Covid-19 Diagnostics Project

Christopher Weaver

August 26th, 2021

1. Introduction

1.1 Background

The past couple of years have presented us with a unique opportunity to utilize increasingly accessible hardware and sophisticated software to enhance individual health outcomes and knowledge. Smart phones continue to increase, not only in computational power, but also in the array of sensors these devices contain. With the release of the Apple Watch Series One, Apple made a marketing and engineering commitment to empowering users to monitor their own health through technology. In recent years, wearables such as the Apple Watch have been fitted with sensors to help users monitor heart rate, blood oxygen levels, and even check for cardiac arrhythmia. As these sensors become more prevalent, it opens up opportunities for software engineers to find new ways to use these sensors. Monitoring heart disease, physical activity, and environments that can lead to hearing loss have already taken shape; but there is more that can be done. Companies such as Cardiogram are banking on just that. Research conducted in partnership with the University of California found that utilizing LSTM neural networks and "off the shelf" wearable heart rate sensors, medical conditions such as diabetes, high cholesterol, high blood pressure, and sleep apnea could all be detected with high accuracy[1]. I propose that similar techniques can be extended beyond just these diseases into a whole host of other ailments and conditions.

1.2 Problem

In the spring of 2020, individual cities, counties, and states began implementing restrictive mitigation strategies around the novel corona virus SARS-C0V-2. Many feared either contracting or spreading of the virus and took to quarantining at home for

[1] AAAI Conference on Artificial Intelligence, Feb 2018 (AAAI-18). https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16967/15916

weeks or months at a time. Many institutions scrambled to find ways to resume semi-normal activity while at the same time keeping all involved safe and comfortable. Different testing solutions were rolled out as a mechanism for helping individuals know if they were infected with the virus so that they could make better informed decisions. But many of these tests are slow, expensive, inconvenient, and too inaccurate for individuals to take on a regular basis. What is needed is an easier way to test and monitor for Covid-19 that empowers more people to make responsible health decisions. The goal is is to build a classification machine learning model that can accurately diagnose Covid-19 using bio-metrics gathered either by api's to Apple's Health app or from user input to specific questions such as "Are you currently experiencing a cough?". Inputs would include numeric values (body temperature, pulse, and Sp02) to output a class output of either 1 or 0, with 1 representing a Covid-19 diagnosis and 0 representing no indication of infection with Covid-19. A further model would be developed which takes binary classification values such as the existence of a cough, sore throat, and headache and again outputs a class value of 1 or 0, with 1 representing a Covid-19 diagnosis and 0 representing no indication of infection with Covid-19. Each feature in the second model will be represented as a 1 if the user is experiencing the symptom or a 0 if they are not.

2. Data Acquisition and Cleaning

2.1 Data Sources

After more than a year and a half of scientific scrutiny over Covid-19, we now have a pretty good idea how the virus impacts individual human health and the symptoms most common to be presented. The first step to finding a solution to the problem statement above is a good data set. I will be utilizing two different datasets to help tackle the issue. The first dataset contains biometric data usually referred to as vitals, such as temperature, pulse rate, and Sp02 levels[2]. Each row of the dataset contains one measurement for each and a binary classification column for either currently being infected with Covid-19 or not at the time of the measurement. Eventually I change Result to 1 for positive cases and 0 for negative cases.

	ID	Oxy	Pulse	Temp	Result
0	0	98	65	95	Negative
1	1	96	92	95	Negative
2	2	95	92	99	Negative
3	3	97	56	96	Negative
4	4	88	94	98	Positive

	ID	Oxy	Pulse	Temp	Result
0	0	98	65	95	0
1	1	96	92	95	0
2	2	95	92	99	0
3	3	97	56	96	0
4	4	88	94	98	1

[2] https://www.kaggle.com/rishanmascarenhas/covid19-temperatureoxygenpulse-rate

The second dataset to be utilized contains many more biometrics in binary classification form. These include cough, muscle aches, tiredness, and sore throat, among many others[3]. Each value is represented as a 1 if the user had the symptom and a 0 if they did not. Each row also contains a value for diagnostics which could be Flu, allergy, cold, or Covid-19.

DIFFICULTY_BREATHING	LOSS_OF_TASTE	LOSS_OF_SMELL	ITCHY_NOSE	ITCHY_EYES	ITCHY_MOUTH	ITCHY_INNER_EAR	SNEEZING	PINK_EYE	TYPE
0	1	0	1	0	0	1	0	1	ALLERGY
0	1	0	1	0	1	1	1	1	ALLERGY
0	1	0	1	0	0	0	0	1	ALLERGY
0	1	1	0	0	1	0	1	1	ALLERGY
0	1	1	0	1	0	1	1	1	ALLERGY

Preliminary investigation into this dataset shows that it does not possess an even distribution among the possible target values. The Flu is significantly higher than any other target. The Flu and Allergy make up more than 90% of all values for our target. Our unbalanced dataset could result in a model that looks very accurate, but does poorly when it comes to predicting Covid-19 as the model found learning about this target not relevant to obtaining a low error rate.

```
FLU        25000
ALLERGY    16381
COVID       2048
COLD        1024
Name: TYPE, dtype: int64
```

I originally ended up choosing 1024 records from each Type for training but found that my models found differentiation between Covid-19 and the Flu very difficult. So I changed this initial modification to 2048 records per case (except for the cold which remained at 1024). This gave my models the data it needed to better handle Covid-19 and Flu cases.

Furthermore, for the second dataset I found that each disease presented with the same 8-10 symptoms every time. This makes dimensionality reduction simple as all I had to do was find any symptom that presents itself in at least on of the four diseases, but not in all four. This works because features are binary and all symptoms present nearly uniformly for a disease which means there is little to no information gain from using a symptom that displays in all four diseases.

3. Exploratory Data Analysis

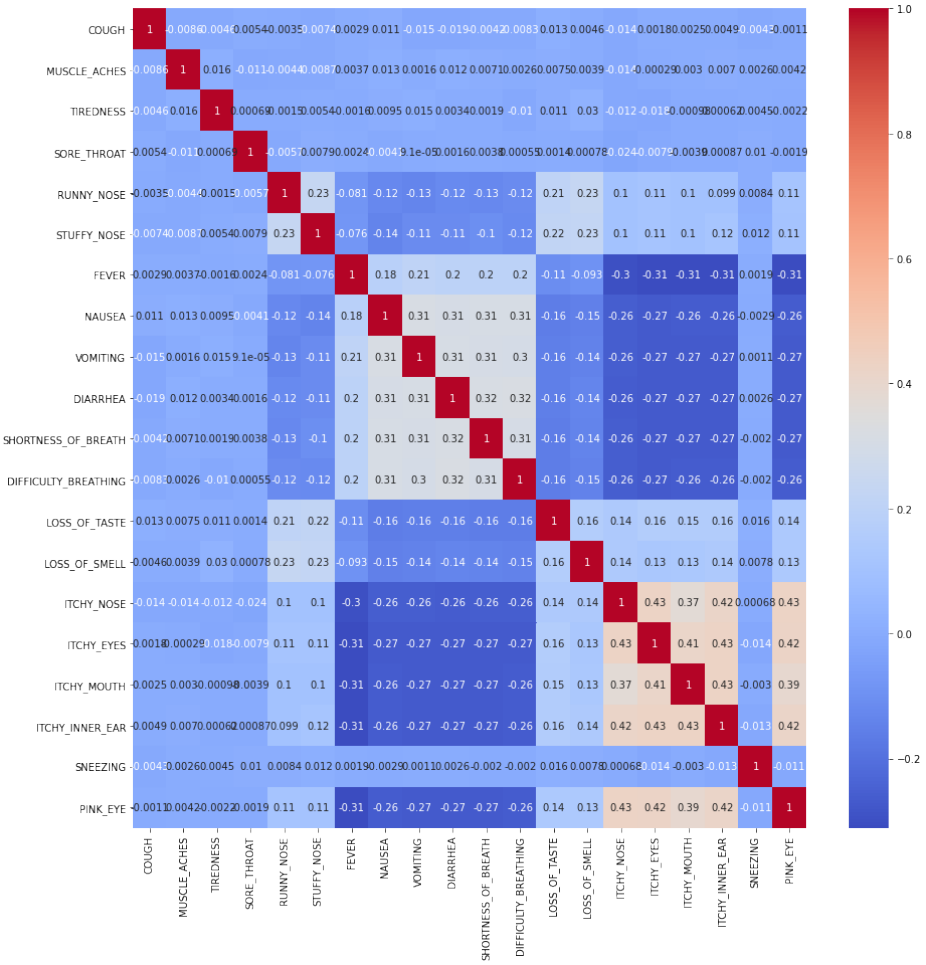
[3] https://www.kaggle.com/walterconway/covid-flu-cold-symptoms

3.1 Correlation

Initial analysis showed to my surprise little correlation between different vitals on our first dataset. I would have expected some correlation between Sp02, Pulse, and Temperature. Despite this lack of a relationship, we do find a high correlation between Sp02 and positive tests for Covid-19. We also see a modest relationship between temperature and positive tests for Covid-19.

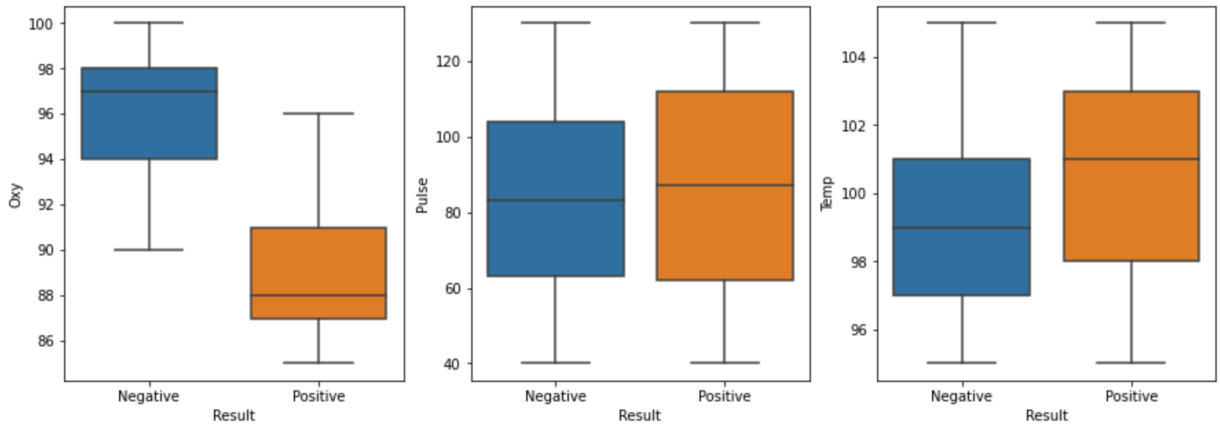
	ID	Oxy	Pulse	Temp	Result
ID	1.000000	-0.002693	-0.012499	-0.009279	0.011395
Oxy	-0.002693	1.000000	-0.005724	-0.015681	-0.777019
Pulse	-0.012499	-0.005724	1.000000	0.009602	0.050590
Temp	-0.009279	-0.015681	0.009602	1.000000	0.262522
Result	0.011395	-0.777019	0.050590	0.262522	1.000000

We see some correlations in the second dataset between different itchy body parts as well as some weak correlation between Fever, Nausea, Vomiting, and Diarrhea.

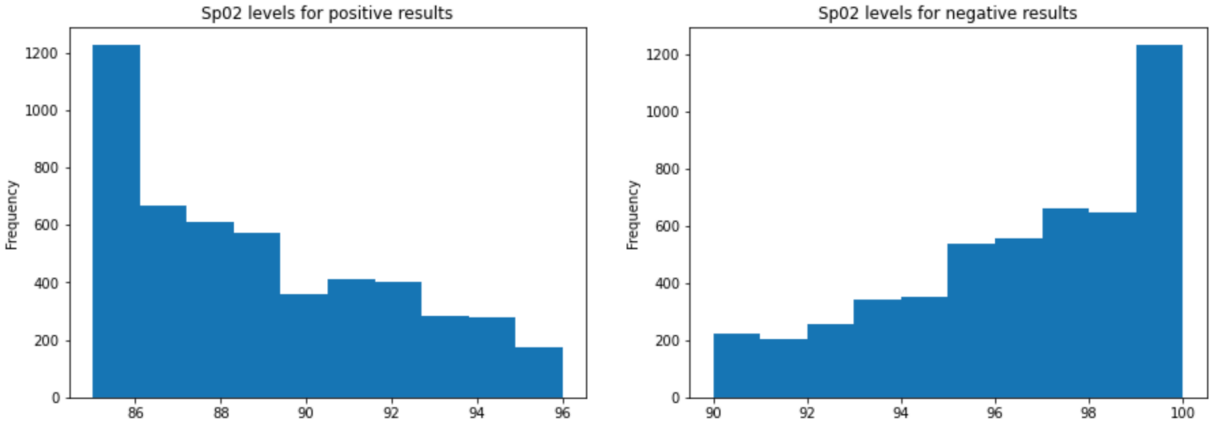


3.2 Further Data Analysis

For our first dataset, we see that Sp02 levels show the largest disparity between positive and negative Covid-19 cases.



Looking at distributions for Sp02 for both positive and negative Covid-19 cases shows values skew towards extremes but no definitive demarcation line exists.

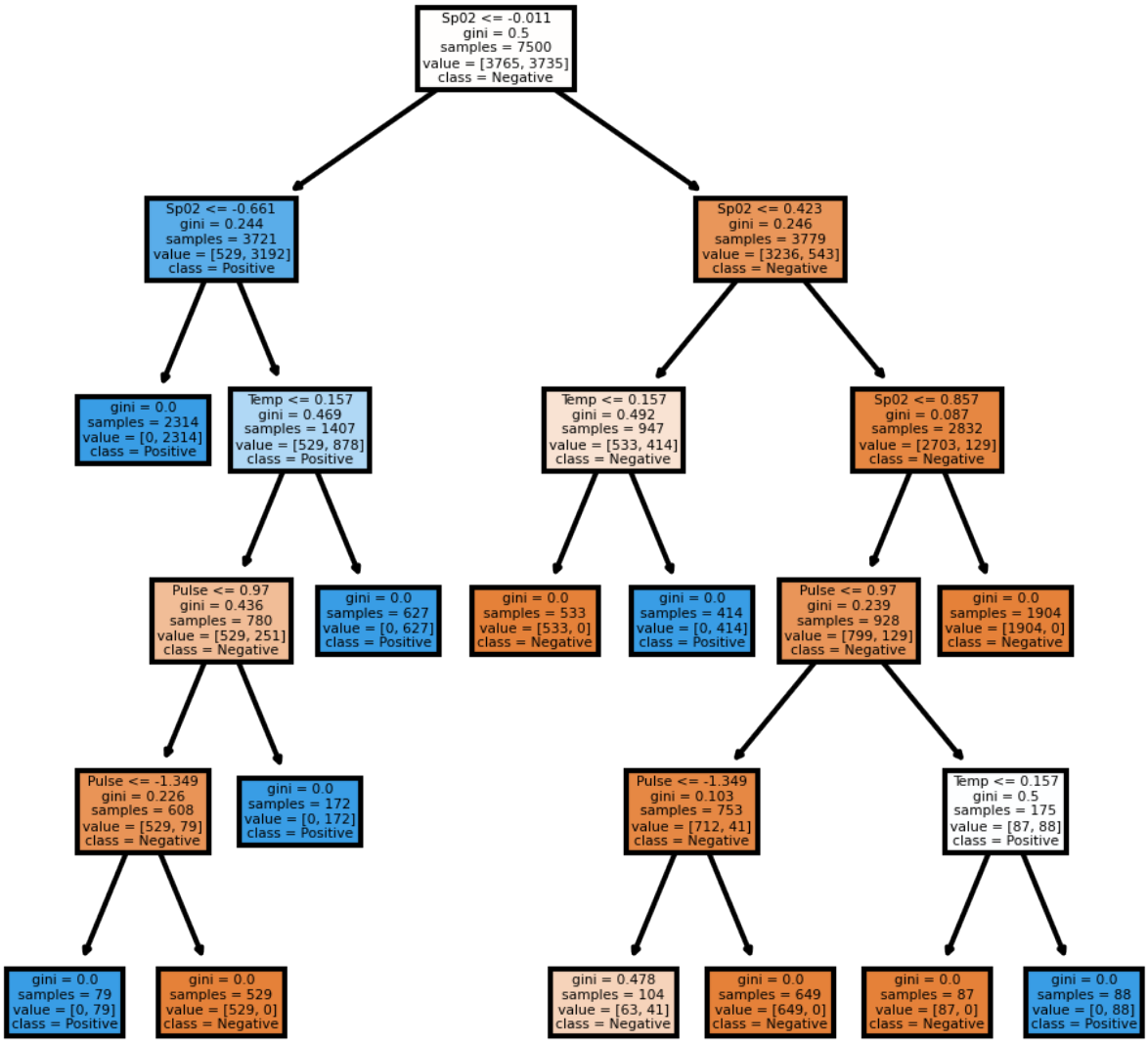


4.0 Predictive Modeling

4.1 Vitals Dataset - All Features

For our first dataset, I ended up training two different models. The first model I used was a decision tree using Sp02, pulse, and temperature as my features. Because we are not dealing with a complicated feature set and because Sp02 showed decent variability between positive and negative Covid-19 cases, I thought a decision tree

might be a good place to start. One of the advantages of the decision tree is its easy to understand how the model came to its conclusion. We can evaluate the model itself by graphing it out and gaining further insights into how Covid-19 presents itself in those infected with it. Below is the decision tree I trained graphed out.



The model ended up have an accuracy score of 100% on both the train and test dataset.

I also trained a logistic regression model as a secondary model. Logistic regressions are very simple and work for our case since we are only looking for a binary classification of positive or negative for Covid-19. There are two advantages of this model over a decision tree. The first is that it is less likely to overfit the data. As I mentioned above, the decision tree model was 100% accurate which suggests it might not generalize well in the real world. The second benefit is the logistic regression model

can output a predictive probability for both positive and negative outcomes which might be helpful for us later. The logistic model I tried had an accuracy score of 92% for both the training and test dataset. While this is lower than the decision tree, it is in the sweet spot of being high without initially indicating an overfitting of the dataset.

4.2 Vitals Dataset Limited Features

Part of the purpose of this project is to design an iOS mobile app that can monitor Covid-19 without much needed from the end user. An Apple Watch series 6 can monitor pulse and Sp02 levels throughout the day, but not temperature. Temperature readings would require user input to achieve. Let's see what our models would look like if we proceeded without temperature as a feature. Due to the high correlation between Sp02 and Result, we might be able to get away with this sort or dimensionality reduction.

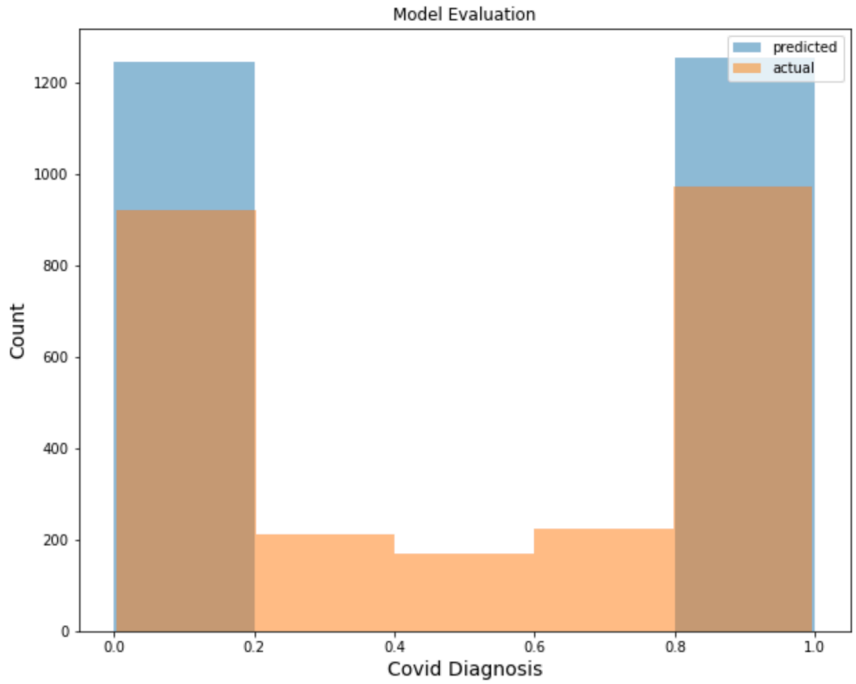
I next attempted to train both my decision tree model and logistic regression model using only Sp02 and pulse features. Both models ended up have an accuracy of 85% on both the train and test dataset. Later on, I decided this limited feature dataset would be more ideal for my iOS app, so focused my more detailed analysis of my modeling on these two. I first evaluated my decision tree model and found the precision was 87%, but recall was only 84%

```
Precision: 0.8716049382716049, Recall: 0.8438247011952191, fscore: 0.8574898785425101
```

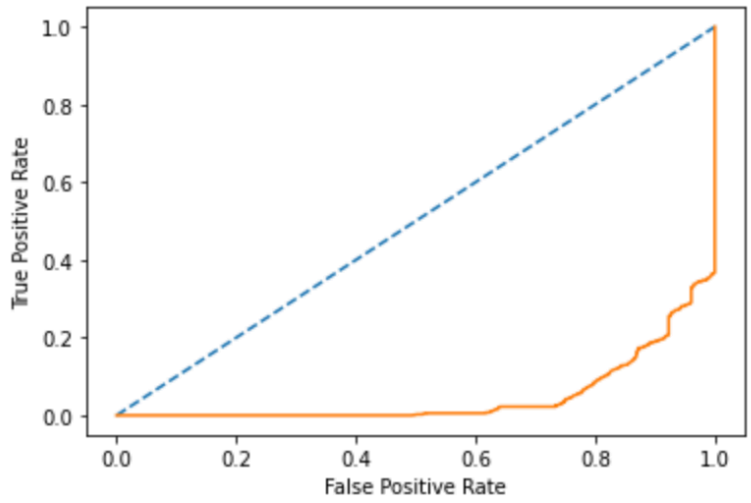
But, for my logistic regression model, If I change the threshold of what is a positive case to only those where the model is higher than 75% certain the user does not have Covid-19, then my recall is as high as 94%. Although I do have a tradeoff in that my precision is 7% lower for the logistic regression model over the decision tree, this is a welcome tradeoff for the task at hand. False negatives are much more serious than a false positive as a false negative lulls users into potentially not seeking treatment and acting in ways that may infect others.

```
Precision: 0.8075085324232082, Recall: 0.9426294820717132, fscore: 0.8698529411764705
```

For this reason, I decided to roll with the logistic regression model for my iOS app which I discuss later. Plotting model predictions against actual results shows the model tends to be relatively confident of its finding with few predictions with low confidence intervals.



Our ROC chart shows what we expected, many more false positives create the inverse elbow shown below.



4.3 Expanded Dataset

Our second dataset has many more binary classification features to work with, but also has four classes to predict between as an output. The extra dimensions we have in this

dataset over the first made me decide to use a random forest as my first model over a decision tree. I found using 5 estimators produced the best outcomes. Our confusion matrix shows perfect precision for allergy and flu, with high precision for cold and Covid-19. Most importantly, recall for all diseases other than the flu are very high.

```
{'ALLERGY': 0, 'COLD': 1, 'COVID': 2, 'FLU': 3}
[[230   4   0   0]
 [  0 137   0   0]
 [  0   4 205   0]
 [  0   9  19 221]]
              precision    recall  f1-score   support

           0       1.00      0.98      0.99       234
           1       0.89      1.00      0.94       137
           2       0.92      0.98      0.95       209
           3       1.00      0.89      0.94       249

    accuracy                           0.96       829
   macro avg       0.95      0.96      0.96       829
weighted avg       0.96      0.96      0.96       829
```

Our problem though is this confusion chart is using our test data from our balanced dataset. If we instead take a look at the entire dataset, I find that our precision for the cold and Covid-19 are very low. The recall is still very high for both, but the model many times predicted Covid-19 when it was actually the flu. Likewise, precision for the cold was very low.

```
{'ALLERGY': 0, 'COLD': 1, 'COVID': 2, 'FLU': 3}
[[15870   511     0     0]
 [    0  1024     0     0]
 [    0    64  1984     0]
 [    0   461  1973 22566]]
              precision    recall  f1-score   support

           0       1.00      0.97      0.98     16381
           1       0.50      1.00      0.66      1024
           2       0.50      0.97      0.66      2048
           3       1.00      0.90      0.95     25000

    accuracy                           0.93     44453
   macro avg       0.75      0.96      0.81     44453
weighted avg       0.97      0.93      0.94     44453
```

Next I attempted a support vector machine. I thought a svm might work well since they tend to be more effective in high dimensional spaces. My best results came from an

‘rbf’ kernel with a high C=1.0. Similar to the random forest, this model does very well with the test dataset from the balanced subset.

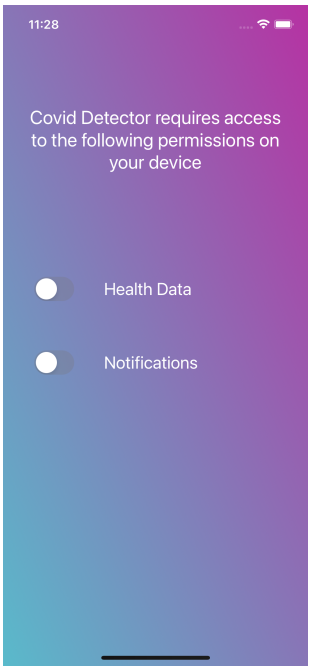
```
{'ALLERGY': 0, 'COLD': 1, 'COVID': 2, 'FLU': 3}
[[230   4   0   0]
 [  0 137   0   0]
 [  0   4 205   0]
 [  0   9  19 221]]
              precision    recall  f1-score   support

           0       1.00      0.98      0.99       234
           1       0.89      1.00      0.94       137
           2       0.92      0.98      0.95       209
           3       1.00      0.89      0.94       249

    accuracy                           0.96       829
   macro avg       0.95      0.96      0.96       829
weighted avg       0.96      0.96      0.96       829
```

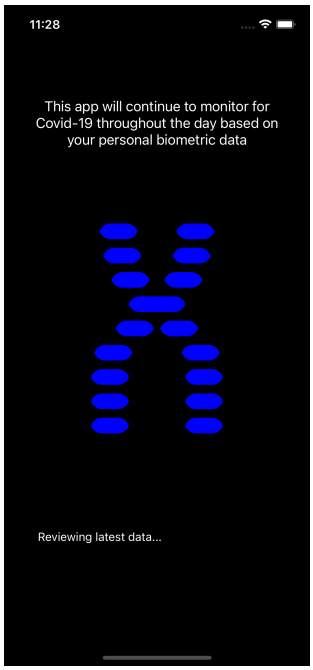
But once again, when we run our model against the entire unbalanced dataset, we find that precision is significantly lower for Covid-19 and Cold.

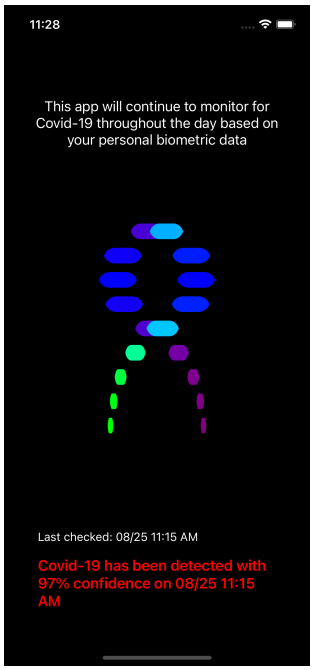
```
{'ALLERGY': 0, 'COLD': 1, 'COVID': 2, 'FLU': 3}

[[15870   511     0     0]
 [    0  1024     0     0]
 [    0    64  1984     0]
 [    0   452  1937 22611]]

              precision    recall  f1-score   support

           0       1.00      0.97      0.98     16381
           1       0.50      1.00      0.67      1024
           2       0.51      0.97      0.66      2048
           3       1.00      0.90      0.95     25000

    accuracy                           0.93     44453
   macro avg       0.75      0.96      0.82     44453
weighted avg       0.97      0.93      0.94     44453
```

This is important because in the real world we should not expect a balanced set of end users. The good news is our recall remains very high for Covid-19, which is most important as it helps users identify with a high level of confidence that they do not have Covid-19 and can act accordingly.

5.0 Deployment







I went ahead and built a quick iOS app that runs in the background throughout the day and queries measurements taken by an Apple Watch. These measurements are run through the logistic regression model built from the limited feature vitals dataset. The model was initially build and trained using Scikit-learn and was then converted to something an iOS app can run using CoreML. The iOS app will send a push alert to the user if it is less than 75% confident that the user does not have Covid-19. The project can be seen here: **https://github.com/crweaver225/Covid-Monitor**. Further video demos can be found here (https://youtu.be/FoL-Q-pcomY, https://youtu.be/oH9bZr6CKja).

A further step to make the app better would be to integrate our second model with the more elaborate feature set as well. Essentially, if this first model detects Covid-19, notify the user and have them answer a bunch of yes or no questions. These questions would match our feature set of symptoms and could be run through our second model to help verify a diagnosis of Covid-19.

6 Conclusion

Based on our datasets, I find that I can produce models that are highly accurate at keeping a user informed about their Covid-19 status. My models had a tendency to predict Covid-19 when the user did not in fact have it, but rarely were the models missing Covid-19 when the user was infected. This matches with the original project proposal as the worst case scenario is a false sense of security that may prevent a user from taking further precautions from spreading the virus to others or seeking healthcare if in a vulnerable outcome group. While false positives are bad, in our case they may

simply require a user to seek further testing or guidance from a health professional for clarification.

There are some import caveats that need further evaluation and consideration. The biggest one is that I do not know how our vital dataset was built or what criteria was used to classify positive and negative Covid-19 infections. For example, PCR tests are notorious for being relatively unreliable indicators for diagnostics. They have a high tendency for false positives due to the nature of looking for virus fragments. Bad criteria for our dataset means our models themselves will be highly inaccurate in the real world. We might find with further testing that our models perform very poorly due to issues with the dataset. But given the limited datasets that we have, our models perform very well. Considering how easily these models can be executed on an iOS app without much input from the end user, a lower standard of accuracy is probably acceptable as a first line of defense.