DS 5220: Supervised Machine Learning and Learning Theory - Fall 2021
Blood Spectroscopy Classification Challenge

Team no.: 16

Apoorva Surendra Malemath | Vaidehi Pareshkumar Parikh

1. Abstract

The challenge is hosted by bloods.ai. [1] They are a team of ambitious, innovative professionals who are passionate about offering a better way to get in touch with and care for our bodies - all with the simple scan of skin. This challenge is hosted by Zindi, [2] and focuses on helping scientists from bloods.ai to make progress towards non-invasive blood analyses. The idea behind this is to classify the level of specific chemical compounds in samples from their spectroscopic data. When a beam of light is directed towards a sample, the light is partially absorbed and/or reflected based on the sample's molecular structure (different chemical compounds present). The amount of light absorbed strongly depends on the wavelength of the light source used. Hence, if the beam of light used contains a range of wavelengths, we can measure the amount of energy absorbed for each wavelength. Such a measurement over different wavelengths (or frequencies) is called a spectrum (or spectral data).

2. Statement of Contribution

Apoorva	● Outlier Detection and Analysis ● Sampling - SMOTE ● Feature Selection - RFE, RFECV, PCA ● Models with Hyper Parameter Tuning ○ Logistic Regression with L2 Regularization ○ Random Forest Classifier ○ Extra-Tree classifier ○ CatBoost Classifier
Vaidehi	● Exploratory Data Analysis ○ Correlation Heat Map ○ Distribution of Data ● Checking NA values ● Feature Selection - RFE, RFECV, PCA ● Models with Hyper Parameter Tuning ○ Random Forest Multi-Output Classifier ○ Radius Neighbor Classifier ○ Neural Network

3. Introduction

The data comprises spectral data from the Near Infra-Red (NIR) wavelengths ranges (950 nm to 1350 nm). NIR has the highest penetration power and goes deep into tissues unattenuated. The difficulty with Spectral data is that there are far too many features compared to the number of data points. And there is a high risk of overfitting.

3.1 Applications:

- NIR spectroscopy is waiting to penetrate our everyday lives on mobile phones, wearables and many other appliances.
- It could be used in a supermarket that suggests a diet on the fly based on our cholesterol levels just by shining a beam of light.
- It could be used in a Weighing machine which not only determines the weight but also the vitamin levels in the blood.

3.2 Problem Statement:

Classify the blood cholesterol and hemoglobin levels in the blood as ok, high and low for each individual.

3.3 Data Description:

- **Absorbance:** There exists 170 columns labelled as absorbance0, absorbance1 and so on. This is an intensity spectrum of the target blood response to pointed light.
- **Temperature:** Temperature at the time of the measurement.
- **Humidity:** Humidity at the time of the measurement.
- **Id:** Unique identifier assigned to each measurement.
- **Std:** Standard deviation value.
- **Hdl_cholesterol_human:** The level of cholesterol high. Can be low, ok or high.
- **Cholesterol_ldl_human:** The level of cholesterol low. Can be low, ok or high.
- **Hemoglobin(hgb)_human:** The level of haemoglobin: Can be low, ok or high

3.4 Exploratory Data Analysis:

EDA will reveal the data set's underlying structure as well as trends, patterns, and linkages that aren't immediately apparent. It will also assist in deriving trustworthy conclusions from a large amount of data by carefully and deliberately looking at it via an analytical lens. These revelations will eventually lead to the selection of a suitable predictive model.

The below figures represent the distribution of the classes of each of the target variables in the dataset. It is observed that the classes are highly imbalanced for each of the target variables.

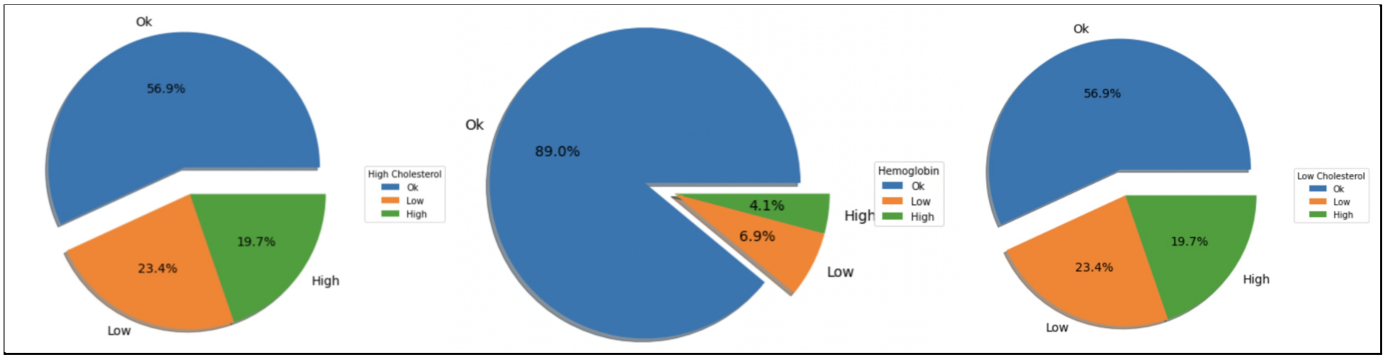


Fig 2.1 Hemoglobin *Fig 2.2 High cholesterol* *Fig 2.3 Low Cholesterol*

Fig 2. Distribution of the Classes of Each of The Target Variables.

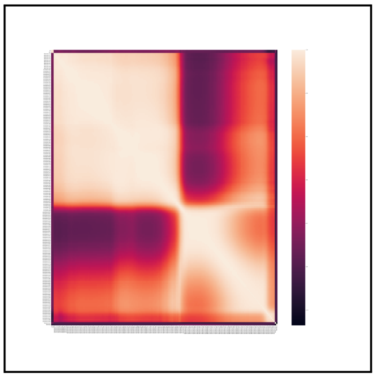


Fig 3. Correlation heatmap of the features

Figure 3 is a correlation heatmap to check the correlation between the features. We observe that there exists multicollinearity in the data i.e. we cannot distinguish the individual effects of independent columns. And one predictor variable in the model can be linearly predicted from others. Thus, we can not make a direct inference as the number of features is very high.

Figure 4 shows histograms for some of the absorbance features to check if they follow any distribution. We found that the absorbance values are normally distributed as seen from the graphs below.

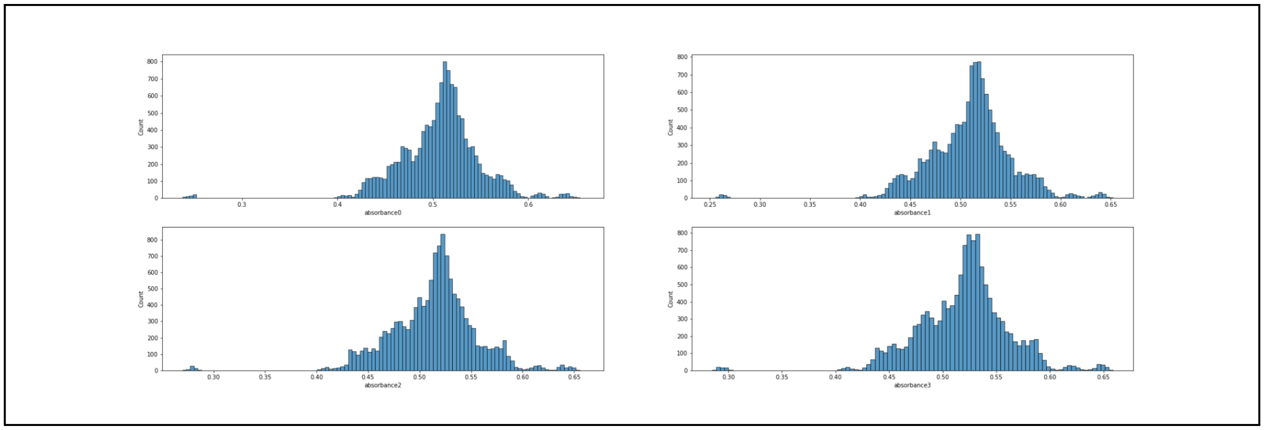


Fig 4. Distribution of Absorbance Values

4. Feature Extraction and Preprocessing

We started by cleaning the data i.e. data pre-processing. As the data consist of many features, feature selection was crucial. We then implemented modelling techniques to classify the target variables.

4.1 Data Preprocessing:

Data preprocessing is an essential process because it has a direct impact on the project's success rate. Because the data in the real world is unclean, this minimizes the complexity of the data.

We followed the below steps as part of preprocessing:

1. **Checking NA values:** The dataset did not contain NA values.
2. **Outlier Detection:** We performed outlier detection by computing quartile values and the interquartile range (IQR). We make use of **Z-Score**, which is also termed the standard score. This value helps us understand how far the data point is from the mean. We set a threshold value for the Z-Score to mark the data points as outliers. [3]

   Z-Score is given by,

   $$Z - Score = \frac{(dataPoint - mean)}{Standard\ Deviation}$$

   We define an outlier threshold as 3, as 99.7% of the data points lie between $\pm$ 3 standard deviation (using the Gaussian Distribution approach).

   IQR is given by, $IQR = Quartile3 - Quartile1$

   Lower and upper bound are calculated using the following,

   $$upper = Q3 + 1.5 * IQR$$
   $$lower = Q1 - 1.5 * IQR$$

   Approach:

   (i) On All the Features: We have 174 features and on performing outlier elimination on all the features. The dataset size reduces to almost half.

   (ii) On Specific Features: We can manually select the features for which we wish to perform outlier detection and elimination based on experimentation.

   Figure 5 depicts the outlier detection and removal for absorbance0. From the first box plot, we observe that there are many points that lie beyond the minimum and maximum value of the plot. These points act as the outliers. The second box plots represent the data for absorbance0 after removing the outliers.

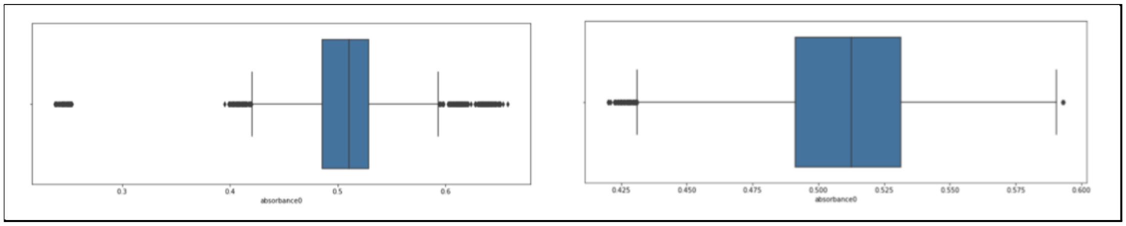


Fig 5. Absorbance Feature: Before and After Detecting Outliers

3. **Normalization:** As the data already follows a normal distribution, normalization will not help us make the model perform better. We implemented Standard Scaling to verify the intuition.
4. **Sampling:** As the classes of the target variables are highly imbalanced, we oversampled the minority classes using SMOTE (Synthetic Minority Oversampling Technique) to balance the data. SMOTE selects the examples that are close in the feature space by drawing a line between the examples in the feature space and drawing a new sample at a point along that line. It first chooses a random example from the minority class. Further, it finds *k* of the nearest neighbours for that point. Then, it randomly selects a neighbour and a synthetic example is created at a randomly selected point between the two examples in feature space. [5]

4.2 Feature Engineering:

The dataset contains 170 Absorbance columns followed by the temperature and humidity at which the light beam was passed through the blood sample. As the number of features was very high and those were the absorbance values, we could not eliminate the features manually and hence, Feature Elimination was a crucial task in this project, as there is a possibility that the model overfits the data.

We performed feature elimination using the techniques mentioned below:

1. **Recursive Feature Elimination (RFE):** Recursive feature elimination (RFE) is a feature selection method that fits a model and removes the weakest feature (or features) until the specified number of features is reached. [5] Given an external estimator that assigns weights to features, the RFE will select features by recursively considering smaller and smaller sets of features. First, the estimator is trained on the initial set of features and the importance of each feature is obtained either through any specific attribute or callable. Then, the least important features are pruned from the current set of features. That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached. We selected SVR (Support Vector Regression) as the estimator for RFE. [6]

2. **Principal Component Analysis (PCA):** Principal Component Analysis performs dimensionality reduction by projecting to a lower dimension and preserving as much data variance as possible. It identifies relationships between different features and then couples them. The data is first centred for each feature, and then Singular Value decomposition is applied in order to project the data to a lower-dimensional space. [7]

5. Models

As this was a multiclass classification challenge and the aim of the project was to classify the Cholesterol and Hemoglobin levels as High, Low and Ok for each entry, we planned to perform the below-mentioned models:

5.1 Logistic Regression with L2 Regularization:

Logistic Regression is a linear model that is used when observations must be assigned to a discrete set of classes. Regularization is a technique used to prevent overfitting. It adds a regularization term, in order to prevent overfitting of the model. The regression model which uses L1 regularization is called Lasso Regression and the model which uses L2 is known as Ridge Regression. [8]

Ridge Regression (L2 norm): L2-norm loss function is also known as the least-squares error (LSE).

$$w\,* \;=\; minimization\;of \sum i \;=\; 1n[log(1\;+\;exp(-\;zi))]\;+\;\lambda\;*\;\sum\,(wj\,)^2$$

$\sum$ **(wj)^2** is a **regularization term** and $\sum$ [log(1+exp(-zi))] is the **Loss term.** λ is a hyperparameter.

We added the regularization term (i.e. squared magnitude) to the loss term to make sure that the model does not undergo an overfit problem.

Hyperparameter Tuning:

- *penalty: {'l1', 'l2', 'elasticnet', 'none'}*
- *solver: {'newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'}*

The model formed the best using 'saga' solver and 'l2' regularisation.

5.2 Random Forest Multi-Output Classifier:

Random Forest Classifier is an ensemble method, which constructs a multitude of decision trees at the time of training. The output of the random forest is the class selected by most of the trees. Multi-Output Classifier classifies multi labels on multiple target variables. It fits one classifier per target and allows the model to estimate a series of target functions on the same predictor matrix, in order to predict a series of responses. [10]

5.3 Random Forest Classifier:

Random Forest Classifier is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. It is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the majority to improve the classification accuracy of that dataset. Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. [9]

Hyperparameter Tuning:

- *max_depth: {4, 6, 8, 10, 12}*
- *min_samples_split: {2, 5, 10, 15, 20, 25, 30}*
- *Criterion: {'gini', 'entropy'}*

The model formed the best for smaller max-depth i.e. 4-8, smaller splits i.e. 5-10 and 'entropy' as the criterion.

5.4 Extra-Tree classifier:

Extra-Trees Classifier is a method of ensemble learning. To increase the predictivity of the classifier, the approach constructs extra trees in sub-samples of datasets and uses majority voting. The strategy lowers variance by taking this approach. To obtain the best of the thresholds as a splitting rule, the approach applies a random threshold to each feature of sub-samples. [11]

5.5 Radius Neighbor Classifier:

Radius Neighbors Classifier is a machine learning technique for classification. It's a variation of the k-nearest neighbours technique that makes predictions based on all examples within a radius of a new example rather than just the k-nearest neighbours. As a result, the radius-based technique to identify neighbours is more suited to sparse data, as it prevents far-flung samples from contributing to a forecast. [14]

Hyperparameter Tuning:

- *model__radius: arange(0.5, 7)*

The model formed the best for the radius in the range of 1.3

5.6 CatBoost Classifier:

CatBoost is a boosted decision tree algorithm which makes use of Gradient boosting. CatBoost uses a slightly different version of gradient boosting called ordered boosting. It yields good results without extensive training of data and provides support for descriptive data formats. Also, it reduces the need for extensive hyper-parameter tuning and reduces the possibility of having an overfitted model. It generates generalised models. [12]

Hyperparameter Tuning:

- *depth: { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}*
- *iterations: {100, 250, 500, 1000}*
- *learning_rate: {0.03, 0.001, 0.01, 0.1, 0.2, 0.3}*
- *l2_leaf_reg: {1, 3, ,5, 10, 100}*
- *border_count: {5, 10, 20, 30, 50, 100, 200}*

The model formed the best for 1000 iterations, smaller depth of 4-7 and a learning rate of 0.001.

5.7 Neural Network:

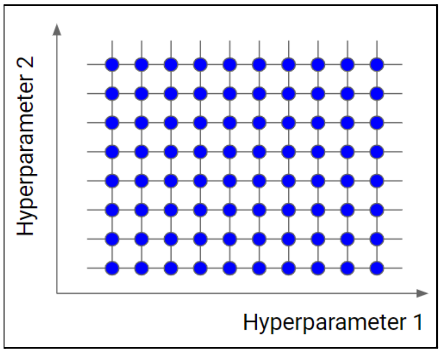
Neural networks are composed of node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network. [13]

We used “Relu” as the activation function in the input and hidden layers, and “softmax” for the output layer as this is a multiclass classification problem. We used “categorical accuracy” as the scoring metric.

6. Experiments and Evaluation

6.1 Hyperparameter Tuning:

One of the most critical components of a machine learning workflow is hyperparameter tuning. The improper values for the hyperparameters can lead to incorrect findings and a model that performs poorly. Hyperparameters are model parameters that have their values set before the training. The process of obtaining the correct settings for the model's hyperparameters is known as hyperparameter tuning. We used Grid Search for the hyperparameter tuning. [15]



Grid Search: Grid search divides the hyperparameter domain into distinct grids. Then, using cross-validation, it attempts every possible combination of values in this grid, computing some performance measures. The ideal combination of values for the hyperparameters is the point on the grid that maximizes the average value in cross-validation. Grid search is a comprehensive technique that considers all possible combinations in order to locate the best point in the domain. [15]

6.2 Model Evaluation Strategies:

As this is a classification problem, we used F1 score, Precision and Recall as the evaluation metrics.

Figure 6 shows the change in evaluation metric values as the number of features increase for Random Forest Classifier, Extra Tree Classifier and Logistic Regression using RFE.

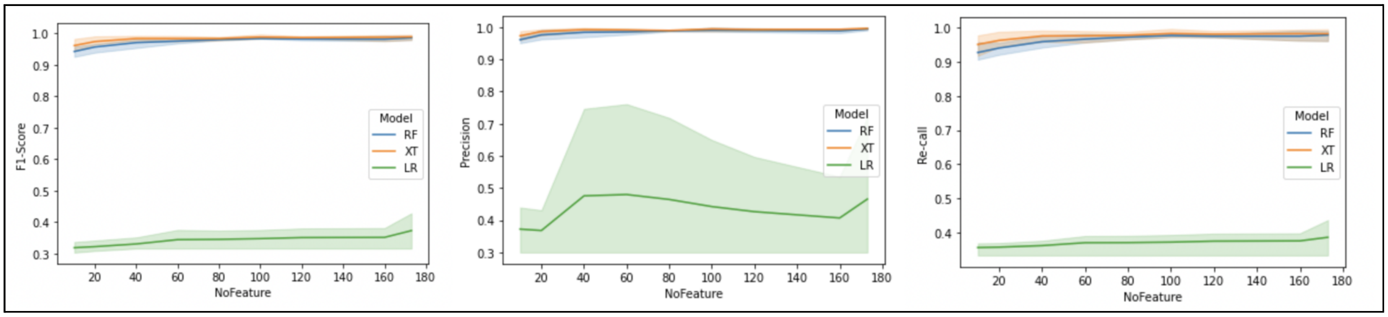


Fig 6. Change in evaluation metric as no of features increase in RFE

Figure 7 shows the change in evaluation metric values as the number of features increase for Random Forest Classifier, Extra Tree Classifier and Logistic Regression using PCA.

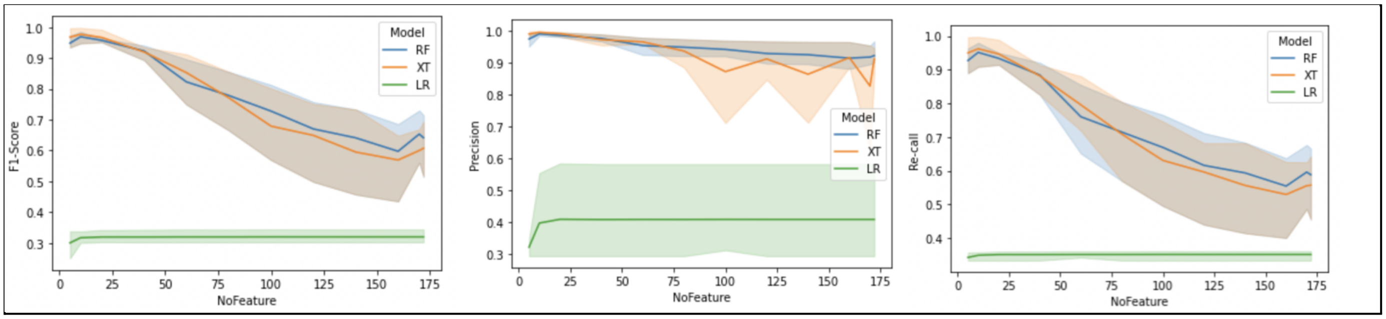


Fig 7. Change in evaluation metric as no of features increase in PCA

Figure 8 shows the change in accuracy and loss after each epoch for neural networks.. The accuracy increased and loss decreased as the number of epochs increased.

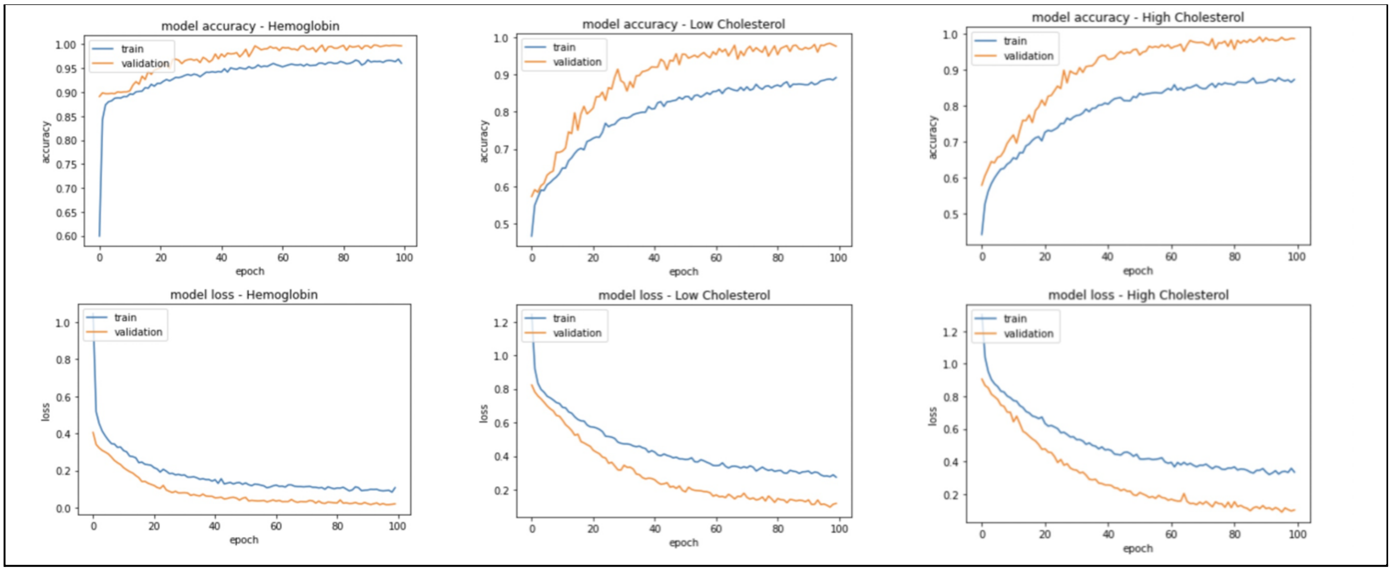


Fig 8.1 Hemoglobin *Fig 8.2 Low Cholesterol* *Fig 8.3 High Cholesterol*

Fig 8. Change in accuracy and loss with respect to epochs

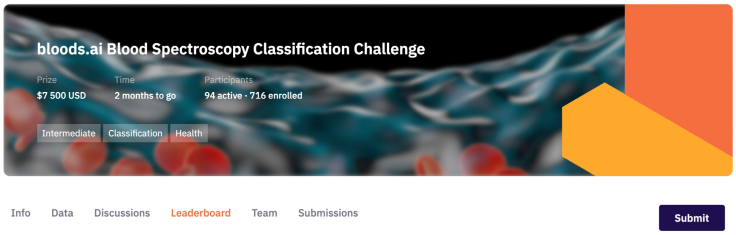
7. Results

1. After Performing sampling on “std” Column and selecting all All the features:

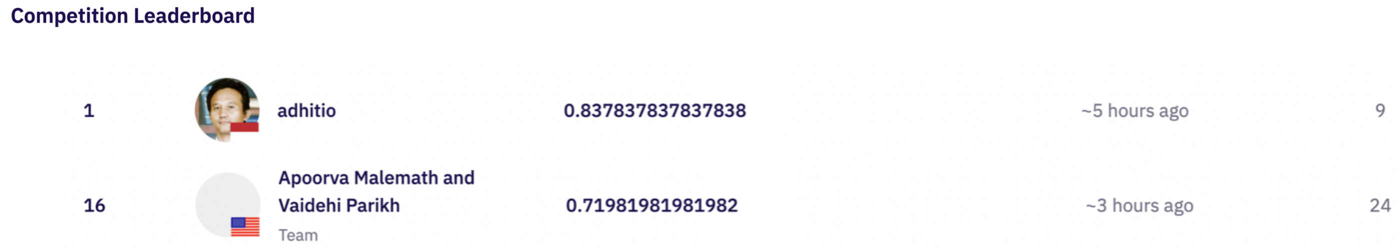
No Features	Model	Target	F1-Score	Precision	Re-call	Score
173	LR	HCH	0.254029	0.346694	0.337460	0.663963963963964
		HHH	0.314635	0.297922	0.333333	
		CLH	0.357864	0.365992	0.375838	
	RF	HDL	0.979096	0.988006	0.970952	0.630630630630631
		HHH	0.981148	0.992222	0.970496	
		CLH	0.987701	0.994017	0.981665	
	XT	HDL	0.999200	0.999519	0.998883	**0.682882882882883**
		HHH	0.981148	0.992222	0.970496	
		CLH	0.987701	0.994017	0.981665	
	RN	HDL	0.875516	0.893462	0.879714	0.639189189189189
		HHH	0.944181	0.952851	0.950815	
		CLH	0.903730	0.908315	0.907237	

2. Using Balanced data and Selecting Features using RFE:

No Features	Model	Target	F1-Score	Precision	Re-call	Score
40	LR	HCH	0.275108	0.403929	0.347955	0.707657657657658
		HHH	0.314875	0.298354	0.333333	
		CLH	0.342810	0.358334	0.365702	
	RF	HDL	0.959325	0.973912	0.947077	0.688288288288288
		HHH	0.967397	0.991555	0.945160	
		CLH	0.953274	0.982848	0.929144	
	XT	HDL	0.994708	0.996140	0.993295	0.645495495495495
		HHH	0.967397	0.991555	0.945160	
		CLH	0.953274	0.982848	0.929144	
	RNC	HDL	0.685510	0.600407	0.496508	**0.71981981981982**
		HHH	0.838045	0.894750	0.845546	
		CLH	0.573294	0.610601	0.576437	



This is an ongoing challenge with 716 participant in it and we are currently on the rank 16 with a score of 0.7198.


Competition Leaderboard

1	adhitio	0.837837837837838	~5 hours ago	9
16	Apoorva Malemath and Vaidehi Parikh Team	0.71981981981982	~3 hours ago	24

8. Conclusion and Future Scope

We arrived at the following conclusion after applying all of the proposed methodologies to the blood spectroscopic dataset, focusing on classifying Cholesterol and Hemoglobin levels as High, Low, or Ok for each individual:

1. Normalization did not work towards improving the model, as the data already shows a normal distribution.
2. Oversampling the minority classes using (SMOTE) helped us get a better F1 score, Precision and Recall for simpler models. However, it did not have an impact on ensemble methods.
3. Outlier Detection also helped us get a better F1 score, Precision and Recall scores.
4. RFE worked better than PCA for feature elimination. As RFE is a wrapper method i.e. it fits and evaluates models with different subsets of data whereas, PCA is a filter method.
5. As there are too many features compared to the data points, Simple models and regularization worked better in preventing overfitting.
6. On using tree-based algorithms, it is observed that the models performed well with lesser depths, as higher depth causes high variance and causes overfitting.
7. Logistic Regression with L2 regularisation worked better with a lesser number of features, as logistic regression is not immune to multicollinearity.
8. Out of all the models, Radius Neighbor Classifier and Logistics Regression with L2 Regularization worked better whereas Extra Tree Classifier and Random Forest were overfitting the data.

In the future, we would like to explore ensemble models which combine the results of multiple models. We would also like to explore neural networks with dropout layers and callbacks to handle overfitting. The project was very domain-specific. Therefore, we would like to gain domain knowledge to better understand the data as well as treat outliers in a better way.

9. References

Github: https://github.com/apoorvamalemath/Projects/tree/main/Blood%20Spectroscopy%20Classification%20Challenge
Dataset: https://zindi.africa/competitions/bloodsai-blood-spectroscopy-classification-challenge/data
Colab: https://colab.research.google.com/drive/1Dzc_5D2ZUVQbQS3Ovr1QxpxWhidgRoHg

[1] bloods.ai - bloods.ai - https://bloods.ai/en
[2] Blood Spectroscopy Classification- bloods.ai Blood Spectroscopy Classification Challenge - https://zindi.africa/competitions/bloodsai-blood-spectroscopy-classification-challenge/data
[3] Outlier Detection - https://www.geeksforgeeks.org/detect-and-remove-the-outliers-using-python/
[4] SMOTE – https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/
[5] RFE - https://www.scikit-yb.org/en/latest/api/model_selection/rfecv.html
[6]RFE - https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html
[7] PCA - https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html
[8] Logistics with L2 - https://medium.com/@aditya97p/l1-and-l2-regularization-237438a9caa6
[9] Random Forest Classifier - https://www.javatpoint.com/machine-learning-random-forest-algorithm
[10] Random Forest Multioutput Classifier - https://scikit-learn.org/stable/modules/multiclass.html
[11] Extra Tree - https://www.datatechnotes.com/2020/06/classification-example-with-scikit.html
[12] CatBoost - https://www.analyticsvidhya.com/blog/2017/08/catboost-automated-categorical-data/
[13] Neural Network - https://www.ibm.com/cloud/learn/neural-networks
[14] Radius Neighbor - https://machinelearningmastery.com/radius-neighbors-classifier-algorithm-with-python/
[15] Hyperparameters - https://www.yourdatateacher.com/2021/05/19/hyperparameter-tuning-grid-search-and-random-search/