First Draft – Classification Model COVID 19

Nachi Lieder
April 19th 2020

Preliminary Assumptions

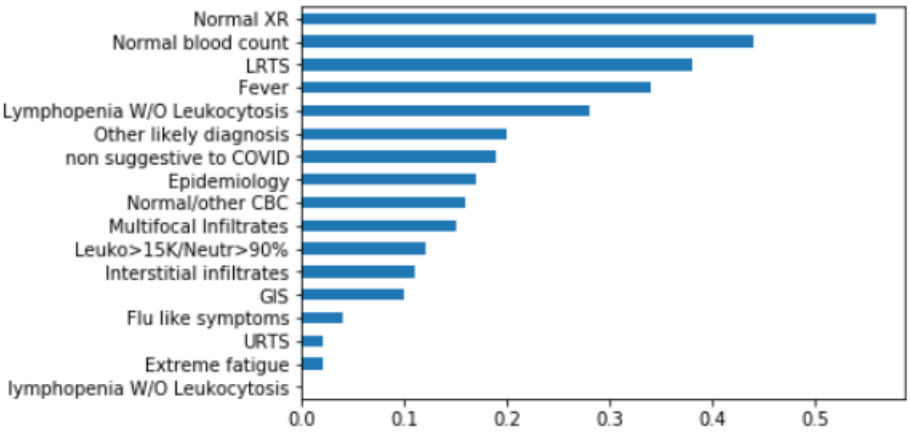
1. Our target variable is PCR , Meaning the objective is to predict whether the PCR was positive or negative.
2. The model needs to be released as quick as possible , which points to an easy implementation and integration if needed
3. Intuitive model , no use of Deep Learning methods , to simplify the model and make it easier to understand and utilize.
4. Adaptive model , the model will be adaptive to new data , and will re-train itself every period that we define to improve performance.
5. There is no meaning to the order of the patients in the dataset.

DATA EXPLORATION

There are 294 patients in the set , 38% of them are labeled with a positive PCR.

The model is based on 8 binary variables plus 2 categorical variables.

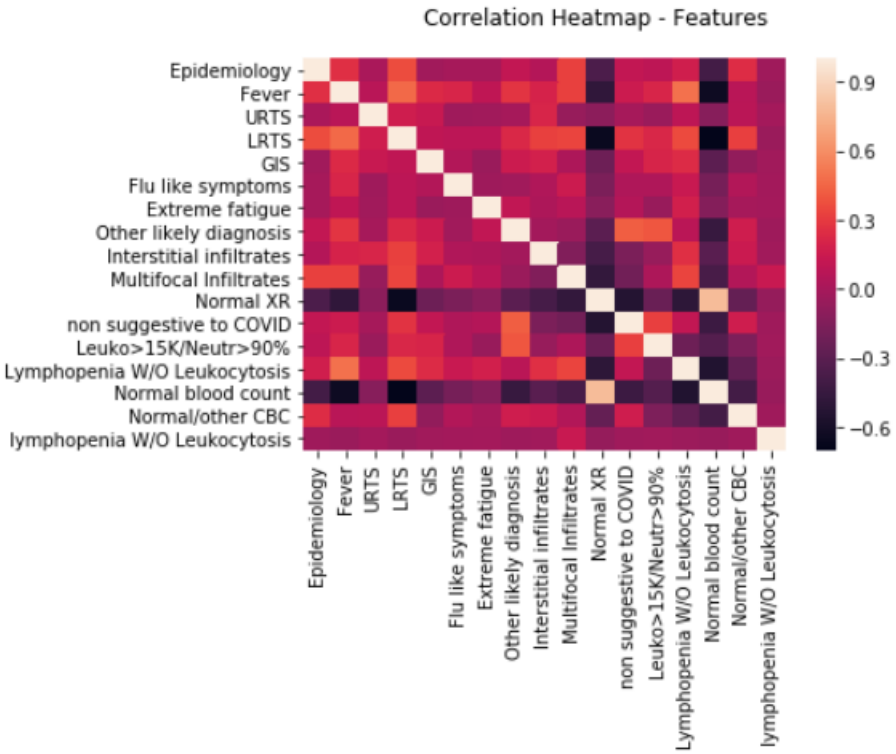
The variables and their distribution:



To create a more sophisticated model , and capture as much variance as possible , I decided to treat the categorical variables as categories and not pool them as a binary variable.
I created encoding - dummy variables to binarize each category within the variable.

HEATMAP Correlation Matrix

I decided to view the inner correlations between variables per patient . Example – Normal XR and Normal Blood Count have a positive correlation. This makes sense.
Also , there is a negative correlation between Multi Infiltrates and Normal XR. This too makes sense.



I decided to create a mutual effect of "And" between the binary variables to capture the inner correlation and variance. Example – We would have a feature called Epidemiology , and another feature called Fever. There will now be a new feature called Epidemiology_Fever which will indicate whether the patient had both symptoms.

Method of test

I split the sample into 2 sets. The first set is the training set , which pertains 70% of the samples. The second set is the Test Out Of Sample set (OOS) , which pertains the remaining 30%.
The training set is built to train the model , while the OOS is to test the performance of the test on a set that hasn't been seen by the model before. This keeps away from overfitting.
In later stages , I performed a cross-validation of the entire set , splitting the entire set into 5 random sets , training the model on every 4 , and testing on the 5^{th}. This was to ensure I wasn't overfitting or wasn't biased towards a certain subset of the larger dataset.

Metrics

I didn't settle on one metric since each metric has its pros and cons, and needs to be taken under consideration while evaluating the models performance.

- **AUC and ROC curve** – Scoring method which measures different thresholds and presents the different outcomes per threshold.

- **Precision Score** – The ability to measure how many positive results relative to all the results I was correct with. Sensitivity

- **Recall Score** - The ability to measure how many positive observation I was correct with , from all the positive predictions my model projects.
- **F1 Score** – joined metric taking into account precision and recall
- **Accuracy** – Percentage of correct labeled observations from set of predictions
- **Confusion Matrix** – Distribution of predictions in a formatted matrix.

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

The Models

I decided to go forward with 2 models, and test them both against the current existing classifier.

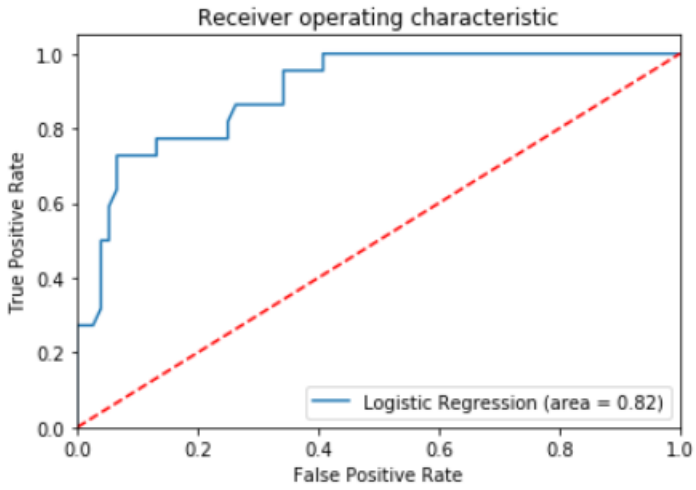
RANDOM FOREST
This model is a decision tree based model. The machine creates multiple trees and creates a voting system . All the trees individually classify a given observation , and through a vote , the decision is made. The fact that it is a "forest" and not a single tree makes the model much more robust for a minor error. This model is used commonly.

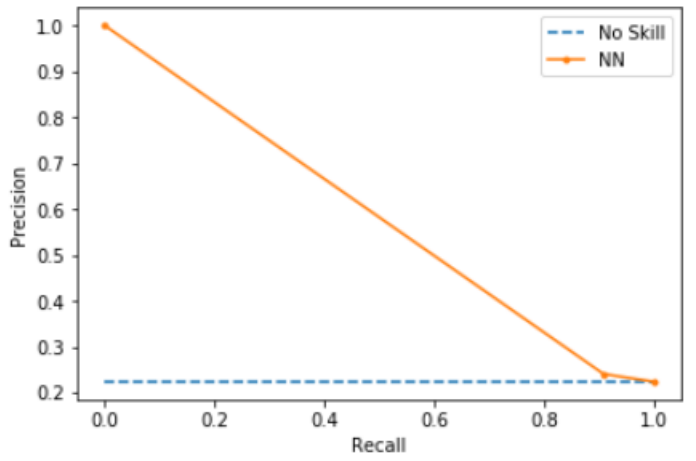
Logistic Regression
The logistic regression is fast , easy and has proven its effectiveness in the ML industry. Using the given features , the model tries to create a sigmoid function that will optimize the differentiation between classifying 0 and 1.
The default is that the model will classify f(x)>0.5 -→ True , else False. Though we can manipulate the thresholds and decide according to the strictness of each type of error , what our thresholds will be and work with those going forward.

Score to PCR matching – Current Classifier

I decided to use the current classifier as my benchmark.

```
Results from Score to PCR matching
f1=0.381 auc=0.585
Recall Score=0.909
Precision Score =0.241
Accuracy: 0.34
RF: confusion_matrix =
[[13 63]
 [ 2 20]]
```

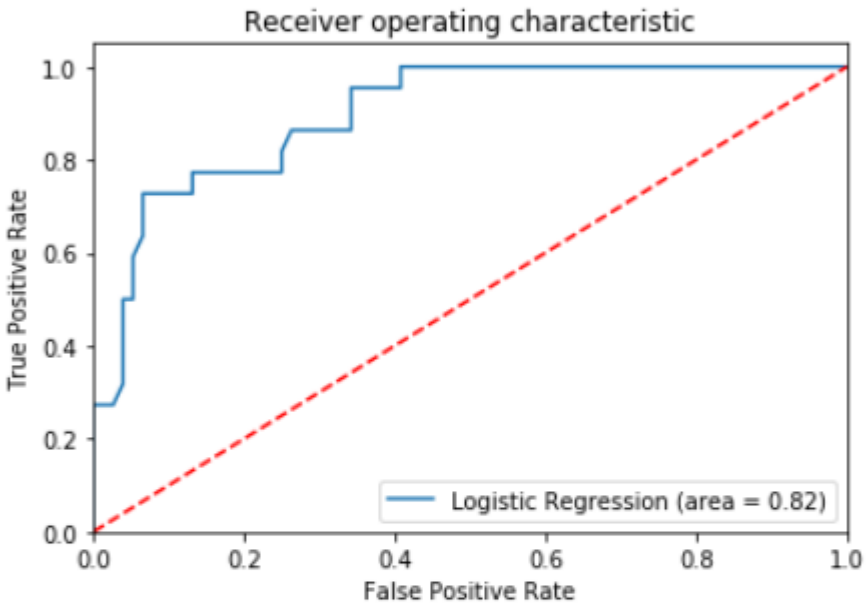


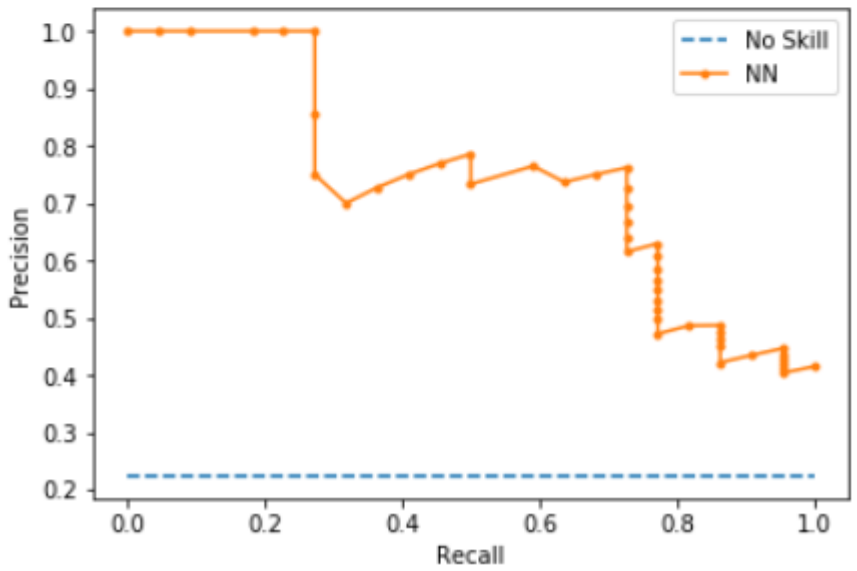


Against this , I compared my two different models , with hope that one will satisfy all the different criteria of the client.

Model 1 - RANDOM FOREST:

```
Results from simple Random Forest
RF: f1=0.727 auc=0.742
RF: Recall Score=0.727
RF: Precision Score =0.727
Accuracy: 0.88
RF: confusion_matrix =
[[70  6]
 [ 6 16]]
```





Model 2 – Logistic Regression

```
Results from simple Logistic Regression
LR: f1=0.650 auc=0.676
LR: Recall Score=0.591
LR: Precision Score =0.722
Accuracy: 0.86
LR: confusion_matrix =
[[71  5]
 [ 9 13]]
```