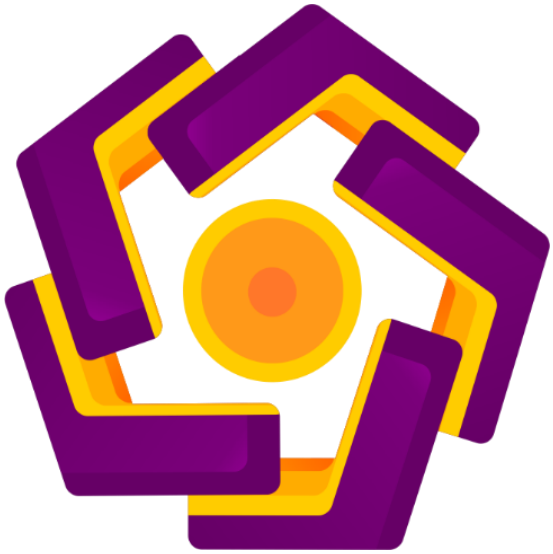
LAPORAN
UJIAN AKHIR SEMESTER
BIG DATA & PREDICTIVE ANALYTICS LANJUT

Dosen Pengampu :
Irwan Oyong, M.Kom



Disusun Oleh :

Sandi Hernanto	21.11.4530
Rifan Azis Ardiansyah	21.11.4525
Farhan Syah Friyanto	21.11.4482
Dino Alfiano	21.11.4540

S1 INFORMATIKA
UNIVERSITAS AMIKOM YOGYAKARTA
2024

1. **Pemilihan Topik**

   a. **Bidang yang dipilih:**

   Bidang kesehatan, khususnya klasifikasi untuk memprediksi penyakit jantung

   Alasan Pemilihan Bidang:

   1. Relevansi dan Dampak Sosial:

      Penyakit jantung adalah penyebab kematian utama di seluruh dunia. Menganalisis data terkait penyakit jantung dapat memberikan wawasan yang berharga untuk mencegah, mendeteksi, dan mengelola penyakit ini.

   2. Pentingnya Pencegahan:

      Fokus pada penyakit jantung memungkinkan pengembangan model machine learning untuk prediksi risiko individu. Hal tersebut dapat membantu dalam upaya pencegahan, seperti perubahan gaya hidup atau penanganan medis dini.

   3. Keanekaragaman Fitur:

      Dataset penyakit jantung umumnya memiliki keanekaragaman fitur, termasuk faktor risiko seperti tekanan darah, kadar kolesterol, aktivitas fisik, dan riwayat keluarga. Ini memberikan peluang untuk analisis yang komprehensif.

   4. Pentingnya Diagnosis Dini:

      Model machine learning dapat membantu dalam diagnosis dini penyakit jantung. Hal ini memungkinkan intervensi yang lebih cepat dan meningkatkan peluang kesembuhan atau manajemen penyakit.

   Tujuan Pemilihan Topik:

   - Mengembangkan model klasifikasi yang dapat memprediksi risiko penyakit jantung berdasarkan faktor-faktor risiko tertentu.
   - Memberikan kontribusi positif terhadap upaya pencegahan dan manajemen penyakit jantung melalui analisis data yang cermat.

   b. **Proses mendapatkan data dan informasi lengkap mengenai dataset heart disease (Penyakit Jantung):**

Pada tanggal 6 Januari 2024, saya mengakses dataset heart disease dari kaggle. Dataset ini tersedia secara gratis dan dapat diunduh dengan mudah.

Dataset ini berisi 13 fitur

Data ini dapat diakses secara gratis di situs Kaggle. Untuk mendapatkan data ini, kita dapat melakukan langkah-langkah berikut:

1. Buka situs Kaggle.

2. Cari dataset Heart Disease UCI Machine Learning Repository.
3. Klik tombol Download.

Setelah data diunduh, kita dapat membukanya menggunakan aplikasi spreadsheet atau pengolah data lainnya.

Berikut adalah penjelasan setiap kolom dalam dataset ini:

- age: Usia pasien dalam tahun.
- sex: Jenis kelamin pasien, yaitu laki-laki (1) atau perempuan (0).
- cp: Jenis nyeri dada, yaitu:
  - Tidak ada (0)
  - Tidak khas (1)
  - Tipe angina klasik (2)
  - Tipe angina varian (3)
- trestbps: Tekanan darah istirahat dalam mmHg.
- chol: Kolesterol total dalam mg/dL.
- fbs: Kadar gula darah puasa, yaitu:
  - Normal (0)
  - Tinggi (1)
- restecg: Hasil elektrokardiogram, yaitu:
  - Normal (0)
  - Tidak normal (1)
  - Tindak lanjut (2)
- thalach: Detak jantung maksimum dalam bpm.
- exang: Gejala angina saat berolahraga, yaitu:
  - Tidak ada (0)
  - Ya (1)
- oldpeak: Depresi ST gelombang Q tertinggi dalam mm.
- slope: Kemiringan ST gelombang Q, yaitu:
  - Naik (1)
  - Datar (2)
  - Turun (3)
- ca: Jumlah pembuluh koroner yang tersumbat, yaitu:

- 0 (0)
- 1 (1)
- 2 (2)
- 3 (3)
- 4 (4)

- thal: Kondisi thalassemia, yaitu;
  - Normal (1)
  - Cacat tetap (2)
  - Cacat dapat dipulihkan (3)

Data ini masih valid untuk digunakan dalam penelitian saat ini. Data ini dikumpulkan dengan metode yang terstandarisasi dan telah divalidasi oleh para ahli. Dataset ini terakhir update 5 tahun lalu oleh David Lapp.

C. Pre-processing data

1. Memeriksa tipe data

```
# Memeriksa tipe data
data.printSchema()
```

```
root
 |-- age: integer (nullable = true)
 |-- sex: integer (nullable = true)
 |-- cp: integer (nullable = true)
 |-- trestbps: integer (nullable = true)
 |-- chol: integer (nullable = true)
 |-- fbs: integer (nullable = true)
 |-- restecg: integer (nullable = true)
 |-- thalach: integer (nullable = true)
 |-- exang: integer (nullable = true)
 |-- oldpeak: double (nullable = true)
 |-- slope: integer (nullable = true)
 |-- ca: integer (nullable = true)
 |-- thal: integer (nullable = true)
 |-- target: integer (nullable = true)
```

Terdapat 13 variabel dengan tipe data integer dan 1 variabel (oldpeak) dengan tipe double.

2. Mengganti nama kolom

```
# Mengganti nama kolom
new_column_names = ["age", "sex", "chest_pain", "resting_bp", "cholesterol", "fasting_blood_sugar",
                    "resting_ecg", "max_heart_rate", "exercise_angina", "old_peak", "slope", "vessels_colored", "thal", "target"]
data = data.toDF(*new_column_names)
```

Mengganti nama kolom cp menjadi chest_paint, trestbps menjadi resting_bp, chol menjadi cholesterol, fbc menjadi fasting_blood_sugar, restecg menjadi resting_ecg, thalach menjadi max_heart_rate, exang menjadi exercise_angina, oldpeak menjadi old_peak, ca menjadi vessels_colored.

3. Memeriksa nilai null

```
[42] # Memeriksa nilai null
     null_counts = [data.filter(col(c).isNull()).count() for c in data.columns]
     null_counts_dict = dict(zip(data.columns, null_counts))

     for column, count in null_counts_dict.items():
         print(f"Jumlah nilai null dalam kolom {column}: {count}")
```

```
Jumlah nilai null dalam kolom age: 0
Jumlah nilai null dalam kolom sex: 0
Jumlah nilai null dalam kolom chest_pain: 0
Jumlah nilai null dalam kolom resting_bp: 0
Jumlah nilai null dalam kolom cholesterol: 0
Jumlah nilai null dalam kolom fasting_blood_sugar: 0
Jumlah nilai null dalam kolom resting_ecg: 0
Jumlah nilai null dalam kolom max_heart_rate: 0
Jumlah nilai null dalam kolom exercise_angina: 0
Jumlah nilai null dalam kolom old_peak: 0
Jumlah nilai null dalam kolom slope: 0
Jumlah nilai null dalam kolom vessels_colored: 0
Jumlah nilai null dalam kolom thal: 0
Jumlah nilai null dalam kolom target: 0
```

Tidak terdapat nilai null pada dataset heart.

4. Mengubah tipe data (agar bisa di proses)

```
[43] # Mengubah tipe data
     data = data.withColumn("sex", col("sex").cast("float"))
     data = data.withColumn("target", col("target").cast("float"))
```

Merubah tipe data sex dan target menjadi float

5. Menampilkan summary

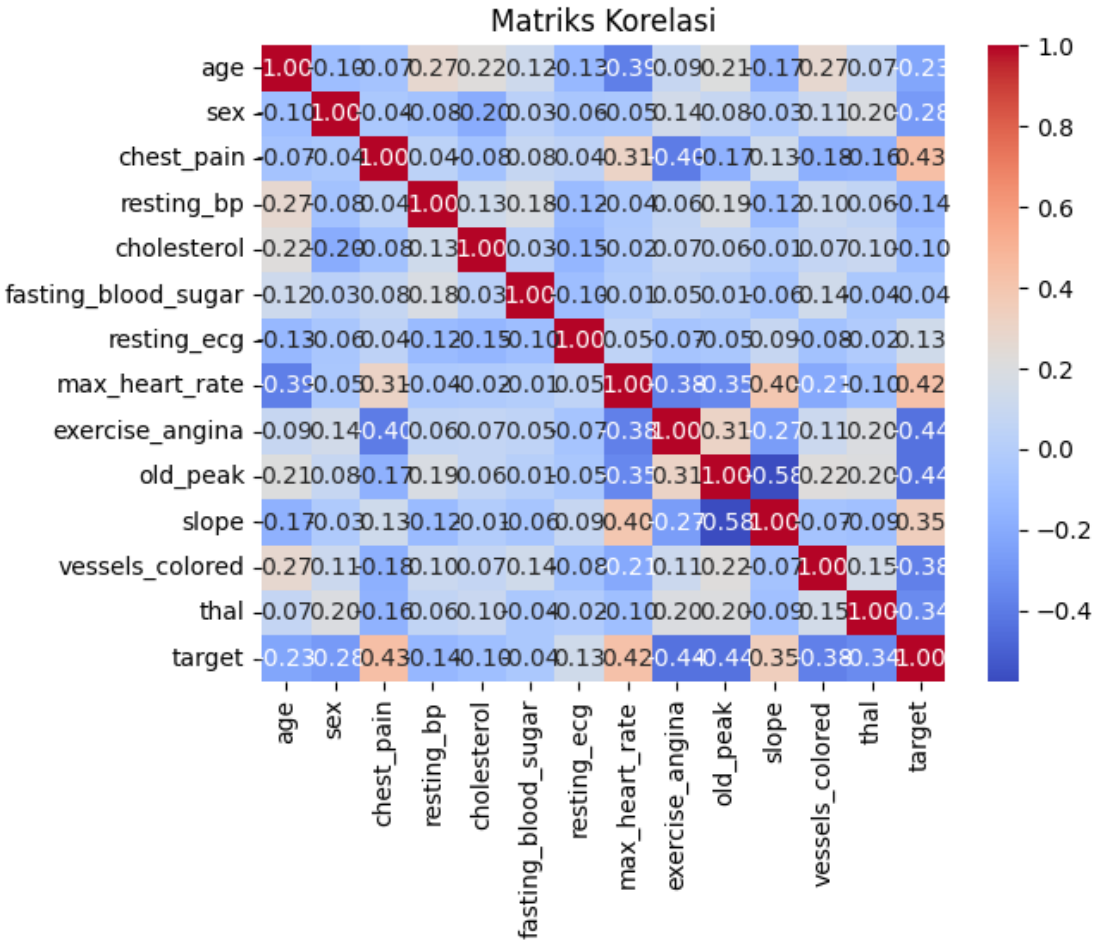
```
[44] # Menampilkan summary
     data.summary().show()
```

```
+-------+------------------+-------------------+------------------+------------------+------------------+--------------------+-------------------+------------------+-------------------+-------------------+
|summary|               age|                sex|        chest_pain|         resting_bp|       cholesterol|fasting_blood_sugar|         resting_ecg|   max_heart_rate|    exercise_angina|           old_peak|
+-------+------------------+-------------------+------------------+------------------+------------------+--------------------+-------------------+------------------+-------------------+-------------------+
|  count|              1025|               1025|              1025|              1025|              1025|                1025|               1025|              1025|               1025|               1025|
|   mean|54.434146341463416|0.6956097560975609|0.9424390243902439|131.61170731707318|           246.0|  0.1492682926829268|0.5297560975609756|149.11414634146342|0.33658536585365856|1.0715121951219524|1.38536
| stddev| 9.07229023324428|0.4603733241196495| 1.029640743645865|17.516718005376408|51.592510206818203|  0.3565266897271575|0.5278775668748918| 23.00572374597721| 0.4727723760037115|1.1750532551501767|0.61775
|    min|                29|                0.0|                 0|                94|               126|                   0|                  0|                71|                  0|                0.0|
|    25%|                48|                0.0|                 0|               120|               211|                   0|                  0|               132|                  0|                0.0|
|    50%|                56|                1.0|                 1|               130|               240|                   0|                  1|               152|                  0|                0.8|
|    75%|                61|                1.0|                 2|               140|               275|                   0|                  1|               166|                  1|                1.8|
|    max|                77|                1.0|                 3|               200|               564|                   1|                  2|               202|                  1|                6.2|
+-------+------------------+-------------------+------------------+------------------+------------------+--------------------+-------------------+------------------+-------------------+-------------------+
```

Menampilkan hasil summary dari dataset heart

6. Menampilkan matriks korelasi

```
[45] # Menampilkan matriks korelasi
     correlation_matrix = data.select([col(c).cast("float") for c in data.columns]).toPandas().corr()
     sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f")
     plt.title("Matriks Korelasi")
     plt.show()
```



Hasil Korelasi dari dataset Heart, dengan warna dan itensitas warna pada Heatmap mencerminkan tingkat korelasi antara 2 Variabel semakin panas/merah maka semakin tinggi nilai kolerasinya.

D. Alasan pemilihan kolom/fitur

1. Sex (Jenis Kelamin):
   - Relevansi Medis: Perbedaan biologis antara jenis kelamin dapat berpengaruh pada risiko penyakit jantung. Beberapa penelitian medis menunjukkan bahwa faktor hormonal dan perbedaan biologis antara pria dan wanita dapat memengaruhi gejala, diagnosis, dan penanganan penyakit jantung.
   - Faktor Risiko Berbeda: Faktor-faktor risiko penyakit jantung dapat bervariasi antara pria dan wanita. Pemilihan jenis kelamin sebagai fitur dapat membantu model memahami dan memperhitungkan perbedaan ini.

2. Target (Hasil):
   - Tujuan Utama: Dalam konteks klasifikasi penyakit jantung, kolom target biasanya menyatakan apakah seseorang memiliki atau tidak memiliki penyakit jantung. Ini adalah variabel yang ingin diprediksi oleh model.
   - Evaluasi Kinerja Model: Dengan menggunakan kolom target, kita dapat mengevaluasi kinerja model dalam memprediksi apakah seseorang berisiko terkena penyakit jantung. Hal ini memungkinkan kita untuk mengukur sejauh mana model mampu.

2. Pengembangan model machine learning

a. 4 Model Machine Learning

1. LogisticRegression
2. RandomForestClassifier
3. GBTClassifier
4. DecisionTreeClassifier

Kami memilih metriks (Akurasi) untuk membandingkan hasil 4 Machine Learning.

b. Memilih Model Terbaik

```
[65] # a. Membuat list model
     models = [
         LogisticRegression(labelCol="target", featuresCol="features"),
         RandomForestClassifier(labelCol="target", featuresCol="features"),
         GBTClassifier(labelCol="target", featuresCol="features"),
         DecisionTreeClassifier(labelCol="target", featuresCol="features")
     ]

     # Melatih dan mengevaluasi model
     evaluator_acc = MulticlassClassificationEvaluator(labelCol="target", metricName="accuracy")

     results = {}

     for model in models:
         model_name = model.__class__.__name__
         print(f"\nTraining {model_name}")
         # Latih model
         model_fit = model.fit(train_data)

         # Membuat prediksi
         predictions = model_fit.transform(test_data)

         # Evaluasi model
         acc = evaluator_acc.evaluate(predictions)

         results[model_name] = {
             "Accuracy": acc,
         }

         print(f"{model_name} - Accuracy: {acc}")
```

```
Training LogisticRegression
LogisticRegression - Accuracy: 0.834319526627219

Training RandomForestClassifier
RandomForestClassifier - Accuracy: 0.8994082840236687

Training GBTClassifier
GBTClassifier - Accuracy: 1.0

Training DecisionTreeClassifier
DecisionTreeClassifier - Accuracy: 0.8816568047337278
```

Dengan hasil tersebut kami memilih 2 performa terbaik yaitu;

- RandomForestClassifier - Accuracy: 0.8994082840236687
- GBTClassifier dengan Accuracy: 1.0

Lakukan Hyperparameter Tuning untuk melihat perubahan :

```
[66] # b. Buat instance dari dua model terbaik
     gbt = GBTClassifier(labelCol="target", featuresCol="features")
     rf = RandomForestClassifier(labelCol="target", featuresCol="features")


     # Penyetelan hyperparameter untuk GBTClassifier
     gbt_paramGrid = ParamGridBuilder() \
         .addGrid(gbt.maxDepth, [5, 10]) \
         .addGrid(gbt.maxIter, [10, 20]) \
         .build()

     gbt_cv = CrossValidator(estimator=gbt, estimatorParamMaps=gbt_paramGrid, evaluator=evaluator_acc, numFolds=3)

     gbt_cvModel = gbt_cv.fit(train_data)

     gbt_bestModel = gbt_cvModel.bestModel

     gbt_best_predictions = gbt_bestModel.transform(test_data)
     gbt_best_acc = evaluator_acc.evaluate(gbt_best_predictions)
     print("GBTClassifier - Best Accuracy (after tuning):", gbt_best_acc)
```

```
     # Penyetelan hyperparameter untuk RandomForestClassifier
     rf_paramGrid = ParamGridBuilder() \
         .addGrid(rf.numTrees, [50, 100]) \
         .addGrid(rf.maxDepth, [5, 10]) \
         .build()

     rf_cv = CrossValidator(estimator=rf, estimatorParamMaps=rf_paramGrid, evaluator=evaluator_acc, numFolds=3)

     rf_cvModel = rf_cv.fit(train_data)

     rf_bestModel = rf_cvModel.bestModel

     rf_best_predictions = rf_bestModel.transform(test_data)
     rf_best_acc = evaluator_acc.evaluate(rf_best_predictions)
     print("RandomForestClassifier - Best Accuracy (after tuning):", rf_best_acc)
```

```
     GBTClassifier - Best Accuracy (after tuning): 1.0
     RandomForestClassifier - Best Accuracy (after tuning): 1.0
```

Hasil perubahan saat dilakukan Hyperparameter Tuning dari 2 model Machine Learning;

- GBTClassifier - Best Accuracy (after tuning): 1.0
- RandomForestClassifier - Best Accuracy (after tuning): 1.0

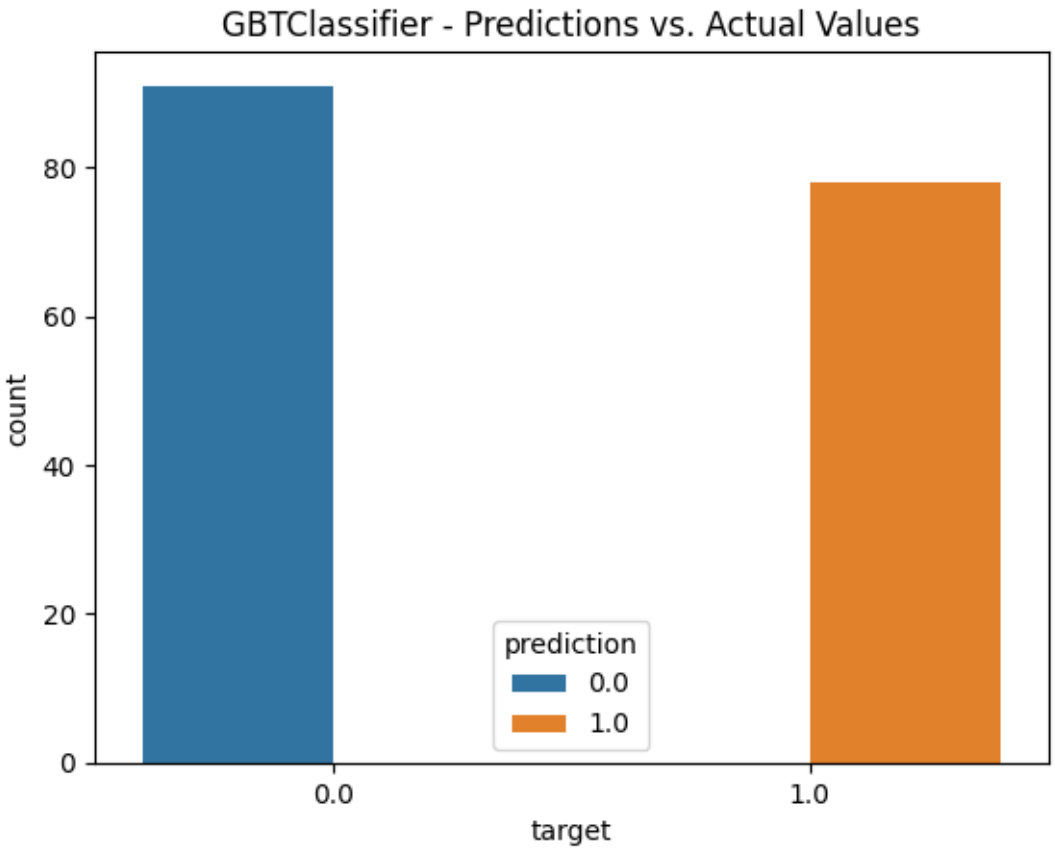
Dari 2 model Machine Learning, kami memilih 1 model Machine Learning GBTClassifier, karena model ini cenderung lebih mudah diinterpretasikan dan dapat memberikan prediksi yang sangat akurat pada hasil data sebelumnya dan hasil data diatas. Jadi, GBTClassifier bisa menjadi solusi pada masalah awal dibandingkan dengan RandomForestClassifier.

c. Menjabarkan Karakteristik Model

1. Overfitting
   Akurasi 1.0 dapat menunjukkan tanda overfitting, yaitu model telah "menghafal" data pelatihan dengan sangat baik
2. Tuning Parameter
   Pengaturan parameter yang tepat pada dataset heart dapat membantu model mencapai keseimbangan yang baik antara underfitting dan overfitting
3. High Correlation between Features
   Ada hubungan yang kompleks atau interaksi antar fitur dalam data, GBTClassifier dapat berhasil mengekstrak pola tersebut.

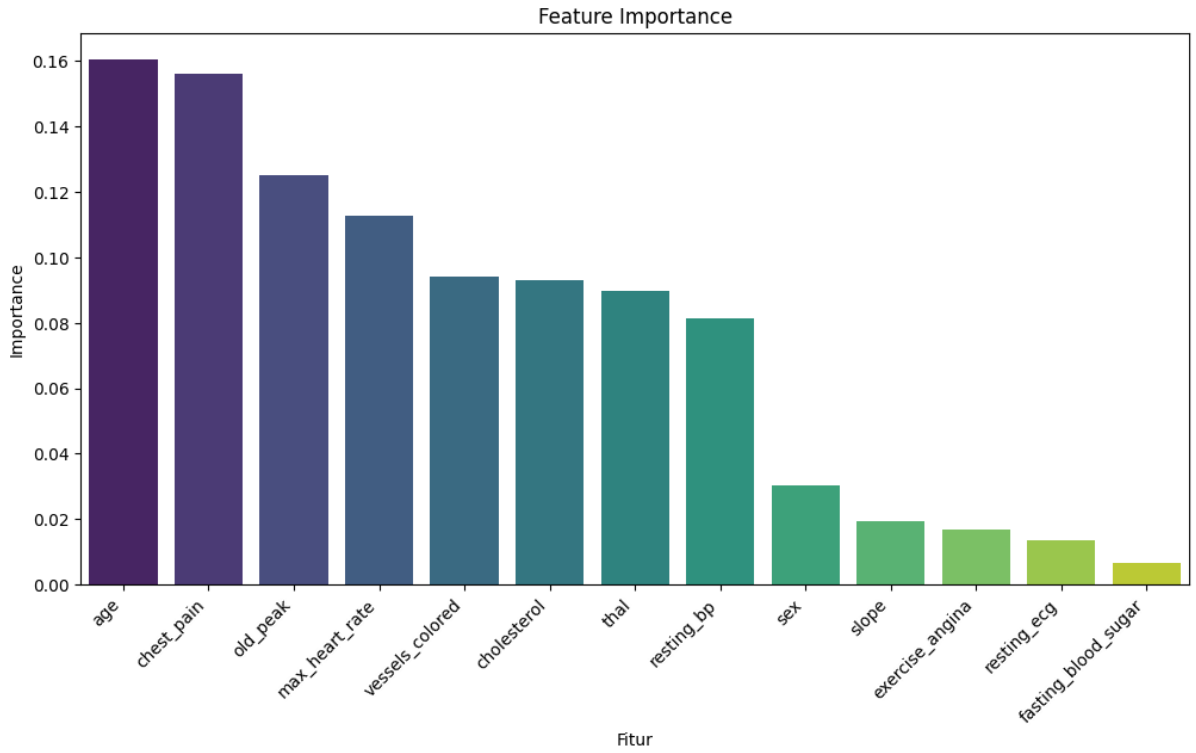
4. Complex or Nonlinear Data
   Menangani hubungan nonlinier dan kompleks antara fitur-fitur dalam data, membuatnya cocok untuk masalah klasifikasi yang melibatkan pola yang kompleks.

3. Hasil



Gambar 1. GBTClassifier - Prediksi vs Nilai Aktual

Garis biru mewakili prediksi, dan garis oranye mewakili nilai aktual. Grafik ini menunjukkan bahwa model klasifikasi GBTClassifier dapat membuat prediksi yang cukup akurat untuk dataset ini. Grafik ini juga menunjukkan bahwa model GBTClassifier membuat prediksi yang lebih akurat untuk nilai aktual 1.0 daripada untuk nilai aktual 0.0. Hal ini kemungkinan disebabkan oleh fakta bahwa ada lebih banyak data pelatihan untuk nilai aktual 1.0 daripada untuk nilai aktual 0.0.



Gambar 2.Grafik Feature Importance

Berikut adalah penjelasan lebih rinci tentang masing-masing fitur pada gambar 2:

- Usia: Semakin tua seseorang, semakin banyak pembuluh darah koronernya yang mengalami penumpukan plak. Penumpukan plak dapat menyebabkan penyempitan atau penyumbatan pembuluh darah koroner, sehingga aliran darah ke jantung menjadi terganggu. Hal ini dapat menyebabkan nyeri dada, serangan jantung, atau bahkan kematian.
- Nyeri dada: Nyeri dada yang disebabkan oleh aktivitas fisik merupakan gejala umum Penyakit Jantung. Nyeri dada tersebut disebabkan oleh kekurangan oksigen ke jantung.
- Tingginya puncak ST: Tingginya puncak ST pada EKG merupakan indikator kerusakan otot jantung yang dapat disebabkan oleh Penyakit Jantung.
- Denyut jantung maksimum: Denyut jantung maksimum yang rendah merupakan faktor risiko Penyakit Jantung. Denyut jantung maksimum yang rendah dapat mengindikasikan adanya kerusakan jantung.
- Jumlah pembuluh koroner yang berwarna: Pembuluh koroner yang berwarna hitam merupakan tanda kerusakan akibat Penyakit Jantung. Pembuluh koroner

yang berwarna hitam menunjukkan bahwa pembuluh darah tersebut telah mengalami penumpukan plak.

- Kolesterol: Kolesterol tinggi merupakan faktor risiko Penyakit Jantung. Kolesterol tinggi dapat menyebabkan penumpukan plak pada pembuluh darah koroner.
- Thal: Thal positif merupakan tanda kerusakan jantung akibat Penyakit Jantung. Thal positif menunjukkan bahwa jantung telah mengalami kerusakan akibat Penyakit Jantung.
- Tekanan darah istirahat: Tekanan darah istirahat tinggi merupakan faktor risiko Penyakit Jantung. Tekanan darah istirahat tinggi dapat merusak pembuluh darah, termasuk pembuluh darah koroner.
- Jenis kelamin: Pria memiliki risiko Penyakit Jantung yang lebih tinggi daripada wanita. Hal ini disebabkan oleh adanya perbedaan hormon pada pria dan wanita.
- Lereng: Lereng positif pada EKG merupakan tanda kerusakan jantung akibat Penyakit Jantung. Lereng positif pada EKG menunjukkan bahwa jantung telah mengalami kerusakan akibat Penyakit Jantung.
- Angina saat berolahraga: Angina saat berolahraga merupakan gejala umum Penyakit Jantung. Angina saat berolahraga disebabkan oleh kekurangan oksigen ke jantung.
- EKG istirahat: EKG istirahat yang abnormal merupakan tanda kerusakan jantung akibat Penyakit Jantung. EKG istirahat yang abnormal dapat menunjukkan adanya kerusakan jantung akibat Penyakit Jantung.
- Gula darah puasa: Gula darah puasa tinggi merupakan faktor risiko Penyakit Jantung. Gula darah puasa tinggi dapat merusak pembuluh darah, termasuk pembuluh darah koroner.

```
# Input untuk Prediksi (termasuk kolom target)
input_data = input_data = [29.0,1.0,1.0,130.0,204.0,0.0,0.0,202.0,0.0,0.0,2.0,0.0,2.0,0.0]

# Transformasi input data menggunakan assembler
input_features = assembler.transform(spark.createDataFrame([input_data], schema=correct_schema)).select("features")

# Prediksi dengan Model Terbaik
prediction_result = gbt_bestModel.transform(input_features).select("prediction").collect()[0][0]

# Menampilkan Hasil Prediksi
print("\nHasil Prediksi untuk Input Data:")
print(f"Prediction: {prediction_result}")
```

```
Hasil Prediksi untuk Input Data:
Prediction: 1.0
```

```
# Prediksi menggunakan model GBTClassifier terbaik
gbt_best_predictions.select("sex", "features", "target", "prediction").show(truncate=False)
```

```
+---+-----------------------------------------------------------+------+----------+
|sex|features                                                   |target|prediction|
+---+-----------------------------------------------------------+------+----------+
|1.0|[29.0,1.0,1.0,130.0,204.0,0.0,0.0,202.0,0.0,0.0,2.0,0.0,2.0]|1.0   |1.0       |
|0.0|[34.0,0.0,1.0,118.0,210.0,0.0,1.0,192.0,0.0,0.0,0.7,2.0,0.0,2.0]|1.0   |1.0       |
|1.0|[34.0,1.0,3.0,118.0,182.0,0.0,0.0,174.0,0.0,0.0,0.0,2.0,0.0,2.0]|1.0   |1.0       |
|0.0|[35.0,0.0,0.0,138.0,183.0,0.0,1.0,182.0,0.0,0.0,1.4,2.0,0.0,2.0]|1.0   |1.0       |
|1.0|[35.0,1.0,0.0,126.0,282.0,0.0,0.0,156.0,1.0,0.0,0.2,2.0,0.0,3.0]|0.0   |0.0       |
|1.0|[35.0,1.0,1.0,122.0,192.0,0.0,1.0,174.0,0.0,0.0,0.0,2.0,0.0,2.0]|1.0   |1.0       |
|1.0|[37.0,1.0,2.0,130.0,250.0,0.0,1.0,187.0,0.0,0.0,3.5,0.0,0.0,2.0]|1.0   |1.0       |
|1.0|[38.0,1.0,2.0,138.0,175.0,0.0,1.0,173.0,0.0,0.0,0.0,2.0,4.0,2.0]|1.0   |1.0       |
|0.0|[39.0,0.0,2.0,94.0,199.0,0.0,1.0,179.0,0.0,0.0,0.0,2.0,0.0,2.0] |1.0   |1.0       |
|0.0|[39.0,0.0,2.0,138.0,220.0,0.0,1.0,152.0,0.0,0.0,0.0,1.0,0.0,2.0]|1.0   |1.0       |
|0.0|[39.0,0.0,2.0,138.0,220.0,0.0,1.0,152.0,0.0,0.0,0.0,1.0,0.0,2.0]|1.0   |1.0       |
|0.0|[39.0,0.0,2.0,138.0,220.0,0.0,1.0,152.0,0.0,0.0,0.0,1.0,0.0,2.0]|1.0   |1.0       |
|1.0|[39.0,1.0,0.0,118.0,219.0,0.0,1.0,140.0,0.0,0.0,1.2,1.0,0.0,3.0]|0.0   |0.0       |
|1.0|[39.0,1.0,2.0,140.0,321.0,0.0,0.0,182.0,0.0,0.0,0.0,2.0,0.0,2.0]|1.0   |1.0       |
|1.0|[40.0,1.0,0.0,152.0,223.0,0.0,1.0,181.0,0.0,0.0,0.0,2.0,0.0,3.0]|0.0   |0.0       |
|0.0|[41.0,0.0,1.0,105.0,198.0,0.0,1.0,168.0,0.0,0.0,0.0,2.0,1.0,2.0]|1.0   |1.0       |
|0.0|[41.0,0.0,2.0,112.0,268.0,0.0,0.0,172.0,1.0,0.0,0.2,2.0,0.0,2.0]|1.0   |1.0       |
|0.0|(13,[0,3,4,7,9,10,12],[42.0,102.0,265.0,122.0,0.6,1.0,2.0]) |1.0   |1.0       |
|1.0|[42.0,1.0,0.0,140.0,226.0,0.0,1.0,178.0,0.0,0.0,0.0,2.0,0.0,2.0]|1.0   |1.0       |
|1.0|[42.0,1.0,2.0,120.0,240.0,1.0,1.0,194.0,0.0,0.0,0.8,0.0,0.0,3.0]|1.0   |1.0       |
+---+-----------------------------------------------------------+------+----------+
only showing top 20 rows
```

Gambar 3. Hasil Input dan Prediksi

Code tersebut digunakan untuk memprediksi penyakit jantung menggunakan dataset jantung. Code tersebut terdiri dari beberapa bagian, yaitu:

- **Input untuk Prediksi**

  Bagian ini berisi data input yang akan diprediksi. Data input tersebut terdiri dari 13 fitur, termasuk kolom target.

- **Transformasi Input Data**

Bagian ini digunakan untuk mengubah data input ke format yang dapat diterima oleh model. Dalam hal ini, data input diubah ke format dataframe Spark.

- **Prediksi dengan Model Terbaik**

Bagian ini digunakan untuk melakukan prediksi menggunakan model terbaik yang telah dilatih sebelumnya. Model terbaik tersebut adalah model Gradient Boosted Trees (GBT).

- **Menampilkan Hasil Prediksi**

Bagian ini digunakan untuk menampilkan hasil prediksi.

Berikut adalah penjelasan lebih rinci dari setiap bagian code tersebut:

Input untuk Prediksi

Data input untuk prediksi dalam kasus ini adalah sebagai berikut:
input_data = [29.0,1.0,1.0,130.0,204.0,0.0,0.0,202.0,0.0,0.0,2.0,0.0,2.0,0.0]

Data input tersebut terdiri dari 13 fitur, yaitu:

- sex: Jenis kelamin
- age: Usia
- cp: Jenis nyeri dada
- trestbps: Tekanan darah istirahat
- chol: Kolesterol total
- fbs: Gula darah puasa
- restecg: Hasil elektrokardiogram istirahat
- thalach: Detak jantung tertinggi
- exang: Angina saat latihan
- oldpeak: Detak jantung saat puncak latihan
- slope: Kemiringan segmen ST pada latihan
- ca: Jumlah pembuluh darah koroner yang tersumbat
- thal: Jenis tes thalium

Fitur target adalah "target", yang menunjukkan apakah pasien menderita penyakit jantung atau tidak. Dalam kasus ini, nilai target adalah 1, yang menunjukkan bahwa pasien menderita penyakit jantung.

Transformasi Input Data

Data input diubah ke format dataframe Spark menggunakan code berikut:

input_features = assembler.transform(spark.createDataFrame([input_data], schema=correct_schema)).select("features")

Kode tersebut menggunakan kelas assembler untuk menggabungkan semua fitur input menjadi satu kolom. Kolom tersebut kemudian diberi nama "features".

Prediksi dengan Model Terbaik

Model terbaik untuk prediksi penyakit jantung dalam kasus ini adalah model GBT. Model tersebut dilatih menggunakan dataset jantung.

Prediksi dilakukan menggunakan code berikut:

```
prediction_result =
gbt_bestModel.transform(input_features).select("prediction").collect()[0][0]
```

Kode tersebut menggunakan model GBT untuk memprediksi penyakit jantung berdasarkan data input yang telah ditransformasi. Hasil prediksi disimpan dalam variabel prediction_result.

Menampilkan Hasil Prediksi

Hasil prediksi ditampilkan menggunakan code berikut:

```
print("\nHasil Prediksi untuk Input Data:")
print(f"Prediction: {prediction_result}")
```

Kode tersebut menampilkan hasil prediksi dalam format berikut:

```
Hasil Prediksi untuk Input Data:
Prediction: 1.0
```

Dalam kasus ini, hasil prediksi adalah 1.0, yang menunjukkan bahwa input data tersebut memiliki kemungkinan 100% untuk menderita penyakit jantung.

4. **Laporan**

Referensi Sumber

[1] S. Rahayu, J. J.Purnama, A. B. Pohan, F. S. Nugraha, S. Nurdiani, S. Hadianti, PREDICTION OF SURVIVAL OF HEART FAILURE PATIENTS USING RANDOM FOREST, Jurnal PILAR Nusa Mandiri Vol.16, no2 September 2020

[2] S.Kacung, E. Prihartono, Sistem Cerdas untuk Mendeteksi Dini Penyakit Jantung Dengan Decision Tree, Jurnal INFORM Vol.1 No.2, Juli 2016

[3] H.Rianto, R.S.Wahono, Resampling Logistic Regression untuk Penanganan Ketidakseimbangan Class pada Prediksi Cacat Software, Journal of Software Engineering, Vol. 1, No. 1, April 2015

[4] R. Bhuvaneeswari, P. Sudhakar, G. Prabakaran, Heart Disease Prediction Model based Ongradient Boosting Tree (GBT) Classification Algorithm, International Journal of Recent Technology and Engineering (IJRTE), Volume-8, Issue-2S11, September 2019

[5] T.A.E.Putri, T.Widiharih , R.Santoso, PENERAPAN TUNING HYPERPARAMETER RANDOMSEARCHCV PADA ADAPTIVE BOOSTING UNTUK PREDIKSI KELANGSUNGAN HIDUP PASIEN GAGAL JANTUNG, JURNAL GAUSSIAN, Volume 11, Nomor 3, Tahun 2022

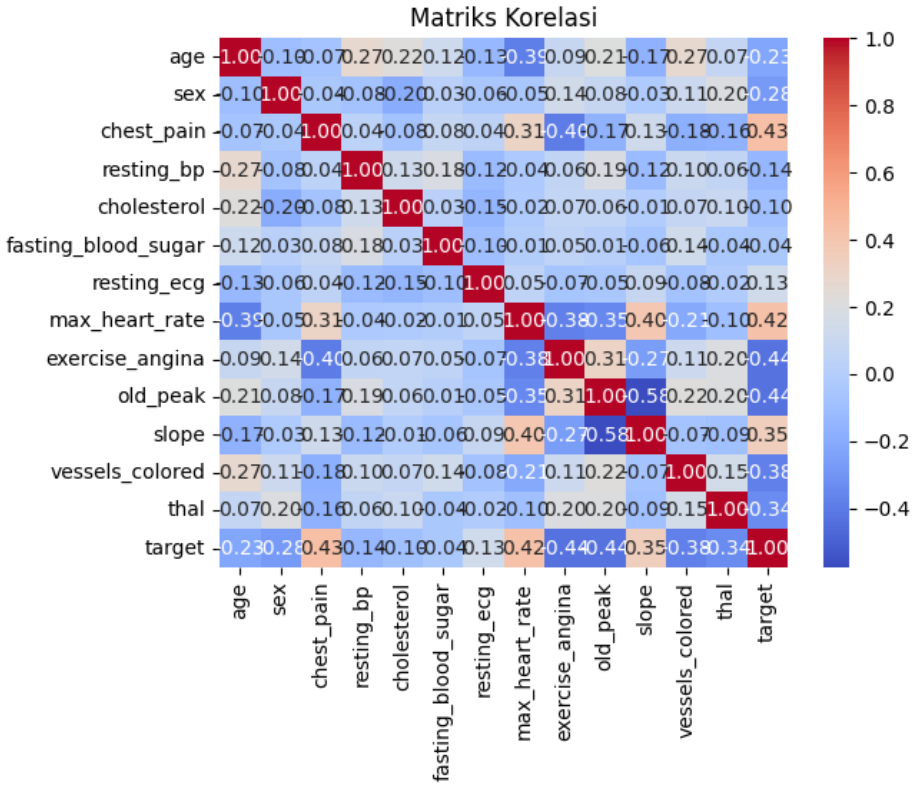
Code

https://colab.research.google.com/drive/1OXFWxDw5HHIi7G0W9VoPyPjdHQ0W_68N?usp=sharing

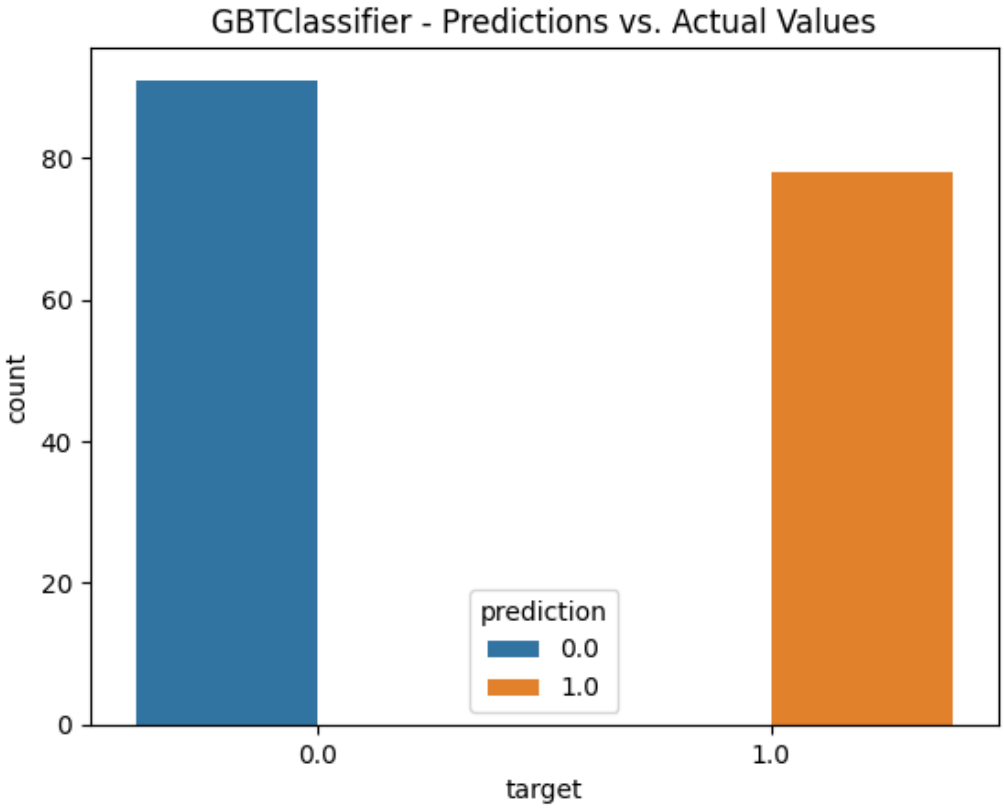
Link Dataset

https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset?resource=download

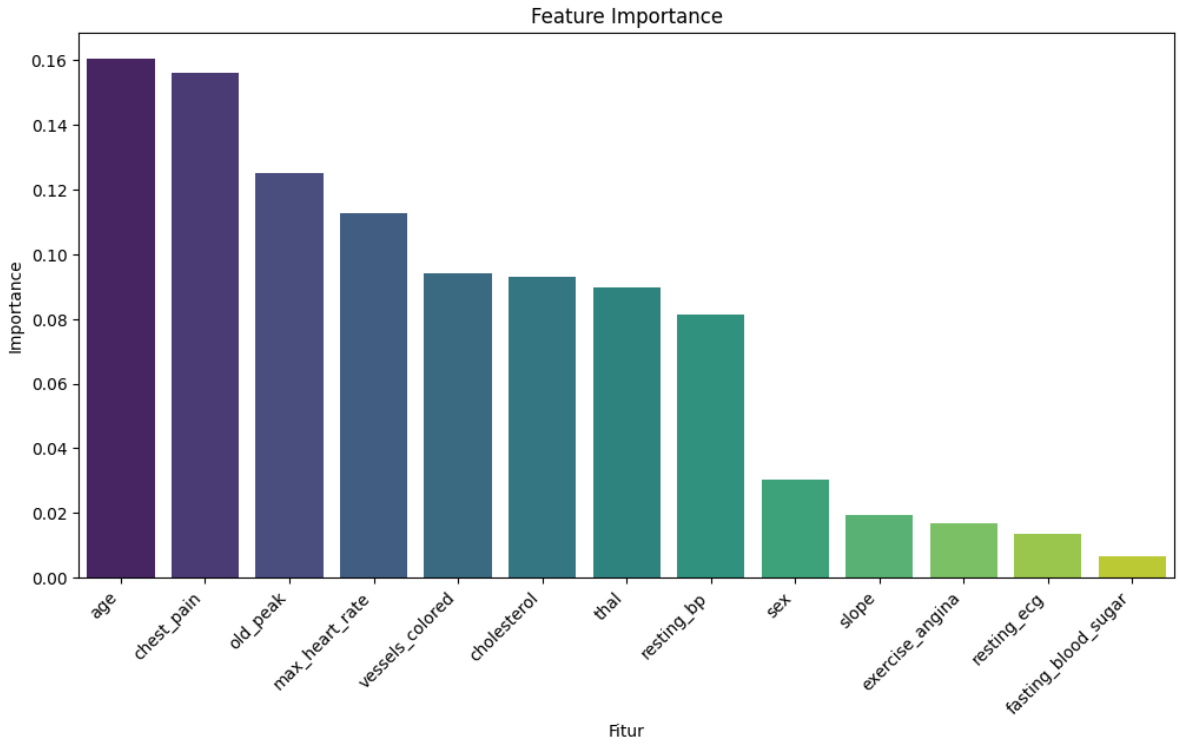
Gambar/Grafik



Gambar 1. Matriks Korelasi



Gambar 2. GBTClassifier - Prediksi vs Nilai Aktual



Gambar 3.Grafik Feature Importance