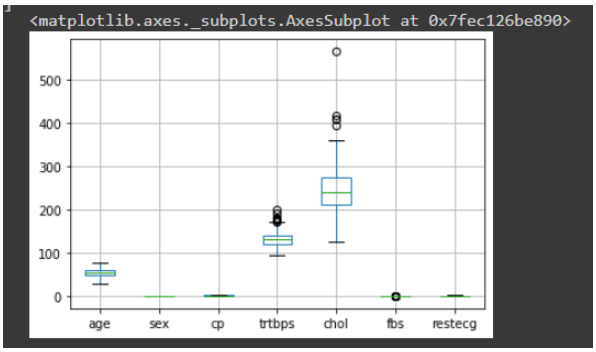
Nama Anggota:
Leonardo Ignatius - 2301853123
Kezia Nathania - 2300000012
Adrian Rianto - 2301859335

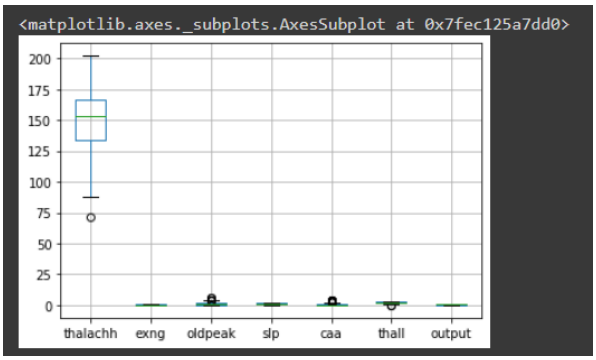
Laporan Machine Learning

(Classifying Heart Disease by Comparing Logistic Regression, SVM and KNn)

1. Background:
   Algoritma klasifikasi yang kami gunakan adalah Logistic Regression, Support Vector Machine, dan K-Nearest Neighbor. Logistic Regression dipilih karena merupakan algoritma yang simpel tapi sangat powerfull dan tepat untuk digunakan pada binary classification problem, seperti dataset kami yang memiliki output 0 untuk kelas 'No heart disease' dan 1 untuk kelas 'Heart disease' [1,3]. SVM digunakan karena SVM dapat mengklasifikasikan data yang non-linearly separable menggunakan kernel trick dengan melakukan mapping data input ke high-dimensional feature space. Kami menggunakan SVC dari Sklearn.SVM karena SVC merupakan algorithm SVM yang tidak mudah dipengaruhi oleh outlier [1]. Logistic regression dan SVM banyak dipakai dalam bidang healthcare karena memiliki aplikasi yang luas dalam bidang healthcare[2], logistic regression dan SVM merupakan salah satu model yang paling banyak dipakai dalam bidang kesehatan[1,6,7,8]. KNN digunakan sebagai pembanding karena algoritma tersebut merupakan memory based approach sehingga dapat beradaptasi dengan data training baru. Kami ingin membandingkan apakah algoritma yang memakan memori lebih banyak seperti SVM dan Logistic Regression akan menghasil akurasi yang lebih baik dibandingkan dengan KNN yang lebih kecil [1,4,5].

2. Preprocessing:
   a. Drop Outlier:

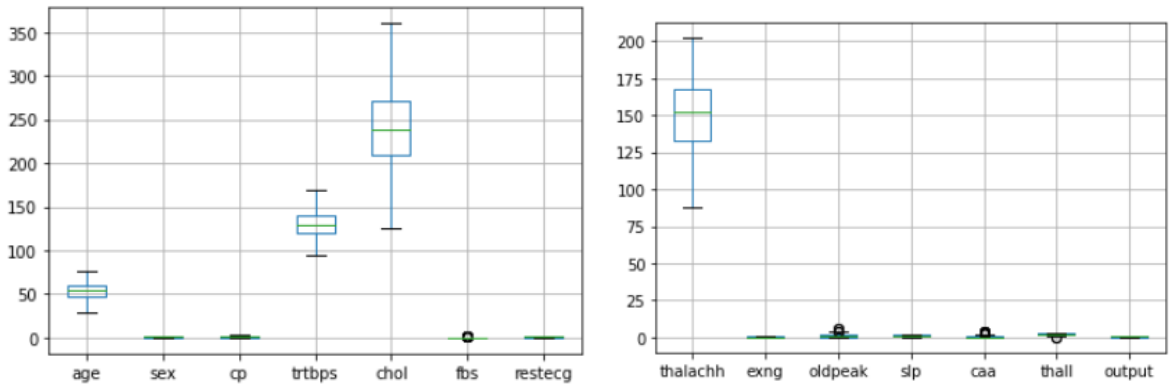




Jika kami melakukan boxplot pada seluruh column yang ada pada dataset, kita dapat melihat column 'trtbps', 'chol' dan 'thalachh' memiliki outliers, maka itu kami akan membuang outliers tersebut dengan menggunakan IQR, dengan code sebagai berikut:

```
def dropoutlier(df,dfcol):
    sorted(dfcol)
    q1 = np.percentile(dfcol,25)
    q3 = np.percentile(dfcol,75)
    iqr = q3 - q1
    lower = q1 - (1.5 * iqr)
    upper = q3 + (1.5 * iqr)
    indexoutlier = df[(dfcol < lower) | (dfcol > upper)].index
    return df.drop(indexoutlier,inplace = True)
```

Kami melakukan sorting dan menghapus data yang lebih kecil dari lower bound dan yang lebih besar dari upper bound.



b. StandardScaler:
Standard Scaler menggunakan Standard Normal distribution untuk mengscale data, kami menggunakan StandardScaler Karena Dataset yang kita miliki sudah diketahui bentangannya, jadi kami menggunakan StandardScaler untuk menskala X dari dataset

3. Feature Selection

```
output      1.000000
oldpeak     0.444884
thalachh    0.431206
exng        0.431165
cp          0.426658
caa         0.388963
thall       0.347019
slp         0.345785
sex         0.313090
age         0.228234
restecg     0.154995
trtbps      0.120231
chol        0.109798
fbs         0.021873
Name: output, dtype: float64
```

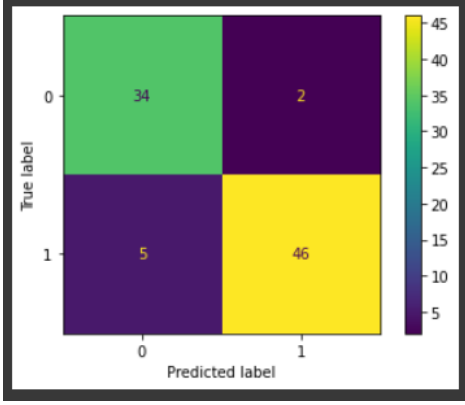
4. Untuk Feature yang kami pilih, kami menggunakan Feature yang berkorelasi dengan column output dengan nilai lebih dari 0.3, maka dari itu kami menggunakan column 'oldpeak','thalachh','cp','caa','slp','thall','slp','sex'. Kami tidak menggunakan column ‘exng’ karena memiliki korelasi yang tinggi dengan column ‘thalachh’.

5. Validation
   Kami Menggunakan Train Test Split Dengan Ratio 7:3 dimana 3 adalah Test Size, dengan random state 42 yang nantinya akan menghasilkan 4 variable yaitu x_train,x_test,y_train,y_test.

6. Logistic Regression
   Hasil Sklearn:

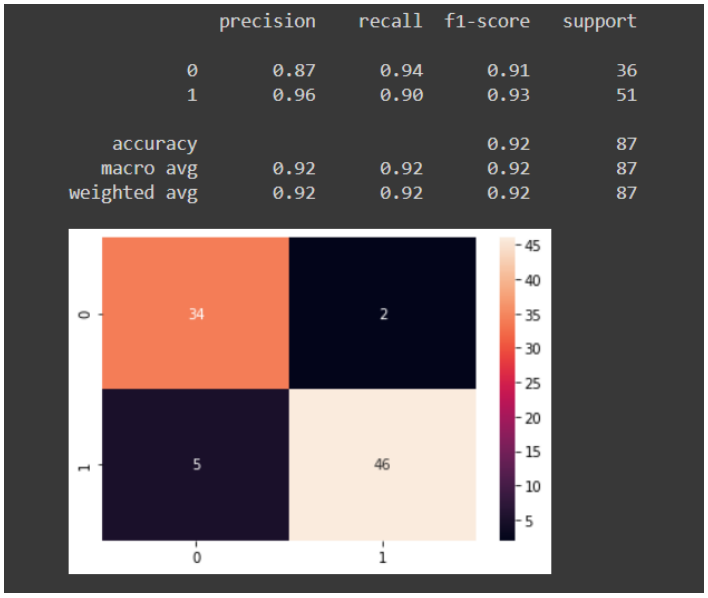
```
logistic_regression = LogisticRegression().fit(x_train,y_train)
y_pred_logistic = logistic_regression.predict(x_test)
accuracy_logistic = logistic_regression.score(x_test,y_test)

print(f"Accuracy logistic Regression : {accuracy_logistic*100: .2f}%")

cm_logistic = plot_confusion_matrix(logistic_regression,x_test,y_test)

print(classification_report(y_test,y_pred_logistic))
```

```
Accuracy logistic Regression :  91.95%
              precision    recall  f1-score   support

           0       0.87      0.94      0.91        36
           1       0.96      0.90      0.93        51

    accuracy                           0.92        87
   macro avg       0.92      0.92      0.92        87
weighted avg       0.92      0.92      0.92        87
```



Berikut adalah code dari penggunaan sklearn kami dilengkapi dengan confusion matrix dan classification report. Dapat dilihat bahwa Logistic Regression menggunakan Sklearn Menghasilkan Accuracy Sebesar 91.95%.

Hasil Manual:



	precision	recall	f1-score	support
0	0.87	0.94	0.91	36
1	0.96	0.90	0.93	51
accuracy			0.92	87
macro avg	0.92	0.92	0.92	87
weighted avg	0.92	0.92	0.92	87

Berikut adalah confusion matrix dan classification report. Dapat dilihat bahwa Logistic Regression Secara manual dengan 10000 epoch dan learning rate 0.3 Menghasilkan Accuracy Sebesar 91.95%

Yang terjadi pada algorithm ini adalah:

- Inisialisasi Weight dan Bias dari model
- Kemudian kita harus mengupdate / mengoptimalkan weight dan bias tersebut menggunakan gradient descent dengan cara menurunkan Weight dan bias tersebut lalu dikalikan dengan learning rate sampai loss dari prediksi tersebut menjadi kecil.
- Kemudian jika sudah weight dan bias tersebut dapat kita gunakan untuk mendapatkan prediksi dengan cara mengalikan input dengan weight lalu ditambah bias dan menggunakan fungsi aktivasi sigmoid untuk mendapatkan hasil dari klasifikasi

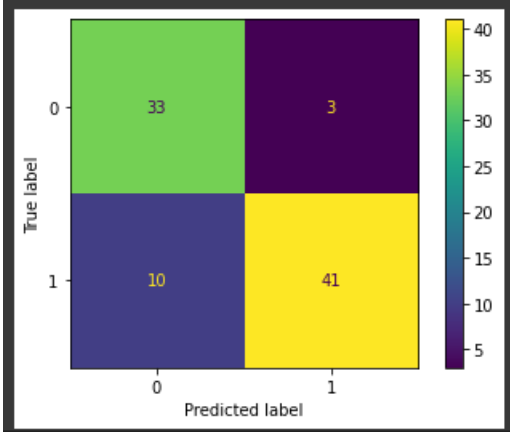
Sigmoid

$$S(x) = \frac{1}{1 + e^{-x}}$$

7. SVM

```
SVC = SVC().fit(x_train,y_train)
y_pred_svc = SVC.predict(x_test)
accuracy_svc = SVC.score(x_test,y_test)

print(f"Accuracy SVC: {accuracy_svc*100: .2f}%")

cm_SVC = plot_confusion_matrix(SVC,x_test,y_test)
print(classification_report(y_test,y_pred_svc))
```

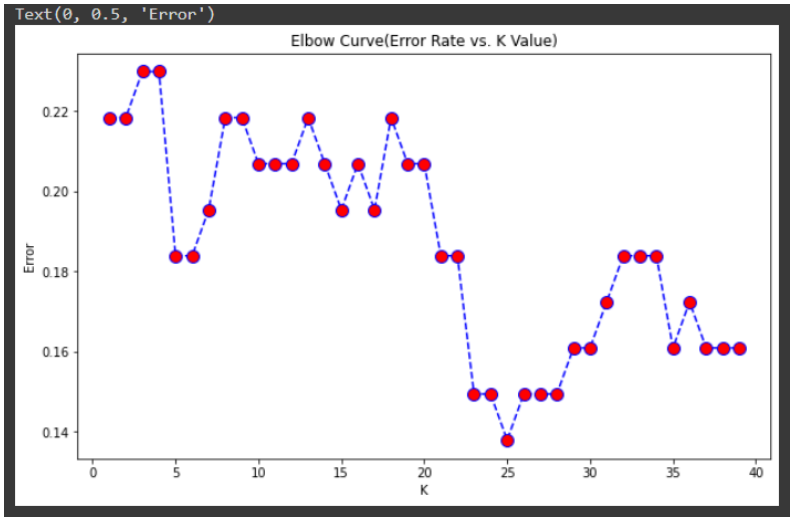
```
Accuracy SVC:  85.06%
              precision    recall  f1-score   support

           0       0.77      0.92      0.84        36
           1       0.93      0.80      0.86        51

    accuracy                           0.85        87
   macro avg       0.85      0.86      0.85        87
weighted avg       0.86      0.85      0.85        87
```



Berikut adalah code dari penggunaan sklearn support vector classification kami, dilengkapi dengan confusion matrix dan classification report. Dalam melakukan klasifikasi dengan SVM, kernel function yang kami gunakan adalah Radial Basis Function (RBF) karena dataset yang digunakan tidak linearly separable.

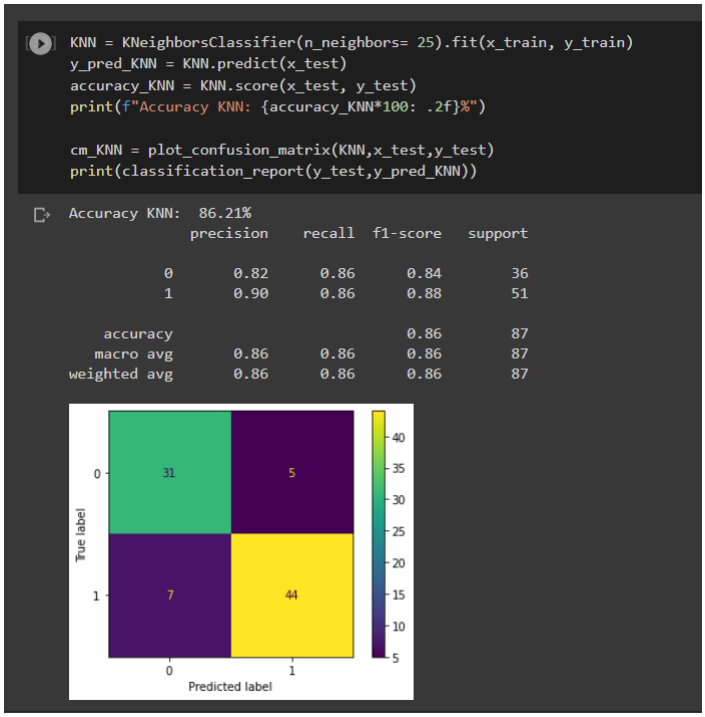
Akurasi yang kami dapat adalah 85.06%.

8. KNN



Pertama-tama kami menggunakan elbow curves terlebih dahulu untuk mengetahui jumlah k berapa yang terbaik/ memiliki loss yang paling rendah, K yang kami dapatkan adalah 25.

Pada project ini kami menggunakan sklearn untuk membantu dalam klasifikasi, dengan code sebagai berikut:



```
KNN = KNeighborsClassifier(n_neighbors= 25).fit(x_train, y_train)
y_pred_KNN = KNN.predict(x_test)
accuracy_KNN = KNN.score(x_test, y_test)
print(f"Accuracy KNN: {accuracy_KNN*100: .2f}%")

cm_KNN = plot_confusion_matrix(KNN,x_test,y_test)
print(classification_report(y_test,y_pred_KNN))
```

```
Accuracy KNN:  86.21%
              precision    recall  f1-score   support

           0       0.82      0.86      0.84        36
           1       0.90      0.86      0.88        51

    accuracy                           0.86        87
   macro avg       0.86      0.86      0.86        87
weighted avg       0.86      0.86      0.86        87
```


Yang menghasilkan Accuracy sebesar 86.21% dan Confusion matrix dan classification report sebagai berikut. Kemudian cara kerja dari knn sendiri adalah:

- Menghitung Euclidean Distance
- Mendapatkan Point terdekat
- Membuat Prediksi berdasarkan mayoritas label sebanyak K dari tetangga poin tersebut

9. Metrics

   Untuk menghitung Accuracy dari model kami, kami menggunakan function score yang disediakan dari sklearn yang mengambil parameter(y_test,y_pred). Rumus dari fungsi score ini sama dengan R2 score yaitu :

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

   F1 Score:

$$F_1 = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} = \frac{\mathrm{TP}}{\mathrm{TP} + \frac{1}{2}(\mathrm{FP} + \mathrm{FN})}$$

   F1 score adalah penilaian model kita dimana dilihat dari precision dan recall.

   Recall:

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

   Recall adalah seberapa banyak prediksi yang benar dibandingkan dengan jumlah prediksi kita untuk class tertentu, dengan kata lain recall merupakan accuracy model kita dalam mengklasifikasi suatu class (dan hanya class itu saja).

   Precision:

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

   Precision adalah kemampuan classifier untuk tidak melabeli kelas positif yang sebenarnya negatif.

10. Conclusion

    Dari dataset tersebut hasil yang kami dapatkan adalah Logistic Regression memiliki accuracy paling tinggi dengan 91.95%, kemudian K Nearest Neighbor dengan 86.21% dan SVM dengan 85.06%

REFERENCE :

[1] S. Dreiseitl and L. Ohno-Machado, “Logistic regression and artificial neural network classification models: a methodology review,” Journal of Biomedical Informatics, vol. 35, no. 5–6, pp. 352–359, Oct. 2002, doi: 10.1016/s1532-0464(03)00034-0.

[2]M. Sotelo, “Introduction to Logistic Regression in Health Care Applications,” Medium, Nov.17,2020.
https://blog.mpirik.com/introduction-to-logistic-regression-in-health-care-applications-d449cacc9bc7 (accessed Jun. 19, 2021).

[3]https://www.facebook.com/kdnuggets, “Logistic Regression: A Concise Technical Overview-KDnuggets,”KDnuggets,2019.
https://www.kdnuggets.com/2019/01/logistic-regression-concise-technical-overview.html#:~:text=Logistic%20Regressions%20are%20some%20of,powerful%20techniques%20in%20Machine%20Learning.&text=Many%20believe%20Logistic%20Regression%20to,helps%20to%20understand%20Neural%20Networks (accessed Jun. 19, 2021).

[4]https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm#:~:text=In%20statistics%2C%20the%20k%2Dnearest,training%20examples%20in%20data%20set.

[5] https://www.ibm.com/docs/en/db2woc?topic=procedures-k-nearest-neighbors-knn

[6] Janardhanan, Padmavathi & Heena, L. & Sabika, Fathima. (2015). Effectiveness of Support Vector Machines in Medical Data mining. Journal of Communications Software and Systems. 11. 25-30. 10.24138/jcomss.v11i1.114.

[7] https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/

[8] https://en.wikipedia.org/wiki/Support-vector_machine