Predicting Covid-19 Hospital Admissions

Abstract—**This paper provides a comparison of three regression models for predicting covid-19 hospital admissions.**

I. Introduction

Our dataset contains data on global positive covid-19 patients. Through a review of the dataset and the available features, we decide to investigate the length of time before admission to hospital after the onset of covid-19 symptoms.

In a real-world implementation of our models this is an appropriate task since there are two fundamental ways that someone can be aware of having covid: a positive diagnosis or the onset of symptoms. The former is not as easily investigated in our given dataset since its relevant feature (‘date confirmation’) contains a large amount of missing data.

We implement and compare three regression models (linear regression, support vector regression, and k-nearest neighbours regression) that predict the number of days before a given case should be admitted to hospital, and one simple classification model (logistic regression) that provides the initial prediction of whether a case will of will not be admitted to hospital. Our models follow from a few assumptions that are discussed throughout the paper.

In a real-world application, the initial classification model will be able to more accurately assess the exact number of incoming patients to a given hospital, with the regression model providing an enhanced level of detail for the hospital to appropriately prepare. It may also function as a metric for severity – i.e., the sooner someone is predicted to be admitted to hospital, the more severe their infection. This could be used to triage patients more effectively so that medical resources are applied where they are most needed. It may also be used to assess how covid-19 affects various demographic groups.

We begin by reviewing our dataset cleaning process in section 2. In section 3 we investigate our initial classification model and in section 4 we implement our three regression models and compare their performance.

II. Dataset Cleaning

A. Dataset Discussion

The main problem associated with our given dataset is the large quantity of missing data which is likely responsible for most performance issues associated with our models. For example, symptomatic data is almost entirely missing from our dataset which would provide a very useful metric for evaluating the severity of the infection and the likelihood of hospital admission.

The dataset also only provides information up until summer 2020, so any changes in the genetic makeup of the virus or treatment used in hospital are not learnt by our models. For example, the virus has already gone through many mutations that changed its transmissibility and severity. As such, if we were to test our models on current data, we would likely find a reduction in their performance as a result.

B. Extracting the Ground Truth Label

Our regression task requires predicting the difference between the features ‘date onset symptoms’ and ‘date admission hospital’, we therefore extract rows of the dataset that satisfy having both features. When calculating the difference between dates we remove cases that provide a negative difference as in a real-world application we should only be predicting the positive time until hospitalisation.

We notice that filtering our dataset in this way implies an underlying assumption in our model that cases will be admitted to hospital. For a real-world implementation of our model, this is not logical since most actual cases are never admitted to hospital. For this reason, we also implement a classification model to initially predict the outcome of our patient as either being or not being admitted to hospital. From this, we can then apply our regression model if our classification model returns a positive affirmation.

Our dataset provides no recognition of cases that are not admitted to hospital. There are a few features that we expect to provide this information, namely ‘date admission hospital’ or ‘outcome’. However, ‘date admission hospital’ does not differentiate between missing data and cases that were never admitted to hospital. Similarly, outcome only provides the outcome of hospital admissions rather than the outcome of all cases and there is no way do differentiate between negative and missing data.

Hence, we apply the heavy assumption that a missing ‘date admission hospital’ value implies that the case was not admitted to hospital. This is obviously not good practice; however, we deem it necessary so that our regression models can have theoretical real-world implications. Secondly, the focus of our project is on the regression not the classification task.

We take our already identified subset for regression as the positive labels, and our negative labels from the assumption above. As a result, we encounter some complications. We find that many cases provide supposedly contradictory negative labels saying that they both were and were not admitted to hospital. Obviously, this is due to our assumption, but to maintain consistency, we restrict our negative samples to those that provide neither an ‘outcome’ nor a ‘date death or discharge’ value since we know for certain that a non-missing value in either of these columns tells us that they definitely were admitted to hospital.

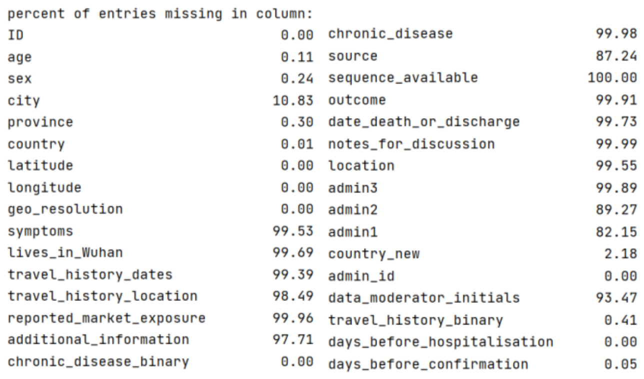
C. Finalising Our Features

When calculating our label, we notice that we can extract more data regarding the time between ‘date onset symptoms’ and ‘date confirmation’. This represents the number of days before getting a positive covid-19 test result. In this case, we accept negative times and retain missing values.

The dataset is rife with formatting inaccuracies, we attempt to correct some of these in the ‘age’ feature since it is a very useful demographic metric. For ages that are displayed as a range, we take a random value from a uniform distribution to retain the continuous formatting of the feature. This assumption is obviously not realistic as age follows a normal distribution, but the number of cases that use this format and the small size of the range of values in most cases validate this simplistic approach. Moreover, respecting the various age structures of different countries is complex and using a world-based distribution has its own assumptions.

D. *Missing Data*

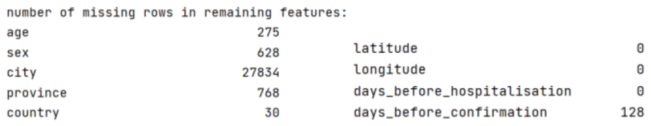
Most features in our dataset contain large amounts of missing data; the exact percentages are shown below. For this reason, we drop features that have 80% or more of missing data. This value could have been dropped to only requiring 11% of missing data to still retain the same features. This highlights the extent to which data is lacking from this dataset.



```
percent of entries missing in column:
ID                          0.00  chronic_disease               99.98
age                         0.11  source                        87.24
sex                         0.24  sequence_available           100.00
city                       10.83  outcome                       99.91
province                    0.30  date_death_or_discharge       99.73
country                     0.01  notes_for_discussion          99.99
latitude                    0.00  location                      99.55
longitude                   0.00  admin3                        99.89
geo_resolution              0.00  admin2                        89.27
symptoms                   99.53  admin1                        82.15
lives_in_Wuhan             99.69  country_new                    2.18
travel_history_dates       99.39  admin_id                       0.00
travel_history_location    98.49  data_moderator_initials       93.47
reported_market_exposure   99.96  travel_history_binary          0.41
additional_information     97.71  days_before_hospitalisation    0.00
chronic_disease_binary      0.00  days_before_confirmation       0.05
```

Of our remaining 15 features, we remove 'ID' to prevent a perfect correlation with our label. We remove 'chronic disease binary' and 'travel history binary' since it contains very little predictive data as almost all cases have the Boolean 'False'. Finally, we remove 'geo resolution', 'country new', and 'admin id' since they only contain administrative data. We are left with 7 features that contain age, sex, and geographical location data alongside our previously computed 'days before confirmation' and label.

Within these features we still have some missing data, the number of missing rows for each respective feature are shown below in figure 2. The feature 'city' contains the most missing data. We decide to bin this missing data into the value 'city other' to retain as many positive labels as possible. We do not choose to impute any of the other missing data since the amount missing is negligible.



```
number of missing rows in remaining features:
age                    275
sex                    628      latitude                        0
city                 27834      longitude                       0
province               768      days_before_hospitalisation     0
country                 30      days_before_confirmation      128
```

E. *Formatting for Our Models*

Our models require strictly numerical data as inputs. To satisfy this, we encode our categorical data using 'one hot encoding', creating many binary dummy variables for each unique value of each feature. This is superior to our previously implemented 'label encoding' – which simply converts string values to a numeric counterpart – since it does not create a false correlation between values in categorical features.

Encoding produces too many features, of which most contain large amounts of negative binary data. We therefore drop any columns that contain 99% or more negative binary values. This threshold is very high; however, most cases are grouped within a select few values, so we only end up removing all but a few well populated features.

We also normalise our (non-label) data to between 0 and 1 as it is required for many machine learning models and allows for comparable features. 'Minmax scaling' is chosen over 'standard scaling' as we wish to retain the weighting of the binary features. We finally split our dataset into a classification and regression dataset.

III. Classification Model

A. *An Initial Implementation*

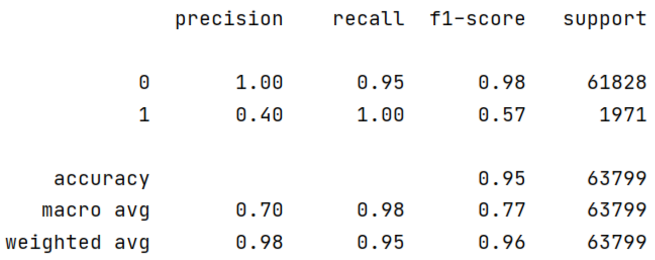
As stated earlier, the classification model is only implemented to give our regression models a real-world implication so is only discussed briefly. Nevertheless, in fully evaluating the performance of our regression models, we need to review the performance of our chosen logistic regression model. The relevant plots are available in the given folder.

We notice a severe imbalance between the positive and negative labels in our dataset which would allow our model to achieve an accuracy of 97% if it were to only predict the negative label. This is obviously not realistic for our task since the positive labels are of most importance. We implement hyperparameter tuning to mitigate this.

B. *Hyperparameter Tuning*

The scoring functions available are not suitable for evaluating our imbalanced data so we create a custom scoring metric that weights the true negative rate (TNR) and the true positive rate (TPR) equally.

The performance of this tuned model is shown below. We notice a significant increase in the TPR (recall) at a slight cost of both accuracy and precision. This is a desired result since the TPR of our model is of most importance. A TPR this high is surprising and may suggest some unforeseen corelations in the data, or simply that given the sheer quantity of our training data we are able to achieve such a high accuracy. Given the focus of this paper we will not investigate further.



```
              precision    recall  f1-score   support

           0       1.00      0.95      0.98     61828
           1       0.40      1.00      0.57      1971

    accuracy                           0.95     63799
   macro avg       0.70      0.98      0.77     63799
weighted avg       0.98      0.95      0.96     63799
```

IV. Regression Models

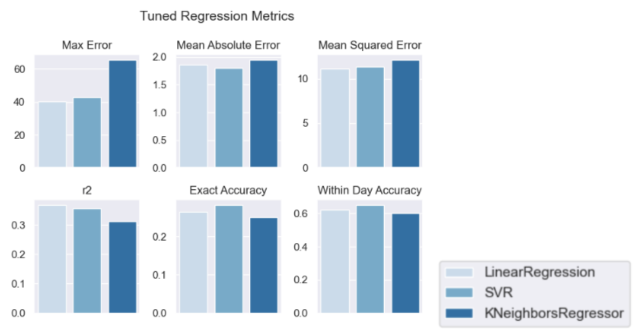
A. *Implementation*

We implement three regression models for predicting the number of days before hospital admissions given the onset of symptoms – linear regression (LR), support vector regression (SVR), and k-nearest neighbours regression (KNR). As with our classification model, we hyperparameter tune our regression models with respect to the scoring metric 'r2'. The corresponding plots for our untuned models are available in the given folder.

Most models have assumptions on the training data. For example, LR has the following: linear relationship, multivariate normality, no or little multicollinearity, no autocorrelation, and homoscedasticity. Although not all assumptions are explicitly checked, it is likely that many of them are violated which affect the performances and reliability of our models.

B. *Regression Metrics*

Our first comparison technique is to evaluate various available regression metrics. The standard definitions apply to all but 'exact accuracy' and 'within day accuracy' which correspond to the percent of predictions that are exactly correct and correct within one day, respectively. The results for our three models are shown below.



From the two accuracy metrics, we can conclude that SVR performs the best overall as it has a higher chance to correctly predict a patient's time before hospitalisation both exactly correct and correct to within one day. Even though most of our observations are admitted within a few days, it is important to check the accuracy outside of this range as discussed later.

The mean absolute error (MAE) of our models are all too similar to make any distinct comparison. Mean squared error (MSE) is more sensitive to outliers than MAE due to the squaring property and we see that KNR has the highest MSE, so its predicted values are much further from the true value. This is supported by the max error of KNR which is significantly higher than the other two models, although this only corresponds to one erroneous value so is not a good measure of overall performance.

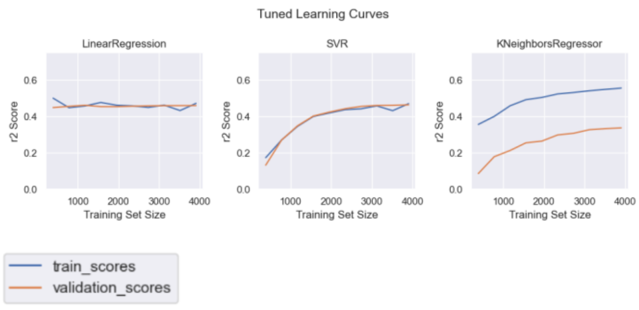
R2 is a useful metric for the 'goodness of fit' since it represents the proportion of variance in the dependent variable that is explained by our independent variables. As before, we can conclude that KNR provides the worst fit since only ~30% of observed variation can be explained by the model's inputs. It is worth noting that despite all our r2 scores being low, this is not always a bad thing since our data set may have an inherently greater amount of unexplained variation.

A related metric to consider is adjusted r2 which adjusts for the number of independent variables that do not fit the model. We could also check Poisson and gamma deviance if our models did not predict some negative values.

In almost all metrics, LR performs slightly worse than SVR but significantly better than KNR so lies somewhere between the two in terms of performance.

C. Learning Curves

Our second comparison technique is to construct learning curves that show the validation and training r2 scores for varying numbers of training samples. From these curves we can assess the model's performance based on the amount of training data and whether it leads to over or underfitting.



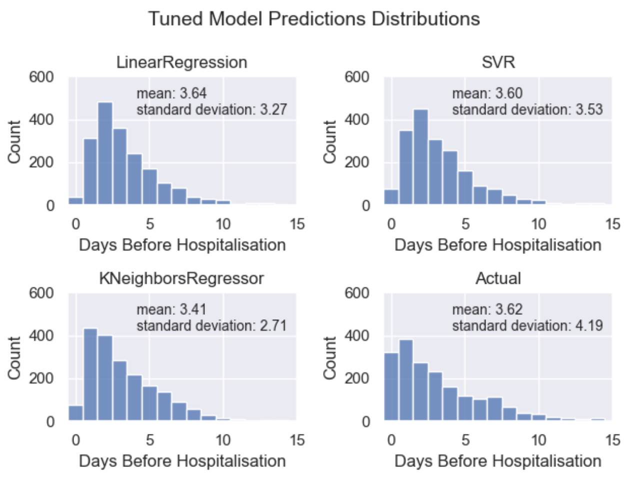
We note that the r2 score of LR does not change as the training set sizes increase, neither do the validation scores stray from the train scores. As a result, we can conclude that LR does not over or underfit and performs well on small datasets.

On the other hand, the r2 score of SVR increases and plateaus as the training set sizes increase so SVR requires more training data than LR. As above, the validation scores do not stray from the training scores so we can conclude that SVR does not over or underfit.

From the third plot, we can see that the r2 score of KNR increases and plateaus similar to SVR. However, more notably, the validation scores are consistently significantly lower than the training scores so KNR suffers from overfitting for all training sizes.

D. Predicted Distributions

Our third comparison technique involves plotting the distribution of our predicted results for each model. The histogram bins each value to the nearest integer as our true label only contains integer days. We compare the mean and standard deviation as a metric for quantifying the distributions.



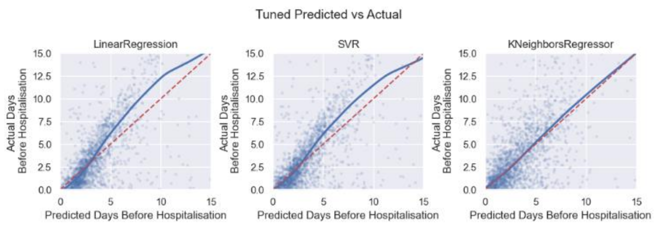
The first obvious difference is the inability of all three models to predict 0 days before hospitalisation which is the second most common value in our actual labels. Visually, although hard to differentiate, it seems that KNR most closely mimics the distribution of our real cases.

However, when comparing the mean and standard deviation, we find that SVR is most similar to that of our actual results. Through these numerical comparisons we can also propose that LR performs somewhere between that of KNR and SVR; this is the same result we reached from comparing regression metrics.

The main problem with this comparison technique is the inability to compare the differences between each unique prediction and its corresponding real label. For example, we could theoretically misclassify every predicted value and still achieve the actual distribution curve, so this not a good visualisation for performance.

E. Predicted Vs Real Pairs

A more appropriate visualisation is a scatter plot of the predicted versus actual values from which we can tell how well each individual case is classified. The red dashed line represents the ideal curve where each case is predicted correctly, and the blue solid curve represents the locally weighted scatterplot smoothing (LOWESS) curve which aims to visualise any trends in the given data. We have also applied a slight jitter to help visualise the results by shifting the values from their integer coordinates.



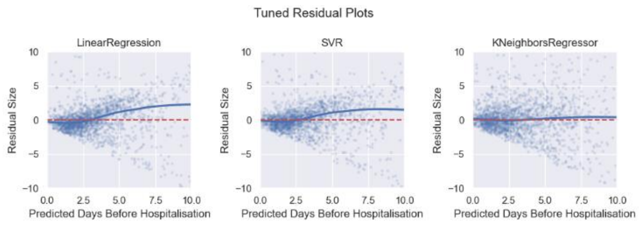
We notice that both LR's and SVR's LOWESS curves tend away from the ideal line. We interpret this as both models underpredicting as the number of days before hospitalisation increases. This is unsurprising as most of our data is clustered between 0 and 5 days.

Interestingly, in juxtaposition with our previous conclusions, the LOWESS curve of KNR seems to most closely reflect the ideal curve so represents the best predictive model. However, it is important to note that the LOWESS curve is only one way of visualising any trends in the data and does not represent the distribution as a whole.

For example, it seems that the distance of each point from the ideal line is much larger for KNR which would reduce its accuracy. This supports the regression metrics in our first comparison technique which quantify this error. Therefore, it is worth considering the distribution of our points from the ideal curve alongside any trends in our visualisation.

F. Residual Plots

To aid this approach, we introduce our final comparison technique to evaluate the distribution of errors or residuals. As above, the red and blue lines represent the ideal and trend lines, respectively. We have also applied a slight jitter here.

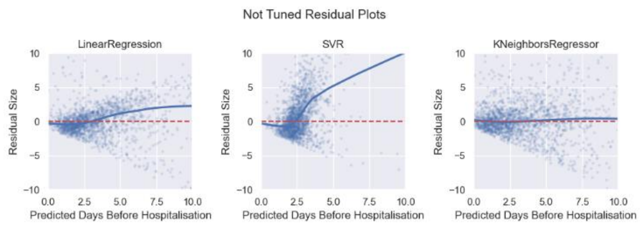


Briefly we first note that the straight line observed on the bottom of all three plots is drawn from the x-axis in our previous plots since the real labels are never negative. We also see this phantom line mirrored at the top of our first two models which reflects their capacities to not predict negative values. This is not satisfied as strongly in our KNR model.

We can now more easily see that the variance of the error in all our models increases so we less accurately predict higher values. This violates LR's assumption of homoscedasticity – that the variance of the residuals is equal for all predicted values – so we cannot trust as highly this model's statistical inferences such as p-values and standard errors.

G. Hyperparameter Tuning

Briefly, we discuss the results of hyperparameter tuning on our models. We choose the residual plots since they display many aspects of a model's performance.



In contrast to our previous conclusions on our tuned model's, we find the SVR performs significantly worse than our other two models. It tends to severely underpredict all observations so highlights the importance of the chosen initial parameters for support vector machines.

This is worth considering when choosing which model to apply. In fact, since the plots for LR and KNR are similar to their untuned models, we may conclude that LR is the best choice for an out-of-the-box regression model without hyperparameter tuning. Of course, we can check the other plots available in the given folder to validate this decision.

V. Conclusion

After evaluation of the various comparison techniques, we may safely conclude that SVR and LR perform very similarly with SVR being the better choice given its slightly better error and distribution results. However, if we had a smaller sample of our given dataset, it would be more appropriate to implement LR given its ability to accurately predict for small training sizes. We also might choose LR given its simpler interpretability than SVR; its easier to understand how each predictor affects the model's performance.

One of the main benefits of KNR is that it is non-parametric – has no assumptions on the data. This is a useful property since we do not have to try to correct any assumptions that have been violated, which is often the case in real-world data. However, it still performs worse than both models in almost all comparison techniques.

To finalise the real-world implications of our models, we combine our best regression model with our classification model. By multiplying our logistic regression's accuracy with the accuracy of SVR, we can conclude that there is 26.8% chance of exactly predicting the time until hospitalization of an unknown patient and a 62.0% chance of correctly predicting the time until hospitalization within one day. If we were to naively predict the median of the training data, we would achieve an exact and within day accuracy of 15.8% and 34.6% respectively. Our machine learning approach is therefore significantly better than the best naïve approach.

The main limitations associated with all our models stems from the initial data; we could significantly improve the performance of all models if we were to sample more detailed data. As such, it may be worth considering imputing missing data or more sophisticated feature selection using an algorithmic approach rather than a human approach to retain as much of our available data as possible.

We may also see further improvements in performance if we treated outliers since many algorithms such as KNR are particularly sensitive to them. We could also experiment with unique scoring metrics for each algorithm during hyperparameter tuning to further optimize their respective performances. Similarly, it may be worth considering an ensemble method to mitigate the relative performance issues associated with each model.