# SmartSwabs: Predicting COVID-19 Prevalence Rates to Optimize Grouped Testing

Researcher: Sahiba Kaur
H. H. Dow High

Table of Contents

## 1. Abstract

In this data science project, we combine the use of three datasets to train a machine learning model to predict prevalence rates of COVID-19 from demographic data and apply this real-world data to analyze testing algorithms. We cleaned and merged the datasets such that we were able to classify the **prevalence** of COVID - the number of people with the virus divided by the total population for each county grouped per month. Machine Learning algorithms were tested and a Principal Component Analysis was applied to the dataset to see how demographics impact COVID predictions. Finally, these prevalence estimates were directly applied into Dorfman's two-stage pooling algorithm for grouped testing. Results show that for every point in the dataset, grouped testing is more cost-efficient and sustainable and regular, individual testing.

## 2. Introduction - Data Science Life Cycle

Our datasets include *pop_est_ny.xlsx*, *us-counties-2021.csv,* and *cov_est_ny* from sources such as the CDC [2]. We first **merge** the pop_est_ny and cov_est_ny datasets on the county column, matching up the population of NY counties to their demographics. Then, we create a new column using the current county column and the us_counties_2021 dataset for the population of the counties. We call this the prevalence (*p)*, which is how common COVID-19 is in a specific area.

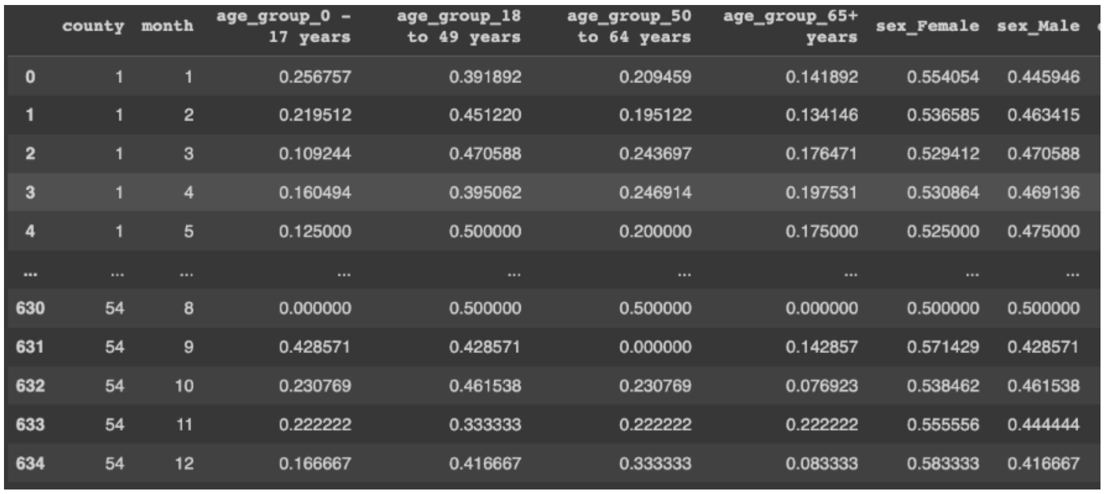
$$p = (number\ of\ cases\ in\ a\ county)/(population\ per\ county\ in\ 2021)$$

We had to clean the data by choosing which columns in our dataset to keep and drop, as there were some rows with null values (blank). The columns we decide to keep are:

*Age_group, sex, exposure_yn, symptom_status, county, case_month*

Initially, there were 236,695 data points, and we are left with 227,029 - which represents 95.916% of the data points.

Additionally, we implemented **one-hot encoding** by giving each county a specific number. This is useful for representing categorical data as numerical values (0s, 1s, 2s, etc.) For our dataset, we further split *age_group* into subcategories of 0-17, 18-49, 50-64, and 65+ years and label each as a number from 0-3, respectively. We continue one-hot encoding for each column of the dataset. We additionally make a county dictionary by replacing each county name with its respective number from 1-54, as shown below. We have now prepared our dataset to analyze it and implement various models.



|  | county | month | age_group_0 - 17 years | age_group_18 to 49 years | age_group_50 to 64 years | age_group_65+ years | sex_Female | sex_Male |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0.256757 | 0.391892 | 0.209459 | 0.141892 | 0.554054 | 0.445946 |
| 1 | 1 | 2 | 0.219512 | 0.451220 | 0.195122 | 0.134146 | 0.536585 | 0.463415 |
| 2 | 1 | 3 | 0.109244 | 0.470588 | 0.243697 | 0.176471 | 0.529412 | 0.470588 |
| 3 | 1 | 4 | 0.160494 | 0.395062 | 0.246914 | 0.197531 | 0.530864 | 0.469136 |
| 4 | 1 | 5 | 0.125000 | 0.500000 | 0.200000 | 0.175000 | 0.525000 | 0.475000 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 630 | 54 | 8 | 0.000000 | 0.500000 | 0.500000 | 0.000000 | 0.500000 | 0.500000 |
| 631 | 54 | 9 | 0.428571 | 0.428571 | 0.000000 | 0.142857 | 0.571429 | 0.428571 |
| 632 | 54 | 10 | 0.230769 | 0.461538 | 0.230769 | 0.076923 | 0.538462 | 0.461538 |
| 633 | 54 | 11 | 0.222222 | 0.333333 | 0.222222 | 0.222222 | 0.555556 | 0.444444 |
| 634 | 54 | 12 | 0.166667 | 0.416667 | 0.333333 | 0.083333 | 0.583333 | 0.416667 |

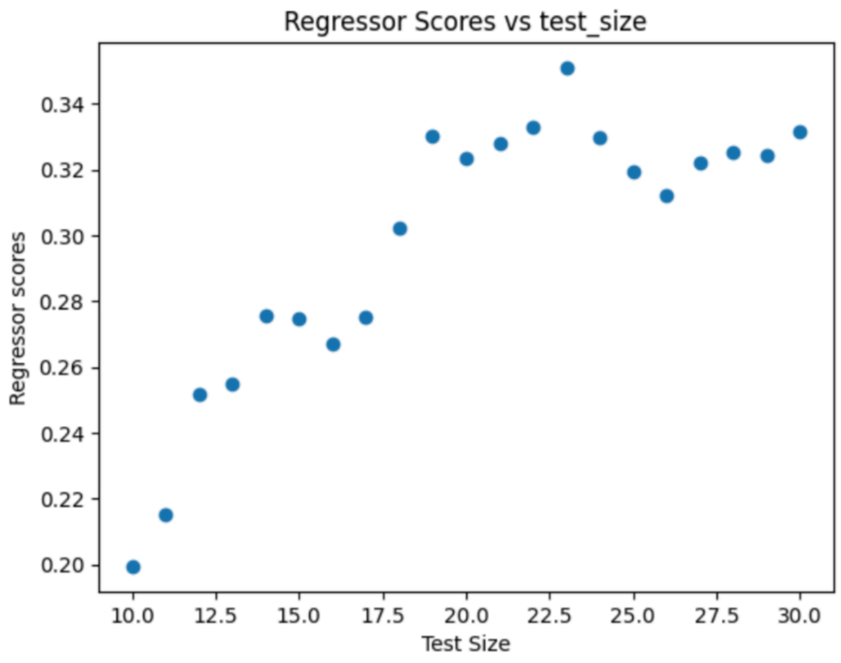
## 3. Machine Learning Models

### 3.1 Linear Regression

A **regression** problem has a *numeric* target variable where we determine the line of best fit.We need to split the data into **training** and **testing**. Training is used to fit the model, so it learns the relationship with features and the target. The test is used to evaluate the performance of the model.

**Supervised Machine Learning** is used when the training data has labeled target data, so we already know the output we want between 0 and 1 (in this case, prevalence).

We applied a simple Machine Learning code from *sk-learn* [3] , and defaulted our train-test split to 80-20, which gave 32.365% accuracy. Since we had low accuracy, we used a for-loop to iterate through various percentages of testing data from 10 to 30 (from 90-10 to 70-30 splits) to determine what the ideal train-test was. Additionally, we had to add the parameter *random_state=10* when we called the linear regression in order to prevent fluctuation from the model. Below is a graph of the test size on the regressor scores:



The optimal value of test-size is *0.23*, with an accuracy of *35.098%*. Further evaluation metrics such as *MAE*, *MSE*, and *RMSE* are as follows:

MAE (Mean Absolute Error): 0.005

MSE (Mean Squared Error): 3.497

RMSE (Root Mean Squared Error): 0.006

Overall, linear regression is unable to obtain accuracy for our project's goals. Another iteration of this model was an attempt at a Random Forest Classifier.

## 3.2 Random Forest

A Random Forest Classifier is another supervised machine learning algorithm used for solving classification problems, meaning a categorical target variable. It combines predictions from other models, where each of the smaller models is a decision tree, hence becoming a forest.

Since our current dataset is numerical, we split the *p* (prevalence) column into 4 quartiles from Q1-Q4 and *pop_density* into 4 categories from 1-4. In the end, the Random Forest will predict the quartile of *p*, not the specific value. However, in many applications of predicted prevalence such as pooled testing mentioned in section 2.1 of our paper, we only need an estimated range of prevalence and not the exact value. Following are the quartile ranges for *p*:

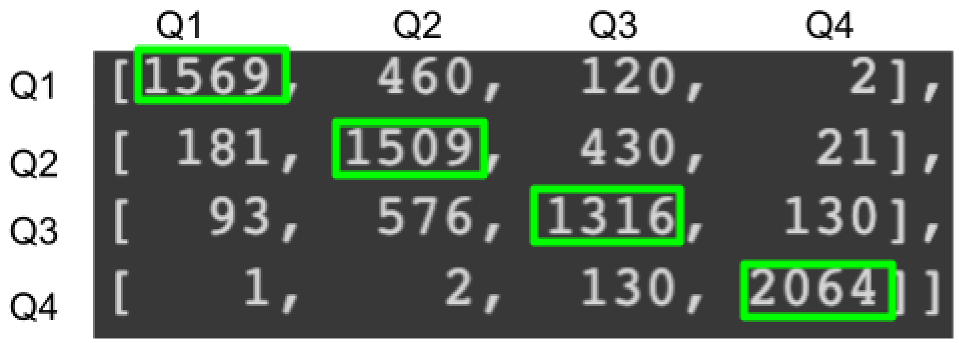
$$Q1\colon\ [0.013\%,\ 0.054\%)$$

$$Q2\colon\ [0.054\%,\ 0.090\%)$$

$$Q3\colon\ [0.090\%,\ 1.650\%)$$

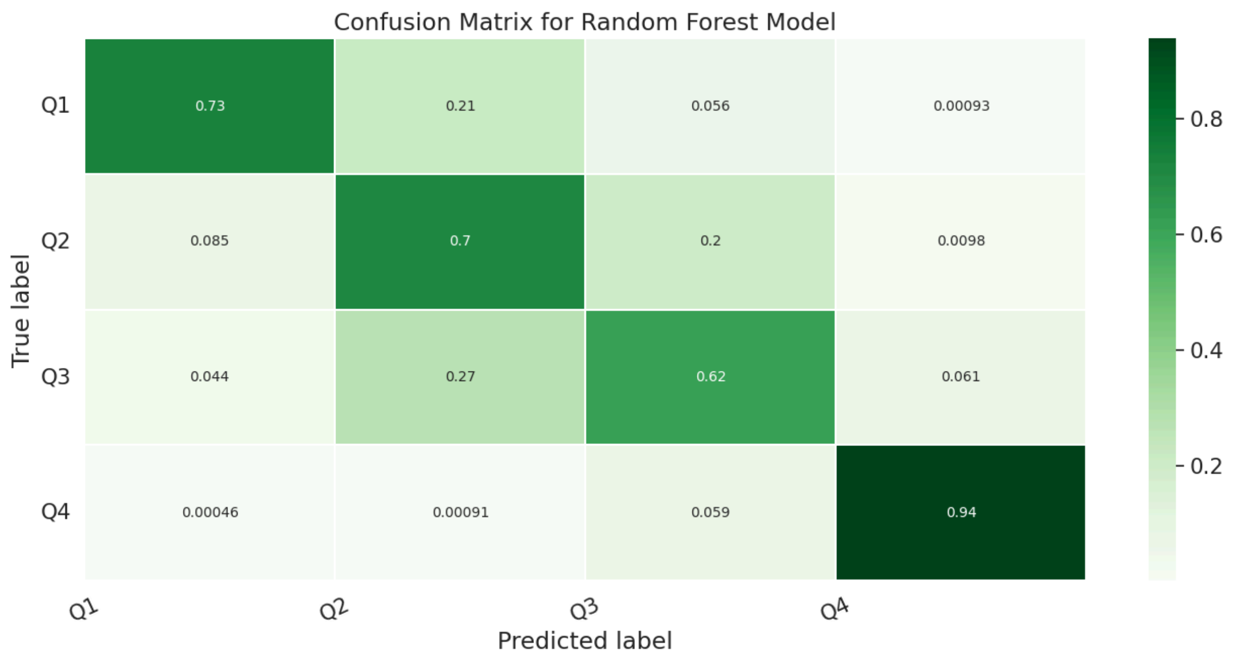
$$Q4\colon\ [1.650\%,\ 4.925\%)$$

We split into a train-test split of 80-20, and had a much higher accuracy of *74.5%*. The *n_estimators* parameter, or the number of decision trees, was also changed to 100. This process of tweaking model parameters to improve accuracy is known as **hyperparameter tuning.** We initially experienced an issue with overfitting where our model was receiving 99.5% accuracy. To fix this, we dropped the *county* column, since county was already being accounted for twice when we originally merged the two datasets. A code from *sk-learn* [4] was used.



```
[[1569,  460,  120,    2],
 [ 181, 1509,  430,   21],
 [  93,  576, 1316,  130],
 [   1,    2,  130, 2064]]
```

Above is this model's **confusion matrix**, which expresses how many of the classifier's predictions were correct, and if incorrect, where the classifier was incorrect. The highlighted green boxes are all correct predictions, meaning the model matches the true label. The rows represent *true labels* and the columns are the *predicted labels*. Additionally, below is a heatmap visual with percentages for the confusion matrix. We can see that the model was very accurate with Q4.

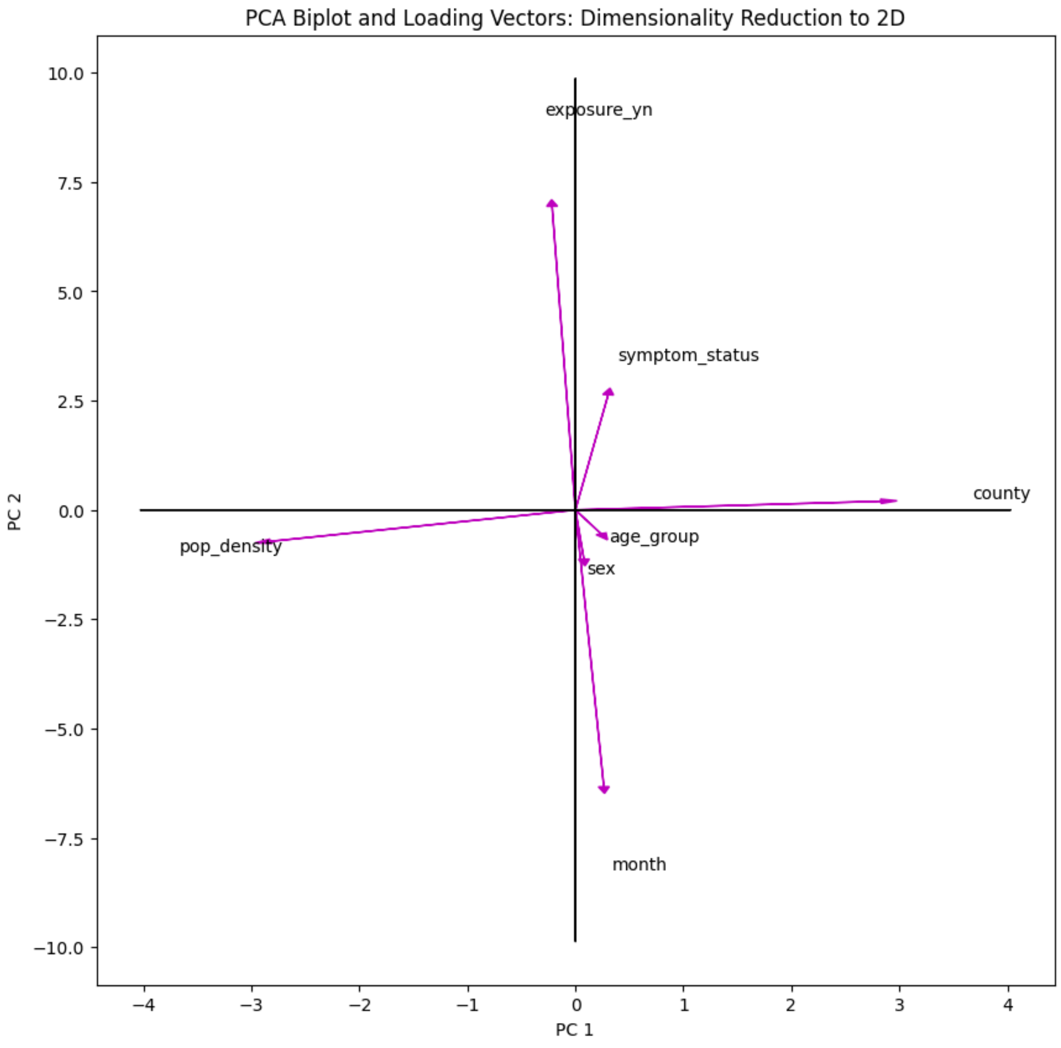


## 4. Principal Component Analysis

PCA, otherwise known as factor analysis, is a dimensionality reduction technique that is used to transform a large dataset into a smaller one by reducing the dimensions, or columns. In this case, multiple columns are combined into a **principal component (PC)** until we are left with a **2D (two-dimensional)** dataset, still containing the information from the higher-dimensional dataset.

PCA is projecting the original variables onto a lower dimensional subspace defined by the two most important components (PC1 and PC2). The most important feature has the most variance, and so on. For our case, we simply examine the variance explained by each PC to see which original features contribute the most to the variance in our dataset. We use PCA to see which columns of demographic data were affecting the prevalence the most by comparing which variables affect PC1 vs. PC2 and correlation between certain variables.
*Note:* It's possible to do PCA for down to any number of dimensions, like 3D, 4D, etc. We choose 2D to allow for a visual representation (the 2D Biplot).

Before our analysis, we must perform scaling and normalization on the features of our merged dataset, which ensures that all the features are on a similar scale, thus having 0 mean and unit variance.
A biplot visualization code [5] is implemented, resulting in the following graph (for the entire state of NY). Note that a **loading vector** is each of the features, and we compare this with the PCs.

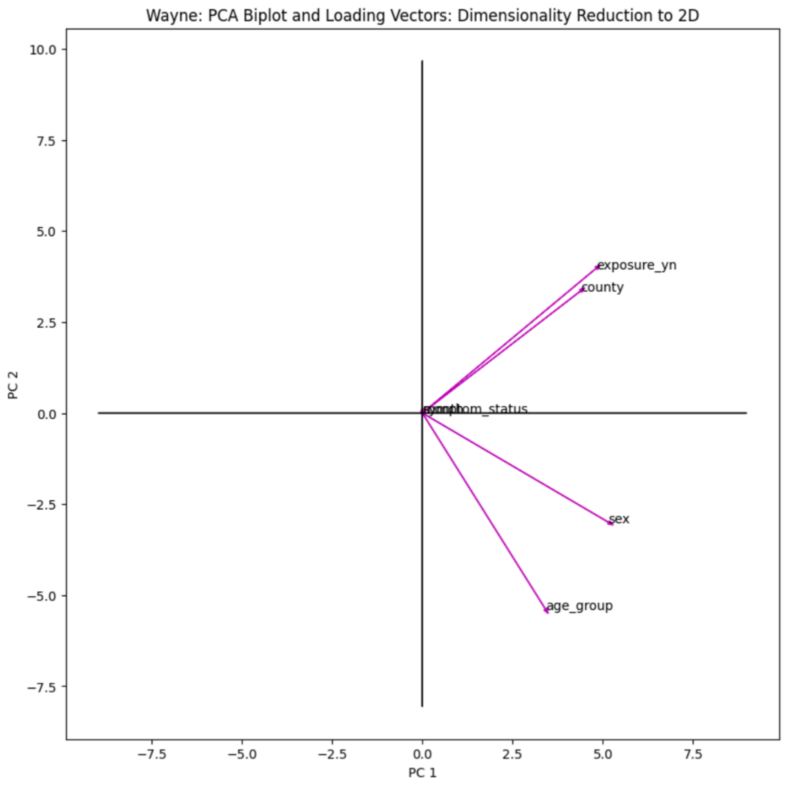


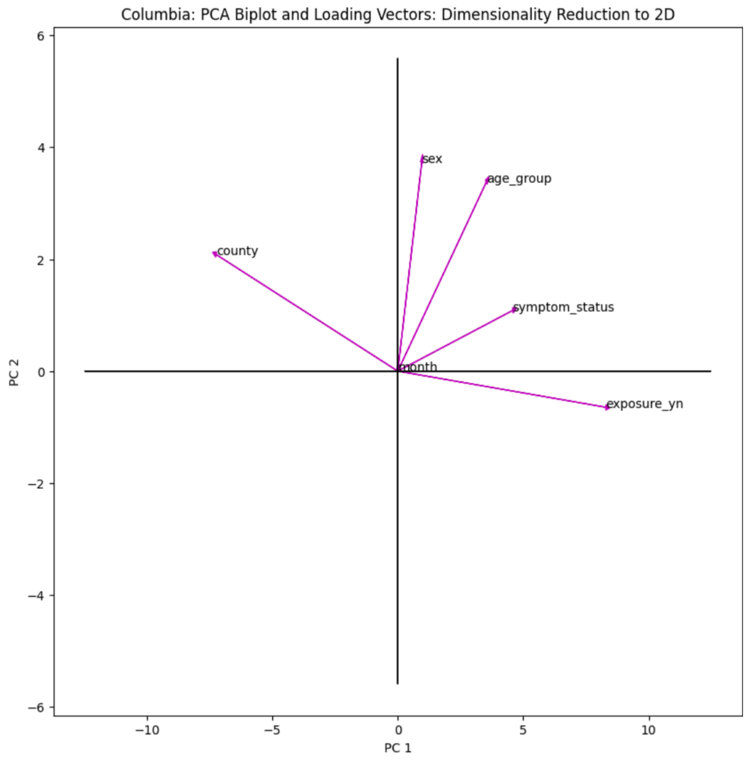
Original variables are shown as vectors which start from (0,0) and extend to the coordinates given by the loading vector.

- *Orientation:* The more parallel a vector is to a principal component, the more it contributes to that principal component.
- *Size:* The longer the vector, the more variability of this variable is represented by the two displayed principal components
- *Angle:* small angles represent high positive correlation, right angles represent lack of correlation, opposite angles represent high negative correlation.

From our plot above, the features *month, symptom_status, sex, and exposure_yn* contribute the most to PC2 (the y-axis) while *pop_density* and *county* contribute strongly to PC1 (the x-axis).

We can also use a similar methodology by plotting for a specific county, and compare 2 counties by seeing how the  features contribute to the PCs. For example, below is an analysis of *Wayne County (left)*  vs. *Columbia County* (right).





In Wayne, *exposure_yn* and *county* are **positively** correlated because there is a very small angle between their vectors but have no correlation with *sex* or *age_group* since they are almost at a right angle. Here, *symptom_status* is too small to visualize or have an effect on PC1 or PC2. We additionally notice that *exposure_yn* and *county* are almost equidistant from each PC and *sex* or *age_group* are not close to either PC, suggesting there is a different PC that is impacted by it. Meanwhile in Columbia, exposure_yn and county are **negatively** correlated since they are opposite angles. Features *sex* and *age_group* contribute the most to PC1 while *symptom_status*, *exposure_yn* and *county* contribute to PC2. Note that month in this case is too small to visualize, meaning it has very little contribution to either PC1 or PC2.

## 5. Pooled/Grouped Testing: Efficiency Analysis

Another use of our machine learning model is for *p (prevalence)* estimates for **pooled**, or grouped COVID testing algorithms. Individually screening large populations of people is wasteful and expensive. The concept of pooled testing is to utilize an algorithm that optimizes grouped testing depending on the likelihood of the group overall to have COVID. The result of each test may not tell us exactly *who* is infected, but it will be negative if no one is infected and positive even if there is just one person who is sick.

Literature review from Chapter 11 of the book *Pandemics: Insurance and Social Protection* outlines **Dorfman's two-stage Algorithm**, as follows:

- Choose a pool with size *s* (divide the population into *N/s* groups). Pool the samples from that group into a specific test.
    - If a pool is negative, we know everybody in that group is negative, and we have saved tests.
    - If a pool is positive, we then follow up individually with the corresponding group which helps clarify which individuals actually have the infection

expected # test

$$\mathbb{E}(T) = \frac{N}{s} + N\left(1 - (1-p)^s\right)$$

$$= \frac{N}{s}\left(1 + s\left(1 - (1-p)^s\right)\right)$$

$$= \left(\frac{1}{s} + 1 - (1-p)^s\right)N$$

$$\therefore \frac{\mathbb{E}(T)}{N} = \left(\frac{1}{s} + 1 - (1-p)^s\right)$$

Here shown are the mathematical calculations (derivatives for optimization function) I conducted to clarify that *E(T)/N*, the expected number of tests per individual is $2\sqrt{p}$ using *p* as the probability of being infected and *(1-p)* as being negative.

Now, we clearly see that for my prevalence range in my data model, the max chance for a city to get covid was 0.05. Note that for Individual testing, we have *E(T)/N* = *N/N* = *1* test per individual. For our dataset, we use our deduced formula to see that $2\sqrt{0.05} < 1$.

$$\therefore (1-p)^s = 1 - ps + \underbrace{\frac{s(s-1)}{2!}p^2 - \frac{s(s-1)(s-2)}{2!}p^3 + \dots + s(-p)^{s-1} + (-p)^n}_{p \text{ appreciably small so this goes to } 0}$$

$$\therefore (1-p)^s \approx 1 - ps$$

$$\frac{1}{s} + 1 - (1-p)^s \approx \frac{1}{s} + 1 - (1-ps)$$

$$= \frac{1}{s} + ps$$

$$\frac{d}{ds}\left(\frac{1}{s} + ps\right) = \frac{-1}{s^2} + p$$

$$-\frac{1}{s^2} + p = 0$$

$$p = \frac{1}{s^2}$$

$$s^2 = \frac{1}{p}$$

$$s = \frac{1}{\sqrt{p}}$$

Hence, we know this machine learning model can be used to output prevalence estimates of counties to be utilized in the formation of pooled testing algorithms for efficient COVID testing!

$$\therefore \frac{\mathbb{E}(T)}{N} = \frac{1}{s} + 1 - (1-ps)$$

$$= \sqrt{p} + 1 - \left(1 - p \cdot \frac{1}{\sqrt{p}}\right)$$

$$= \sqrt{p} + 1 - 1 + p^{1-\frac{1}{2}}$$

$$= \sqrt{p} + \sqrt{p}$$

$$= 2\sqrt{p}$$

## 6. Future Work & Impact

This research explores a variety of techniques to explore demographic factors and their impact on the prevalence of COVID. The use of real-world data with COVID prevalence further allows us to optimize pandemic testing through deriving generalized algorithms. There is great potential for these prevalence estimates to be applied for other group testing algorithms as well, such as *binary splitting by halving, grid (r,s) design,* and *P-BEST algorithms*. Given a population of $N$ people, their demographics can be inputted to predict what the overall prevalence of COVID in their population is, and hence an estimate of the number of infected people. This estimate can be applied to group testing formulas to find the expected number of tests per individual - whenever this value is less than 1, pool testing is more sustainable than individual testing for that given population.

## 7. References

[1] Pandemics: Insurance and Social Protection. (2022). *Springer Actuarial*. doi: 10.1007/978-3-030-78334-1

[2]: Datasets used:
- (2023). Retrieved 16 August 2023, from https://data.cdc.gov/Case-Surveillance/COVID-19-Case-Surveillance-Public-Use-Data-with-Ge/n8mc-b4w4
- nytimes. (2023). covid-19-data/rolling-averages at master · nytimes/covid-19-data. Retrieved 16 August 2023, from https://github.com/nytimes/covid-19-data/tree/master/rolling-averages
- Bureau, U. C. (2023). County population totals and components of change: 2020-2022. Retrieved from https://www.census.gov/data/tables/time-series/demo/popest/2020s-counties-total.html

[3]: sklearn.linear_model.LinearRegression. (2023). Retrieved 16 August 2023, from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

[4]: sklearn.ensemble.RandomForestClassifier. (2023). Retrieved 16 August 2023, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[5]: Interpretation and visualization. (2023). Retrieved 16 August 2023, from https://www.geo.fu-berlin.de/en/v/soga-py/Advanced-statistics/Multivariate-Approaches/Principal-Component-Analysis/PCA-the-basics/Interpretation-and-visualization/index.html