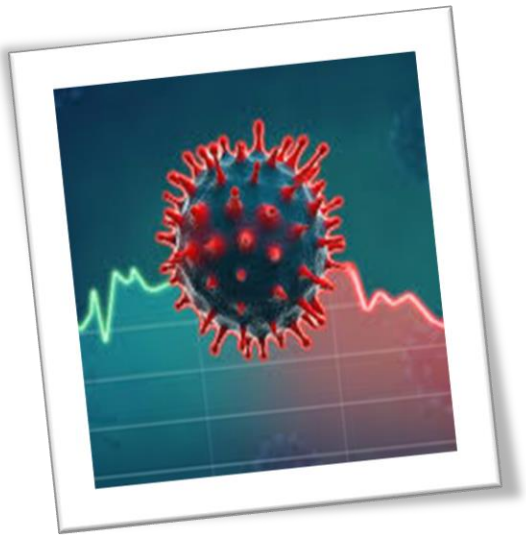
Covid 19 Diagnosis



1. Executive summary
    1. Motivation
    2. Objective
    3. Road map
    4. Insights and Recommendations
2. Dataset
3. Metrics
4. Preprocessing
5. Exploratory Data Analysis
6. Building Machine learning models
7. Results
8. Conclusion

1. Executive summary

Motivation:

In pandemics and overwhelmed health system, the possible of limitation to perform tests to detect SARS-CoV-2 and test every case would be impractical. Tests results could be delayed even if small sample of population would be tested.

Objective:

Predicting the chances of being positive or negative for covid19 and identify the factors that influence it. Provide the recommendations to the hospital on how they can better manage the admission of patients to the general ward, semi-intensive unit, or intensive care unit.

Plan to solve the problem:

I will follow the following step:

1. Understanding the dataset, cleaning, prepare for process.
2. Divided the data into train data and testing data
3. Building classification machine learning models: logistic Regression, Random Forest, Gradient Boosting, and XGBoost
4. Applied all the models on training data, then test it on test data to find the best model.
5. Find the most important features that have the highest impact of the prediction.

Insights and Recommendations:

97% of patients were not admitted in hospital, while 0.7% were admitted in intensive care.
45% of patients accepted to the Regular ward have a positive Covid19, while 18% of patients accepted to the semi-intensive care unit have a positive Covid 19, and 20% of patients accepted to the intensive care unit have a positive Covid 19.
There are some other viruses could have similarity in symptoms with Covid19 such as Influenza B, Rhinovirus or Corona virus63.

Patient age quantiles between 9 and 19 has higher positive covid19 cases than rest.

Respiratory test are important factors in predicting the covid19. Base access (Vinous gas blood analysis) is a major variable in our study.

Blood test is essential to track the infections and they are indicators of covid19. E.g., Platelets that shows values less than average for positive covid19, while for red_blood_cells test, the values were higher than average. We recommend investing in respiratory and blood tests for patients coming with symptoms because that’s the key to track positive covid cases.

2. Dataset

How the data collected?

This dataset contains anonymized data from patients seen at the Hospital Israelita Albert Einstein, in São Paulo, Brazil, and who had samples collected to perform the SARS-CoV-2 RT-PCR and additional laboratory tests during a visit to the hospital. All clinical data were standardized to have a mean of zero and a unit standard deviation. The variables in the dataset are all self-explanatory.

The Dataset Description:

File name	covid19_dataset
Base format of the file	Xlsx
Total number of files	1
Total number of observations	5644
Total number of features	111
Dtype: Float64 Dtype: Object Dtype: Integer	70 37 4
Size of the data	485kB

3. Metrics:

Model can make wrong predictions as:
1. Model predicts that a patient will test positive but in reality the patient tests negative.

2. Model predicts that a patient will test negative but in reality the patient tests positive.
Which case is more important?
In health domain reducing the false negative is more important. If a patient tests negative when it had to be positive a patient will go back to family and community to spread the virus.
How to reduce the losses?
`Recall Score` can be used as the metric for evaluation of the model, greater the Recall score are the chances of minimizing False Negatives.

4. Preprocessing

Rename the features:

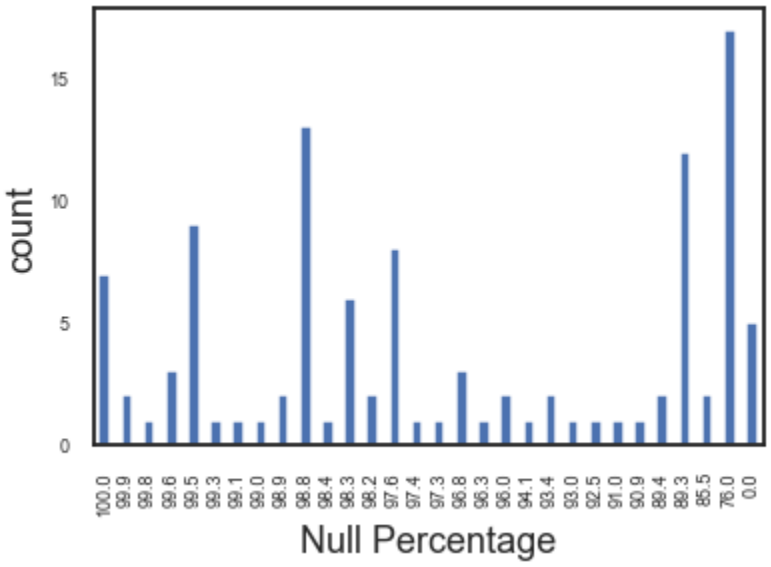
The feature names look inconsistent and wordy. We applied a function to clean up the white spaces, remove extra symbols, rename, or shorten the feature names.

Finding the classes for each feature and labeling with numeric values.

Drop meaningless features such as the 'Patient Id'. It is unique number without any duplication.

Missing values:

We calculate the percentage of the nan values in each column. We plot the results.



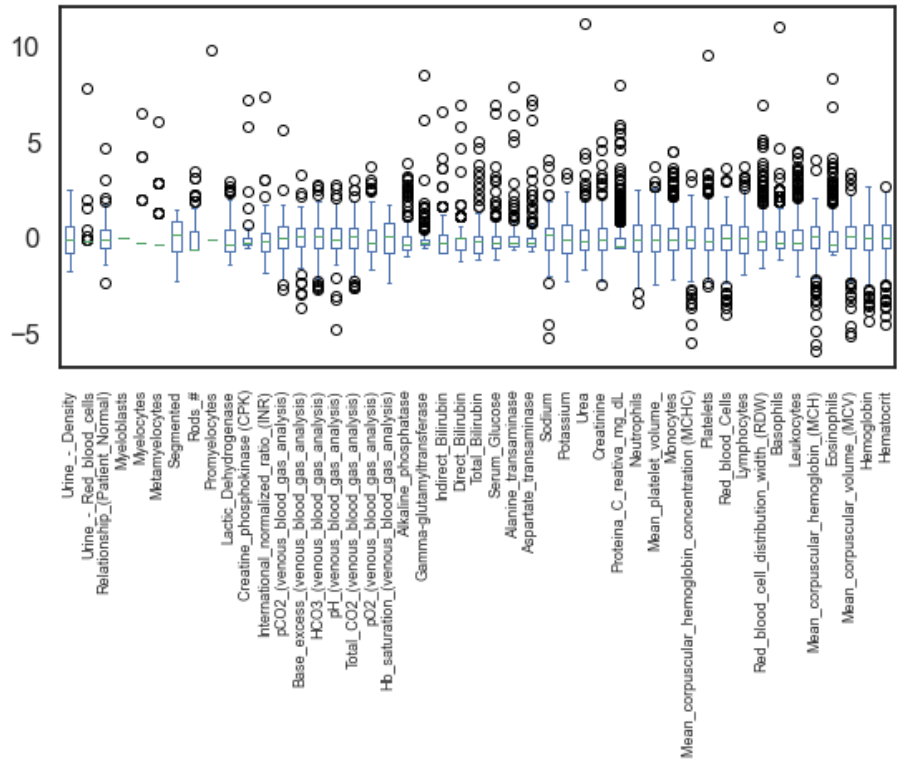
We choose to select features with less than 99% nan values because we trying to include as much possible of lab test results. The total number of the feature at this step is 85.

Fill Null values:

From the last figure, the missing values were grouped in multiple group based on the lab tests. Therefore, to fill NaN values, we applied KNN Imputer because it will consider the relationship between features to fill the empty cells. First, we separated the numerical and categorical features then we applied KNN for each group. For categorical we chose 'most frequent' and for numerical we choose 'mean' the k values is 7. We average 7 neighbors' values to predict the missing values.

Outliers:

There are no outliers in this dataset, we accepted all the values.



Feature engineering

Patient Age Quantile column was converted to categorical feature with 4 categories. The reason is to reduce the dummy variable later in the process.

Melting the patient admission in hospital into one column called 'Admission'. The reason for this feature is to make EDA analysis.

Feature Selection:

Drop Low Variance features.

Collinearity:

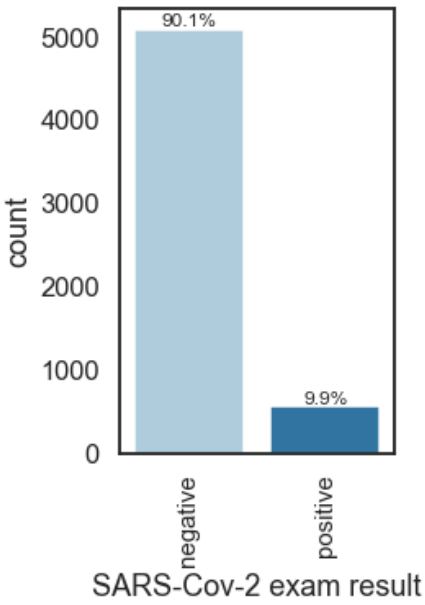
To solve the collinearity problem. First, Split the data into predictors data frame and target data series using pandas. Then, we apply the Pearson correlation on the numerical features of the predictors and drop the features that have a high positive or negative correlation value >0.7. In this way, we be sure that all the features are independent.

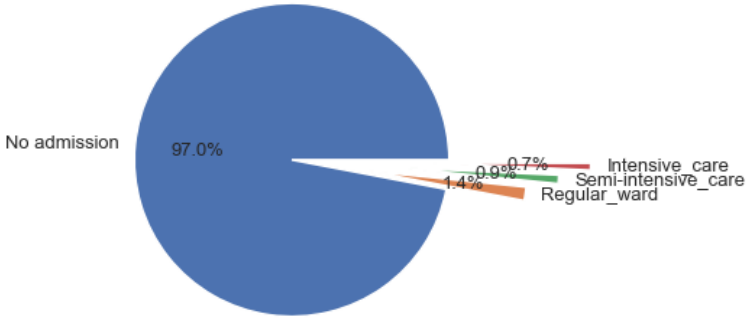
Ch2 contangoes tests whether two features are independent or depended on each other. By applying selectKBest from sklearn and choosing score function ='chi2' on each of the categorical feature in the predictors and the target, the feature depend on output will be chosen.

The total number of features after this step is 55 and the row is 5644.

5. Exploratory Data Analysis

Univariant Analysis:

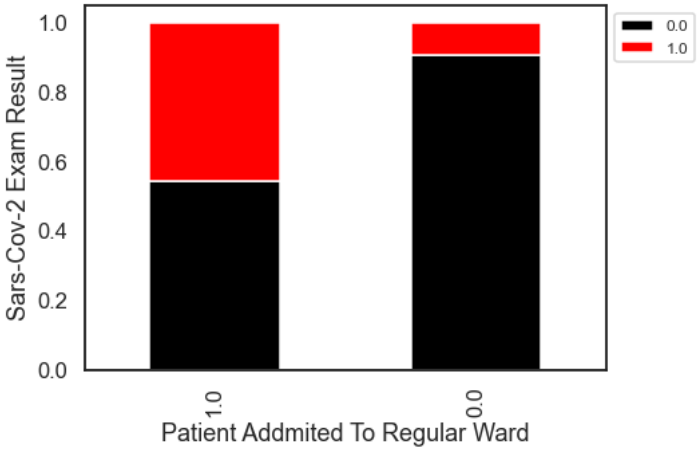


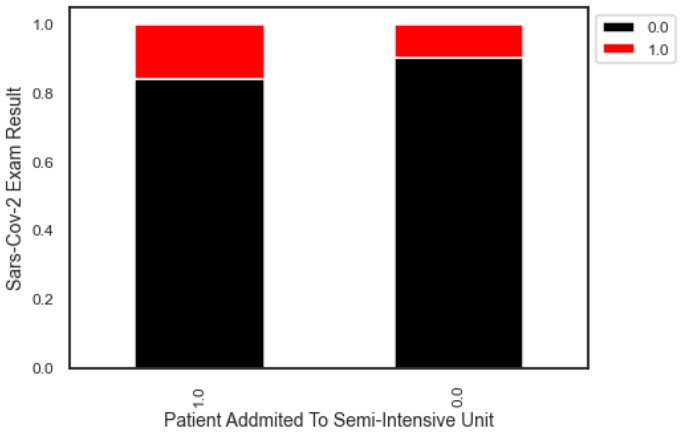


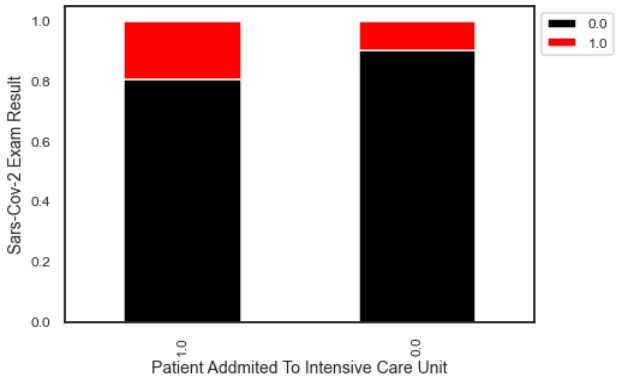
- The Cov12 exam result is imbalanced feature, 90% negative and 10% positive. The ratio is 558/5086=0.1
- 97% of patients did not admit in hospital, while 0.7% were admitted in intensive care.

Bivariant Analysis:

Covid Test variation with Patient Admission to Care Unit

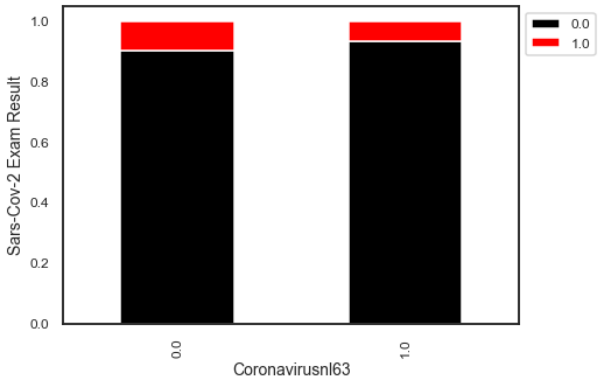


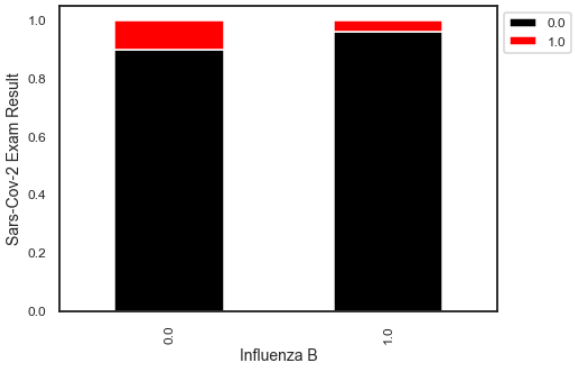


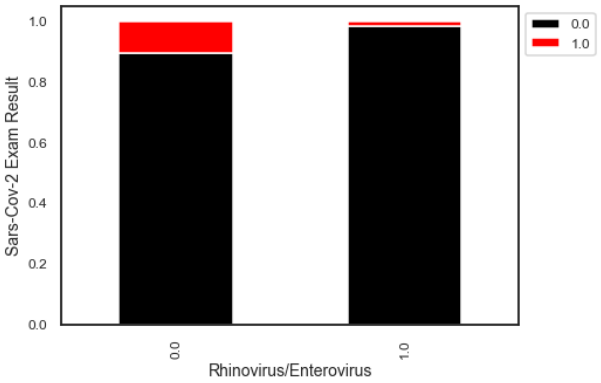


- 45% of patients accepted to the Regular ward have a positive Covid 19.
- 18% of patients accepted to the semi-intensive care unit have a positive Covid 19.
- 20% of patients accepted to the intensive care unit have a positive Covid 19

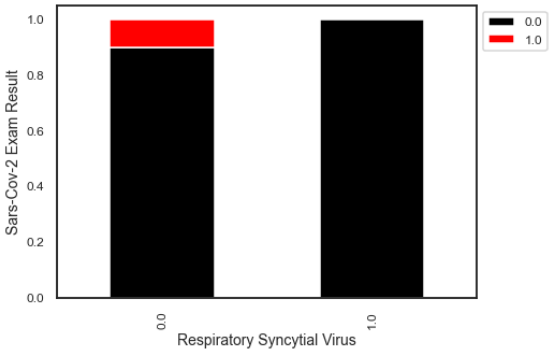
Covid 19 and other viruses

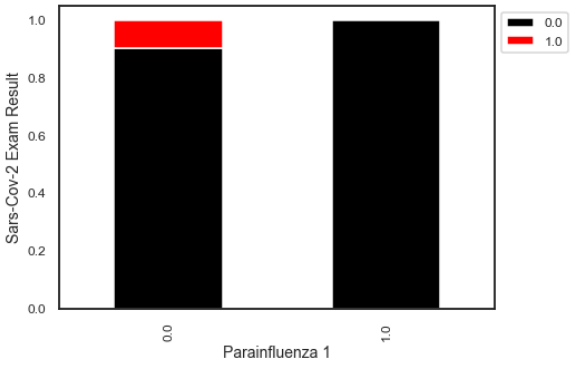


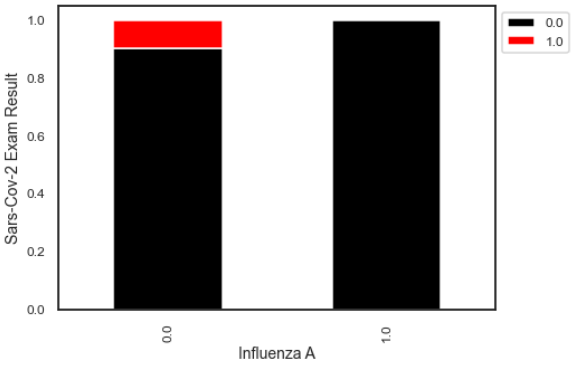




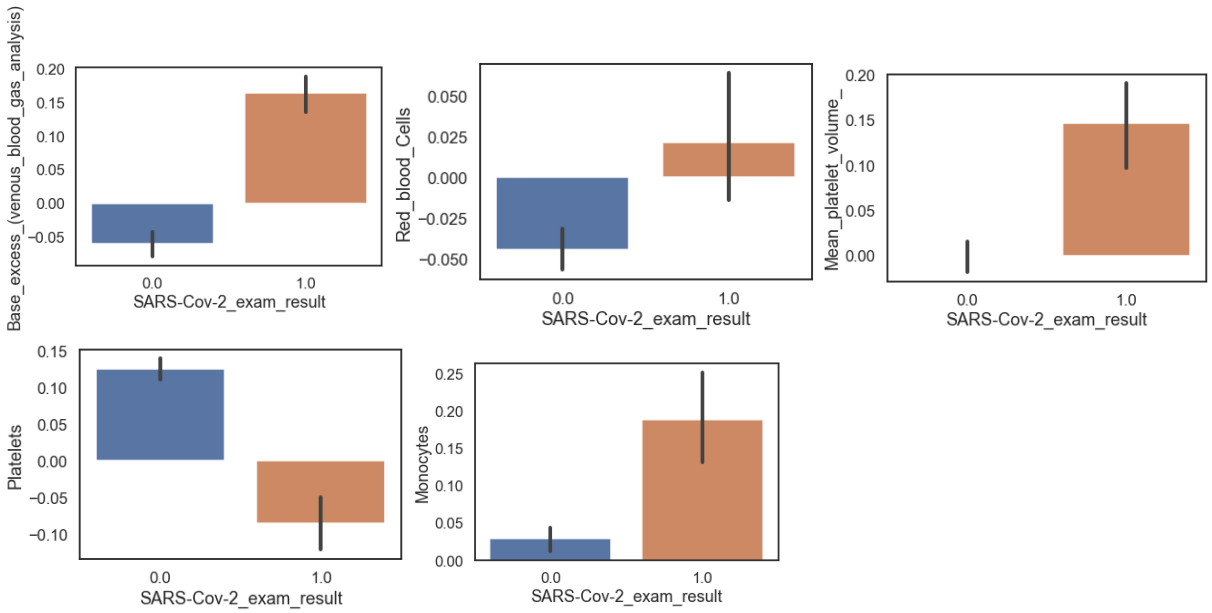
- Some virus's symptoms could match with Covid 19 symptoms:
- Cases were approved to Influenza B, Rhinovirus or Corona virus 63 have 2% ,1%, and 8% positive covid 19, respectively. While cases have not been approved showed 15% positive covid19.



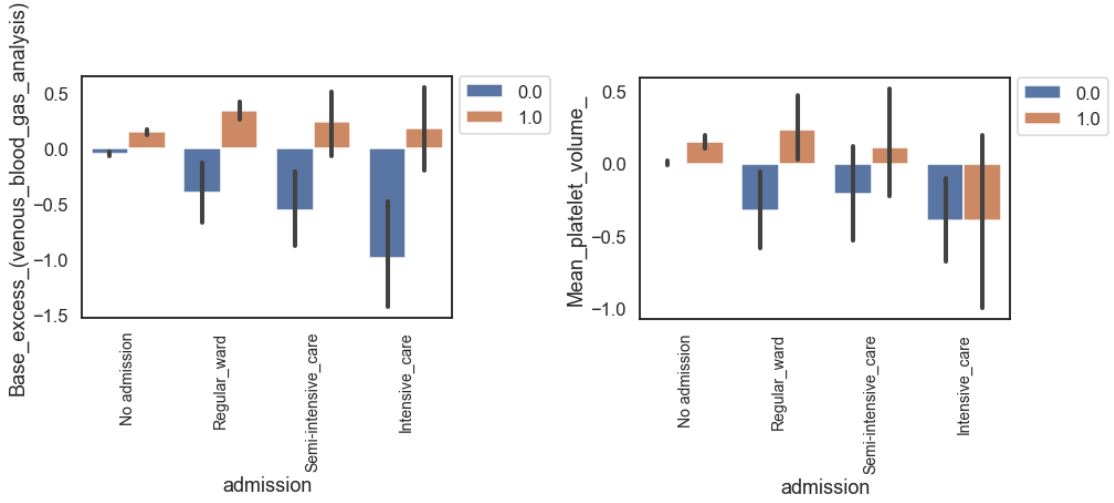




- When Influenza A, Parainfluenza 1, and Respiratory syncytial virus haven't been detected, 10% of the cases tested positive to the covid 19.
- Symptoms of Influenza A, Parainfluenza 1, and Respiratory syncytial virus most likely do not mix with Covid 19 symptoms.



Base excess venous blood gas, Red blood cells, Mean Platelet volume, Platelets Monocytes variables Show distinctive differences between positive and negative covid cases. Blood tests are essential indicators for infections.



in case of venous blood gas, most of the patients were accepted in the intensive care were not positive covid19. While for mean platelet volume with negative values were accepted in intensive care.

6. Building Machine learning Models

We choose to run multiple classification models: Logistic Regression, RandomForest, AdaBoost, GradientBoost, and XgBoost.

Our data is imbalanced 90% positive and 0.1 negative. Therefore, the parameter class_weight='balanced' is important to be applied to the models. Some models don't have this parameter such as AdaBoost and Gradient boosting. To overcome this limitation, we assign the initial model to be a decision tree classifier with balanced classes.

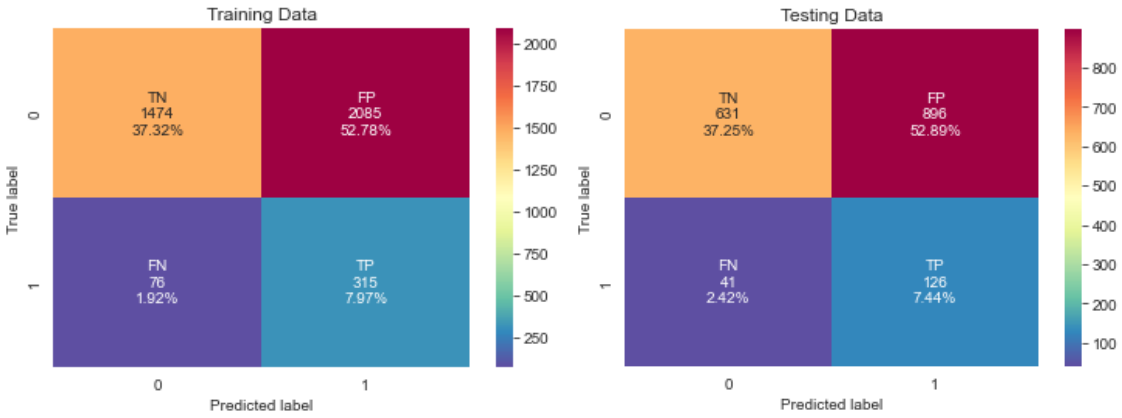
To find the Hyperparameters we applied Gridsearchcv for logistic regression and Randomforestclassifier. For gradientBoostingClassifier and xgboost we applied Randomizesearchcv because they are computationally more expensive, and the more hyperparameters to train. We set the score to recall metrics.

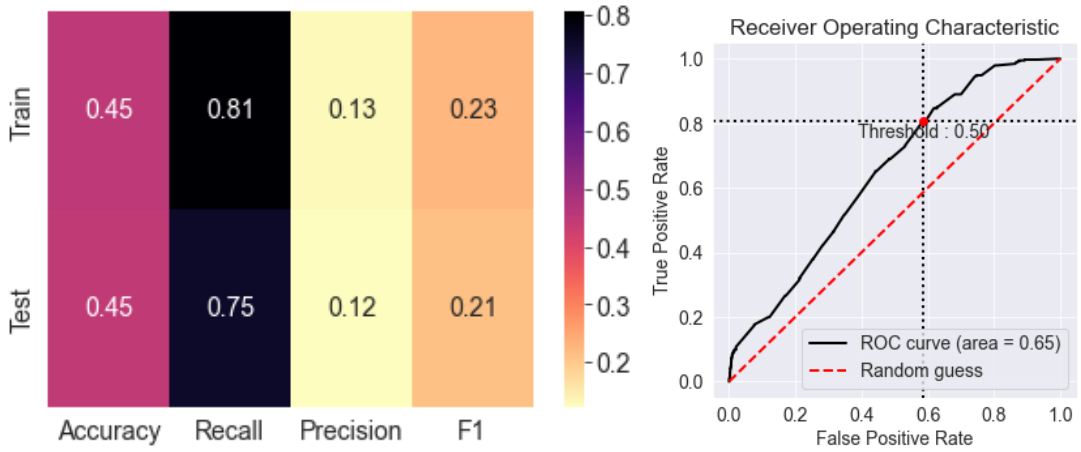
Logistic regression model:

We applied the logistic Regression model with Cross Validation technique to find the best model.

```
LogisticRegressionCV(Cs=[0.01,0.05,0.02],class_weight='balanced',penalty='l1'
,random_state=1,scoring=make_scorer(recall_score), solver='liblinear')
```

Then we fit the model on the Training data and predict the output of the test dataset. The confusion matrix of both training and test data:

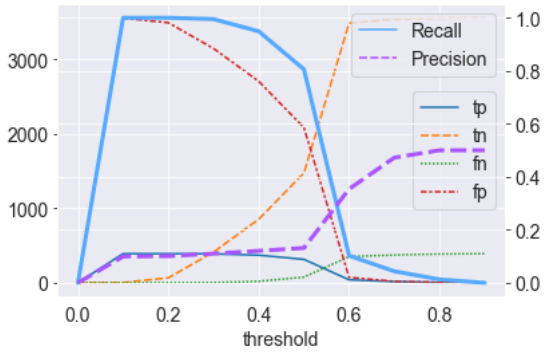




By comparing the metrics, we find the recall is overfit and the AUC is 65%.

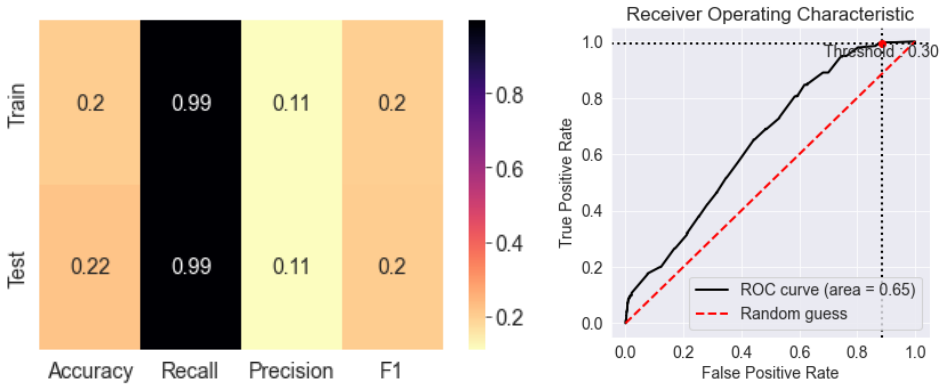
Optimizing Logistic Regression model:.

Finding the threshold that results in highest recall metrics.



This plot shows the default 0.5 can be changed to 0.3 to match the highest value of Recall.

The Recall result was improved.



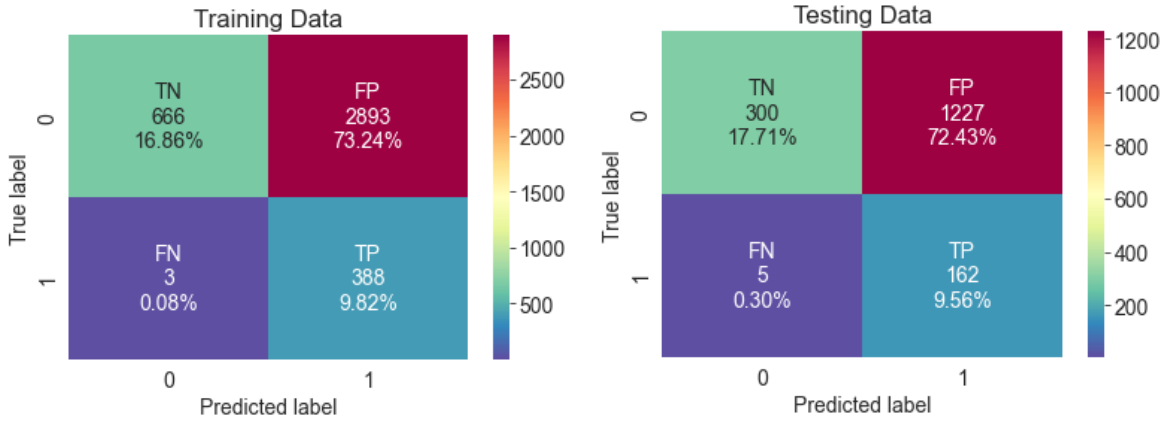
This model is accepted.

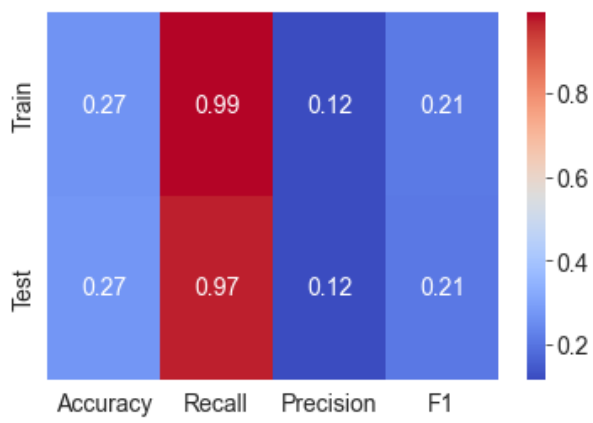
RandomForestClassifier:

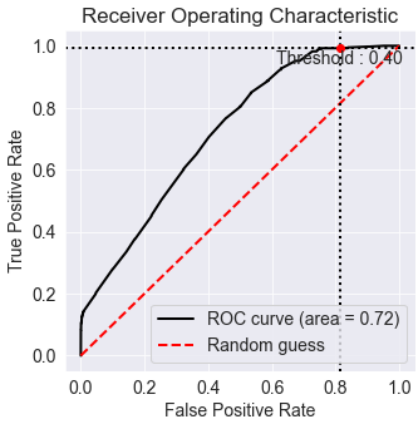
We applied Randomforestclassifier. To tunning the hyperparameters, the grid search combined with cross validation applied on lists of values. The metrics is recall. The best model was fit on training data. Then we optimize the model through choosing the threshold value that maximize Recall.

The best model resulted from the gridsearch:RandomForestClassifier(class_weight='balanced', max_depth=3,min_samples_split=5, n_estimators=40, oob_score=True,random_state=1)

Then we fit the model on the Training data and predict the output of the test dataset. The confusion matrix of both training and test data.







By comparing the metrics, the model is accepted.

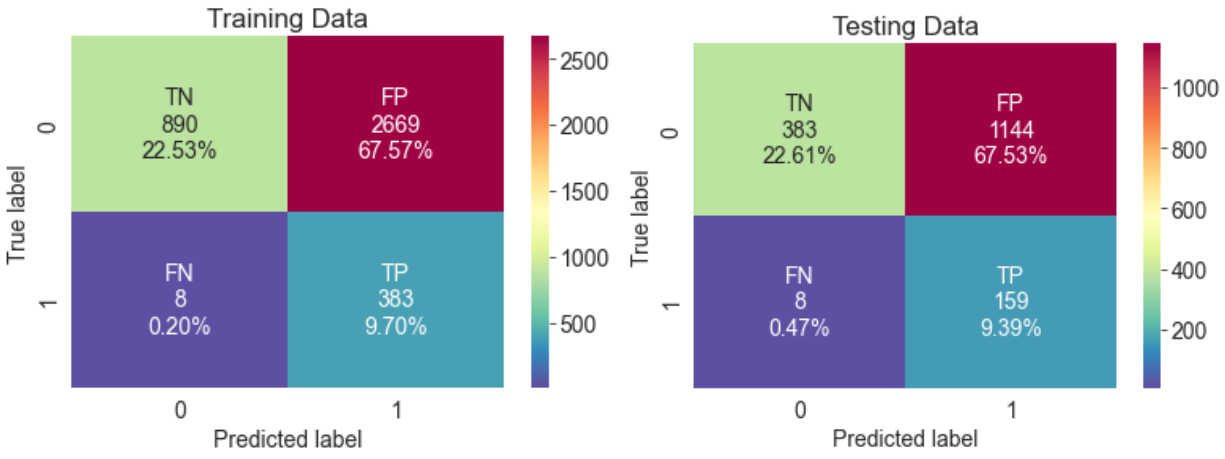
AdaBoostClassifier:

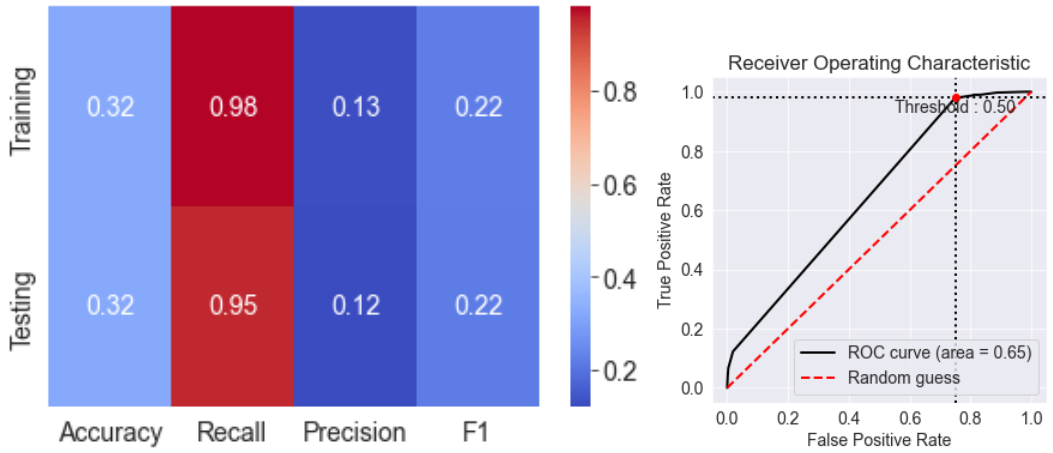
We applied AdaBoostClassifier. To tunning the hyperparameters, the grid search combined with cross validation applied on lists of values. The metrics is recall. The best model was fit on training data. We choose the initial model is Decision Tree because we can balance the imbalanced data by adding class_weight='Balanced'. Then we optimize the model through choosing the threshold value that maximize Recall.

The best model resulted from the gridsearchcv:

AdaBoostClassifier(base_estimator=DecisionTreeClassifier(class_weight='balanced', max_depth=1)learning_rate=0.01, n_estimators=100, random_state=1)
base_estimator:
DecisionTreeClassifier(DecisionTreeClassifier(class_weight='balanced', max_depth=1)

Then we fit the model on the Training data and predict the output of the test dataset. The confusion matrix of both training and test data.





This model is overfit.

GradientBoostingClassifier:

We applied GradientBoostClassifier. To tunning the hyperparameters, the Randomize search combined with cross validation applied on lists of values. The metrics is recall. The best model was fit on training data. We choose the initial model is Decision Tree because we can balance the imbalanced data by adding class_weight='Balanced'. Then we optimize the model through choosing the threshold value that maximize Recall.

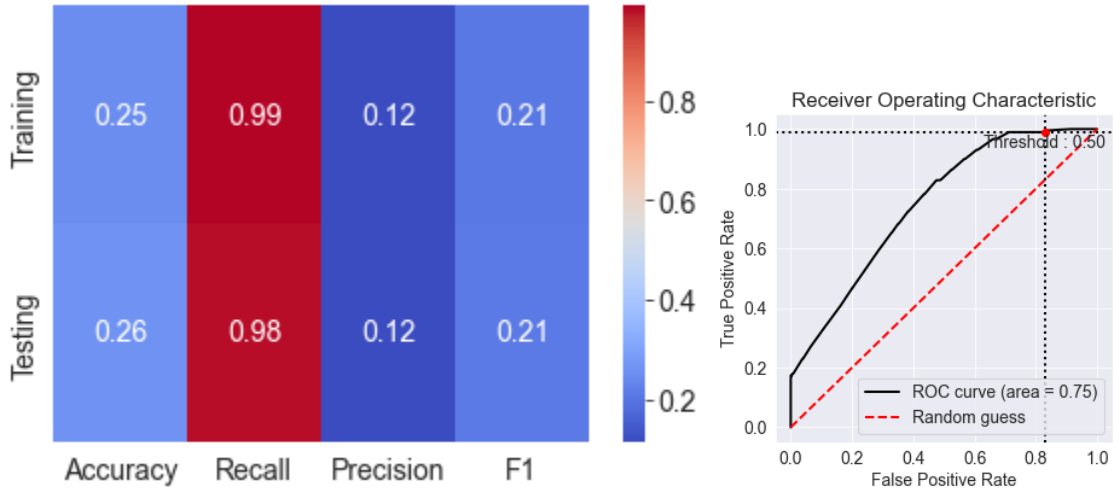
The best model resulted from the RandomizeSearchcv:

```
GradientBoostingClassifier(init=DecisionTreeClassifier(class_weight='balanced', max_depth=1,random_state=1),learning_rate=0.001, max_depth=15, max_features=0.9,n_estimators=50, subsample=0.8)
```

init: DecisionTreeClassifier

```
DecisionTreeClassifier(class_weight='balanced', max_depth=1, random_state=1)
```

Then we fit the model on the Training data and predict the output of the test dataset. The confusion matrix of both training and test data:



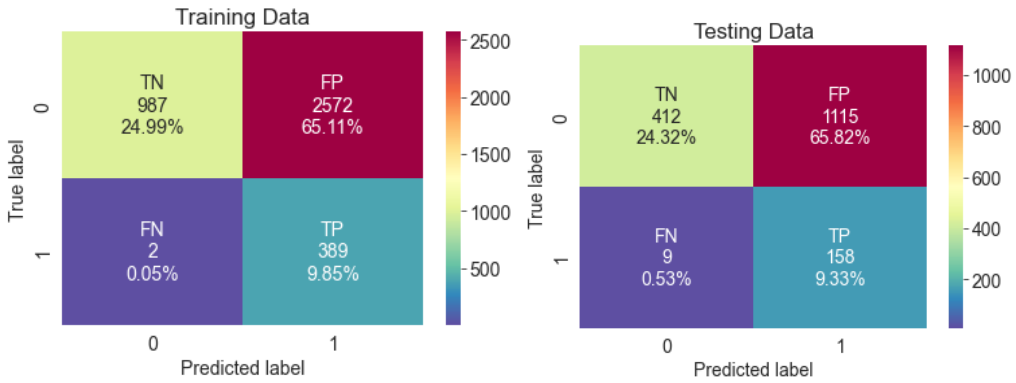
This model is accepted.

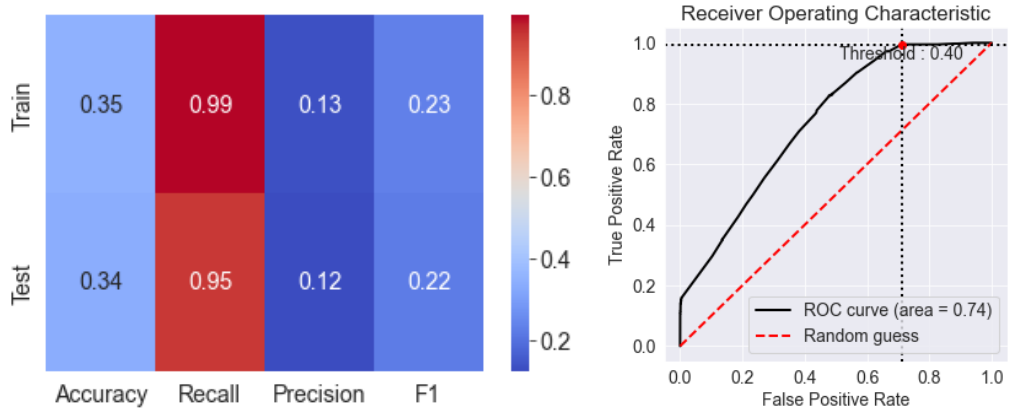
XGBClassifier:

We applied XGBClassifier. To tunning the hyperparameters, the Randomize search combined with cross validation was applied on lists of values. The metrics is recall. The best model was fit on training data and test on test dataset. We balanced the data using 'scale_pos_weight=9'.9 is count of zeros to count of ones of the target. Then we optimize the model through choosing the threshold value that maximize Recall.

```
The best model resulted from the gridsearchcv:
XGBClassifier(base_score=0.5, booster='gbtree',
callbacks=None,colsample_bylevel=1, colsample_bynode=1,
colsample_bytree=0.9,early_stopping_rounds=None,
enable_categorical=False,eval_metric='logloss', gamma=3, gpu_id=-
1,grow_policy='depthwise', importance_type=None,interaction_constraints='',
learning_rate=0.01, max_bin=256,max_cat_to_onehot=4, max_delta_step=0,
max_depth=6, max_leaves=0,min_child_weight=1, missing=nan,
monotone_constraints='()',n_estimators=150, n_jobs=0, num_parallel_tree=1,
predictor='auto',random_state=1, reg_alpha=0, reg_lambda=1)
```

Then we fit the model on the training data and predict the output of the test dataset. The confusion matrix of both training and test data:





This model is overfit.

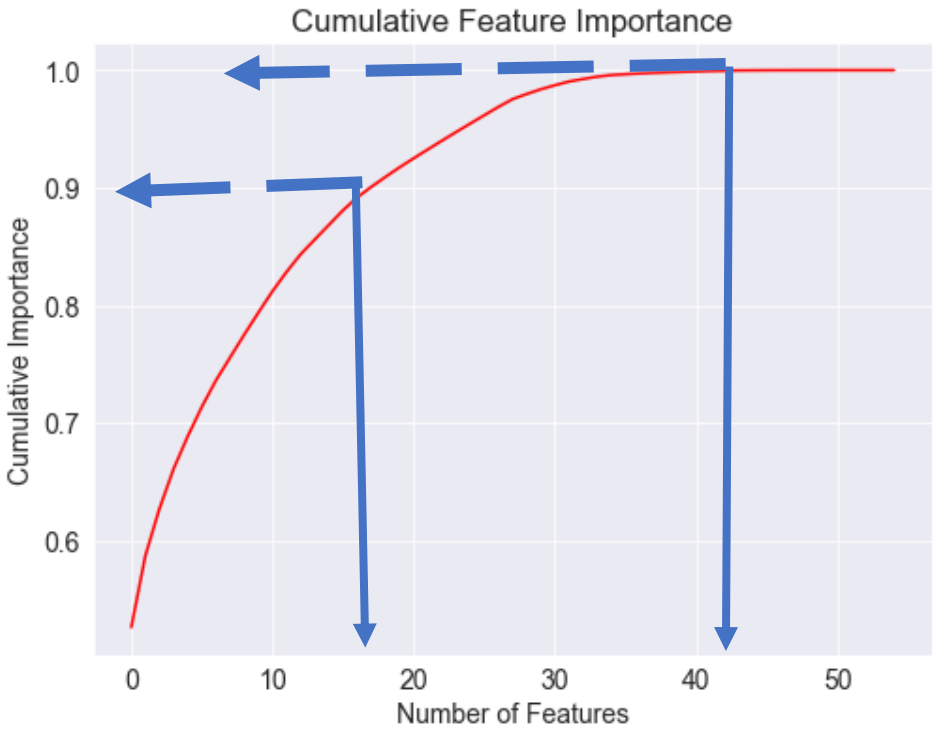
7. Result

By comparing all the previous models (Logistic regression, RandomForestClassification, AdaboostClassofier, GradientBoostingClassifier and XGBoostClassifier). We accepted three models: Logistic regression, RandomForest and GradientBooting. Based

on AUC result, GradiantBoostClassifier has the highest value (0.75). That means the positive and negative values are 75% not randomly separated.

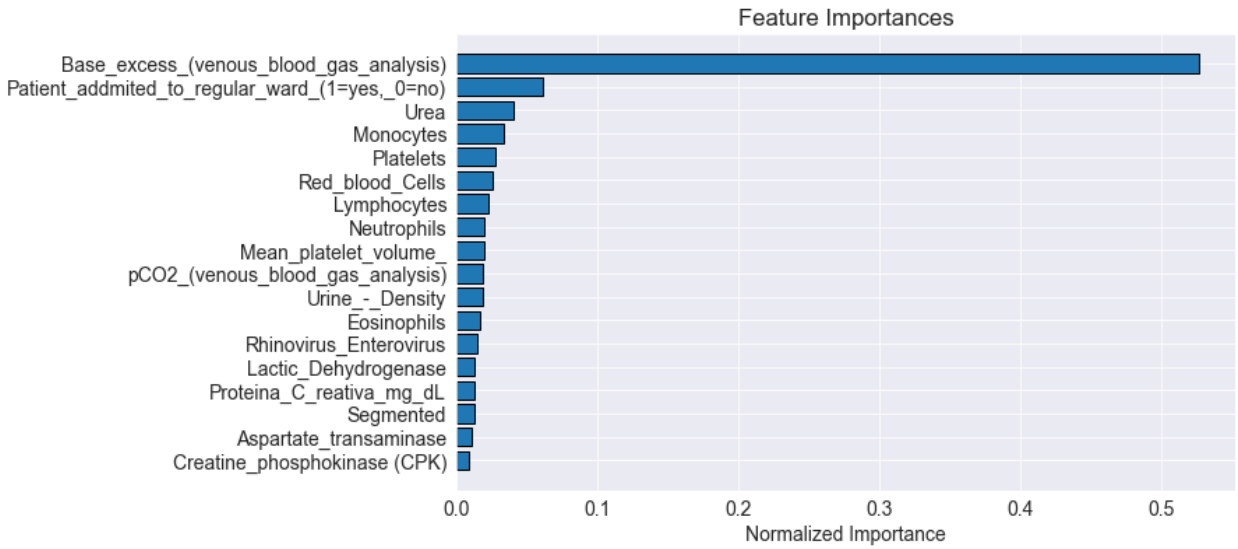
Additionally, the Recall values for the test and training models with threshold 0.5 is 99% and 98%, respectively.

Feature importance of the Gradient boosting classifier model.

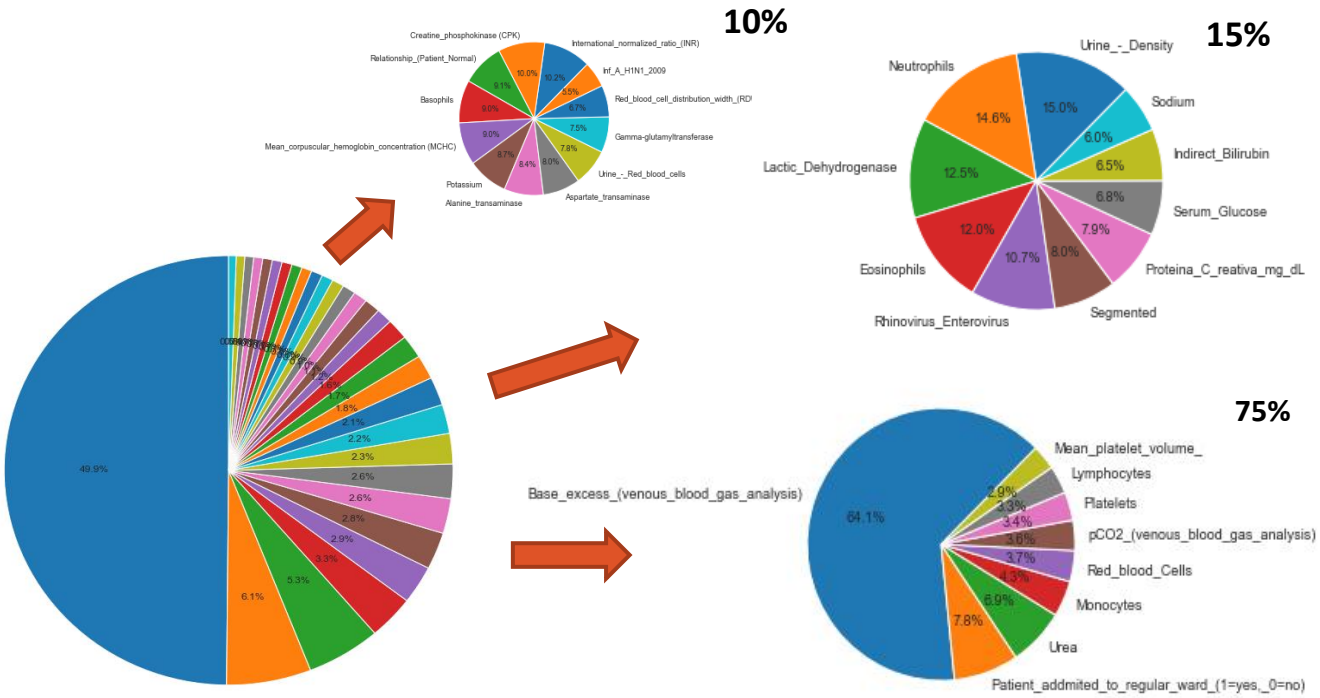


By calculating the cumulative importance of all features, we find that by choosing the first 30 features we still maintain hundred percent of the productive model. To recover 90% of the results, 18 features is enough. The most important 18 features show in the following figure.

The following plot shows the most 18 important features.



the most important feature is the venous blood gas analysis and it has



a high positive impact one the covid19 results.

8. Conclusion

All the ML models were able to effectively predict COVID-19 diagnosis. The GradienBoostingClassifier was the model with the best performance (99% and 98%) and, thus, could be used as a supporting decision tool for healthcare professionals.

We recommend adapting the following 8 tests to have primarily prediction for the covid 19: Base_excess_(venous_blood_gas_analysis), Urea, Monocytes, Platelets, Red_blood_Cells, Lymphocytes, Neutrophils, Mean_platelet_volume, pCO2_(venous_blood_gas_analysis), Urine-Density, and Eosinophil were associated with COVID-19 diagnosis and disease severity.

Accepting in the hospital mostly is related to how extreme the lab test and how intense is the symptoms. Most patients admitted in regular ward has mild symptoms with only 45% tested positive covid19 test. Patients who accepted in intensive care mostly haveing another complications besides the covid 19 or just severe illness but not covid19.

Additionally, these should be more effectively investigated in further and future works.