COVID-19 Time to Hospitalization

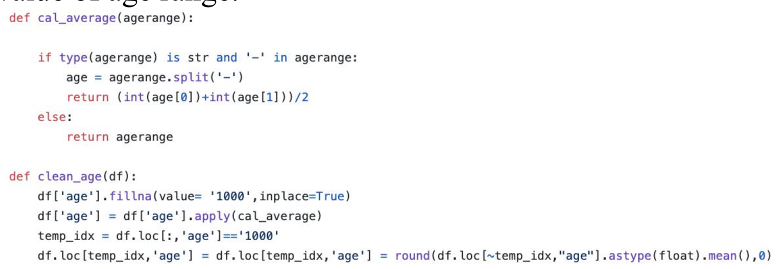
Introduction

This module is about prediction of the time between symptom onset and hospitalization (the *duration* of the time until hospitalization, in days) for confirmed hospitalized cases of COVID-19 in January and February, 2020. In this paper, in order to predict the duration (the time between symptom onset and hospitalization) for the patients in the testing set (in days, provided as integers), we started with modifying the data sets by filling missing data and converting categorical entries into numerical. Then, we tried a couple of models including linear regression, random forest and gaussian model to fit the data. By comparing the results we get from each model, we conclude that the Gaussian model fits the data best under this condition.

Feature Engineering

Before fitting the dataset, we need to modify some of the entries.

In the dataset, some age values are not accurate as they are shown as a range. Under this condition, we could use the mean of the range to replace the original value. Since there are also a few missing values in the age attribute, we could fill them in with mean. In addition, since age exists non-numeric values (age range), we need to convert them to numeric values. We choose to convert the age range to the mean value of age range.

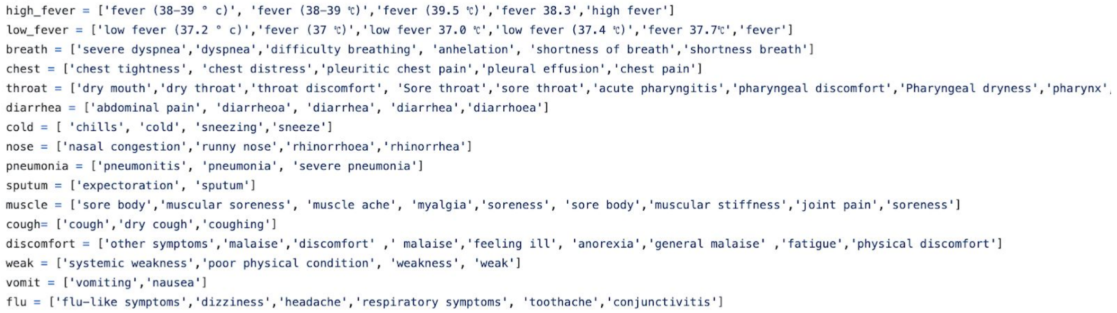


```
def cal_average(agerange):

    if type(agerange) is str and '-' in agerange:
        age = agerange.split('-')
        return (int(age[0])+int(age[1]))/2
    else:
        return agerange

def clean_age(df):
    df['age'].fillna(value= '1000',inplace=True)
    df['age'] = df['age'].apply(cal_average)
    temp_idx = df.loc[:,'age']=='1000'
    df.loc[temp_idx,'age'] = df.loc[temp_idx,'age'] = round(df.loc[~temp_idx,"age"].astype(float).mean(),0)
```

In addition, the **symptoms** are all strings separated by commas, which has no contribution to our data analysis. Therefore, we transform all symptoms to different groups of symptoms. We group similar symptoms into one specific group, and add the group as a new column. For patients with no symptoms, we simply added a new column called new symptom and assigned 1 to those who had no symptoms. We added 16 more columns to classify all symptoms. They are shown below.



```
high_fever = ['fever (38-39 ° c)', 'fever (38-39 ℃)','fever (39.5 ℃)','fever 38.3','high fever']
low_fever = ['low fever (37.2 ° c)','fever (37 ℃)','low fever 37.0 ℃','low fever (37.4 ℃)','fever 37.7℃','fever']
breath = ['severe dyspnea','dyspnea','difficulty breathing', 'anhelation', 'shortness of breath','shortness breath']
chest = ['chest tightness', 'chest distress','pleuritic chest pain','pleural effusion','chest pain']
throat = ['dry mouth','dry throat','throat discomfort', 'Sore throat','sore throat','acute pharyngitis','pharyngeal discomfort','Pharyngeal dryness','pharynx',
diarrhea = ['abdominal pain', 'diarrheoa', 'diarrhea', 'diarrhea','diarrhoea']
cold = [ 'chills', 'cold', 'sneezing','sneeze']
nose = ['nasal congestion','runny nose','rhinorrhoea','rhinorrhea']
pneumonia = ['pneumonitis', 'pneumonia', 'severe pneumonia']
sputum = ['expectoration', 'sputum']
muscle = ['sore body','muscular soreness', 'muscle ache', 'myalgia','soreness', 'sore body','muscular stiffness','joint pain','soreness']
cough= ['cough','dry cough','coughing']
discomfort = ['other symptoms','malaise','discomfort' ,' malaise','feeling ill', 'anorexia','general malaise' ,'fatigue','physical discomfort']
weak = ['systemic weakness','poor physical condition', 'weakness', 'weak']
vomit = ['vomiting','nausea']
flu = ['flu-like symptoms','dizziness','headache','respiratory symptoms', 'toothache','conjunctivitis']
```

In addition, we also transformed the **date** column. The date itself does not provide any useful information, therefore we transformed this column to the time interval between the confirmed date to the most earliest date, which is an integer value.

We have also altered the sex, city, province, country column. Instead of using labels, we used numbers to denote labels. This will make our regression analysis easier.

Data Modelling

We have used 3 models to fit our data. These models are linear regression model, random forest model and gaussian regression model. Overall speaking, the gaussian regression model performs the best with the smallest rmse of 4.27, therefore we only included the gaussian regression model in our final code submission. However we will talk about all three models here.

Linear regression

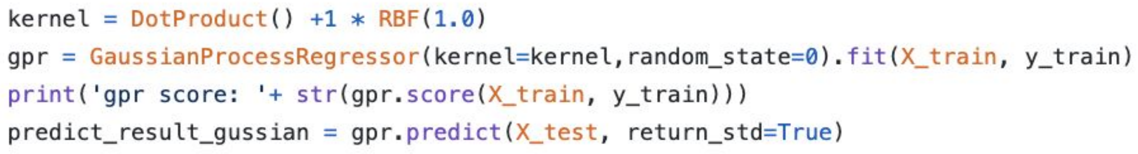
First we use the linear regression base stepwise to construct, and select the ordinary linear model with smallest AIC to implement the least square estimation. There use the residuals to follow a normal distribution. We can not determine the correctness of independent variables. In figure 2, there exists some outlier, normally the time between symptom onset and hospitalization lage than 30 days, There obviously exist unconfirmed elements. The outlier would be enormous affection to the linear model.

Random Forest

Then apply random forest to construct models, select important variables for objects. We choose City and province is null due to not clear tree nodes. The "tuneRF" to select the best number of variables for splitting at each tree node. Using the "train" to find the best ntree is around 800. Although with those updating info, the RMSE of the model around 2. That value is higher than we expected. We consider the following reason for this condition.The splitting nodes isn't clearly defined. Some variables are less relevant. In the random forest algorithm increase in the conversion tree, it can modify the tolerance of noise data, relieving some overfitting and obtain non-perfect stable results. But it can not achieve the best selection in this dataset.

Gaussian Regression Model

The gaussian regression model, it adapts well on the small dataset. The cleaning data has some uncertainty factors. The kernel function can produce a correlation coefficient matrix, if two parameters x are close together, like some symptoms of covid, then the correlation of the corresponding y (duration) is also higher. The function to fit the date and add uncertainty to the prediction results. The corresponding code is shown below.



```
kernel = DotProduct() +1 * RBF(1.0)
gpr = GaussianProcessRegressor(kernel=kernel,random_state=0).fit(X_train, y_train)
print('gpr score: '+ str(gpr.score(X_train, y_train)))
predict_result_gussian = gpr.predict(X_test, return_std=True)
```

Since the gaussian model performs the best on the dataset, we further improved our model by selecting features, tuning parameters as well as cross validation. The Gaussian regression model finally results in a rmse of 4.27, which is our best submission.

Conclusion

Since there are many attributes and many missing values but the size of the population is not large enough, therefore data preprocessing in this situation is difficult. The Gaussian regression model has the advantage to fit with condition.

The consideration of data preprocessing in this paper is simple, and we can continue to consider the degree of linear relationship between attributes, as well as the combination of attributes, or attribute splits.