WYKRYWANIE CHORÓB SERCA Z WYKORZYSTANIEM ALGORYTMÓW UCZENIA MASZYNOWEGO

WOJCIECH GRZYWOCZ (GR. 3/5), PIOTR OLASIK (GR. 3/5), WITOLD PACHOLIK (GR. 3/5)

Projekt przedstawia wykorzystanie klasycznych algorytmów uczenia maszynowego w problemie klasyfikacji medycznej. Klasyfikacja ta polega na wykrywaniu czy dana osoba cierpi na chorobę serca czy też nie, na podstawie danych medycznych tej osoby. W projekcie wykorzystane zostały klasyczne algorytmy i techniki uczenia maszynowego takie jak na przykład Algorytm K-Najbliższych Sąsiadów (KNN), algorytm Naiwnego klasyfikatora Bayesa czy również algorytm PCA redukcji wymiarowości. Celem projektu było przetestowanie kilku klasyfikatorów na różnych wartościach parametrów celem znalezienia modelu, który najlepiej spełnia zadanie klasyfikacji pacjentów na podstawie ich danych medycznych.

KROK 1 – ANALIZA DANYCH

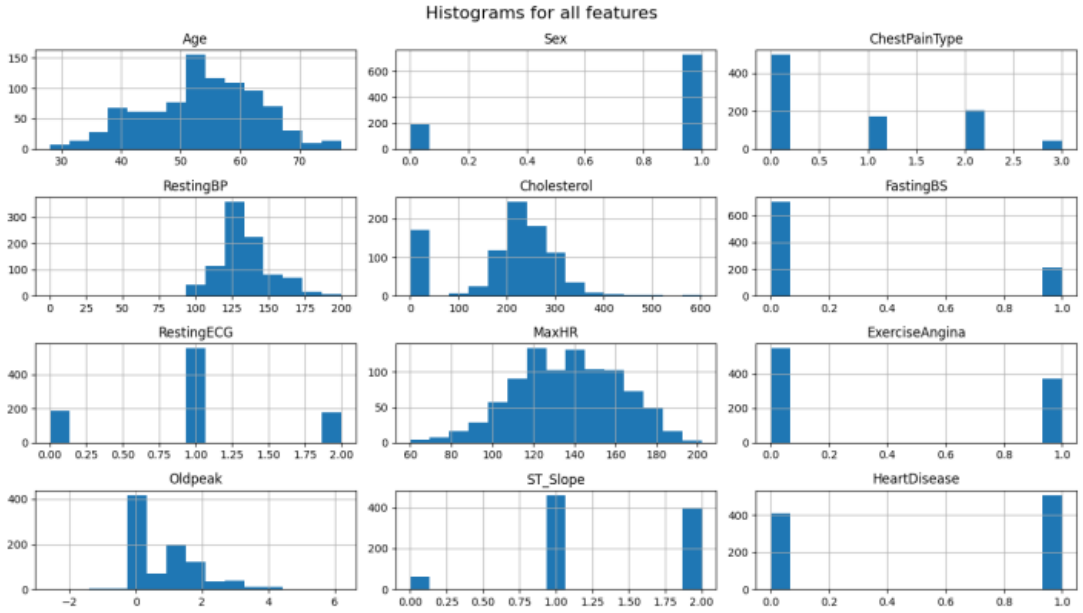
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Age             918 non-null    int64
 1   Sex             918 non-null    int32
 2   ChestPainType   918 non-null    int32
 3   RestingBP       918 non-null    int64
 4   Cholesterol     918 non-null    int64
 5   FastingBS       918 non-null    int64
 6   RestingECG      918 non-null    int32
 7   MaxHR           918 non-null    int64
 8   ExerciseAngina  918 non-null    int32
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    int32
 11  HeartDisease    918 non-null    int64
dtypes: float64(1), int32(5), int64(6)
```

```
Missing data:
 Age               0
Sex                0
ChestPainType      0
RestingBP          0
Cholesterol        0
FastingBS          0
RestingECG         0
MaxHR              0
ExerciseAngina     0
Oldpeak            0
ST_Slope           0
HeartDisease       0
dtype: int64
```

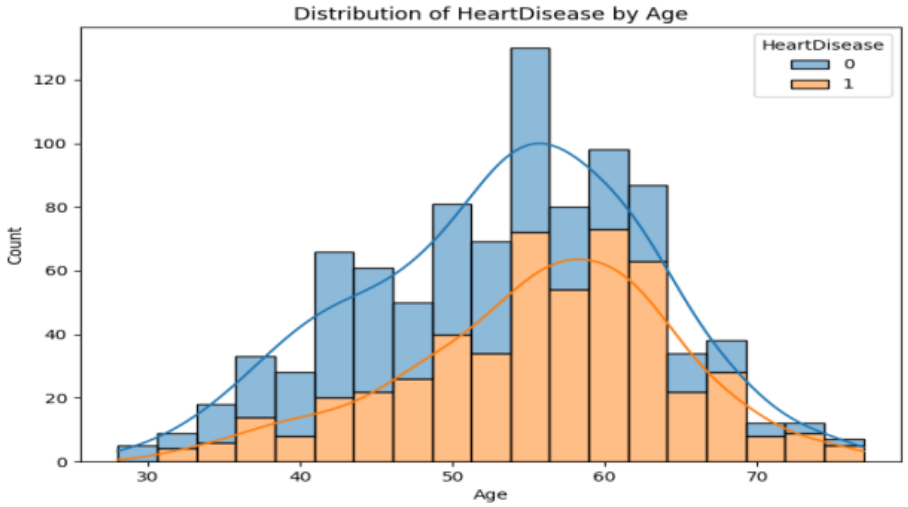
```
Number of samples in each class:
 HeartDisease
1    508
0    410
Name: count, dtype: int64
```

```
First 5 rows:
    Age  Sex  ChestPainType  ...  Oldpeak  ST_Slope  HeartDisease
0   40    1              1  ...      0.0         2             0
1   49    0              2  ...      1.0         1             1
2   37    1              1  ...      0.0         2             0
3   48    0              0  ...      1.5         1             1
4   54    1              2  ...      0.0         2             0

[5 rows x 12 columns]
```

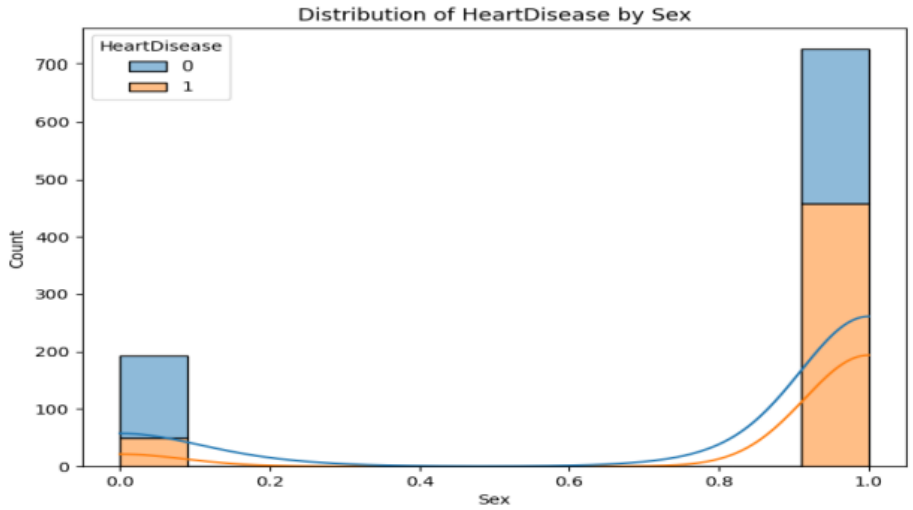
```
Database description:
                Age         Sex  ...     ST_Slope  HeartDisease
count   918.000000  918.000000  ...   918.000000    918.000000
mean     53.510893    0.789760  ...     1.361656      0.553377
std       9.432617    0.407701  ...     0.607056      0.497414
min      28.000000    0.000000  ...     0.000000      0.000000
25%      47.000000    1.000000  ...     1.000000      0.000000
50%      54.000000    1.000000  ...     1.000000      1.000000
75%      60.000000    1.000000  ...     2.000000      1.000000
max      77.000000    1.000000  ...     2.000000      1.000000
```



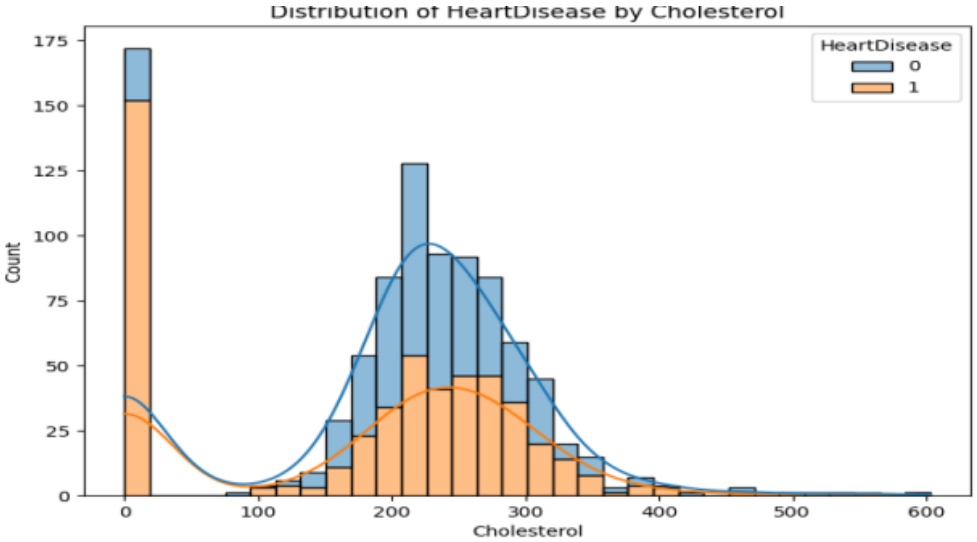
Rysunek 1: Histogram cech



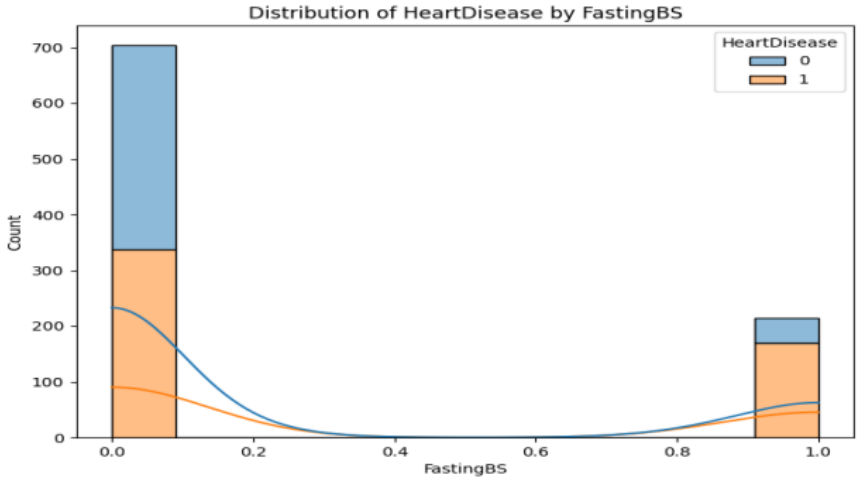
Rysunek 2: Rozkład klas względem wieku



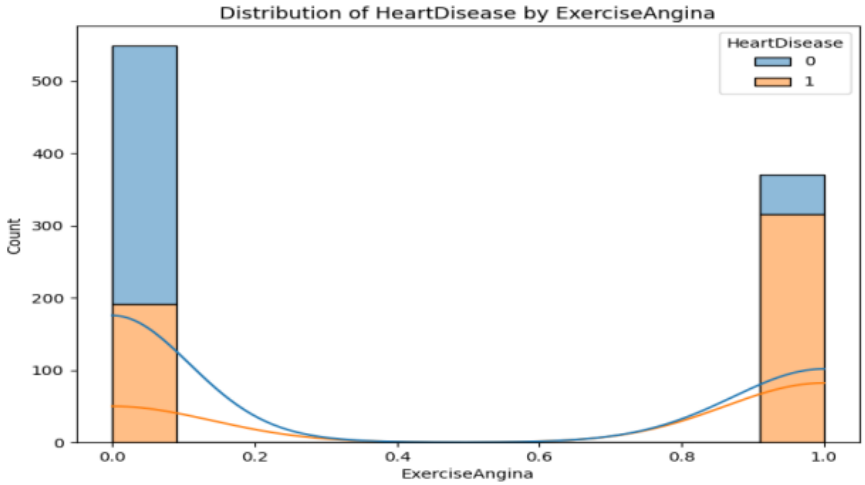
Rysunek 3: Rozkład klas względem płci



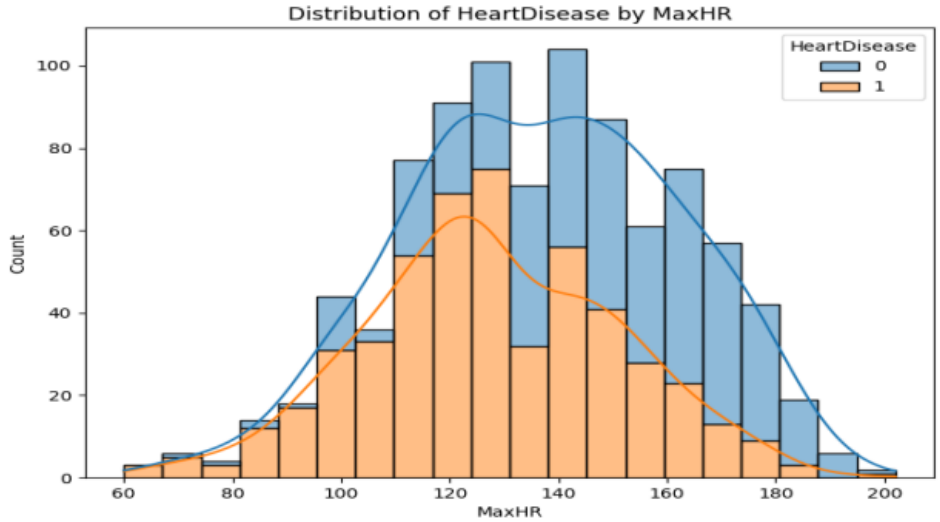
Rysunek 4: Rozkład klas względem poziomu cholesterolu



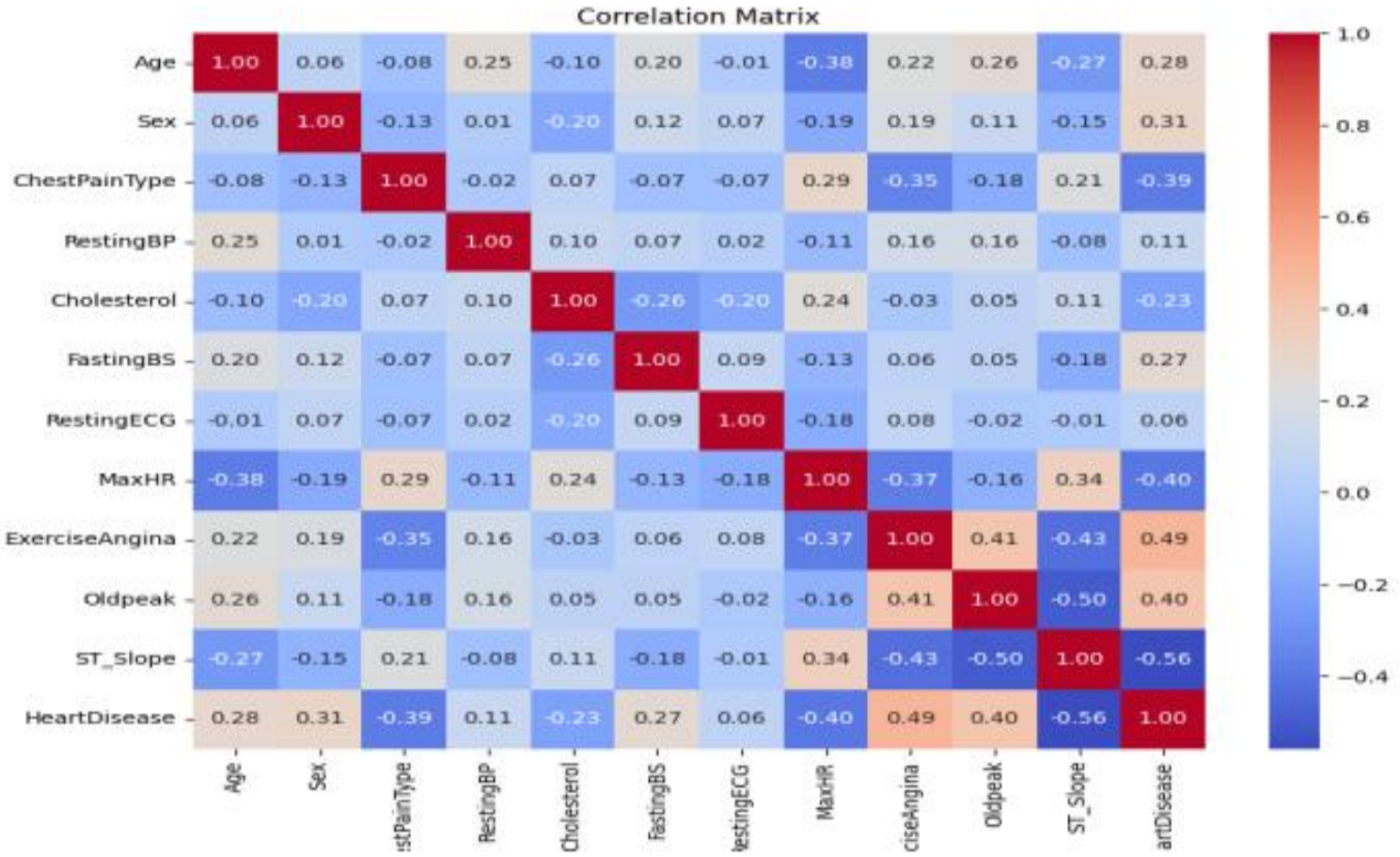
Rysunek 5: Rozkład klas względem poziomu cukru



Rysunek 7: Rozkład klas a dławica piersiowa spowodowana wysiłkiem



Rysunek 6: Rozkład klas względem maksymalnego tętna

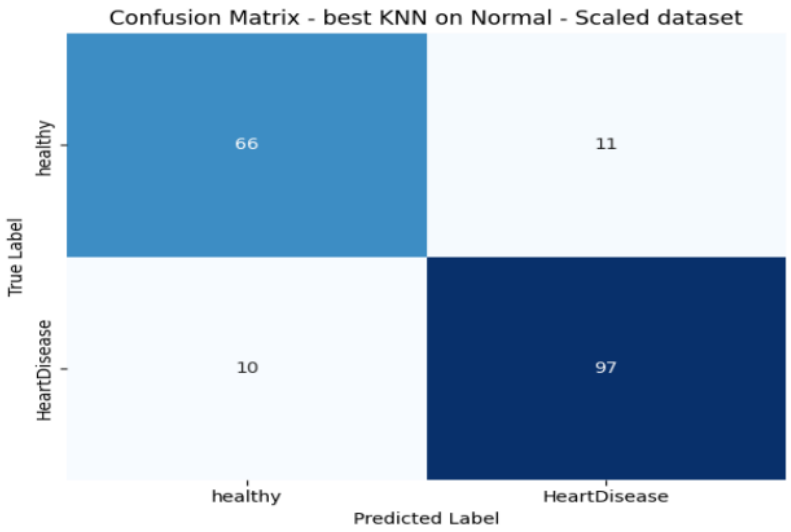


Rysunek 8: Macierz korelacji

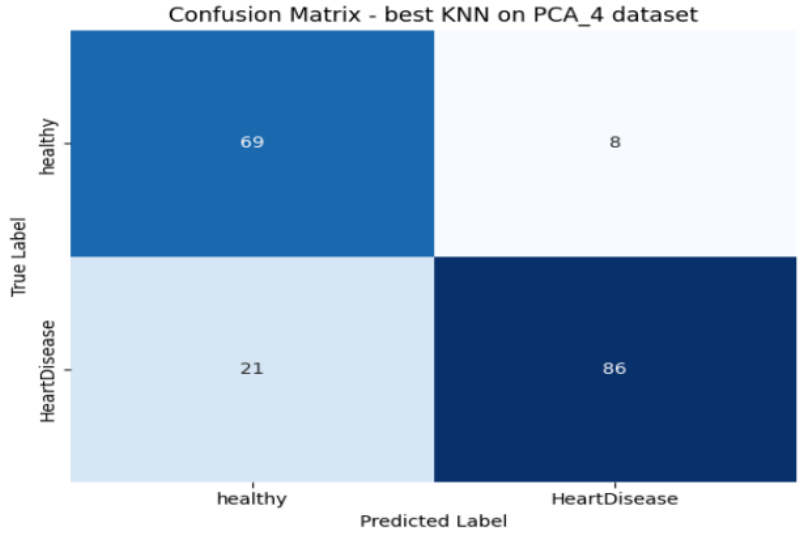
KROK 2 – PRZYGOTOWANIE 4 ZBIORÓW DANYCH

```
X_train, X_test, Y_train, Y_test = preprocess_data(dataset)

#Zbiory po redukcji wymiarowości
pca_4 = PCA(n_components=4)
pca_6 = PCA(n_components=6)
pca_8 = PCA(n_components=8)
X_train_pca_4 = pca_4.fit_transform(X_train)
X_test_pca_4 = pca_4.transform(X_test)
X_train_pca_6 = pca_6.fit_transform(X_train)
X_test_pca_6 = pca_6.transform(X_test)
X_train_pca_8 = pca_8.fit_transform(X_train)
X_test_pca_8 = pca_8.transform(X_test)


datasets = [
    ('Normal - Scaled', X_train, X_test, Y_train, Y_test),
    ('PCA_4', X_train_pca_4, X_test_pca_4, Y_train, Y_test),
    ('PCA_6', X_train_pca_6, X_test_pca_6, Y_train, Y_test),
    ('PCA_8', X_train_pca_8, X_test_pca_8, Y_train, Y_test)
]
```

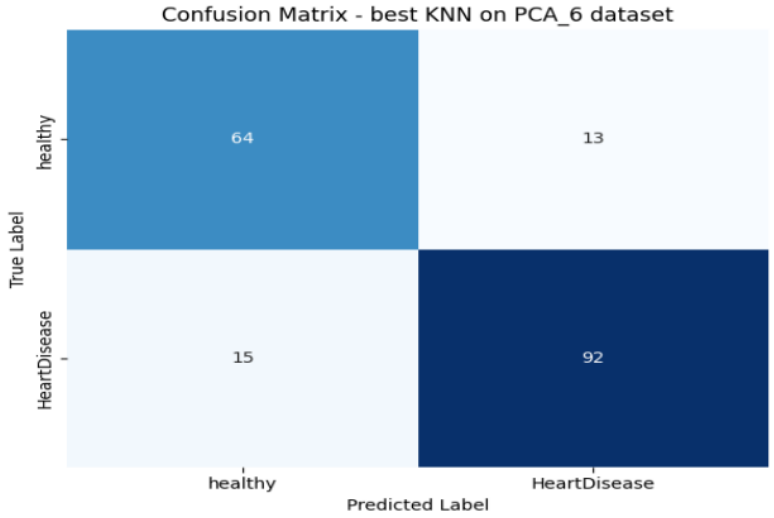
KROK 3 – SZUKANIE NAJLEPSZYCH PARAMETRÓW DLA KNN Z UŻYCIEM GRIDSEARCH NA KAŻDYM ZBIORZE



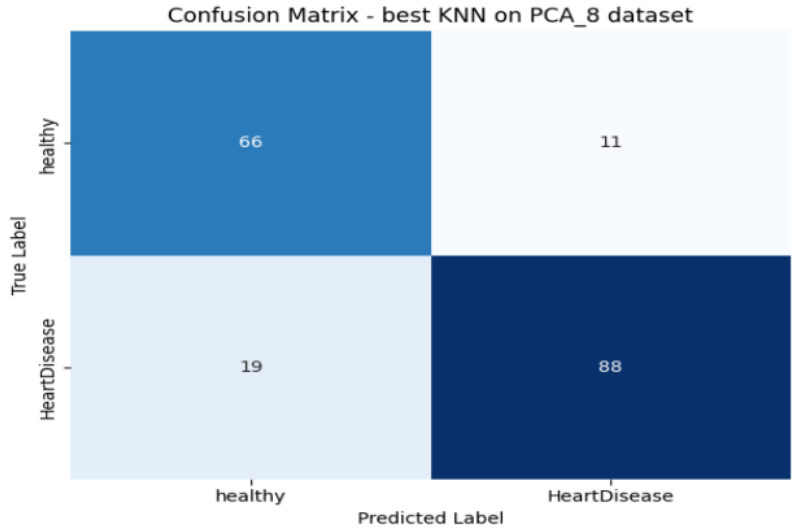
Rysunek 9: Najlepszy KNN na zbiorze przeskalowanym



Rysunek 10: Najlepszy KNN na zbiorze PCA 4



Rysunek 11: Najlepszy KNN na zbiorze PCA 6



Rysunek 12: Najlepszy KNN na zbiorze PCA 8

```
----- KNN on Normal - Scaled dataset -----

Best parameters found:  {'metric': 'canberra', 'n_neighbors': 11, 'weights': 'distance'}

Best cross-validation accuracy: 0.915246913580247

TEST ACCURACY: 0.8858695652173914

Classification Report:

              precision    recall  f1-score   support

           0       0.87      0.86      0.86        77
           1       0.90      0.91      0.90       107

    accuracy                           0.89       184
   macro avg       0.88      0.88      0.88       184
weighted avg       0.89      0.89      0.89       184
```

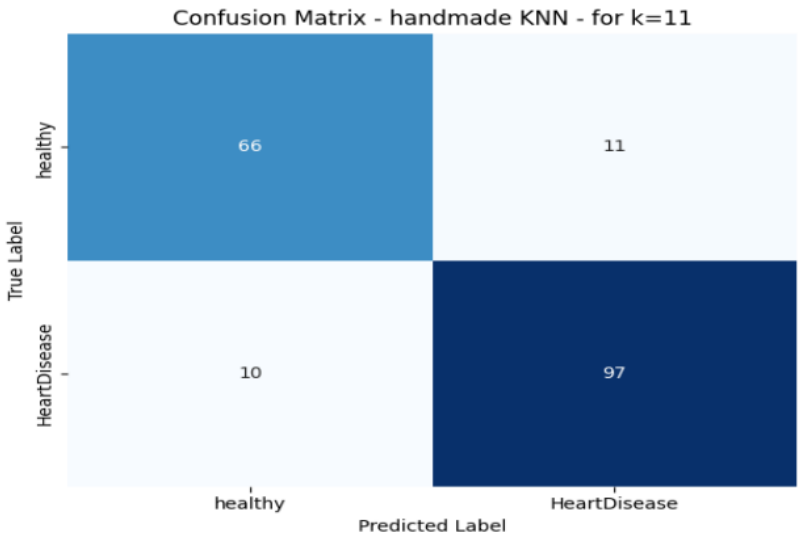
```
----- KNN on PCA_4 dataset -----

Best parameters found:  {'metric': 'euclidean', 'n_neighbors': 9, 'weights': 'distance'}

Best cross-validation accuracy: 0.8753703703703704

TEST ACCURACY: 0.842391304347826

Classification Report:

              precision    recall  f1-score   support

           0       0.77      0.90      0.83        77
           1       0.91      0.80      0.86       107

    accuracy                           0.84       184
   macro avg       0.84      0.85      0.84       184
weighted avg       0.85      0.84      0.84       184
```

```
----- KNN on PCA_6 dataset -----

Best parameters found:  {'metric': 'euclidean', 'n_neighbors': 6, 'weights': 'distance'}

Best cross-validation accuracy: 0.8878703703703705

TEST ACCURACY: 0.8478260869565217

Classification Report:

              precision    recall  f1-score   support

           0       0.81      0.83      0.82        77
           1       0.88      0.86      0.87       107

    accuracy                           0.85       184
   macro avg       0.84      0.85      0.84       184
weighted avg       0.85      0.85      0.85       184
```

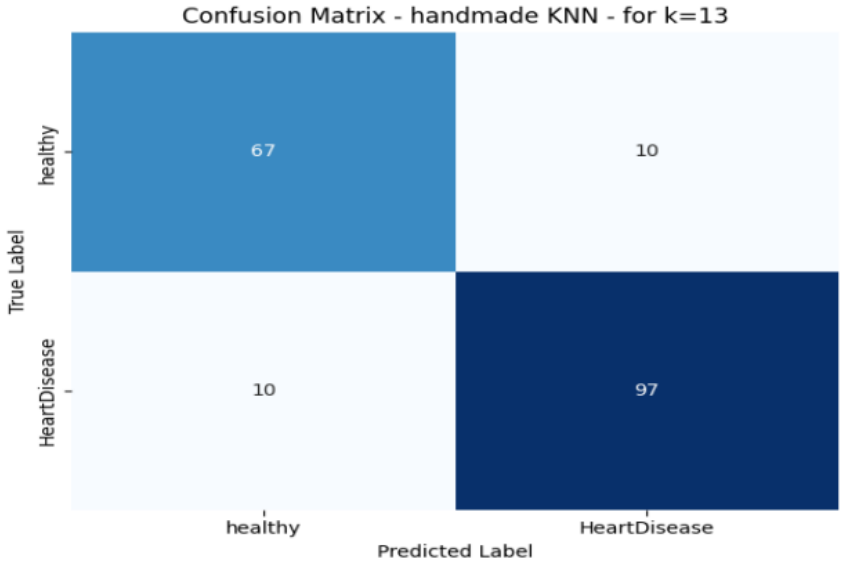
```
----- KNN on PCA_8 dataset -----

Best parameters found:  {'metric': 'euclidean', 'n_neighbors': 6, 'weights': 'distance'}

Best cross-validation accuracy: 0.8928395061728395

TEST ACCURACY: 0.8369565217391305

Classification Report:

              precision    recall  f1-score   support

           0       0.78      0.86      0.81        77
           1       0.89      0.82      0.85       107

    accuracy                           0.84       184
   macro avg       0.83      0.84      0.83       184
weighted avg       0.84      0.84      0.84       184
```

KROK 4 – NASZ KNN

Wyniki naszego KNN-a na zbiorze przeskalowanym, ponieważ wcześniej na nim uzyskaliśmy najlepsze wyniki. (najlepsze wyniki, co ciekawe, dla k=13 i k=17) (Wcześniej przy grid searchu najlepsze wyniki dla k=11)



Rysunek 13: Nasz KNN na zbiorze przeskalowanym dla k=11



Rysunek 14: Nasz KNN na zbiorze przeskalowanym dla k=13

FOR K=11

Accuracy: 0.8858695652173914

Classification Report:

	precision	recall	f1-score	support
0	0.87	0.86	0.86	77
1	0.90	0.91	0.90	107
accuracy			0.89	184
macro avg	0.88	0.88	0.88	184
weighted avg	0.89	0.89	0.89	184

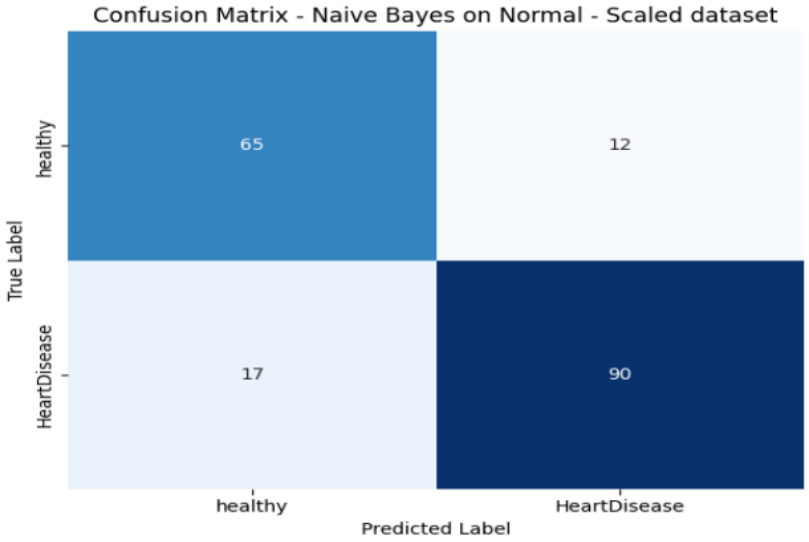
FOR K=13

Accuracy: 0.8913043478260869

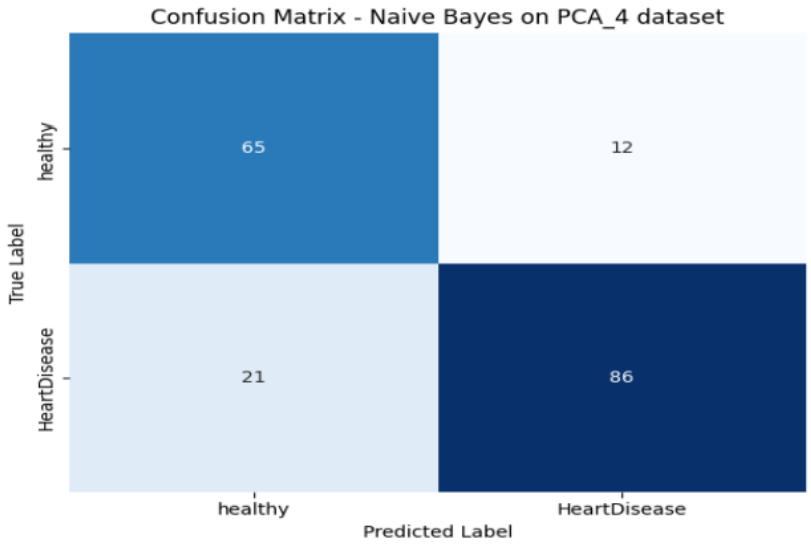
Classification Report:

	precision	recall	f1-score	support
0	0.87	0.87	0.87	77
1	0.91	0.91	0.91	107
accuracy			0.89	184
macro avg	0.89	0.89	0.89	184
weighted avg	0.89	0.89	0.89	184

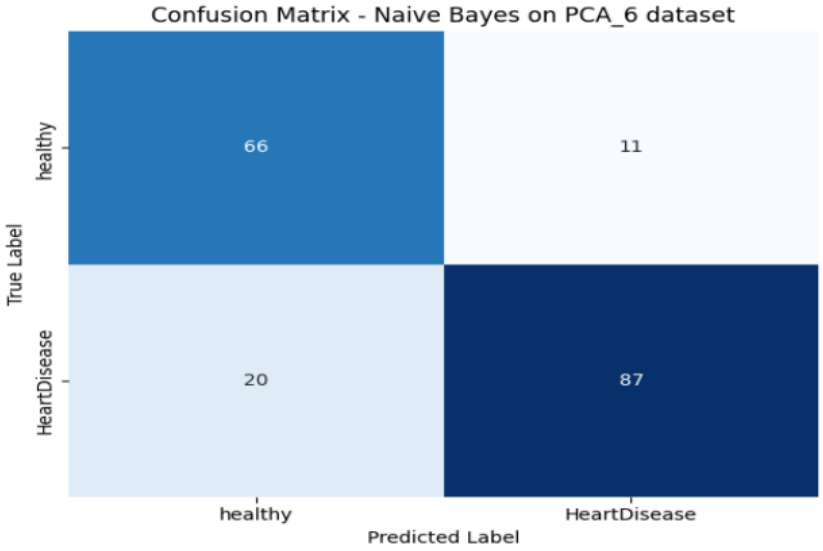
KROK 5 – NAIVE BAYES



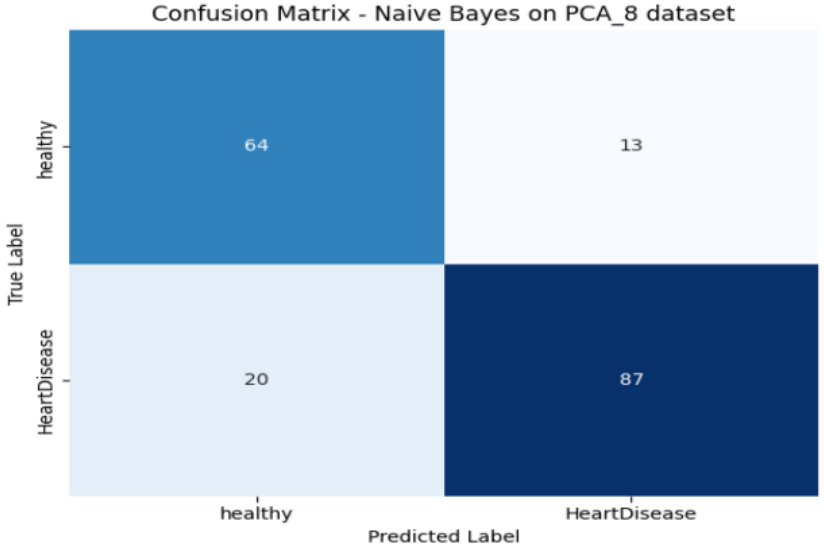
Rysunek 16: Naiwny Bayes na zbiorze przeskalowanym



Rysunek 17: Naiwny Bayes na zbiorze PCA 4



Rysunek 18: Naiwny Bayes na zbiorze PCA 6



Rysunek 19: Naiwny Bayes na zbiorze PCA 8

-----NAIVE BAYES, DATASET: Normal - Scaled

TEST ACCURACY: 0.842391304347826

Classification Report:

	precision	recall	f1-score	support
0	0.79	0.84	0.82	77
1	0.88	0.84	0.86	107
accuracy			0.84	184
macro avg	0.84	0.84	0.84	184
weighted avg	0.84	0.84	0.84	184

-----NAIVE BAYES, DATASET: PCA_4

TEST ACCURACY: 0.8206521739130435

Classification Report:

	precision	recall	f1-score	support
0	0.76	0.84	0.80	77
1	0.88	0.80	0.84	107
accuracy			0.82	184
macro avg	0.82	0.82	0.82	184
weighted avg	0.83	0.82	0.82	184

-----NAIVE BAYES, DATASET: PCA_6

TEST ACCURACY: 0.8315217391304348

Classification Report:

	precision	recall	f1-score	support
0	0.77	0.86	0.81	77
1	0.89	0.81	0.85	107
accuracy			0.83	184
macro avg	0.83	0.84	0.83	184
weighted avg	0.84	0.83	0.83	184

-----NAIVE BAYES, DATASET: PCA_8

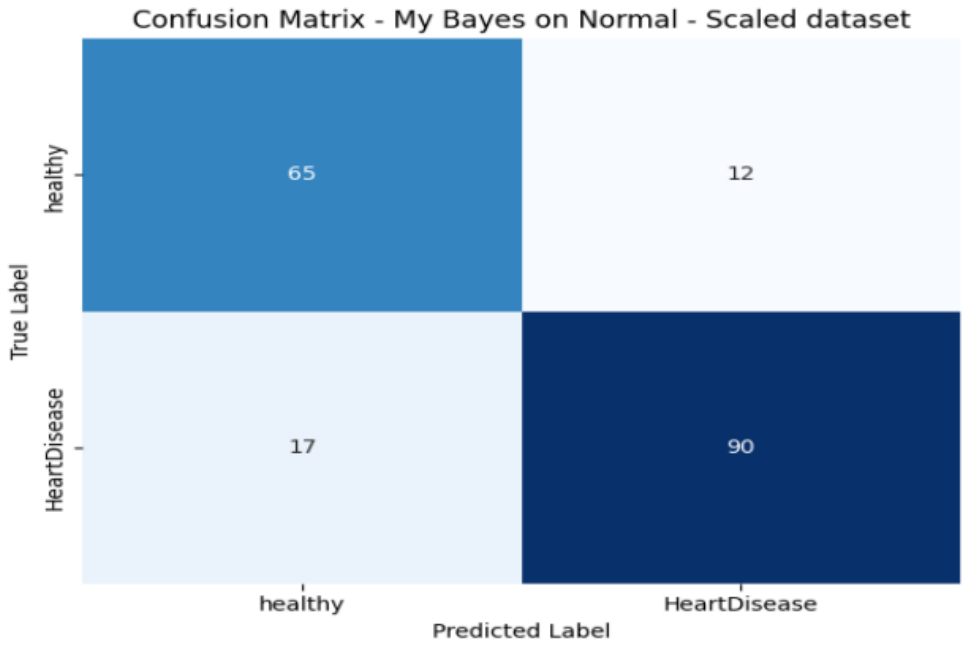
TEST ACCURACY: 0.8206521739130435

Classification Report:

	precision	recall	f1-score	support
0	0.76	0.83	0.80	77
1	0.87	0.81	0.84	107
accuracy			0.82	184
macro avg	0.82	0.82	0.82	184
weighted avg	0.82	0.82	0.82	184

KROK 6 – NASZ BAYES

Wyniki były dokładnie takie same jak dla Naive Bayes z Scikit-learn. Najlepszy wynik osiągnięto na pełnym zbiorze:

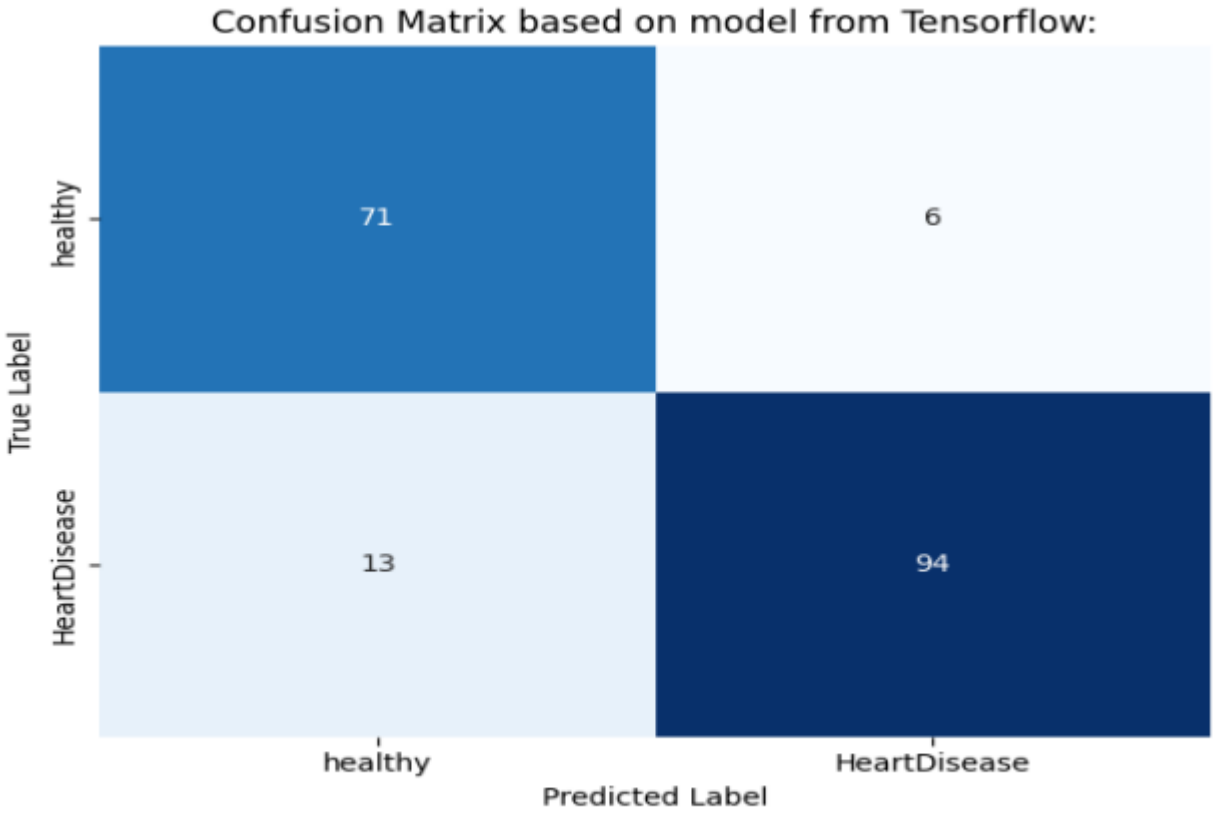


Rysunek 20: Nasz Bayes na zbiorze przeskalowanym

```
-----MY Naive BAYES, DATASET:  Normal - Scaled

TEST ACCURACY: 0.842391304347826

Classification Report:

              precision    recall  f1-score   support

           0       0.79      0.84      0.82        77
           1       0.88      0.84      0.86       107

    accuracy                           0.84       184
   macro avg       0.84      0.84      0.84       184
weighted avg       0.84      0.84      0.84       184
```

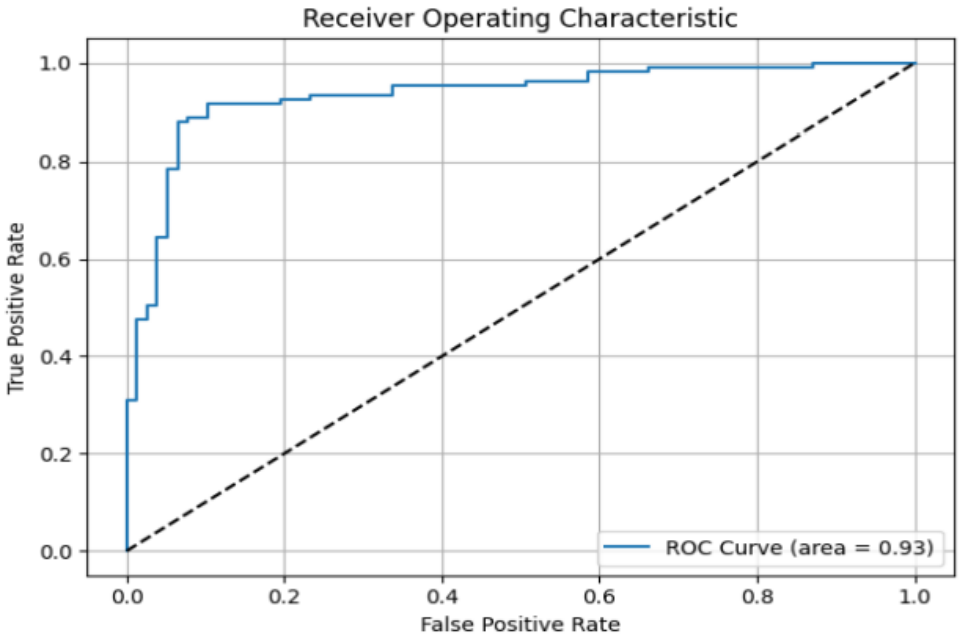
KROK 7 – DODATKOWE PORÓWNANIE Z PROSTĄ SIECIĄ

Prosta sieć w celach porównawczych do naszych poprzednich wyników:



Rysunek 21: Prosta sieć neuronowa z TensorFlow

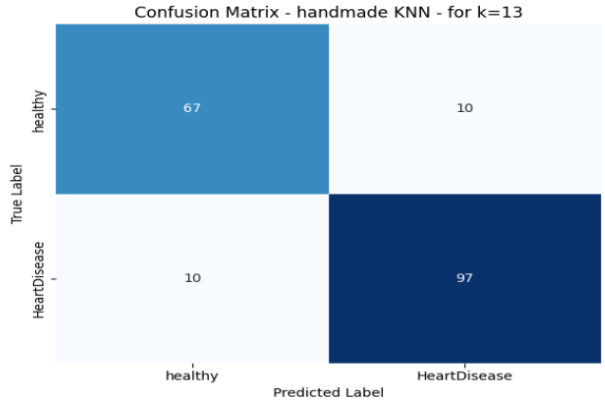
```
Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.92      0.88        77
           1       0.94      0.88      0.91       107

    accuracy                           0.90       184
   macro avg       0.89      0.90      0.90       184
weighted avg       0.90      0.90      0.90       184

ROC AUC Score: 0.9343366913460371
```



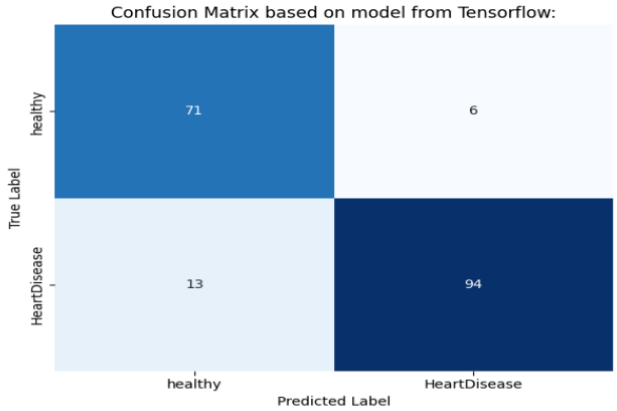
Rysunek 22: Wykres dla prostej sieci neuronowej z TensorFlow

PODSUMOWANIE:

Jeśli chodzi o metrykę recall - ważną pod kątem medycznym - to najlepsze parametry uzyskaliśmy dla naszego KNN, następnie dla KNN z Sk-learna, później dla sieci neuronowej, a najsłabsze dla Naiwnych Bayesów, co potwierdziło nasze wstępne przypuszczenia.
Porównanie naszego KNN (k=13) z siecią:



Rysunek 14: Nasz KNN na zbiorze przeskalowanym dla k=13



Rysunek 21: Prosta sieć neuronowa z TensorFlow

FOR K=13
Accuracy: 0.8913043478260869
Classification Report:

	precision	recall	f1-score	support
0	0.87	0.87	0.87	77
1	0.91	0.91	0.91	107
accuracy			0.89	184
macro avg	0.89	0.89	0.89	184
weighted avg	0.89	0.89	0.89	184

Classification Report:

	precision	recall	f1-score	support
0	0.85	0.92	0.88	77
1	0.94	0.88	0.91	107
accuracy			0.90	184
macro avg	0.89	0.90	0.90	184
weighted avg	0.90	0.90	0.90	184

ROC AUC Score: 0.9343366913460371

PODSUMOWANIE CZ.2:

Najlepsza pod względem accuracy okazała się sieć neuronowa, którą możemy dzięki temu uznać za najbardziej zbilansowany model, jednakże jeśli zależy nam na wykryciu jak największej liczby przypadków czyli metryce recall, to najlepszy okazuje się nasz model KNN dla k=13.

DZIĘKUJEMY ZA UWAGĘ!