Group Project
Data Science and Machine Learning

Authors:
Jonas Brandenburg if20b243, Stephan Düx if20b245, Violeta Garcia Espin if20b091, Rawan Mousa if20b129

1. Data Acquisition

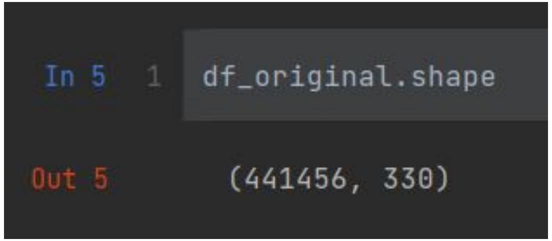
The **Behavioral Risk Factor Surveillance System** (BRFSS) is the nation’s premier system of health-related telephone surveys that collect state data about U.S. residents regarding their health-related risk behaviors, chronic health conditions, and use of preventive services.

- 400,000 adult interviews each year
- Largest continuously conducted health survey in the world
- 336 columns
- ~600mb of data per year
- Datasets from 1986 - 2020 available

https://www.cdc.gov/brfss/annual_data/annual_data.htm

https://www.kaggle.com/datasets/cdc/behavioral-risk-factor-surveillance-system

2. Data Exploration



```
In 5   df_original.shape

Out 5     (441456, 330)
```

441.456 Samples
330 Features

2. Data Exploration: Feature Selection

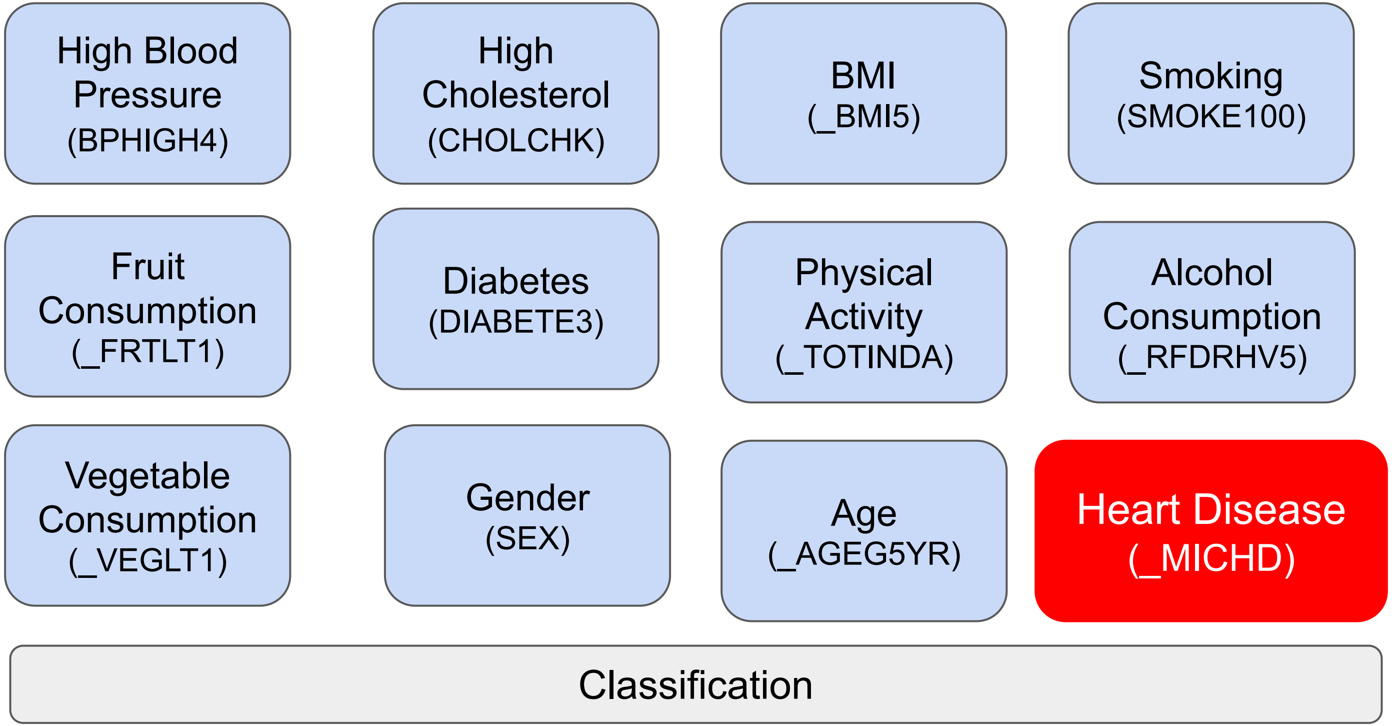
World Health Organization

NHS

CDC Centers for Disease Control and Prevention

CDC 24/7: Saving Lives, Protecting People™

2. Data Preprocessing: Feature Selection



High Blood Pressure (BPHIGH4)

High Cholesterol (CHOLCHK)

BMI (_BMI5)

Smoking (SMOKE100)

Fruit Consumption (_FRTLT1)

Diabetes (DIABETE3)

Physical Activity (_TOTINDA)

Alcohol Consumption (_RFDRHV5)

Vegetable Consumption (_VEGLT1)

Gender (SEX)

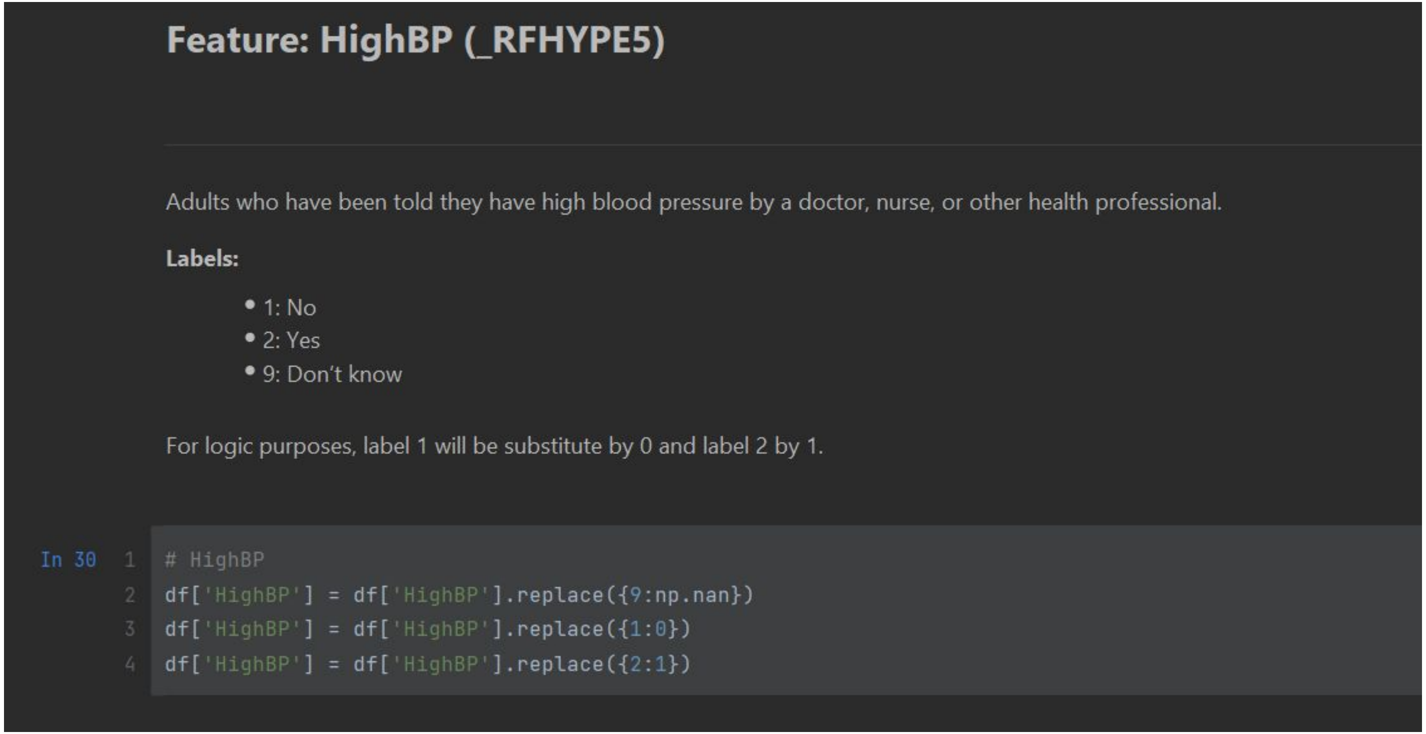
Age (_AGEG5YR)

Heart Disease (_MICHD)

Classification

https://www.cdc.gov/brfss/annual_data/2015/pdf/codebook15_llcp.pdf

3. Data Processing: Label Encoding



Feature: HighBP (_RFHYPE5)

Adults who have been told they have high blood pressure by a doctor, nurse, or other health professional.

Labels:

- 1: No
- 2: Yes
- 9: Don't know

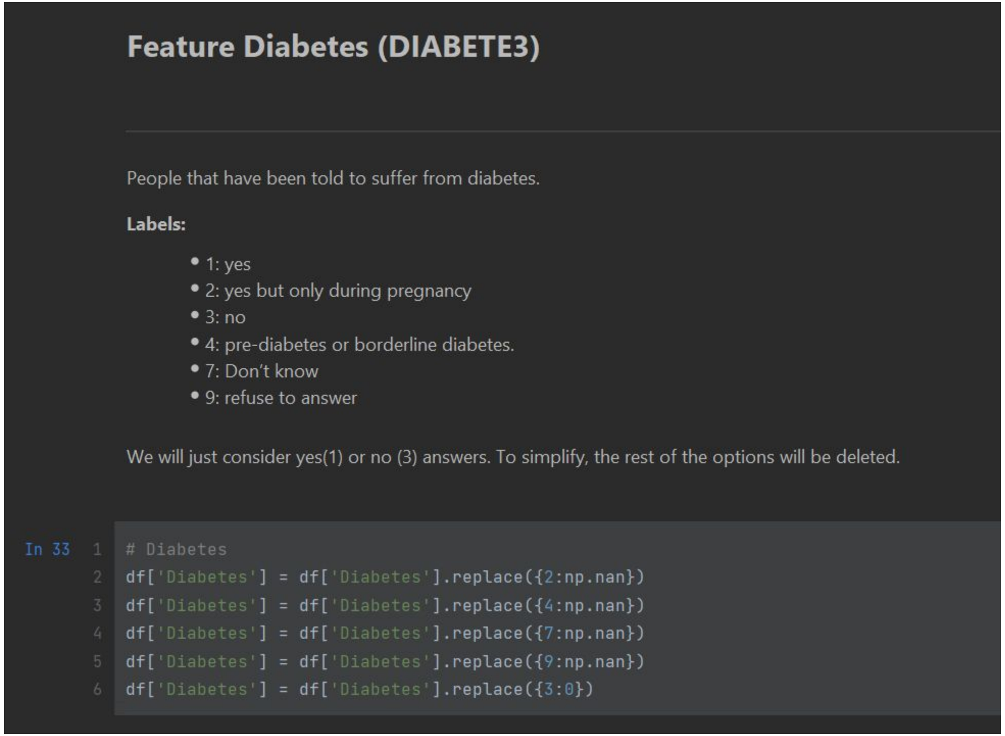
For logic purposes, label 1 will be substitute by 0 and label 2 by 1.

In 30

```python
# HighBP
df['HighBP'] = df['HighBP'].replace({9:np.nan})
df['HighBP'] = df['HighBP'].replace({1:0})
df['HighBP'] = df['HighBP'].replace({2:1})
```

3. Data Preprocessing: Label Encoding

One Hot Encoding would have been indicated.



Feature Diabetes (DIABETE3)

People that have been told to suffer from diabetes.

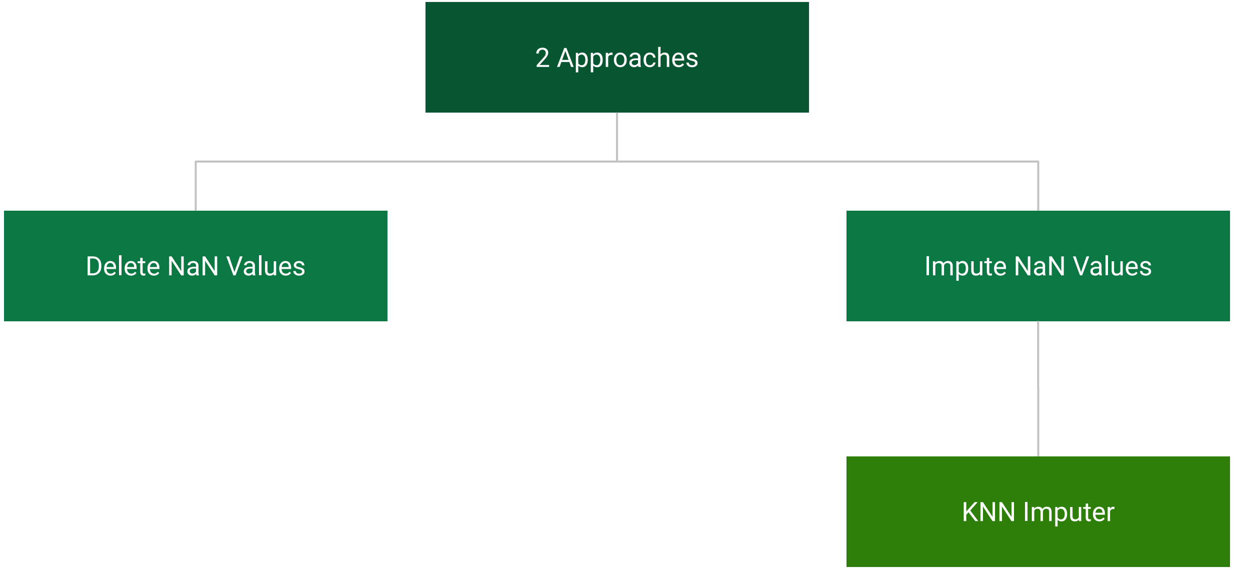
Labels:

- 1: yes
- 2: yes but only during pregnancy
- 3: no
- 4: pre-diabetes or borderline diabetes.
- 7: Don't know
- 9: refuse to answer

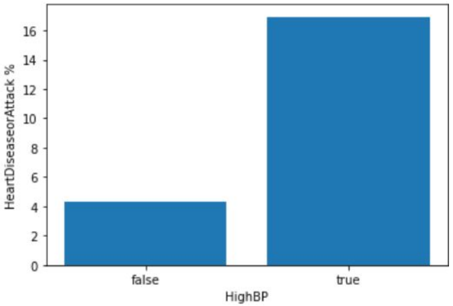
We will just consider yes(1) or no (3) answers. To simplify, the rest of the options will be deleted.

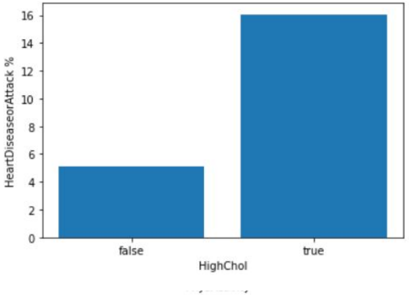
```
# Diabetes
df['Diabetes'] = df['Diabetes'].replace({2:np.nan})
df['Diabetes'] = df['Diabetes'].replace({4:np.nan})
df['Diabetes'] = df['Diabetes'].replace({7:np.nan})
df['Diabetes'] = df['Diabetes'].replace({9:np.nan})
df['Diabetes'] = df['Diabetes'].replace({3:0})
```

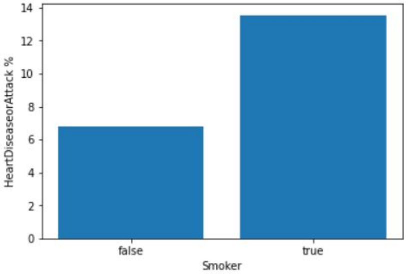
4. Data Preprocessing: Handling NaN Values

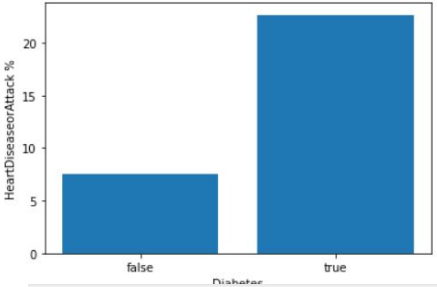


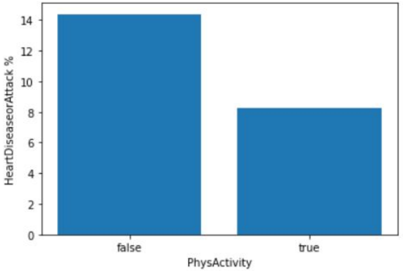
5. Data Preprocessing: Examination of individual features

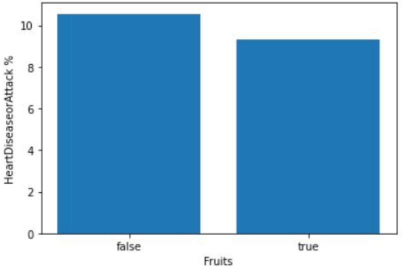


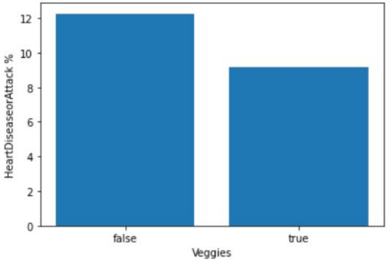


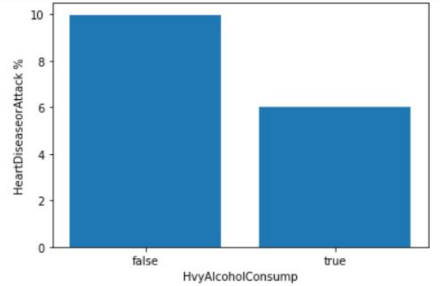


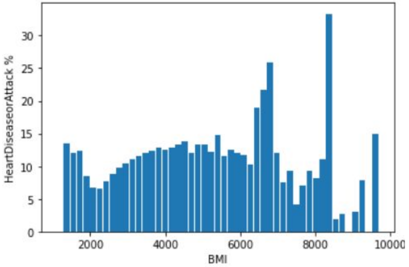


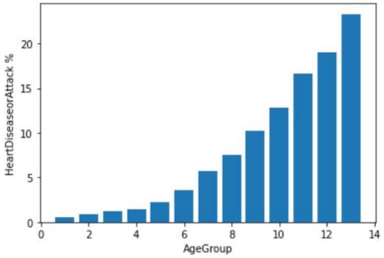




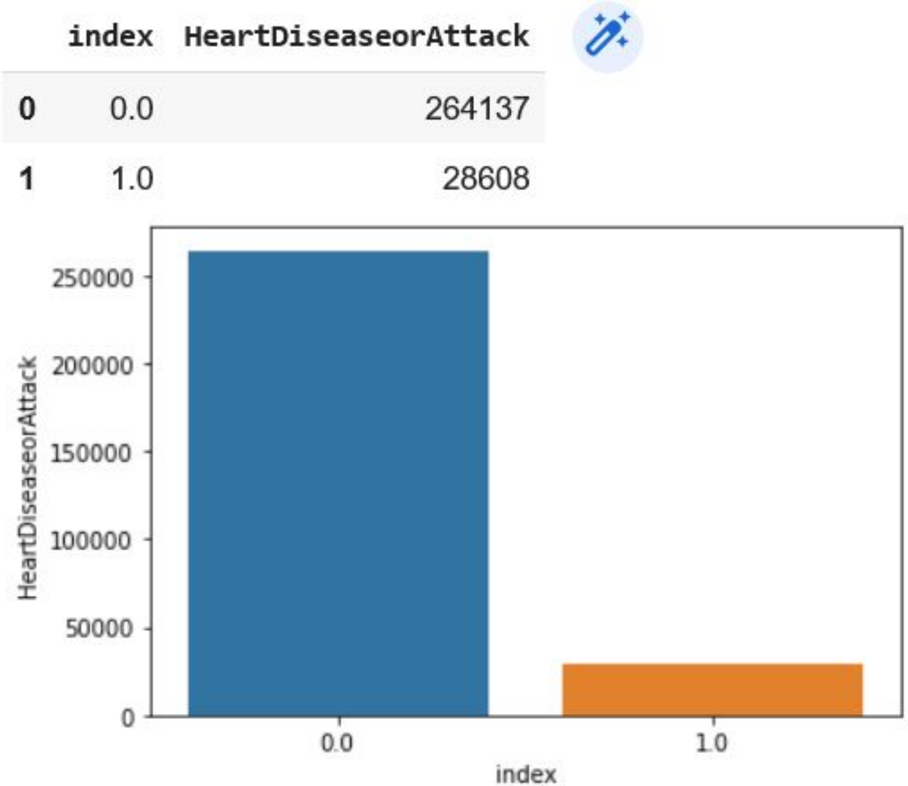








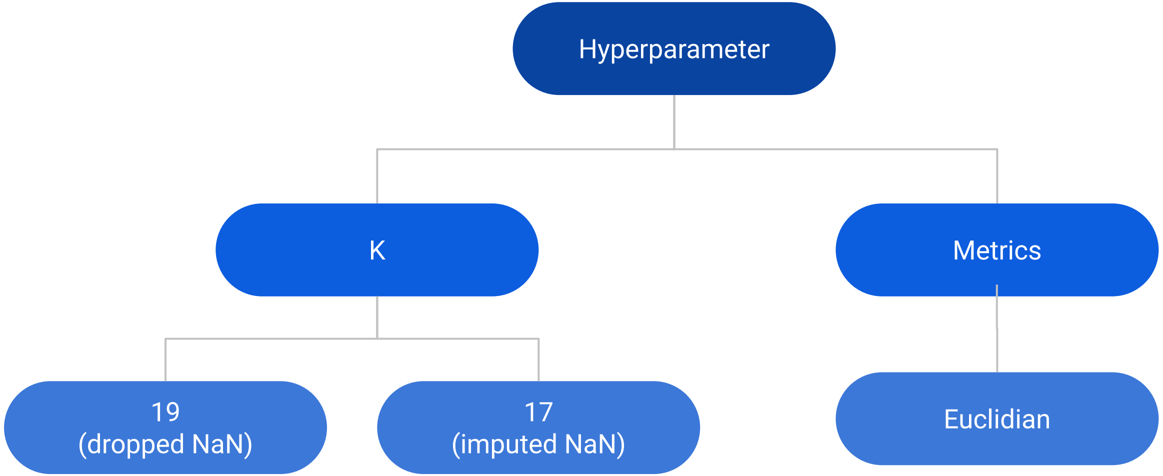
6. Data Preprocessing: Balanced data

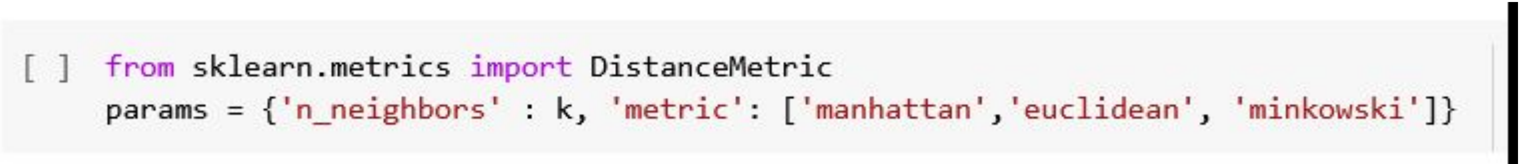


	index	HeartDiseaseorAttack
0	0.0	264137
1	1.0	28608

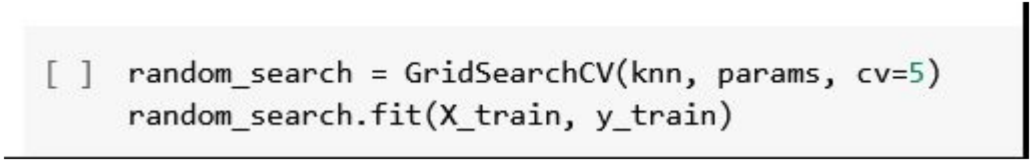
7. Model: KNN - Hyperparameter

GridSearchCV



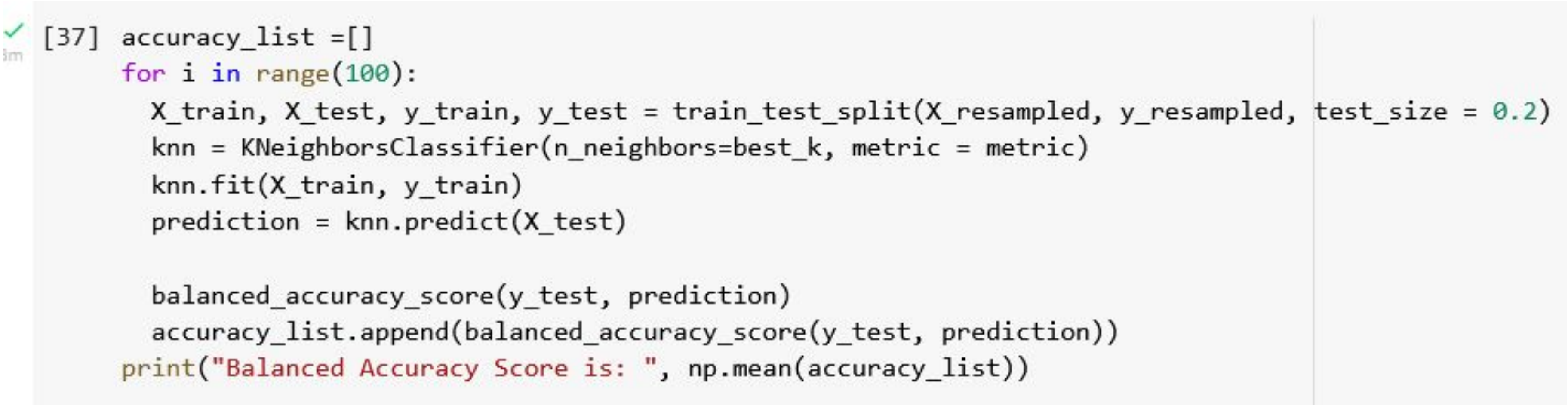


```
[ ]  from sklearn.metrics import DistanceMetric
     params = {'n_neighbors' : k, 'metric': ['manhattan','euclidean', 'minkowski']}
```



```
[ ]  random_search = GridSearchCV(knn, params, cv=5)
     random_search.fit(X_train, y_train)
```

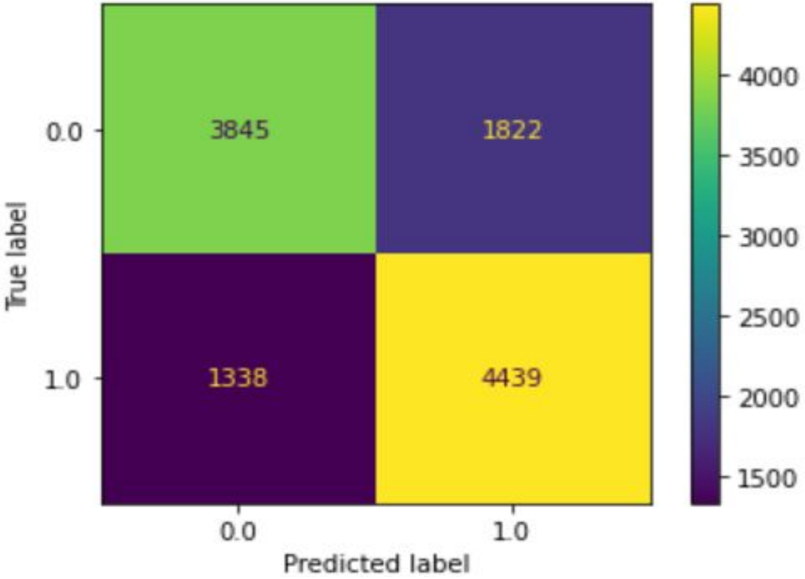
7. Model: KNN



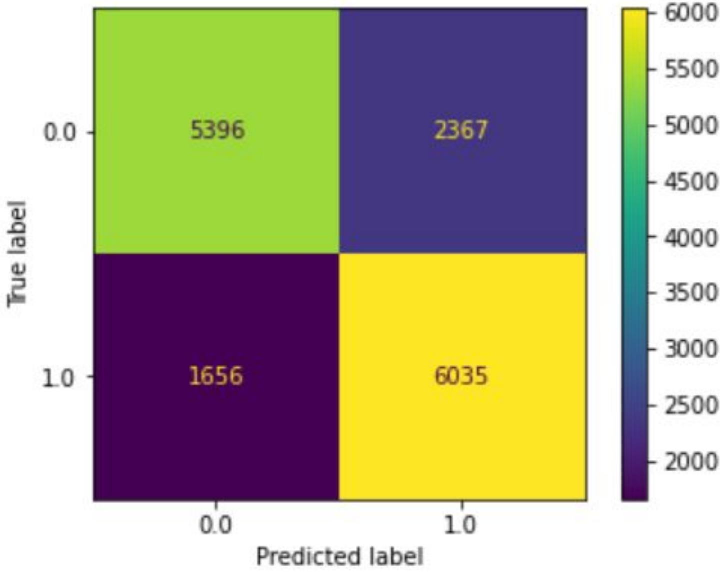
```
accuracy_list =[]
for i in range(100):
  X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size = 0.2)
  knn = KNeighborsClassifier(n_neighbors=best_k, metric = metric)
  knn.fit(X_train, y_train)
  prediction = knn.predict(X_test)

  balanced_accuracy_score(y_test, prediction)
  accuracy_list.append(balanced_accuracy_score(y_test, prediction))
print("Balanced Accuracy Score is: ", np.mean(accuracy_list))
```

7. Model KNN - Performance

Accuracy = 0.72 for dataset with dropped NaN values

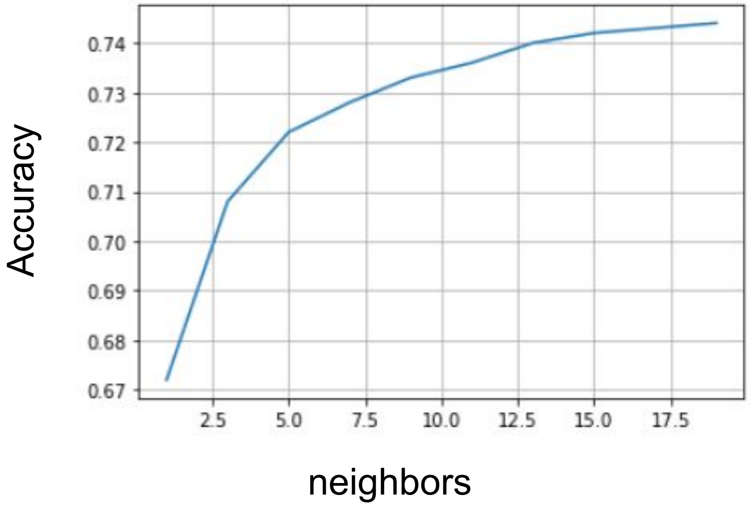


Accuracy = 0.73 for dataset with imputed NaN values

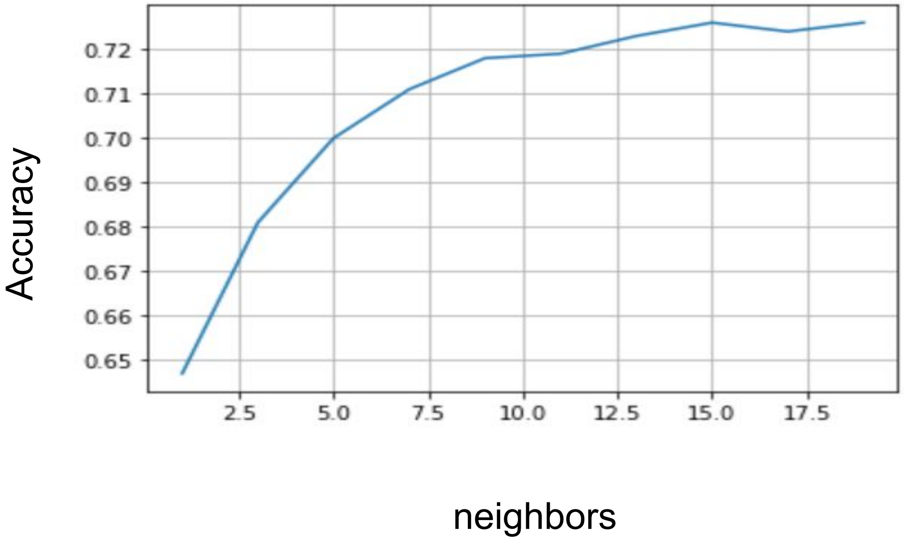


7. Model KNN - Performance (k vs. Accuracy)

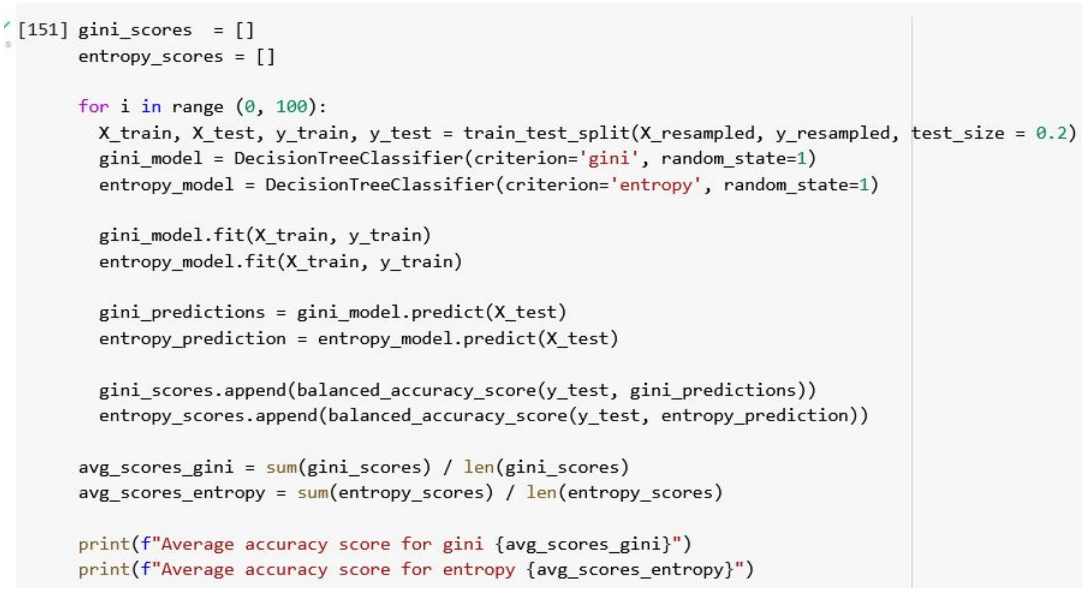
dropped NaN values



Imputed NaN values



8. Model: Decision Trees - Gini vs. Entropy

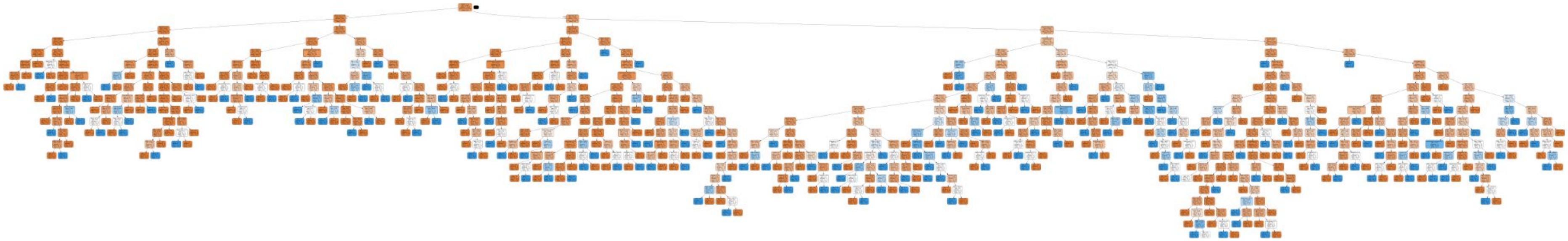


```
[151] gini_scores  = []
      entropy_scores = []

      for i in range (0, 100):
        X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size = 0.2)
        gini_model = DecisionTreeClassifier(criterion='gini', random_state=1)
        entropy_model = DecisionTreeClassifier(criterion='entropy', random_state=1)

        gini_model.fit(X_train, y_train)
        entropy_model.fit(X_train, y_train)

        gini_predictions = gini_model.predict(X_test)
        entropy_prediction = entropy_model.predict(X_test)

        gini_scores.append(balanced_accuracy_score(y_test, gini_predictions))
        entropy_scores.append(balanced_accuracy_score(y_test, entropy_prediction))

      avg_scores_gini = sum(gini_scores) / len(gini_scores)
      avg_scores_entropy = sum(entropy_scores) / len(entropy_scores)

      print(f"Average accuracy score for gini {avg_scores_gini}")
      print(f"Average accuracy score for entropy {avg_scores_entropy}")
```

Gini	Entropy	
0.642	0.647	Dropped NaN
0.673	0.674	Imputed NaN

Entropy might perform slightly better but computationally more demanding.

8. Model: Decision Trees



I think we need pruning ….

Source: https://www.123rf.com/photo_138703028_vector-illustration-of-a-boy-pruning-bush-with-shears-isolated-on-white-background-cute-kid-doing-ga.html?vti=oenrrzu1j54jpx7ld5-1-6

8. Model: Decision Trees - Pruning

Pre-Pruning:

max_depth

max_leaf_nodes

min_samples_split

min_impurity_decrease

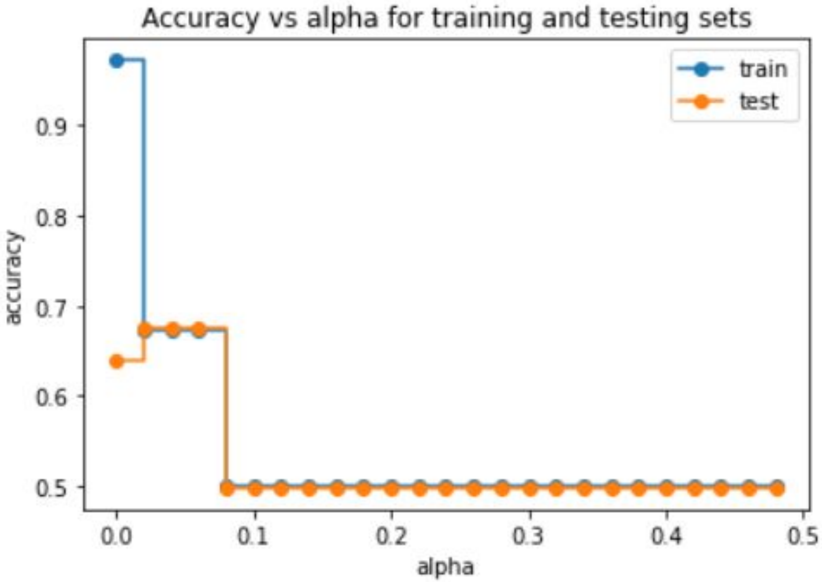
min_samples_leaf

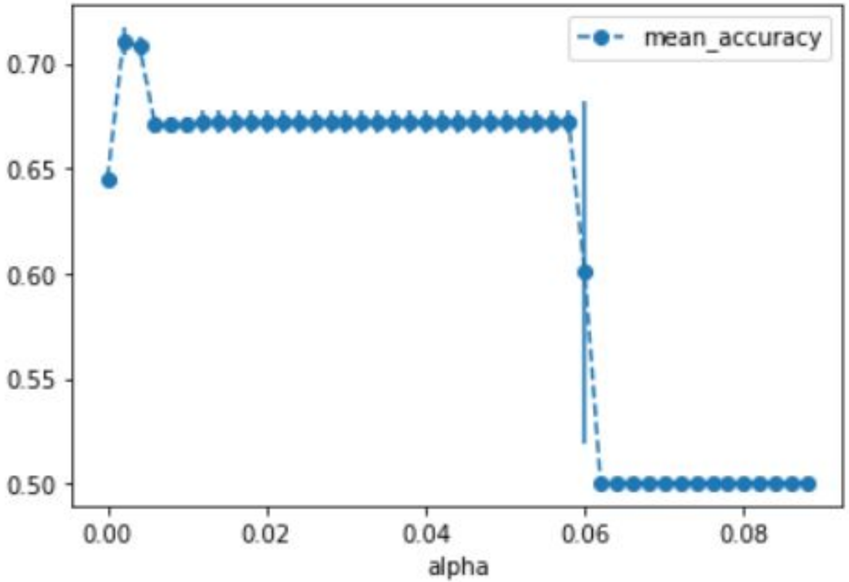
GridSarchCV()

RandomizedSearchCV()

8. Model: Decision Trees - Pruning

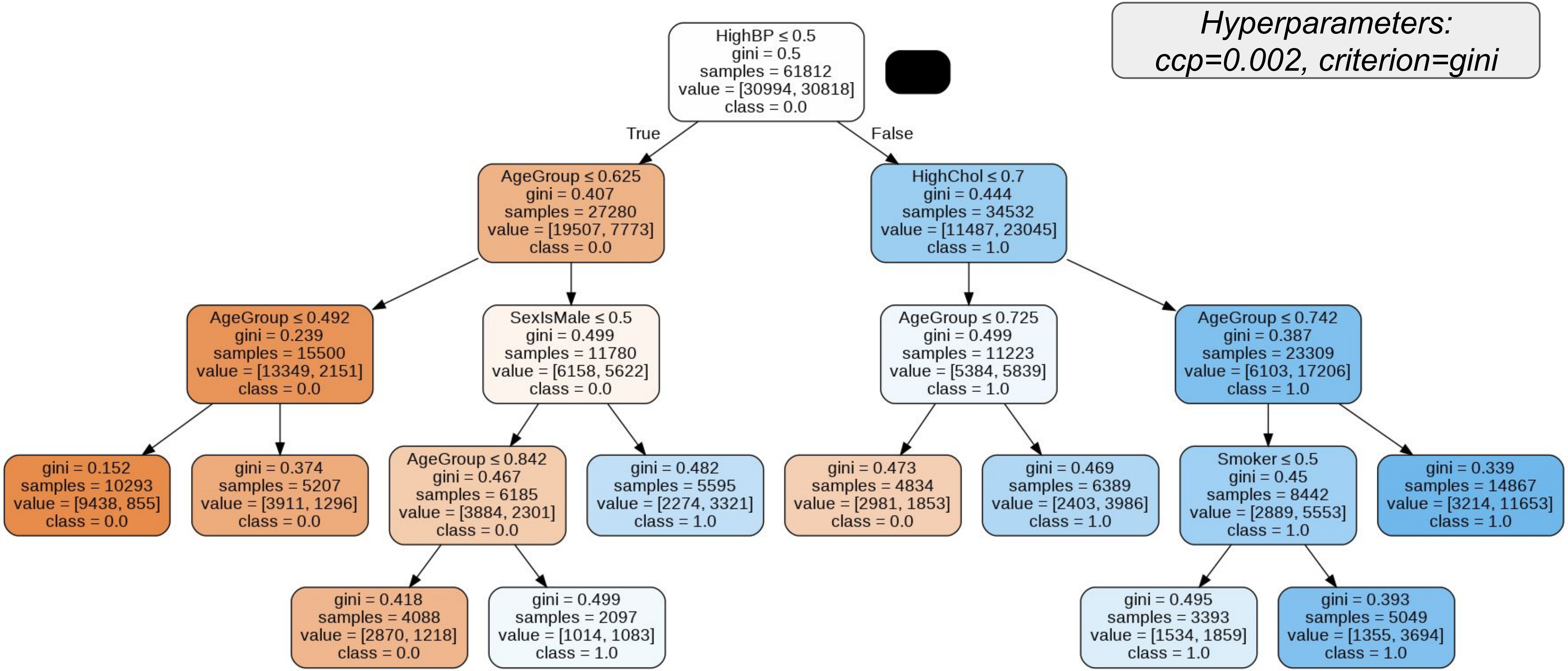
Post-Pruning: ccp_alpha





8. Model: Decision Trees - Optimized Tree

Hyperparameters:
ccp=0.002, criterion=gini



[redacted]

HighBP ≤ 0.5
gini = 0.5
samples = 61812
value = [30994, 30818]
class = 0.0

True

False

AgeGroup ≤ 0.625
gini = 0.407
samples = 27280
value = [19507, 7773]
class = 0.0

HighChol ≤ 0.7
gini = 0.444
samples = 34532
value = [11487, 23045]
class = 1.0

AgeGroup ≤ 0.492
gini = 0.239
samples = 15500
value = [13349, 2151]
class = 0.0

SexIsMale ≤ 0.5
gini = 0.499
samples = 11780
value = [6158, 5622]
class = 0.0

AgeGroup ≤ 0.725
gini = 0.499
samples = 11223
value = [5384, 5839]
class = 1.0

AgeGroup ≤ 0.742
gini = 0.387
samples = 23309
value = [6103, 17206]
class = 1.0

gini = 0.152
samples = 10293
value = [9438, 855]
class = 0.0

gini = 0.374
samples = 5207
value = [3911, 1296]
class = 0.0

AgeGroup ≤ 0.842
gini = 0.467
samples = 6185
value = [3884, 2301]
class = 0.0

gini = 0.482
samples = 5595
value = [2274, 3321]
class = 1.0

gini = 0.473
samples = 4834
value = [2981, 1853]
class = 0.0

gini = 0.469
samples = 6389
value = [2403, 3986]
class = 1.0

Smoker ≤ 0.5
gini = 0.45
samples = 8442
value = [2889, 5553]
class = 1.0

gini = 0.339
samples = 14867
value = [3214, 11653]
class = 1.0

gini = 0.418
samples = 4088
value = [2870, 1218]
class = 0.0

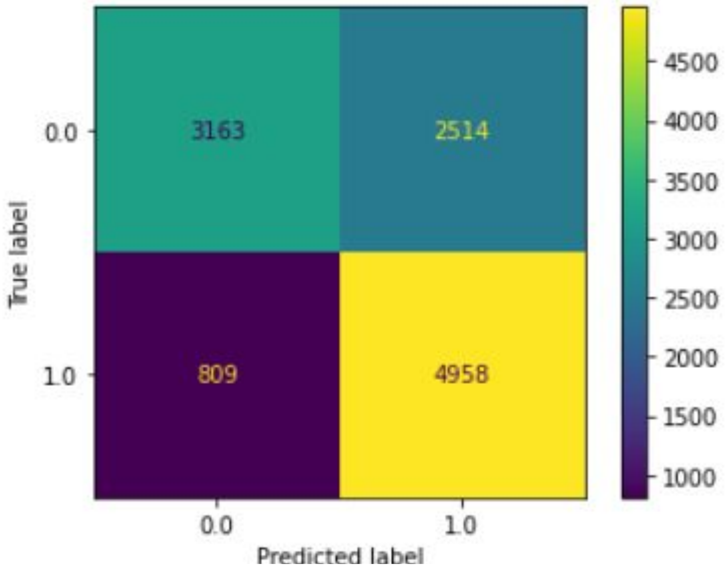
gini = 0.499
samples = 2097
value = [1014, 1083]
class = 1.0

gini = 0.495
samples = 3393
value = [1534, 1859]
class = 1.0

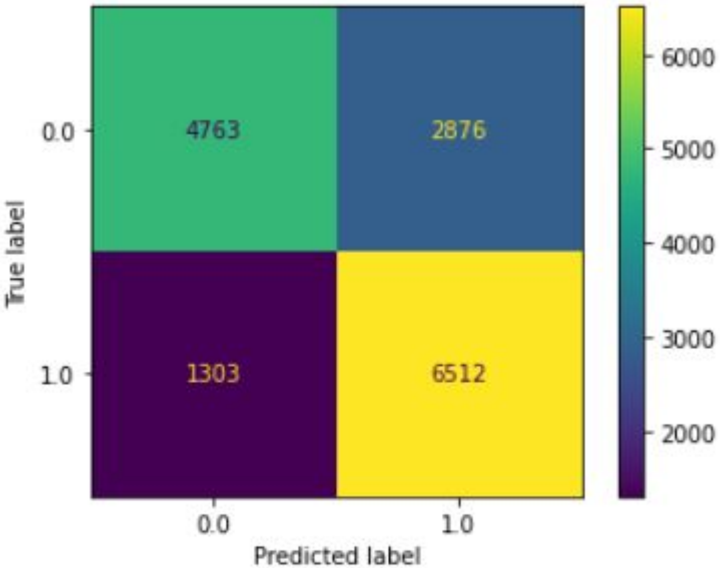
gini = 0.393
samples = 5049
value = [1355, 3694]
class = 1.0

8. Model: Decision Trees - Performance

Accuracy = 0.708 for dataset with dropped NaN values



Accuracy = 0.728 for dataset with imputed NaN values

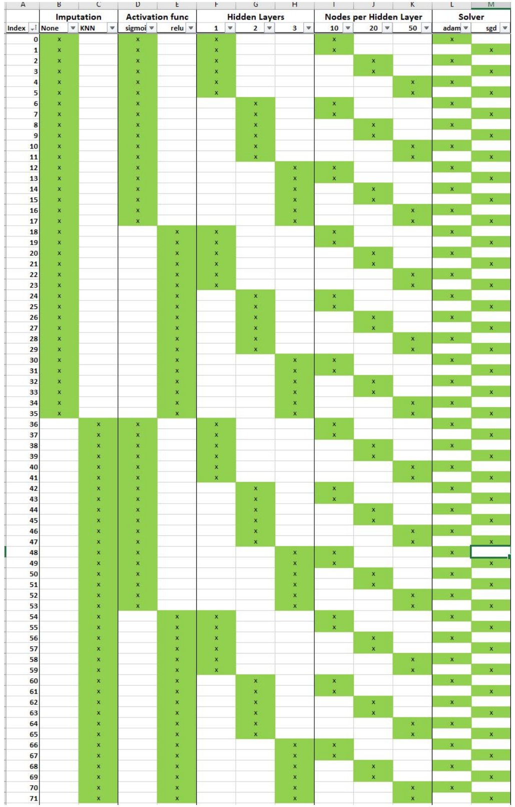


9. Model: MLP - Overview

Hyper Parameters

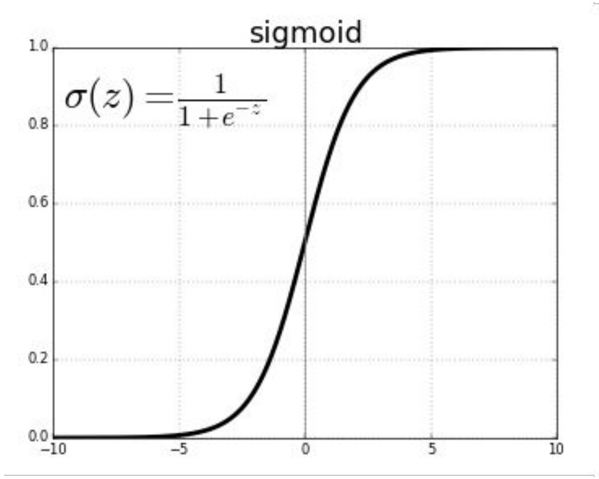
Type	**Values**		
KNN Imputation	NONE	KNN	
Activation function	sigmoid	relu	
ANN Arch: hidden layers	1	2	3
ANN Arch: hidden layers nodes	10	20	50
solver	adam	sgd	

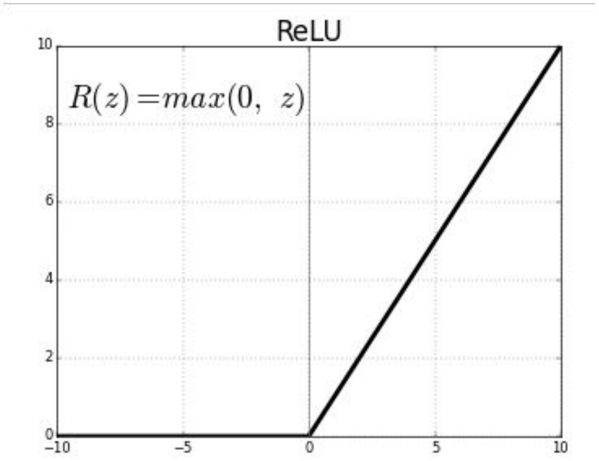
2 x 2 x 3 x 3 x 2 = 72 combinations

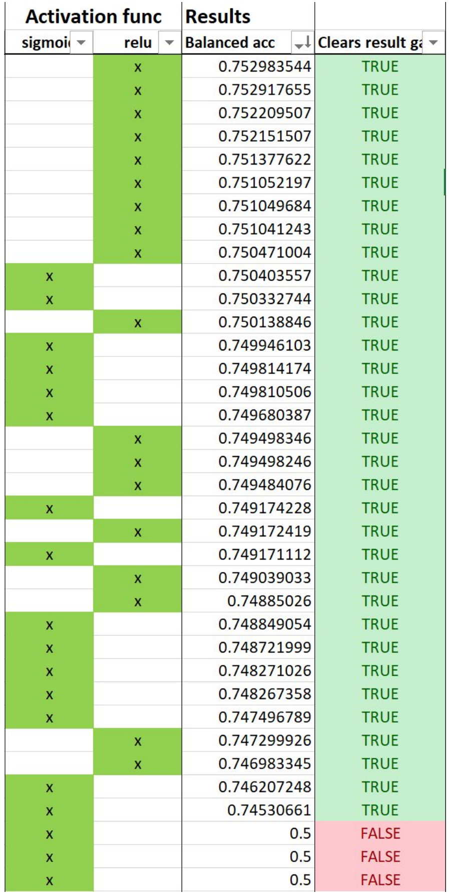


A	B	C	D	E	F	G	H	I	J	K	L	M
	Imputation		Activation func		Hidden Layers			Nodes per Hidden Layer			Solver	
Index	None	KNN	sigmoi	relu	1	2	3	10	20	50	adam	sgd
0	x		x		x			x			x	
1	x		x		x			x				x
2	x		x		x				x		x	
3	x		x		x				x			x
4	x		x		x					x	x	
5	x		x		x					x		x
6	x		x			x		x			x	
7	x		x			x		x				x
8	x		x			x			x		x	
9	x		x			x			x			x
10	x		x			x				x	x	
11	x		x			x				x		x
12	x		x				x	x			x	
13	x		x				x	x				x
14	x		x				x		x		x	
15	x		x				x		x			x
16	x		x				x			x	x	
17	x		x				x			x		x
18	x			x	x			x			x	
19	x			x	x			x				x
20	x			x	x				x		x	
21	x			x	x				x			x
22	x			x	x					x	x	
23	x			x	x					x		x
24	x			x		x		x			x	
25	x			x		x		x				x
26	x			x		x			x		x	
27	x			x		x			x			x
28	x			x		x				x	x	
29	x			x		x				x		x
30	x			x			x	x			x	
31	x			x			x	x				x
32	x			x			x		x		x	
33	x			x			x		x			x
34	x			x			x			x	x	
35	x			x			x			x		x
36		x	x		x			x			x	
37		x	x		x			x				x
38		x	x		x				x		x	
39		x	x		x				x			x
40		x	x		x					x	x	
41		x	x		x					x		x
42		x	x			x		x			x	
43		x	x			x		x				x
44		x	x			x			x		x	
45		x	x			x			x			x
46		x	x			x				x	x	
47		x	x			x				x		x
48		x	x				x	x			x	
49		x	x				x	x				x
50		x	x				x		x		x	
51		x	x				x		x			x
52		x	x				x			x	x	
53		x	x				x			x		x
54		x		x	x			x			x	
55		x		x	x			x				x
56		x		x	x				x		x	
57		x		x	x				x			x
58		x		x	x					x	x	
59		x		x	x					x		x
60		x		x		x		x			x	
61		x		x		x		x				x
62		x		x		x			x		x	
63		x		x		x			x			x
64		x		x		x				x	x	
65		x		x		x				x		x
66		x		x			x	x			x	
67		x		x			x	x				x
68		x		x			x		x		x	
69		x		x			x		x			x
70		x		x			x			x	x	
71		x		x			x			x		x

9. Model: MLP - Activation Function

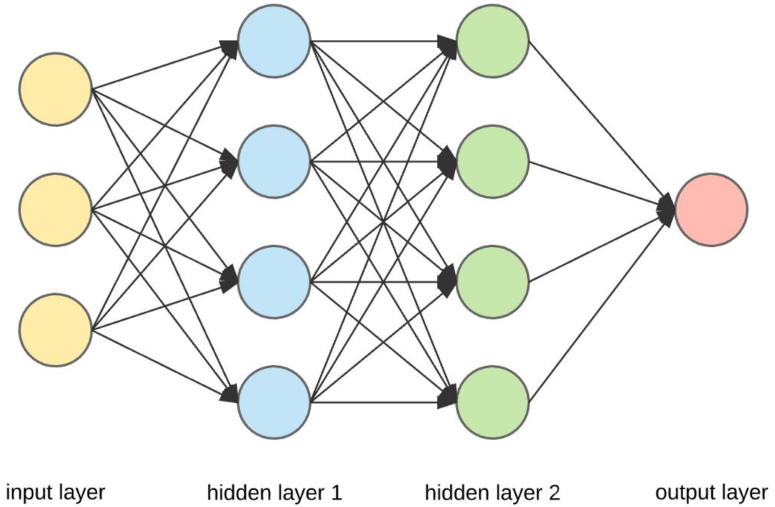




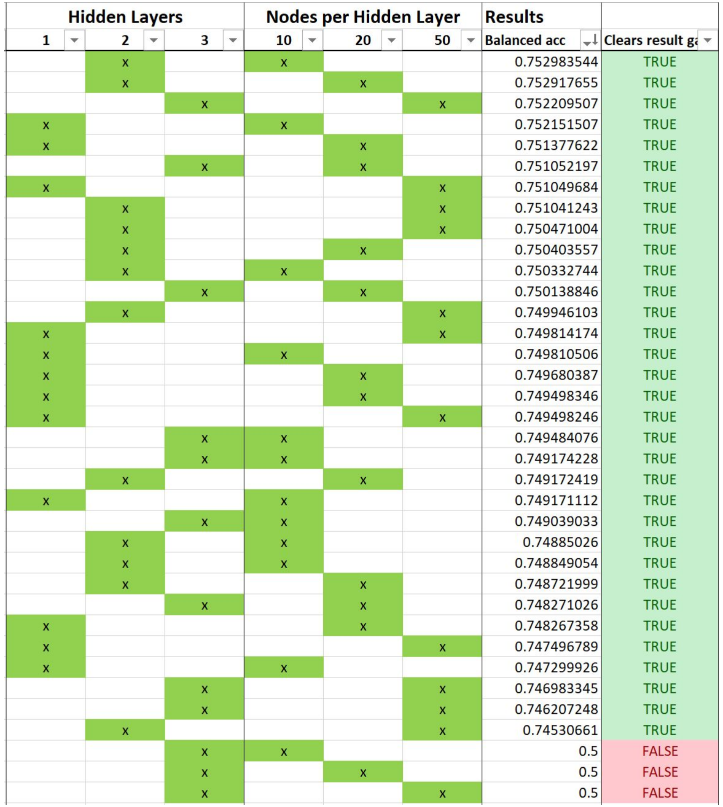


Activation func		Results	
sigmoi	relu	Balanced acc	Clears result ga
	x	0.752983544	TRUE
	x	0.752917655	TRUE
	x	0.752209507	TRUE
	x	0.752151507	TRUE
	x	0.751377622	TRUE
	x	0.751052197	TRUE
	x	0.751049684	TRUE
	x	0.751041243	TRUE
	x	0.750471004	TRUE
x		0.750403557	TRUE
x		0.750332744	TRUE
	x	0.750138846	TRUE
x		0.749946103	TRUE
x		0.749814174	TRUE
x		0.749810506	TRUE
x		0.749680387	TRUE
	x	0.749498346	TRUE
	x	0.749498246	TRUE
	x	0.749484076	TRUE
x		0.749174228	TRUE
	x	0.749172419	TRUE
x		0.749171112	TRUE
	x	0.749039033	TRUE
	x	0.74885026	TRUE
x		0.748849054	TRUE
x		0.748721999	TRUE
x		0.748271026	TRUE
x		0.748267358	TRUE
x		0.747496789	TRUE
	x	0.747299926	TRUE
	x	0.746983345	TRUE
x		0.746207248	TRUE
x		0.74530661	TRUE
x		0.5	FALSE
x		0.5	FALSE
x		0.5	FALSE

9. Model: MLP - Architecture

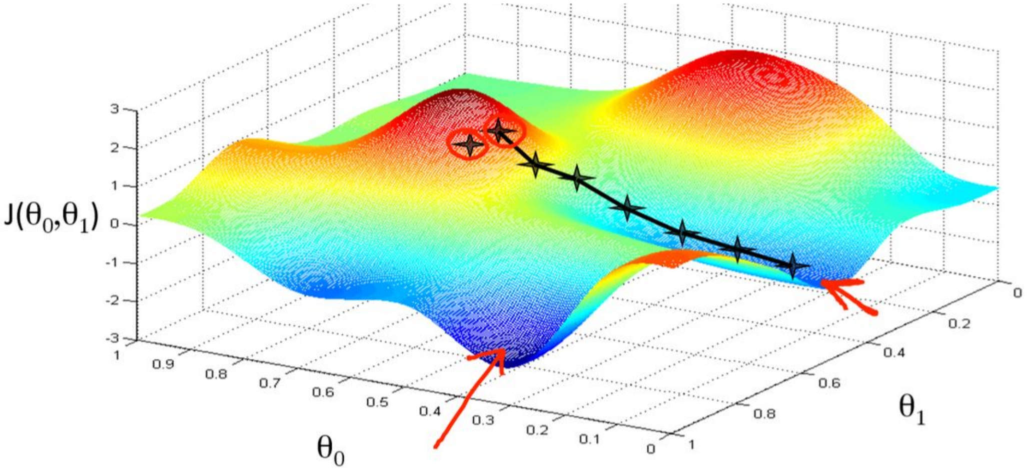


How does that make sense ???

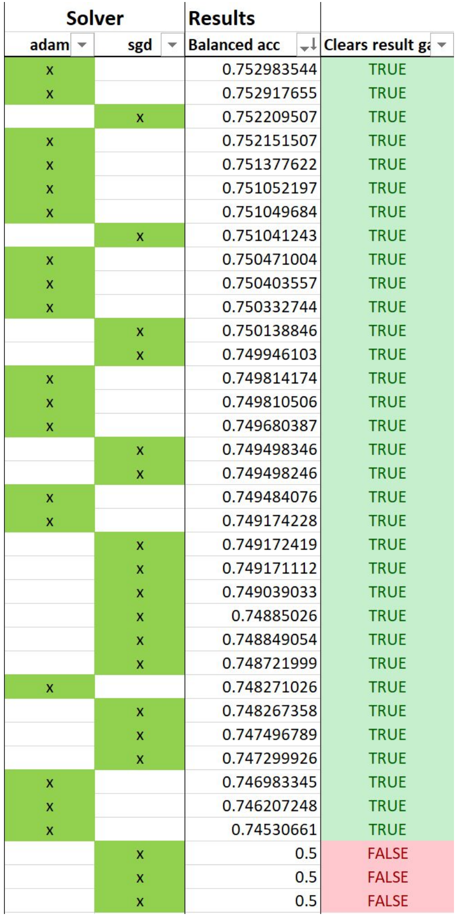


Hidden Layers			Nodes per Hidden Layer			Results	
1	2	3	10	20	50	Balanced acc	Clears result ga
	x		x			0.752983544	TRUE
	x			x		0.752917655	TRUE
		x			x	0.752209507	TRUE
x			x			0.752151507	TRUE
x				x		0.751377622	TRUE
		x		x		0.751052197	TRUE
x					x	0.751049684	TRUE
	x				x	0.751041243	TRUE
	x				x	0.750471004	TRUE
	x			x		0.750403557	TRUE
	x		x			0.750332744	TRUE
		x		x		0.750138846	TRUE
	x				x	0.749946103	TRUE
x					x	0.749814174	TRUE
x			x			0.749810506	TRUE
x				x		0.749680387	TRUE
x				x		0.749498346	TRUE
x					x	0.749498246	TRUE
		x	x			0.749484076	TRUE
		x	x			0.749174228	TRUE
	x			x		0.749172419	TRUE
x			x			0.749171112	TRUE
		x	x			0.749039033	TRUE
	x		x			0.74885026	TRUE
	x		x			0.748849054	TRUE
	x			x		0.748721999	TRUE
		x		x		0.748271026	TRUE
x				x		0.748267358	TRUE
x					x	0.747496789	TRUE
x			x			0.747299926	TRUE
		x			x	0.746983345	TRUE
		x			x	0.746207248	TRUE
	x				x	0.74530661	TRUE
		x	x			0.5	FALSE
		x		x		0.5	FALSE
		x			x	0.5	FALSE

9. Model: MLP - Optimizers



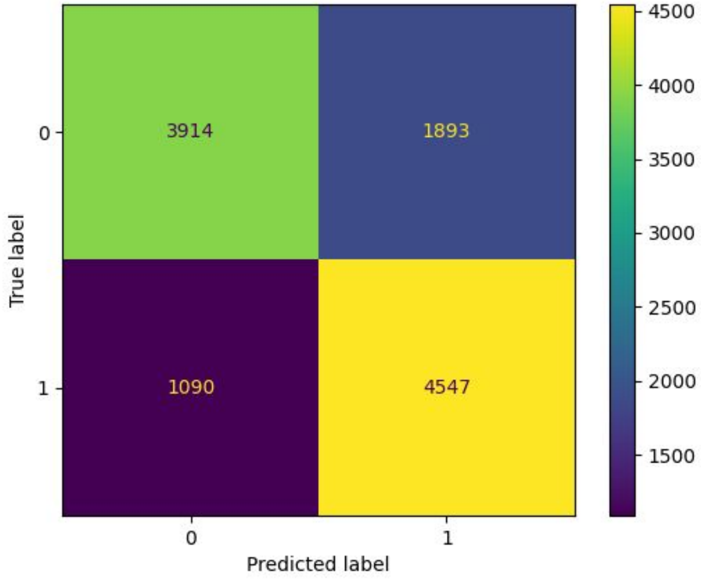
Adaptive Moment Estimation
vs
Stochastic Gradient Descent



Solver		Results	
adam	sgd	Balanced acc	Clears result ga
x		0.752983544	TRUE
x		0.752917655	TRUE
	x	0.752209507	TRUE
x		0.752151507	TRUE
x		0.751377622	TRUE
x		0.751052197	TRUE
x		0.751049684	TRUE
	x	0.751041243	TRUE
x		0.750471004	TRUE
x		0.750403557	TRUE
x		0.750332744	TRUE
	x	0.750138846	TRUE
	x	0.749946103	TRUE
x		0.749814174	TRUE
x		0.749810506	TRUE
x		0.749680387	TRUE
	x	0.749498346	TRUE
	x	0.749498246	TRUE
x		0.749484076	TRUE
x		0.749174228	TRUE
	x	0.749172419	TRUE
	x	0.749171112	TRUE
	x	0.749039033	TRUE
	x	0.74885026	TRUE
	x	0.748849054	TRUE
	x	0.748721999	TRUE
x		0.748271026	TRUE
	x	0.748267358	TRUE
	x	0.747496789	TRUE
	x	0.747299926	TRUE
x		0.746983345	TRUE
x		0.746207248	TRUE
x		0.74530661	TRUE
	x	0.5	FALSE
	x	0.5	FALSE
	x	0.5	FALSE

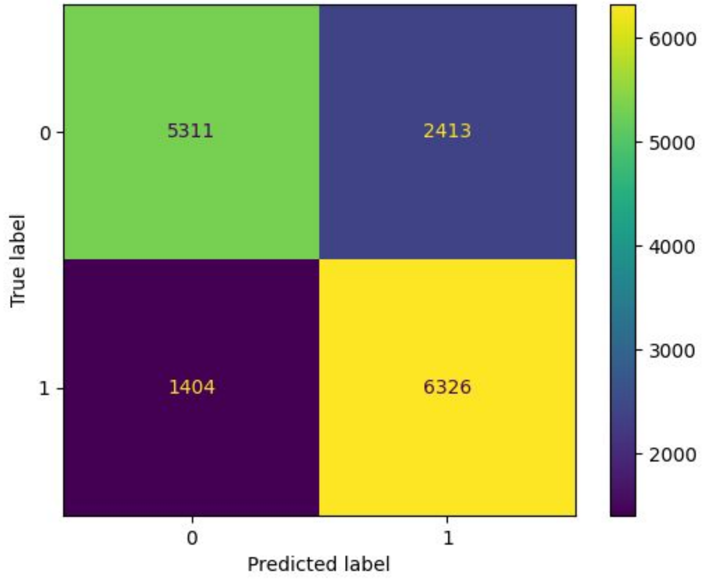
9. Model: MLP - Performance

Accuracy = 0.742 for dataset with dropped NaN values



Architecture: 20, 20, 20

Accuracy = 0.753 for dataset with imputed NaN values



Architecture: 10, 10

10. Algorithm performance comparison

Accuracy:

- Highest balanced accuracy of implemented algorithms: 0.753 MLP(imputed)
- Worst balanced accuracy of implemented algorithms: 0.708 Decision Tree(dropped)
- Undersampling the data lead to a moderate number of false negatives

Speed:

- The optimised decision tree is very fast with predictions once trained and optimized
- KNN has the disadvantage of a huge the dataset requiring intensive computational work but is still quite fast
- MLP is extremely dependent on the hyperparameter setup

Dropped NaN vs. Imputed NaN in best case:

- Imputed NaN values consistently improved accuracy over dropped Nan

Accuracy for KNN:
Dropped = 0.72
Imputed = 0.74
Difference = 0.010

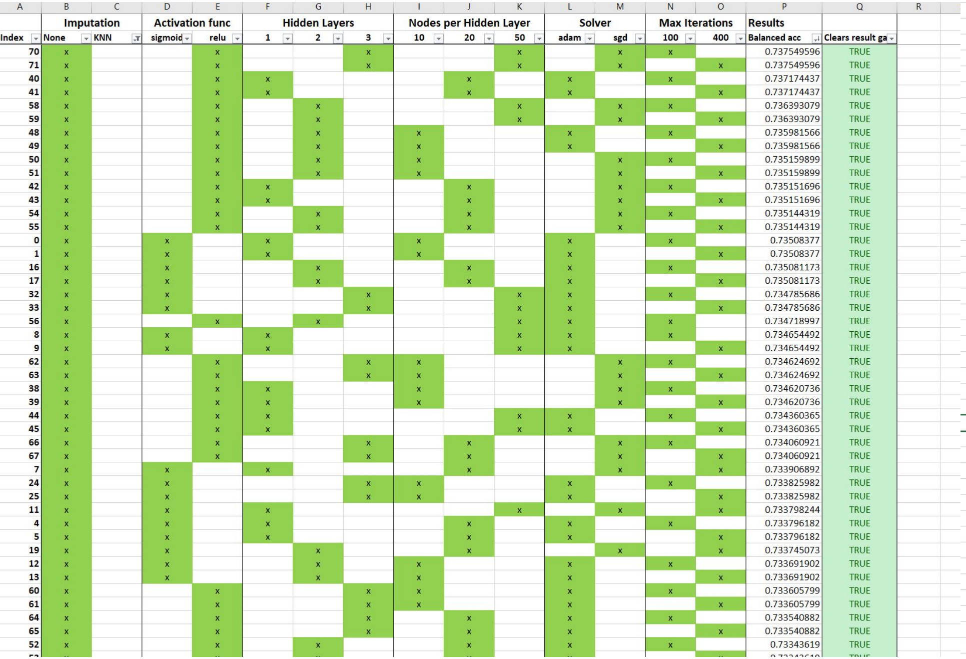
Accuracy for Decision Tree:
Dropped = 0.708
Imputed = 0.728
Difference = 0.020

Accuracy for MLP:
Dropped = 0.742
Imputed = 0.753
Difference = 0.011

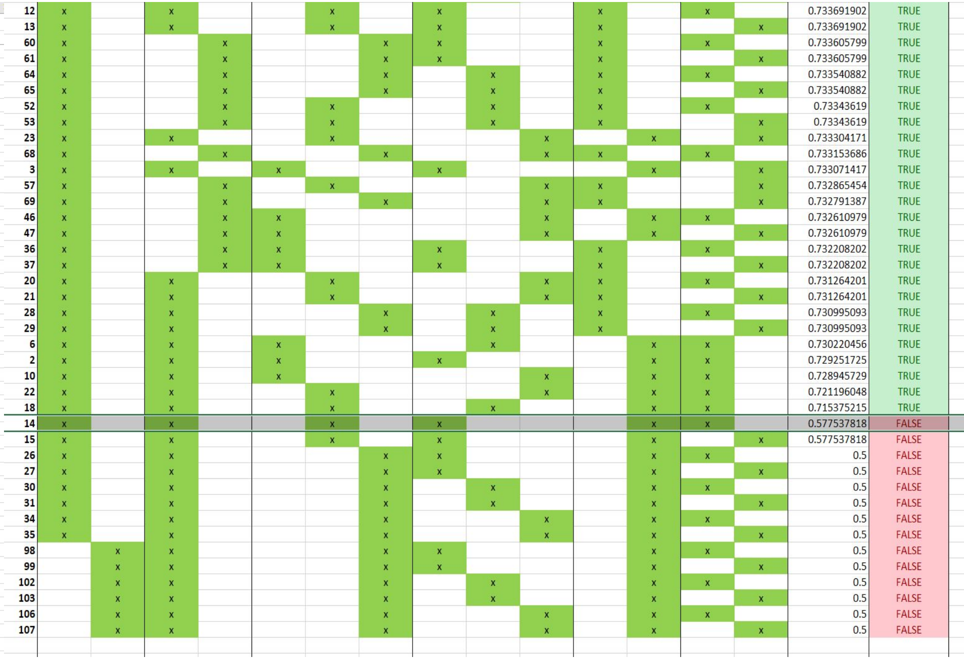
11. Conclusions

- Real-world implications of data preprocessing: e.g. simplifying diabetes label could introduce hidden dangers when assessing heart disease risk
- Preprocessing of data has major influence on final result
- While using imputed values risk introducing bias, a big and representative dataset can mitigate them
- Great number of Hyperparameters can make it very hard to understand their interactions

Complex hyperparameters example



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
	Imputation		Activation func		Hidden Layers			Nodes per Hidden Layer			Solver		Max Iterations		Results		
Index	None	KNN	sigmoid	relu	1	2	3	10	20	50	adam	sgd	100	400	Balanced acc	Clears result ga	
70	x			x			x			x		x	x		0.737549596	TRUE	
71	x			x			x			x		x		x	0.737549596	TRUE	
40	x			x	x				x		x		x		0.737174437	TRUE	
41	x			x	x				x		x			x	0.737174437	TRUE	
58	x			x		x				x		x	x		0.736393079	TRUE	
59	x			x		x				x		x		x	0.736393079	TRUE	
48	x			x		x		x			x		x		0.735981566	TRUE	
49	x			x		x		x			x			x	0.735981566	TRUE	
50	x			x		x		x				x	x		0.735159899	TRUE	
51	x			x		x		x				x		x	0.735159899	TRUE	
42	x			x	x				x			x	x		0.735151696	TRUE	
43	x			x	x				x			x		x	0.735151696	TRUE	
54	x			x		x			x			x	x		0.735144319	TRUE	
55	x			x		x			x			x		x	0.735144319	TRUE	
0	x		x		x			x			x		x		0.73508377	TRUE	
1	x		x		x			x			x			x	0.73508377	TRUE	
16	x		x			x			x		x		x		0.735081173	TRUE	
17	x		x			x			x		x			x	0.735081173	TRUE	
32	x		x				x			x	x		x		0.734785686	TRUE	
33	x		x				x			x	x			x	0.734785686	TRUE	
56	x			x		x				x	x		x		0.734718997	TRUE	
8	x		x		x					x	x		x		0.734654492	TRUE	
9	x		x		x					x	x			x	0.734654492	TRUE	
62	x			x			x	x				x	x		0.734624692	TRUE	
63	x			x			x	x				x		x	0.734624692	TRUE	
38	x			x	x			x				x	x		0.734620736	TRUE	
39	x			x	x			x				x		x	0.734620736	TRUE	
44	x			x	x					x	x		x		0.734360365	TRUE	
45	x			x	x					x	x			x	0.734360365	TRUE	
66	x			x			x		x			x	x		0.734060921	TRUE	
67	x			x			x		x			x		x	0.734060921	TRUE	
7	x		x		x				x			x		x	0.733906892	TRUE	
24	x		x				x	x			x		x		0.733825982	TRUE	
25	x		x				x	x			x			x	0.733825982	TRUE	
11	x		x		x					x		x		x	0.733798244	TRUE	
4	x		x		x				x		x		x		0.733796182	TRUE	
5	x		x		x				x		x			x	0.733796182	TRUE	
19	x		x			x			x			x		x	0.733745073	TRUE	
12	x		x			x		x			x		x		0.733691902	TRUE	
13	x		x			x		x			x			x	0.733691902	TRUE	
60	x			x			x	x			x		x		0.733605799	TRUE	
61	x			x			x	x			x			x	0.733605799	TRUE	
64	x			x			x		x		x		x		0.733540882	TRUE	
65	x			x			x		x		x			x	0.733540882	TRUE	
52	x			x		x			x		x		x		0.73343619	TRUE	



12	x		x			x		x			x		x		0.733691902	TRUE
13	x		x			x		x			x			x	0.733691902	TRUE
60	x			x			x	x			x		x		0.733605799	TRUE
61	x			x			x	x			x			x	0.733605799	TRUE
64	x			x			x		x		x		x		0.733540882	TRUE
65	x			x			x		x		x			x	0.733540882	TRUE
52	x			x		x			x		x		x		0.73343619	TRUE
53	x			x		x			x		x			x	0.73343619	TRUE
23	x		x			x				x		x		x	0.733304171	TRUE
68	x			x			x			x	x		x		0.733153686	TRUE
3	x		x		x			x				x		x	0.733071417	TRUE
57	x			x		x				x	x			x	0.732865454	TRUE
69	x			x			x			x	x			x	0.732791387	TRUE
46	x			x	x					x		x	x		0.732610979	TRUE
47	x			x	x					x		x		x	0.732610979	TRUE
36	x			x	x			x			x		x		0.732208202	TRUE
37	x			x	x			x			x			x	0.732208202	TRUE
20	x		x			x				x	x		x		0.731264201	TRUE
21	x		x			x				x	x			x	0.731264201	TRUE
28	x		x				x		x		x		x		0.730995093	TRUE
29	x		x				x		x		x			x	0.730995093	TRUE
6	x		x		x				x			x	x		0.730220456	TRUE
2	x		x		x			x				x	x		0.729251725	TRUE
10	x		x		x					x		x	x		0.728945729	TRUE
22	x		x			x				x		x	x		0.721196048	TRUE
18	x		x			x			x			x	x		0.715375215	TRUE
14	x		x			x		x				x	x		0.577537818	FALSE
15	x		x			x		x				x		x	0.577537818	FALSE
26	x		x				x	x				x	x		0.5	FALSE
27	x		x				x	x				x		x	0.5	FALSE
30	x		x				x		x			x	x		0.5	FALSE
31	x		x				x		x			x		x	0.5	FALSE
34	x		x				x			x		x	x		0.5	FALSE
35	x		x				x			x		x		x	0.5	FALSE
98		x	x				x	x				x	x		0.5	FALSE
99		x	x				x	x				x		x	0.5	FALSE
102		x	x				x		x			x	x		0.5	FALSE
103		x	x				x		x			x		x	0.5	FALSE
106		x	x				x			x		x	x		0.5	FALSE
107		x	x				x			x		x		x	0.5	FALSE

Questions
Feedback