Introduction

I will use the COVID-19 dataset from Kaggle, which contains features about patients who were diagnosed with COVID, including sex, age, pre-existing medical conditions etc., to predict which patients died based on the most prominent health features using two supervised learning methods, decision trees and k-nearest neighbors. A decision tree consists of answering a set of yes or no questions about a specific patient to traverse the pathways of a tree which ends in death or survival. On the other hand, k-nearest neighbors aims to similarly predict death but based on whether patients similar to the given one died. The goal of classification is to being able to categorize patients benefits the public as it brings awareness to those who are at a higher risk of mortality and should therefore take more precautions such as wearing masks in indoor spaces and avoiding crowds.

Data

The dataset that I have choosen contains characteristics about users organized into columns that describe their medical features such as sex, age, and pre-existing conditions such as pregnancy, pnemounia, diabetes, etc. The data was congregated by the Mexican government but collected by nurses at hospitals who recorded patient data. According the CDC, data about COVID-19 cases helps track the spread of the virus in different locations, identify trends in exposures, which indicates how fast the disease spreads, and outcomes, which indicates how different categories of people are affected. Similarly, different treatments are tracked for the same virus meaning the CDC can identify which ones are effective. Tracking the effect of the disease can also help establish a safe environment for healthcare workers who treat the patients (Covid-19 Data, CDC).

Here is a sample of the data:

```
import pandas as pd
df = pd.read_csv('Covid Data.csv')
print(df.head())
```

```
   USMER  MEDICAL_UNIT  SEX  PATIENT_TYPE   DATE_DIED  INTUBED  PNEUMONIA  \
0      2             1    1             1  03/05/2020       97          1
1      2             1    2             1  03/06/2020       97          1
2      2             1    2             2  09/06/2020        1          2
3      2             1    1             1  12/06/2020       97          2
4      2             1    2             1  21/06/2020       97          2

   AGE  PREGNANT  DIABETES  ...  ASTHMA  INMSUPR  HIPERTENSION  OTHER_DISEASE  \
0   65         2         2  ...       2        2             1              2
1   72        97         2  ...       2        2             1              2
2   55        97         1  ...       2        2             2              2
3   53         2         2  ...       2        2             2              2
4   68        97         1  ...       2        2             1              2

   CARDIOVASCULAR  OBESITY  RENAL_CHRONIC  TOBACCO  CLASIFFICATION_FINAL  ICU
0               2        2              2        2                     3   97
1               2        1              1        2                     5   97
2               2        2              2        2                     3    2
3               2        2              2        2                     7   97
4               2        2              2        2                     3   97

[5 rows x 21 columns]
```

There are 21 columns of patient features with some columns being boolean, where 1 means yes and 2 means no or in the case of sex, 1 is female while 2 is male.

I will clean the data by replacing the date of death with a boolean since supervised learning requires labels for classification. 1 will represent death while 0 will represent survival. For my analysis, I will only be using 500,000 random rows because the original sample of 1 million is too compututational heavy to run in an appropriate amount of time.

```
death = df['DATE_DIED'].apply(lambda x: 0 if x=='9999-99-99' else 1) #converts death date to 1 and survival to 0
df['DEATH'] = death
df = df.drop('DATE_DIED',axis=1) #removes death date from columns
df = df.sample(50000)
```

Methodology

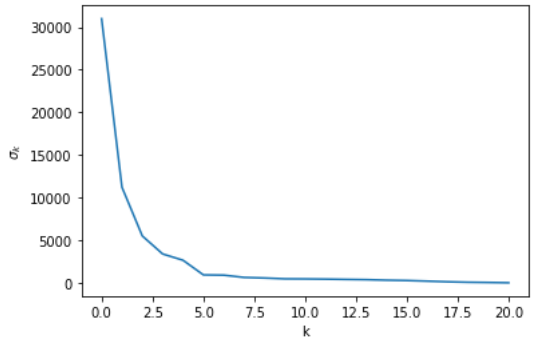
The first analysis conducted data is SVD. The purpose of SVD is to reduce the number of features of the dataset, therefore reducing the number of dimensions which avoids the problem associated with high dimensionality during a k-nearest neighbors analysis. The dataset has many features that are appear to be correlated with each other, such as tobacco use and presence of cardiovascular disease. Reducing the

dimensionality of the dataset will allow the subsequent analysis to run extremely faster in comparison with limited error. The second analysis conducted will be a comparison between decision trees and k-nearest neighbors to determine which method predicts with higher accuracy that a patient has died or survived.

```
import numpy as np
import matplotlib.pyplot as plt

u, s, vt = np.linalg.svd(df, full_matrices = False)
plt.plot(s)
plt.xlabel('k')
plt.ylabel('$\sigma_k$')
```

```
Text(0, 0.5, '$\\sigma_k$')
```



Based on the graph above, the effective rank is 5. Therefore, SVD with 5 features or dimensions will be able to capture the majority of the data with low error.

```
from sklearn.decomposition import TruncatedSVD #performing truncated svd because of the size of the dataset
svd = TruncatedSVD(n_components = 5, random_state = 1)
svd_covid = svd.fit_transform(df)
```

Analysis

My analysis began with the implementation of a decision tree using the sklearn package. The independent variable are the values with the features from the SVD, while my dependent variable is the column indicating whether the patient had died or survived. I used train-test split with a test size of 20% of the original data to fit the model and calculated the predicted labels. I stored the accuracy of the labels in a list and repeated the process by using different random training data with the same sized test data.

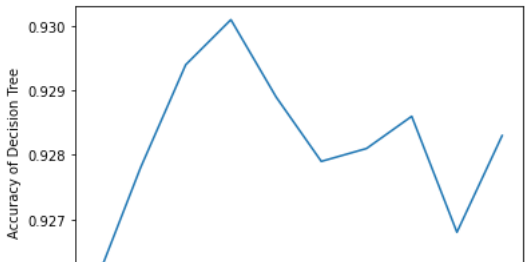
```
import sklearn.model_selection as model_selection
from sklearn.tree import DecisionTreeRegressor
from sklearn import metrics

x = svd_covid
y = df['DEATH']
accuracy = []

for i in range(10):
  x_train, x_test, y_train, y_test = model_selection.train_test_split(x, y, test_size = 0.2, random_state = i)
  decisionTree = DecisionTreeRegressor()
  decisionTree = decisionTree.fit(x_train,y_train)
  predictions = decisionTree.predict(x_test)
  accuracy.append(metrics.accuracy_score(y_test, predictions).round(4))


plt.xlabel('Random State')
plt.ylabel('Accuracy of Decision Tree')
plt.plot(range(10),accuracy)
plt.show()

print('Mean:', sum(accuracy)/len(accuracy))
print('Range:', min(accuracy), 'to', max(accuracy))
```



Based on the graph, the decision tree was extremely precise and accurate. The accuracy ranged from 0.926 at the lowest and 0.932 at the highest for multiple trees, which appears to be the ceiling of accuracy. The mean of the accuracy of the decision tree is 0.931.

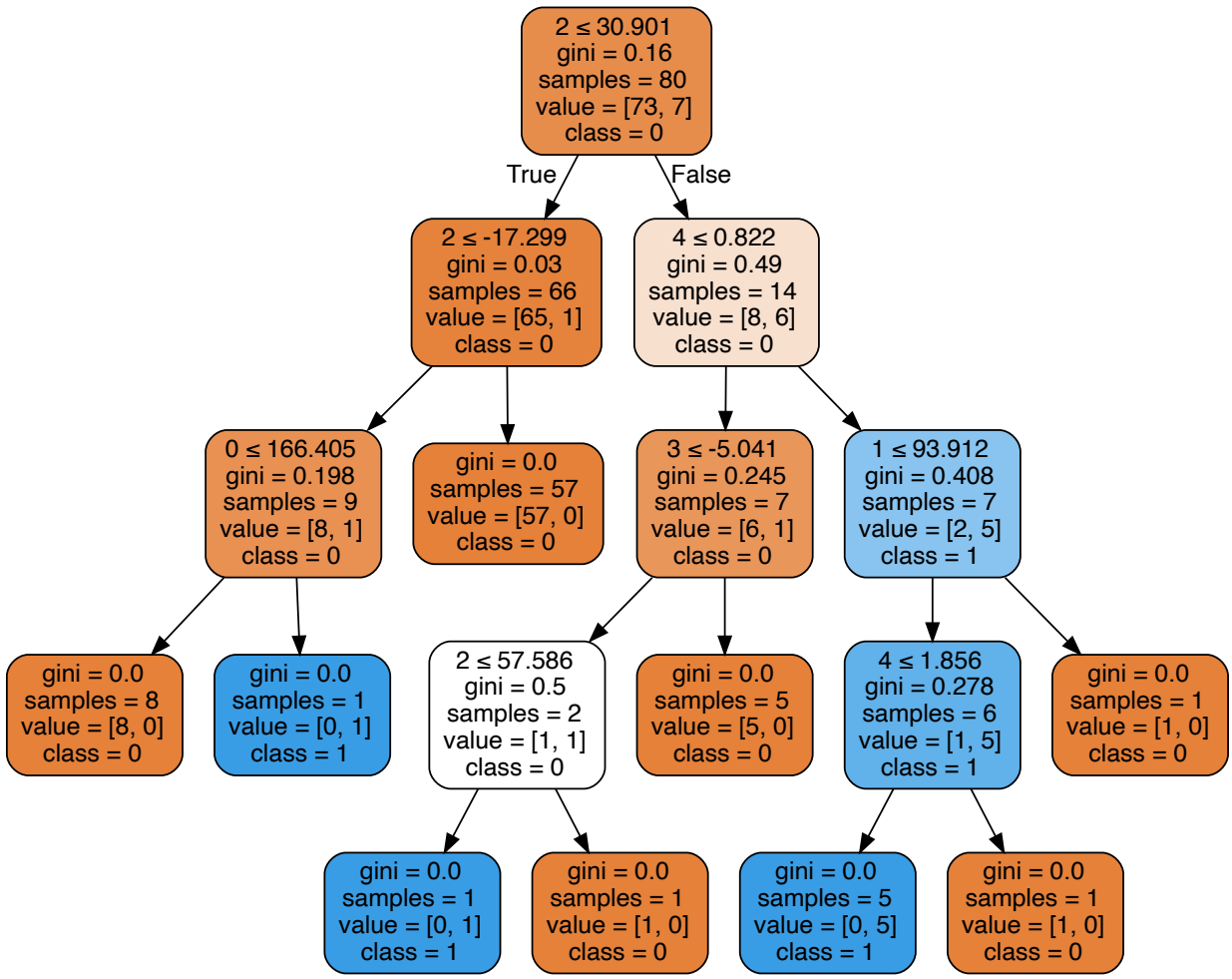
```
Mean: 0.9281900000000001
```

```
import graphviz
from sklearn import tree

x_train, x_test, y_train, y_test = model_selection.train_test_split(x[:100], y[:100], test_size = 0.2, random_state = 0)

clf = tree.DecisionTreeClassifier()
clf = clf.fit(x_train, y_train)

dot_data = tree.export_graphviz(clf, out_file=None)
graph = graphviz.Source(dot_data)
dot_data = tree.export_graphviz(clf, out_file=None, feature_names=range(5),  class_names=['0','1'],  filled=True, rounded=True,  s
graph = graphviz.Source(dot_data)
graph
```



This is a sample of a decision tree using the first 100 samples of the SVD data.

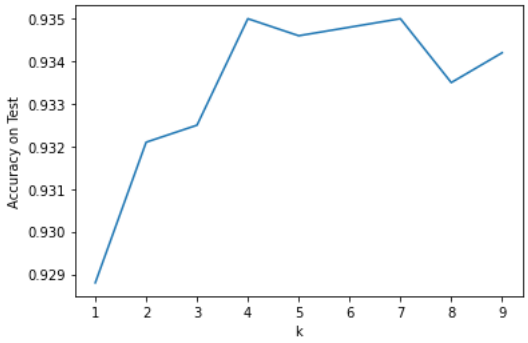
```
import sklearn.model_selection as model_selection
from sklearn.neighbors import KNeighborsClassifier

acc = []
```

```
for i in range(1, 10):
    test_acc = []
    for j in range(10):
        X_train, X_test, y_train, y_test = model_selection.train_test_split(x, y, test_size = 0.2, random_state = 1)
        knn = KNeighborsClassifier(n_neighbors = i)
        knn.fit(X_train, y_train)
        test_acc.append(knn.score(X_test, y_test))
    acc.append(sum(test_acc)/len(test_acc))

plt.xlabel('k')
plt.ylabel('Accuracy on Test')
plt.plot(range(1,10), acc)
plt.show()

print('Mean:', sum(acc)/len(acc))
print('Range:', min(acc), 'to', max(acc))
```




```
Mean: 0.933388888888889
Range: 0.9288000000000001 to 0.9350000000000003
```

Results

The results of the analysis conclude that k-nearest neighbor is better than decision trees at predicing whether a given patient died or survived COVID-19. The range of accuracy of decision trees is 0.926 to 0.931. The range of accuracy of k-nearest neighbors ranged from 0.929 to 0.94. The standard deviation of the accuracy of the k-nearest neighbors was greater, but the mean accuracy was greater at 0.934 compared to 0.931. The accuracy of decision trees and k-nearest neighbors is extremely similar and to be expected given that the dataset had SVD performed on it. SVD alleviated the problem of higher dimensionality, which significantly improves the performance of k-nearest neighbors since the points will not be significantly far apart. If k-nearest neighbors was performed on the original dataset, then due to the number of dimensions being 21, it would have taken far longer and produced a significantly worse accuracy. SVD also increased the accuracy for decision trees because by reducing the number of features, the tree is not as prone to overfitting and was able to minimize the number of leaf nodes possible. Due to SVD, k-nearest neighbors and decision tree classifications were able to produce extremely similar models in terms of accuracy with k-nearest neighbors producing the model with the greatest mean accuracy.

Conclusion

I started by cleaning a dataset of COVID-19 patients that contained features about their identity such as sex, age etc. and pre-existing medical conditions such as the prescence of a cardiovascular disease, whether they used tobacco regularly, etc. I performed SVD to reduce the number of features based on the effective rank of the matrix and used the new reduced matrix to build a decision tree and a k-nearest neighbor model to predict whether a specific patient died or survived. I built the model and tested the accuracy which was 0.931 for decision trees and 0.936 for k-nearest neighbors. I concluded that both models produced extremely precise and accurate results, but k-nearest neighbor was superior. The viability of both of these models was predicated on SVD because without it, the k-nearest neighbors model would suffer from high dimensionality which would cause the accuracy to drop since the points would all be equally far away from each other while the decision tree would be prone to overfitting on the training data, which means less accuracy on test data.

References

Link to dataset: https://www.kaggle.com/datasets/meirnizri/covid19-dataset?resource=download

Lecture Notes: 20, 21, 22

“FAQ: Covid-19 Data and Surveillance.” Centers for Disease Control and Prevention, Centers for Disease Control and Prevention, 5 Oct. 2022, https://www.cdc.gov/coronavirus/2019-ncov/covid-data/faq-surveillance.html.

https://www.datacamp.com/tutorial/decision-tree-classification-python https://scikit-learn.org/stable/modules/tree.html#classification

✓ 38s completed at 7:58 PM