Srinivas Reddy Pachika
IST 707 – HW5

OBJECTIVE

My objective of this assignment is to analyze what are the major factors causing death in COVID-19 patients, we know that different patients are showing different symptoms when affected by corona virus. We will try to analyze does showing any specific symptoms put them at higher risk over the others. And finally, we will build a model to predict the risk of death in a patient with given symptoms.

DATA

I took my data (patient medical data for novel coronavirus covid-19) from WOLFRAM DATA REPOSITORY which collected data randomly from different patients belonging to 35 different countries from 3 Feb 2020 to 4 May 2020.

DATA CLEANING AND PREPARATION

The actual data has 21 columns and more than 10,000 observations but more than 9000 observations have missing values in more than 10 columns which are very important in my case for building a model. And hence I had to narrow down my observations to 761.

I have made some of the below mentioned steps in excel while others were performed using pandas.

INDEPENDENT VARIABLES

1) **Age:**

17	30
18	30
19	58
20	65
21	65
22	82
23	Interval[{30, 39}]
24	Interval[{30, 39}]

Some of the age observations were in intervals while others were normal continuous values, so I have divided all the ages into 10 bins from 1 to 100 with each bin size =10.

2) **Gender:**

1 – male

0 – Female

Srinivas Reddy Pachika
IST 707 – HW5

3) Country

Country
Entity["Country", "China"]
Entity["Country", "China"]
Entity["Country", "China"]
Entity["Country", "China"]
Entity["Country", "China"]
Entity["Country", "China"]
Entity["Country", "China"]
Entity["Country", "China"]
Entity["Country", "China"]
Entity["Country", "China"]
Entity["Country", "China"]

Original Data set had country variable as a string shown above but using excel functions I parsed out country names from each observation and created a new column.

- In the Original data set there is a column named symptoms which is a string with different symptoms shown by each patient, From this column I copied every different symptom from all observations a created a new column for each one of those and converted them to binary categorical variables. 1 is given if that symptom is observed in that patient and zero is given if the patient does not show that symptom.

Symptoms column from original data

Symptoms
{"cough", "fatigue", "fever", "sputum", "myalgias"}
{"shortness of breath", "respiratory symptoms"}
Missing["NotAvailable"]
{"cough", "sputum", "fever"}
{"cough"}
{"chest pain", "cough", "fever"}
{"chest pain", "shortness of breath", "fever"}
{"chest pain", "shortness of breath", "fever"}
{"cough", "fever", "shortness of breath"}
{"cough", "fever"}
{"cough", "diarrhea", "fever", "rhinorrhea", "sneezing"}
{"cough", "diarrhea", "fever", "rhinorrhea", "sneezing"}
{"diarrhea", "fever", "rhinorrhea", "chest pain", "sore throat"}
{"diarrhea", "fever", "rhinorrhea", "chest pain", "sore throat"}
{"fatigue"}
{"cough", "fever", "sore throat"}
{"cough", "fever", "sore throat"}
{"chest pain", "cough", "fever", "gasp"}
{"cough", "fever", "weakness"}
{"cough", "fever", "weakness"}

Srinivas Reddy Pachika
IST 707 – HW5

Individual columns for Each symptom after conversion

E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
Diarrhea	Sputum	Chills	Shortness	Other Res	Malaise	Rhinorrhe	Headache	Pneumon	mild symp	chest pain	sore throa	cough	fever	sneezing	Pneumon
0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0
0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
1	0	0	0	0	0	1	0	0	0	0	0	1	1	1	0
1	0	0	0	0	0	1	0	0	0	0	0	1	1	1	0

The newly created columns after conversion are

4) **Diarrhea**
5) **Sputum**
6) **Shortness of Breath**
7) **Other Respiratory Symptoms**
8) **Malaise**
9) **Rhinorrhea**
10) **Headache**
11) **chest pain**
12) **sore throat**
13) **cough**
14) **fever**
15) **sneezing**
16) **Pneumonitis**

Each one of the above columns is 1 if the patient has that symptom and 0 if he does not have it.

17) **Lives in Wuhan**

1 - if the patient lives in Wuhan
0 – if the patient does not live in Wuhan

18) **Travel History Location**

This column is the name of the country if the patient has travelled to any country if past 15 days. This column has 24 distinct values.

19) **Chronic DiseaseQ**

1 - if the patient has any Chronic disease history
0 - if the patient has No Chronic disease history

Srinivas Reddy Pachika
IST 707 – HW5

TARGET VARIABLE

20) DeathQ

1- If the patient is dead
0- If the patient recovered or still under treatment

CHECKING THE DATA DISTRIBUTION

- The next step I did after reading my data in pandas is to check the distribution of target variable. I found that 93% of the data belongs to class DeathQ = 0 while only 6% of data belongs to class DeathQ = 1.
- This means that out data is **not balanced.** Feeding imbalanced data to our classifier can make it biased in favor of the majority class, simply because it did not have enough data to learn about the minority and hence to overcome this I will be using **SMOTE** (Synthetic Minority Over-sampling Technique) technique.

HANDLING MISSING VALUES

- I found only two variables with missing values and the are “Age” which has 23 missing values and the other one is “Travel History Location” which has 143 missing values.
- Since the age column is already divided into bins and taking mean or median would not be an appropriate estimation of age and hence, I have decided to drop these 23 observations.
- In case of variable “Travel History Location” I don not want to lose information and hence I have replaced all the null values with value “Unknown”.

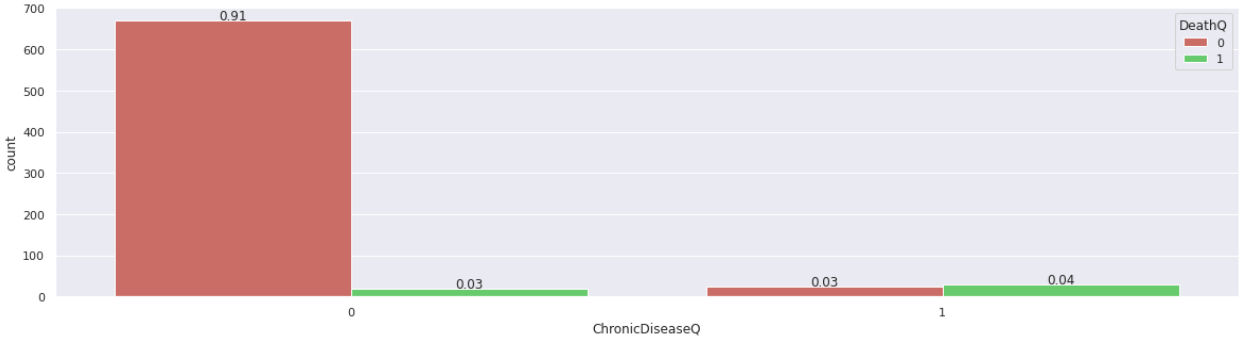
EXPLORATORY DATA ANALYSIS

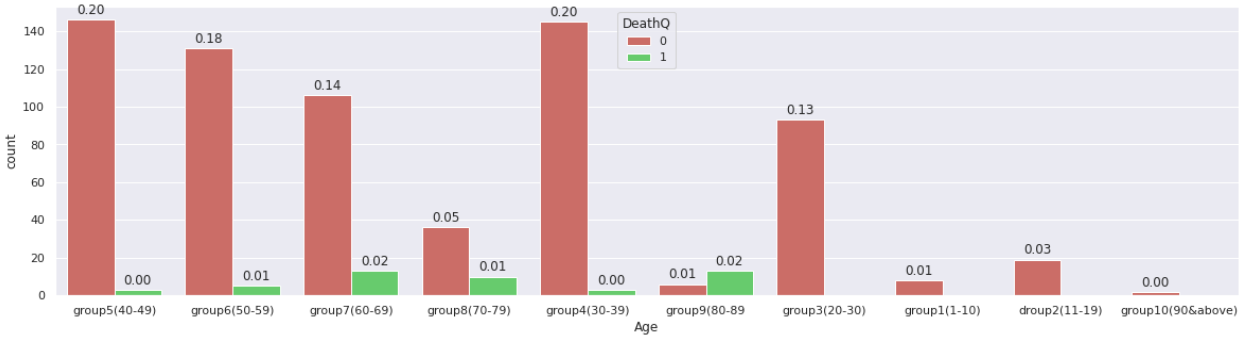
- In the exploratory analysis my intention was to see how deaths are related to each individual variable and try to find any insignificant variables. I plotted count plots with each individual variable and some of my important observations are as below.

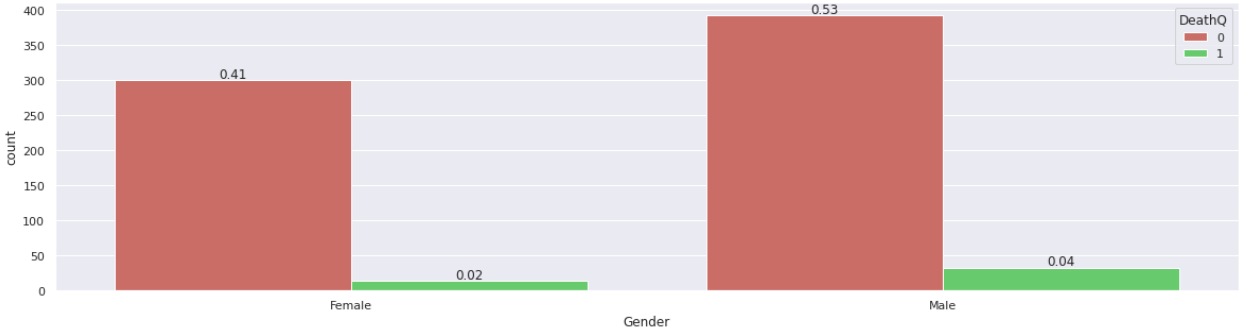
1) out of total 6% died, 4% of the population have some chronic diseases
2) I noticed high death rate among older population. Out of 6% total deaths, 5% are among the population with age group 60 and above
3) Out of total 6% deaths, 4% are men and 2% are women
4) there are zero deaths from patients reporting Diarrhea, sputum, chills, Other Respiratory Symptoms, Malaise, Rhinorrhea, sneezing and hence these variables might be insignificant. I will consider dropping them after confirming my results from Logistic regression.

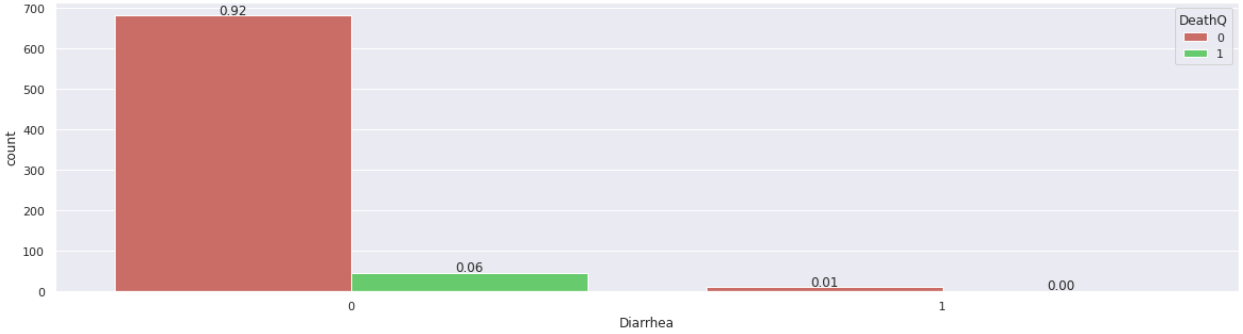
Srinivas Reddy Pachika
IST 707 – HW5

I am attaching few visualizations in the report, but full visualizations can be found in my Jupiter notebook

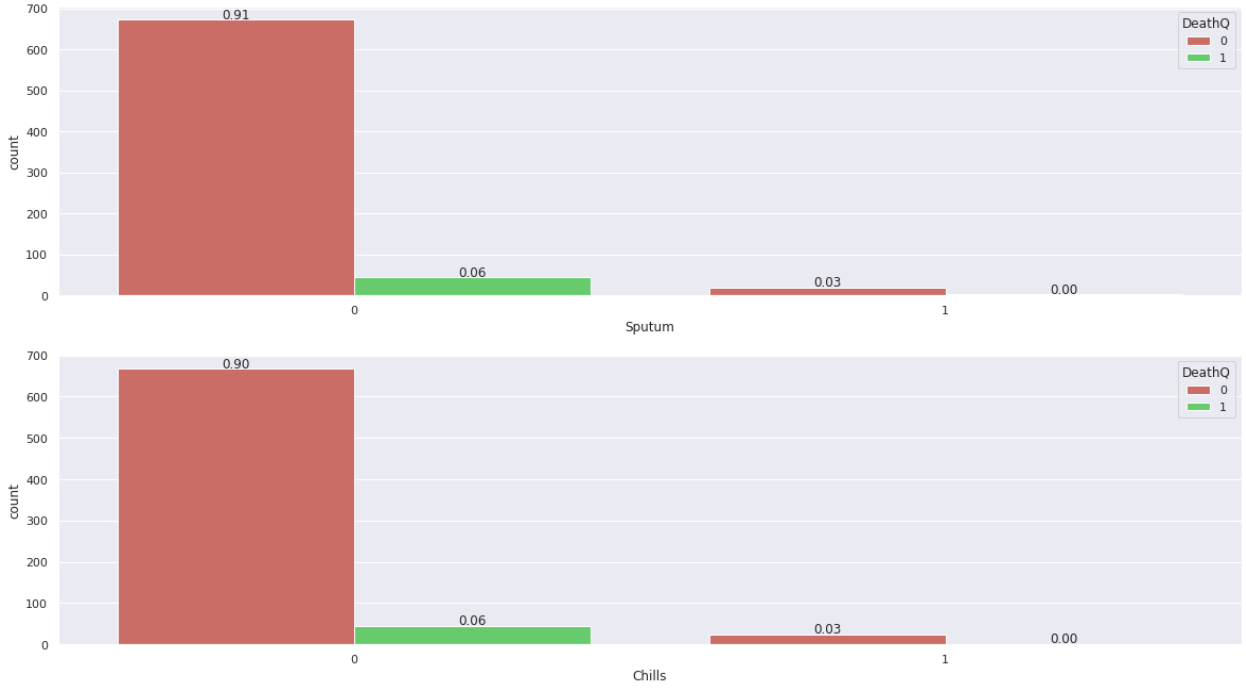








Srinivas Reddy Pachika
IST 707 – HW5



FEATURE ENGINEERING

- There might be a possibility that each individual symptom may not cause the death in covid patients but a combination of symptoms may be fetal and hence I have created a new column which is the sum of all the individual symptoms.

```
df["combined_symptoms"] = df['Diarrhea'] + df ['Sputum'] + df['Chills'] + df['Shortness of Breath'] + df ['Other Resperatory Symptoms'] + df['Malaise'] + df['Rhinorrhea'] + df['Headache'] + df[
```

```
df.head()
```

Other Resperatory Symptoms	Malaise	Rhinorrhea	Headache	Pneumonia	mild symptoms	chest pain	sore throat	cough	fever	sneezing	Pneumonitis	LivesInWuhan	TravelHistoryLocation	ChronicDiseaseQ	DeathQ	combined_symptoms
0	0	0	0	0	0	0	0	1	1	0	0	1	unknown	1	1	4
0	0	0	0	0	0	0	0	1	1	0	0	1	unknown	0	1	3
0	0	0	0	0	0	0	0	1	0	0	0	1	unknown	0	1	1
0	0	0	0	0	0	1	0	1	1	0	0	1	unknown	1	0	4
0	0	0	0	0	0	1	0	0	1	0	0	1	China	0	0	3

Srinivas Reddy Pachika
IST 707 – HW5

- Next, I used Label encoder to encode all my categorical variables (Country, Gender and Travel History Location.)

```
# Import label encoder
from sklearn import preprocessing

# label_encoder object knows how to understand word labels.
label_encoder = preprocessing.LabelEncoder()

# Encode labels
df['Age']= label_encoder.fit_transform(df['Age'])
```

```
df['Country']= label_encoder.fit_transform(df['Country'])
```

```
df['Gender']= label_encoder.fit_transform(df['Gender'])
```

```
df['TravelHistoryLocation']= label_encoder.fit_transform(df['TravelHistoryLocation'])
```

- Then I used standard scalar to scale all values of my features between 0 and 1
- Then as a next step I did my train test split with 0.3 as my test split. And then performed SMOTE to oversample data to handle imbalace in original data.
- As a next step I wanted to include only variables which have significant impact and hence I decided to run a logistic regression using stat modules.
- Using backward elimination method and by observing Psedo R squared and Pvalue for individual variables I decided the significance of variables.
- When I first run my full model in which all the independent variables were included Psedo R squared value was 0.41 and it included all 23 variables
- Then I performed backward elimination and after multiple iterations the best model also had an Psedo R-squared value = 0.41 but the number of features were reduced to 15
- I choose my alpha here as 0.05 and hence I concluded a variable as significant if p-value is less than 0.05 at 95% confidence according to the model with reduced features the varibles which had p-value less than 0.05 were Gender, LivesInWuhan, country, fever,TravelHistoryLocation,chronicDiseaseQ, combines symptoms, chills, shortness of breadth, chest pain and headache

Srinivas Reddy Pachika
IST 707 – HW5

Data distribution after SMOTE oversampling

SMOTE

```
print("Before OverSampling, counts of label '1': {}".format(sum(y_train == 1)))
print("Before OverSampling, counts of label '0': {} \n".format(sum(y_train == 0)))

# import SMOTE module from imblearn library
# pip install imblearn (if you don't have imblearn in your system)
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state = 2)
X_train_res, y_train_res = sm.fit_sample(X_train, y_train.ravel())

print('After OverSampling, the shape of train_X: {}'.format(X_train_res.shape))
print('After OverSampling, the shape of train_y: {} \n'.format(y_train_res.shape))

print("After OverSampling, counts of label '1': {}".format(sum(y_train_res == 1)))
print("After OverSampling, counts of label '0': {}".format(sum(y_train_res == 0)))
```

```
Before OverSampling, counts of label '1': 33
Before OverSampling, counts of label '0': 484

After OverSampling, the shape of train_X: (968, 20)
After OverSampling, the shape of train_y: (968,)

After OverSampling, counts of label '1': 484
After OverSampling, counts of label '0': 484
```

Results of my full model (which includes all independent variables)

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  968
Model:                          Logit   Df Residuals:                      946
Method:                           MLE   Df Model:                           21
Date:                Sat, 09 May 2020   Pseudo R-squ.:                  0.4185
Time:                        01:59:18   Log-Likelihood:                -390.19
converged:                      False   LL-Null:                       -670.97
Covariance Type:            nonrobust   LLR p-value:                1.899e-105
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
x1             0.4537      0.100      4.519      0.000       0.257       0.650
x2            -0.2591      0.109     -2.387      0.017      -0.472      -0.046
x3             0.3177      0.094      3.370      0.001       0.133       0.503
x4             0.0574   8.01e+05   7.17e-08      1.000   -1.57e+06    1.57e+06
x5             0.0258   1.18e+06   2.19e-08      1.000   -2.32e+06    2.32e+06
x6            -0.0018   1.15e+06  -1.58e-09      1.000   -2.26e+06    2.26e+06
x7            -0.0084   1.21e+06  -6.97e-09      1.000   -2.37e+06    2.37e+06
x8            -0.2986   1.06e+06  -2.82e-07      1.000   -2.08e+06    2.08e+06
x9            -0.0346   1.12e+06  -3.08e-08      1.000    -2.2e+06     2.2e+06
x10           -0.2535   1.58e+06  -1.61e-07      1.000   -3.09e+06    3.09e+06
x11           -0.0760   1.34e+06  -5.68e-08      1.000   -2.62e+06    2.62e+06
x12            0.1303   1.15e+06   1.13e-07      1.000   -2.26e+06    2.26e+06
x13           -0.1698   8.44e+05  -2.01e-07      1.000   -1.65e+06    1.65e+06
x14           -0.0263   9.59e+05  -2.74e-08      1.000   -1.88e+06    1.88e+06
x15           -0.0925   1.52e+06  -6.09e-08      1.000   -2.98e+06    2.98e+06
x16            0.1310      3e+06   4.36e-08      1.000   -5.89e+06    5.89e+06
x17            0.0250   2.81e+06   8.92e-09      1.000    -5.5e+06     5.5e+06
x18            0.0775   4.65e+05   1.67e-07      1.000   -9.12e+05    9.12e+05
x19           -0.1798   1.31e+06  -1.37e-07      1.000   -2.58e+06    2.58e+06
x20            0.2831      0.102      2.778      0.005       0.083       0.483
x21            0.6580      0.091      7.203      0.000       0.479       0.837
x22            0.8392    1.5e+06    5.6e-07      1.000   -2.93e+06    2.93e+06
x23            0.0931    6.1e+06   1.53e-08      1.000    -1.2e+07     1.2e+07
==============================================================================
```

Srinivas Reddy Pachika
IST 707 – HW5

Results of my Restricted model (Model with best Pseudo R-squared after multiple iterations by backward elimination)

```
print(answer_6.summary())
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  968
Model:                          Logit   Df Residuals:                      953
Method:                           MLE   Df Model:                           14
Date:                Sat, 09 May 2020   Pseudo R-squ.:                 0.4177
Time:                        02:22:27   Log-Likelihood:                -390.71
converged:                       True   LL-Null:                       -670.97
Covariance Type:            nonrobust   LLR p-value:                1.330e-110
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
x1             0.0394      0.049      0.808      0.419      -0.056       0.135
x2             0.4856      0.193      2.518      0.012       0.108       0.864
x3             0.3563      0.149      2.396      0.017       0.065       0.648
x4            -0.1449      0.017     -8.314      0.000      -0.179      -0.111
x5            -2.1112      0.249     -8.494      0.000      -2.598      -1.624
x6             0.0191      0.258      0.074      0.941      -0.486       0.524
x7             0.0846      0.009      9.046      0.000       0.066       0.103
x8             2.6910      0.409      6.578      0.000       1.889       3.493
x9             0.7933      0.484      1.638      0.101      -0.156       1.742
x10            0.3732      0.165      2.262      0.024       0.050       0.697
x11           -3.9077      1.138     -3.435      0.001      -6.137      -1.678
x12           -1.3645      0.665     -2.053      0.040      -2.667      -0.062
x13           -3.9408      1.250     -3.152      0.002      -6.391      -1.491
x14           -5.3294      1.293     -4.121      0.000      -7.864      -2.795
x15          -59.3052   1.02e+12   -5.8e-11      1.000      -2e+12       2e+12
==============================================================================
```

CHOOSING EVALUATION METRIC

In this case my priority would be minimizing false negatives because if a model gives a wrong prediction that a patient would not die then he may not be given required attention by the staff and may die. at the same time for us minimizing false positives is also important because if our model predicts that a patient is at high risk even if he is in stable condition who does not require extra attention, If he is sent to intensive care unit, then there may be shortage of beds in the hospital and some patient who is in real need of intensive care may not get it . hence, we need a balance between Precision and recall and hence I would choose F1 score which is hormonic mean of precision and recall as my evaluation metric.

Srinivas Reddy Pachika
IST 707 – HW5

Model Building

- **I have Build all my models with only 15 features which I selected after performing logistic regression and deciding the significance of variables as explained in the above mentioned steps**

Base Model:

Base model F1-scores

```
f1_df
```

	name	f1 score
0	DecisionTree :	0.905371
1	Naive Bayes :	0.717622
2	KNeighbours :	0.881013
3	LogisticRegression :	0.825553
4	XgBoostClassifier :	0.865591
5	Random Forest :	0.856369

Base Model Recall Scores:

```
recall_df
```

	name	recall score
0	DecisionTree :	0.850962
1	Naive Bayes :	0.812500
2	KNeighbours :	0.836538
3	LogisticRegression :	0.807692
4	XgBoostClassifier :	0.774038
5	Random Forest :	0.759615

Srinivas Reddy Pachika
IST 707 – HW5

Hyperparameter Tuning

- I performed my hyperparameter tuning on Three models, Decision tree and KNN

Decision Tree

- I used grid search cv with 3 fold cross validation with following hyper parameter options 'criterion': ['gini', 'entropy'], 'max_depth': [5,8,10,15,20,25,30],'max_leaf_nodes': [7,8,9,10,12,15,20]

Results:

My F1-score improved by 1% while recall stayed same after hyper parameter tuning

```
dt_tuned.fit(X_train_res3,y_train_res3)
dt_pred = dt_tuned.predict(X_test_res3)
print(classification_report(y_test_res3, dt_pred))
```

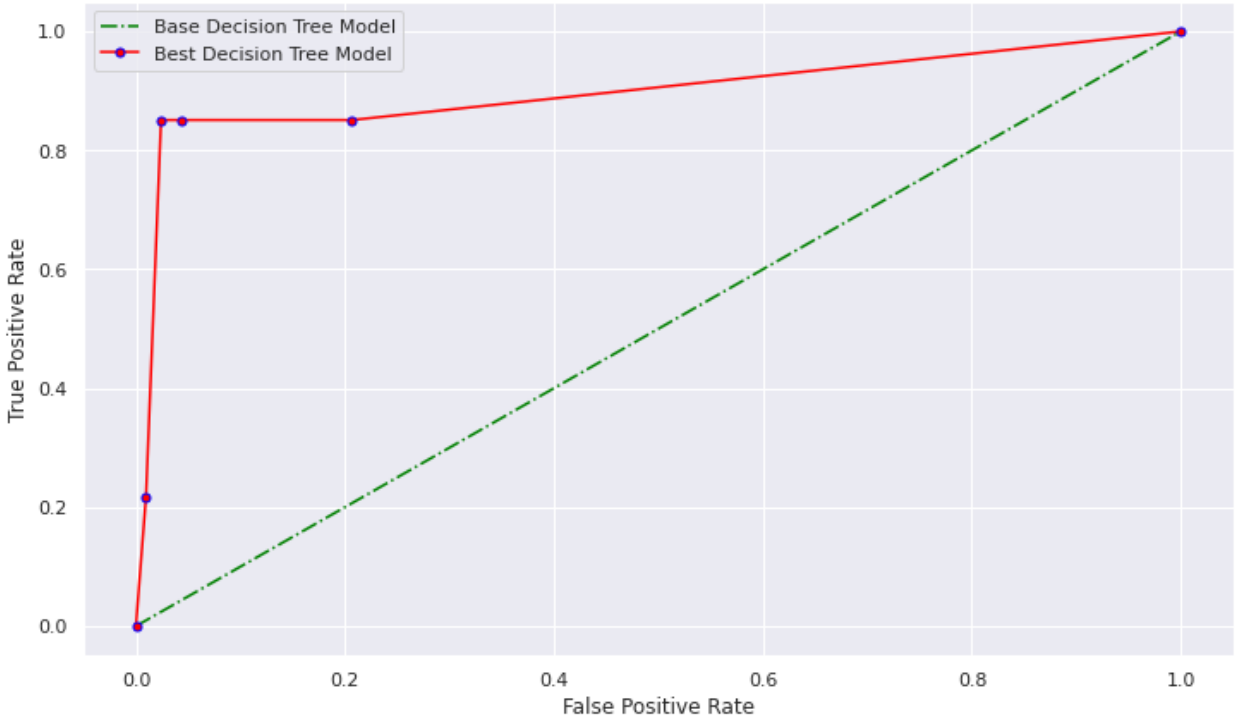
```
              precision    recall  f1-score   support

           0       0.87      0.98      0.92       208
           1       0.97      0.85      0.91       208

    accuracy                           0.91       416
   macro avg       0.92      0.91      0.91       416
weighted avg       0.92      0.91      0.91       416
```

Srinivas Reddy Pachika
IST 707 – HW5

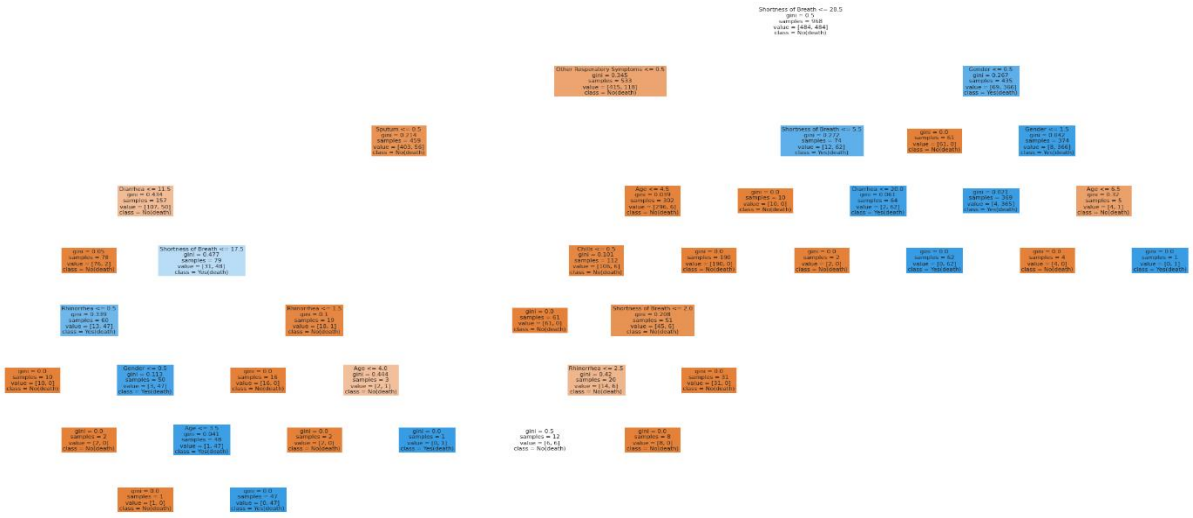
ROC-AUC

AUC = 0.89 which says that our model is pretty good classifier.

```
Best Decision Tree Model: Auc = 0.898
```

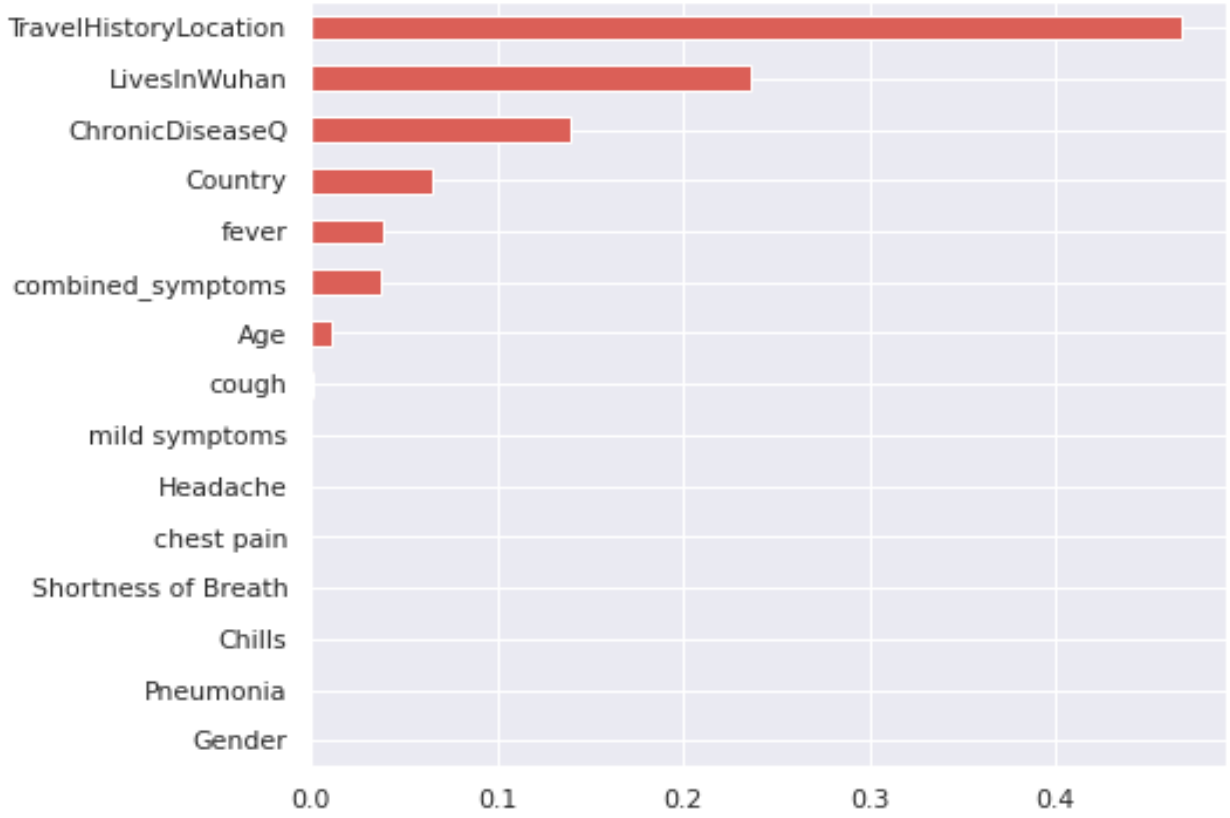


Tree Plot



Srinivas Reddy Pachika
IST 707 – HW5

Feature Importance Graph



KNN Hyperparameter tuning

- I used grid search cv with 3 fold cross validation with following hyper parameter options

  leaf_size = list(range(1,10))
  n_neighbors = list(range(1,10))
  p=[1,2]

- Best model parameters
  eaf_size=1, metric='minkowski', n_neighbors=3, p=1,

Srinivas Reddy Pachika
IST 707 – HW5

Results:

My F1 score increased by ~ 3%

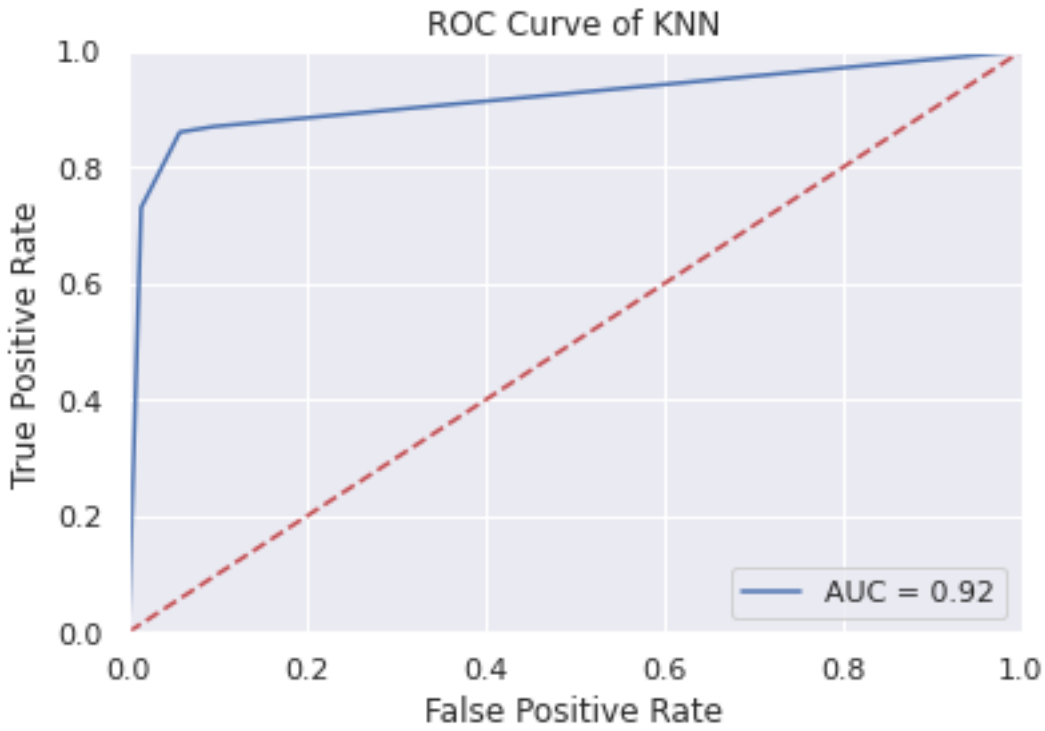
My recall score is increased by ~ 3%

```
knn_tuned.fit(X_train_res3,y_train_res3)
knn_pred = knn_tuned.predict(X_test_res3)
print(classification_report(y_test_res3, knn_pred))
```

```
              precision    recall  f1-score   support

           0       0.87      0.94      0.91       208
           1       0.94      0.86      0.90       208

    accuracy                           0.90       416
   macro avg       0.90      0.90      0.90       416
weighted avg       0.90      0.90      0.90       416
```

AUC- ROC

My AUC = 0.92 which indicates my model is very good at classifying.



Srinivas Reddy Pachika
IST 707 – HW5