AN8005 Fundamentals of Machine Learning

Business Analytics, AY 2020-21, Trimester 2

Group Project Report

Members	Wang Jingyi	G2000751D
	Wei Kexian	G2000380D
	Wu Di	G2000642A
	Wu Mengran	G2000818D
	Xiong Yike	G2000638G
Group	Group 5	
Instructor	Adams Wai Kin Kong	
Submission Date	15 Feb 2021	

Table of Contents

1. Problem Statement

On March 11,2020, the World Health Organization (WHO) officially announced the new coronavirus pneumonia (COVID-19) as a global pandemic (Pandemic). This was the first time that coronavirus infection was assessed as a global pandemic. By 2020, COVID-19 has killed 1798154 lives in the world. To reduce the death rate, it is important to maximize the limited medical resources through analysis of personal features and pathological characteristics of patients. In this paper, we would build up machine learning models to do COVID-19 cases death prediction. In this way, we hope to identify what are the personal features that lead to death and offer suggestions on special care to weak people. To solve the problem, we collected COVID-19 personal cases data from CDC(Centers for Disease Control and Prevention) in the U.S.

2. Data source

On April 5, 2020, COVID-19 was added to the Nationally Notifiable Condition List and classified as “immediately notifiable, urgent (within 24 hours)” by a Council of State and Territorial Epidemiologists (CSTE) Interim Position Statement (Interim-20-ID-01). CSTE updated the position statement on August 5, 2020 to clarify the interpretation of antigen detection tests and serologic test results within the case classification. Collected by jurisdictions and shared voluntarily with CDC, the COVID-19 case surveillance system database includes individual-level data reported to U.S. states and autonomous reporting entities, including New York City and the District of Columbia (D.C.), as well as U.S. territories and states. It includes demographic characteristics, exposure history, disease severity indicators and outcomes, clinical data, laboratory diagnostic test results, and comorbidities. Detailed description for each attribute in the dataset is as follows.

Attribute	Description
cdc_report_dt	Date of confirmed case reported
pos_spec_dt	Date of first positive specimen collection
onset_dt	Onset date
current_status	Current status of the person
sex	Gender
age_group	Age group categories
Race and ethnicity (combined)	Case demographic
hosp_yn	Whether the patient hospitalized
icu_yn	Whether the patient admitted to an intensive care unit (icu
death_yn	Whether the patient die as a result of this illness
medcond_yn	Whether the patient have any underlying medical conditio and/or risk behaviors

Table 1 Description for attributes

3. Data Cleaning

We have 11 columns in total. The target variable we want to analyze is death_yn, whether the patient will die as a result of this illness. There are three types of dates that indicate the status of the person. However, this has no relation with the final result, so we decide to drop when building the models.

The process for data cleaning can be divided into several parts:

1) First, we should check with the columns and their corresponding types and convert them respectively if needed.

2) Then, we should identify abnormal records which contains missing values and duplicates in this dataset. As our dataset is large enough, directly dropping them will not affect the final result. Sometimes, there are some strings that can affect our analysis but can not be directly identified. So we should check with each column and find out abnormality. Then replace them with NA.

3) Another mistake is the logic relationship between some variables. For example, if a person has been admitted to intensive care unit, he must be hospitalized.

After data cleaning, we then apply it into modelling.

4. Data Modelling

4.1 Logistics Regression

Logistic regression is a combination of linear regression and logistic function, or sigmoid function. It is a simple but it useful and applicable to many situations and has many extensions. In this study, we build our own version of logistic regression in Python and we will discuss the process here. The formula of logistic regression is written as $\hat{y} = g(z) = 1/(1 + e^{-z})$, where $\hat{y}$ is the predicted value of y and $z = w \cdot x$. Here, we are using an entropy-like loss function.

$$Entropy(t) = -\sum_{j} \quad p(j \mid t)p(j \mid t)$$

However, since we are predicting 0 or 1, namely binary values, the entropy function needs some modifications in case 0 ruins it. So, it should be like

$$Entropy(x) = -\sum \quad \{\hat{y} \qquad y = 0 \; 1 - \hat{y} \qquad y = 1$$

$$= -\frac{1}{n}\sum^{n} \quad y \cdot \hat{y} + (1 - y) \cdot 1 - \hat{y}$$

while a typical cross entropy function is like:

$$Entropy(x) = -\tfrac{1}{n}\textstyle\sum^{n} \quad y \cdot ln\, ln\, \hat{y} + (1 - y) \cdot ln\, ln\, 1 - \hat{y} \,.$$

And if we add l2 penalty to our loss function, it will be:

$$L(\Theta) = -\frac{1}{n}\sum^{n} \quad y\cdot\hat{y} + (1-y)\cdot 1-\hat{y} + \frac{\lambda}{2n}\cdot\|w\|^2$$

So, the derivative of our loss function is:

$$\frac{\partial L(\Theta)}{\partial w} = \frac{\partial(-\frac{1}{n}\Sigma^{n} \quad y\cdot\hat{y} + (1-y)\cdot 1-\hat{y} + \frac{\lambda}{2n}\cdot\|w\|^2)}{\partial w}$$

$$= \frac{\partial(-\frac{1}{n}\Sigma^{n} \quad y\cdot\hat{y} + (1-y)\cdot 1-\hat{y}\,)}{\partial\hat{y}}\cdot\frac{\partial\hat{y}}{\partial w} + \lambda\cdot w$$

$$= \left(\frac{-\frac{1}{n}\Sigma^{n} \quad y\cdot\hat{y}}{\partial\hat{y}} + \frac{-\frac{1}{n}\Sigma^{n} \quad (1-y)\cdot 1-\hat{y}}{\partial\hat{y}}\right)\cdot\frac{\partial\hat{y}}{\partial w} + \lambda\cdot w$$

$$= \left(\frac{-y}{\hat{y}\, ln\, ln\, 2} + \frac{1-y}{(1-\hat{y})\, ln\, ln\, 2}\right)\cdot\frac{\partial\hat{y}}{\partial z}\cdot\frac{\partial z}{\partial w} + \lambda\cdot w$$

$$= \frac{\hat{y}-y}{\hat{y}(1-\hat{y})\, ln\, ln\, 2}\cdot\big(\hat{y}(1-\hat{y})\big)\cdot\frac{\partial z}{\partial w} + \lambda\cdot w = \frac{(\hat{y}-y)x}{ln\, ln\, 2} + \lambda\cdot w$$

and this function is used in our gradient descending process. The result of our build-from scratch model is shown in the following confusion matrix (on the left).

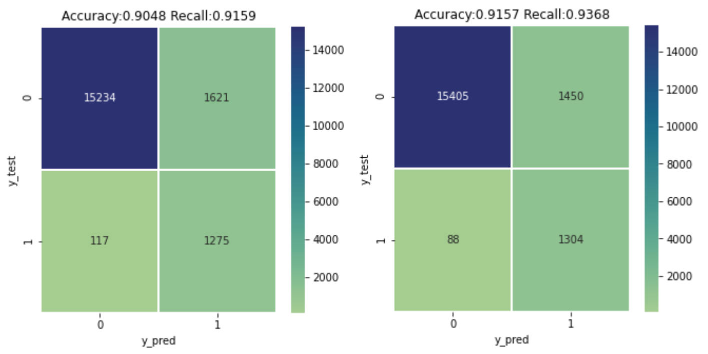


Figure 1 Accuracy of LR Model: Self-built & Sklearn

To compare the performance of our model, we also used the Logistic Regression class in sklearn package with the similar settings: cross entropy with l2 penalty, and the result of that model is as the confusion matrix above on the right.

By comparison, we can notice that the biggest advantage of package-built model is a quicker training process. Besides, the accuracy is slightly better, yet we can still believe that they are on the same level. Anyhow, through this process, we apply our knowledge to a case and give out a satisfying solution.

4.2 Decision Tree

Decision Tree is a strong and popular model for classification prediction. Different from Logistic Regression, Decision Tree is not sensitive to outliers and it can handle multicollinearity problem. However, it is easy for decision tree to exist overfitting if building the tree to max without setting restrictive conditions. There are two ways for addressing overfitting, one is pre-pruning and another one is post pruning. For this project, we chose pre-pruning to avoid the overfitting problem.

We chose the existing package sklearn to do the Decision Tree prediction. We set the measure of node impurity as Gini then build the tree to max, the accuracy for trainset is 96.81% while the accuracy for testset is 89.98%. In order to apply pre-pruning, we tested two parameters, one is max depth ranging from 1 to 20, another one is minimum samples leaf ranging from 1 to 20. Then we drew the diagrams showing the errors for train set and test set respectively. The most appropriate value for the parameter occurs at the point with lowest error of testset. After doing pre-pruning, the accuracy of testset increases from 89.98% to 90.75%, and the recall rate increases from 89.97% to 93.29%, which is a positive progress because when applying to death prediction, the accuracy for predicting label 1 is quite important.

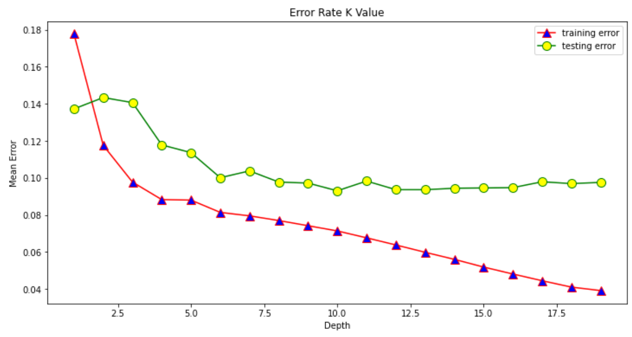
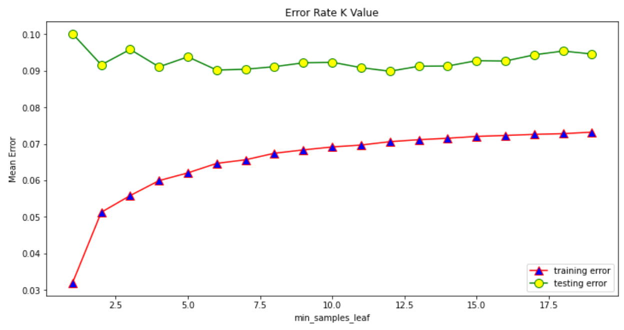


Figure 2 Relation of Errors & Max Depth



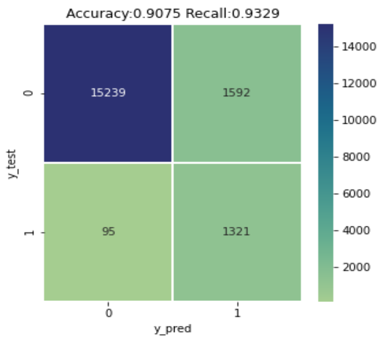


Figure 3 Relation of Errors & Min Sample Leaf Figure 4 Accuracy

4.3 Random Forest

The project used the Random forest model from SkLearn to facilitate the programming process. Random Forest can generate multiple individual trees parallelly and then make the majority vote to select the class to be the model prediction. Such an ensemble model can provide a more reliable result than a single decision tree because even though some of the trees are wrong, many others trees are still correct. Grid search is introduced in this project for hyperparameter tuning. Two parameters are tested, which are the max depth of the tree (ranging from 3 to 7), controlling the overfitting issue of trees, and loss function consisting of Gini and Entropy.

The result of the Random Forest can be shown as below, with the accuracy at 88.68% and recall rate at 95.31%.

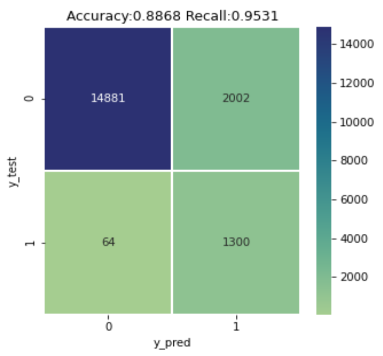


Figure 5 Random Forest Accuracy

Another reason for choosing a random forest model is that it provides a direct result of features importance. The result below showed the importance of icu_yn, hosp_yn and age group over 80. It means that if patients are sent to hospital or even to intensive care units, they are more likely to be predicted to be death caused by the virus. It makes sense because only people with severe symptoms would stay in hospital. In addition, the result also proves that people older than 80 are more likely to be harmed by the virus. It is suggested that the aged group should be protected carefully and should be prior to the covid-19 vaccine injection

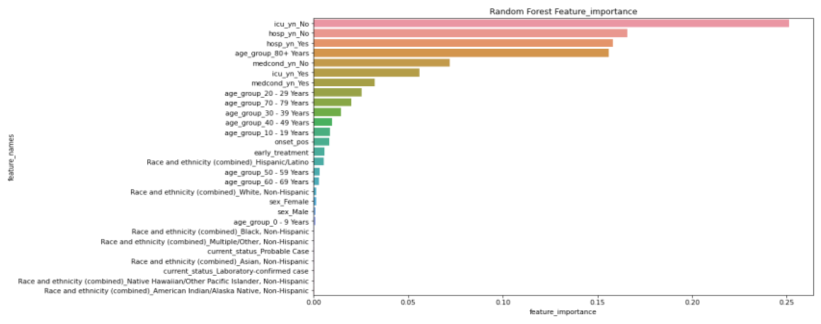


Figure 6 Feature Importance

4.4 SVM

Support vector machines were popular for two-group classification in the past. Kernel function used in the SVM from Sklearn makes our prediction more accurate. Hyperparameter tuning on the kernel function such as linear, rbf,poly,sigmoid is conducted by Grid Search. The prediction result is shown as below. The accuracy of SVM prediction on the test set is 90.17%, which is slightly higher than the random forest. However, the recall rate of SM was 92.32%.

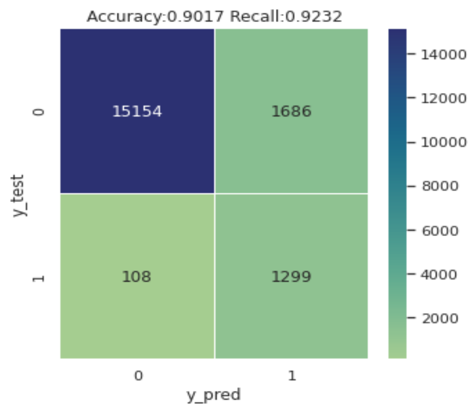


Figure 7 SVM Accuracy

5. Conclusion

In this project, we worked on COVID-19 Case Surveillance Data to map the relation between patient's death with 10 features, including intensive care unit, hospitalized status, sex, age, race and ethnicity, current status, early treatment status, and onset date. As we have enough labelled data, we kept 42956 records after we removed all the NAN values. Then we split the data into train-set and test-set at a ratio of 7:3 to ensure accurate model evaluation. To solve this classification problem, we build up 4 machine learning models from Sklearn package in python: Logistic Regression, Decision Tree, Random forest and SVM. GridSearch method is performed to find the best parameters. For Decision Tree, we choose pre-pruning methods and Gini index, max depth, minimum samples leaf are used as measures. For Random Forest, we try two parameters including max depth and loss function consisting of Gini and Entropy. For SVM model, hyperparameter tuning on the kernel function such as linear, rbf, poly, sigmoid is conducted by Grid Search. The best model is Random Forest given recall at 95.31%. Especially, we try to build up the Logistic Regression from scratch, which achieved recall at 91%. The final result shows intensive care unit is an especially important feature in death prediction, achieving importance value at 0.25, followed by hospitalized status and age status over 80 years, therefore we conclude that intensive care unit should be given special attention if aiming to reduce death rate.