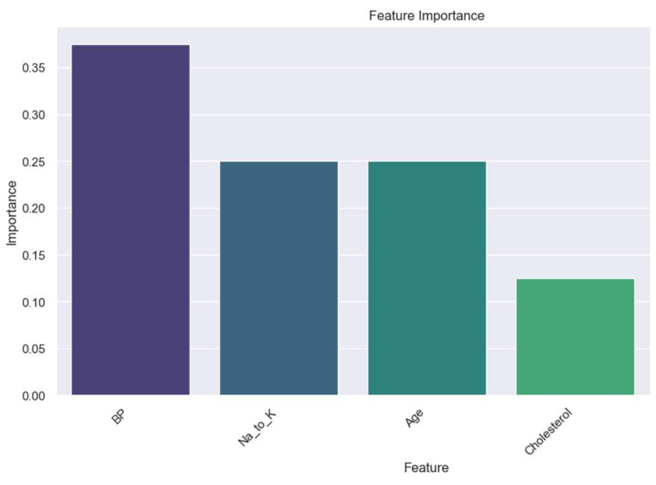
When going through the whole tree one could construct a table that holds all medications and their prescription criteria.



Medication	Condition(s) for Prediction
drugA	Na_to_K <= 0.039 AND (Blood Pressure <= 0.474 AND (Age <= 0.422 OR Cholesterol <= -0.022)) OR (Na_to_K > 0.039 AND Cholesterol <= -0.022 AND (bp + chol) > -0.705)
drugB	Na_to_K <= 0.039 AND Blood Pressure <= 0.474 AND Age > 0.422
drugC	Na_to_K > 0.039 AND Cholesterol <= -0.022 AND (bp + chol) <= 0.375
drugX	Na_to_K > 0.039 AND (Cholesterol <= -0.022 AND (bp + chol) <= -0.705) OR (Cholesterol > -0.022 AND (bp + chol) > 0.375)
drugY	Na_to_K <= 0.039 AND Blood Pressure > 0.474

Another handy insight a decision tree brings is a deeper insight into feature relevance. This holds valuable information such as what features contribute the most to the decision process.

The feature “Sex” was removed from the plot because the relevance was zero.



Strengths:

[4] The Gini impurity formula, a fundamental concept in decision tree algorithms and machine learning, is widely employed in the field. There is not a specific original source because, its applications are extensively discussed in literature related to machine learning and decision trees.

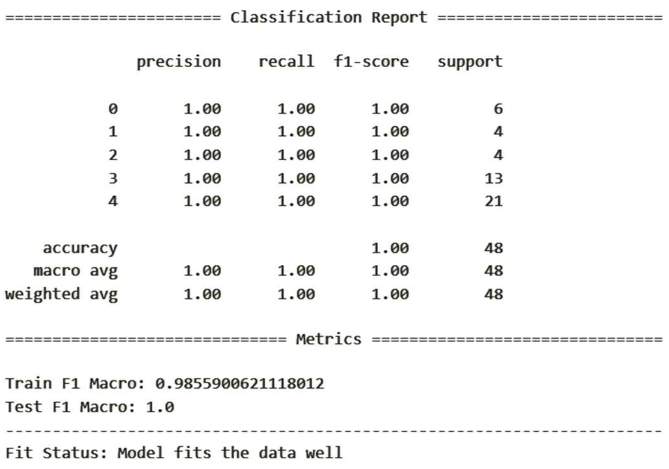
No assumptions about data	Interpretability	Handling non-linearity

Limitations:

Overfitting	Sensitive to Small Variations	Bias Towards Dominant Classes	Limited Expressiveness

Random Forest Model

```
estimator__criterion : gini
estimator__max_depth : 4
estimator__min_samples_leaf : 1
estimator__min_samples_split : 2
```



```
====================== Classification Report =======================

              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       1.00      1.00      1.00         4
           2       1.00      1.00      1.00         4
           3       1.00      1.00      1.00        13
           4       1.00      1.00      1.00        21

    accuracy                           1.00        48
   macro avg       1.00      1.00      1.00        48
weighted avg       1.00      1.00      1.00        48

============================ Metrics ==============================

Train F1 Macro: 0.9855900621118012
Test F1 Macro: 1.0
---------------------------------------------------------------------
Fit Status: Model fits the data well
```

Since the Random Forest[5] is just a bagging method of decision trees, it has the same parameters and performance as the optimal underlying decision tree. Normally Random Forest is more reliable than decision tree, because it bases it prediction through aggregating the predictions of multiple decision trees into on final prediction. Therefore, it is not so prone to overfitting and genuinely more reliable.

Strengths:

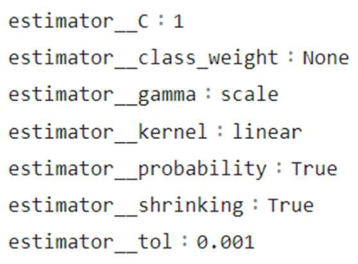
Ensemble Learning	High Accuracy	Variable Importance

Limitations:

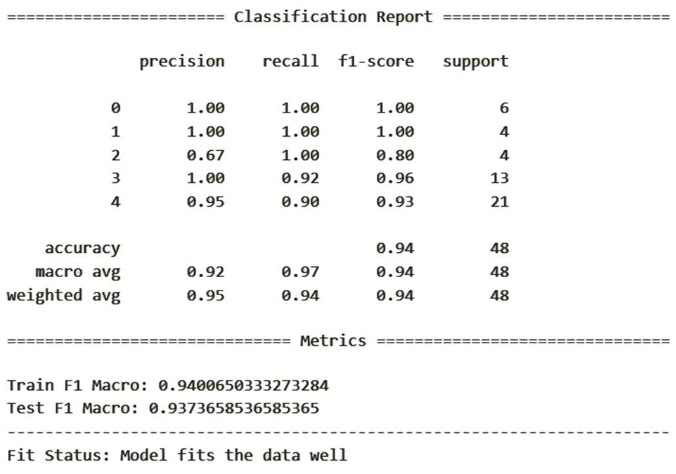
Complexity (Expensive)	Less Interpretable	Bias Towards Dominant Classes

[5] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

Support Vector Machine (SVM) Model



```
estimator__C : 1
estimator__class_weight : None
estimator__gamma : scale
estimator__kernel : linear
estimator__probability : True
estimator__shrinking : True
estimator__tol : 0.001
```



```
====================== Classification Report =======================

              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       1.00      1.00      1.00         4
           2       0.67      1.00      0.80         4
           3       1.00      0.92      0.96        13
           4       0.95      0.90      0.93        21

    accuracy                           0.94        48
   macro avg       0.92      0.97      0.94        48
weighted avg       0.95      0.94      0.94        48

============================ Metrics =============================

Train F1 Macro: 0.9400650333273284
Test F1 Macro: 0.9373658536585365
------------------------------------------------------------------
Fit Status: Model fits the data well
```

The SVM[6] was grid searched to employ optimal parameters. It showcases its prowess in constructing an optimal hyperplane within a high-dimensional space, defined by individual dataset features. Since there are just 5 features, there are not enough distinct dimensions for SVM to be able to perform as well as it does in other situations. This results in a 94% accuracy even though it is a model that captures non-linear relationships very well. Non the less it showed within those 94% percent strong generalisation capability with a 0.2% tendency towards overfitting. This is very far away from the 10 and even 5 percent threshold.

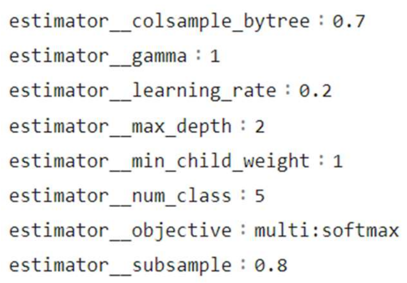
Strengths:

Effective in High-Dimensional Spaces	Optimal Hyperplane	Versatility in Kernels

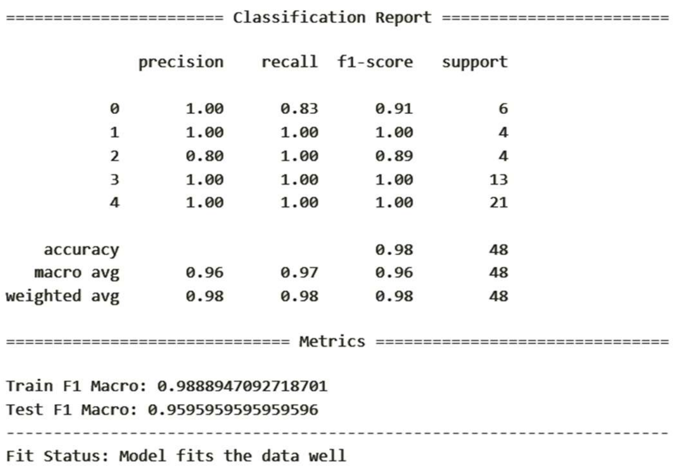
Limitations:

Sensitivity to Noise	Computational Intensity	Interpretability

XGBoost Model



```
estimator__colsample_bytree : 0.7
estimator__gamma : 1
estimator__learning_rate : 0.2
estimator__max_depth : 2
estimator__min_child_weight : 1
estimator__num_class : 5
estimator__objective : multi:softmax
estimator__subsample : 0.8
```



```
====================== Classification Report =======================

              precision    recall  f1-score   support

           0       1.00      0.83      0.91         6
           1       1.00      1.00      1.00         4
           2       0.80      1.00      0.89         4
           3       1.00      1.00      1.00        13
           4       1.00      1.00      1.00        21

    accuracy                           0.98        48
   macro avg       0.96      0.97      0.96        48
weighted avg       0.98      0.98      0.98        48

============================ Metrics =============================

Train F1 Macro: 0.9888947092718701
Test F1 Macro: 0.9595959595959596
------------------------------------------------------------------
Fit Status: Model fits the data well
```

XGBoost[7] is a very strong model that dominates a lot of competitions in Kaggle currently. Due to its many capabilities like handling missing data

[6] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.
[7] Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16), 785–794. https://doi.org/10.1145/2939672.2939785

out of the box, or handling categorical data without encoding, it is essential to hyperparameter tune it carefully. Therefore, my hyperparameter grid was very big for the first tuning time. To speedup the execution time of the jupyter notebook I reduced the grid down to a reasonable amount through identifying the optimal parameters. After that I only left the optimal parameters and some surrounding steps. XGB excels in high-dimensional spaces, making the most of the limited dimensions within the dataset. With just 5 features, XGB showcases remarkable accuracy at 95%. Its strength lies in iteratively refining weak learners, ensuring robust generalization. With ~3% deviation between test and train set accuracy, it demonstrates strong generalization capabilities, but ends up still on the last place in a comparison to the other models discussed here.
These results might be caused because the model is too complex and sophisticated for such a small and rather simple set compared to normal inputs.

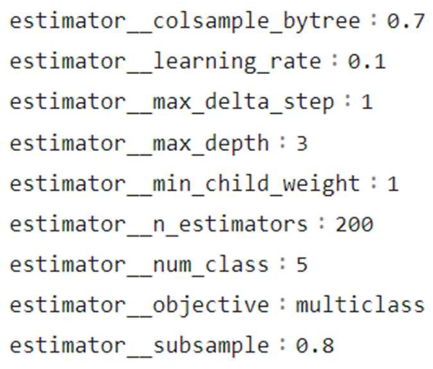
Strengths:

Gradient Boosting | Handling Missing Data | Regularization

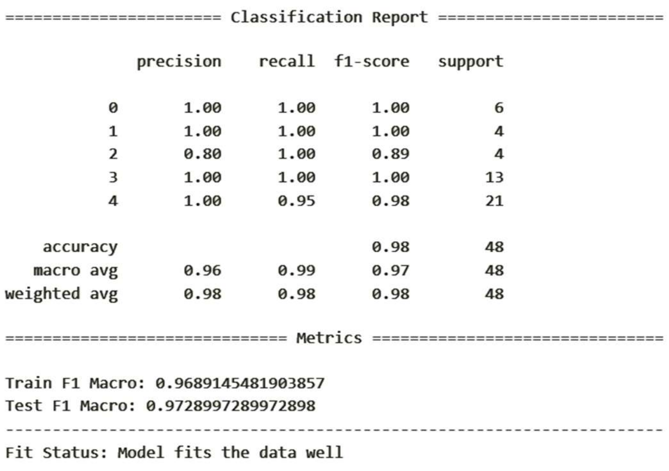
Limitations:

Computational Intensity | Sensitive to Hyperparameter Tuning

LightGBM Model



```
estimator__colsample_bytree : 0.7
estimator__learning_rate : 0.1
estimator__max_delta_step : 1
estimator__max_depth : 3
estimator__min_child_weight : 1
estimator__n_estimators : 200
estimator__num_class : 5
estimator__objective : multiclass
estimator__subsample : 0.8
```



```
====================== Classification Report ========================

              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       1.00      1.00      1.00         4
           2       0.80      1.00      0.89         4
           3       1.00      1.00      1.00        13
           4       1.00      0.95      0.98        21

    accuracy                           0.98        48
   macro avg       0.96      0.99      0.97        48
weighted avg       0.98      0.98      0.98        48

============================= Metrics ==============================

Train F1 Macro: 0.9689145481903857
Test F1 Macro: 0.9728997289972898
--------------------------------------------------------------------
Fit Status: Model fits the data well
```

LightGBM[8] is also a cutting-edge gradient boosting framework. As well as XGB it is recently prominent in Kaggle competitions. Further it is also capable of handling missing data seamlessly and processing categorical data without encoding. Since it is also a gradient boosting algorithm, careful hyperparameter tuning is crucial. To address this, I used the same approach for tuning as for XGB. LightGBM excels in high-dimensional spaces like SVC or XGB. Despite its computational efficiency, the model delivers an impressive 95% accuracy. However, with a ~2% deviation between test and train set accuracy, it

[8] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Advances in Neural Information Processing Systems 30 (NIPS 2017), 3146–3154.

exhibits strong generalization capabilities, securing a notable position in the comparison with other models. The nuanced performance suggests the model's adaptability to the dataset's simplicity and highlights its potential for more complex scenarios.

Strengths:

Efficient Gradient Boosting	Handling Missing Data	Categorical Feature Support

Limitations:

Less Interpretability	Sensitivity to Outliers

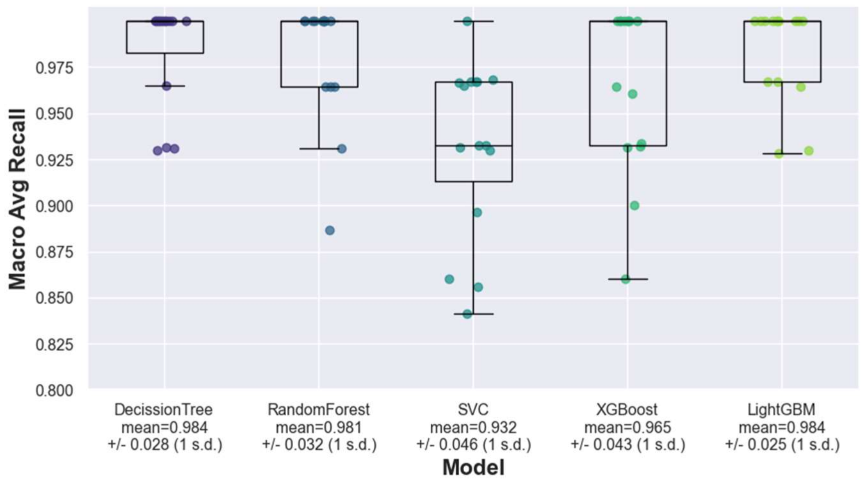
4. Discussion and Conclusion

In this comprehensive exploration of machine learning models for drug classification, the methodology encompassed careful dataset selection, preprocessing steps, model selection, hyperparameter tuning, and evaluation metrics. The chosen dataset, sourced from Kaggle, underwent meticulous preprocessing, including null value handling, outlier detection, and feature encoding.

Decision Tree, Random Forest, Support Vector Machine (SVM), XGBoost, and LightGBM were selected for their unique strengths in handling drug classification tasks. The models were subjected to rigorous hyperparameter tuning using grid search and evaluated based on the F1 Macro score, providing a balanced metric for precision and recall across multiple classes.

The decision tree model showcased exceptional accuracy and interpretability, revealing insights into its decision-making process. Random Forest, an ensemble of decision trees, offered reliability and superior performance through aggregation. SVM demonstrated its proficiency in constructing optimal hyperplanes despite the small dataset dimensionality.

XGBoost, known for its dominance in Kaggle competitions, exhibited strong generalization capabilities, emphasizing the need for careful hyperparameter tuning. LightGBM, a cutting-edge gradient boosting framework, emerged as a promising contender, efficiently handling high-dimensional spaces.



To enable a comprehensive visual comparison among the models and facilitate a nuanced assessment, I created a boxplot, capturing the macro-average recall scores for each model. This graphical representation offers a clear depiction of the variability and central tendencies within the recall scores across all models. Notably, all models exhibited remarkably high macro-average recall scores, underscoring their effectiveness in capturing relevant instances across multiple classes. The boxplot not only allows for a quick overview of model performances but also highlights any variations or outliers that may impact their comparative evaluation.

Taking these boxplots as well as evaluation reports into account, it appears that the decision tree is the best performing model on this dataset. It leads in terms of computational costs as well as the overall accuracy and F1 score. This aligns also quite well with the spirit of Occans´s Razor[9]. His principle suggests that one even when exploring complex architectures, should give weight to the efficacy of simpler configurations. He claimed that the most effective solution might indeed be the one that makes the fewest assumptions. This could also be an interpretation on why the complex tree-based models like XGBoost or LightGBM are outperformed by the simple decision tree.

In conclusion, the evaluation and comparison of these models provide valuable insights into their applicability for drug classification tasks. The study emphasizes the importance of considering model intricacies, computational efficiency, and dataset characteristics when selecting an appropriate machine learning model. Future work may involve exploring more complex datasets and fine-tuning models for enhanced performance in specific drug classification scenarios.

[9] "Occam's Razor" refers to the philosophical principle advocating for simplicity in scientific explanations, often associated with William of Ockham. The ideas behind Occam's Razor have been widely discussed in various philosophical and scientific writings.
(n.d.). Occam's Razor. In Encyclopaedia Britannica. https://www.britannica.com/topic/Occams-razor