A Quantitative Test for SARS-Cov-2 Infection

Zefeng Xu

Data Science Institute (DSI), Brown University

GitHub: https://github.com/GunnerForever/1030_project.git

1. Introduction

The world has been plagued by the Coronavirus Disease 2019 (COVID-19) pandemic that originated in Wuhan, China in December 2019 for 4 years now. The disease is caused by infection with a strain of coronavirus known as Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2)[1]. Testing (such as Nucleic Acid Tests on infected secretions[1]) for infections of COVID-19 is critical for treatments and medications used. This project aims to add a quantitative approach to the testing and diagnosis of COVID-19 by using patient gene expression data.

1.1 Data Source

The dataset used in this project is publicly available in the Gene Expression Omnibus (GEO) database supported by National Center for Biotechnology Information (NCBI)[2] with dataset ID GSE212041[3]. This dataset was the supplementary data for the study conducted by Lasalle et al. (2022)[4], and was constructed by Neutrophil Bulk RNA-seq enriched from fresh whole blood drawn from patients at the start of the pandemic[4].

1.2 Data Description

This data is longitudinal, including 781 samples from 306 hospitalized COVID-19 positive patients, 78 symptomatic controls[1], and 8 healthy controls from a window of 3 timepoints: Day 0, Day 3, and Day 7 post-hospitalization[3]. Each sample can be identified by a unique sample ID of format [patient id]_D[time point], with the exception of healthy controls

[1] These numbers are inconsistent with reality. The actual numbers should be 304 COVID-19 positive patients (310 possible IDs minus 6 patients without any data, which will be mentioned in Section 2.3), 76 symptomatic controls, and 8 healthy controls.

with ID of format H[1-9]_H. Each sample consists of values for 60640 features where each feature is a unique human gene identified by Ensembl IDs[2], and the corresponding value is its normalized expression level in unit of transcripts per million (TPM). The dataset provides two sets of labels for each sample: covid-19 status and acuity.max, with values shown in Table 1. The target labels of this project are the covid-19 status labels. Since this label set is binary, this project is a binary classification problem.

Patient Group	covid-19 status	acuity.max
COVID-19 Positive	1	1 to 5
Symptomatic Control	0	1 to 5
Healthy Control	0	H

Table 1: Summary of values of label sets. The acuity.max labels range from 1 to 5 inclusive and the higher value, the more severe the symptoms.

2. Exploratory Data Analysis (EDA)

The Exploratory Data Analyses were performed on original dataset without any kind of preprocessing. Below are three major insights from EDA.

2.1 Insight 1 – No Negative Entries

All features of this data are non-negative because all entries are expression levels, which are normalized counts but counts can never be less than zero.

2.2 Insight 2 – Duplicated Samples

This dataset contains three different types of duplicated samples.

- Type 1 Duplication

For some patients, there are two samples (A and B) at a timepoint.
There are 4 such cases in the dataset: e.g., (43_D7A, 43_D7B).

[2] Ensembl IDs: identifiers for genes as per the Ensembl (European Bioinformatics Institute and the Wellcome Trust Sanger Institute) database[9].

- Type 2 Duplication

For some patients, there exists a DE timepoint apart from D0, D3, and D7. Such timepoints correspond to changes in patient status (e.g. intubation, removal from ventilator)[3]. There are 43 such DE samples.

- Type 3 Duplication

This is a combination of Type 1 and 2 duplications. There is only one such pair in the dataset: (302_DEA, 302_DEB).

2.3 Insight 3 – Missing Samples

This dataset exhibits a massive number of missing samples. There are in total 310+76+8=394 patients. Note that six patients with IDs 141, 142, 183, 202, 269, and 270 do not have any samples. Excluding the DE samples, the healthy controls, who only have data from one special timepoint, and counting A/B samples as one, the information about missing samples is summarized in Table 2.

Number of Available Timepoints	Number of Patients
1	160
2	99
3	121
Total	380

Table 2: Summary of missing samples. The number 380 is calculated as total patients - patients with no data - healthy controls = 394-6-8 = 380.

3. Methods

This section focuses on two general topics: data preprocessing, which includes approaches adopted to make the original messy dataset suitable for machine learning algorithms, and the cross-validation pipeline, which includes the models trained and the hyperparameters tuned.

3.1 Preprocessing

- Sample removal

All duplicated samples mentioned in Section 2.2 and healthy controls were removed to avoid potential confusion.

- Missing samples ignored rather than imputed

There are two major reasons for this decision: first, the missing samples are also missing corresponding target labels but imputation of target labels is impossible; second, there does not exist any biologically reasonable method to impute genomics data.

- Transformed onto log scale & Dimensionality reduced

All entries in this dataset were transformed onto a logarithm with base 2 scale by first adding 1 and then take log2. Also, the original number of features (60640) is too many for machine learning. This project extracted the top 300 Highly-Variable Genes (HVG) by adopting methods from Scanpy package[5][6]. HVGs are genes whose expression levels exhibit significant variations across samples. These variations may be indicative of disease states and biological characteristics.

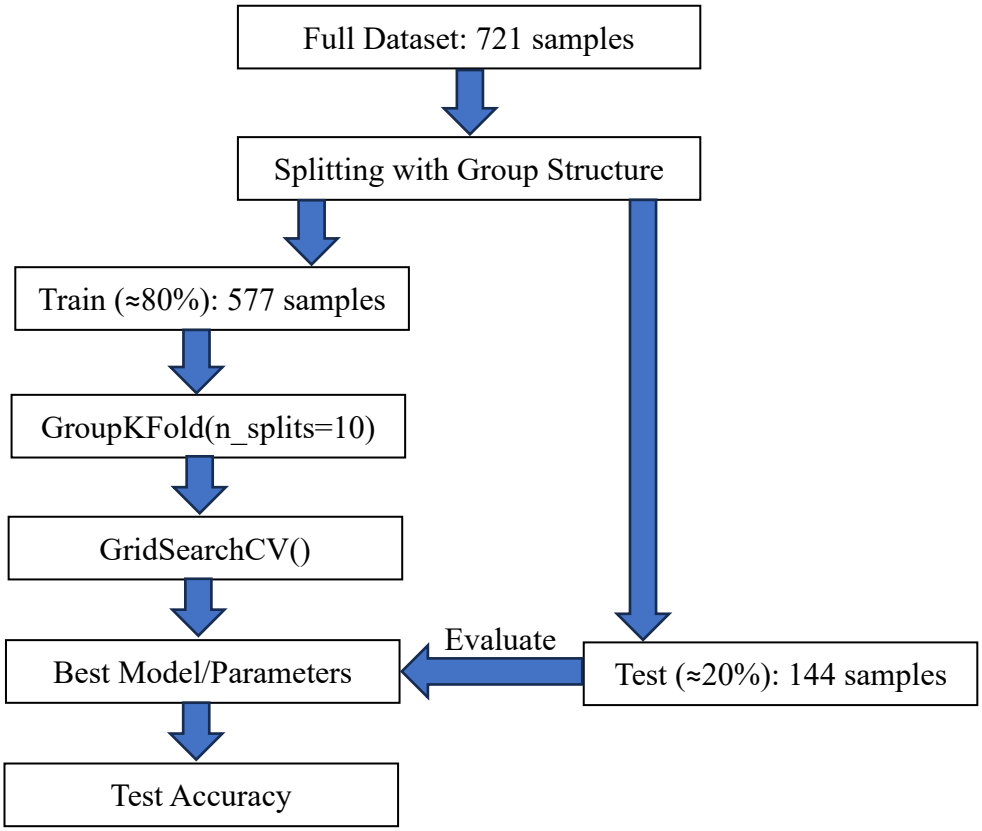
- Columns normalized

Each column was normalized by removing the mean and scaling to unit variance using StandardScaler from scikit-learn package[7].

The resulting dataset had 721 samples and 300 columns after preprocessing.

3.2 Cross Validation Pipeline

In order to respect the group structure of patients, group-based splitting strategies from scikit-learn[7] were adopted. Below is a flow chart for the cross-validation pipeline used in this project. The pipeline is repeated 10 times using 10 different random states for each ML algorithm.



3.3 Models Trained and Hyperparameter Tuned

The models trained and hyperparameters tuned for each model are shown in Table 3.

Table 3 Panel A	K Nearest Neighbor (KNN)
Tuning Round	Tune 1
Parameters Tuned	n_neighbors
Value Ranges	n_neighbors: [3, 4, 5, 6, 7, 8, 9]

Table 3 Panel B	Logistic Regression	
Tuning Round	Tune 1	Tune 2
Parameters Tuned	penalty C	l1_ratio C
Parameters Controlled	solver='saga'	penalty='elasticnet' solver='saga'

Value Ranges	penalty: [None, ‘l1’, ‘l2’] C: [0.01, 0.1, 1, 10]	l1_ratio: [0.1, 0.3, 0.5, 0.7, 0.9] C: [0.01, 0.1, 1, 10]

Table 3 Panel C	Support Vector Machine		
Tuning Round	Tune 1	Tune 2	Tune 3
Parameters Tuned	C	kernel C gamma	C degree gamma
Parameters Controlled	kernel=‘linear’		kernel=‘poly’
Value Ranges	C: [0.01, 0.1, 1, 10]	kernel: [‘rbf’, ‘sigmoid’] C: [0.01, 0.1, 1, 10] gamma: [‘scale’, ‘auto’, 1e-3, 1e-2, 1e-1, 1, 10]	C: [0.01, 0.1, 1, 10] degree: [3, 4, 5, 6, 7] gamma: [‘scale’, ‘auto’, 1e-3, 1e-2, 1e-1, 1, 10]

Table 3 Panel D	Random Forest	eXtreme Gradient Boosting (XGBoost)
Tuning Round	Tune 1	Tune 1
Parameters Tuned	max_depth criterion	max_depth learning_rate
Value Ranges	max_depth: [5, 10, 15, 20, 25, 30] criterion:	l1_ratio: [0.1, 0.3, 0.5, 0.7, 0.9] C:

	[‘gini’, ‘entropy’, ‘log_loss’]	[0.01, 0.1, 1, 10]

Table 3 Panels A,B,C,D: Hyperparameters tuned for five algorithms

4. Results

4.1 Test Accuracy

After hyperparameter tuning, each tuning round returns test accuracies for 10 different random states. Table 4 summarizes these values. The baseline accuracy of this dataset is 0.890430, which is the accuracy if we always predict the majority class (Class 1).

Table 4 Panel A	K Nearest Neighbor (KNN)
Tuning Round	Tune 1
Test Accuracy Mean	0.902083
Test Accuracy Standard Deviation	0.020471

Table 4 Panel B	Logistic Regression	
Tuning Round	Tune 1	Tune 2
Test Accuracy Mean	0.909028	0.913194
Test Accuracy Standard Deviation	0.019752	0.017361

Table 4 Panel C	Support Vector Machine (SVM)		
Tuning Round	Tune 1	Tune 2	Tune 3
Test Accuracy Mean	0.904861	0.917361	0.904167
Test Accuracy Standard Deviation	0.017025	0.015972	0.022352

Table 4 Panel D	Random Forest	eXtreme Gradient Boosting (XGBoost)
Tuning Round	Tune 1	Tune 1
Test Accuracy Mean	0.913194	0.902083
Test Accuracy Standard Deviation	0.028336	0.024932

Table 4 Panels A,B,C,D: Summary of test performance of the five algorithms

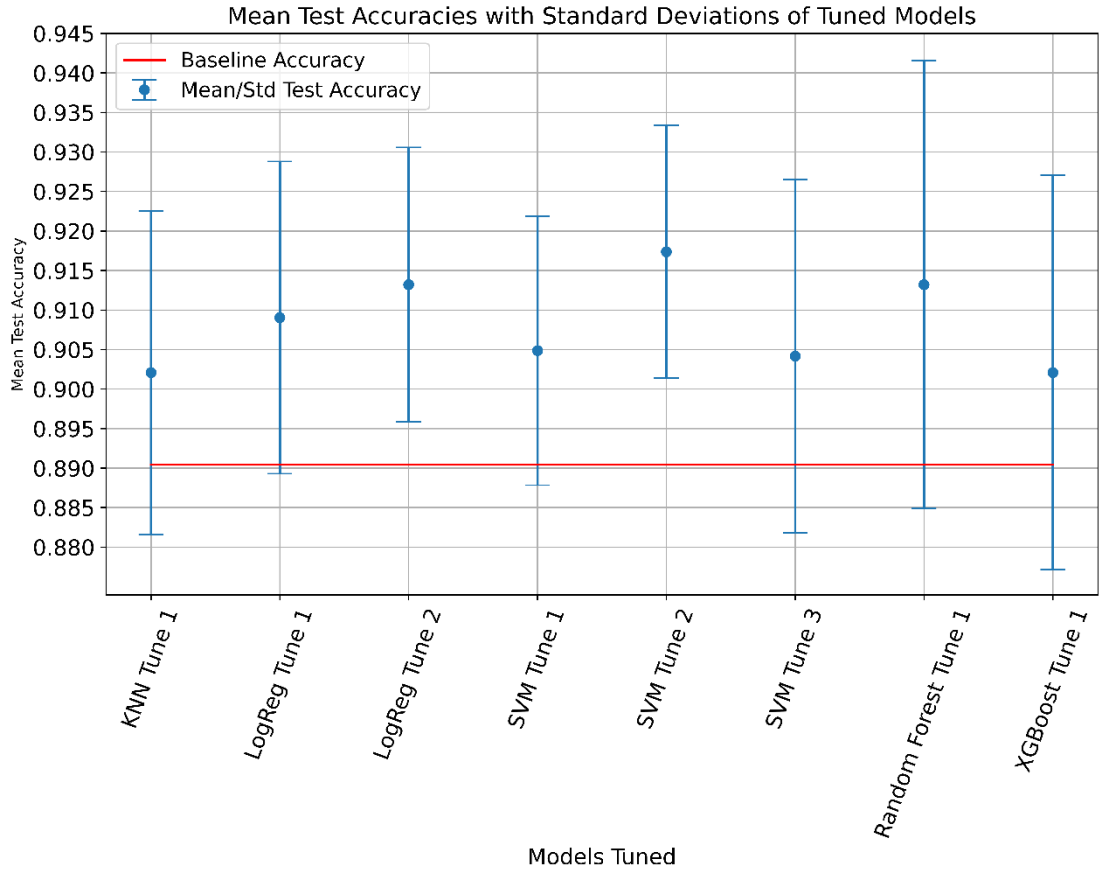


Figure 1: Plot of information in Table 4

As can be seen from Table 4, as well as in Figure 1, all the tuned mean test accuracies are higher than the baseline accuracy with fairly small standard deviations. However, after adding the uncertainties (stds), some models may underperform than the baseline one. The best-performing model is SVM Tune 2, with the highest accuracy mean and lowest accuracy standard

deviation. Note that its performance is still better than the baseline even if uncertainties are added.

4.2 Global Feature Importance

This section focuses on interpretation of the models. Three different metrics for global feature importance, that is, the importance of a feature on the final prediction of a model, were calculated, as shown below.

4.2.1 Permutation Score

This project experimented with Permutation Scores using SVM. The top 20 features sorted by permutation score are shown in Figure 2.

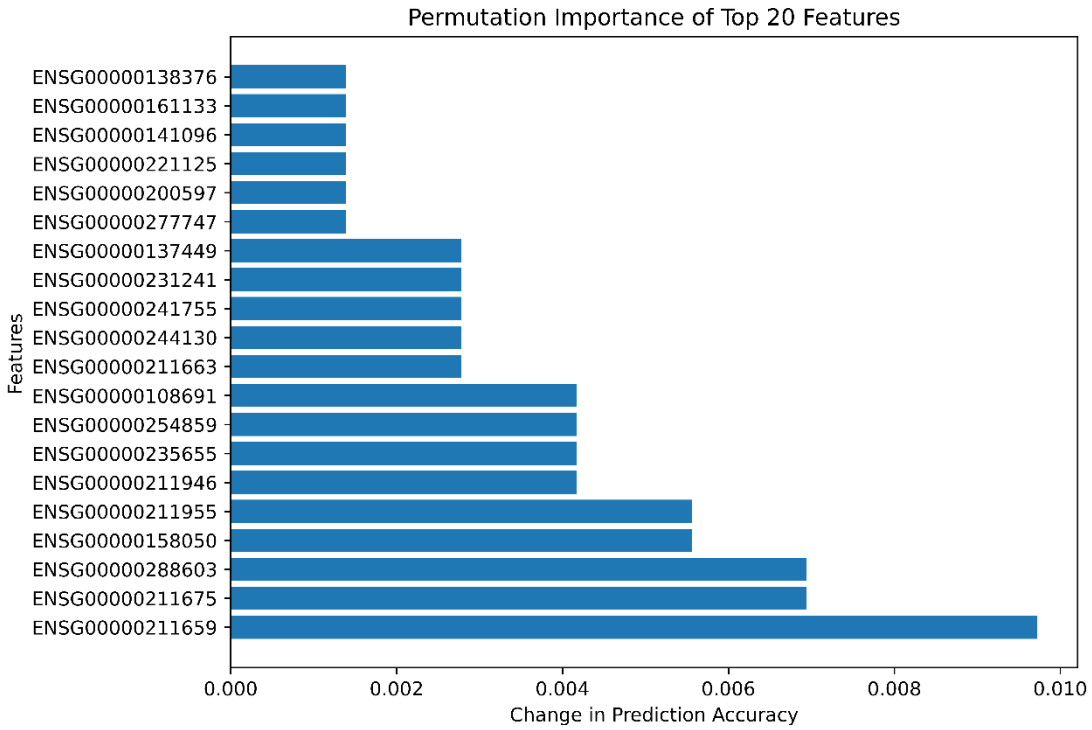


Figure 2: Top 20 features sorted by Permutation Importance

4.2.2. Regression Coefficients

The coefficients of Logistic Regression can also be viewed as a measure of feature importance. Figure 3 shows the top 20 features sorted by magnitude of coefficient regardless of the sign. A positive/negative coefficient imply positive/negative impact on the log-odds (and thus the probability) of

predicting the positive class.

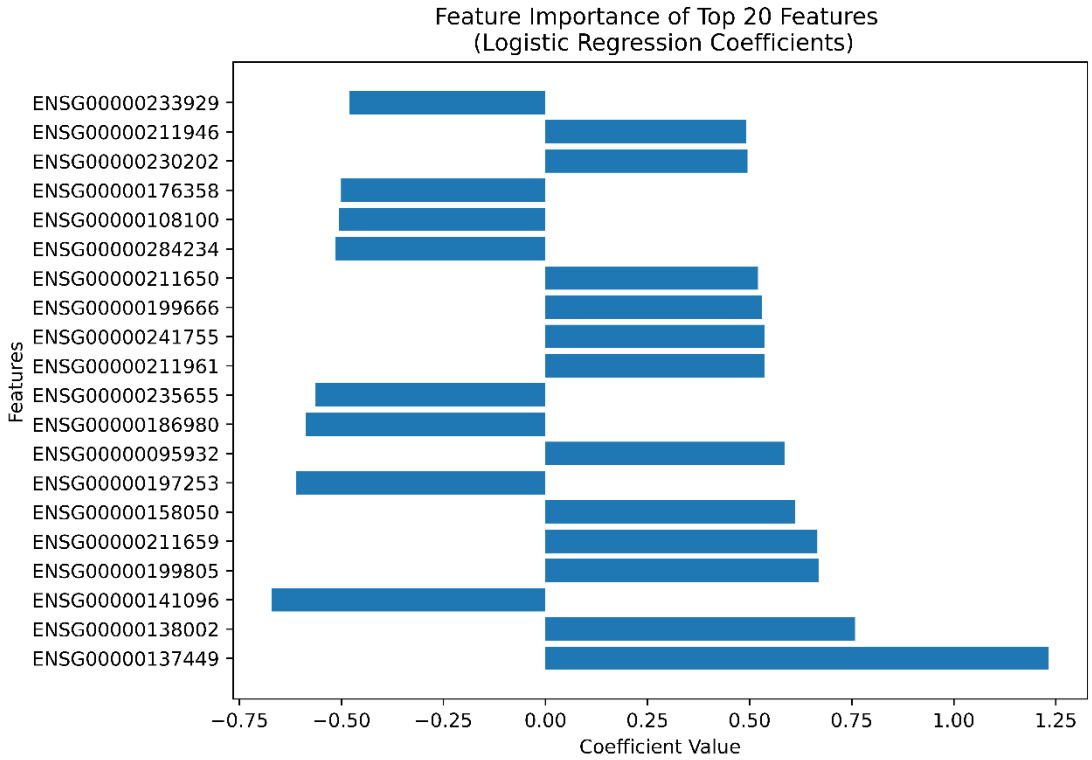


Figure 3: Top 20 features sorted by magnitude of coefficient in Logistic Regression

4.2.3 Random Forest Feature Importance

This project also experimented with the built in impurity-based feature importance metric of Random Forest. This metric measures the reduction of impurity, which is a measure of how mixed the classes are in the dataset, when using a specific feature in construction of a "random forest". Figure 4 (next page) shows the top 20 features.

4.2.4 Implications

Since all features of this dataset are human genes, a subset of "important" features is just a subset of human genes that has some quantitative meaning for the target label, which represents disease status. Particularly important features (genes) (such as the gene with Ensembl ID ENSG00000211659, ranked as 1st, 5th, 3rd in top 20 genes selected by the three metrics respectively) may have stronger correlation with the disease. These selected genes can then be given to biologists to study the specific

biological relationships between them and the disease.

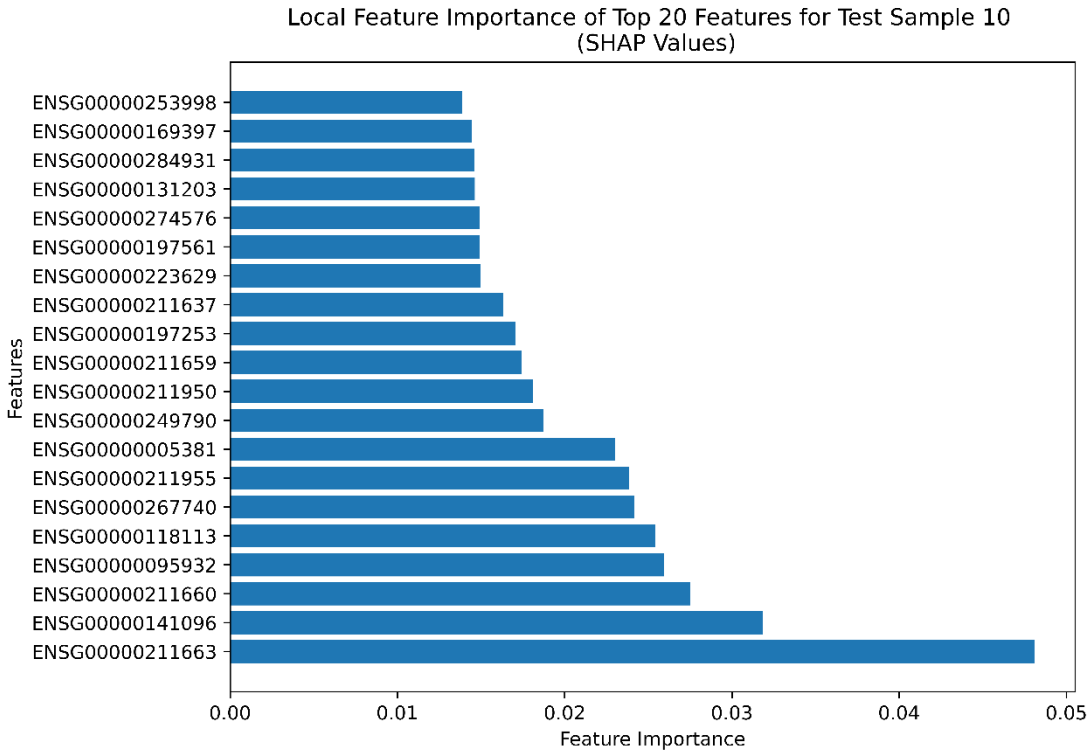


Figure 4: Top 20 feature sorted by Random Forest's feature importance metric

4.3 Local Feature Importance

As opposed to global feature importance, to study the impact of the features on single observations, SHAP[9] values were calculated. Figure 5 (next page) shows the top 20 features sorted by SHAP values for the testing sample at index 10 (randomly picked). As can be seen, the particularly important gene (ENSG00000211659) mentioned in Section 4.2.4 is still important for this specific sample (ranked as 11th).

5. Outlook

The issue of missing/lack of data must be properly addressed. Since there is not enough data, it is hard for models to perform well. Looking for datasets with more, or at least less missing, data could be helpful. In addition, the samples removed in Section 3.1 could have been used but the way of using them is subject to further consideration. The issue that the

models did not well outperform the baseline might be alleviated by proposing a strategy that stratify samples according to target labels while still keeping group structure. Also, there might be some models specifically targeting biomedical noisy data, and experimenting with them could possibly be beneficial.

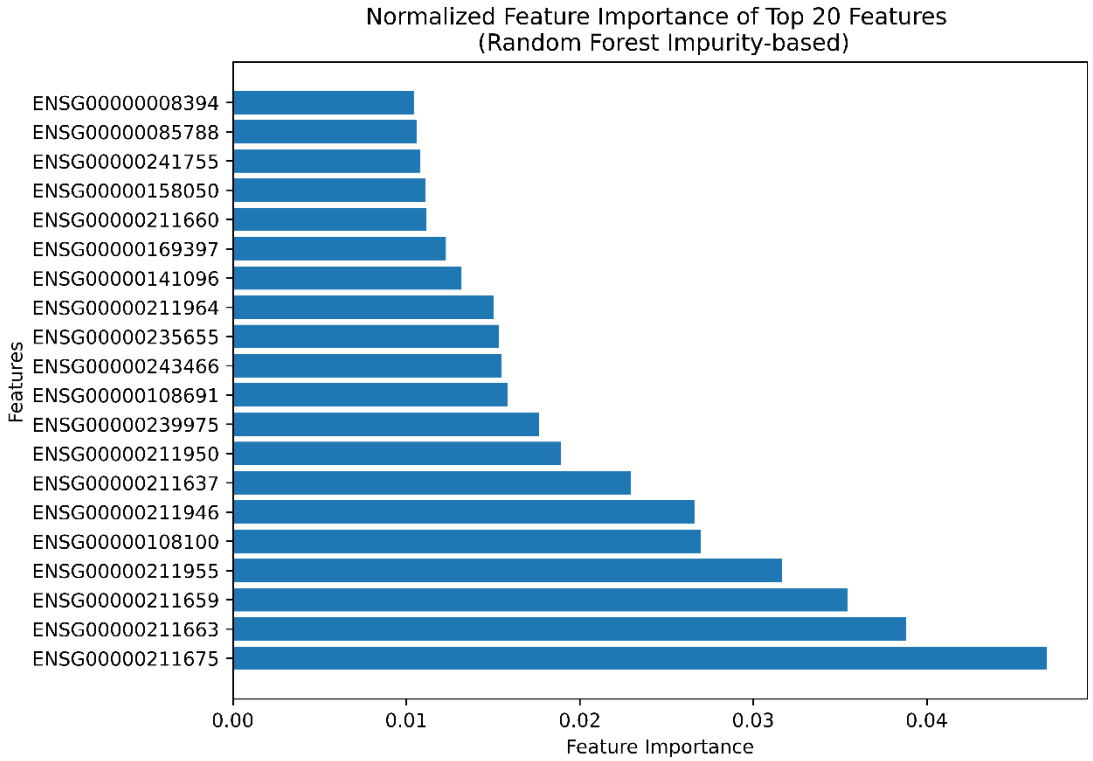


Figure 5: SHAP values of top 20 features for test sample at index 10

6. References

1. COVID-19: https://en.wikipedia.org/wiki/COVID-19
2. The Gene Expression Omnibus database: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4944384/#:~:text=Gene%20Expression%20Omnibus%20(GEO)%20is,studies%20of%20high%2Dthroughput%20gene
3. GSE212041 https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE212041
4. Original article where this dataset is supplementary data
   LaSalle TJ, Gonye ALK, Freeman SS, Kaplonek P et al. Longitudinal

characterization of circulating neutrophils uncovers phenotypes associated with severity in hospitalized COVID-19 patients. Cell Rep Med 2022 Oct 18;3(10):100779. PMID: 36208629

5. Scanpy package: https://scanpy.readthedocs.io/en/stable/index.html#
   Wolf, F., Angerer, P. & Theis, F. SCANPY: large-scale single-cell gene expression data analysis. Genome Biol 19, 15 (2018). https://doi.org/10.1186/s13059-017-1382-0
6. anndata: Annotated data
   Isaac Virshup, Sergei Rybakov, Fabian J. Theis, Philipp Angerer, F. Alexander Wolf
   bioRxiv 2021 Dec 19. doi: 10.1101/2021.12.16.473007.
7. scikit-learn package: https://scikit-learn.org/stable/index.html
   Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
8. SHAP (SHapley Additive exPlanations)
   Lundberg, S., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. https://doi.org/10.48550/arXiv.1705.07874
9. Ensembl Gene ID: https://www.wikidata.org/wiki/Property:P594