Hyperparameter Optimization of XGBoost-model on Cancer-DNA Dataset

Team 43
CENTURY TAN
LYNGBAEK LAURITS WIESLANDER

Abstract

Identifying cancer within patients is crucial for several reasons. First, it shows the genetic traits and alterations responsible for tumour development, progression, and heterogeneity, providing insights into the underlying mechanisms of cancer. Second, it encourages the discovery of biomarkers that can aid in early detection, prognosis, and treatment response prediction. Third, understanding cancer genomes enables the development of targeted therapies tailored to individual differences, leading to more personalised, and thus effective, treatment strategies. Overall, detecting cancer accurately is crucial for timely intervention, effective treatment, and improved patient outcomes.

The machine learning techniques utilised in this study include a plethora of models designed for classification tasks. These include methods like Gradient Boosting, specifically XGBoost as well as Principal Component Analysis (PCA). The integration of these ML techniques with feature extraction from DNA fragment length frequencies enhances the accuracy and reliability of cancer stage predictions, enabling clinicians and researchers to make informed decisions regarding patient care, treatment strategies, and disease prognosis.

Introduction

This report will describe our machine learning project using the XGBoost (Extreme Gradient Boosting) algorithm to predict cancer progression from a patient's DNA fragments.

Feature Selection and Visualization

The selected dataset is a large class imbalanced tabular dataset with large amounts of correlations. A robust classification algorithm is therefore necessary to avoid overfitting. The algorithm chosen would need to be able to handle correlation in features and have a weighted loss-function. To choose the most relevant features from the dataset, we utilised an extended algorithm for boosting decision-trees known as XGBoost.

The figure below represents a correlation heatmap of 50 randomly sampled DNA fragments. The plot visualises how almost all features are slightly to strongly correlated with each other.

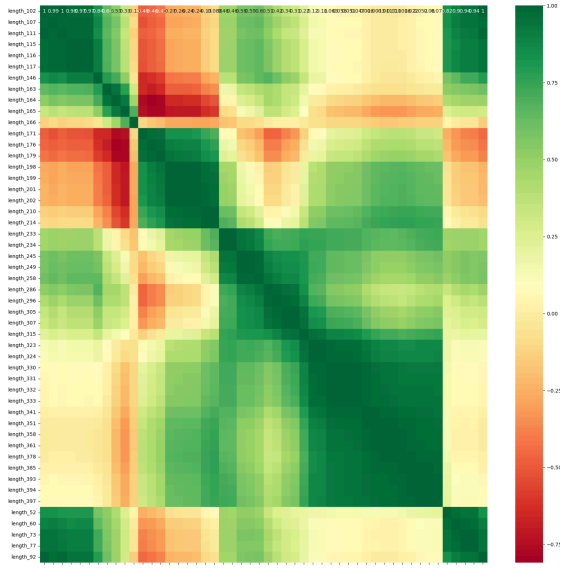

Figure 1: Heatmap of 50 DNA Fragments

While testing the feasibility of PCA, we used a table and a line graph to plot the cumulative explained variance of the PCA components.

This enabled us to visualise the amount of variance in the original dataset that was captured by each additional principal component. By analysing the variance plot, we can assess how many principal components are required to retain a desired proportion of the total variance while preserving important information within the original dataset. As seen in the plot below by having only 3 principal components 95% of the variance is explained.

We utilised PCA to determine if it could improve the prediction accuracy of our model. The objective of PCA is to reduce the number of dimensions of the dataset while retaining the most important features, thereby improving the efficiency of hyperparameter testing. It reduces the number of features, allowing for better visualisation of data. However, PCA did not benefit us much, thus we fitted our prediction model on all 350 features to preserve model interpretation within the dataset. Fitting the model to the PCA-components was therefore left for future investigation.

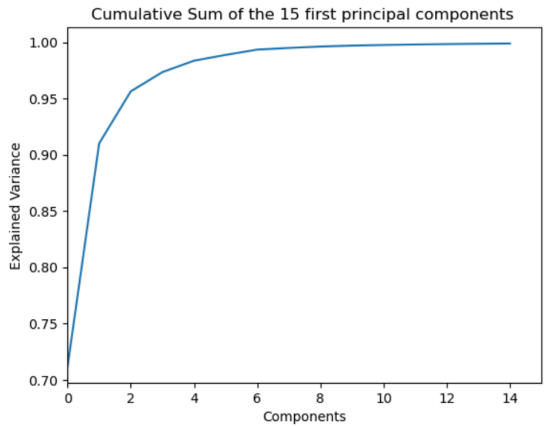

Figure 2: Cumulative Explained Variance

Missing Value Imputation

The dataset that we used was complete and had no missing values. However, if there were missing values, it would be approximated by taking the average value of the class.

Related Works

Our project drew inspiration from "Classification of tumor types using XGBoost machine learning model: a vector space transformation of genomic alterations" (Zelli et al., 2023) and "Machine Learning for eHealth Systems" (Schneider and Xhafa, 2022). They provided a foundation for understanding the application of ML algorithms and techniques to classify diseases based on genomic data and the broader implications of ML in healthcare. The primary limitation that was mentioned in these two sources is data imbalance, leading to worsened performance of the models.

To address data imbalance, hyperparameter testing was conducted. The use of parameters such as scale_pos_weight within the XGBoost library ensured that we had a balance between our healthy and unhealthy samples. The parameter scale_pos_weight assigns a greater weight to the underrepresented class, in this case, the healthy samples, helping to counteract the imbalance by increasing the model's sensitivity to these samples. This improves the model performance by accounting for the data imbalance. Our model was fitted with scale_pos_weight 0.0028129 as estimated by the class imbalance based on the number of health and cancer samples we had (60 healthy and 2133 cancer samples).

We then implemented cross-validation alongside hyperparameter testing to get a reliable estimate of this model's performance across the different train-test splits and the various hyperparameters. This allowed us to find the most optimal hyperparameters for our model, ensured model generalisation and reduced the risks of overfitting.

Machine Learning Algorithm Details

Classification decision trees:

Decision trees work by stratifying predictor space into subregions with more homogeneity in predicted classes. Each of these subregions can be continuously split until the purity of each region, also referred to as each node, is too small to split again. The stopping point of these decision trees can be regulated by a set of parameters. When constructing new trees, a greedy-first tree-building algorithm known as recursive binary splitting is used. When used for classification the algorithm generally constructs the tree by selecting the split that minimises tree entropy or the Gini index. But decision trees are very sensitive to outliers and new data, even if the construction is restricted by multiple rules, so they have a tendency to overfit. Therefore they are overpruned, also known as strongly restricted, so that they can only do weak underfitting predictions. By combining multiple of these weak learners in an ensemble their mean prediction will be very accurate but still retain a relatively low bias.

Ensemble of trees by boosting:

Boosting is a framework that combines a large number of learners fitted iteratively, in this case a large number of classification decision trees. The boosting algorithm can be generalised to **1)** setting residuals $r_i = y_i$ and $\hat{f}(x) = 0$, then for all trees 1,2 … b, **2)** fit tree $\hat{f}^b(x)$ with d terminal nodes to data (X, r) and then **3)** update the model with the new tree $\hat{f}(x) \leftarrow \hat{f}(x) + \hat{f}^b(x)$, and finally update the residuals with the new predictions $r_i \leftarrow r_i - \lambda \hat{f}_b(x_i)$. After all the trees have been fitted, the model is ready. The model uses the log odds of each class and the logistic function to fit the model numerically, so it can update the residuals. This also means that the model outputs a probability distribution for each class occurring, instead of a singular prediction. Due to the trees being fitted to the updated residuals instead of the original data it is important to be careful when interpreting a singular decision tree.

Extreme Gradient Boosting

XGBoost (Extreme Gradient Boosting) is a supervised ensemble gradient boosting algorithm. Generally, gradient boosting is used to build a strong predictive model by combining multiple weak learners sequentially, as it is based on the ideas of boosting. However gradient boosting refers to a set of algorithms that fits the model to the negative gradient of a chosen loss-function. Using the gradient results in a more stable and computational effective algorithm, especially when regulated by a learning rate. The individual learners are supposed to not overfit and are therefore limited in complexity. In XGBoost, the decision trees are also built by minimising the negative gradient. XGBoost differs by restricting the construction of the decision trees in multiple ways, such as stochastically choosing features for available at each

iteration and minimum loss reduction gain needed to split a node.

We decided on XGBoost as it handles high-dimensional data effectively and is robust to overfitting when tuned properly. As the data is imbalanced, 60 healthy samples and 2133 cancer samples, XGBoost's ability to handle class imbalances within the dataset through weighted sampling makes it a particularly suitable algorithm to handle this dataset. Additionally, XGBoost provides multiple hyperparameters, such as learning rate and maximum depth of trees, allowing for optimization that is specific to the features within the dataset. Overall, it allowed us to build an effective prediction model that could handle our high-dimensional data and class imbalance. The exact algorithm used was the function XGBoostClassifier from the XGB library created by Chen & Guestrin, 2016.

Optimization of hyper parameters:
XGBoost is a highly flexible algorithm with many tunable hyperparameters. It has multiple parameters that constrict overfitting and interact at different levels. Some affect the decision tree construction such as max terminal leaf nodes, minimum loss reduction when splitting data, subsample size and how many features are used in construction. Other parameters interact at a higher level such as number of trees constructed, learning rate when updating residuals and L1 and L2 regularisation. However, they all try to combat overfitting of the model, which is especially important when working with high-dimensional data. When the amount of parameters increases, it becomes more computationally expensive to estimate an optimal combination through a grid-search.

It is therefore deemed reasonable to use a randomised cross-validation search instead. This was done with the RandomizedSearchCV function from the sklearn package. Parameters from a given parameter space are sampled, to construct a smaller set of plausible parameter-combinations. Each of these combinations are then cross validated, where the respective effectiveness of each parameter-combination is ranked by its performance on a scoring metric. As the amount of fitted combinations increases this should approximate an optimal solution.

Classification score:
To determine if the XGBoost is performing well, it is important to first find a classification score that reflects a good performance. This is a rather difficult task as the dataset is very imbalanced. This means that we cannot use the normal prediction accuracy score, $accuracy = \frac{TN+TP}{TN+FN+TP+FP}$, as a model that always predicts cancer would have a impressive accuracy of $\frac{2133}{2133+60} = 97.2\%$ on the training data.

Moreover, as this model is used to determine the presence of cancer, we might also want to make sure certain wrong predictions are punished more severely in our prediction score. If the DNA samples are taken from assumed healthy individuals, and we run the model on their DNA as a routine check, then it would be wise to favour avoidance of false positives. For this project, we will assume that the DNA is a sample from an assumed cancerous cell, and we therefore wish to reward both true positives and true negatives.

It is not recommended to use ROC-AUC for imbalanced datasets (Saito and Rehmsmeier, 2015). Another common classification metric for imbalanced datasets is the F1-metric, however F1 fails to consider the true negatives, and is therefore only good for imbalanced datasets where more attention is needed on the positives. The 'Balanced Accuracy' was therefore for the classification score. It is defined as:

$$Balanced\ Accuracy = \frac{(specificity + recall)}{classes}$$

$$Sensitivity = \frac{TP}{(TP+FN)}$$

$$Recall = \frac{TN}{(TN+FP)}.$$

This is also easily generalisable to multiple classes.

Experiments and Results
We first want to attempt to correctly classify healthy vs unhealthy cells. By doing a grid search of the parameter space an optimal. A total of 40 XGBoost models were fitted and the optimal model reached a 'balanced accuracy' of 72.1% on the test-data not used in cross validation. It also performed well on other metrics; notably a Precision-Recall score of 0.977 and an F1 score of 0.88.

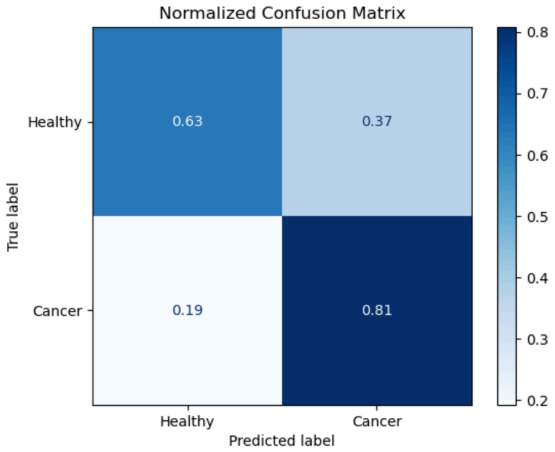

Figure 3: Normalised Confusion Matrix

The model was expanded to multiclass identification, as the model had a reasonable performance in binary classification. Even though the probability of a correct prediction decreases as the amount of predictable classes increases, it can be seen from the confusion multiclass matrix that the model scales relatively well to higher dimensions.

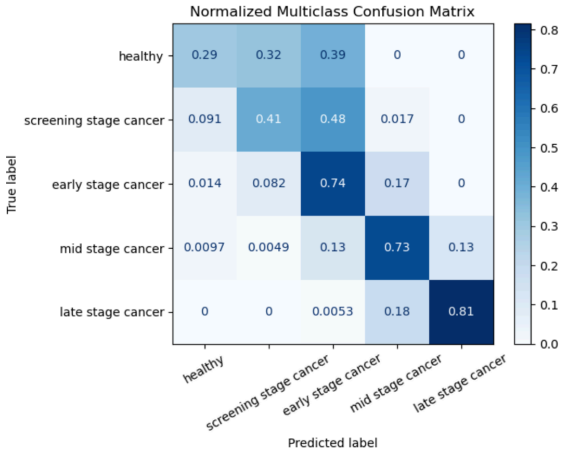

Figure 4: Normalised Multiclass Confusion Matrix

The model has a tendency to overpredict cancer rather than predict the cell to be healthy, achieving a balanced accuracy score of 59.7% and a f1-score of 0.588. However the model rarely predicts false health, and is therefore safe to use as a diagnostic tool on assumed cancer patients. This could allow healthcare professionals to easily diagnose the 30% subjects declared healthy. Future studies could investigate if including the output of a binary healthy-cancer model as an input for the multiclass model could increase its effectiveness in the healthy-early stages. This model should not be used on suspected healthy subjects, as it predicts too many false cancers.

The built-in plot_importance function within the XGBoost library allowed us to plot the features that were the most important for our model's prediction. However the feature importance should be interpreted with care as highly correlated features share their 'importance', and should ideally be represented by confounding variables.

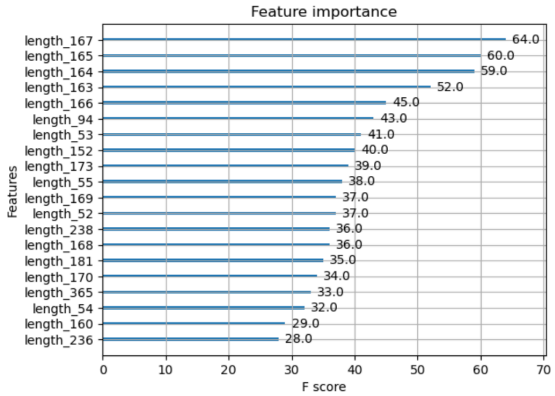

Figure 5: F-score of most important features

Conclusion

To conclude, this project has managed to demonstrate an implementable hyperparameter optimization method for enhancing the predictive accuracy of the XGBoost algorithm for cancer-DNA analysis. By iteratively tuning key parameters, we achieved significant improvements in model performance, highlighting the potential of machine learning in cancer diagnostics. However, this project faced constraints such as dataset diversity and imbalance, which may affect the generalizability of the findings. Future research should focus on applying these optimization strategies to larger and more balanced datasets to fully determine their effectiveness and utility in real-world applications. Ultimately, the optimised XGBoost model represents a step forward in the pursuit of accurate cancer diagnostics.

References

Schneider, P., & Xhafa, F. (2022, January 14). *Machine learning: ML for eHealth Systems. Chapter 8 - Machine learning: ML for eHealth systems.* https://www.sciencedirect.com/science/article/abs/pii/B9780128238189000195

Zelli, V., Manno, A., Compagnoni, C., Ibraheem, R. O., Zazzeroni, F., Alesse, E., Rossi, F., Arbib, C., & Tessitore, A. (2023, November 21). *Classification of tumor types using XGBoost machine learning model: a vector space transformation of genomic alterations.* https://link.springer.com/article/10.1186/s12967-023-04720-4

Saito, T., & Rehmsmeier, M. (2015). *The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PloS one, 10(3), e0118432.* https://doi.org/10.1371/journal.pone.0118432

Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785-794).