Machine Learning Scientific Report

1. Problem framing

The Covid-19 pandemic is still developing today. Between December 2019 and May 2021, there have been over 160 million cases and over 3 million deaths globally. To better identify trends and assist in medical care, we can turn to machine learning (ML) for support.

In this paper, we build and compare predictive models using ML algorithms and epidemiological data from the Covid-19 outbreak. The dataset we use is by Xu et al. (2020) and is publicly available on GitHub.[1] The problem we want to solve is of predicting confirmed cases' outcomes as either *died* or *discharged* (from hospital). The motivation for solving this problem is that the prediction model built may be helpful for quickly allocating medical care to patients at a time when there are limited resources.

2. Experimental procedure

2.1 Data preparation

First, to better understand the data, we produced a number of visualisations: we created a bar plot of up to the 10 most frequent values of each categorical attribute (e.g., *symptoms*); we created a scatter plot of the *latitude* and *longitude* of each case; and, we plotted the daily and monthly number of cases globally and by country. From these visualisations, we got a strong overview of the spread of Covid-19 by location and by time.

Second, we began preparing the dataset for ML algorithms by removing irrelevant instances. These were instances that were not useful for the task. Since we were training models to predict outcome, irrelevant instances were those whose value of the target, *outcome*, was not one of either *died* or *discharged*. To remove such instances, we had to correct structural errors, such as inconsistent capitalisation and wording among target values: we replaced *Died*, *dead*, or similar values with *died*; and, we replaced *Discharged*, *recovered*, or similar values with *discharged*. Instances with target values whose meaning was dissimilar to either of these two values, including *null* values, were dropped.

Third, we kept attributes which we thought would be useful. The attributes kept, which became features, were *age*, *sex*, *latitude*, *longitude*, *date_onset_symptoms*, *symptoms*, *chronic_disease_binary*, and *travel_history_binary*. After this, we replaced the *symptoms* and *date_onset_symptoms* features with a new *symptoms_binary* feature, and cleaned up the dataset by dropping any remaining instances with missing values.

Fourth, we transformed numerical data. We performed data binning on the *age* feature, and created categorical values for the age ranges *0-14*, *15-34*, *35-59*, *60-79*, and *80+*. These were chosen because the *age* feature was in fact already categorical, and the latter four categories were the four most frequent. We also performed feature scaling on the *latitude* and *longitude* features by standardisation, also known as Z-score normalisation, which rescales the feature values to have zero-mean and unit-variance.

1. github.com/beoutbreakprepared/nCoV2019/

Fifth, we transformed categorical data. We performed one-hot encoding on the *age* and *sex* features since they had nominal data. Also, we encode the target values so that *discharged* corresponds to 1 and *died* corresponds to 0. The reason for performing these transformations is that ML algorithms require features to be numbers.

Finally, we randomise the resulting dataset, which has 5,206 instances, as we do not want the order of the instances, which is irrelevant, to affect the model training process; and, we split the data into a training set, validation set, and test set to prevent overfitting. We create a test set by selecting a 20% subset of the dataset so there are at least 1,000 test instances, which is large enough to yield statistically meaningful results. Since we suspect the *chronic_disease_binary* attribute to be an important feature to predict *outcome*, we chose to ensure that the the test set was representative of the overall *chronic_disease_binary* distribution by doing stratified sampling. The overall distribution was that 98.2% of values were *False* while the remaining 1.8% were *True*. We also observe that we maintain an exact balance of train and test target values (75.3% being *discharged*), which makes it easier to train models.

2.2 Model selection, training, and testing

The three ML algorithms we chose to build predictive models were logistic regression classification (LRC), gradient boosting classification (GBC), and support vector classification (SVC). For each of these three models, we iteratively train the model on the training set, evaluate the model on the validation set, and tweak the model according to the results on the validation set. In particular, we employ 4-fold cross-validation so that the size of the validation set is equal to the size of the test set. This further reduces the chance of overfitting. Then, we pick the model with the set of hyperparameters that does best on the validation set and confirm the results on the test set. This approach allows us to compare different trained models in an unbiased way, by comparing model performance using the test set, which is kept apart from the training process and is therefore unseen data.

2.3 Hyperparameter tuning

We tried different hyperparameter combinations for each model. For LRC, we tried values 0.001, 0.01, 0.1, 1, 10, 100, and 1000 for C, the regularisation hyperparameter; and, we tried L_1 and L_2 regularisation for the norm used in the penalisation. For GBC, we tried values 2, 4, 8, and 16 for the minimum number of samples required to be at a leaf node (*min_samples_leaf*); and, values 0.001, 0.01, and 0.1 for the learning rate. And, for SVC, we tried values 0.75, 0.85, 0.95, and 1 for C; and, linear, polynomial, radial-basis, and sigmoid kernel types.

For LRC, the best model achieved a 0.80 accuracy score with hyperparameter combination $C = 1000$ and L_2 regularisation. For GBC, the best model achieved a 0.85 accuracy score with hyperparameter combination *min_samples_leaf* $= 2$ and learning rate 0.01. And, for SVC, the best model achieved a 0.84 accuracy score with hyperparameter combination $C = 1$ and radial-basis kernel type.

3. Results

Having found the best hyperparameter combinations to train the best models for all three ML algorithms, we then evaluate the trained models on the test set. To enable evaluation, we produce the following: the confusion matrix, positive predictive value (PPV), negative predictive value (NPV), sensitivity, specificity, F_1 score, accuracy, and receiver operating characteristic (ROC) curve, for each model.

Table 1: Confusion matrices produced by each trained model on the test set.

(a) LRC.

Pred. / Real.	0	1	all
0	185	55	240
1	148	654	802
all	333	709	1042

(b) GBC.

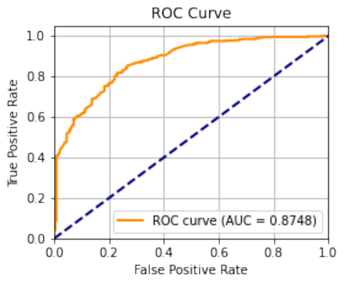
Pred. / Real.	0	1	all
0	112	128	240
1	19	783	802
all	131	911	1042

(c) SVC.

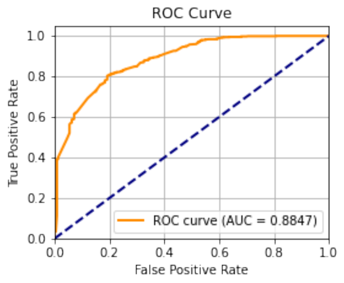
Pred. / Real.	0	1	all
0	109	131	240
1	24	778	802
all	133	909	1042

Table 2: Statistics produced by each trained model on the test set.

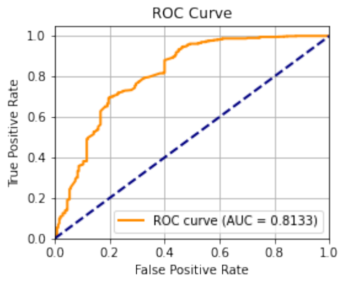
Model	**NPV**	**PPV**	**Specificity**	**Sensitivity**	F_1 **Score**	**Accuracy**
LRC	0.56	**0.92**	**0.77**	0.82	0.87	0.81
GBC	**0.85**	0.86	0.47	**0.98**	**0.91**	**0.86**
SVC	0.82	0.86	0.45	0.97	**0.91**	0.85



(a) LRC.



(b) GBC.



(c) SVC.

Figure 1: ROC curves produced by each trained model on the test set.

By looking at the area under the curve (AUC), which is a measure of separability, of the ROC curves in Figure 1, we see that GBC produces the highest value followed by LRC then SVC. Since a higher AUC value corresponds to more ‘idealistic’ behaviour (i.e., more correct classifications), we determine that GBC has the best performance for solving our classification problem, followed by LRC then SVC. This ordering is also reflected by the statistics in Table 2, which show that GBC performs best in 4 metrics (highlighted in bold), followed by LRC in 2 metrics, and SVC in only 1.

4. Discussions

4.1 Chosen models

The reason we chose to implement LRC and GBC is that they were demonstrated in the practical *Exercise for Logistic Regression*[2] with breast cancer prediction. As such, we base our code on these demonstrations. As for SVC, we chose this algorithm because it was recommended for our problem after consulting the 'ML Cheat Sheet' (for scikit-learn)[3], since we had less than 100k samples and no text data.

4.2 Experimental procedure

Although standardisation was useful for LRC and SVC, it does not affect tree based models such as GBC since they are not affected by linear transformations. Nonetheless, standardisation would have helped the model better deal with outliers. However, because we only performed standardisation, we do not know whether min-max normalisation would have produced better results. In future, we can try fitting the model to raw data, min-max normalised data, and standardised data, and compare the results to find the best.

Another procedure we could have done differently is to split the dataset into train and test sets before performing any data cleaning or transformation. Had we done so, we may have gotten different results.

4.3 Limitations

One limitation of the implementation is that the predictive models use only a handful of attributes that were present in the raw dataset to perform classification. We were going to include the *country* feature in model training; however, we found that the values were skewed towards one country (*Phillipines*), which might affect training performance. Similarly, we wanted to include the *date_confirmation* attribute; however, our initial exploration of the dataset showed that there were many outliers (spikes in the number of cases around the start or end of each month), and so the attribute might not contain much useful information.

Another limitation of the implementation is that we do not explicitly perform down sampling or up-weighted during model training to account for the imbalanced data set, since *discharged* accounts for about 75% of *outcome* values. Had we done this, we may have observed fewer false positives or false negatives in the confusion matrix.

Finally, we could have included more hyperparameters to try in our combinations, such as the deviance and exponential loss functions to be optimised for GBC and degree of the polynomial kernel function for SVC.

5. Conclusions and lessons learnt

Overall, we conclude that GBC is the best ML algorithm at solving our problem of predicting confirmed cases' outcomes as either *died* or *discharged*. Moreover, we learned the lesson that we can never conduct enough experiments in ML, since there are always more algorithms to try, attributes to select, and hyperparameters to tune.

2. tinyurl.com/COMP2261prac6
3. scikit-learn.org/stable/tutorial/machine_learning_map/index.html

References

Bo Xu, Bernardo Gutierrez, Sumiko Mekaru, Kara Sewalk, Lauren Goodwin, Alyssa Loskill, Emily Cohn, Yulin Hswen, Sarah C. Hill, Maria M Cobo, Alexander Zarebski, Sabrina Li, Chieh-Hsi Wu, Erin Hulland, Julia Morgan, Lin Wang, Katelynn O'Brien, Samuel V. Scarpino, John S. Brownstein, Oliver G. Pybus, David M. Pigott, and Moritz U. G. Kraemer. Epidemiological data from the COVID-19 outbreak, real-time case information. *Scientific Data*, 7(106), 2020. doi: doi.org/10.1038/s41597-020-0448-0.