Predicting Diabetes

With Machine Learning

Group 3:
Luke, Nikhil, Aleksey, Maxiel, Darrick, Samantha, Ruqayyah

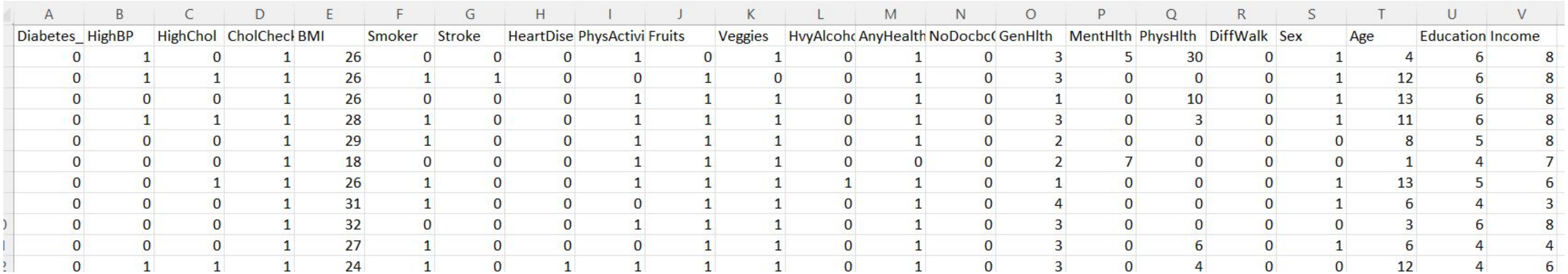
Introduction

Diabetes is a serious chronic disease in which individuals lose the ability to effectively regulate levels of glucose in the blood and can lead to reduced quality of life and life expectancy.

The Behavioral Risk Factor Surveillance System (BRFSS) is a health-related telephone survey that is collected annually by the CDC.

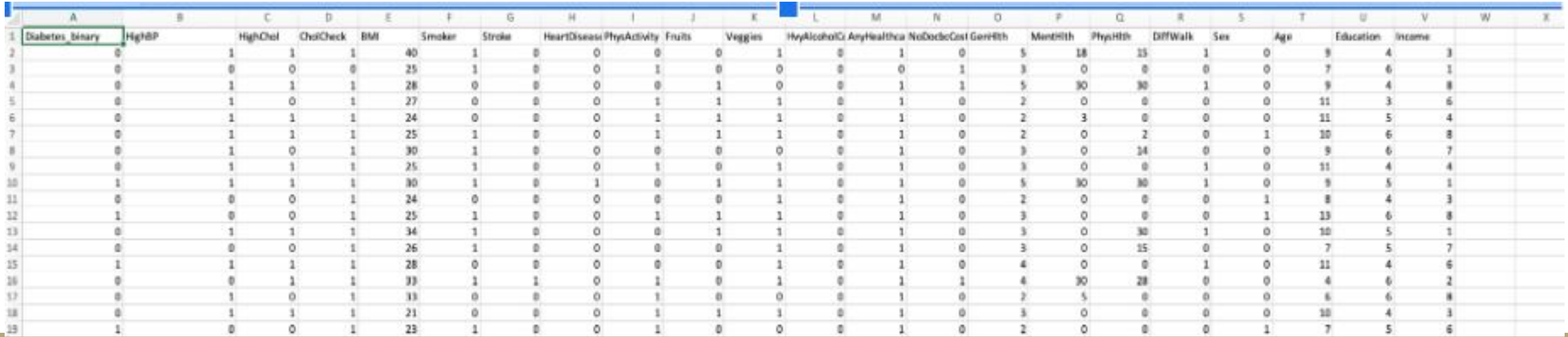
The Survey collects responses from over 400,000 Americans on health-related risk behaviors, chronic health conditions, and the use of preventative services.

diabetes_binary_5050split_health_indicators_BRFSS2015.csv



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
Diabetes_	HighBP	HighChol	CholChecl	BMI	Smoker	Stroke	HeartDise	PhysActivi	Fruits	Veggies	HvyAlcohc	AnyHealth	NoDocbc(	GenHlth	MentHlth	PhysHlth	DiffWalk	Sex	Age	Education	Income
0	1	0	1	26	0	0	0	1	0	1	0	1	0	3	5	30	0	1	4	6	8
0	1	1	1	26	1	1	0	0	1	0	0	1	0	3	0	0	0	1	12	6	8
0	0	0	1	26	0	0	0	1	1	1	0	1	0	1	0	10	0	1	13	6	8
0	1	1	1	28	1	0	0	1	1	1	0	1	0	3	0	3	0	1	11	6	8
0	0	0	1	29	1	0	0	1	1	1	0	1	0	2	0	0	0	0	8	5	8
0	0	0	1	18	0	0	0	1	1	1	0	0	0	2	7	0	0	0	1	4	7
0	0	1	1	26	1	0	0	1	1	1	1	1	0	1	0	0	0	1	13	5	6
0	0	0	1	31	1	0	0	0	1	1	0	1	0	4	0	0	0	1	6	4	3
0	0	0	1	32	0	0	0	1	1	1	0	1	0	3	0	0	0	0	3	6	8
0	0	0	1	27	1	0	0	0	1	1	0	1	0	3	0	6	0	1	6	4	4
0	1	1	1	24	1	0	1	1	1	1	0	1	0	3	0	4	0	0	12	4	6

diabetes_binary_health_indicators_BRFSS2015.csv

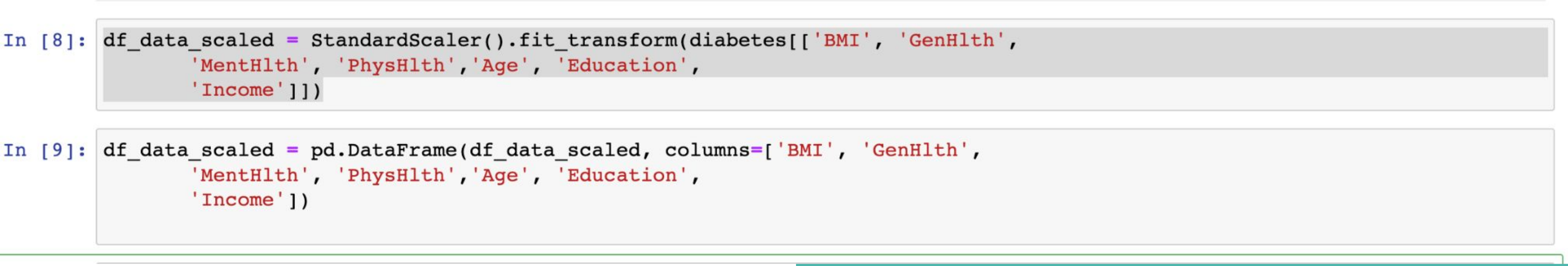


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	Diabetes_binary	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke	HeartDiseas	PhysActivity	Fruits	Veggies	HvyAlcoholC	AnyHealthca	NoDocbcCost	GenHlth	MentHlth	PhysHlth	DiffWalk	Sex	Age	Education	Income		
2	0	1	1	1	40	1	0	0	0	0	1	0	1	0	5	18	15	1	0	9	4	3		
3	0	0	0	0	25	1	0	0	1	0	0	0	0	1	3	0	0	0	0	7	6	1		
4	0	1	1	1	28	0	0	0	0	1	0	0	1	1	5	30	30	1	0	9	4	8		
5	0	1	0	1	27	0	0	0	1	1	1	0	1	0	2	0	0	0	0	11	3	6		
6	0	1	1	1	24	0	0	0	1	1	1	0	1	0	2	3	0	0	0	11	5	4		
7	0	1	1	1	25	1	0	0	1	1	1	0	1	0	2	0	2	0	1	10	6	8		
8	0	1	0	1	30	1	0	0	0	0	0	0	1	0	3	0	14	0	0	9	6	7		
9	0	1	1	1	25	1	0	0	1	0	1	0	1	0	3	0	0	1	0	11	4	4		
10	1	1	1	1	30	1	0	1	0	1	1	0	1	0	5	30	30	1	0	9	5	1		
11	0	0	0	1	24	0	0	0	0	0	1	0	1	0	2	0	0	0	1	8	4	3		
12	1	0	0	1	25	1	0	0	1	1	1	0	1	0	3	0	0	0	1	13	6	8		
13	0	1	1	1	34	1	0	0	0	1	1	0	1	0	3	0	30	1	0	10	5	1		
14	0	0	0	1	26	1	0	0	0	0	1	0	1	0	3	0	15	0	0	7	5	7		
15	1	1	1	1	28	0	0	0	0	0	1	0	1	0	4	0	0	1	0	11	4	6		
16	0	0	1	1	33	1	1	0	1	0	1	0	1	1	4	30	28	0	0	4	6	2		
17	0	1	0	1	33	0	0	0	1	0	0	0	1	0	2	5	0	0	0	6	6	8		
18	0	1	1	1	21	0	0	0	1	1	1	0	1	0	3	0	0	0	0	10	4	3		
19	1	0	0	1	23	1	0	0	1	0	0	0	1	0	2	0	0	0	1	7	5	6		

Supervised Learning Model

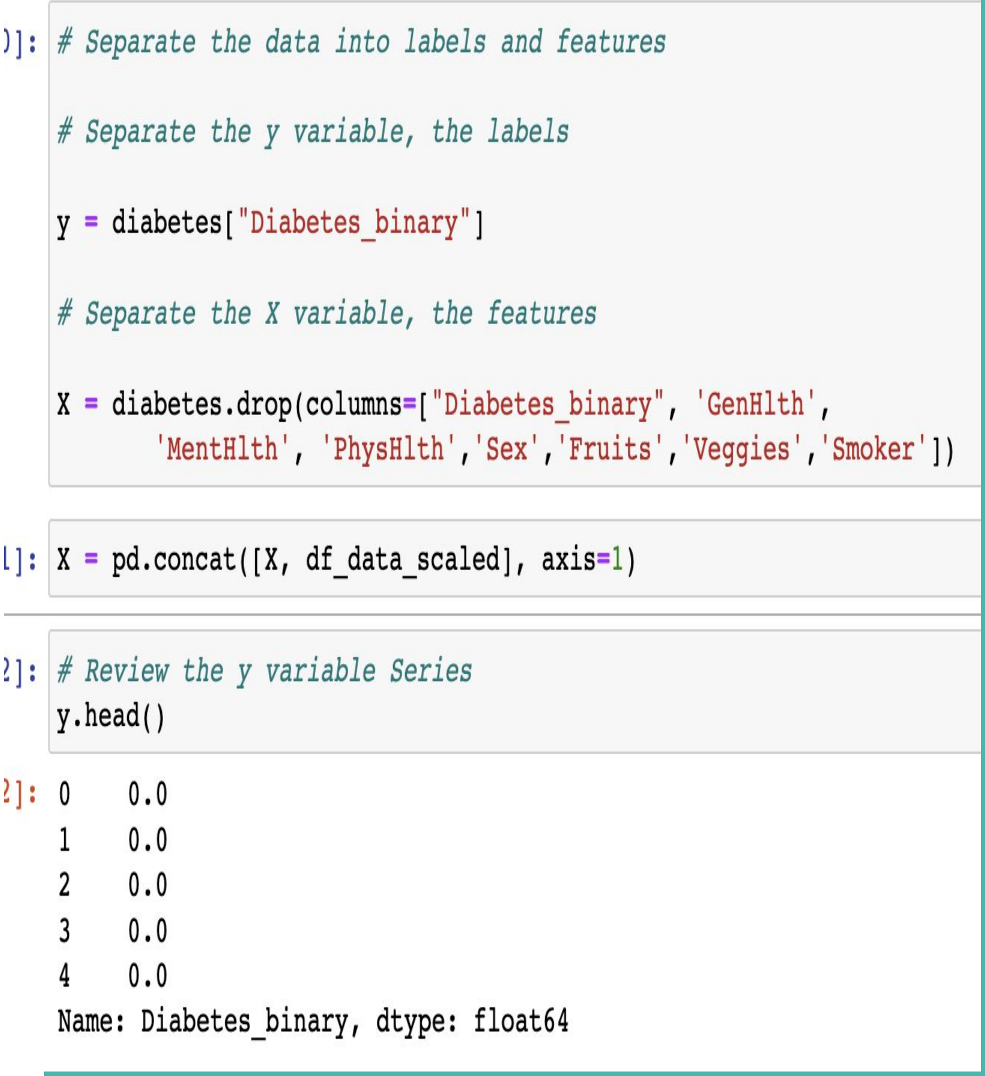
We attempted to analyze the data with a supervised model

- We scaled the non-binary variables to to ensure that all features contribute equally to the model training process and to prevent certain variables from dominating the learning process due to differences in their scales



```
In [8]: df_data_scaled = StandardScaler().fit_transform(diabetes[['BMI', 'GenHlth',
                'MentHlth', 'PhysHlth','Age', 'Education',
                'Income']])

In [9]: df_data_scaled = pd.DataFrame(df_data_scaled, columns=['BMI', 'GenHlth',
                'MentHlth', 'PhysHlth','Age', 'Education',
                'Income'])
```

- We then created the labels, setting (`y`) from the “Diabetes_binary” column, and then created the features (`X`) DataFrame from the remaining columns.
- We ran the model many times, dropping variables that did not add accuracy to the model



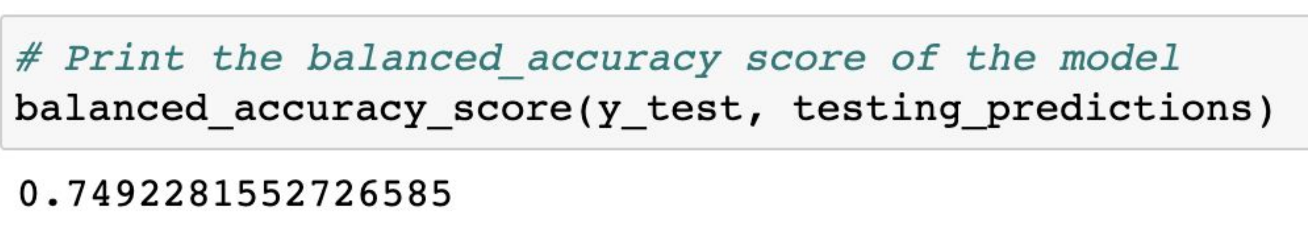
```
# Separate the data into labels and features

# Separate the y variable, the labels

y = diabetes["Diabetes_binary"]

# Separate the X variable, the features

X = diabetes.drop(columns=["Diabetes_binary", 'GenHlth',
        'MentHlth', 'PhysHlth','Sex','Fruits','Veggies','Smoker'])
```

```
X = pd.concat([X, df_data_scaled], axis=1)
```

```
# Review the y variable Series
y.head()
```

```
0    0.0
1    0.0
2    0.0
3    0.0
4    0.0
Name: Diabetes_binary, dtype: float64
```

- The balanced 50/50 data set was used, so there was no further work necessary to balance the data
- We then split the data into a training and test split and ran the logistic regression model
- A balanced accuracy score of 0.749 means that, on average, the model correctly predicts the target variable for about 74.9% of the instances

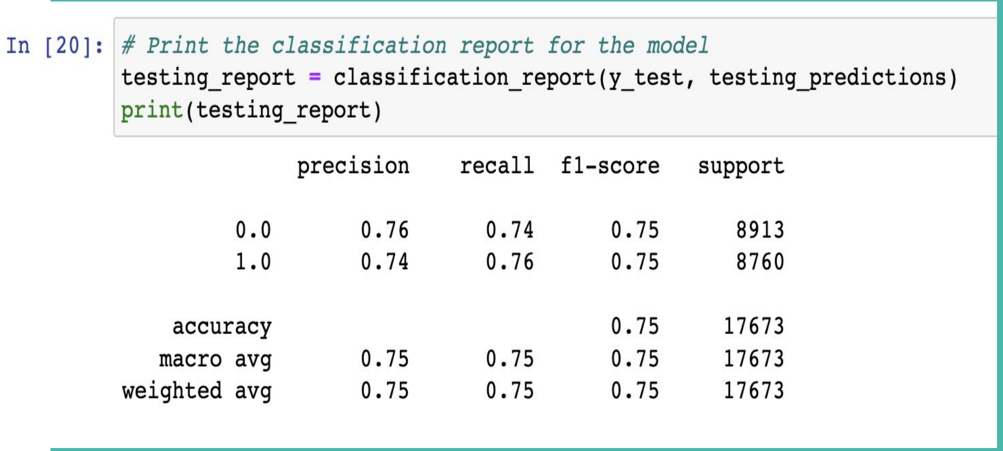


```
# Print the balanced_accuracy score of the model
balanced_accuracy_score(y_test, testing_predictions)
```

```
0.7492281552726585
```

Logistic Regression with the Original Data

- Precision: Model is correct about 76% of the time for class 0.0 and 74% for class 1.0.
- Recall: Correctly identifies about 74% of instances for class 0.0 and 76% for class 1.0.
- F1-Score: Balanced performance with F1-scores around 0.75 for both classes.
- Overall Accuracy: Model correctly predicts target variable for 75% of instances.



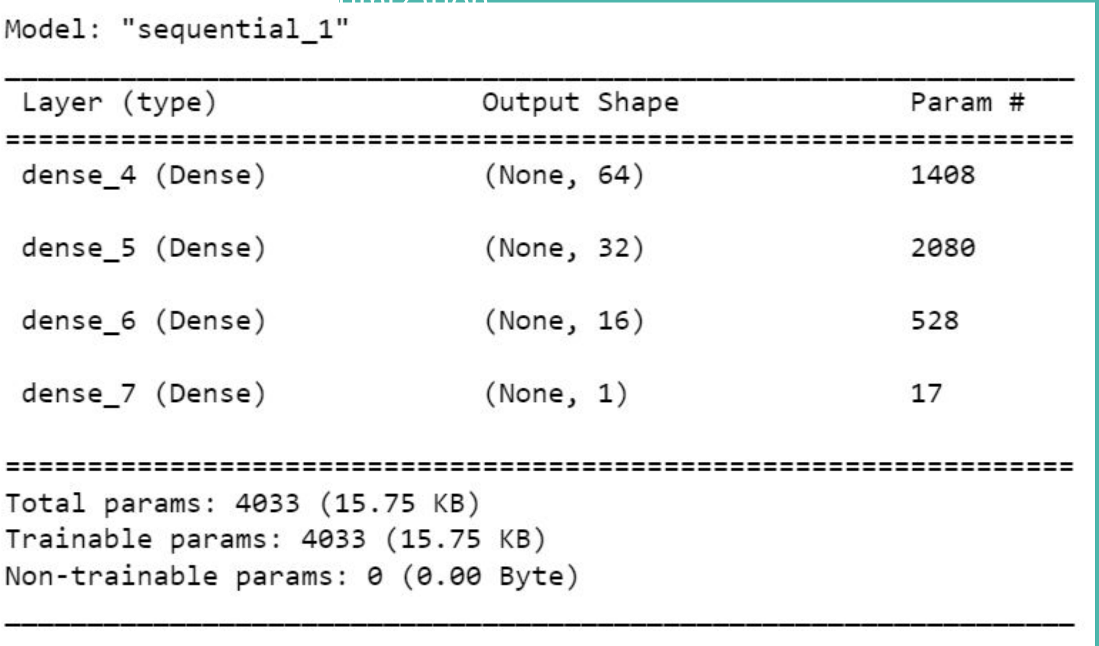
```
In [20]: # Print the classification report for the model
         testing_report = classification_report(y_test, testing_predictions)
         print(testing_report)
```

```
              precision    recall  f1-score   support

         0.0       0.76      0.74      0.75      8913
         1.0       0.74      0.76      0.75      8760

    accuracy                           0.75     17673
   macro avg       0.75      0.75      0.75     17673
weighted avg       0.75      0.75      0.75     17673
```

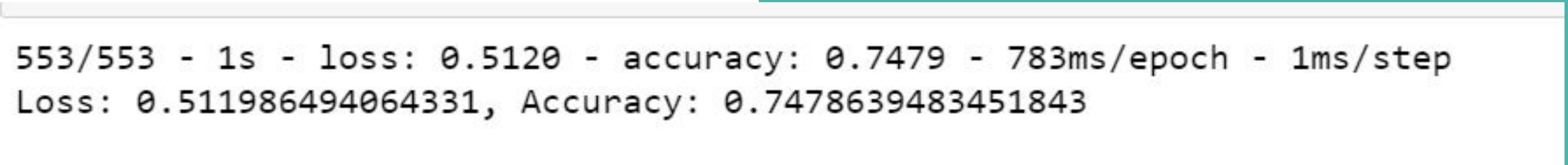
Neural Network Learning Model

timization

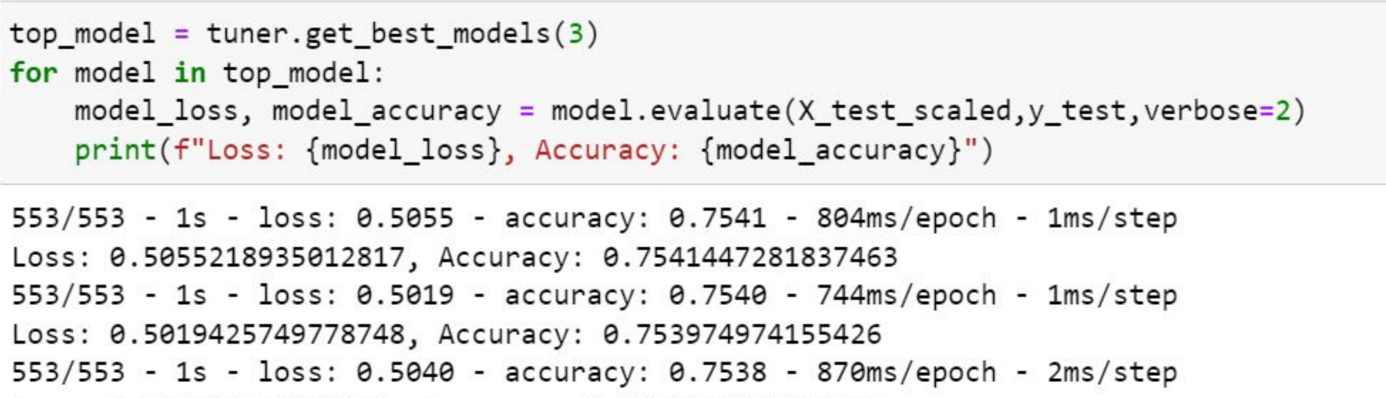
Original NN Built W/O Optimizer



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_4 (Dense)             (None, 64)                1408

 dense_5 (Dense)             (None, 32)                2080

 dense_6 (Dense)             (None, 16)                528

 dense_7 (Dense)             (None, 1)                 17

=================================================================
Total params: 4033 (15.75 KB)
Trainable params: 4033 (15.75 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

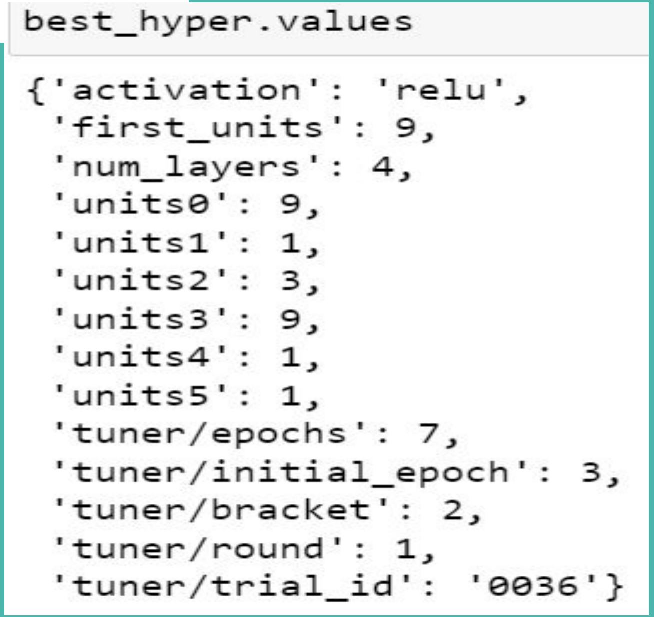


```
553/553 - 1s - loss: 0.5120 - accuracy: 0.7479 - 783ms/epoch - 1ms/step
Loss: 0.511986494064331, Accuracy: 0.747863948345184З
```



```python
top_model = tuner.get_best_models(3)
for model in top_model:
    model_loss, model_accuracy = model.evaluate(X_test_scaled,y_test,verbose=2)
    print(f"Loss: {model_loss}, Accuracy: {model_accuracy}")
```

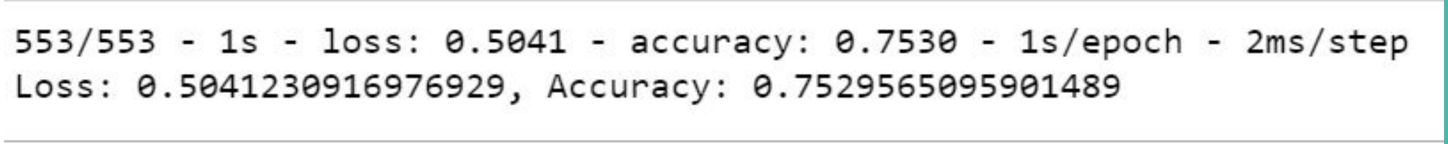
```
553/553 - 1s - loss: 0.5055 - accuracy: 0.7541 - 804ms/epoch - 1ms/step
Loss: 0.5055218935012817, Accuracy: 0.7541447281837463
553/553 - 1s - loss: 0.5019 - accuracy: 0.7540 - 744ms/epoch - 1ms/step
Loss: 0.5019425749778748, Accuracy: 0.753974974155426
553/553 - 1s - loss: 0.5040 - accuracy: 0.7538 - 870ms/epoch - 2ms/step
Loss: 0.503968358039856, Accuracy: 0.7538052201271057
```



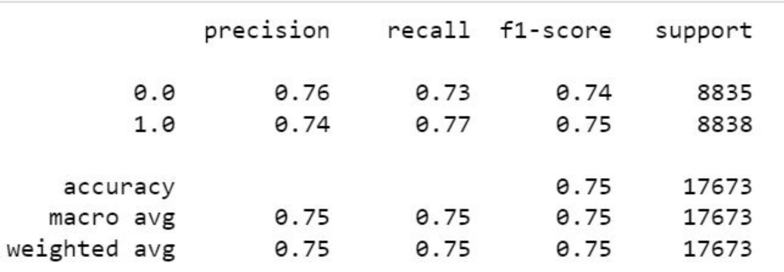
```
best_hyper.values
```

```
{'activation': 'relu',
 'first_units': 9,
 'num_layers': 4,
 'units0': 9,
 'units1': 1,
 'units2': 3,
 'units3': 9,
 'units4': 1,
 'units5': 1,
 'tuner/epochs': 7,
 'tuner/initial_epoch': 3,
 'tuner/bracket': 2,
 'tuner/round': 1,
 'tuner/trial_id': '0036'}
```

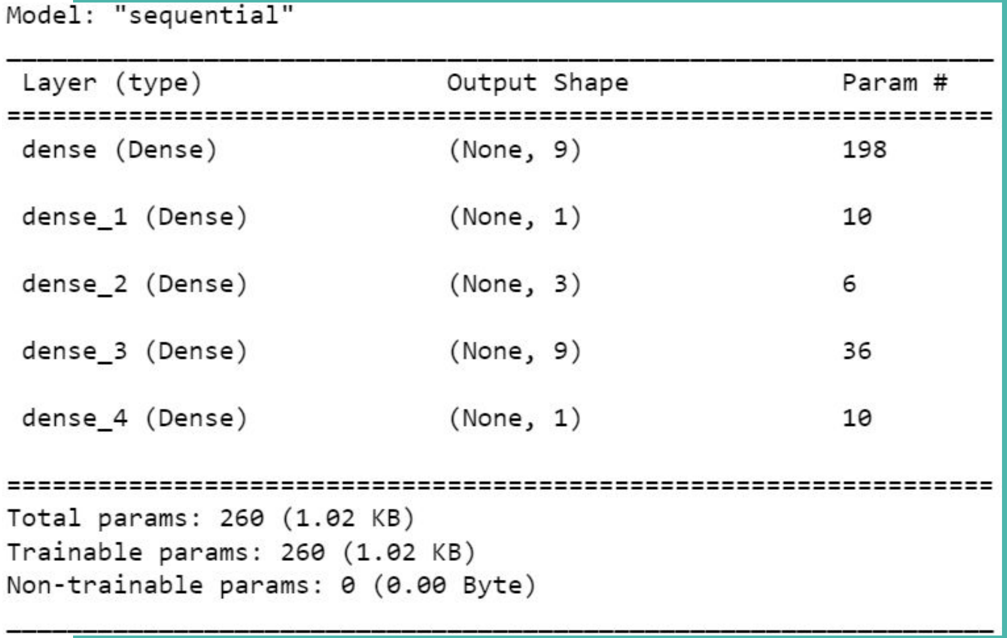
Optimizer Data, Top three models and best model values



```
553/553 - 1s - loss: 0.5041 - accuracy: 0.7530 - 1s/epoch - 2ms/step
Loss: 0.5041230916976929, Accuracy: 0.7529565095901489
```



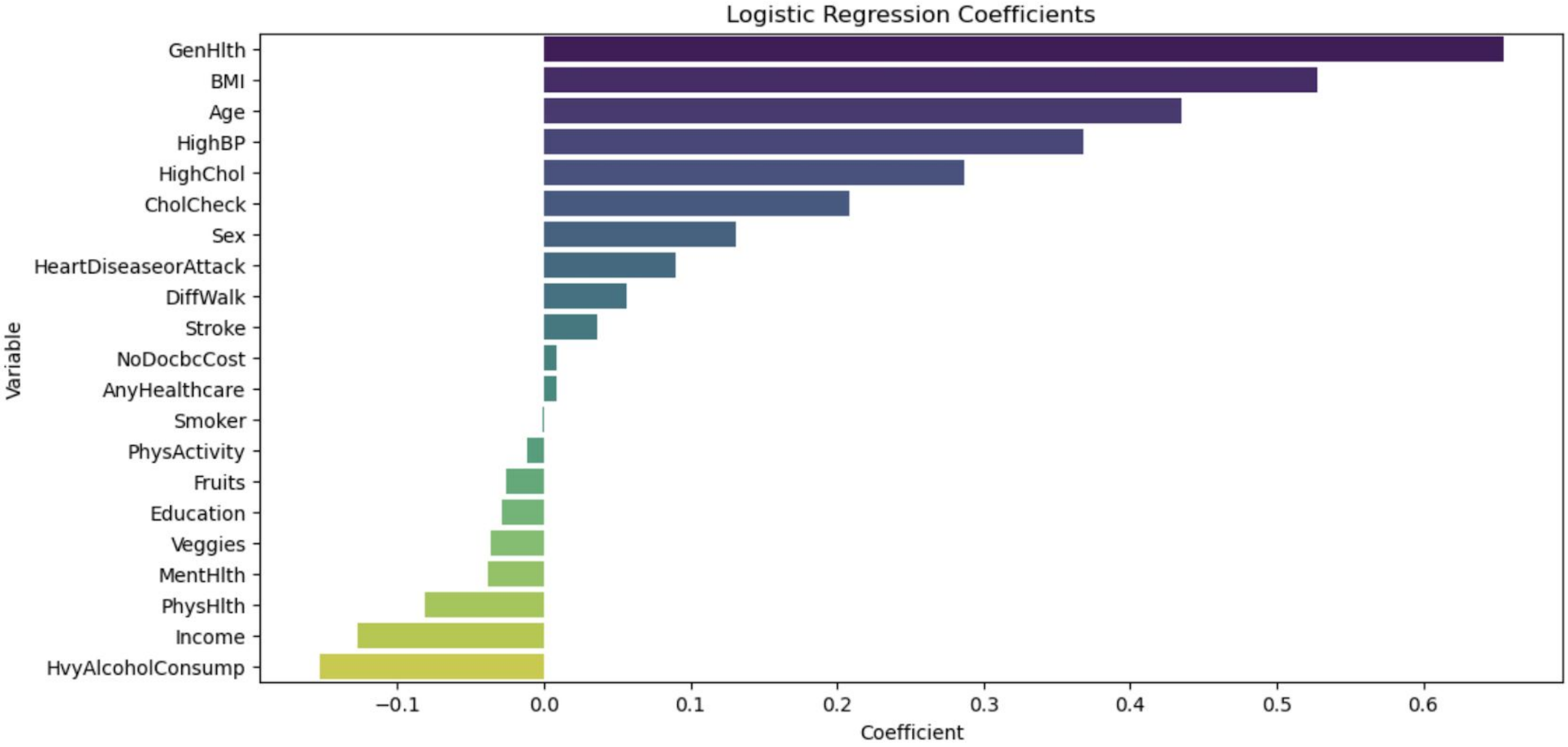
```
              precision    recall  f1-score   support

         0.0       0.76      0.73      0.74      8835
         1.0       0.74      0.77      0.75      8838

    accuracy                           0.75     17673
   macro avg       0.75      0.75      0.75     17673
weighted avg       0.75      0.75      0.75     17673
```

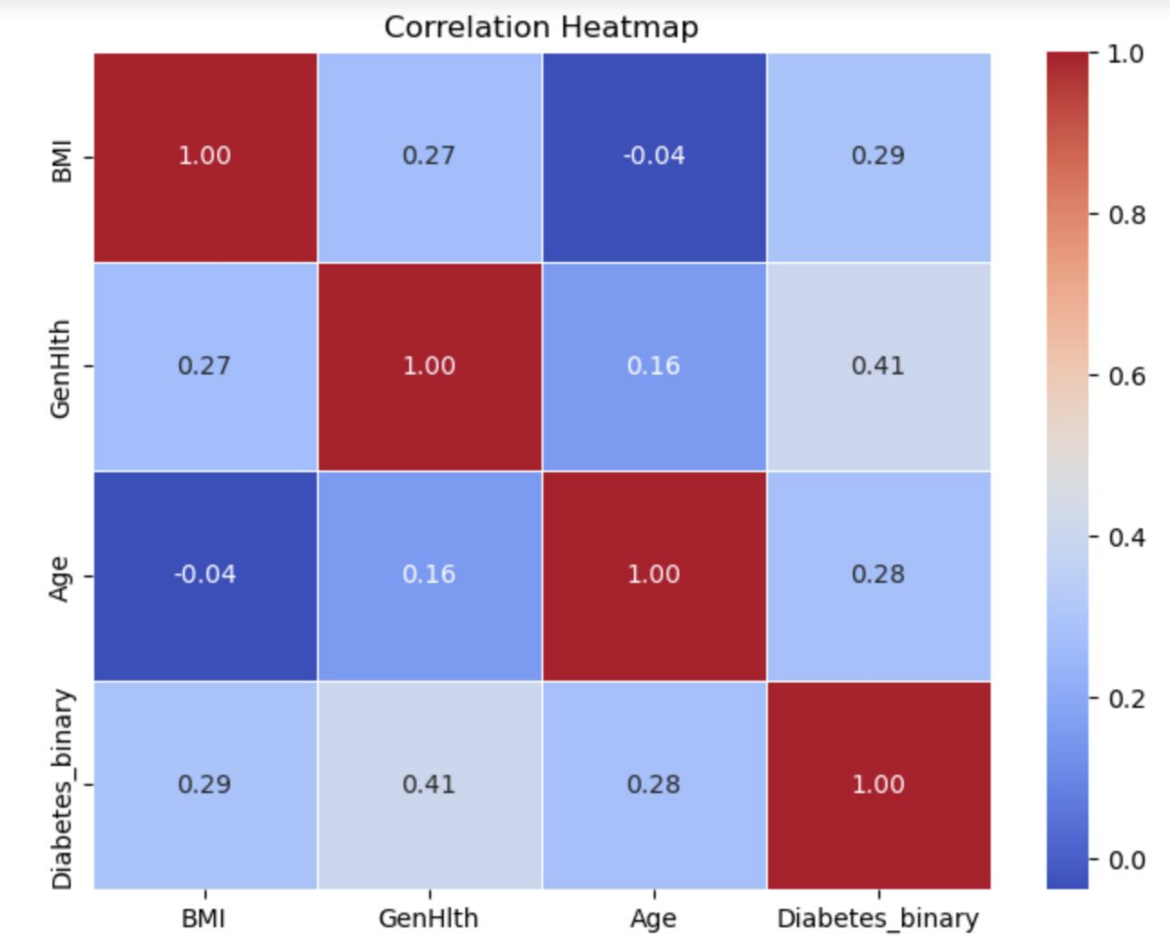


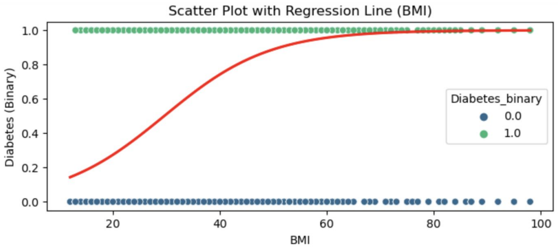
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 9)                 198

 dense_1 (Dense)             (None, 1)                 10

 dense_2 (Dense)             (None, 3)                 6

 dense_3 (Dense)             (None, 9)                 36

 dense_4 (Dense)             (None, 1)                 10

=================================================================
Total params: 260 (1.02 KB)
Trainable params: 260 (1.02 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

ort

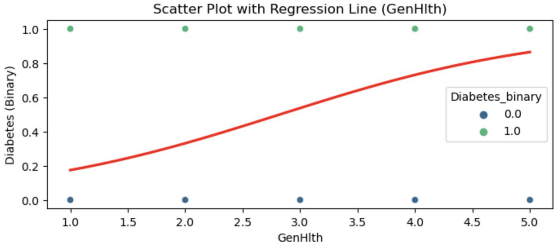
Diabetes Risk Factors



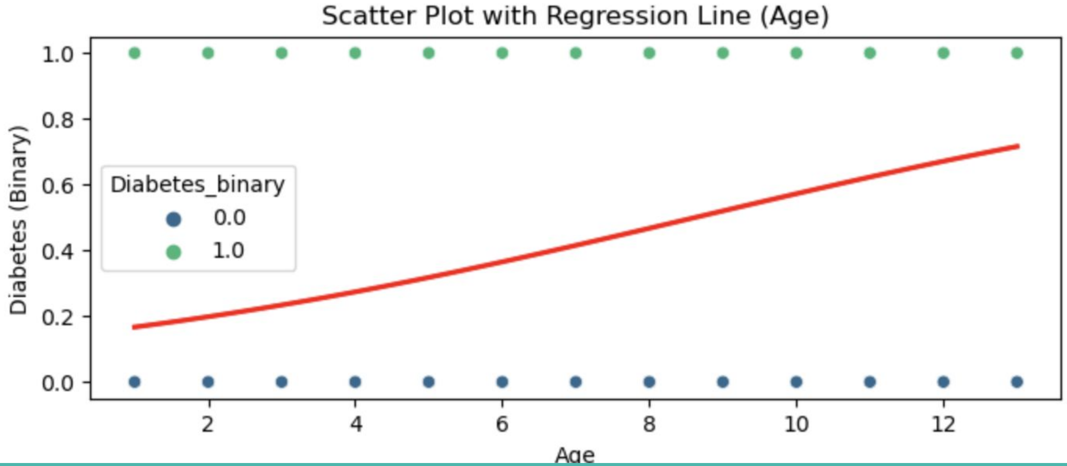

Correlation Heatmap
BMI
GenHlth
Age
Diabetes_binary
1.00
0.27
-0.04
0.29
0.27
1.00
0.16
0.41
-0.04
0.16
1.00
0.28
0.29
0.41
0.28
1.00
BMI
GenHlth
Age
Diabetes_binary
1.0
0.8
0.6
0.4
0.2
0.0



Scatter Plot with Regression Line (BMI)

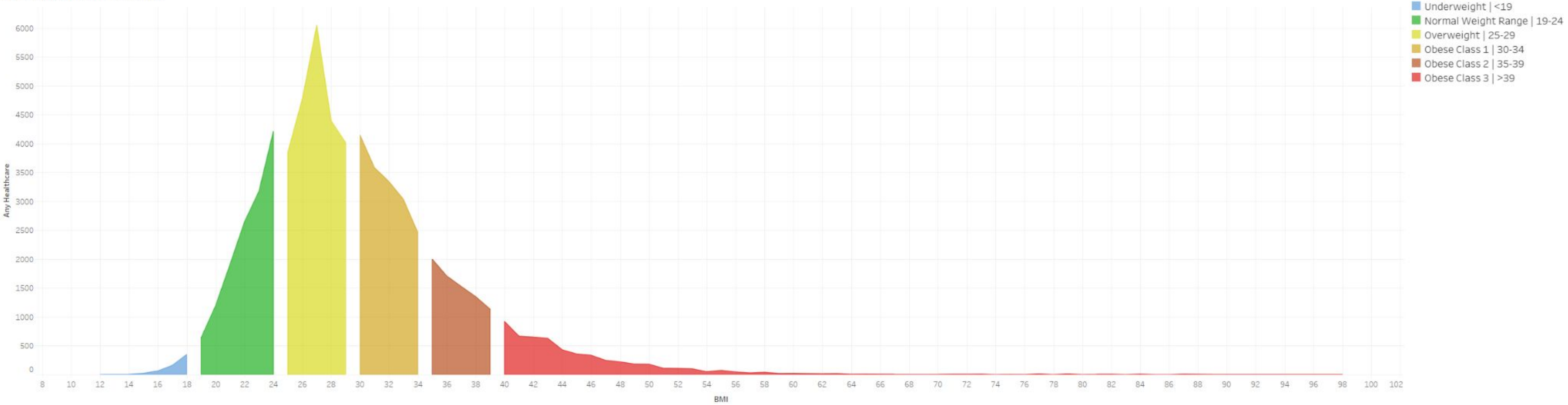


Scatter Plot with Regression Line (GenHlth)

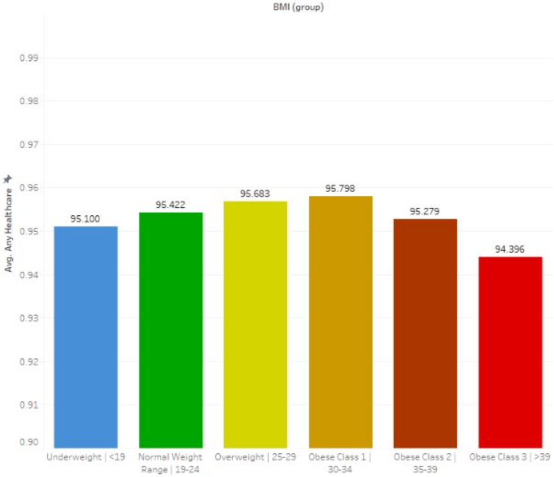


Scatter Plot with Regression Line (Age)

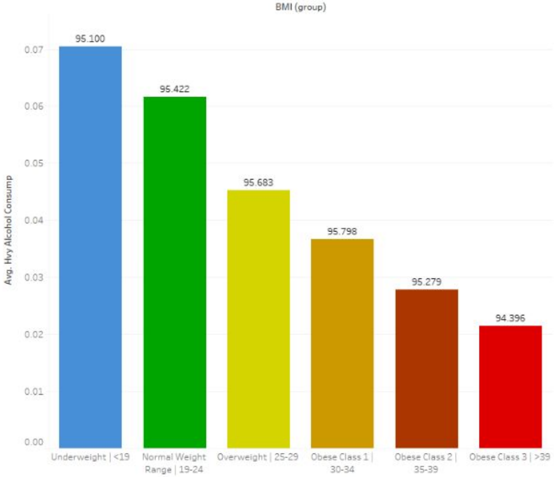
BMI vs Healthcare Coverage



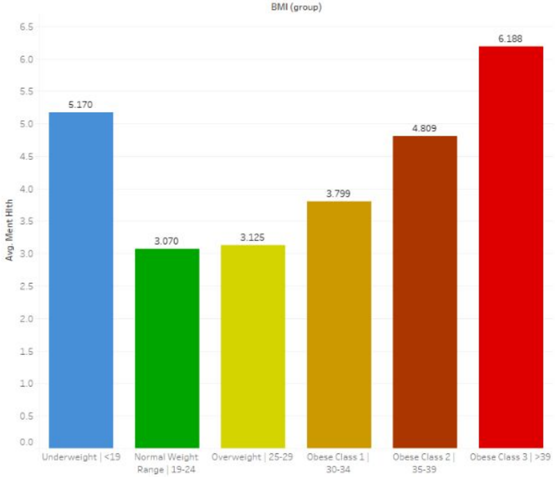
BMI Group % of Healthcare Coverage



BMI Group % of Heavy Alcohol Consumption



BMI Group Average Mental Health Score



Limitations & Challenges

- ❑ Imbalanced Dataset. Originally had accuracy of about 80%.
- ❑ The balanced dataset had an initial accuracy of 70%

Thank you