COVID-19 Case Surveillance Public Use Data

Jeffrey, Nathan, Nathanael

Introduction

COVID-19:

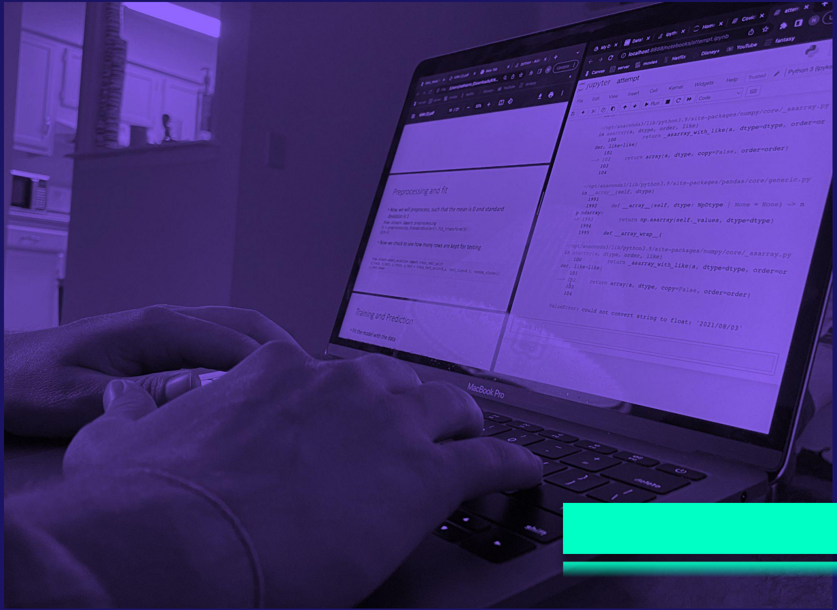
- Is an infectious disease started back in December 12, 2019
- The CDC has a database that stores all covid-19 cases. (US)

Data:

- Clean and Visualize data by different distributions and predict the outcome by different data algorithms.

Algorithms:

- Using Naive Bayes, K-Means, kNN and Spectral Clustering



Purpose

- Quantitatively compare and contrast spectral clustering with K-means and KNN clustering in order to determine the relationship between peaks of Covid-19 transmission and likelihood of death.
- Predict the likelihood of death given multiple factors about a patient.

Techniques & Tools

Predicting the likelihood of death given the factors of Covid-19.

Naive Bayes

Used to apply data with categorical predictors and compare probability or the likelihood of an event

KNN

- Used to identify k records in the training dataset that are similar to the new record we intend to classify.
- Uses the similar records to classify the new record associated to its nearest neighbors.
- does not make assumptions about the relationship between the class membership and the predictors.

K-Means

Analyzes unlabeled samples and attempts to place in related clusters. The variable k represents the number of clusters imposed on the data.

Spectral Clustering

Divides the data points into several groups such that points in the same group are similar and points in different groups are dissimilar to each other.

Expected Outcome

- We predict that during peak transmission points of Covid-19 that there will be more deaths.
- We predict that elderly people will be more susceptible to dying from Covid than other age groups
- We will also be looking at gender and race to determine if a specific demographic is more susceptible to dying from Covid

Project’s Outcome

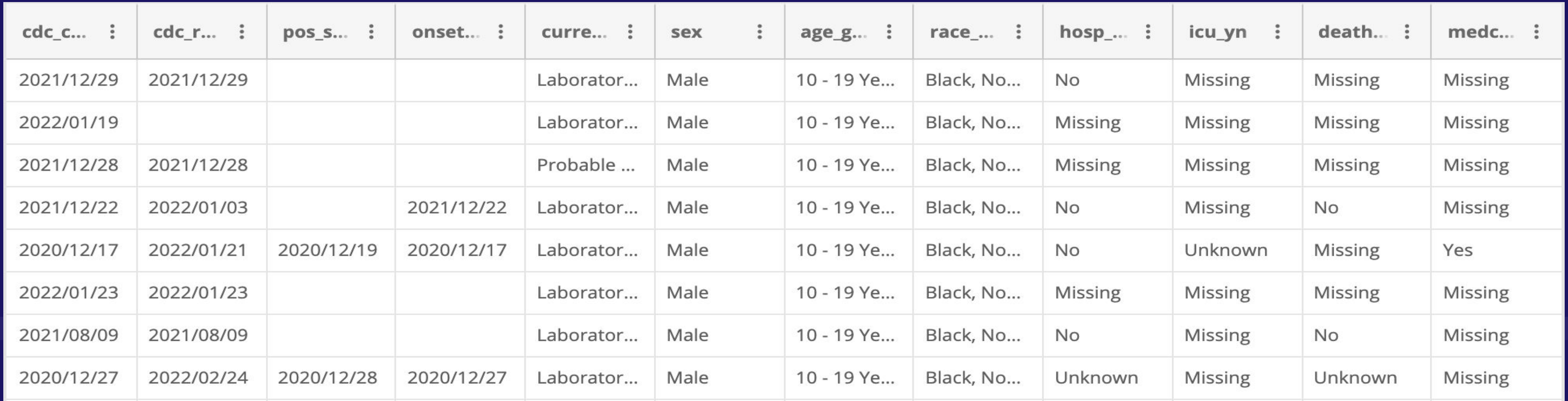
01 Data

02 Cleaning & Visualization

03 Use of Data Models

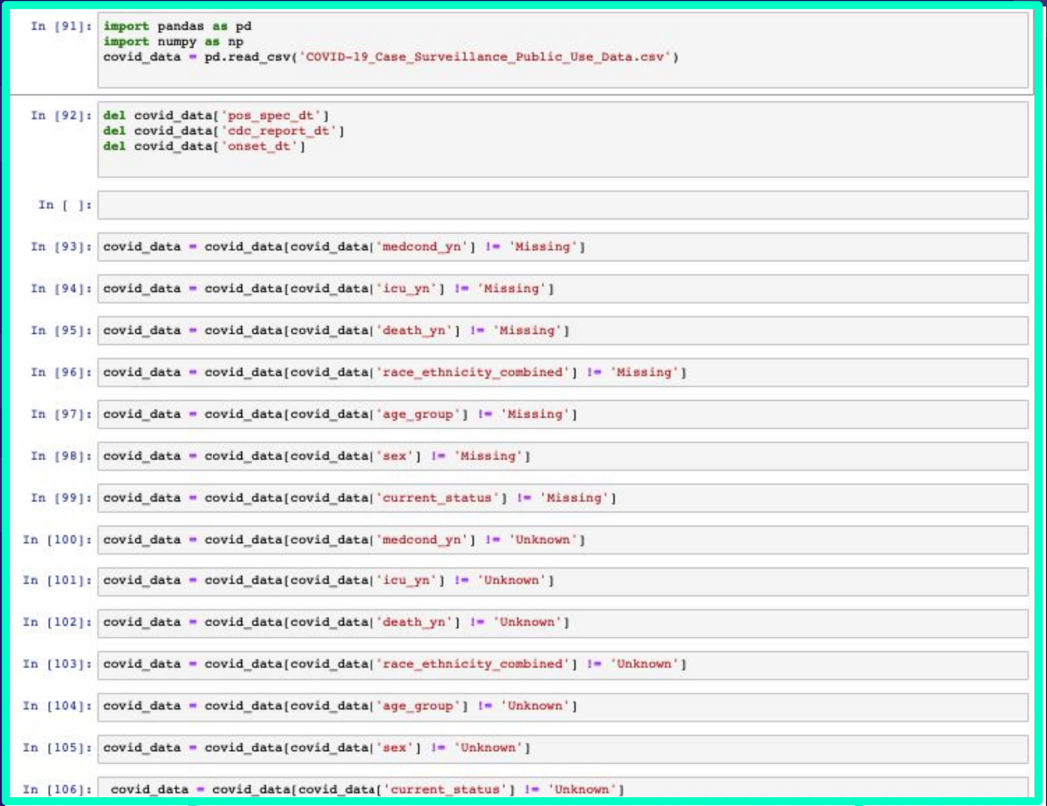
04 Results

Data



cdc_c...	cdc_r...	pos_s...	onset...	curre...	sex	age_g...	race_...	hosp_...	icu_yn	death...	medc...
2021/12/29	2021/12/29			Laborator...	Male	10 - 19 Ye...	Black, No...	No	Missing	Missing	Missing
2022/01/19				Laborator...	Male	10 - 19 Ye...	Black, No...	Missing	Missing	Missing	Missing
2021/12/28	2021/12/28			Probable ...	Male	10 - 19 Ye...	Black, No...	Missing	Missing	Missing	Missing
2021/12/22	2022/01/03		2021/12/22	Laborator...	Male	10 - 19 Ye...	Black, No...	No	Missing	No	Missing
2020/12/17	2022/01/21	2020/12/19	2020/12/17	Laborator...	Male	10 - 19 Ye...	Black, No...	No	Unknown	Missing	Yes
2022/01/23	2022/01/23			Laborator...	Male	10 - 19 Ye...	Black, No...	Missing	Missing	Missing	Missing
2021/08/09	2021/08/09			Laborator...	Male	10 - 19 Ye...	Black, No...	No	Missing	No	Missing
2020/12/27	2022/02/24	2020/12/28	2020/12/27	Laborator...	Male	10 - 19 Ye...	Black, No...	Unknown	Missing	Unknown	Missing

Cleaning Code



```
In [91]: import pandas as pd
         import numpy as np
         covid_data = pd.read_csv('COVID-19_Case_Surveillance_Public_Use_Data.csv')

In [92]: del covid_data['pos_spec_dt']
         del covid_data['cdc_report_dt']
         del covid_data['onset_dt']

In [ ]:

In [93]: covid_data = covid_data[covid_data['medcond_yn'] != 'Missing']

In [94]: covid_data = covid_data[covid_data['icu_yn'] != 'Missing']

In [95]: covid_data = covid_data[covid_data['death_yn'] != 'Missing']

In [96]: covid_data = covid_data[covid_data['race_ethnicity_combined'] != 'Missing']

In [97]: covid_data = covid_data[covid_data['age_group'] != 'Missing']

In [98]: covid_data = covid_data[covid_data['sex'] != 'Missing']

In [99]: covid_data = covid_data[covid_data['current_status'] != 'Missing']

In [100]: covid_data = covid_data[covid_data['medcond_yn'] != 'Unknown']

In [101]: covid_data = covid_data[covid_data['icu_yn'] != 'Unknown']

In [102]: covid_data = covid_data[covid_data['death_yn'] != 'Unknown']

In [103]: covid_data = covid_data[covid_data['race_ethnicity_combined'] != 'Unknown']

In [104]: covid_data = covid_data[covid_data['age_group'] != 'Unknown']

In [105]: covid_data = covid_data[covid_data['sex'] != 'Unknown']

In [106]:  covid_data = covid_data[covid_data['current_status'] != 'Unknown']
```

PROCESS: Cleaning Data

Step 1

Download Covid-19 Case Surveillance Public Use Data csv file

Step 2

Deleted first positive specimen date, CDC report date, and onset of symptoms date columns.

Step 3

Deleted all rows that were missing elements

Step 4

Deleted all rows that were unknown elements

Step 5

Save new data set to new csv file

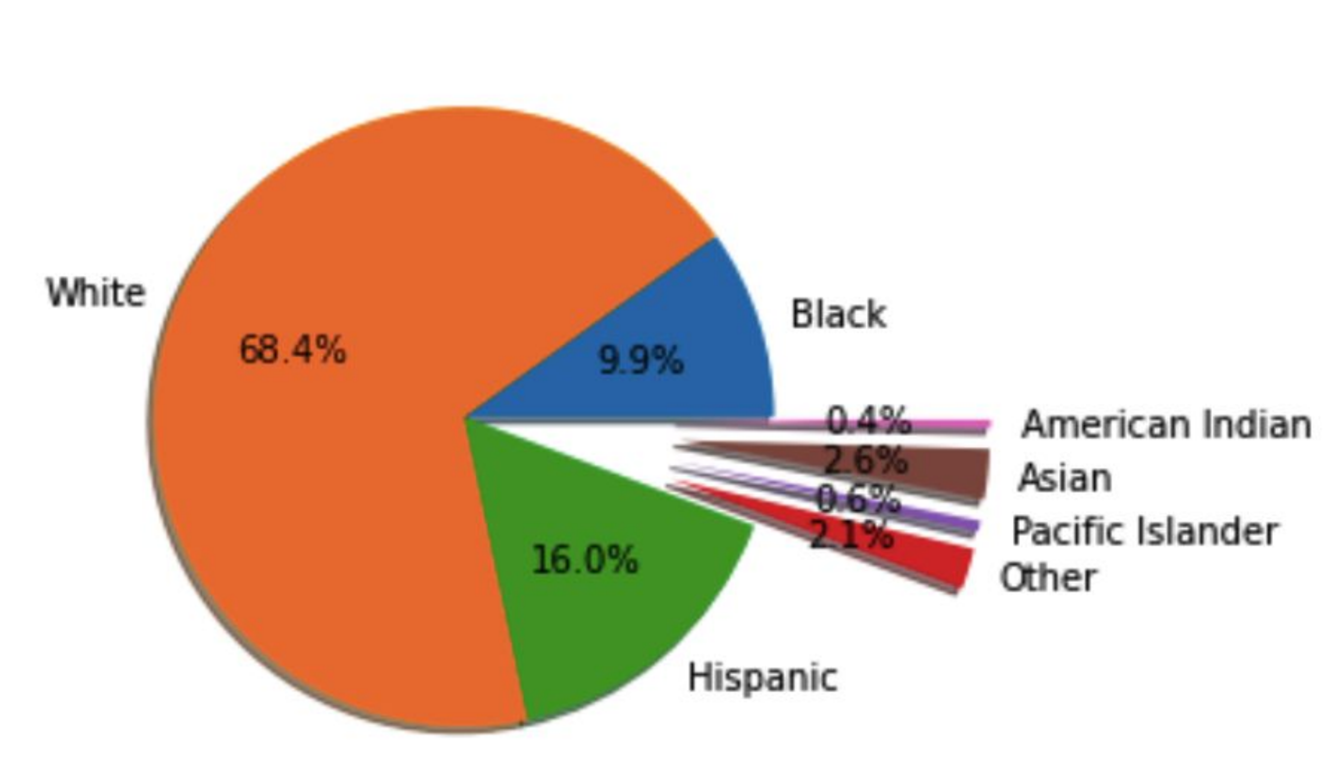
Result

Cleaned 69,664,983 rows to 1,094,551 rows

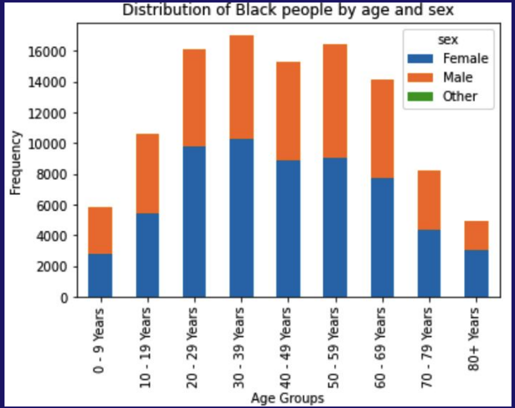
[ ]

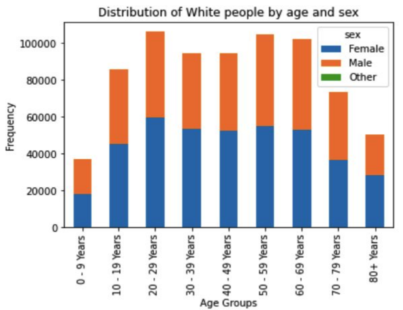
Visualizations

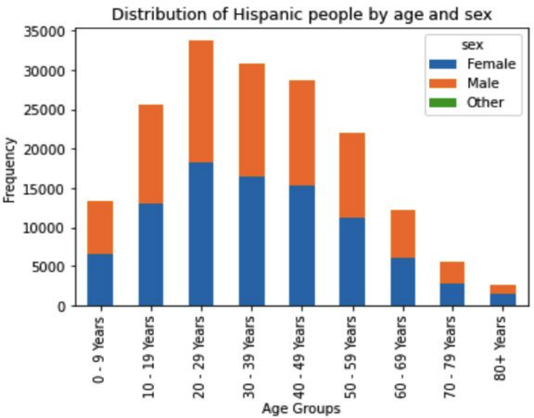
Distribution of People

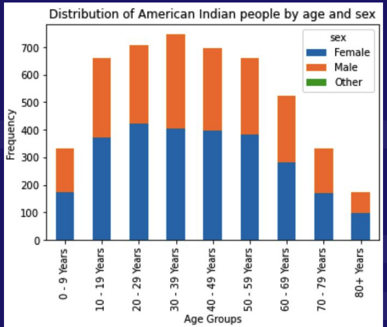


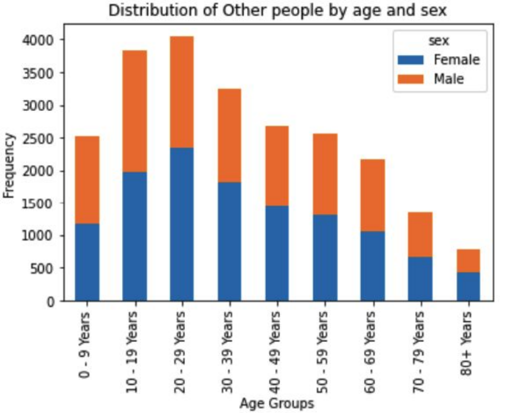
Visualizations of different Races

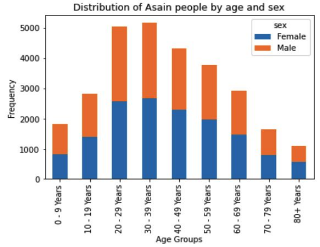


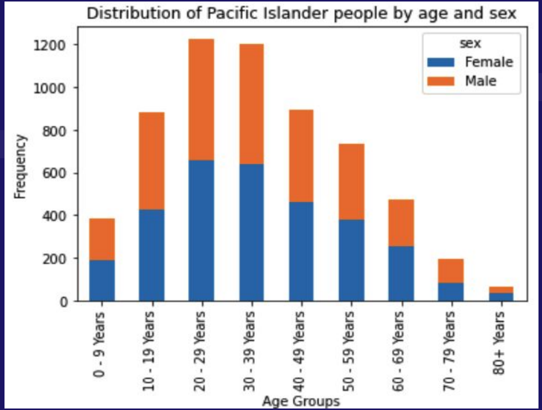




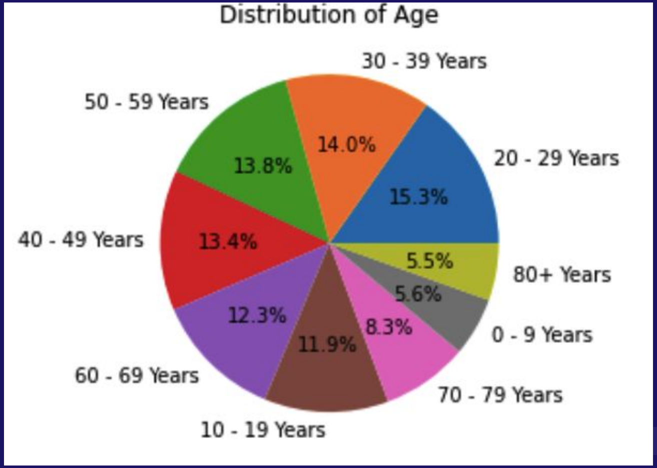


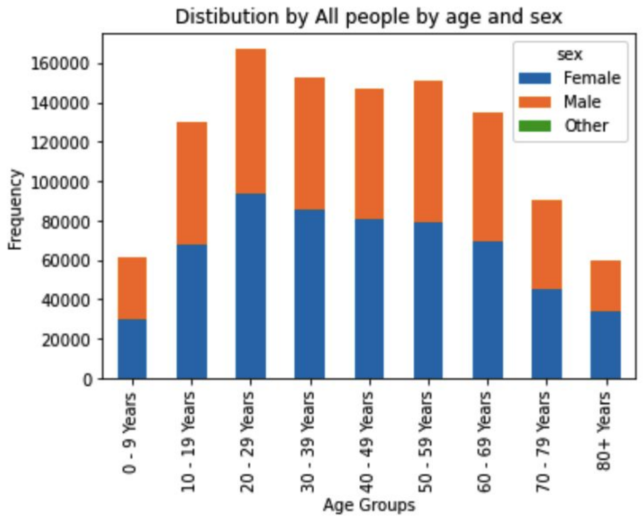


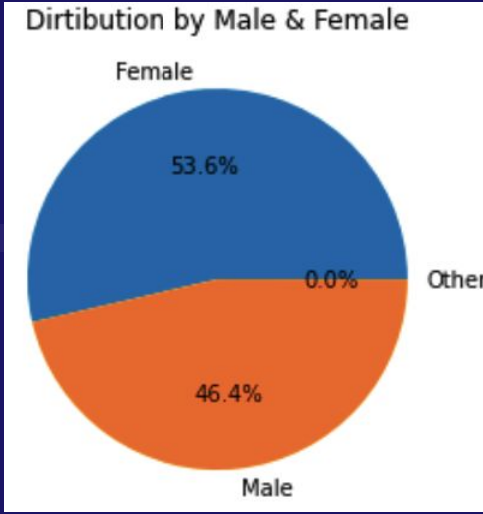




Visualizations of All



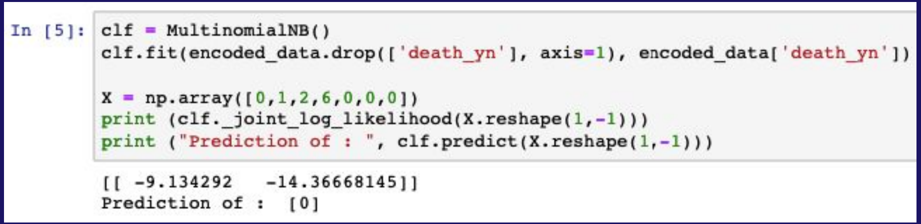




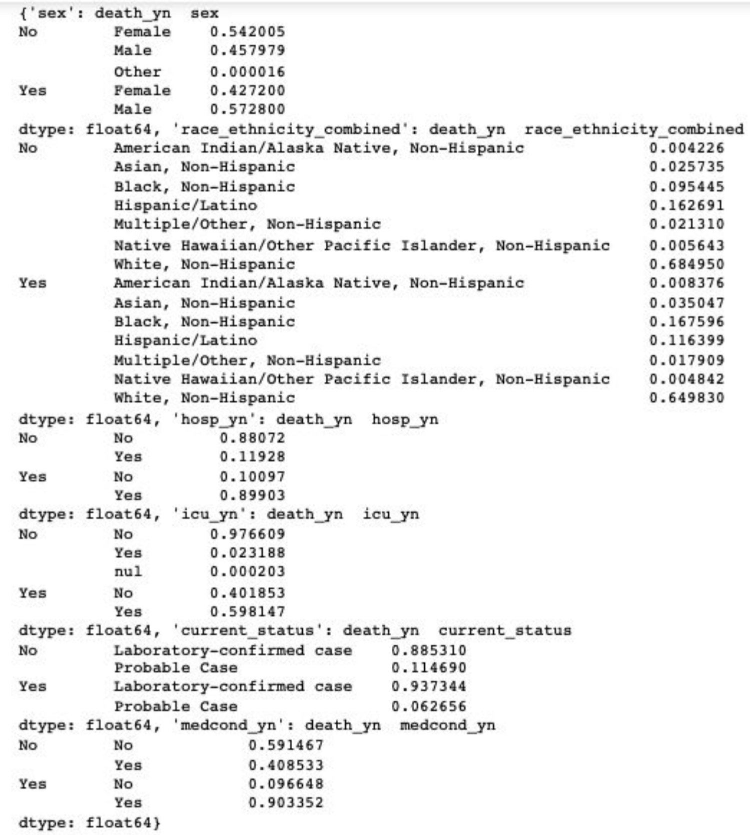
Methods

Naive-Bayes

First we calculated the percentages that ended in death, then we calculated a likelihood table to determine the individual percentage of death based on specific feature. Lastly, we used a naive-bayes model to create an angorthim that would predict the possibility of death based on the input of these specific features such as; lab confirmed case, sex, age, race, hospitalization, ICU, and underlying conditions.



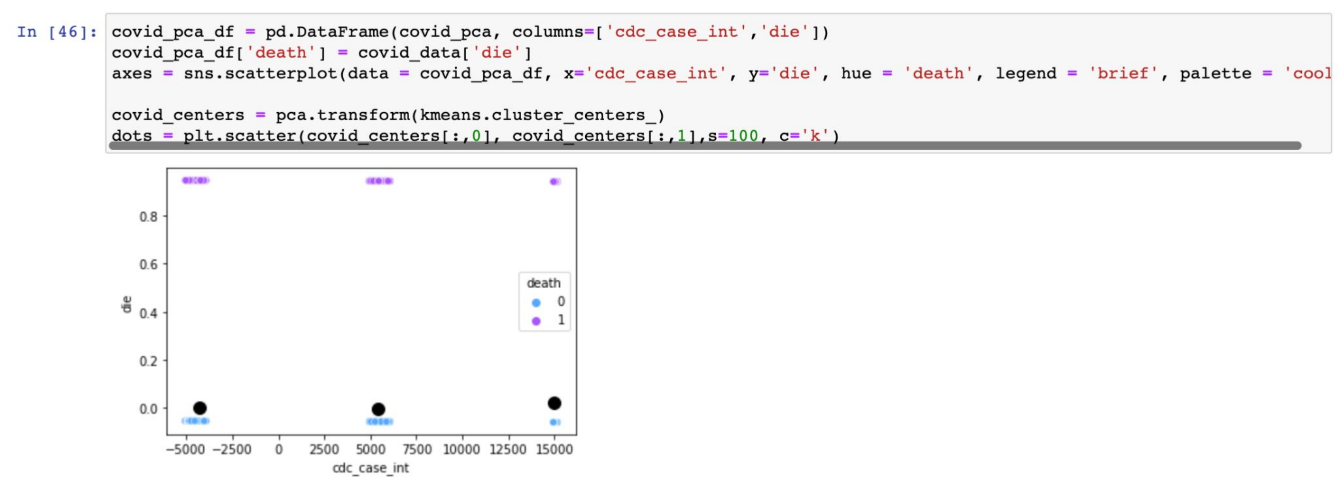
```
In [5]: clf = MultinomialNB()
        clf.fit(encoded_data.drop(['death_yn'], axis=1), encoded_data['death_yn'])

        X = np.array([0,1,2,6,0,0,0,0])
        print (clf._joint_log_likelihood(X.reshape(1,-1)))
        print ("Prediction of : ", clf.predict(X.reshape(1,-1)))

        [[ -9.134292   -14.36668145]]
        Prediction of :  [0]
```



```
{'sex': death_yn  sex
No          Female    0.542005
            Male      0.457979
            Other     0.000016
Yes         Female    0.427200
            Male      0.572800
dtype: float64, 'race_ethnicity_combined': death_yn  race_ethnicity_combined
No          American Indian/Alaska Native, Non-Hispanic              0.004226
            Asian, Non-Hispanic                                      0.025735
            Black, Non-Hispanic                                      0.095445
            Hispanic/Latino                                          0.162691
            Multiple/Other, Non-Hispanic                             0.021310
            Native Hawaiian/Other Pacific Islander, Non-Hispanic    0.005643
            White, Non-Hispanic                                      0.684950
Yes         American Indian/Alaska Native, Non-Hispanic              0.008376
            Asian, Non-Hispanic                                      0.035047
            Black, Non-Hispanic                                      0.167596
            Hispanic/Latino                                          0.116399
            Multiple/Other, Non-Hispanic                             0.017909
            Native Hawaiian/Other Pacific Islander, Non-Hispanic    0.004842
            White, Non-Hispanic                                      0.649830
dtype: float64, 'hosp_yn': death_yn  hosp_yn
No          No         0.88072
            Yes        0.11928
Yes         No         0.10097
            Yes        0.89903
dtype: float64, 'icu_yn': death_yn  icu_yn
No          No        0.976609
            Yes       0.023188
            nul       0.000203
Yes         No        0.401853
            Yes       0.598147
dtype: float64, 'current_status': death_yn  current_status
No          Laboratory-confirmed case    0.885310
            Probable Case                0.114690
Yes         Laboratory-confirmed case    0.937344
            Probable Case                0.062656
dtype: float64, 'medcond_yn': death_yn  medcond_yn
No          No            0.591467
            Yes           0.408533
Yes         No            0.096648
            Yes           0.903352
dtype: float64}
```

K-Means

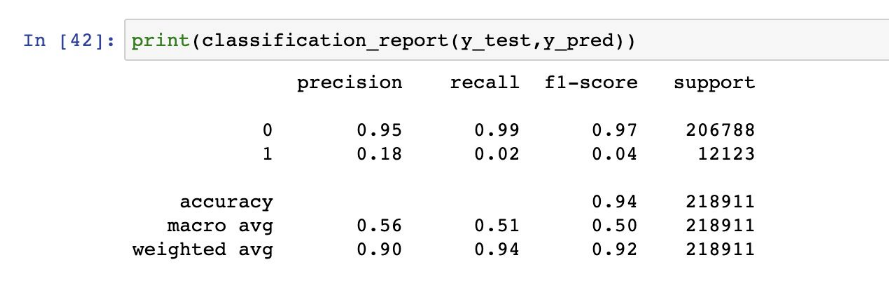
- First we took the earliest report column and changed it from dates to ints, then we used sklearn’s k-means model and Principle Component Analysis (PCA) estimator to help us plot the centroid locations. The columns that we plotted were the earliest report column and death column.



In [46]:
```
covid_pca_df = pd.DataFrame(covid_pca, columns=['cdc_case_int','die'])
covid_pca_df['death'] = covid_data['die']
axes = sns.scatterplot(data = covid_pca_df, x='cdc_case_int', y='die', hue = 'death', legend = 'brief', palette = 'cool

covid_centers = pca.transform(kmeans.cluster_centers_)
dots = plt.scatter(covid_centers[:,0], covid_centers[:,1],s=100, c='k')
```

k-Nearest Neighbors

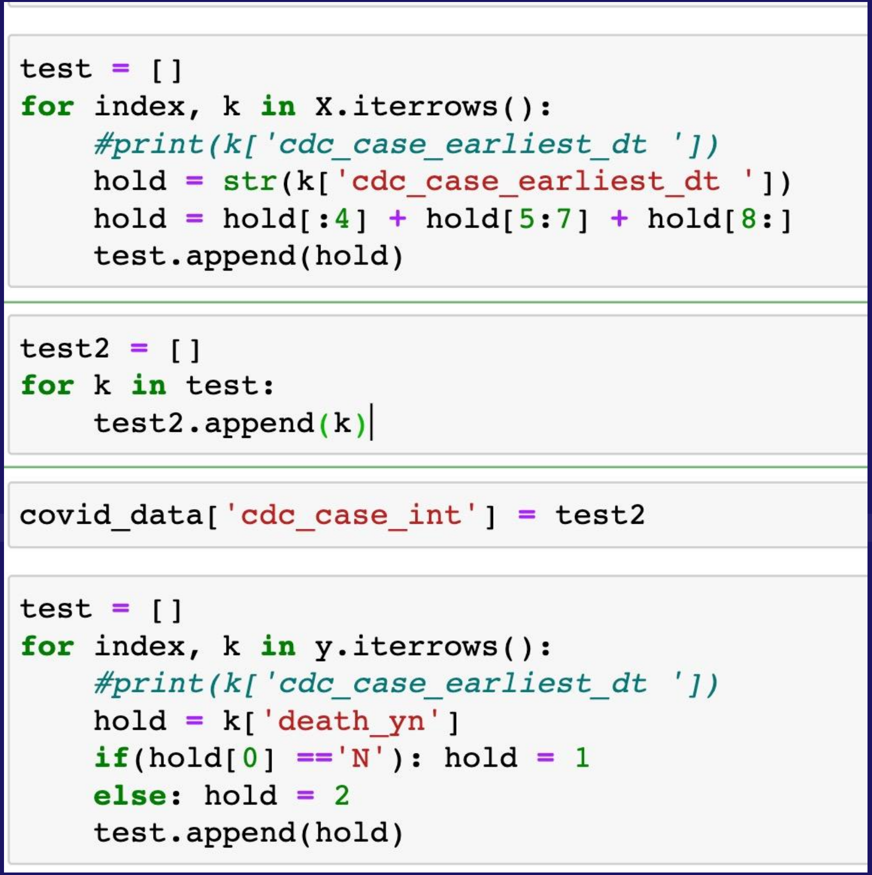
First we took the earliest report column and changed it from dates to ints, and changed the deaths column to a binary value (0,1), then we selected the training set and testing set, then we did preprocessing using sklearn's standardscaler. Then we used the sklearn's k-neighbors classifier to test the predicted data. Lastly. to determine the results of the tests we made a confusion matrix and called classification_report.



```
In [42]: print(classification_report(y_test,y_pred))

              precision    recall  f1-score   support

           0       0.95      0.99      0.97    206788
           1       0.18      0.02      0.04     12123

    accuracy                           0.94    218911
   macro avg       0.56      0.51      0.50    218911
weighted avg       0.90      0.94      0.92    218911
```

Spectral Clustering

First we took the earliest report column and changed it from dates to ints, and changed the deaths column to a binary value (0,1), then we imported numpy in order to change our dataframe into more friendly arrays. With these arrays we used sklearning spectral clustering to fit our data. Unfortunately our dataset was too large to work with the spectral clustering model, and even with a minimal amount of clusters the jupyter notebook would not complete and die.



```
test = []
for index, k in X.iterrows():
    #print(k['cdc_case_earliest_dt '])
    hold = str(k['cdc_case_earliest_dt '])
    hold = hold[:4] + hold[5:7] + hold[8:]
    test.append(hold)
```

```
test2 = []
for k in test:
    test2.append(k)
```

```
covid_data['cdc_case_int'] = test2
```

```
test = []
for index, k in y.iterrows():
    #print(k['cdc_case_earliest_dt '])
    hold = k['death_yn']
    if(hold[0] =='N'): hold = 1
    else: hold = 2
    test.append(hold)
```

{ }

Analysis / Conclusion

Analyzation of Data

Naive-Bayes

The Naive-Bayes model provided us with an accurate algorithm to predict likelihood of death based on select features. According to our demographic, there is a high probability that we would not die.

K-Means

The K-Means model showed an accurate distribution of peak covid transmission time periods. While deaths increased in peak periods, the proportionality of deaths to not deaths stay consistent regardless of peak times.

kNN

Given the date of someone contracting covid, there is a low probability that death can be predicted. On the other hand, the kNN model had a high accuracy of predicting non-deaths.

Spectral Clustering

Unfortunately, our dataset was too large to accommodate spectral clustering given the constraints of our environments.

Model Comparison

- Both the Naive-Bayes and kNN models struggled to classify instances where death occurred. However they were both highly accurate in predicting instances of non-deaths. The Naive-Bayes did allow for more specification then the kNN model.
- The K-means model worked well to cluster our data into three clustoids, but still struggled to classify deaths similarly to the kNN model.
- Project Specification- 70:30

Reference

[1] https://data.cdc.gov/Case-Surveillance/COVID-19-Case-Surveillance-Public-Use-Data/vbim-akqf

Thanks for listening.
Any Questions?