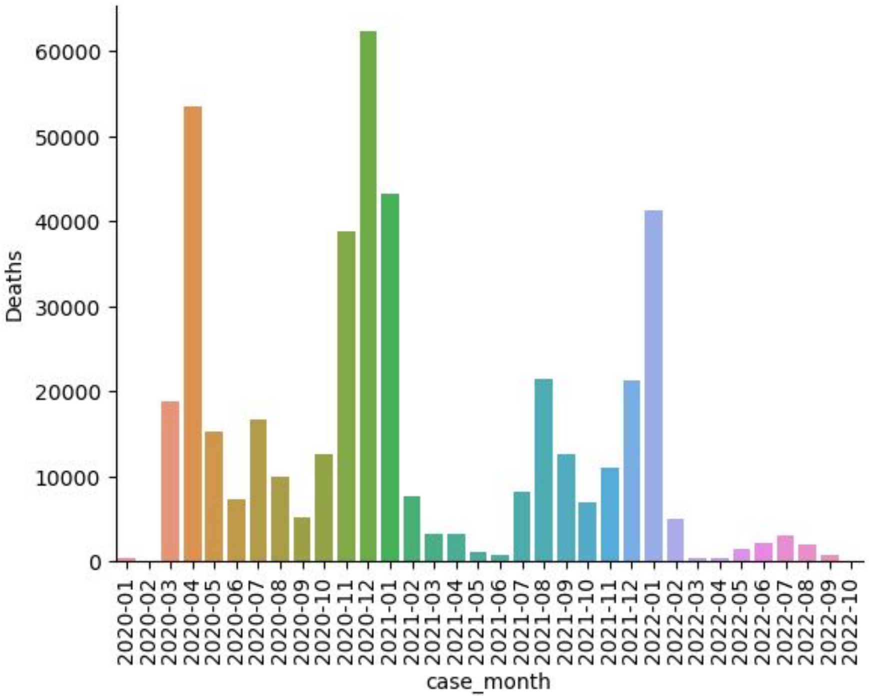
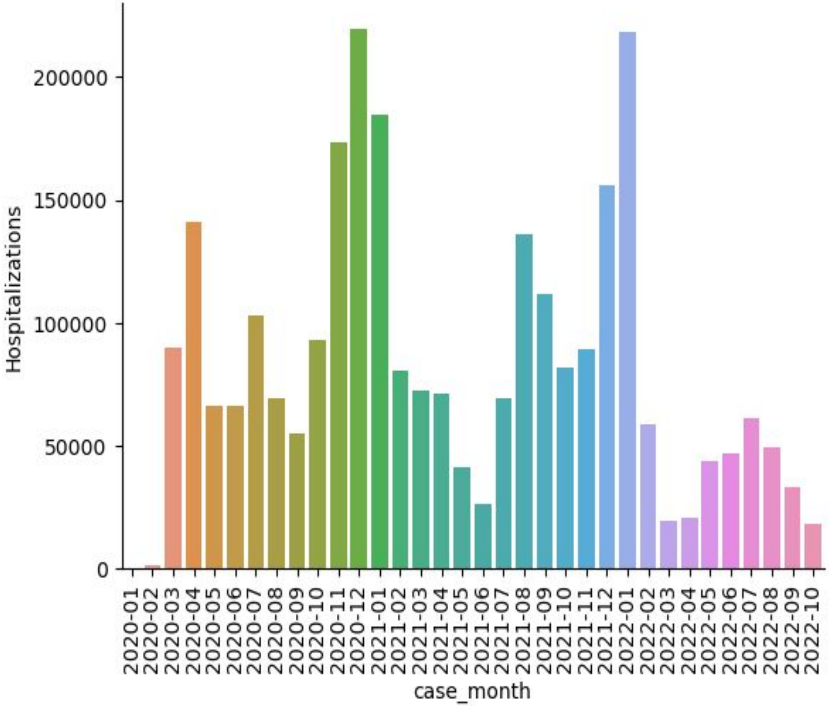
Effects of Geography & Demography on COVID-19 Hospitalizations and Deaths (Part 2)

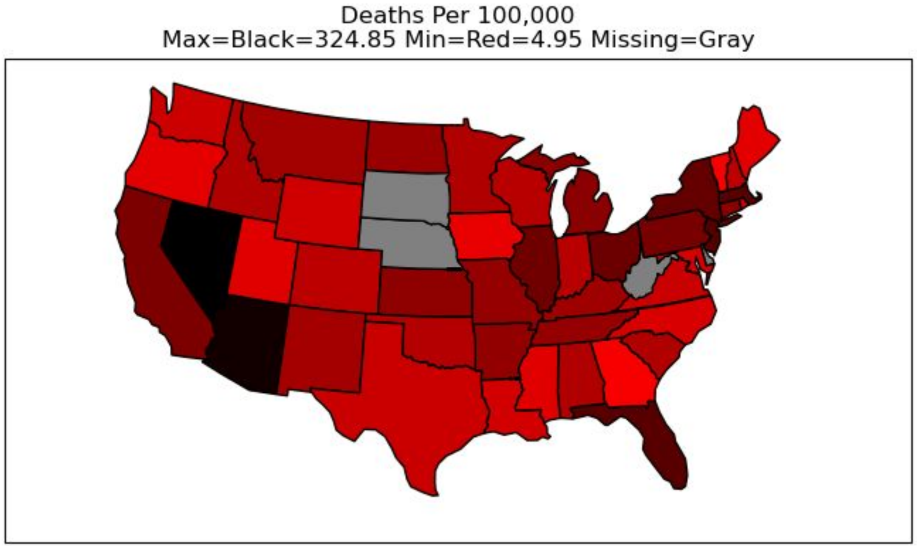
- Daniel Johnson
- Brian Li
- Braden Lockwood
- Evan Powell
- Jowaki Merani
- Mason Hawks

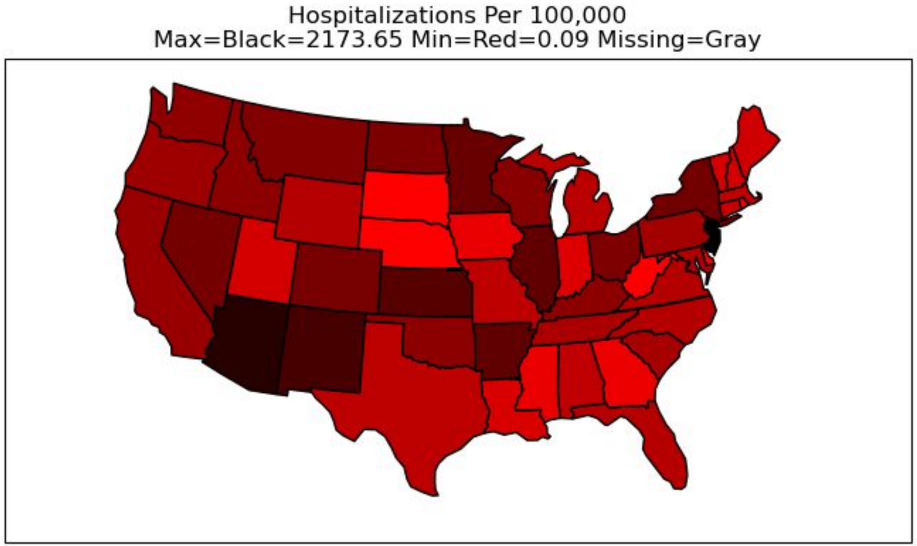
Statistics & EDA Results

- ➔ Target (hosp/death) visualizations over different features (age, month, etc)
- ➔ State ranking T-tests for hosps/deaths per capita
- ➔ Visualizations for illness severity by age and race
- ➔ Feature ranking based on P-values and critical regions


Deaths
60000
50000
40000
30000
20000
10000
0
2020-01
2020-02
2020-03
2020-04
2020-05
2020-06
2020-07
2020-08
2020-09
2020-10
2020-11
2020-12
2021-01
2021-02
2021-03
2021-04
2021-05
2021-06
2021-07
2021-08
2021-09
2021-10
2021-11
2021-12
2022-01
2022-02
2022-03
2022-04
2022-05
2022-06
2022-07
2022-08
2022-09
2022-10
case_month


Hospitalizations
200000
150000
100000
50000
0
2020-01
2020-02
2020-03
2020-04
2020-05
2020-06
2020-07
2020-08
2020-09
2020-10
2020-11
2020-12
2021-01
2021-02
2021-03
2021-04
2021-05
2021-06
2021-07
2021-08
2021-09
2021-10
2021-11
2021-12
2022-01
2022-02
2022-03
2022-04
2022-05
2022-06
2022-07
2022-08
2022-09
2022-10
case_month


Deaths Per 100,000
Max=Black=324.85 Min=Red=4.95 Missing=Gray


Hospitalizations Per 100,000
Max=Black=2173.65 Min=Red=0.09 Missing=Gray

< 0.05

```python
top_25_pop_deathspc = [deaths_pc[state] for state in top_25_population]
bottom_25_pop_deathspc = [deaths_pc[state] for state in bottom_25_population]
pop_pval = ttest_ind(top_25_pop_deathspc, bottom_25_pop_deathspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 25 vs Bottom 25 Population Deaths Per Capita: {pop_pval/2:.3f}")
```

[18] ✓ 0.2s Python

```
P-Value for Means of Top 25 vs Bottom 25 Population Deaths Per Capita: 0.048
```

> 0.05

```python
top_10_pop_deathspc = [deaths_pc[state] for state in top_10_population]
bottom_40_pop_deathspc = [deaths_pc[state] for state in bottom_40_population]
pop_pval = ttest_ind(top_10_pop_deathspc, bottom_40_pop_deathspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 10 vs Bottom 40 Population Deaths Per Capita: {pop_pval/2:.3f}")
```

[19] ✓ 0.2s Python

```
P-Value for Means of Top 10 vs Bottom 40 Population Deaths Per Capita: 0.089
```

> 0.05

```python
top_25_pop_hospspc = [hosps_pc[state] for state in top_25_population]
bottom_25_pop_hospspc = [hosps_pc[state] for state in bottom_25_population]
pop_pval = ttest_ind(top_25_pop_hospspc, bottom_25_pop_hospspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 25 vs Bottom 25 Population Hosps Per Capita: {pop_pval/2:.3f}")
```

[45] ✓ 0.2s Python

```
P-Value for Means of Top 25 vs Bottom 25 Population Hosps Per Capita: 0.230
```

> 0.05

```python
top_10_pop_hospspc = [hosps_pc[state] for state in top_10_population]
bottom_40_pop_hospspc = [hosps_pc[state] for state in bottom_40_population]
pop_pval = ttest_ind(top_10_pop_hospspc, bottom_40_pop_hospspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 10 vs Bottom 40 Population Hosps Per Capita: {pop_pval/2:.3f}")
```

[46] ✓ 0.1s Python

```
P-Value for Means of Top 10 vs Bottom 40 Population Hosps Per Capita: 0.427
```

```python
top_25_education_deathspc = [deaths_pc[state] for state in top_25_education]
bottom_25_education_deathspc = [deaths_pc[state] for state in bottom_25_education]
pop_pval = ttest_ind(top_25_education_deathspc, bottom_25_education_deathspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 25 vs Bottom 25 Education Deaths Per Capita: {pop_pval/2:.3f}")
```

[20] ✓ 0.2s Python

> 0.05

```
P-Value for Means of Top 25 vs Bottom 25 Education Deaths Per Capita: 0.383
```

```python
top_10_education_deathspc = [deaths_pc[state] for state in top_10_education]
bottom_40_education_deathspc = [deaths_pc[state] for state in bottom_40_education]
pop_pval = ttest_ind(top_10_education_deathspc, bottom_40_education_deathspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 10 vs Bottom 40 Education Deaths Per Capita: {pop_pval/2:.3f}")
```

[21] ✓ 0.1s Python

> 0.05

```
P-Value for Means of Top 10 vs Bottom 40 Education Deaths Per Capita: 0.400
```

```python
top_25_education_hospspc = [hosps_pc[state] for state in top_25_education]
bottom_25_education_hospspc = [hosps_pc[state] for state in bottom_25_education]
pop_pval = ttest_ind(top_25_education_hospspc, bottom_25_education_hospspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 25 vs Bottom 25 Education Hosps Per Capita: {pop_pval/2:.3f}")
```

[47] ✓ 0.3s Python

> 0.05

```
P-Value for Means of Top 25 vs Bottom 25 Education Hosps Per Capita: 0.167
```

```python
top_10_education_hospspc = [hosps_pc[state] for state in top_10_education]
bottom_40_education_hospspc = [hosps_pc[state] for state in bottom_40_education]
pop_pval = ttest_ind(top_10_education_hospspc, bottom_40_education_hospspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 10 vs Bottom 40 Education Hosps Per Capita: {pop_pval/2:.3f}")
```

[48] ✓ 0.1s Python

> 0.05

```
P-Value for Means of Top 10 vs Bottom 40 Education Hosps Per Capita: 0.206
```

```python
top_25_GDP_deathspc = [deaths_pc[state] for state in top_25_GDP]
bottom_25_GDP_deathspc = [deaths_pc[state] for state in bottom_25_GDP]
pop_pval = ttest_ind(top_25_GDP_deathspc, bottom_25_GDP_deathspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 25 vs Bottom 25 GDP Deaths Per Capita: {pop_pval/2:.3f}")
```
[23] ✓ 0.1s Python

```
P-Value for Means of Top 25 vs Bottom 25 GDP Deaths Per Capita: 0.039
```

< 0.05

```python
top_10_GDP_deathspc = [deaths_pc[state] for state in top_10_GDP]
bottom_40_GDP_deathspc = [deaths_pc[state] for state in bottom_40_GDP]
pop_pval = ttest_ind(top_10_GDP_deathspc, bottom_40_GDP_deathspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 10 vs Bottom 40 GDP Deaths Per Capita: {pop_pval/2:.3f}")
```
[24] ✓ 0.2s Python

```
P-Value for Means of Top 10 vs Bottom 40 GDP Deaths Per Capita: 0.057
```

> 0.05

```python
top_25_GDP_hospspc = [hosps_pc[state] for state in top_25_GDP]
bottom_25_GDP_hospspc = [hosps_pc[state] for state in bottom_25_GDP]
pop_pval = ttest_ind(top_25_GDP_hospspc, bottom_25_GDP_hospspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 25 vs Bottom 25 GDP Hosps Per Capita: {pop_pval/2:.3f}")
```
[49] ✓ 0.1s Python

```
P-Value for Means of Top 25 vs Bottom 25 GDP Hosps Per Capita: 0.160
```

> 0.05

```python
top_10_GDP_hospspc = [hosps_pc[state] for state in top_10_GDP]
bottom_40_GDP_hospspc = [hosps_pc[state] for state in bottom_40_GDP]
pop_pval = ttest_ind(top_10_GDP_hospspc, bottom_40_GDP_hospspc, equal_var=False).pvalue
print(f"P-Value for Means of Top 10 vs Bottom 40 GDP Hosps Per Capita: {pop_pval/2:.3f}")
```
[50] ✓ 0.1s Python

```
P-Value for Means of Top 10 vs Bottom 40 GDP Hosps Per Capita: 0.182
```

> 0.05

```python
blue_states_deathspc = [deaths_pc[state] for state in blue_states]
red_states_deathspc = [deaths_pc[state] for state in red_states]
pop_pval = ttest_ind(blue_states_deathspc, red_states_deathspc, equal_var=False).pvalue
print(f"P-Value for Means of Red State vs Blue State Deaths Per Capita: {pop_pval/2:.3f}")
```

[25] ✓ 0.1s Python

> 0.05

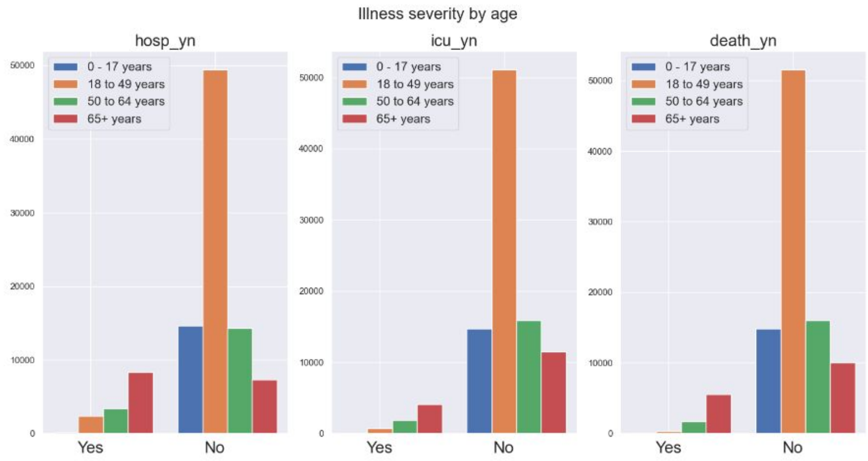
```
P-Value for Means of Red State vs Blue State Deaths Per Capita: 0.116
```

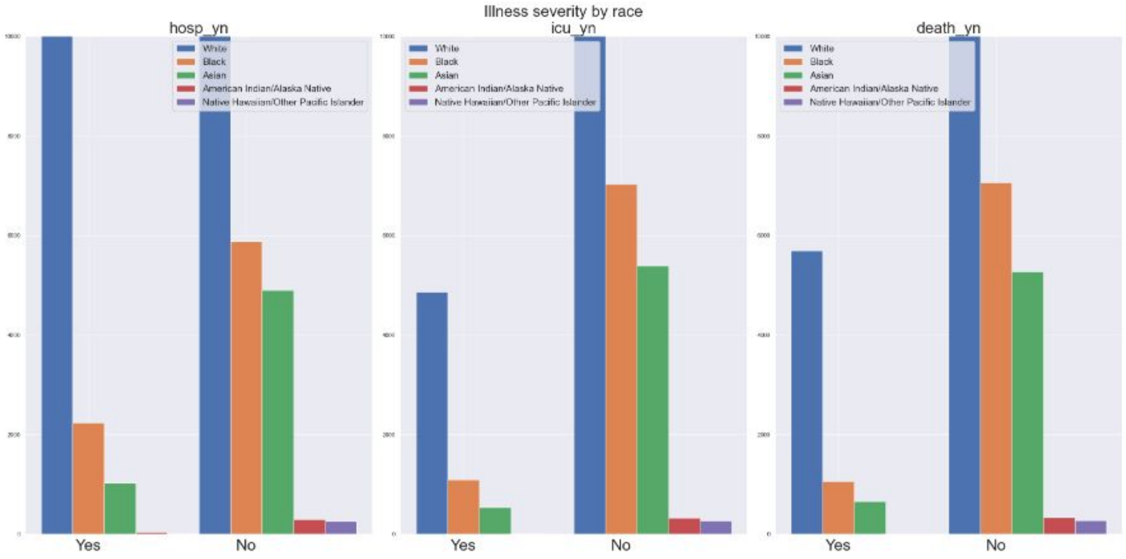
```python
blue_states_hospspc = [hosps_pc[state] for state in blue_states]
red_states_hospspc = [hosps_pc[state] for state in red_states]
pop_pval = ttest_ind(blue_states_hospspc, red_states_hospspc, equal_var=False).pvalue
print(f"P-Value for Means of Red State vs Blue State Hosps Per Capita: {pop_pval/2:.3f}")
```

[51] ✓ 0.1s Python

> 0.05

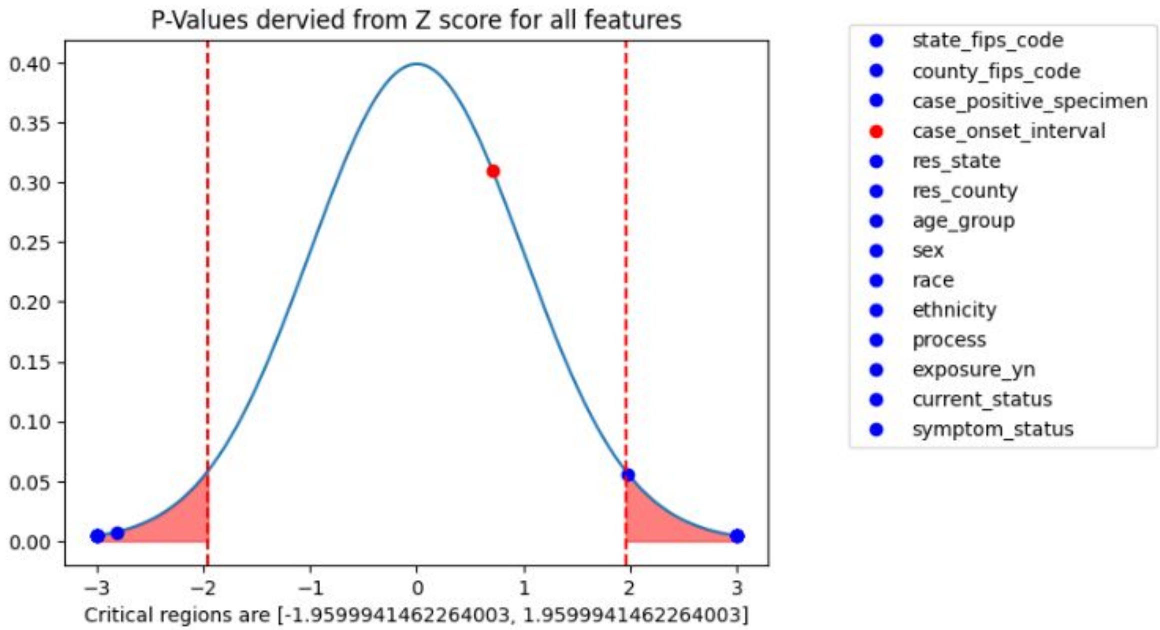
```
P-Value for Means of Red State vs Blue State Hosps Per Capita: 0.123
```



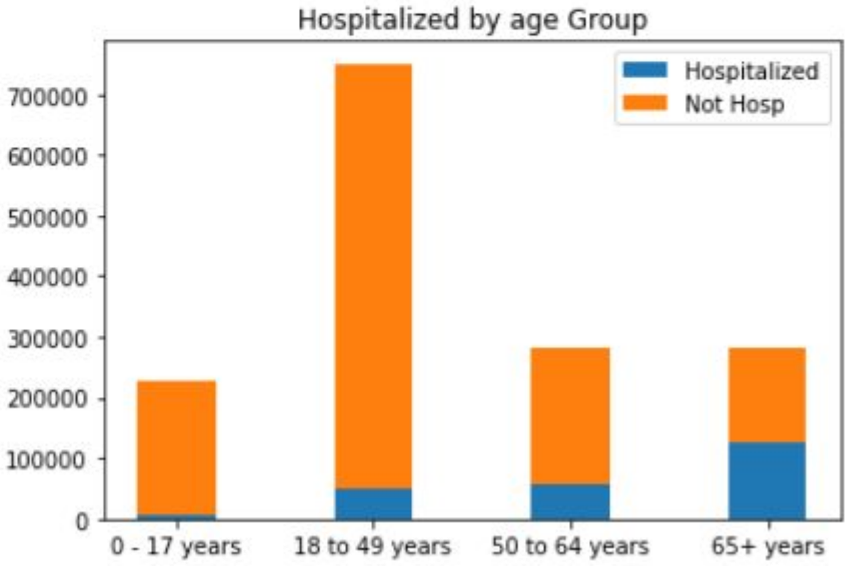


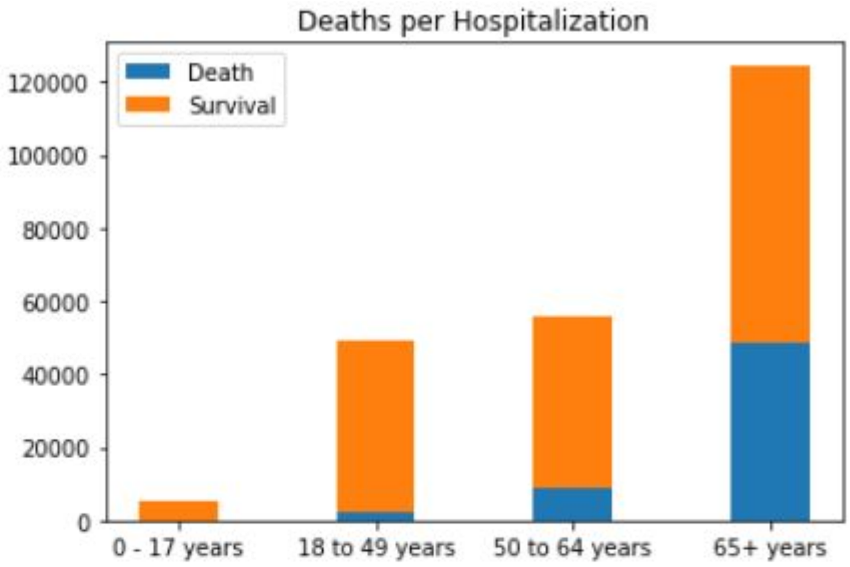
As you can see there is major imbalances in our dataset. For example there are vastly more many datapoint for people ages 18-48 then other ages.

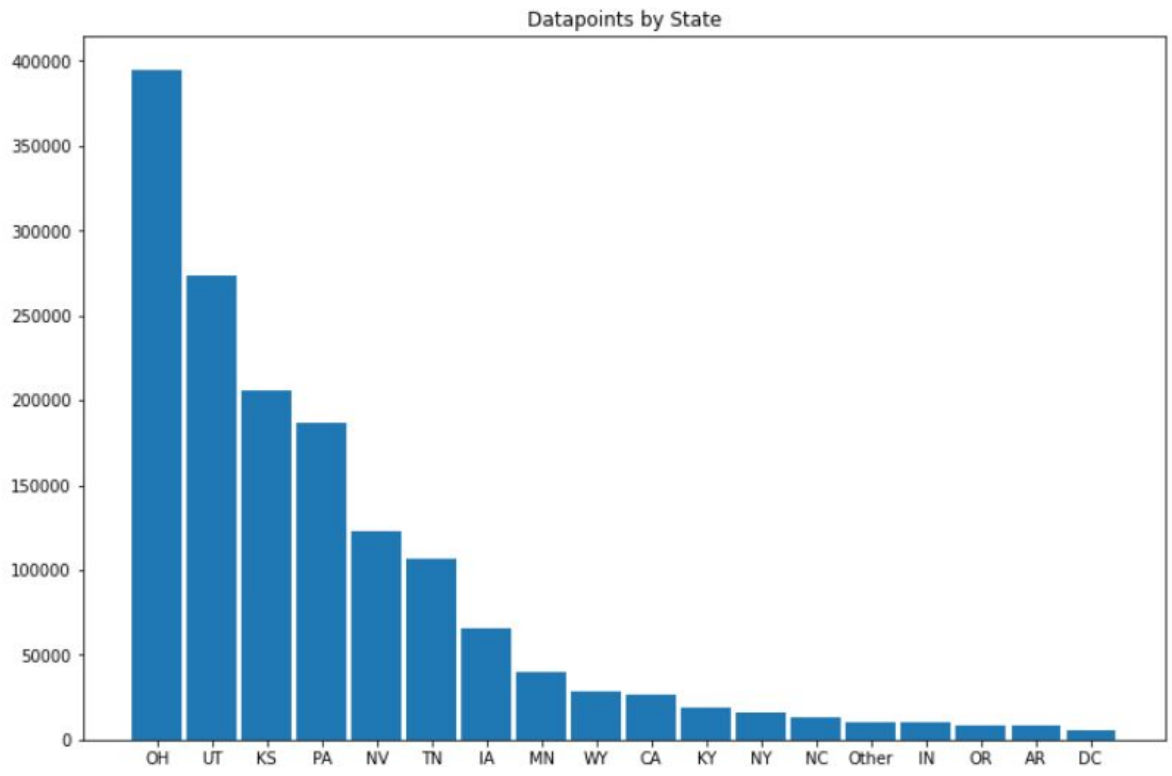
Here we can see most features are statistically significant with one being not statistically significant. This poses a challenge in feature selection since the statistical test shows that almost all features are significant.



Feature ranking based on P-values and critical regions


Hospitalized by age Group
Hospitalized
Not Hosp
700000
600000
500000
400000
300000
200000
100000
0
0 - 17 years
18 to 49 years
50 to 64 years
65+ years


Deaths per Hospitalization
Death
Survival
120000
100000
80000
60000
40000
20000
0
0 - 17 years
18 to 49 years
50 to 64 years
65+ years

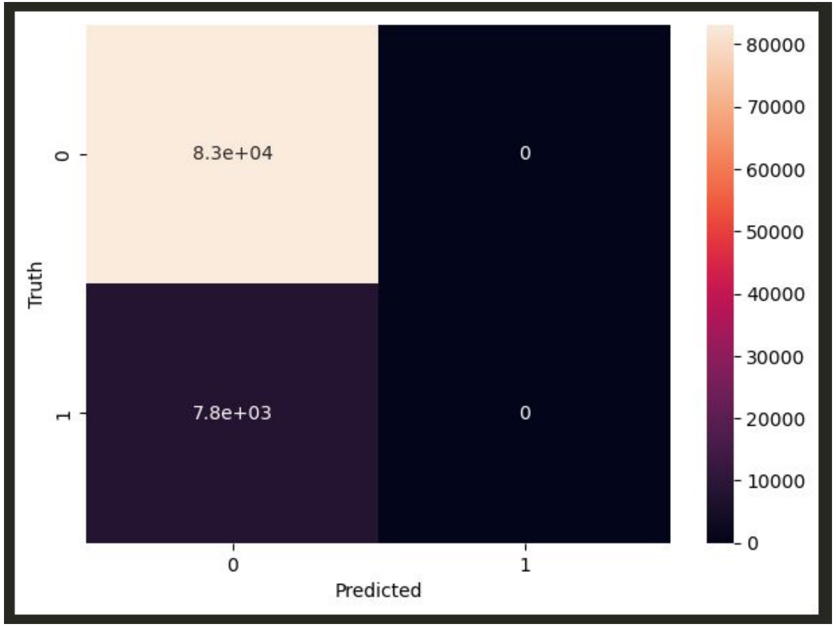

Datapoints by State
400000
350000
300000
250000
200000
150000
100000
50000
0
OH
UT
KS
PA
NV
TN
IA
MN
WY
CA
KY
NY
NC
Other
IN
OR
AR
DC

Classification

- Tried to classify hospitalization, death
- Models used: KNN, Decision Trees, NBC
- Techniques used: bagging, SMOTE, feature selection

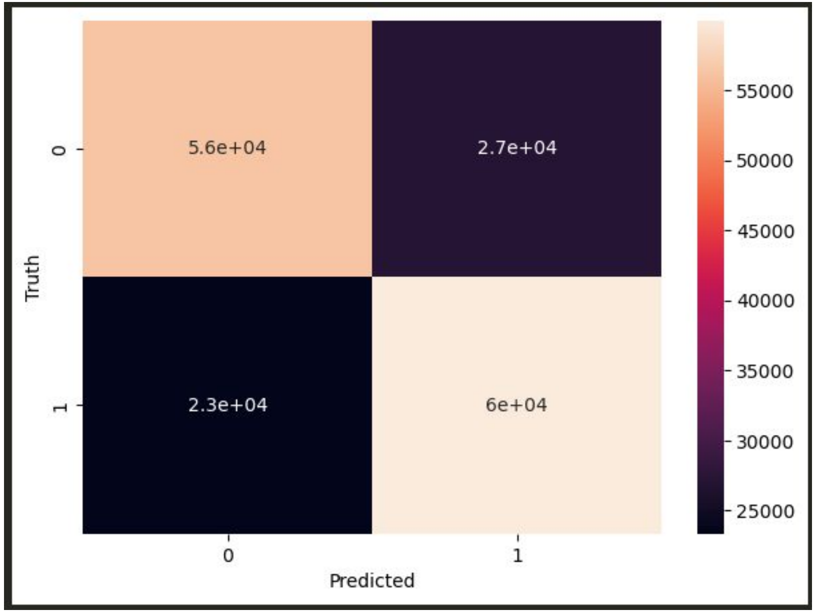
Confusion matrix - KNN (Hospitalization)

Before SMOTE



0.9138499406619489

After SMOTE



0.6975989812039334

Scores - KNN (hospitalization)

Before SMOTE

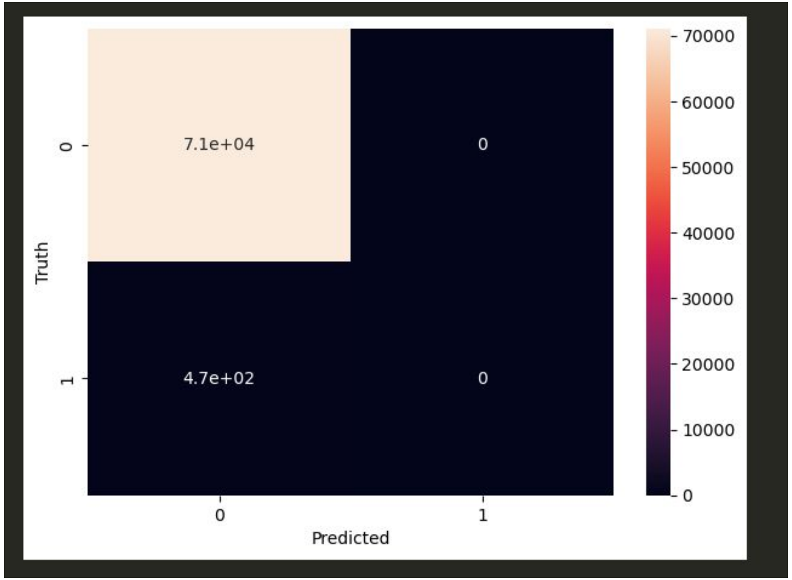
	precision	recall	f1-score	support
0	0.91	1.00	0.95	83164
1	0.00	0.00	0.00	7840
accuracy			0.91	91004
macro avg	0.46	0.50	0.48	91004
weighted avg	0.84	0.91	0.87	91004

After SMOTE

	precision	recall	f1-score	support
0	0.71	0.68	0.69	83146
1	0.69	0.72	0.70	83325
accuracy			0.70	166471
macro avg	0.70	0.70	0.70	166471
weighted avg	0.70	0.70	0.70	166471

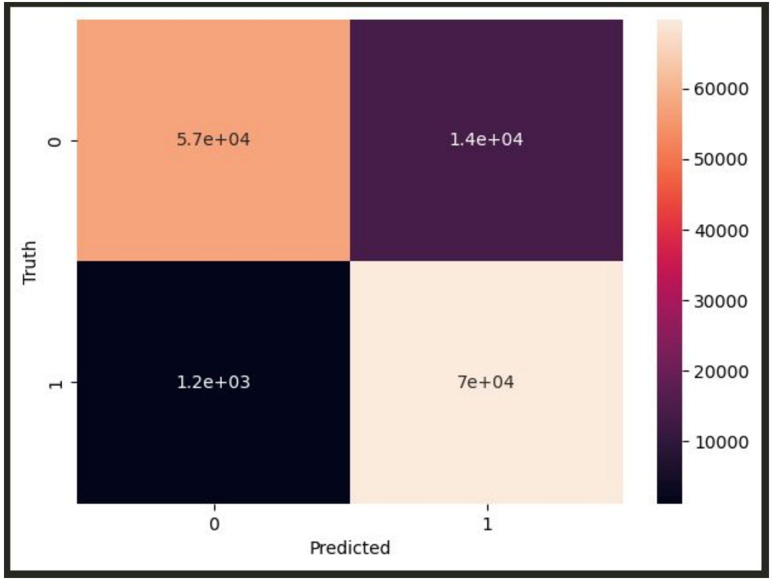
Confusion matrix - KNN (Death)

Before SMOTE



0.9934316260219411

After SMOTE



0.8942884079587988

Scores - KNN (Deaths)

Before SMOTE

	precision	recall	f1-score	support
0	0.99	1.00	1.00	71085
1	0.00	0.00	0.00	470
accuracy			0.99	71555
macro avg	0.50	0.50	0.50	71555
weighted avg	0.99	0.99	0.99	71555

After SMOTE

	precision	recall	f1-score	support
0	0.98	0.81	0.88	71199
1	0.83	0.98	0.90	70933
accuracy			0.89	142132
macro avg	0.91	0.89	0.89	142132
weighted avg	0.91	0.89	0.89	142132

KNN Summary

- Model performed marginally better with more than 20 neighbors.
- No significant improvement after 100 neighbors.

```
0.6161553664001538
```

```
0.6975989812039334
```

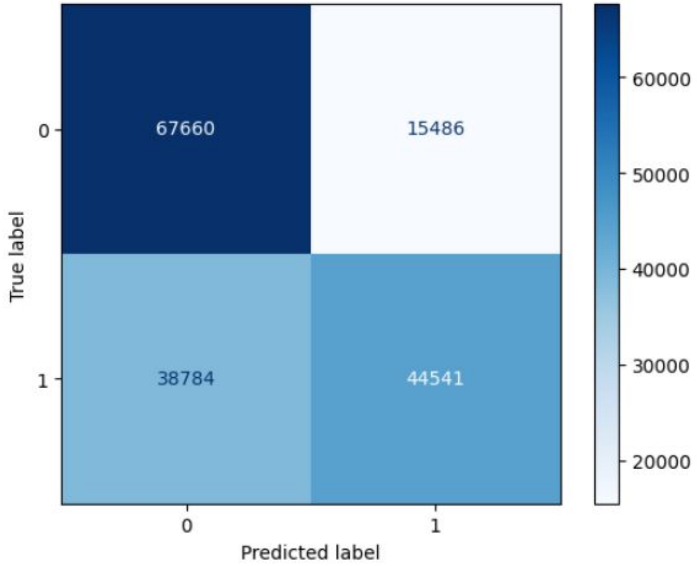
- Training time was long.
- Predicting time was actually around the same.
- Bagging - no interesting results to show.

```
knn.fit(X_train,y_train)
✓ 22.3s
```

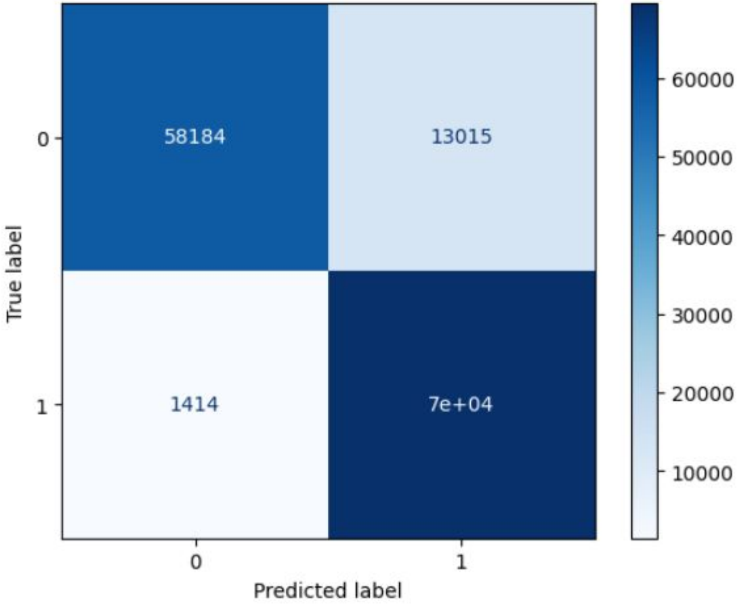
```
y_pre = knn.predict(X_test)
✓ 22.3s
```

- Conclusion: bad data

Decision Trees: Hosp. and Death



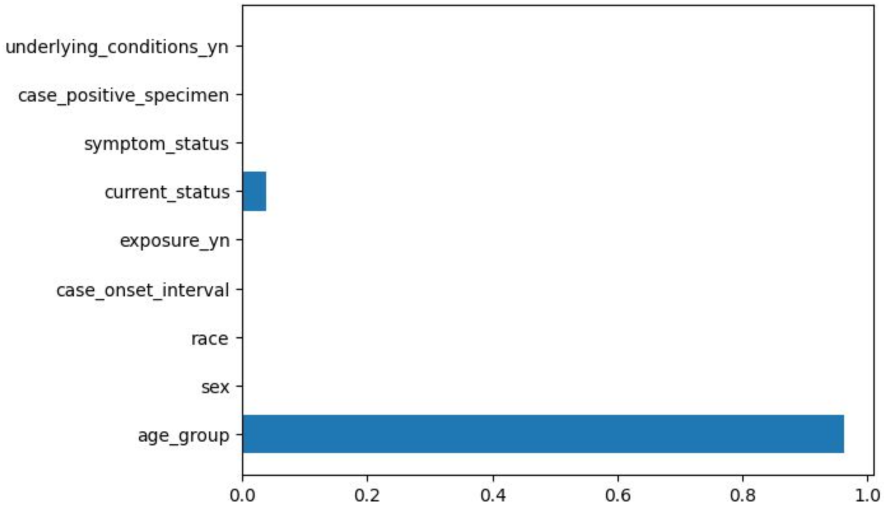
	precision	recall	f1-score	support
0	0.64	0.81	0.71	83146
1	0.74	0.53	0.62	83325
accuracy			0.67	166471
macro avg	0.69	0.67	0.67	166471
weighted avg	0.69	0.67	0.67	166471



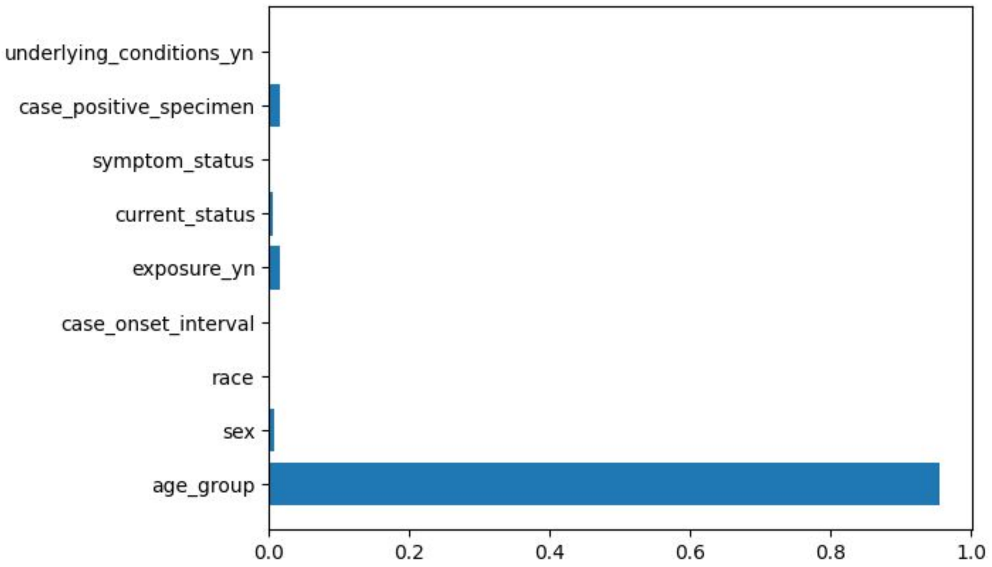
	precision	recall	f1-score	support
0	0.98	0.82	0.89	71199
1	0.84	0.98	0.91	70933
accuracy			0.90	142132
macro avg	0.91	0.90	0.90	142132
weighted avg	0.91	0.90	0.90	142132

Tree Feature Importance

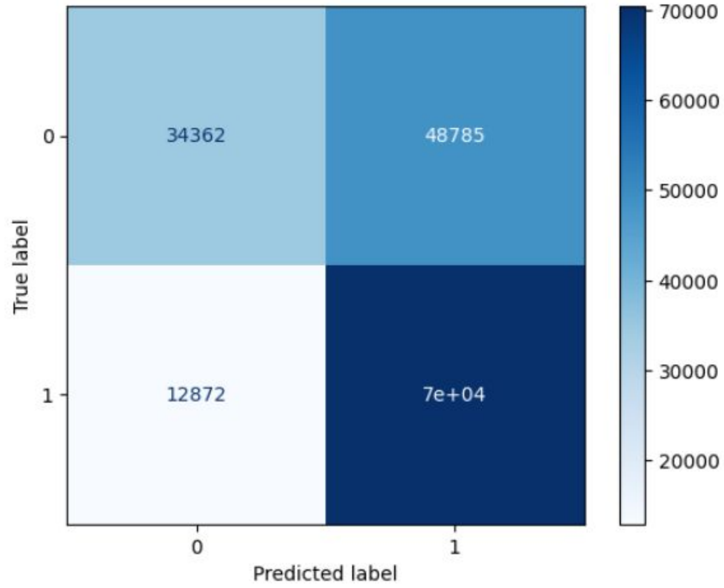
Hospitalization



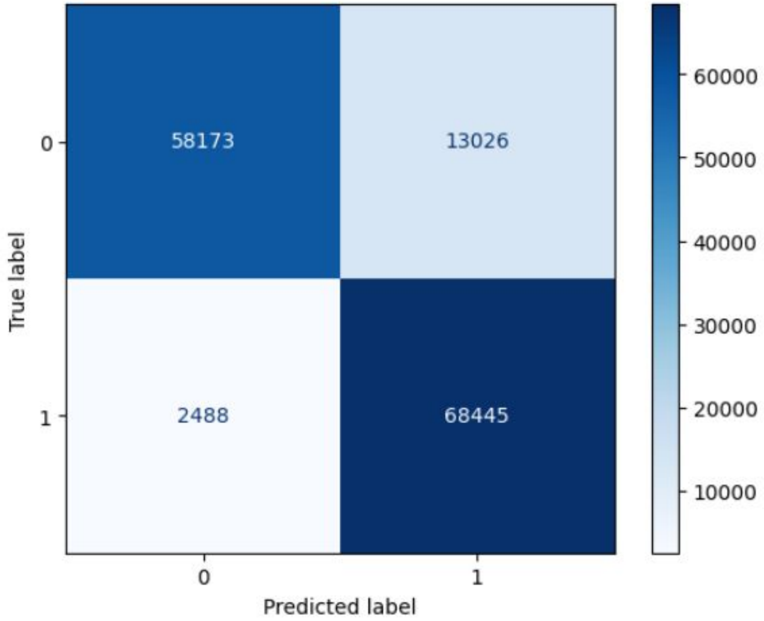
Death



NBC: Hosp. and Death



	precision	recall	f1-score	support
0	0.73	0.41	0.53	83147
1	0.59	0.85	0.70	83324
accuracy			0.63	166471
macro avg	0.66	0.63	0.61	166471
weighted avg	0.66	0.63	0.61	166471



	precision	recall	f1-score	support
0	0.96	0.82	0.88	71199
1	0.84	0.96	0.90	70933
accuracy			0.89	142132
macro avg	0.90	0.89	0.89	142132
weighted avg	0.90	0.89	0.89	142132

Classification: Some Conclusions and Concerns

- Data points did not take into account time or geography
- No incorporation of vaccine impacts
- Hosp Vs. Death