# A Comparative Study to Solve a Binary Classification Problem Predicting COVID-19 Outcomes using 3 Models

Charlie Galvin
Computer Science Department
Durham University

Durham, Durham
lbsr71@durham.ac.uk

***Abstract*- Within this paper, we first introduce the use of predictive models in the Covid-19 Pandemic. We then discuss the workflow used, experiments taken and conclusions in comparing the 3 models.**

## I. Introduction

SARS-CoV-2 and the overall COVID 19 pandemic has shaken the world. The first case, an outbreak in early December 2019 in the Hubei province, spread to more than 180 countries worldwide. Since this publication, 157 million people have tested positive and 3.7 million people have died as a result. While many countries are planning to ease lockdown and covid-19 restrictions, including the UK who could be "past the pandemic" and plan to lift all restrictions on June 21st , others like India are said to be "spiralling out of control" as millions continue to suffer. Although this harsh contrast between countries highlights the pandemics global affect, the regret of not having better mitigations and being more prepared for what many epidemiologists claimed "inevitable" is shared by everyone. Technology has played a huge part in fighting the pandemic, and whilst most relied on using digital technology for data management and integrating it into healthcare systems, predictive modelling was used for planning and response, including the "spread" of the virus. The power of technology has been deemed useful in many areas of the pandemic and the question this experiment aims to answer is "Will a patient live or die from COVID 19?". This experiment aims to solve this classification problem through building 3 predictive models and make comparisons between them, and will use demographic, geographic and medical information for a patient.

## II. Project Motivations

*The motivations for this project include:*

On March 17th, a letter from NHS chief executive Simon Stevens called for NHS bodies to modify health service to expand capacity for coronavirus patients. The establishment of emergency Nightingale hospitals set up to cope with the surge at the start of the pandemic led to a general scare across the country. Although many of these beds were never used, these facilities were established as the Government and the NHS felt the UK would become overwhelmed. Although this did not happen, according to a WHO global pulse survey, 90% of countries report disruptions to essential health services and just two weeks ago, "India's healthcare system was buckling as a record surge in Covid-19 cases puts pressure on hospital beds and drains oxygen supplies". Through using a model to predict if a patient is going to live or die accurately, this can be adopted into healthcare systems in the hope of only helping those that can be saved. This means refusing to waste resources on patients that are going to die and instead focus on patients that can be saved, meaning that at least two people can be saved rather than refusing help to the patient that has a chance. Although the above motivation seems harsh, the predictive model also allows the patient and their friends and family to make peace with the outcome with the limited time they have left. Rather than having the final days in hospital where they will inevitably die, they can enjoy it at home and at peace. This highlights how accurate the model must be however, as to give a false positive where the patient will be turned away, only to die because they have been refused medical help, would not only support the model but would also be ethically wrong.

As a personal motivation, using a binary classification model will mean it will be easier to measure how successful an algorithm is and so model comparisons will be better justified. Furthermore, I find the models used in solving binary classification problems easier to understand.

If deployed, the predictive models used could be extended to be incorporated in online public health services, for example the NHS "Track and Trace" app. Through filling in your personal details and questions it asks, it could tell you whether you will live or die IF you get covid. This may help people understand how at risk they personally are and so may make them obey guidelines, stay indoors and inadvertently decrease the number of patients going to hospital, reducing pressure on healthcare services.

## III. Project Methodology

Within this report, I plan to test 3 different models and review their performance on several different experiments. In order to do this, I will use the following work flow (refer to figure 1):

**Data Collection:** Before any analysis occurs, I will collect data from the real world. This data is called "latest_data"[].

**Data Preparation:** This will include handling different values as well as feature selection. It will also refer to splitting the data into training data used during model development, and testing data used during model evaluation. In certain experiments, part of the training data may also be split into validation data.

**Experiment setup:** To compare the 3 chosen algorithms, we will run several experiments on each model, keeping the control variables the same in each experiment to aid fairness in comparisons. Although this is not normal in an experimental procedure, I thought running different datasets would allow more comparisons between the different models.

**Model development:** For each algorithm we will compare, a model is then built using the training data (not including the validation set when used). Several different model types named hyperparameters may be tested at this stage.

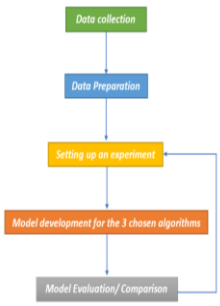



*Figure 1*

**Model Evaluation:** After each final model and hyperparameters are chosen, the final performance of each model is reported. Test data should not be used before this step in order to allow fair representation in each model. The results from each model will be compared against one another using the following performance metrics: accuracy, f1 score and area under the receiver operating curve (AUC). Talk about how we expect to find many false positives as this is the case for a lot of medical data.

***Data Collection***: I used the following dataset in order to achieve my machine learning algorithm. Attached the dataset link . This dataset contain 26763311 samples and includes 32 features including demographic information (age, sex, country), geographical information (latitude, longitude, geo_resolution), past knowledge of the patient (travel_history_location, travel_history_dates) as well as medical information.

## IV. The Experiment

### A. Data Preparation

Within this dataset, there are 2.67 million samples and 32 different features. For this binary classification problem, this is more than enough data. Within the problem, our target variable is "outcome", as this contains the information on the outcome of the patient including whether they have: died, been hospitalised, migrated, recovered, have severe illness, been discharged, are alive, etc. Since our target variable is "outcome" and we have a plethora of data, we can remove any samples that have null values in the "outcome" column as these have no informational value that we can use when training the model.

In order to convert this into a binary classification problem we must "relabel" the data so that any "outcome" relating to death is a 0, and any "outcome" relating to recovered/alive is a 1. In the problem frame, we decided to ignore outcomes including: migrated, severe illness, stable condition, under treatment, "Treated in intensive Care", critical condition as these were not conclusive in their outcome. For motivations such as using the algorithm to decide whether a patient should be allocated a hospital bed, these outcomes will not help decide the basic problem and so I have chosen to remove said samples, including those with null values. This led to the removal of 96.1 % of data, which one could argue may not reflect the overall population, but since we are left with 307382 samples I think this is still a large amount of data to accurately reflect the population. One limitation of this however, is that specific regions may not have recorded all data, such as the removal from India. This is one of the primary countries right now that this model could help so as a secondary motivation we will try and solve this problem as well.

Furthermore, I removed several features that I felt weren't in the problem scope. Although we must be sensitive when removing features due to unknown correlations, I felt we couldn't use:

- Notes_for_discussion, source, sequence_available, additional_information: These were too difficult to process and didn't contain enough results (GET FIGURES) to be deemed useful.
- ID: Each person is unique in the column. Their ID shouldn't contribute and it is safe to assume it will not influence any other feature and so we removed it.
- Latitude and Longitude: I felt this was too difficult to process and their country/ province, if included in the processed dataset, should correlate with these variables.
- Chronic_disease: This feature seemed to be well represented by chronic_disease_binary and it would be difficult to handle the information within the column. Chronic_disease_binary, if included in the processed dataset, should correlate with this variable.
- Reported_market_exposure: This only contains 16 data points so could not accurately represent any population.

With the removal of these features, our model contained 24 features and 307382 results.

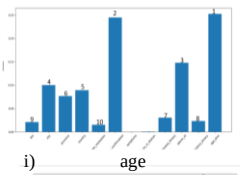
Within our feature selection, we were left with the problem of focusing on one specific group. Although features such as "Lives_in_Wuhan" and "Symptoms" have very few data values, so much so that one could argue it doesn't represent the entire population and should be removed, after reading about how a group of researchers from the University of Pittsburgh created a predictive model that concluded "patients with asthma were less likely to die from pneumonia", I had to be sensitive. Obviously if everyone who lives in Wuhan has died from COVID, this could give light to living conditions or lack of healthcare and so should be included. For all experiments bar one, I decided to relabel all null values into "no", in order to have just two options in the column. I did this for symptoms for the same reasons.

Originally, to filter down results for feature selection, which requires a complete dataset, I decided that age was an important feature, so removed any values not containing this information. This reduced the dataset to 5813 samples. This led me to remove:

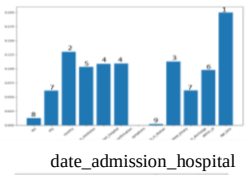
- date_onset_symptoms, date_admission_hospital, date_death_or_discharge: Assumed they were similarly distributed to date_confirmation and they were missing too many values to be create a complete dataset.
- travel_history_location and travel_history_dates: Assumed they correlated with travel_history_binary and were missing too many values.
- location: The three largest hospitals with the most results all followed the same pattern, each with approx.. 96% survival rate or higher. This showed no location had a large influence on outcome. It was also missing too many values.
- admin1, admin2, admin3: These were missing too many values.

With the remaining features, I used SelectKBest, with the chi squared estimator as a statistical measure on how expected 'X' and observed 'O' deviate. We use it between every feature and the target feature and only keep the k number of features with the highest chi^2 value. We then use a RandomForestClassifier to show how relevant each feature is to the target variable. Although it may make sense to use OneHotEncoding (OHE) rather than Ordinal encoding for my features, as there is no relationship between the values, for example "country", and so OHE would be better suited, I found it was too difficult to implement this into my feature selection process and I could not resolve the issue. This may mean that the model assumes a natural order between values in each column but I think the results I achieved with this strategy make sense. Furthermore I decided to use Chi-squared rather than mutual information as well as SVC and Logistic Regression on the age-reduced dataset. All these models achieved the same order of relevant features, so I think it is suitable.

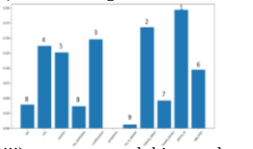
However, I found this method of feature selection to be incomplete, as many features may have lost relevancy only due to the initial reduction of the dataset, by removing all samples with a null value in age. E.g. If a feature directly correlated with the outcome, but not at all with age, then by removing all samples that did not include age may mean that this feature does not appear relevant. Instead, we repeated this process of removing all null values for a particular feature from the original dataset, seeing which features we could use for feature extraction and then performing the feature selection algorithm for 6 of the features out of the 24. This is because these features allowed us to use all the features in at least one relevancy comparison, with features occurring in more comparisons, or with a higher relevancy in one comparison, indicating relevancy. In doing this, we obtain the following relevancy graphs:



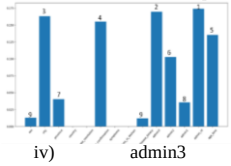
i) age



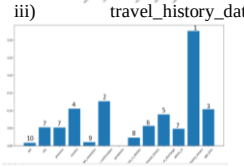
ii) date_admission_hospital



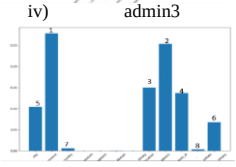
iii) travel_history_dates



iv) admin3



v) date_death_or_discharge



vi) location

From these results on the 6 different subsets of data, we chose the following features: admin_id, history dates, date_confirmation, country, city, age and disease_binary and travel history binary. This meant I had a dataset containing 5000+ results and 8 features. A limitation is still that within these 5 subsets, they may not represent the data as a whole and some "relevant features" may still not be selected.

### *B. The Models*

The models that I will be comparing within this report will be a Logistic Regression Classifier, Support Vector Machine Classifier as well as a Random Forest Classifier. The reasons for choice of the models are as followed:

- Logistic Regression: It was one of the easiest classification algorithms to implement, was fast so could return results quickly and I had also used it within the Bias in AI coursework so I had a strong understanding of how it worked. This was also used for predicting Coronary Heart Disease and obtained relatively high accuracy compared to more complex algorithms so I thought it was a good baseline to compare other models to. A disadvantage with logistic regression is if the number of observations is less than the number of features, it may lead to overfitting. Since that was not the case within my dataset, I thought it was applicable.
- Random Forest : This consists of several individual decision trees that operate together. For a binary classification task, it will output the mode of the classes. I decided to choose this as it was easy to implement and seems to have many differences to Logistic regression. I decided to choose Random Forest over just a decision tree as I felt it was more flexible and would likely be more accurate, allowing the algorithm to compete with SVM and Logistic Regression which are well known for performing well in classification tasks.
- Lastly, I decided to choose SVM as I found it was very easy to visualise and had read that it is very effective and can compete with both models above. After using hyperparameter tuning on a subsample of the processed dataset, this suggested that the optimum kernel was 'rbf' rather than linear. I wanted to see how Logistic Regression could compete with the model as using 'rbf' means SVM can work in higher dimension spaces.

### *C. Hyperparamter Tuning*

For SVM and Logistic Regression, I have decided to the sklearn.GridSearchCV function for hyperparameter tuning. This loops through all the predefined hyperparameters that I want to test and fits the estimator on the training set. From this it selects the best parameters for the predictive task. I decided to use this as I thought when comparing model performance, I should be trying to compare how effective the model is when using all the model's resources, such as SVM's kernel trick to split the data non-linearly. It was relatively easy to implement and model agnostic so seemed like a sensible decision. It is also beneficial as we use cross validation to prevent overfitting, so that the model can perform well to knew data. For a Random Forest, we will use sklearn.RandomSearch [reference] instead as there are too many possibilities in hyperparameters. This means we are selecting a sample of the combinations at random and getting the best result from them. We will use 3 cross validation rather than 5 in order to save time. I have only been able to use hyperparameter tuning once for SVM as it was too time consuming for doing it on each experiment and taking a subsample that was quick enough to tune I felt would not reflect the population. In each experiment I will compare the two tuned models with the SVM model that has only been tuned on the original processed subset rather than in the experiment conditions, which is a limitation.

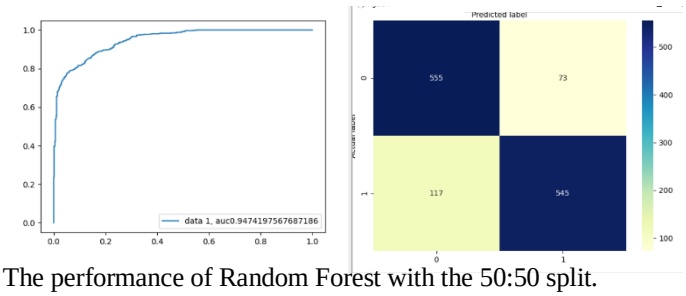
### *D. Experiments*

1: Test each model with the "outcome" value ratio set to 50:50 and 94.2: 5.8
2: Compare the results on the original processed dataset using k-fold cross validation
3: Build a Voting Ensemble Classifier from the models to compare how successful they are together.

Comparing results for different "outcome" ratios

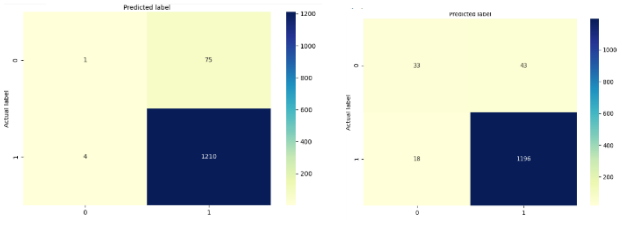
All hyperparameters have were optimised for handling the 50:50 split. I resampled the outcome variable using oversampling and compared how each model handles the balanced target variable.

When we split our data, all the models achieved accurate results. Linear Regression was the lowest with 0.72, which I was surprised with, whilst Random Forest had an accuracy of 0.85 and SVM had an accuracy of 0.78. Looking at the Random Forest's ROC curve, we can determine it has performed the best out of all three of them, due to it's much higher AUC. Although this may be overfitting, I doubt it as Random Forest tends not to overfit data, and the testing performance does not decrease as the number of trees increases so having I believe this result is due to the hyperparameters chosen. It also achieved the highest precision, recall and f1 score out of the models.

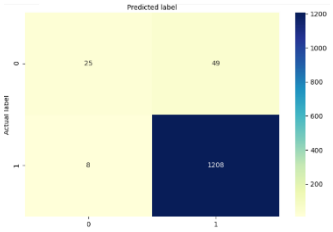



The performance of Random Forest with the 50:50 split.

For the 94.2: 5.8 split, which I have performed as this was the original proportion of the large dataset at the beginning, the LR model had more false negatives than both RF and SVM and had a much lower accuracy overall. I believe this is because with such skewed data, whilst SVM can use kernel tricks to find a hyperplane that is more balanced, the fact we have set the hyperparameters up for the 50:50 split will mean that logistic regression will work best when the data is balanced and so is not as flexible in handling unbalanced data, unlike the Random Forest. This was a hypothesis I had before the experiment and is something I expected. Overall, the RF achieved the highest accuracy within the model again, although all the models suffered with the unbalanced data.




Logistic Regression Random Forest



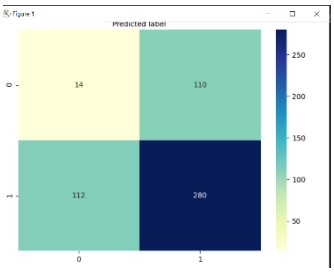

SVM

Random Forest seemed to handle to resist the nature of just predicting the outcome as 1 every single time, meaning that it detected more true negatives than any other model. This It also gave less false positives fewer people would have been sent home only to then die from COVID. Minimising false positives was one of the motivations for this project and so Random Forest has performed far better than the other two models within this experiment.

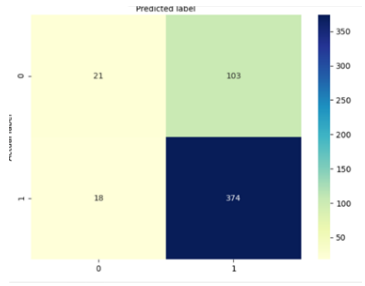
Comparing the models using Stratified K-fold Cross Validation

I decided to us Stratified Cross Validation to compare these predictive models as I think that it is easy to understand and implement. When we split the dataset into folds, this ensures there is a good representative of the whole population each fold. Using this method gives confidence in the performance of an algorithm so I feel like it makes sense to use it in comparisons. The only drawback is that for SVM, I could only use a sample size of 40% of the dataset so it may not be completely representative/ comparable. However, after looking at the runtime of SVM from 20% - 70% of data, we can see that it stays consistent so I have faith that these results are somewhat fair. (figure 2)

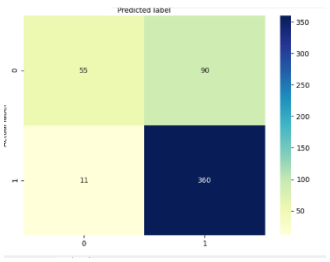
SVM was also much more sensitive to patients dying and had a higher precision overall than either of the other two models. Random Forest also performed well, but Logistic Regression was somewhat lacking as it had the lowest precision, recall and f1-score for the 0 outcome. Although we could argue that SVM has more balanced ratio, I think overall it has performed much better based on these results. Logistic regression looked unsuitable for the task.



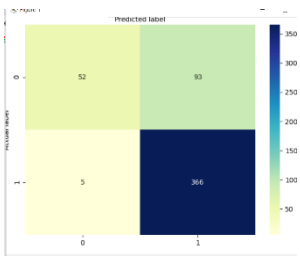

Logistic Regression




Random Forest




SVM




Hard Voting Ensemble

| Sample Size (%) | Run Time (minutes) | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|---|
| 20 % | 2 | 0.83 | 0.9 | 0.4 | 0.55 |
| 30% | 6 | 0.80 | 0.84 | 0.34 | 0.48 |
| 40% | 5 | 0.80 | 0.83 | 0.38 | 0.52 |
| 50% | 7 | 0.82 | 0.76 | 0.33 | 0.46 |
| 60% | 7 | 0.81 | 0.74 | 0.39 | 0.51 |
| 70% | 21 | 0.82 | 0.82 | 0.38 | 0.52 |

(The run time of SVM with k-fold cross validation)

The Ensemble Voting System

As a final experiment, I decided to try and build an ensemble classifier using the models within this paper, to see if them working together was better than individually. I decided to use a "hard" voting system as this is for models that predict class labels, and since all my models were working with binary classification (even if logistic regression gives a probabilistic value first), I felt this was better suited as an ensemble compared to "soft". This achieved the best results in my opinion, as it was very similar to SVM in f1-score, recall and accuracy but had a much higher average precision of 0.865, meaning that it is closer to the actual values.

*E. Conclusion*

I think that SVM is the most suitable model, followed by Random Forest and Logistic Regression is the worst. The reason that I think this is mainly due to the validation experiment, however SVM just seems more consistent across all the experiments that I have conducted. Random Forest model has performed very well, especially for the balanced and skewed data from experiment 1, however , I have decided to choose SVM as I think that it also consistently produces the least false positives which was one of our motivations.

This assignment has taught me a great deal. Not only about how sensitive machine learning tasks can be and that we must be careful when handling data, but it has also taught me a lot about the machine learning pipeline. I was surprised at how much time I had to dedicate to pre-processing to ensure that the results I got were fair and represented the population correctly. I have also learned how much time must go into hyperparameter tuning and, had I started the coursework earlier, I would have been able to properly tune the SVM model. I now know what we are looking for when we come to measuring successful machine learning algorithms, including how it is topic dependent and that we must weight up a lot of things such as false positive rate, accuracy, bias, etc. as well as how to ensure the measurements we take are accurate through different types of validation. Lastly, I've learned that I must manage my time more carefully, but that no matter how hard things can be, I can achieve anything.

## ACKNOWLEDGMENT

I would like to acknowledge Lei Shai for being an awesome lecturer.


## REFERENCES


[1]https://www.thelancet.com/journals/landig/article/PIIS2589-7500(20)30142-4/fulltext

[2]https://www.thelancet.com/journals/landig/article/PIIS2589-7500(20)30196-5/fulltext

[3] https://www.future-science.com/doi/10.2144/fsoa-2020-0051

[4] https://www.bbc.co.uk/news/world-asia-56858403

[5] https://www.bbc.co.uk/news/live/uk-55505777

[6] https://www.health.org.uk/news-and-comment/charts-and-infographics/did-hospital-capacity-affect-mortality-during-the-pandemic

[7] https://www.bbc.co.uk/news/health-56327214

[8]https://www.telegraph.co.uk/news/0/do-many-nhs-nightingale-hospitals-remain-empty/

[9] https://www.who.int/news/item/31-08-2020-in-who-global-pulse-survey-90-of-countries-report-disruptions-to-essential-health-services-since-covid-19-pandemic

[10] https://www.nature.com/articles/s41598-020-79000-y

[11]https://www.forbes.com/sites/tomtaulli/2019/08/04/bias-the-silent-killer-of-ai-artificial-intelligence/?sh=498a4da97d87

[12] https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/#:~:text=How%20Does%20SVM%20Work%3F,-The%20basics%20of&text=A%20support%20vector%20machine%20takes,to%20the%20other%20as%20red Outcomes for SVM

[13]https://www.thepythoncode.com/article/feature-selection-and-feature-engineering-using-python
This helped me with features selection, using Random Forest Estimator and Chi-squared to obtain features.

[14https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74
Performing Random Forest Hyperparamter tuning