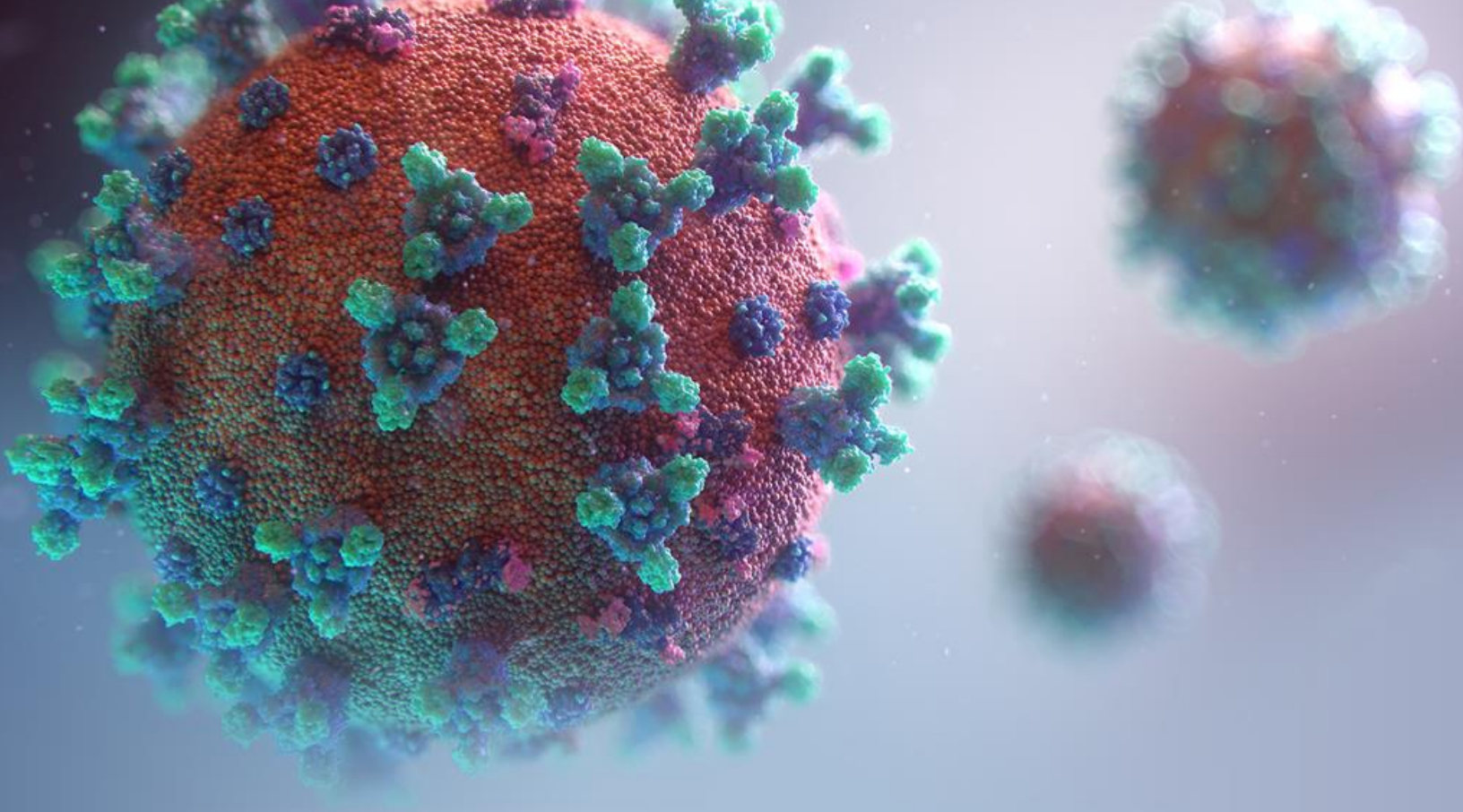


COVID-19 Project Outcome Prediction

By: Yara Elzahy
ID: 20398570

Problem Statement

Based on some pre-defined standard symptoms, the data collected in this study will help determine whether a person will recover from coronavirus symptoms or not.

1: indicates they won't recover (label: death)
0: indicates they will recover (label: recovered)

Data Exploration

- There are 13 features which are: Country, Location, Age, Gender, Visited_Wuhan, From_Wuhan, Symptoms and Time_before_symptoms_appear as shown in **Fig.1.0**
- Also, the label which indicates whether the patient died (1) or recovered (0).
- It has also been mentioned that data has already been cleaned and preprocessed; however, the data is highly imbalanced as shown in **Fig.1.1**. This could lead to having biased models.

	location	country	gender	age	vis_wuhan	from_wuhan	symptom1	symptom2	symptom3	symptom4	symptom5	symptom6	diff_sym_hos	result
673	45	13	1	47.0	0	1	6	31	19	12	3	1	1	0
602	12	12	1	49.4	0	0	14	31	19	12	3	1	0	0
436	108	24	0	59.0	0	0	14	31	19	12	3	1	0	0
663	121	30	2	49.4	0	0	14	31	19	12	3	1	0	0
433	108	24	1	30.0	0	0	14	31	19	12	3	1	0	0
582	107	2	0	60.0	0	1	14	31	19	12	3	1	0	1
454	108	24	0	34.0	0	0	14	31	19	12	3	1	0	0
26	83	8	1	29.0	1	0	14	31	19	12	3	1	0	0
684	63	13	0	56.0	0	0	6	31	19	12	3	1	0	0
595	124	7	1	45.0	0	0	14	31	19	12	3	1	0	0

Fig.1.0. A sample of the data

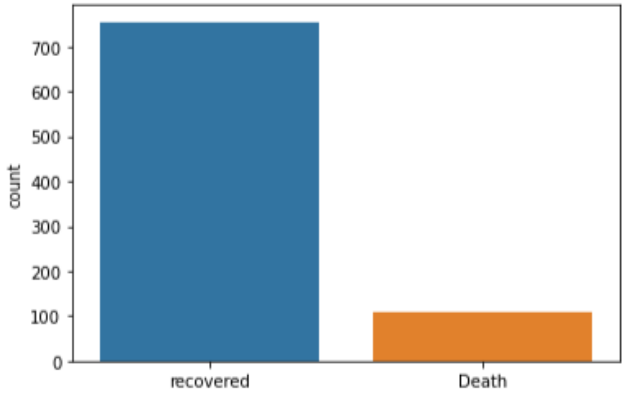


Fig.1.1. Imbalanced Data (87% recovered, 13% Death)

The figure below (**Fig.1.2.)** shows the correlations between all data correlations. As we can see that the most prominent features that have a high correlation with the result are: the age, vis_wuhan, from_wuhan, symptom1, symptom2, symptom4 and diff_sym_hos. Nevertheless, using all features produces better results than using the important features only.

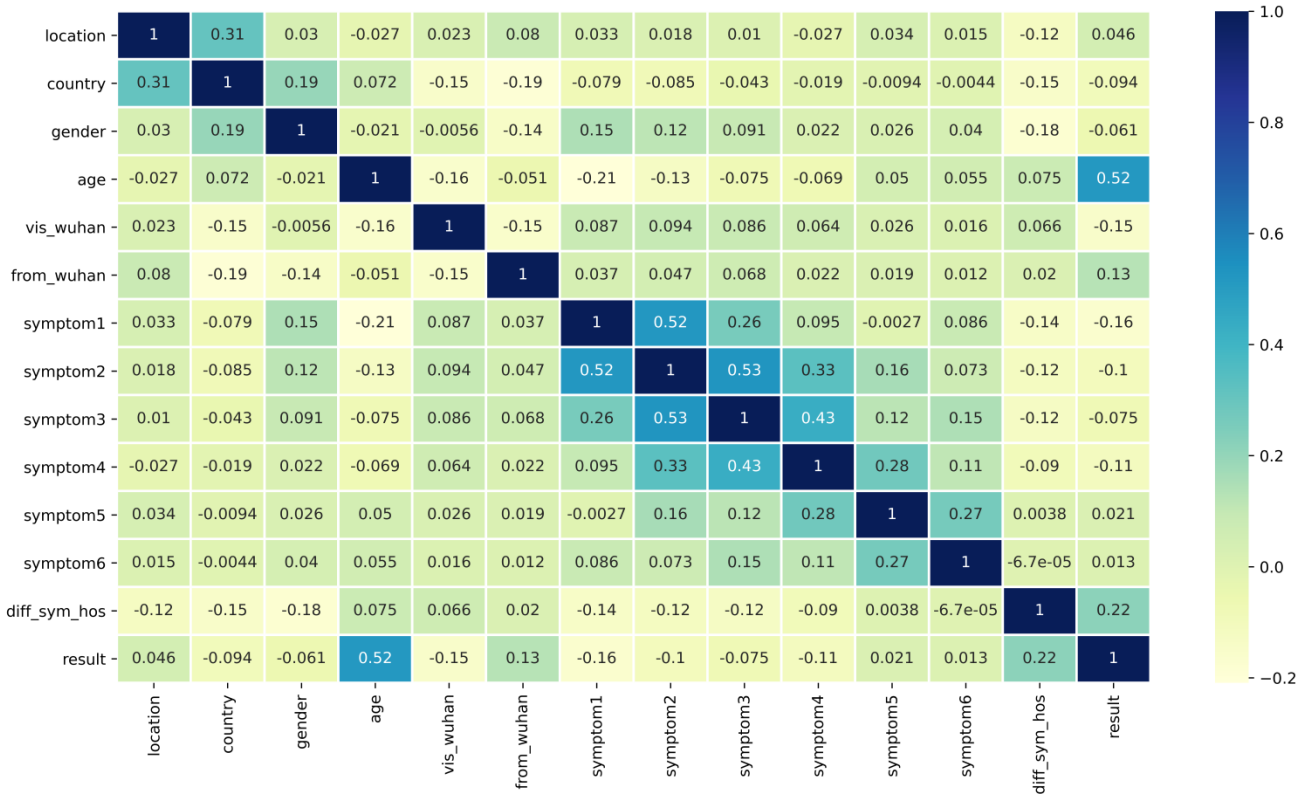


Fig.1.2. Data Correlation

Project Pipeline

Why was Grid Search Cross Validation used?

In order to find the best parameter values in a grid given a set of parameters. It is, in essence, a cross-validation approach. Both the model and the parameters must be supplied. Predictions are created after obtaining the optimal parameter values.

Function signature:

class `sklearn.model_selection.`**`GridSearchCV`**(*estimator*, *param_grid*, *, *scoring=None*, *n_jobs=None*, *refit=True*, *cv=None*, *verbose=0*, *pre_dispatch='2*n_jobs'*, *error_score=nan*, *return_trai n_score=False*

Parameters:

- **Estimator: *estimator object,*** which implements the scikit-learn estimator interface
- **param_grid: *dict or list of dictionaries,*** which contains a list of dictionaries or lists where the exploration of the grids spanned by each dictionary in the list takes place.
- **Scoring: *str callable, list, tuple or dict, default=None,*** which is the strategy used to evaluate the performance of the cross-validated model on the test set e.g. accuracy, precision, f,..etc.
- **n_jobs: *int, default=None ,*** which indicates the number of jobs that will run in parallel.
- **Cv: *int, cross-validation generator or an iterable, default=None,*** which indicates the number of folds of the cross validation.

Results:

- **best_estimator_*estimator***, is the model with the highest score
- **best_score_*float:, is the mean cross-validated score of the best_estimator_***
- **best_params_*dict***: are the set of parameters that produced the best results

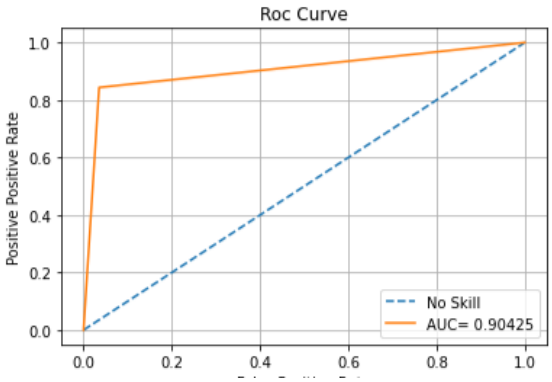
Decision Tree Classifier:

The hyperparameters of the decision tree are the max_depth , min samples leaf and the criterion (meaning: gini index or entropy).

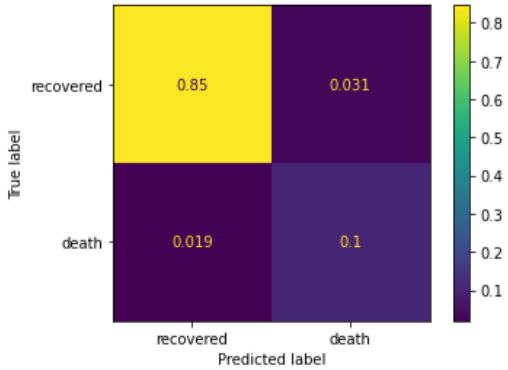
The cross-validated decision tree model results were as follows:

	Accuracy	Precision	Recall	F1 Score	AUC Score
Train	0.948675	0.784810	0.815789	0.80000	0.891796
Test	0.949807	0.771429	0.843750	0.80597	0.904254

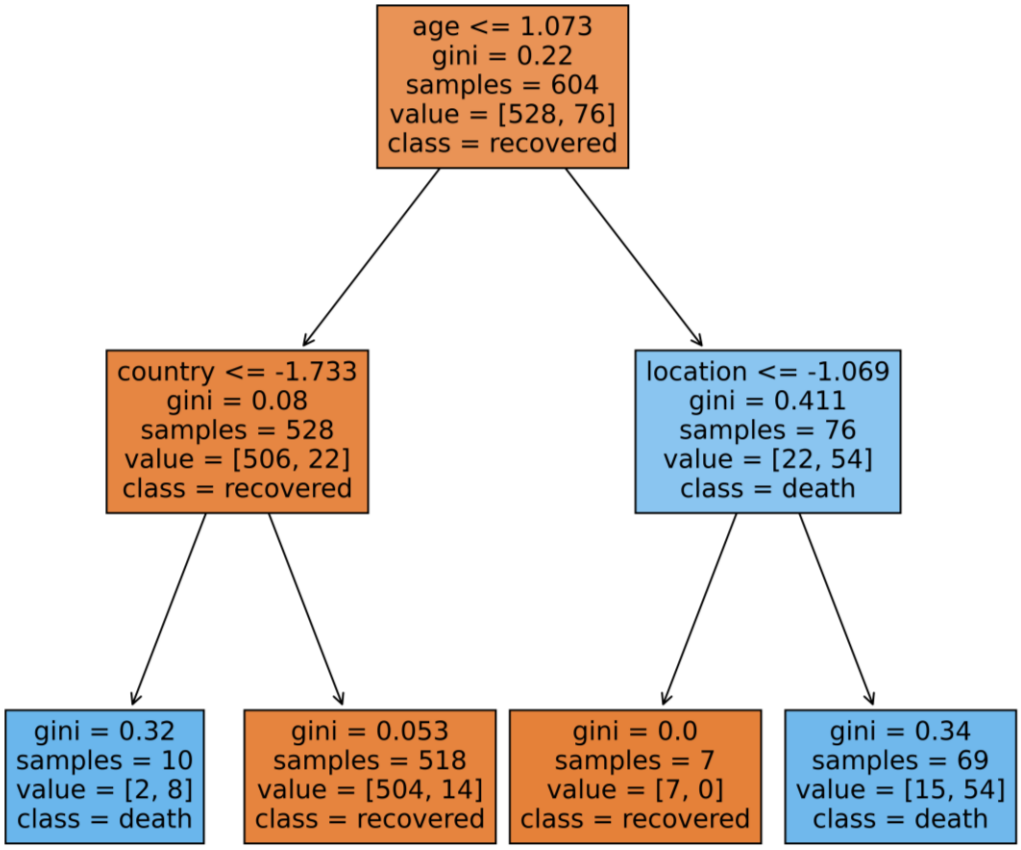
ROC Curve

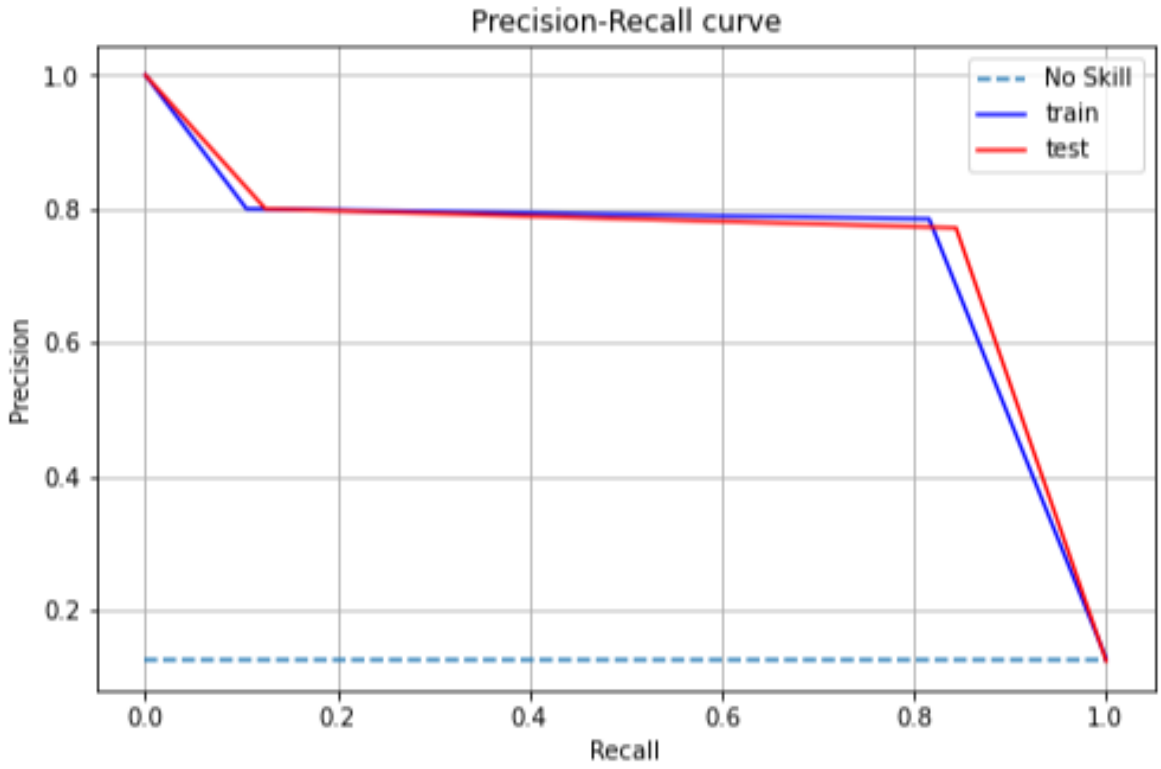


Confusion Matrix



Decision Tree





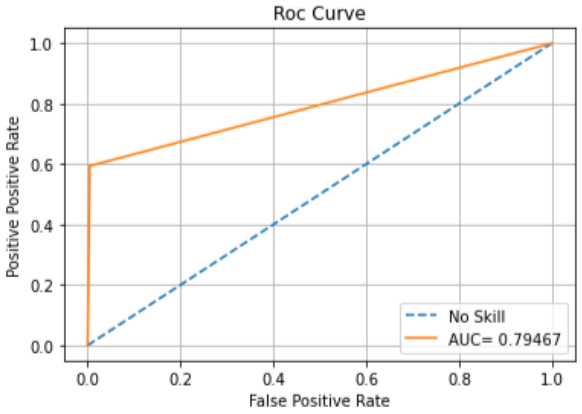
K- Nearest Neighbor Classifier:

The hyperparameters of the KNN are the k (number of neighbors) and weights (could be uniform or distance).

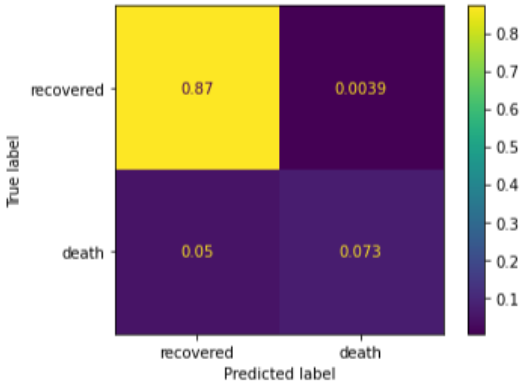
The cross-validated KNN model results were as follows:

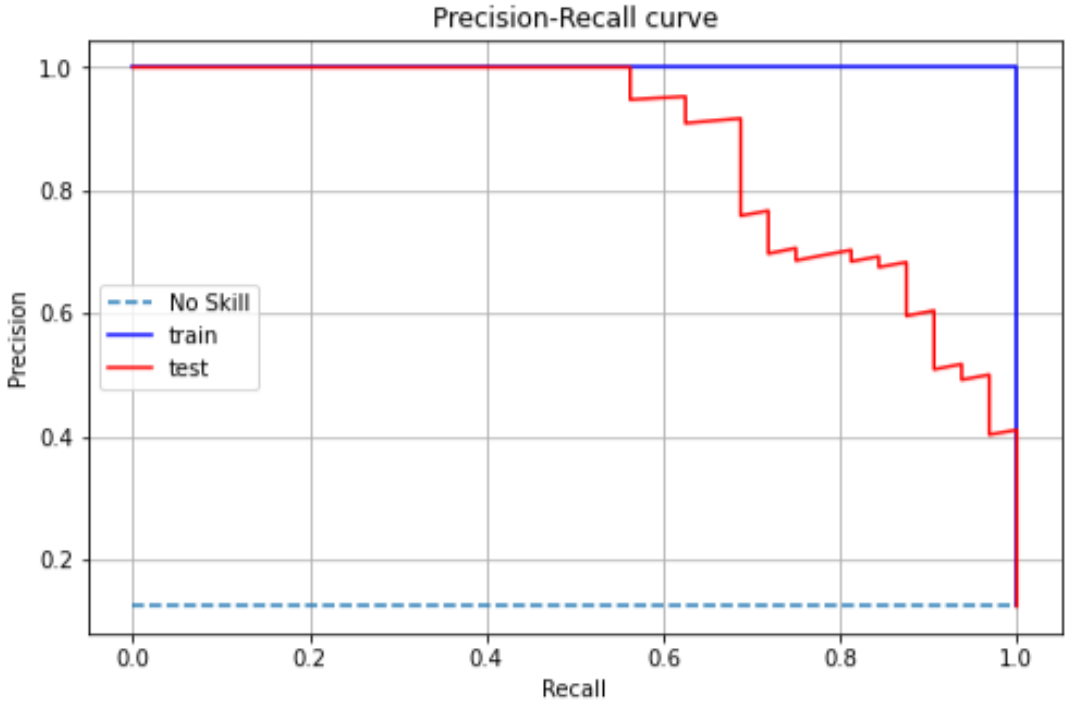
	Accuracy	Precision	Recall	F1 Score	AUC Score
Train	1.000000	1.0000000	1.0000000	1.0000000	1.0000000
Test	0.945946	0.950000	0.59375	0.730769	0.794672

ROC Curve



Confusion Matrix





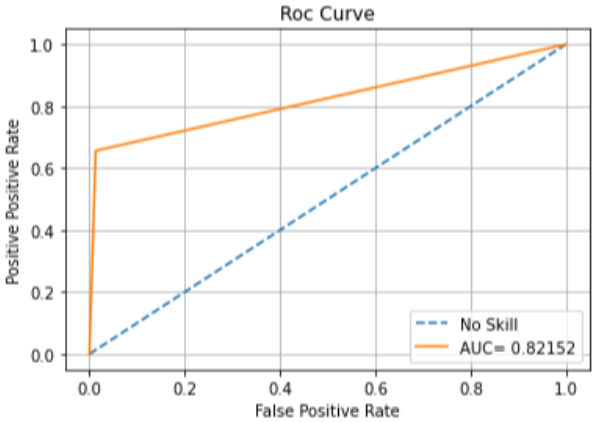
Logistic Regression Classifier:

The hyperparameters of the logistic regression are the penalty (type of regularization e.g. L1 or L2), C (the higher its value the better) and solver (could be newton-cg, LBFS or liblinear).

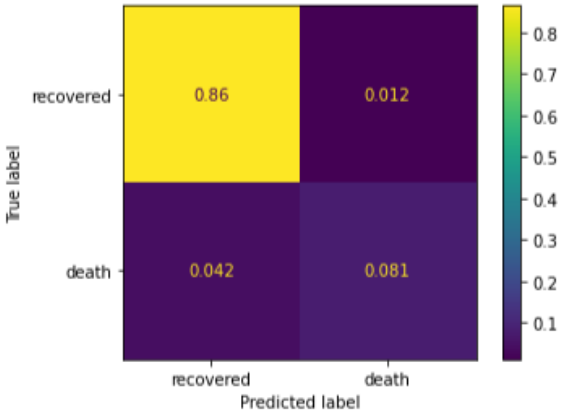
The cross-validated logistic regression model results were as follows:

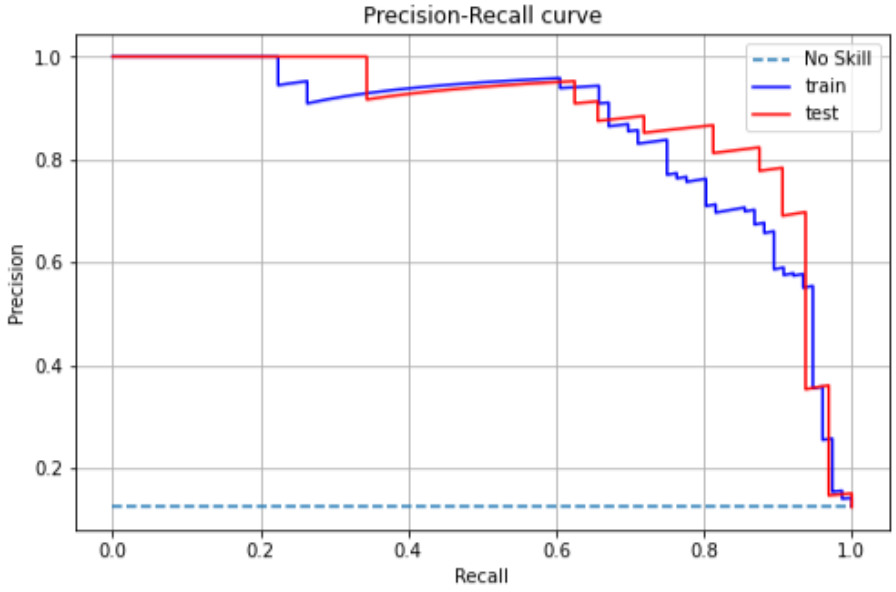
	Accuracy	Precision	Recall	F1 Score	AUC Score
Train	0.948675	0.868852	0.697368	0.773723	0.841108
Test	0.945946	0.875000	0.656250	0.750000	0.821517

ROC Curve



Confusion Matrix



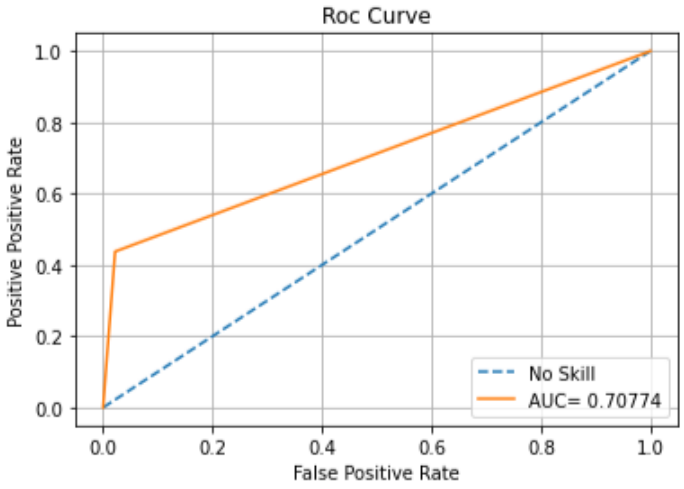


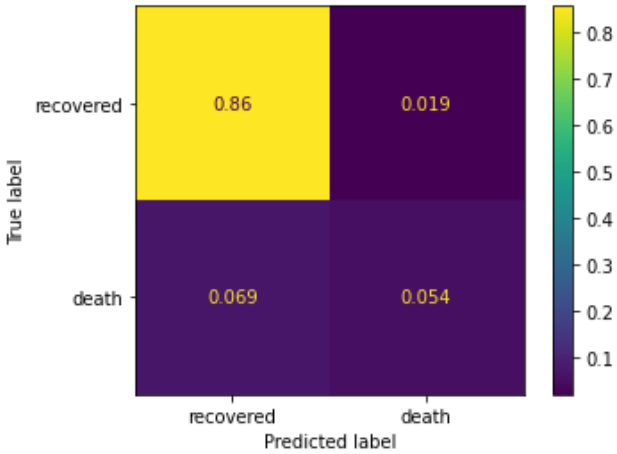
Naïve Bayes Classifier:

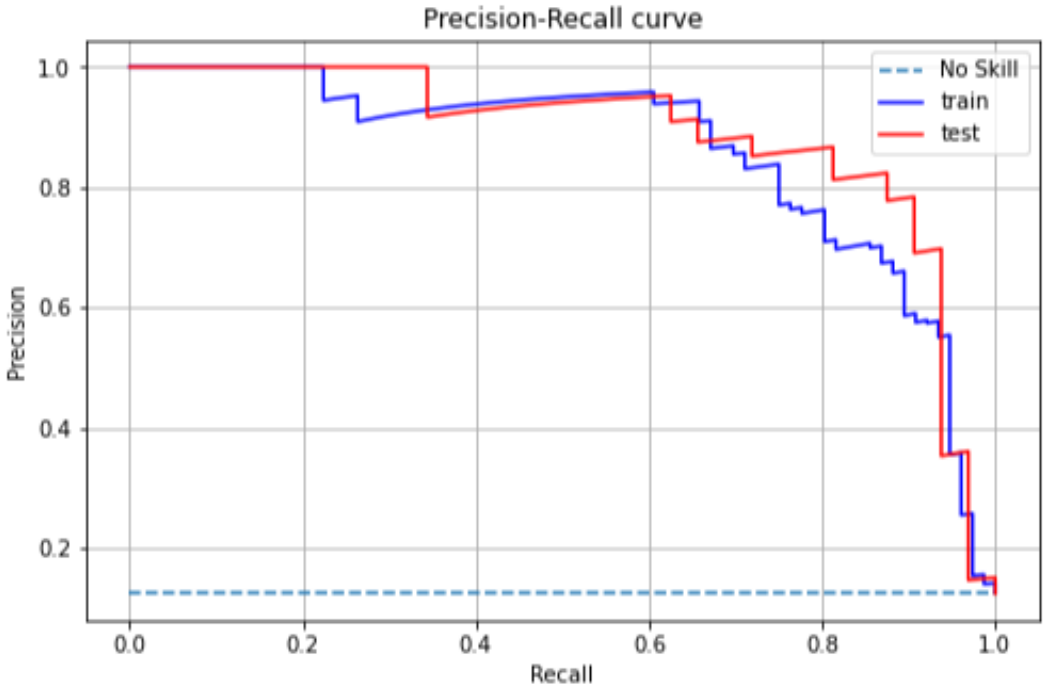
The hyperparameters of the Naïve Bayes are the var_smoothing which is the laplace smoothing.
The cross-validated Naïve Bayes model results were as follows:

	Accuracy	Precision	Recall	F1 Score	AUC Score
Train	0.887417	0.617647	0.276316	0.381818	0.625847
Test	0.911197	0.736842	0.437500	0.549020	0.707737

ROC Curve **Confusion Matrix**


Roc Curve
Positive Positive Rate
False Positive Rate
No Skill
AUC= 0.70774


recovered
death
True label
0.86
0.019
0.069
0.054
recovered
death
Predicted label

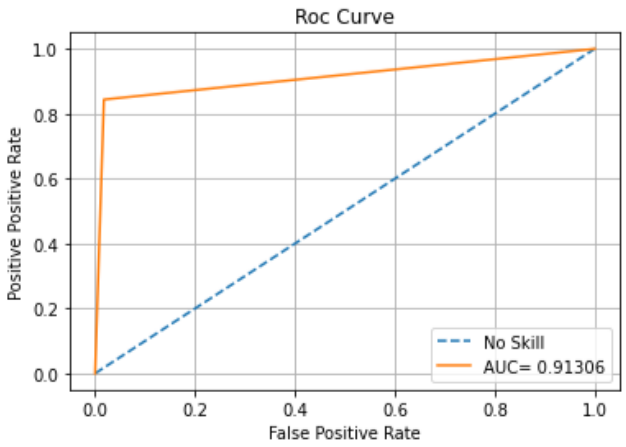

Precision-Recall curve
Precision
Recall
No Skill
train
test

Support Vector Machine Classifier:

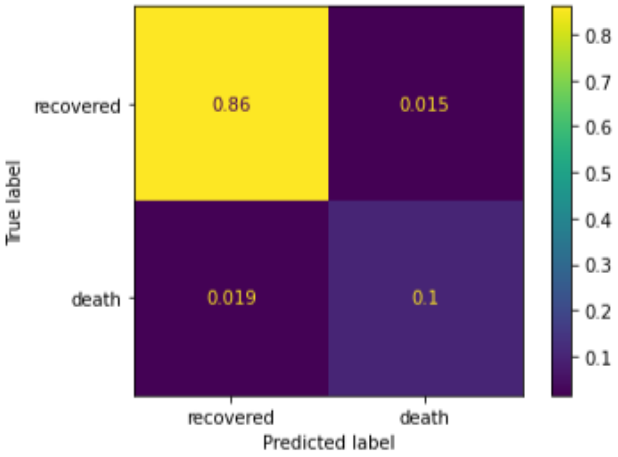
The hyperparameters of the SVM are the C, gamma and kernel.
The cross-validated Naïve Bayes model results were as follows:

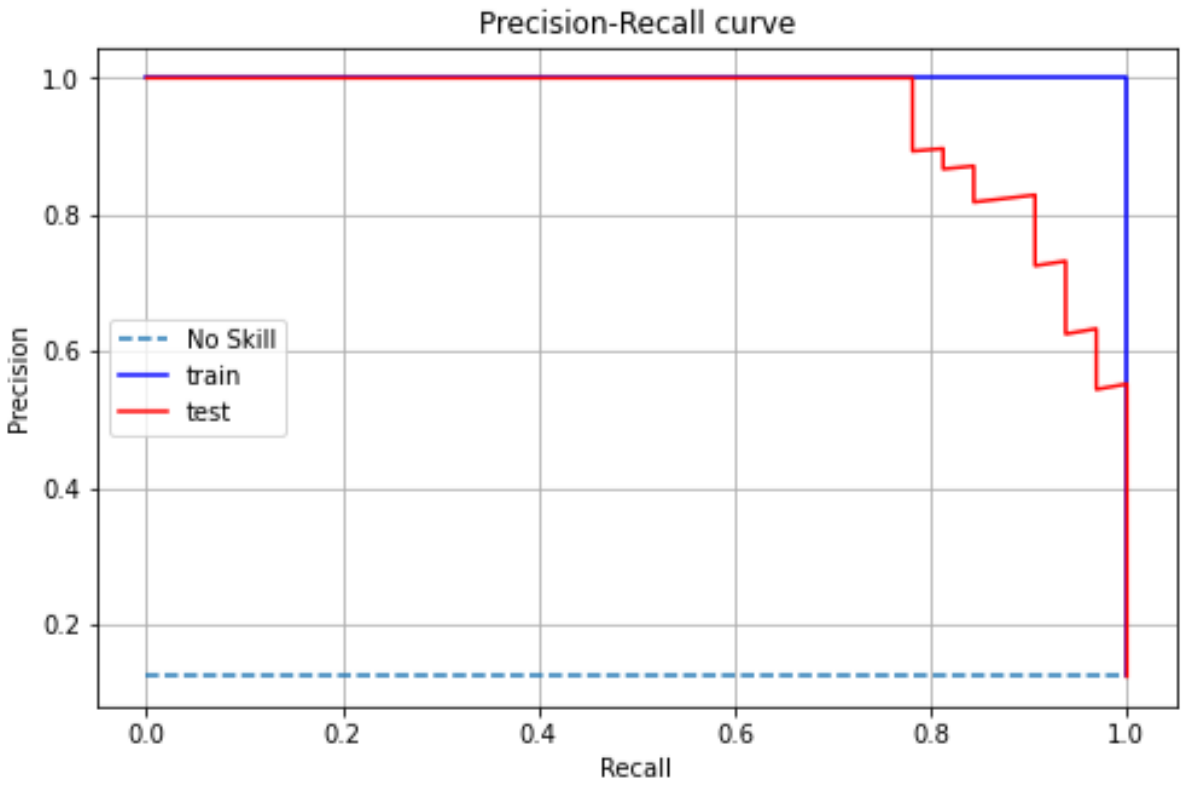
	Accuracy	Precision	Recall	F1 Score	AUC Score
Train	1.000000	1.000000	1.00000	1.000000	1.000000
Test	0.965251	0.870968	0.84375	0.857143	0.913064

ROC Curve

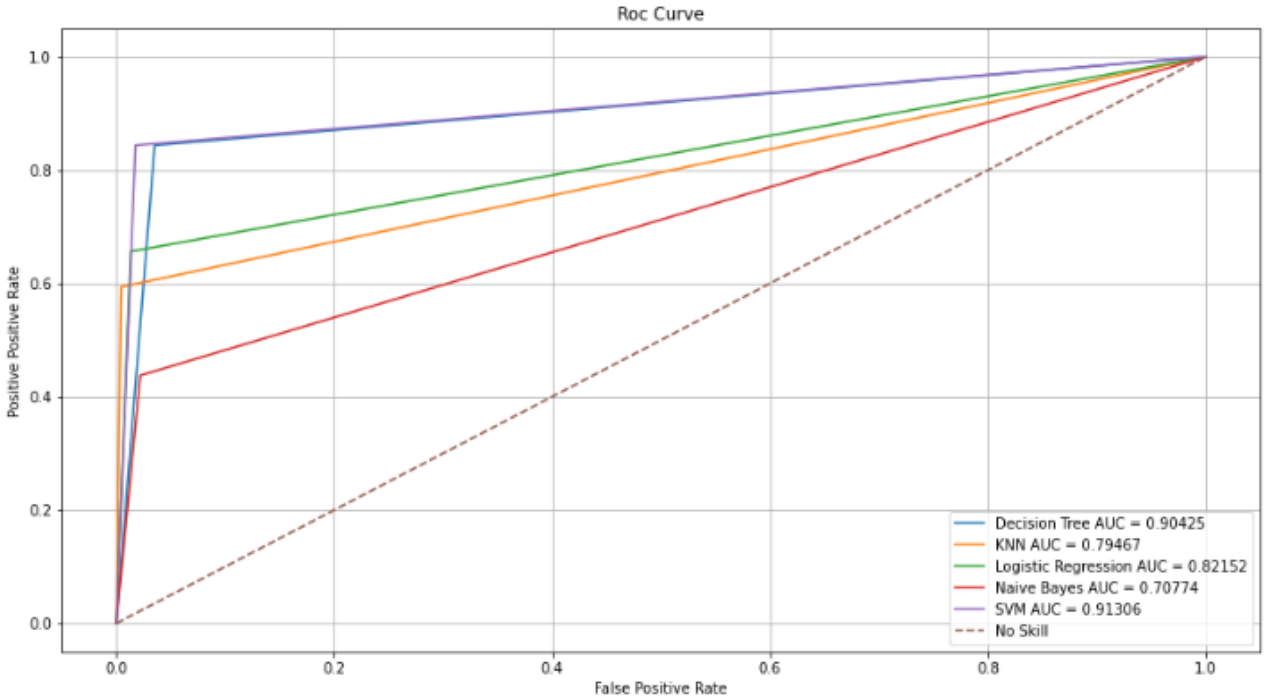


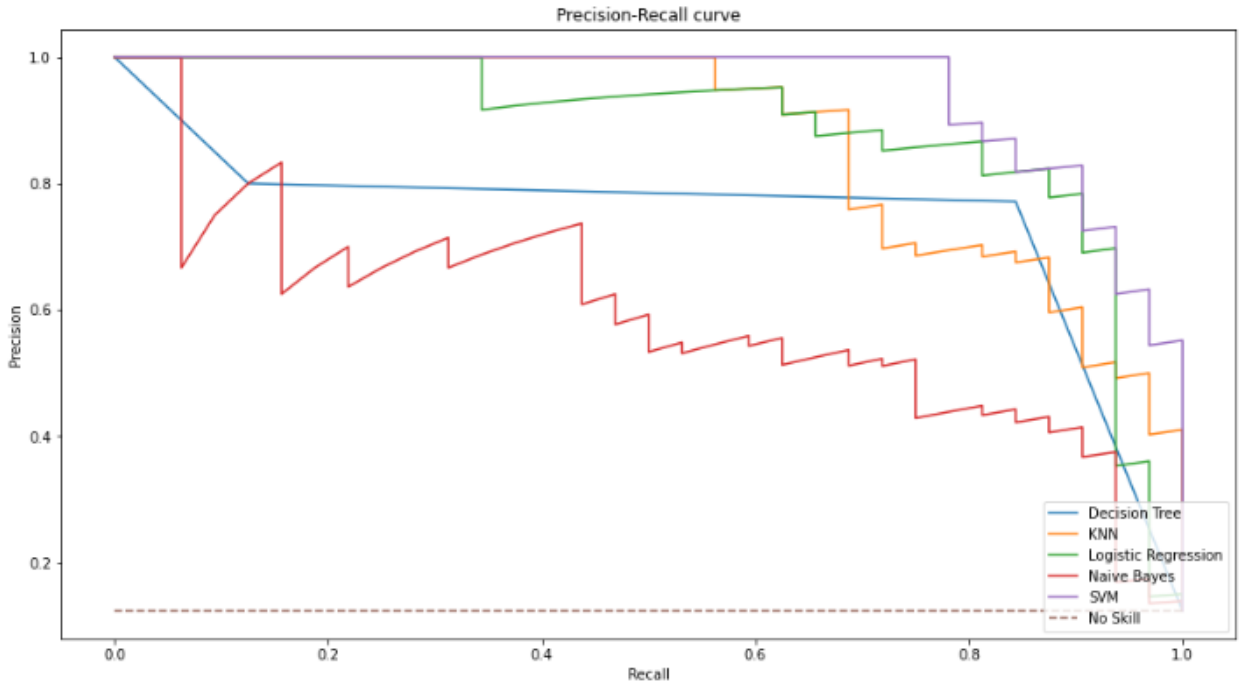
Confusion Matrix




Precision-Recall curve
Precision
Recall
No Skill
train
test
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0

Comparing between Classifiers





	Accuracy	Precision	Recall	F1 Score	AUC Score
Decision Tree	0.949807	0.771429	0.843750	0.80597	0.904254
KNN	0.945946	0.950000	0.59375	0.730769	0.794672
Logistic Regression	0.945946	0.875000	0.656250	0.750000	0.821517
Naïve Bayes	0.911197	0.736842	0.437500	0.549020	0.707737
SVM	0.965251	0.870968	0.84375	0.857143	0.913064

Conclusion:

The SVM Classifier outperformed all other classifiers.