BAB 3 ANALISIS DAN PERANCANGAN SISTEM

Bab ini memaparkan analisis masalah yang diatasi beserta pendekatan dan alur kerja dari perangkat lunak yang dikembangkan, mengimplementasikan metode yang digunakan dan hasil yang akan ditampilkan.

3.1 Analisis Masalah

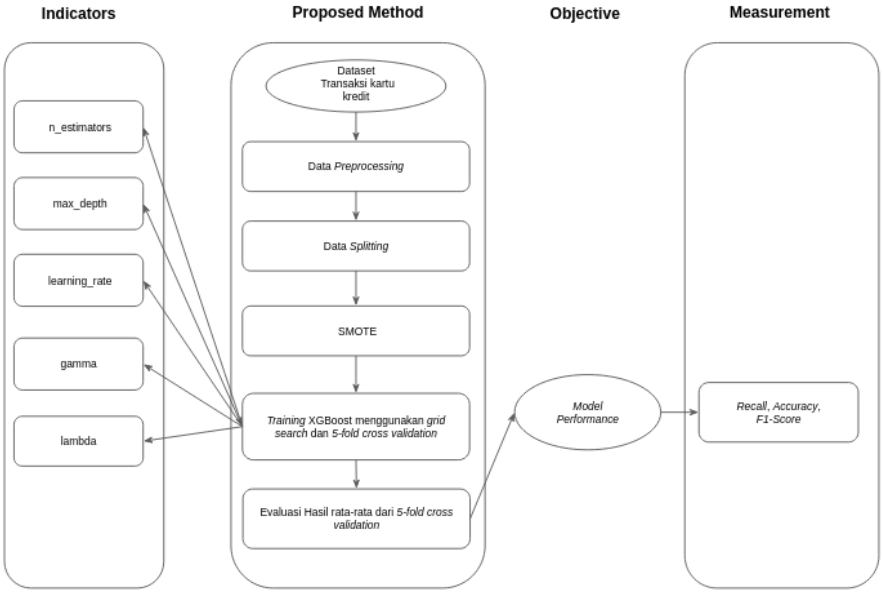
Pada Bab 1 telah dijelaskan masalah yang dihadapi terkait deteksi transaksi kartu kredit. Kemudian dijelaskan juga bahwa metode yang akurat dalam mendeteksi penipuan kartu kredit dibutuhkan untuk mencegah terjadinya kerugian yang besar akibat penipuan tersebut. Pada penelitian ini, metode klasifikasi *machine learning* XGBoost digunakan karena memiliki hasil *recall* tertinggi pada pengujian [9]. *Hyperparameter tuning* akan digunakan untuk membantu memaksimalkan akurasi XGBoost dengan memilih nilai *hyperparameter* dengan hasil evaluasi terbaik. Metode *grid search* digunakan dalam penelitian karena implementasinya yang paling mudah dibandingkan dengan dua metode lainnya, serta karena keterbatasan sumber daya yang digunakan. Agar penelitian ini lebih akurat dalam menguji keseluruhan dataset, maka akan digunakan *k-fold cross-validation* dengan jumlah fold sebanyak 5.

Pada penelitian yang dilakukan oleh penulis bertujuan untuk mencari nilai *hyperparameter* terbaik XGBoost untuk mendeteksi transaksi penipuan kartu kredit. *Hyperparameter* terbaik dengan hasil terbaik akan dievaluasi kembali pada pengujian selanjutnya dengan menggunakan teknik *oversampling* SMOTE.

Masukkan pada penelitian ini adalah dataset transaksi kartu kredit. Keluaran atau hasil dari sistem deteksi penipuan kartu kredit yang dirancang adalah berupa nilai *hyperparameter* terbaik, akurasi, *precision*, *recall*, *F-measure* pada data latih sebagai data *hold out* pada proses evaluasi terakhir untuk membuktikan seberapa baik kinerja XGBoost dalam mendeteksi transaksi penipuan kartu kredit.

3.2 Kerangka Pemikiran

Berikut ini adalah kerangka pemikiran dari metode yang diusulkan untuk melakukan deteksi transaksi penipuan kartu kredit.



Gambar 3.1 Kerangka Pemikiran

Pada gambar 3.1 adalah kerangka pemikiran yang telah disusun dalam bentuk diagram:

1. Data *Preprocessing*: Menghilangkan *missing values* dan data yang duplikat dan memilih fitur yang akan digunakan.
2. Data *Splitting*: Membagi data menjadi data latih dan data uji.
3. SMOTE: Menerapkan metode *SMOTE* untuk meningkatkan proporsi jumlah data transaksi *fraud*.
4. *Training*: Melatih model XGBoost menggunakan *grid search* dan *5-fold cross validation* menggunakan data latih.
5. Evaluasi: Mengambil rata-rata hasil pengukuran dari proses *cross validation* pada setiap fold.
6. *Indicators*: Merupakan *hyperparameter* pada model XGBoost. *Hyperparameter* yang akan diuji adalah *n_estimators*, *learning_rate* (η), gamma (γ), dan lambda (λ).

3.2.1 Penjelasan Indikator

Pada bagian ini akan dijelaskan mengenai indikator *hyperparameter* yang digunakan pada XGBoost. Berikut ini merupakan penjelasan indikator yang akan digunakan pada XGBoost:

1. *n_estimators*: Merupakan banyaknya *tree* yang akan dibentuk oleh XGBoost.

2. *max_depth*: Merupakan kedalaman maksimum *tree* yang dapat dibentuk oleh XGBoost.

3. *Learning rate* (η): Merupakan besaran nilai yang harus diambil untuk meminimalkan *loss function*. Pada XGBoost, Nilai ini akan digunakan sebagai nilai *scaling* pada nilai *output* setiap *tree*. *Learning rate* yang terlalu besar mengakibatkan model akan *overshoot*, jika nilainya terlalu kecil maka proses pelatihan model akan semakin lama.

4. *gamma* (γ): Merupakan konstanta pengurang pada perhitungan yang mengurangi nilai *gain*. Apabila nilai gain < 0 maka *branch* tidak akan dikembangkan.

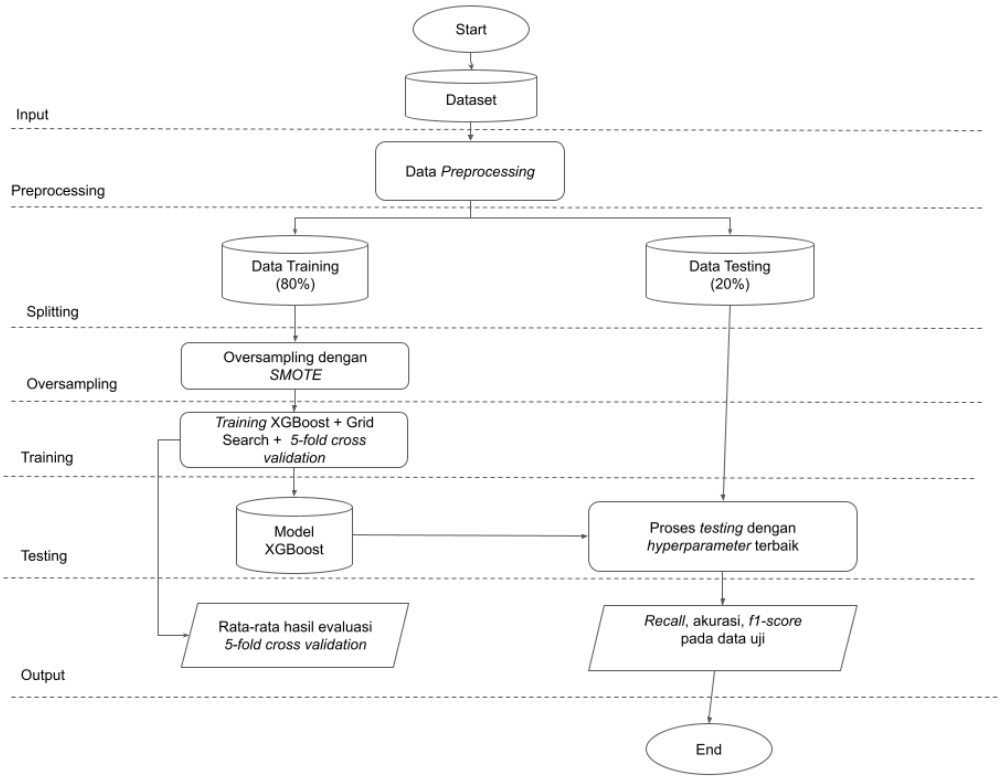
5. *lambda* (λ): Merupakan konstanta regularisasi pada XGBoost.

Berdasarkan *hyperparameter* yang diuji, terdapat beberapa *hyperparameter* yang tidak masuk ke dalam pengujian ini, diantaranya adalah *subsample*, *colsample*, *min_child_weight*, dan *base_score*. *subsample* dan *colsample* berperan dalam seleksi data latih, sedangkan dalam pengujian ini semua data latih akan digunakan dalam proses pelatihan model. *Hyperparameter min_child_weight* tidak masuk ke dalam pengujian dikarenakan *hyperparameter* ini juga memiliki fungsi yang sama seperti *gamma* yaitu mencegah pembangunan *tree* terlalu dalam. *Min_child_weight* ini akan membandingkan nilai *hessian* dan apabila nilainya lebih rendah maka akan berhenti membangun *tree* lebih dalam. *Hyperparameter base_score* tidak masuk ke dalam pengujian karena *base_score* merupakan nilai awal prediksi dan nilai tersebut tidak terlalu berpengaruh dan terus diperbaharui seiring banyaknya jumlah *tree* yang dibangun [19]. Selain itu jumlah kombinasi hyperparameter yang secara eksponensial akan bertambah banyak karena pencarian *hyperparameter* menggunakan metode *grid search*, beban komputasi yang berat serta waktu pelatihan yang lama juga menjadi pertimbangan.

3.3 Analisis Urutan Proses Global

Dalam sistem deteksi transaksi penipuan kartu kredit terdapat beberapa proses, yang berisi data *preprocessing*, data *splitting*, penerapan teknik *oversampling* menggunakan SMOTE pada data latih, proses *training*

XGBoost menggunakan *5-fold cross validation* dan *grid search*, evaluasi rata-rata hasil metrik *5-fold cross-validation*), memilih *hyperparameter* terbaik untuk selanjutnya dilakukan proses pengujian pada data uji, dan evaluasi menggunakan *confusion matrix*.



Gambar 3.2 *Flowchart* Proses Global

Berikut adalah uraian proses global yang dilakukan dalam penelitian ini yang terlihat pada gambar 3.2:

1. Masukan berupa file CSV berisi data transaksi kartu kredit [10].
2. *File* CSV akan disimpan kedalam bentuk DataFrame, yang kemudian akan dilakukan *data preprocessing* untuk standarisasi data.
3. Tahap data *preprocessing* menghilangkan *missing value* pada data dan menghilangkan data yang duplikat.
4. Data *splitting* membagi data menjadi data latih dan data uji. Data uji akan digunakan sebagai data *hold out* yang berarti data uji akan digunakan sebagai evaluasi terakhir dan tidak akan digunakan dalam proses training dan *hyperparameter* tuning seperti pada gambar 2.12.
5. Menerapkan teknik *oversampling* SMOTE pada data latih untuk

memperbanyak data dengan label *fraud* (*'Class'* = 1).

6. Membuat kombinasi *hyperparameter* yang akan diuji menggunakan metode *grid search*.

7. Proses *training* pada XGBoost akan menggunakan *5-fold cross validation* dengan menggunakan kombinasi *hyperparameter* hasil dari proses sebelumnya. Berikut merupakan proses *cross-validation*:

   (a) Membuat model XGBoost menggunakan *hyperparameter* dari kombinasi *hyperparameter* yang telah dibuat sebelumnya.

   (b) Memisahkan data latih menjadi data latih untuk 4 fold lainnya dan data uji untuk proses cross-validation. Data uji disini berbeda dengan data uji yang bersifat sebagai *hold out*. Data uji akan menjadi 1 *fold* dan 4 *fold* lainnya merupakan data latih. Setiap iterasi *cross-validation*, fold yang digunakan pada data uji harus berbeda sehingga semua fold pernah menjadi data uji.

   (c) Melakukan proses pelatihan menggunakan data latih dan *hyperparameter* yang sudah di *tuning*.

   (d) Melakukan testing menggunakan data uji menggunakan model yang didapat dari proses sebelumnya dari proses sebelumnya menggunakan .

   (e) Mengukur performa model menggunakan metrik *recall, accuracy, f1-score*.

8. Mencatat hasil evaluasi dan menghitung rata-rata setiap metrik pengukuran dari proses *5-fold cross validation*.

9. Memilih *hyperparameter* terbaik berdasarkan hasil rata-rata evaluasi *5-fold cross-validation*.

10. Membentuk model XGboost dengan *hyperparameter* terbaik berdasarkan rata-rata evaluasi *5-fold cross-validation* untuk dilakukan proses pengujian menggunakan data uji yang telah ditetapkan sebagai data *hold out* untuk evaluasi akhir, dengan hasil pengukuran menggunakan *confusion matrix* untuk mengetahui *precision, recall, accuracy, dan f1-score*.

11. Keluaran berupa XGBoost dengan *hyperparameter* yang memiliki

rata-rata evaluasi *5-fold cross-validation* terbaik dan waktu pemrosesan terbaik serta hasil pengujian berupa *confusion matrix* pada data uji sebagai data *hold out*.

3.4 Analisis Data Sampling

Dalam penelitian ini, dataset berupa 1 file dengan format *Comma Separated Value* (CSV). Fitur yang digunakan dalam penelitian adalah fitur Time, V1-V28, *Amount*, dan *Class* karena seluruh fitur tersebut memuat data penting mengenai transaksi yang terjadi. Data yang akan dipakai dari *dataset* sejumlah 47773 dengan 47300 diantaranya merupakan transaksi *non-fraud* dan 473 diantaranya merupakan transaksi *fraud*, sehingga dataset ini sangat *imbalance* dengan rasio sekitar 1% transaksi penipuan dari dataset yang akan dipakai.

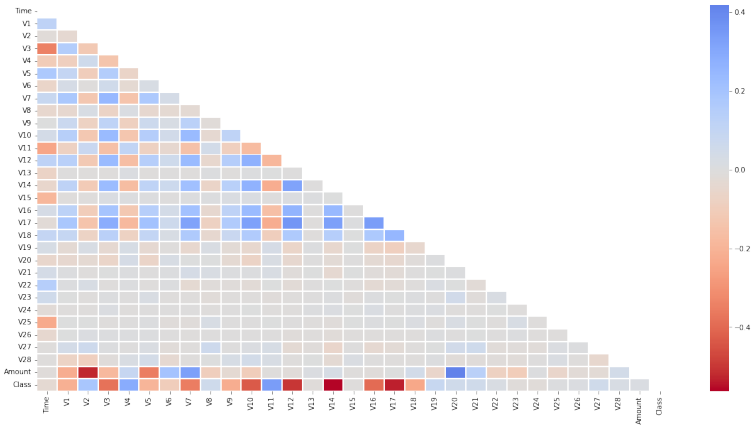
Detail Compact Column 10 of 31 columns

# Time	# V1	# V2	# V3	# V4	# V5	# V6
0	-1.3598071336738	-0.07278117330984497	2.536346737969144	1.3781552242744 3	-0.338320769942518	0.46238777792
0	1.1918571113148 6	0.2661507120596 3	0.1664801133532 1	0.4481540784609 11	0.0600176492822243	-0.0823608 5687
1	-1.3583540615982 3	-1.340163074736 09	1.7732093426311 9	0.3797795930343 28	-0.503198133318193	1.800499388 3
1	-0.9662717115720 87	-0.1852260080828 98	1.7929933395787 2	-0.8632912750364 53	-0.010308879603 0823	1.247203167 6
2	-1.1582330934952 3	0.8777367548484 51	1.548717846511	0.4030339339551 21	-0.407193377311 653	0.095921462 256
2	-0.4259658844124 54	0.9605230448829 85	1.1411093423221 9	-0.1682520797603 02	0.4209868807721 9	-0.0297275 9742
4	1.2296576345079 3	0.1410035070493 26	0.0453707735899 449	1.2026127367359 4	0.1918809885976 45	0.272708122 98
7	-0.6442694423481 46	1.4179635454738 5	1.0743803763556	-0.4921990184950 15	0.9489340947641 57	0.428118462 89
7	-0.8942860822028 2	0.2861571962765 44	-0.1131922127298 71	-0.2715261300886 04	2.6695986595986	3.721818061 1
9	-0.3382617524257 5	1.1195933764156 6	1.0443665515731 6	-0.2221872767382 96	0.4993608064972 7	-0.2467611 91

Gambar 3.3 Dataset transaksi pembayaran kartu kredit

3.5 Data *Preprocessing*

Tahap ini dilakukan untuk membersihkan data dari *missing values* dan *duplicate values* yang dapat merusak model yang akan dibangun. Setelah dilakukan pemeriksaan, tidak ada satupun *missing values* dalam dataset. Tahap selanjutnya adalah menghapus data yang duplikat. Terdapat 1081 duplikat dalam dataset yang perlu dihapus. Pada tahap ini juga dilakukan visualisasi matrix korelasi untuk melihat hubungan antar variabel sesuai gambar 3.4 dan gambar 3.5.



Gambar 3.4 *Pearson Heatmap Correlation Matrix*

Matriks korelasi *pearson* mengukur korelasi linear antar 2 variabel. Gambar 3.4 menunjukkan bahwa terdapat beberapa variabel yang memiliki korelasi yang tinggi baik korelasi positif yang ditandai dengan *heatmap* berwarna biru ataupun korelasi negatif yang ditandai dengan *heatmap* berwarna merah apabila diukur menggunakan matriks korelasi pearson, namun ada banyak variabel yang tidak memiliki korelasi yang ditandai dengan *heatmap* berwarna putih. Berdasarkan gambar 3.4 dapat disimpulkan sebagai berikut:

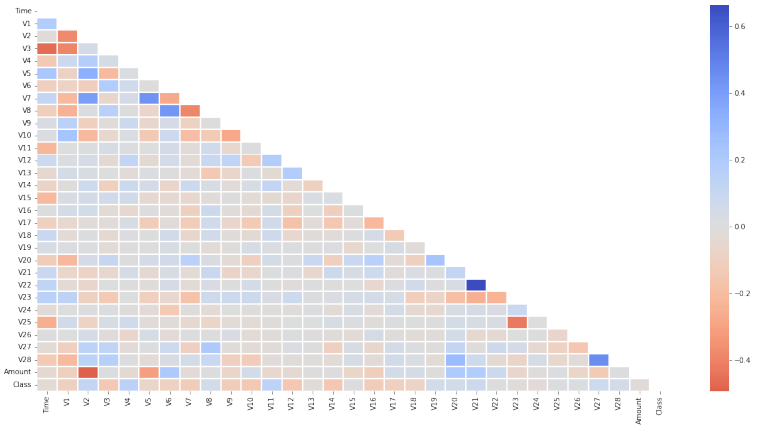
1. Variabel *Amount* memiliki korelasi negatif yang tinggi dengan variabel V2, dan memiliki korelasi positif yang tinggi dengan variabel V7, 20.

2. Variabel V3 memiliki korelasi negatif yang tinggi dengan variabel *Time* dan memiliki korelasi negatif yang cukup dengan variabel *Amount* dan *Class*

3. Variabel *Time* memiliki beberapa korelasi negatif dengan V11 dan V25 dan korelasi negatif yang tinggi dengan V3. Variabel ini juga memiliki sedikit korelasi positif dengan variabel V5, V12, dan V22.

4. Variabel *Class* memiliki korelasi negatif yang cukup tinggi dengan variabel V12, V14, dan V17. Variabel ini juga memiliki korelasi positif dengan variabel V11 meskipun nilainya cukup rendah.

5. Banyak variabel yang tidak memiliki korelasi antar satu dengan yang lainnya yang ditandai dengan *heatmap* berwarna putih. Hal ini dapat terjadi apabila korelasi antar variabel tersebut tidak memiliki hubungan yang linear.



Gambar 3.5 *Spearman Heatmap Correlation Matrix*

Matriks korelasi *spearman* mengukur korelasi non-linear antar 2 variabel. Gambar 3.5 menunjukkan bahwa terdapat beberapa variabel yang memiliki korelasi yang tinggi baik korelasi positif ataupun korelasi negatif apabila diukur menggunakan matriks korelasi *spearman*. Ada beberapa variabel yang tidak memiliki korelasi namun jumlahnya jauh lebih sedikit apabila dibandingkan dengan matriks korelasi *pearson* sesuai gambar 3.4. Berdasarkan gambar 3.5 dapat disimpulkan sebagai berikut.

1. Variabel *Time* memiliki korelasi negatif dengan variabel V11, V15, dan V25 dan korelasi negatif yang cukup tinggi dengan variabel V3. Variabel ini juga memiliki korelasi positif dengan variabel V1, V5, V21, V22, dan V23.

2. Variabel V1 memiliki korelasi positif dengan variabel *time* dan variabel V10 serta memiliki korelasi negatif yang cukup tinggi dengan variabel V2 dan V3.

3. Variabel V2 memiliki korelasi negatif dengan cukup tinggi dengan variabel V1 dan *Amount* serta memiliki korelasi positif yang cukup tinggi dengan variabel V5 dan V7.

4. Variabel V5 memiliki korelasi positif yang cukup tinggi dengan variabel V2 dan V7 dan korelasi negatif yang cukup tinggi dengan variabel *Amount*.

5. Terdapat beberapa variabel lainnya yang memiliki korelasi positif yang cukup tinggi diantaranya V6 dengan V8, V21 dengan V22, dan V27 dengan V28.

6. Terdapat beberapa variabel yang memiliki korelasi positif dan negatif

yang rendah.

Matriks korelasi *spearman* pada gambar 3.5 mampu menangkap korelasi antar variabel lebih baik daripada matriks korelasi *pearson* pada gambar 3.4 karena jumlah variabel yang tidak memiliki korelasi lebih sedikit jika dibandingkan dengan matriks korelasi *spearman*, meskipun matriks korelasi pearson juga mampu menangkap beberapa korelasi antar variabel yang ditandai dengan warna yang lebih tebal dibandingkan matriks korelasi spearman. Dapat disimpulkan bahwa terdapat lebih banyak korelasi non-linear antar variabel dalam dataset yang dipakai.

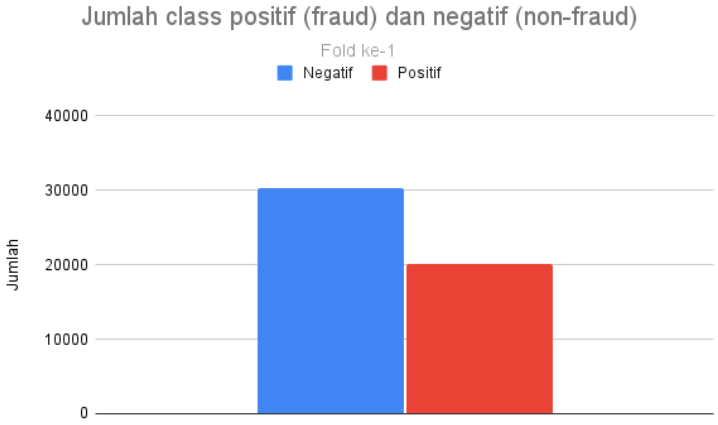
Berdasarkan penelitian [9], semua fitur dalam dataset digunakan. Oleh karena itu, dalam penelitian ini, semua fitur dalam dataset akan digunakan.

3.6 Data *splitting*

Pada tahap ini akan dilakukan pembagian dataset untuk proses pelatihan dan proses pengujian. Dataset akan dibagi menjadi 20% data uji dan 80% data latih. Data latih terdiri dari 38.218 data dengan 37.840 diantaranya merupakan data transaksi bukan penipuan dan 378 diantaranya merupakan data transaksi penipuan. Data uji terdiri dari sebanyak 9.555 data dengan 9.460 diantaranya merupakan data transaksi bukan penipuan dan 95 diantaranya merupakan data transaksi penipuan.

3.7 *Oversampling* dengan SMOTE

Berdasarkan gambar 2.16, terlihat bahwa dataset sangat *imbalance* dengan rasio kelas *fraud* sebesar 1% dari dataset. Pada tahap ini dilakukan *oversampling* pada data latih untuk menyelesaikan masalah dataset yang *imbalance* menggunakan SMOTE. Proses Oversampling menggunakan SMOTE ini akan digunakan dalam *5-fold cross-validation* ketika membagi data latih menjadi 5 fold dengan 4 fold akan menjadi data latih dan 1 fold akan menjadi data uji dan ketika melakukan pelatihan dengan kombinasi *hyperparameter* terbaik sebelum melakukan pengujian menggunakan data uji. Oversampling SMOTE akan dilakukan pada data pada kelas positif yang terdapat dalam data latih untuk meningkatkan jumlah data tersebut. Gambar 3.6 menunjukkan perbandingan data kelas positif yang berjumlah 20.181 dan kelas negatif yang berjumlah 30.272 pada fold pertama dalam data latih.



Gambar 3.6 Data latih setelah dilakukan *oversampling* dengan SMOTE pada fold ke 1.

3.8 Analisis kasus XGBoost

Pada bagian ini dilakukan analisis tahapan proses dengan melakukan perhitungan manual algoritme XGBoost. Untuk analisis kasus, *hyperparameter* regularisasi yang akan digunakan adalah λ sebesar 1, kemudian nilai γ sebesar 0 dan *min_child_weight* sebesar 0 untuk mencegah adanya pemberhentian pada saat membangun *tree*. Sedangkan *hyperparameter* untuk η (*learning rate*) senilai 0.4 dan jumlah *tree* yang akan digunakan (*n_estimators*) senilai 2 akan digunakan. Jumlah fitur yang akan digunakan pada analisis kasus sebanyak 2, yaitu fitur V1 dan V2. Karena nilai *hyperparameter* untuk jumlah fitur yang akan digunakan (*max_features*) adalah 'auto' [19], maka jumlah fitur yang akan digunakan untuk proses pelatihan adalah sebanyak 2 ('auto' merupakan jumlah fitur maksimal yang akan digunakan, jumlah fitur dalam dataset analisis kasus adalah 2).

Berikut merupakan contoh dataset yang akan digunakan dalam analisis kasus seperti yang terlihat pada gambar di bawah ini.

Index	v1	v2	Class
0	-30.55238004	16.71338924	1
1	-25.82598215	19.16723901	1
2	-12.80368875	9.666883426	0
3	-9.587359122	8.688022993	0
4	-28.70922925	22.05772899	1
5	-10.81352565	8.932731526	0
6	-2.312226542	1.951992011	1

Tabel 3.1 Dataset untuk analisis kasus.

3.8.1 Pembentukan *Tree* Pertama

Pada contoh kasus, data dengan index 0, 1, 2, dan 3 akan digunakan sebagai data latih, sedangkan data dengan index 4, 5 dan 6 akan digunakan sebagai data yang akan diuji. Langkah pertama adalah membuat prediksi inisial. XGboost menggunakan nilai 0.5 sebagai prediksi awal [19] sebagai inisialisasi nilai $f_0(x)$ algoritme XGBoost pada gambar 2.2. Kemudian konversi nilai prediksi awal ke dalam probabilitas menggunakan fungsi sigmoid sesuai persamaan 2.18.

Index	v1	v2	Class	Initial Prediction	Probability
0	-30.55238004	16.71338924	1	0.5	0.62
1	-25.82598215	19.16723901	1	0.5	0.62
2	-12.80368875	9.666883426	0	0.5	0.62
3	-9.587359122	8.688022993	0	0.5	0.62

Tabel 3.2 Inisialisasi nilai prediksi dan probabilitas untuk setiap data latih

Perhitungan nilai probabilitas untuk setiap data latih.

probabilitas $= \frac{1}{1+e^{-0.5}} = 0.62$

Langkah selanjutnya adalah menghitung nilai *gradient* dan *hessian* pada setiap data latih. Nilai *gradient* dapat dihitung menggunakan persamaan 2.8, sedangkan nilai *hessian* dapat dihitung menggunakan persamaan 2.9.

Index	v1	v2	Class	Initial Prediction	Probability	Gradient	Hessian
0	-30.55238004	16.71338924	1	0.5	0.62	-0.38	0.24
1	-25.82598215	19.16723901	1	0.5	0.62	-0.38	0.24
2	-12.80368875	9.666883426	0	0.5	0.62	0.62	0.24
3	-9.587359122	8.688022993	0	0.5	0.62	0.62	0.24

Tabel 3.3 Menghitung nilai *gradient* dan *hessian* untuk setiap data latih.

$$Gradient_0 = 0.62 - 1 = -0.38$$
$$Gradient_1 = 0.62 - 1 = -0.38$$
$$Gradient_2 = 0.62 - 0 = 0.62$$
$$Gradient_3 = 0.62 - 0 = 0.62$$

$$Hessian_0 = 0.62(1 - 0.62) = 0.24$$
$$Hessian_1 = 0.62(1 - 0.62) = 0.24$$
$$Hessian_2 = 0.62(1 - 0.62) = 0.24$$
$$Hessian_3 = 0.62(1 - 0.62) = 0.24$$

Setelah menghitung *gradient* dan *hessian* pada masing-masing data latih, selanjutnya algoritme akan membentuk sejumlah *tree* yang digunakan untuk melakukan prediksi. Untuk membentuk *tree* pertama, algoritme menentukan *split* terbaik fitur v1 dan v2. Sebelum menentukan split, algoritme mengurutkan nilai pada masing-masing fitur untuk mempermudah perhitungan. Algoritme menentukan *split* terbaik berdasarkan nilai *gain* terbesar.

Nilai *gain* dihitung berdasarkan masing-masing fitur. Untuk menghitung nilai *gain*, diperlukan juga nilai *gradient left*, *gradient right*, *hessian left*, *hessian right* pada masing-masing fitur yang telah diurutkan sesuai persamaan 2.11. *Gradient left* menghitung nilai *gradient* pada data index tertentu terhadap data lain yang nilai fiturnya lebih kecil sama dengan dengan dari data tersebut. Sedangkan *gradient right* menghitung nilai *gradient* data index tertentu dengan data lain yang nilai fiturnya lebih besar dari data tersebut. *Hessian left* menghitung nilai *hessian* pada data index tertentu terhadap data lain yang nilai fiturnya lebih kecil sama dengan dengan dari data tersebut. Sedangkan *Hessian right* menghitung nilai *hessian* data index tertentu dengan data lain yang nilai fiturnya lebih besar dari data tersebut.

Sebagai contoh, nilai fitur v1 pada data latih index 2 adalah -12.803688754721. Untuk menghitung nilai *gl, gr, hl, hr*, dan *gain* fitur v1, perlu dicari data lain yang nilai fitur v1 <= -12.803688754721. Dalam hal ini, data latih index ke 0 dan 1 memenuhi kriteria tersebut dengan nilai fitur

v1 sebesar -30.552380043581 dan -25.825982149082. Kemudian lakukan penjumlahan *gradient* untuk mendapatkan *gradient left* fitur v1 pada data latih index ke 2. Untuk menghitung *gradient right*, maka gunakan kriteria lebih besar daripada nilai fitur, sehingga untuk menghitung *gradient right* pada fitur v1 untuk data indeks ke 2, perlu dicari data lain yang nilai fitur v1 > 9.66688342609514. Terdapat data indeks ke 0 dan ke 1 yang memenuhi kriteria ini dengan nilai fitur v2 sebesar 16.7133892350242 dan 19.1672390103062, lalu jumlahkan *gradient*-nya. Lakukan kriteria yang sama untuk mencari *hessian left* dan *hessian right*, namun dengan menjumlahkan nilai *hessian*-nya. Gambar 3.4 menunjukkan data pada fitur v1 yang telah diurutkan untuk mempermudah perhitungan. Warna merah menunjukkan data yang akan digunakan untuk perhitungan *gradient left* dan *hessian left* fitur v1 untuk data indeks ke 2 dengan kriteria <= -12.803688754721. Warna biru menunjukkan data yang akan digunakan untuk perhitungan *gradient right* dan *hessian right* fitur v1 untuk data indeks ke 2 dengan kriteria > -12.803688754721.

Index	v1	v2	Class	Initial Prediction	Probability	Gradient	Hessian
0	-30.55238004	16.71338924	1	0.5	0.62	-0.38	0.24
1	-25.82598215	19.16723901	1	0.5	0.62	-0.38	0.24
2	-12.80368875	9.666883426	0	0.5	0.62	0.62	0.24
3	-9.587359122	8.688022993	0	0.5	0.62	0.62	0.24

Tabel 3.4 Data pada fitur v1 yang telah diurutkan.

Perhitungan *gradient left, gradient right, hessian left* dan *hessian right* fitur v1 pada data latih index 2 pada *tree* pertama.

$gl_2 = -0.38 + -0.38 + 0.62 = -0.14$
$gr_2 = 0.62$
$hl_2 = 0.24 + 0.24 + 0.24 = 0.72$
$hr_2 = 0.24$

Kemudian menghitung nilai *gradient left (gl), gradient right (gr), hessian left (hl)* dan *hessian right (hr)* fitur v1 untuk data latih sisanya, yaitu data indeks 0, 1, dan 3.

Perhitungan *gl, gr, hl* dan *hr* fitur v1 untuk data indeks 0, 1, dan 3 pada *tree* pertama. $gl_0 = -0.38$ $gr_0 = -0.38 + 0.62 + 0.62 = 0.86$
$gl_1 = -0.38 - 0.38 = -0.76$ $gr_1 = 0.62 + 0.62 = 1.24$
$gl_3 = -0.38 - 0.38 + 0.62 + 0.62 = 0.48$ $gr_3 = 0$
$hl_0 = 0.24$ $hr_0 = 0.24 + 0.24 + 0.24 = 0.72$
$hl_1 = 0.24 + 0.24 = 0.48$ $hr_1 = 0.24 + 0.24 = 0.48$
$hl_3 = 0.24 + 0.24 + 0.24 + 0.24 = 0.96$ $hr_3 = 0$

Selanjutnya, menghitung gain fitur v1 untuk setiap data latih sesuai persamaan 2.11 dengan nilai λ sebesar 1.

Perhitungan *gain* sesuai persamaan 2.11 untuk fitur v1 pada setiap data latih pada *tree* pertama.

$$gain_0 = \frac{1}{2}[\frac{-0.38^2}{0.24+1} + \frac{0.86^2}{0.72+1} - \frac{(-0.38+0.86)^2}{0.24+0.72+1}] - 0 = 0.21$$

$$gain_1 = \frac{1}{2}[\frac{-0.76^2}{0.48+1} + \frac{1.24^2}{0.48+1} - \frac{(-0.76+1.24)^2}{0.48+0.48+1}] - 0 = 0.66$$

$$gain_2 = \frac{1}{2}[\frac{-0.14^2}{0.72+1} + \frac{0.62^2}{0.24+1} - \frac{(-0.14+0.62)^2}{0.72+0.24+1}] - 0 = 0.10$$

$$gain_3 = \frac{1}{2}[\frac{0.48^2}{0.96+1} + \frac{0^2}{0+1} - \frac{(0.48+0)^2}{0.96+0+1}] - 0 = 0$$

Index	v1	gl	hl	gr	hr	gain
0	-30.55238004	-0.38	0.24	0.86	0.72	0.21
1	-25.82598215	-0.76	0.48	1.24	0.48	0.66
2	-12.80368875	-0.14	0.72	0.62	0.24	0.10
3	-9.587359122	0.48	0.96	0.00	0.00	0.00

Tabel 3.5 Hasil perhitungan *gradient*, *hessian*, dan *gain* pada *tree* pertama fitur v1.

Tabel 3.5 menunjukkan hasil perhitungan *gradient left* (gl), *gradient right* (gr), *hessian left* (hl), *hessian right* (hr), *gain* pada fitur v1 yang telah diurutkan untuk setiap data latih pada untuk pembentukan *tree* pertama.

Lakukan perhitungan *gradient left, gradient right, hessian left, hessian right* dan nilai *gain* untuk fitur v2 pada setiap data latih.

Perhitungan *gl, gr, hl, hr* fitur v2 pada *tree* pertama. $gl_0 = 0.62 + 0.62 - 0.38 = 0.86 gr_0 = -0.38$
$gl_1 = 0.62 + 0.62 - 0.38 - 0.38 = 0.48 gr_1 = 0$
$gl_2 = 0.62 + 0.62 = 1.24$ $gr_2 = -0.38 - 0.38 = -0.76$
$gl_3 = 0.62$ $gr_3 = 0.62 - 0.38 - 0.38 = -0.14$
$hl_0 = 0.24 + 0.24 + 0.24 = 0.72$ $hr_0 = 0.24$
$hl_1 = 0.24 + 0.24 + 0.24 + 0.24 = 0.96$ $hr_1 = 0$
$hl_2 = 0.24 + 0.24 = 0.48$ $hr_2 = 0.24 + 0.24 = 0.48$
$hl_3 = 0.24$ $hr_3 = 0.24 + 0.24 + 0.24 = 0.72$

Perhitungan *gain* untuk fitur v2 pada setiap data latih pada *tree* pertama.

$$gain_0 = \frac{1}{2}[\frac{0.86^2}{0.72+1} + \frac{-0.38^2}{0.24+1} - \frac{(0.86+-0.38)^2}{0.72+0.24+1}] - 0 = 0.21$$

$$gain_1 = \frac{1}{2}[\frac{0.48^2}{0.96+1} + \frac{0^2}{0+1} - \frac{(0.48+0)^2}{0.96+0+1}] - 0 = 0$$

$$gain_2 = \frac{1}{2}[\frac{1.24^2}{0.48+1} + \frac{-0.76^2}{0.48+1} - \frac{(1.24+-0.76)^2}{0.48+0.48+1}] - 0 = 0.66$$

$$gain_3 = \frac{1}{2}[\frac{0.62^2}{0.24+1} + \frac{-0.14^2}{0.72+1} - \frac{(0.62+-0.14)^2}{0.24+0.72+1}] - 0 = 0.10$$

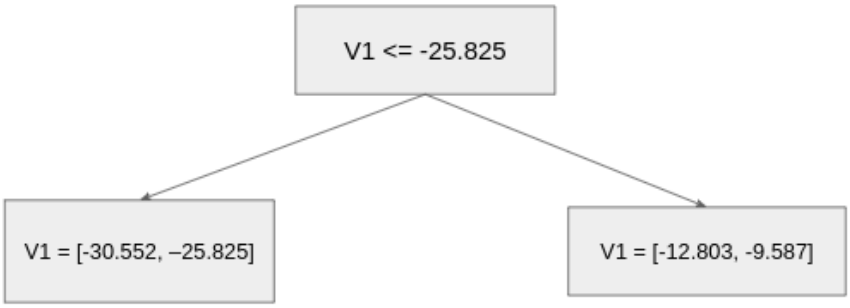
Index	v2	gl	hl	gr	hr	gain
3	8.688022993	0.62	0.24	-0.14	0.72	0.10
2	9.666883426	1.24	0.48	-0.76	0.48	0.66
0	16.71338924	0.86	0.72	-0.38	0.24	0.21
1	19.16723901	0.48	0.96	0.00	0.00	0.00

Tabel 3.6 Hasil perhitungan *gradient*, *hessian*,dan *gain tree* pertama fitur v2.

Tabel 3.6 menunjukkan hasil perhitungan *gradient left* (gl), *gradient*

right (gr), *hessian left* (hl), *hessian right* (hr), *gain* pada fitur v2 yang telah diurutkan untuk setiap data latih untuk pembentukan *tree* pertama.

Pada fitur v1 diperoleh nilai *gain* terbesar sebesar 0.66 pada kondisi ketika v1 <= -25.82598215, sedangkan pada fitur v2 diperoleh nilai *gain* tertinggi sebesar 0.66 pada kondisi ketika v2 <= 9.666883426. Oleh karena fitur v1 merupakan fitur pertama pada dataset, maka fitur v1 akan menjadi *split* pertama. Gambar 3.7 merupakan visualisasi *tree* pertama setelah melakukan *split* berdasarkan fitur v1 dengan kondisi v1 <= -25.82598215.



Gambar 3.7 Visualisasi *tree* pertama setelah melakukan *split*.

v1	v2	Class	Initial Prediction	Probability	gradient	hessian
-30.55238004	16.71338924	1	0.5	0.62	-0.38	0.24
-25.82598215	19.16723901	1	0.5	0.62	-0.38	0.24

Tabel 3.7 Tabel pada *leaf node* kiri berdasarkan gambar 3.7

v1	v2	Class	Initial Prediction	Probability	gradient	hessian
-12.80368875	9.666883426	0	0.5	0.62	0.62	0.24
-9.587359122	8.688022993	0	0.5	0.62	0.62	0.24

Tabel 3.8 Tabel pada *leaf node* kanan berdasarkan gambar 3.7

Setelah melakukan *split* berdasarkan fitur v1, algoritme harus perlu melakukan *split* kembali hingga mencapai kedalaman tertentu, atau hingga mendapatkan node yang *pure*. Berdasarkan tabel 3.7, *leaf node* kiri telah berisikan semua data latih dengan kelas 1, sedangkan pada tabel 3.8 , *right node* kanan juga telah berisikan semua data latih dengan kelas 0, artinya *leaf node* kiri dan kanan merupakan *leaf node* yang *pure*. *Leaf node* tersebut mampu membedakan setiap nilai pada label kelas (*Class*), sehingga tidak perlu melakukan *split* lagi. Setiap *leaf node* memiliki nilai bobot tersendiri, yang bisa dihitung menggunakan persamaan 2.12.

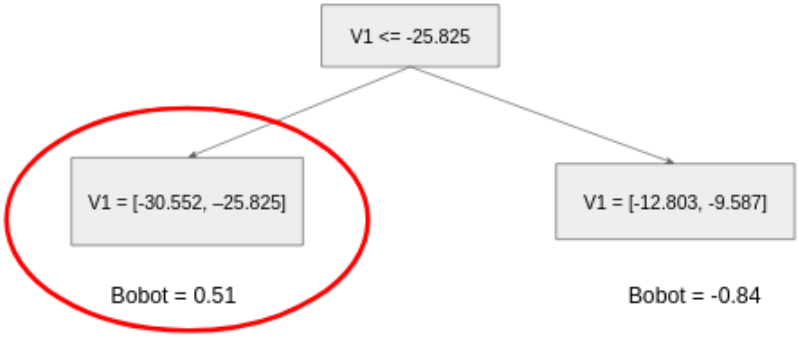
Perhitungan nilai bobot pada *leaf node* dengan fitur v1 <= -25.82598215 sesuai dengan tabel 3.7 pada *tree* pertama.

$G = -0.38 - 0.38 = -0.76$
$H = 0.24 + 0.24 = 0.48$
$\lambda = 1$
$w = -\frac{0.76}{0.48+1} = 0.51$

Perhitungan nilai bobot pada *leaf node* dengan fitur v1 > -25.82598215 sesuai dengan tabel 3.8 pada *tree* pertama.

$G = 0.62 + 0.62 = 1.24$
$H = 0.24 + 0.24 = 0.48$
$\lambda = 1$
$w = -\frac{1.24}{0.48+1} = -0.84$

Tahap selanjutnya adalah membuat prediksi baru menggunakan nilai bobot yang telah dihitung sebelumnya. Misalnya untuk membuat prediksi pada data indeks ke 0 dengan nilai v1 = -30.552380043581, maka perlu dilakukan penelusuran *tree* sesuai gambar 3.7. Terlihat bahwa fitur v1 data indeks ke 0 memiliki nilai yang lebih kecil daripada v1 = -25.82598215, sehingga data indeks ke 0 merupakan anggota dari *leaf node* kiri, sehingga nilai bobotnya adalah 0.51. Gambar 3.8 merupakan visualisasi penelusuran *tree* pertama untuk mendapatkan nilai bobot yang sesuai untuk setiap data latih. Terlihat bahwa data indeks ke 0 masuk ke dalam anggota *leaf node* dengan fitur v1 <= -25.82598215, dengan nilai bobot sebesar 0.51.



Gambar 3.8 Visualisasi penelusuran *tree* pertama.

Selanjutnya adalah memperbaharui nilai prediksi awal dengan nilai bobot dari *leaf node* yang dikalikan dengan *learning rate* sesuai persamaan 2.14.

Memperbaharui nilai prediksi awal untuk setiap data latih dan mengkonversi menjadi probabilitas untuk setiap data latih.

$$NewPrediction_0 = 0.5 + 0.4 \times 0.51 = 0.70$$
$$NewPrediction_1 = 0.5 + 0.4 \times 0.51 = 0.70$$
$$NewPrediction_2 = 0.5 + 0.4 \times -0.84 = 0.16$$
$$NewPrediction_3 = 0.5 + 0.4 \times -0.84 = 0.16$$

$$probability_0 = \frac{1}{1+e^{-0.70}} = 0.67$$

$$probability_1 = \frac{1}{1+e^{-0.70}} = 0.67$$

$$probability_2 = \frac{1}{1+e^{-0.16}} = 0.54$$

$$probability_3 = \frac{1}{1+e^{-0.16}} = 0.54$$

3.8.2 Pembentukan *Tree* Kedua

Hitung nilai *gradient* dan *hessian* untuk *tree* kedua dengan nilai probabilitas baru yang telah didapat. Lalu urutkan data berdasarkan setiap fitur untuk menentukan *split* terbaik.

Indeks	v1	v2	Class	Initial Prediction	Probability	Gradient	Hessian	New Prediction	New Probability
0	-30.55238004	16.71338924	1	0.5	0.62	-0.38	0.24	0.70	0.67
1	-25.82598215	19.16723901	1	0.5	0.62	-0.38	0.24	0.70	0.67
2	-12.80368875	9.666883426	0	0.5	0.62	0.62	0.24	0.16	0.54
3	-9.587359122	8.688022993	0	0.5	0.62	0.62	0.24	0.16	0.54

Tabel 3.9 Hasil prediksi dan probabilitas baru untuk setiap data latih .

Indeks	v1	v2	Class	New Prediction	Probability	New Gradient	New Hessian
0	-30.55238004	16.71338924	1	0.70	0.67	-0.33	0.22
1	-25.82598215	19.16723901	1	0.70	0.67	-0.33	0.22
2	-12.80368875	9.666883426	0	0.16	0.54	0.54	0.25
3	-9.587359122	8.688022993	0	0.16	0.54	0.54	0.25

Tabel 3.10 Nilai *gradient* dan *hessian* untuk *tree* kedua .

$$Gradient_0 = 0.67 - 1 = -0.33$$
$$Gradient_1 = 0.67 - 1 = -0.33$$
$$Gradient_2 = 0.54 - 0 = 0.54$$
$$Gradient_3 = 0.54 - 0 = 0.54$$

$$Hessian_0 = 0.67(1 - 0.67) = 0.22$$
$$Hessian_1 = 0.67(1 - 0.67) = 0.22$$
$$Hessian_2 = 0.54(1 - 0.54) = 0.25$$
$$Hessian_3 = 0.54(1 - 0.54) = 0.25$$

Kemudian menghitung *gradient left*, *gradient right*, *hessian left*, dan *hessian right* untuk tree kedua untuk setiap data latih.

Perhitungan *gl, gr, hl, hr* fitur v1 untuk setiap data latih pada *tree* kedua. $gl_0 = -0.33$ $gr_0 = -0.33 + 0.54 + 0.54 = 0.75$
$gl_1 = -0.33 - 0.33 = -0.66$ $gr_1 = 0.54 + 0.54 = 1.08$
$gl_2 = -0.33 - 0.33 + 0.54 = -0.12$ $gr_2 = 0.54$
$gl_3 = -0.33 - 0.33 + 0.54 + 0.54 = 0.42$ $gr_3 = 0$
$hl_0 = 0.22$ $hr_0 = 0.22 + 0.25 + 0.25 = 0.72$
$hl_1 = 0.22 + 0.22 = 0.44$ $hr_1 = 0.25 + 0.25 = 0.50$
$hl_2 = 0.22 + 0.22 + 0.25 = 0.69$ $hr_2 = 0.25$
$hl_3 = 0.22 + 0.22 + 0.25 + 0.25 = 0.94$ $hr_3 = 0$

Perhitungan *gain* fitur v1 untuk setiap data latih pada *tree* kedua.

$$gain_0 = \frac{1}{2}[\frac{-0.33^2}{0.22+1} + \frac{0.75^2}{0.72+1} - \frac{(-0.33+0.75)^2}{0.22+0.72+1}] - 0 = 0.16$$

$$gain_1 = \frac{1}{2}[\frac{-0.66^2}{0.44+1} + \frac{1.08^2}{0.50+1} - \frac{(-0.66+1.24)^2}{0.44+0.50+1}] - 0 = 0.5$$

$$gain_2 = \frac{1}{2}[\frac{-0.12^2}{0.69+1} + \frac{0.54^2}{0.25+1} - \frac{(-0.12+0.54)^2}{0.69+0.25+1}] - 0 = 0.08$$

$$gain_3 = \frac{1}{2}[\frac{0.42^2}{0.94+1} + \frac{0^2}{0+1} - \frac{(0.42+0)^2}{0.94+0+1}] - 0 = 0$$

Index	v1	gl	hl	gr	hr	gain
0	-30.55238004	-0.33	0.22	0.75	0.72	0.16
1	-25.82598215	-0.66	0.44	1.08	0.50	0.50
2	-12.80368875	-0.12	0.69	0.54	0.25	0.08
3	-9.587359122	0.42	0.94	0.00	0.00	0

Tabel 3.11 Nilai *gradient*, *hessian*, dan *gain tree* kedua fitur v1.

Tabel 3.11 menunjukkan nilai *gradient left, gradient right, hessian left, hessian right*, *gain* fitur v1 yang telah diurutkan untuk pembentukan *tree* kedua.

Kemudian menghitung nilai *gradient left, gradient right, hessian left,*

hessian right dan *gain* fitur v2 untuk *tree* kedua.

Perhitungan *gl, gr, hl, hr* dan *gain* fitur v2 pada *tree* kedua. $gl_0 = 0.54 + 0.54 - 0.33 = 0.75 gr_0 = -0.33$
$gl_1 = 0.54 + 0.54 - 0.33 - 0.33 = 0.42 gr_1 = 0$
$gl_2 = 0.54 + 0.54 = 1.08\ gr_2 = -0.33 - 0.33 = -0.66$
$gl_3 = 0.54\ gr_3 = 0.54 - 0.33 - 0.33 = -0.12$
$hl_0 = 0.25 + 0.25 + 0.22 = 0.72\ hr_0 = 0.22$
$hl_1 = 0.25 + 0.25 + 0.22 + 0.22 = 0.94\ hr_1 = 0$
$hl_2 = 0.25 + 0.25 = 0.50\ hr_2 = 0.22 + 0.22 = 0.44$
$hl_3 = 0.25\ hr_3 = 0.25 + 0.22 + 0.22 = 0.69$

Perhitungan *gain* fitur v2 pada *tree* kedua.
$gain_0 = \frac{1}{2}[\frac{0.75^2}{0.72+1} + \frac{-0.33^2}{0.22+1} - \frac{(0.75+-0.33)^2}{0.72+0.22+1}] - 0 = 0.16$
$gain_1 = \frac{1}{2}[\frac{0.42^2}{0.94+1} + \frac{0^2}{0+1} - \frac{(0.42+0)^2}{0.94+0+1}] - 0 = 0$
$gain_2 = \frac{1}{2}[\frac{1.08^2}{0.50+1} + \frac{-0.66^2}{0.44+1} - \frac{(1.08+-0.66)^2}{0.50+0.44+1}] - 0 = 0.50$
$gain_3 = \frac{1}{2}[\frac{0.54^2}{0.25+1} + \frac{-0.12^2}{0.69+1} - \frac{(0.54+-0.12)^2}{0.25+0.69+1}] - 0 = 0.08$

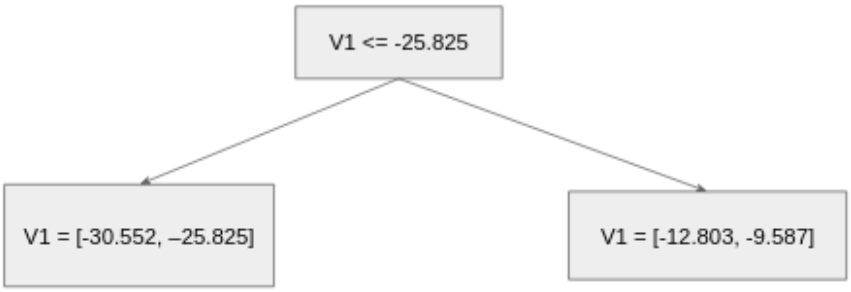
Index	v2	gl	hl	gr	hr	gain
3	8.688022993	0.54	0.25	-0.12	0.69	0.08
2	9.666883426	1.08	0.50	-0.66	0.44	0.50
0	16.71338924	0.75	0.72	-0.33	0.22	0.16
1	19.16723901	0.42	0.94	0.00	0.00	0.00

Tabel 3.12 Hasil perhitungan *gradient*, *hessian*, dan *gain tree* kedua fitur v2.

Tabel 3.12 menunjukkan hasil perhitungan *gradient left* (gl), *gradient right* (gr), *hessian left* (hl), *hessian right* (hr), dan *gain* pada fitur v2 yang telah diurutkan untuk setiap data latih untuk pembentukan *tree* kedua.

Berdasarkan hasil perhitungan nilai *gain* pada fitur v1 dan v2 untuk pembentukan tree kedua, fitur v1 memiliki nilai *gain* tertinggi sebesar 0.50 ketika nilai v1 <= -25.825982149082 dan fitur v2 memiliki nilai *gain* tertinggi sebesar 0.50 ketika fitur v2 memiliki nilai <= 9.66688342609514.

Karena fitur v1 merupakan fitur paling pertama pada dataset, maka fitur v1 <= -25.825982149082 akan digunakan sebagai *split* untuk membuat *tree* kedua. Gambar 3.9 merupakan visualisasi *tree* kedua setelah melakukan split berdasarkan fitur v1 dengan kondisi v1 <= -25.82598215.



Gambar 3.9 Visualisasi *tree* kedua setelah melakukan split.

Selanjutnya adalah menghitung nilai bobot untuk setiap *leaf node* sesuai persamaan 2.12.

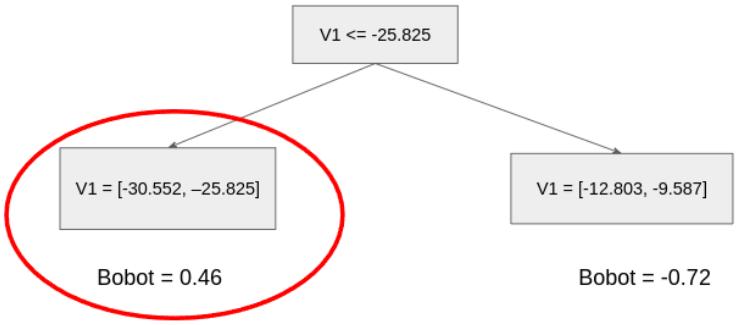
Perhitungan nilai bobot pada *leaf node* dengan fitur v1 <= -25.82598215 sesuai dengan tabel 3.8 pada *tree* kedua.

$G = -0.33 - 0.33 = -0.66$
$H = 0.22 + 0.22 = 0.44$
$\lambda = 1$
$w = -\frac{-0.66}{0.44+1} = 0.46$

Perhitungan nilai bobot pada *leaf node* dengan fitur v1 > -25.82598215 sesuai dengan tabel 3.7 pada *tree* kedua.

$G = 0.54 + 0.54 = 1.08$
$H = 0.25 + 0.25 = 0.50$
$\lambda = 1$
$w = -\frac{1.08}{0.50+1} = -0.72$

Langkah selanjutnya adalah memperbaharui nilai prediksi untuk setiap data latih berdasarkan *tree* yang telah dibuat sebelumnya. Sebagai contoh, untuk mendapatkan prediksi terbaru data indeks ke 0, dapat dilakukan dengan melakukan penelusuran *tree* kedua untuk mendapatkan nilai bobot.



Gambar 3.10 Visualisasi penelusuran *tree* kedua.

Gambar 3.10 menunjukkan bahwa nilai bobot untuk data indeks ke 0 adalah 0.46 pada *tree* kedua.

Selanjutnya adalah memperbaharui nilai prediksi dengan nilai bobot dari *leaf node* yang dikalikan dengan *learning rate* sesuai persamaan 2.14.

Memperbaharui nilai prediksi sebelumnya untuk setiap data latih dan mengkonversi menjadi probabilitas untuk setiap data latih sesuai persamaan.

$$NewestPrediction_0 = 0.71 + 0.4 \times 0.46 = 0.89$$
$$NewestPrediction_1 = 0.71 + 0.4 \times 0.46 = 0.89$$
$$NewestPrediction_2 = 0.71 + 0.4 \times -0.72 = -0.12$$
$$NewestPrediction_3 = 0.71 + 0.4 \times -0.72 = -0.12$$

$$Newestprobability_0 = \frac{1}{1+e^{-0.89}} = 0.71$$

$$Newestprobability_1 = \frac{1}{1+e^{-0.89}} = 0.71$$

$$Newestprobability_2 = \frac{1}{1+e^{-(-0.12)}} = 0.47$$

$$Newestprobability_3 = \frac{1}{1+e^{-(-0.12)}} = 0.47$$

Indeks	v1	v2	Class	Initial Prediction	New prediction	Newest Prediction	Final Probability
1	-30.55238004	16.71338924	1	0.5	0.71	0.89	0.71
2	-25.82598215	19.16723901	1	0.5	0.71	0.89	0.71
3	-12.80368875	9.666883426	0	0.5	0.16	-0.12	0.47
4	-9.587359122	8.688022993	0	0.5	0.16	-0.12	0.47

Tabel 3.13 Hasil prediksi akhir

3.8.3 Melakukan prediksi

Data yang akan digunakan untuk melakukan prediksi adalah data indeks ke 4, 5, dan 6 berdasarkan tabel 3.1. Prediksi dapat dihitung menggunakan persamaan 2.16.

Menghitung prediksi dan probabilitas untuk data indeks ke 4, 5, dan 6.

$Prediction_4 = 0.5 + 0.4 \times 0.51 + 0.4 \times 0.46 = 0.89$

$Probability_4 = \frac{1}{1+e^{-0.89}} = 0.71$

$Prediction_5 = 0.5 + 0.4 \times -0.84 + 0.4 \times -0.72 = -0.124$

$Probability_5 = \frac{1}{1+e^{-(-0.124)}} = 0.47$

$Prediction_6 = 0.5 + 0.4 \times -0.84 + 0.4 \times -0.72 = -0.124$

$Probability_6 = \frac{1}{1+e^{-(-0.124)}} = 0.47$

Karena hasil probabilitas lebih besar dari 0.5, maka data indeks ke 4 diprediksi merupakan kelas 1 dan memang kelas sesungguhnya adalah kelas 1 (*true positive*). Data indeks ke 5 diprediksi sebagai kelas 0 dan memang kelas sesungguhnya adalah kelas 0 (*true negative*). Data indeks ke 6 diprediksi sebagai kelas 0, tetapi kelas sesungguhnya adalah kelas 1 (*false negative*). Berikut merupakan perhitungan nilai akurasi, *recall* dan *f1-score* atau *f-measure*.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$
$$Accuracy = \frac{1+1}{1+1+0+1}$$
$$Accuracy = \frac{2}{3} \times 100\%$$
$$Accuracy = 67\%$$

$$Recall = \frac{TP}{TP+FN}$$
$$Recall = \frac{1}{1+1}$$
$$Recall = \frac{1}{2} \times 100\%$$
$$Recall = 50\%$$

$$F-measure = \frac{2\times TP}{2\times TP+FP+FN}$$
$$F-measure = \frac{2\times 1}{2\times 1+0+1}$$
$$F-measure = \frac{2}{3}$$
$$F-measure = \frac{2}{3} \times 100\%$$
$$F-measure = 67\%$$

3.9 Analisis Manual Perhitungan Korelasi

Pada bagian ini akan membahas analisis manual perhitungan korelasi menggunakan metode *pearson correlation coefficient* dan *spearman correlation coefficient*. Data yang akan digunakan dalam analisis korelasi pada bagian ini dapat dilihat seperti pada tabel 3.14. Berdasarkan tabel tersebut, terdapat kolom indeks yang digunakan sebagai penunjuk, dan terdapat 2 buah variabel yang akan dihitung nilai korelasinya, yaitu variabel pertama bernama *max_depth* dan variabel kedua bernama *recall*.

indeks	max_depth	recall
1	3	20
2	5	40
3	5	50
4	6	60
5	10	80
6	15	80

Tabel 3.14 Dataset yang digunakan untuk analisis manual korelasi

3.9.1 Perhitungan Pearson Correlation Coefficient

Untuk menghitung korelasi antara variabel *max_depth* dengan *recall* dengan menggunakan metode *pearson correlation coefficient*, perlu mencari terlebih dahulu nilai rata-rata masing-masing variabel. Variabel *max_depth* akan dilambangkan dengan x dan *recall* akan dilambangkan dengan y dalam perhitungan, sedangkan nilai rata-rata masing-masing variabel akan dilambangkan dengan variabel $\overline{x}$ untuk nilai rata-rata *max_depth* dan juga $\overline{y}$ untuk nilai-rata-rata *recall*.

$$\overline{x} = \frac{3+5+5+6+10+15}{6} = 7.3333$$
$$\overline{y} = \frac{20+40+50+60+80+80}{6} = 55$$

Setelah mendapatkan nilai rata-rata, maka selanjutnya adalah menghitung nilai $x - \overline{x}$ dan $y - \overline{y}$ pada setiap data.

indeks	max_depth	recall	$x - \overline{x}$	$y - \overline{y}$
1	3	20	-4.3333	-35
2	5	40	-2.3333	-15
3	5	50	-2.3333	-5
4	6	60	-1.3333	5
5	10	80	2.6667	25
6	15	80	7.6667	25

Tabel 3.15 Hasil perhitungan $x - \overline{x}$ dan $y - \overline{y}$ menggunakan metode *pearson*

$$x_1 - \overline{x} = 3 - 7.3333 = -4.3333$$
$$x_2 - \overline{x} = 5 - 7.3333 = -2.3333$$
$$x_3 - \overline{x} = 5 - 7.3333 = -2.3333$$
$$x_4 - \overline{x} = 6 - 7.3333 = -1.3333$$
$$x_5 - \overline{x} = 10 - 7.3333 = 2.6667$$
$$x_6 - \overline{x} = 15 - 7.3333 = 7.6667$$

$$y_1 - \overline{y} = 20 - 55 = -35$$
$$y_2 - \overline{y} = 40 - 55 = -15$$
$$y_3 - \overline{y} = 50 - 55 = -5$$
$$y_4 - \overline{y} = 60 - 55 = 5$$
$$y_5 - \overline{y} = 80 - 55 = 25$$
$$y_6 - \overline{y} = 80 - 55 = 25$$

Selanjutnya adalah menghitung nilai $x - \overline{x}^2$ dan juga nilai $y - \overline{y}^2$ untuk setiap data yang dapat dilihat pada perhitungan di bawah ini.

indeks	max_depth	recall	$x - \overline{x}$	$y - \overline{y}$	$(x - \overline{x})^2$	$(y - \overline{y})^2$
1	3	20	-4.3333	-35	18.7778	1225
2	5	40	-2.3333	-15	5.4444	225
3	5	50	-2.3333	-5	5.4444	25
4	6	60	-1.3333	5	1.7778	25
5	10	80	2.6667	25	7.1111	625
6	15	80	7.6667	25	58.7778	625

Tabel 3.16 Hasil perhitungan $(x - \overline{x})^2$ dan $(y - \overline{y})^2$ menggunakan metode *pearson*

$(x_1 - \overline{x})^2 = -4.3333^2 = 18.7778$
$(x_2 - \overline{x})^2 = -2.3333^2 = 5.4444$
$(x_3 - \overline{x})^2 = -2.3333^2 = 5.4444$
$(x_4 - \overline{x})^2 = -1.3333^2 = 1.7778$
$(x_5 - \overline{x})^2 = 2.6667^2 = 7.1111$
$(x_6 - \overline{x})^2 = 7.6667^2 = 58.7778$

$(y_1 - \overline{y})^2 = -35^2 = 1225$
$(y_2 - \overline{y})^2 = -15^2 = 225$
$(y_3 - \overline{y})^2 = -5^2 = 25$
$(y_4 - \overline{y})^2 = 5^2 = 25$
$(y_5 - \overline{y})^2 = 25^2 = 625$
$(y_6 - \overline{y})^2 = 25^2 = 625$

Kemudian, hitung nilai $(x - \overline{x})(y - \overline{y})$ pada setiap data yang dapat dilihat pada perhitungan di bawah ini.

indeks	max_depth	recall	$x - \overline{x}$	$y - \overline{y}$	$(x - \overline{x})^2$	$(y - \overline{y})^2$	$(x - \overline{x})(y - \overline{y})$
1	3	20	-4.3333	-35	18.7778	1225	151.6667
2	5	40	-2.3333	-15	5.4444	225	35.0000
3	5	50	-2.3333	-5	5.4444	25	11.6667
4	6	60	-1.3333	5	1.7778	25	-6.6667
5	10	80	2.6667	25	7.1111	625	66.6667
6	15	80	7.6667	25	58.7778	625	191.6667

Tabel 3.17 Hasil perhitungan $(x - \overline{x})(y - \overline{y})$ menggunakan metode *pearson*

$$(x_1 - \overline{x})(y_1 - \overline{y}) = -4.3333 * -35 = 151.6667$$
$$(x_2 - \overline{x})(y_2 - \overline{y}) = -2.3333 * -15 = 35$$
$$(x_3 - \overline{x})(y_3 - \overline{y}) = -2.3333 * -5 = 11.6667$$
$$(x_4 - \overline{x})(y_4 - \overline{y}) = -1.3333 * 5 = -6.6667$$
$$(x_5 - \overline{x})(y_5 - \overline{y}) = 2.6667 * 25 = 66.6667$$
$$(x_6 - \overline{x})(y_6 - \overline{y}) = 7.6667 * 25 = 191.6667$$

Lalu hitung nilai penjumlahan pada masing-masing data untuk mendapatkan $\sum(x - \overline{x}^2)$, $\sum(y - \overline{y}^2)$, dan $\sum(x - \overline{x})(y - \overline{y})$.

$$\sum(x - \overline{x}^2) = 18.7778 + 5.4444 + 5.4444 + 1.7778 + 7.1111 + 58.7778 = 97.3333$$

$$\sum(y - \overline{y}^2) = 1225 + 225 + 25 + 25 + 625 + 625 = 2750$$

$$\sum(x - \overline{x})(y - \overline{y}) = 151.6667 + 35 + 11.6667 + -6.6667 + 66.6667 + 191.6667 = 450$$

Selanjutnya adalah menghitung *pearson correlation coefficient* dengan menggunakan persamaan 2.19.

$$r = \frac{450}{\sqrt{97.3333 \times 2750}} = 0.8697$$

Berdasarkan hasil perhitungan menggunakan metode *pearson*, didapatkan hasil *pearson correlation coefficient* sebesar 0.8697 dan nilainya mendekati 1. Hal ini menandakan bahwa variabel *max_depth* memiliki korelasi positif yang kuat dengan variabel *recall*.

3.9.2 Perhitungan Spearman Rank Correlation Coefficient

Untuk menghitung korelasi antara variabel *max_depth* dengan *recall* dengan menggunakan metode *spearman rank correlation coefficient*, langkah-langkah yang dilakukan hampir mirip dengan menggunakan *pearson correlation coefficient* karena rumus yang digunakan juga sama. Perbedaannya adalah, perlu melakukan *ranking* pada setiap data pada masing-masing variabel. Variabel *max_depth* akan dilambangkan dengan x dan variabel *recall* akan dilambangkan dengan y.

Langkah pertama dalam perhitungan ini adalah melakukan *ranking* dari data yang memiliki nilai data. Data yang memiliki nilai terendah mendapatkan nilai *ranking* sebesar 1 dan seterusnya hingga semua data pada setiap variabel mendapatkan nilai *ranking*. Apabila terdapat data dengan memiliki nilai yang sama, maka *ranking* yang diambil adalah rata-rata dari *ranking* yang terlibat. Pada dataset 3.14, pada variabel *max_depth* terdapat data dengan nilai yang sama, yaitu data indeks ke 2 dan 3. *Ranking* yang terlibat pada data indeks ke 2 dan 3 adalah *ranking* 2 dan 3, maka rata-rata *ranking* adalah 2.5.

indeks	max_depth	recall	Rank x	Rank y
1	3	20	1	1
2	5	40	2.5	2
3	5	50	2.5	3
4	6	60	4	4
5	10	80	5	5.5
6	15	80	6	5.5

Tabel 3.18 Hasil *ranking* x dan y

$Rankx_1 = 1$
$Rankx_2 = \frac{2+3}{2} = 2.5$
$Rankx_3 = \frac{2+3}{2} = 2.5$
$Rankx_4 = 4$
$Rankx_5 = 5$
$Rankx_6 = 6$

$Ranky_1 = 1$
$Ranky_2 = 2$
$Ranky_3 = 3$
$Ranky_4 = 4$
$Ranky_5 = \frac{5+6}{2} = 5.5$
$Ranky_6 = \frac{5+6}{2} = 5.5$

Selanjutnya, nilai x dan y yang akan digunakan adalah nilai *ranking* yang telah ditentukan sebelumnya.

indeks	x	y
1	1	1
2	2.5	2
3	2.5	3
4	4	4
5	5	5.5
6	6	5.5

Tabel 3.19 Hasil *ranking* menjadi data x dan y

Lalu hitunglah nilai rata-rata $\overline{x}$ dan nilai rata-rata $\overline{y}$ seperti perhitungan di bawah.

$\overline{x} = \frac{1+2.5+2.5+4+5+6}{6} = 3.5$
$\overline{y} = \frac{1+2+3+4+5.5+5.5}{6} = 3.5$

Langkah selanjutnya adalah menghitung $x - \overline{x}$ dan $y - \overline{y}$.

indeks	x	y	$x - \overline{x}$	$y - \overline{y}$
1	1	1	-2.5000	-2.5
2	2.5	2	-1.0000	-1.5
3	2.5	3	-1.0000	-0.5
4	4	4	0.5000	0.5
5	5	5.5	1.5000	2
6	6	5.5	2.5000	2

Tabel 3.20 Hasil perhitungan $x - \overline{x}$ dan $y - \overline{y}$ pada metode *spearman*.

$$x_1 - \overline{x} = 1 - 3.5 = -2.5$$
$$x_2 - \overline{x} = 2.5 - 3.5 = -1$$
$$x_3 - \overline{x} = 2.5 - 3.5 = -1$$
$$x_4 - \overline{x} = 4 - 3.5 = -0.5$$
$$x_5 - \overline{x} = 5 - 3.5 = 1.5$$
$$x_6 - \overline{x} = 6 - 3.5 = 2.5$$

$$y_1 - \overline{y} = 1 - 3.5 = -2.5$$
$$y_2 - \overline{y} = 2 - 3.5 = -1.5$$
$$y_3 - \overline{y} = 3 - 3.5 = -0.5$$
$$y_4 - \overline{y} = 4 - 3.5 = 0.5$$
$$y_5 - \overline{y} = 5.5 - 3.5 = 2$$
$$y_6 - \overline{y} = 5.5 - 3.5 = 2$$

Lalu hitunglah nilai dari $(x - \overline{x})^2$ dan $(y - \overline{y})^2$ seperti pada perhitungan di bawah.

indeks	max_depth	recall	$x-\overline{x}$	$y-\overline{y}$	$(x-\overline{x})^2$	$(y-\overline{y})^2$
1	1	1	-2.5000	-2.5	6.2500	6.25
2	2.5	2	-1.0000	-1.5	1.0000	2.25
3	2.5	3	-1.0000	-0.5	1.0000	0.25
4	4	4	0.5000	0.5	0.2500	0.25
5	5	5.5	1.5000	2	2.2500	4
6	6	5.5	2.5000	2	6.2500	4

Tabel 3.21 Hasil perhitungan $(x-\overline{x})^2$ dan $(y-\overline{y})^2$ pada metode *spearman*

$$x_1-\overline{x}^2=-2.5^2=6.25$$
$$x_2-\overline{x}^2=-1^2=1$$
$$x_3-\overline{x}^2=-1^2=1$$
$$x_4-\overline{x}^2=0.5^2=0.25$$
$$x_5-\overline{x}^2=1.5^2=2.25$$
$$x_6-\overline{x}^2=2.5^2=6.25$$

$$y_1-\overline{y}^2=-2.5^2=6.25$$
$$y_2-\overline{y}^2=-1.5^2=2.25$$
$$y_3-\overline{y}^2=-0.5^2=0.25$$
$$y_4-\overline{y}^2=0.5^2=0.25$$
$$y_5-\overline{y}^2=2^2=4$$
$$y_6-\overline{y}^2=2^2=4$$

Kemudian hitung nilai dari $(x-\overline{x})(y-\overline{y})$ seperti pada perhitungan di bawah.

indeks	max_depth	recall	$x-\overline{x}$	$y-\overline{y}$	$(x-\overline{x})^2$	$(y-\overline{y})^2$	$(x-\overline{x})(y-\overline{y})$
1	1	1	-2.5000	-2.5	6.2500	6.25	6.2500
2	2.5	2	-1.0000	-1.5	1.0000	2.25	1.5000
3	2.5	3	-1.0000	-0.5	1.0000	0.25	0.5000
4	4	4	0.5000	0.5	0.2500	0.25	0.2500
5	5	5.5	1.5000	2	2.2500	4	3.0000
6	6	5.5	2.5000	2	6.2500	4	5.0000

Tabel 3.22 Hasil perhitungan $(x-\overline{x})(y-\overline{y})$ pada metode *spearman*

$$(x_1-\overline{x})(y_1-\overline{y}) = -2.5 * -2.5 = 6.25$$
$$(x_2-\overline{x})(y_2-\overline{y}) = -1.0 * -1.5 = 1.5$$
$$(x_3-\overline{x})(y_3-\overline{y}) = -1.0 * -0.5 = 0.5$$
$$(x_4-\overline{x})(y_4-\overline{y}) = 0.5 * 0.5 = 0.25$$
$$(x_5-\overline{x})(y_5-\overline{y}) = 1.5 * 2 = 3$$
$$(x_6-\overline{x})(y_6-\overline{y}) = 2.5 * 2 = 5$$

Lalu hitung nilai penjumlahan pada masing-masing data untuk mendapatkan $\sum(x-\overline{x}^2)$, $\sum(y-\overline{y}^2)$, dan $\sum(x-\overline{x})(y-\overline{y})$.

$$\sum(x-\overline{x}^2) = 6.25+1+1+0.25+2.25+6.25 = 17$$

$$\sum(y-\overline{y}^2) = 6.25+2.25+0.25+0.25+4+4 = 17$$

$$\sum(x-\overline{x})(y-\overline{y}) = 6.25+1.50+0.50+0.25+3+5 = 16.5$$

Selanjutnya adalah menghitung *spearman rank correlation coefficient* dengan menggunakan persamaan 2.19 karena terdapat data dengan nilai *rank* yang sama.

$$r = \frac{16.5}{\sqrt{17 \times 17}} = 0.9705$$

Berdasarkan perhitungan diatas, didapat hasil nilai korelasi menggunakan *spearman rank correlation coefficient* sebesar 0.9705 dan nilainya lebih besar daripada *pearson* sehingga dalam hal ini terlihat bahwa *spearman* mampu menangkap korelasi lebih baik daripada *pearson*.