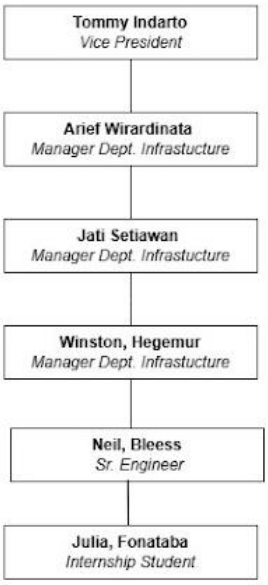
BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Pada kerja magang di PT Freeport Indonesia, awalnya penulis ditempatkan sebagai *MIS (Management Information System) Administration* yang dimentori oleh Ebit Darso *MIS Administration*. Namun hanya berlangsung selama 3 minggu awal saja, karena penulis merasa *jobdesk* yang diberikan kurang berhubungan dengan jurusan Teknik Komputer. Oleh karena itu penulis meminta kepada mentor untuk dipindahkan ke divisi yang lebih berhubungan dengan jurusan Teknik Komputer. Sehingga di minggu ke 4 periode magang, penulis dipindahkan di departemen yang berbeda namun masih tetap pada divisi yang sama. Penulis berada pada divisi *Management Information* System (MIS) di department *Operational Infrastructure*. Penulis berada dibawah koordinasi Manager Dept. Infrastructure Jati Setiawan, dan dimentori oleh Yulius Bless sebagai Supervisor seperti yang dilihat pada Gambar 3.1



Gambar 3. 1 Kedudukan Penulis di Divisi *Management Information System,* Dept. *Operational Infrastructure*

3.2 Tugas dan Urian Kerja Magang

Beberapa tugas, uraian, kendala dan Solusi yang dikerjakan penulis selama kegiatan magang dijelaskan pada subbab dibawah ini.

3.2.1 Tugas yang dilakukan

Selama periode magang di PT Freeport Indonesia, penulis terlibat dalam berbagai kegiatan. Ada 2 tahap utama yang dilalui penulis selama kegiatan magang. Pertama, penulis wajib mengikuti *training* yang difasilitasi oleh PT Freeport Indonesia yaitu *Safety Induction General*, *Safety Induction UG,* dan *Freeport Underground Mining* (FUM). Setelah mengikuti *training*, tahap kedua yaitu penulis diserahkan ke *user* yaitu departemen penempatan penulis untuk memulai kerja magang di departemen yang ditentukan. Tugas dan kegiatan yang dilakukan yaitu mengikuti *daily meeting* setiap pagi, mengikuti *safety meeting,* mengikuti survey lapangan, turun lapangan ke *underground* untuk mendokumentasi pekerjaan yang dilakukan untuk membuat laporan. Selain itu, penulis juga terlibat dalam beberapa proyek salah satunya yaitu proyek *oreflow network automation improvement.* Pada proyek ini penulis bertugas mendokumentasi kondisi eksisting jaringan dengan mengikuti survey lapangan, mengkoordinasi dengan klien terkait topologi jaringan, membuat topology jaringan menggunakan visio sebagai *future plan* dari proyek tersebut, serta membuat laporan hasil proyek. Selain itu, tugas dan kegiatan yang diberikan oleh mentor, penulis juga mengerjakan proyek utama yang berjudul 'Analisis *range* optimal antara *loader* dan Cisco Access Points (CAP) di *underground mine* - Grasberg Block Cave (GBC) di PT Freeport Indonesia'

3.2.2 Uraian Pelaksanaan Kerja Magang

Uraian pelaksanaan project magang yang dilakukan penulis pada project Analisis *Range* Optimal antara *Load Haul Dump* dan *Cisco Access*

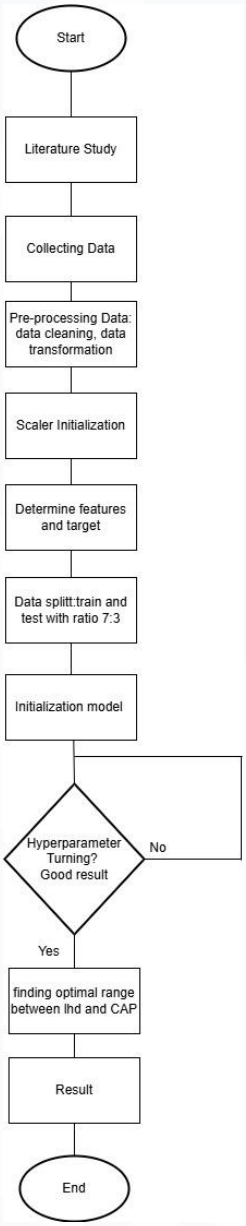
Points di *Underground Mine* - Grasberg Block Cave dengan *Random Forest Regression* selama dua bulan tiga minggu ini yaitu:

Tabel 3. 1 Linimasa Project Magang

Waktu	Tugas yang dilakukan
Minggu ke 1 – 4	Mengumpulkan Data: pengumpulan data diambil dari FTP.
Minggu ke - 5	Studi Literatur: membaca jurnal atau penelitian sebelumnya untuk mencari model *machine learning* apa yang digunakan berdasarkan tujuan dan data yang digunakan. Selain itu, membaca dari website penelitian seperti Kaggle, Medium, serta dari dokumentasi proyek penelitian sebelumnya di Github.
Minggu ke 6 - 7	Pengolahan Data: Dalam tahap ini dilakukan secara garis besar dilakukan *Preprocessing data*, *Processing data*, dan Implementasi model *machine learning.*
Minggu ke- 8-9	Menganalisis hasil prediksi dan melakukan analisis untuk rentang optimal antara *loader* dan CAP.
Minggu ke – 10	Presentasi Progress Proyek magang penulis di depan tim PCN.
Minggu ke – 11	Presentasi Akhir Proyek magang di depan tim MIS wajib menggunakan bahasa inggris.

- **Alur Pengerjaan project Magang**

Dalam project magang ini yang dilakukan terdiri dari beberapa tahap seperti yang ditunjukkan pada Gambar 3.2



Gambar 3. 2 Alur Pengerjaan Project Magang

Tabel 3. 2 Alur pengerjaan Project Magang

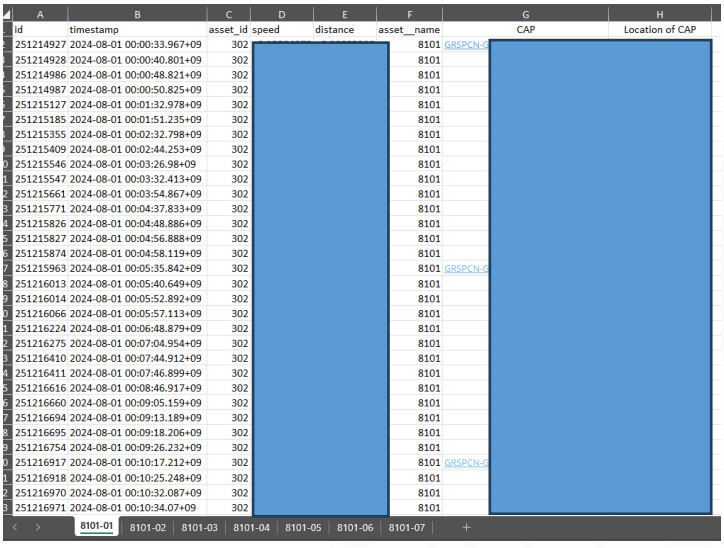
No	Tahap Pengerjaan Project Magang	Deskripsi
1	Studi Literatur	Pada tahap ini yang dilakukan yaitu membaca jurnal-jurnal dari penelitian sebelumnya terkait judul yang dipilih penulis. Selain itu, membaca dari website ilmiah seperti kaggle dan medium serta dokumentasi proyek di Github. Tujuan dari studi literatur untuk mencari tahu model *machine learning* apa yang paling baik digunakan agar sesuai dengan tujuan dari proyek penelitian serta sesuai dengan dataset yang digunakan. Selain itu, belajar lebih mendalam tentang *data cleaning*, *data transformation*, *handling outlier* dll.
2	*Data Colleting*	Pada tahap ini dilakukan pengumpulan data. Data dikumpulkan dari FTP data diambil dari 5 *loader* manual yang *running* yaitu 8101, 8105, 8108, 8119, dan 8135. Dengan periode 1 Agustus 2024 sampai 7 gustus 2024 per 5 menit. Alasan mengambil dari 5 *loader* manual saja karena pengumpulan data untuk beberapa parameter yang harus dilakukan manual yaitu untuk parameter *distance CAP LHD (m)* yang merepresentasikan jarak antara *loader* dan CAP data tersebut tidak ada di FTP, akan tetapi terdapat data latitude dan longitude dari kedua titik antara *loader* dan CAP, lalu dengan menggunakan rumus *haversine* untuk mencari jarak antara dua titik berdasarkan latitude dan longitudenya. Akan tetapi data latitude dan longitude dari *loader* dan CAP harus dikumpulkan secara manual dengan *tracking* per loader dalam satu minggu per lima menit yang berarti membutuhkan banyak waktu sedangkan penulis memiliki keterbatasan waktu. Oleh karena itu, hanya menggunakan 5 *loader* manual yang *running* dari 1 Agustus 2024 sampai 7 Agustus 2024. Data yang diambil bukan per area tetapi berdasarkan 5 loader yang dipilih, kemudian dilihat dari 1 Agustus 2024 sampai 7 Agustus 2024 kelima loader tersebut dalam

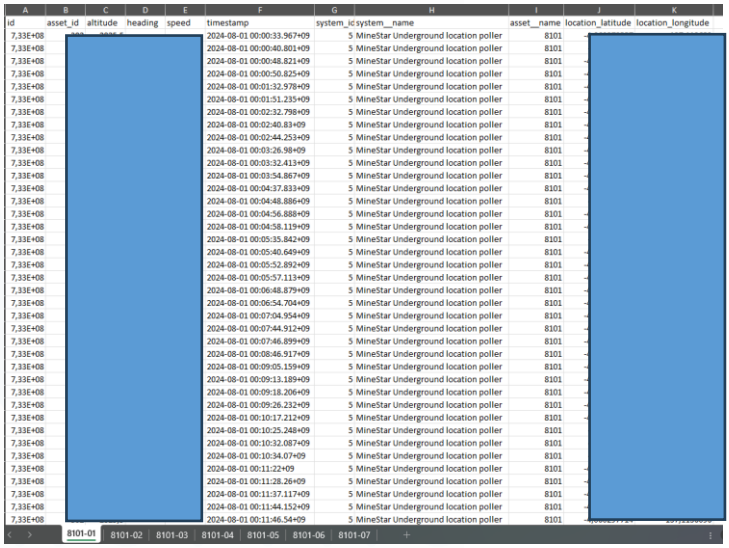
		melakukan *roaming data*, loader terhubung dengan CAP mana saja. Dalam pengerjaan project ini, data yang digunakan untuk proses *trainig* model terbagi dalam dua skenario. • Skenario 1: Dalam skenario ini, pelatihan dan pengujian model dilakukan pada kumpulan data yang merupakan kombinasi lima loader: loader 8101, 8105, 8108, 8119, dan 8135. • Skenario 2: Pelatihan dan Pengujian dengan Data dari Setiap Loader: Model dilatih dan diuji menggunakan data dari setiap loader secara terpisah.
3	*Pre-processing Data*	Dalam tahap ini dilakukan *data cleaning* dan *data transformation*. yaitu untuk mempersiapkan data sebelum diproses, mulai dari menghapus kolom-kolom data yang tidak digunakan, mengganti nama-nama kolom data agar memudahkan untuk memahami data, menggabungkan data berdasarkan nama *loader*, resampling data agar menjadi rata-rata per 5 menit karena untuk beberapa data seperti data 'speed-lhd-movement','altitude-lhd' dari hasil *export* datanya per 9 deik. Oleh karena itu, dilakukan resampling data per 5 menit. Setelah itu data disinkronkan berdasarkan *timestamp*. Selain itu, dilakukan *handling outlier* pada data, mengatasi *missing values* dari data dengan menghapus missing values, mengubah tipe data menjadi numerik, mengubah tipe data timestamp karena masih berupa object jadi diubah menjadi *datetime*, mencari jarak antara loader dan CAP berdasarkan data latitude dan longiude dari kedua titik menggunakan rumus *hevarsine.*
4	Inisialisasi Scaler	Pada tahap ini dalam project ini dilakukan inisialisasi scaler menggunakan *Robustscaler* karena, scaler ini tahan terhadap *outlier* pada data.
5	Menentukan *features* dan *target*	Pada tahap ini dalam project ini ditentukan *features* sebagai variabel independen sedangkan *target* adalah variabel dependen. Dalam project ini, *features* yang digunakan

		Jadi untuk *target* akan dilakukan per *target* terhadap *features* untuk melihat hubungan antara setiap *target* dengan *features.*
6	Data Split	Dalam project ini penulis juga membagi dataset menjadi data training dan data test dengan rasio 7:3. Dalam proses training data dilakukan dua skenario. Skenario 1: Dalam skenario ini, pelatihan dan pengujian model dilakukan pada kumpulan data yang merupakan kombinasi lima loader: loader 8101, 8105, 8108, 8119, dan 8135. Sedangkan, Skenario 2: Pelatihan dan Pengujian dengan Data dari Setiap Loader: Model dilatih dan diuji menggunakan data dari setiap loader secara terpisah.
7	Inisialisasi model	Dalam tahap ini dilakukan inisialisasi model yang digunakan yaitu Random Forest Regression.
8	Hyperparameter Tuning	Memilih kumpulan hyperparameter yang optimal untuk model pembelajaran mesin untuk meningkatkan performanya dan menghindari *overfitting*/*underfitting*. Dalam project ini, digunakan *Grid Search* untuk menentukan hyperparameter terbaik dengan *scoring* yang digunakan yaitu R2.
9	Prediksi	Pada tahap ini, dilakukan prediksi menggunakan model tebaik dari hasil perbandinan traininig model menggunakan skenario 1 dan skenario 2. Prediksi dilakukan untuk semua *target* terhadap *features.*
10	Mencari rentang optimal	Dalam tahap ini dari hasil prediksi kemudian dicari rentang optimal antara loader dan CAP, standar optimal yan g digunakan untuk setiap *target* dipakai dari standar dokumentasi Cisco seperti pada Tabel 3.3. Alasan penggunaan standar karena produk yang digunakan adalah produk Cisco. Untuk memperoleh rentang optimal antara loader dan CAP adalah hasil prediksi untuk semua *target* terhadap *features,* kemudian di lakukan *boolean indexing* atau *conditional filtering* seperti yang ditunjukkan pada Gambar 3.15.

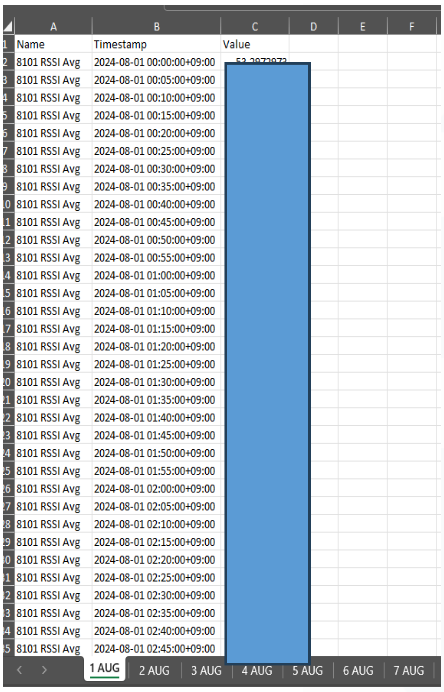
		Setelah itu, dibandingkan keseluruhan hasil dari *optimal_conditions.describe()* dari semua *target* yaitu nilai minimal dan nilai maximum per *features*. Dari hasil pencarian nilai minimal dan maximum inilah ditentukan jarak optimal antara loader dan CAP.

Gambar 3. 3 Tampilan data setelah di import dari FTP

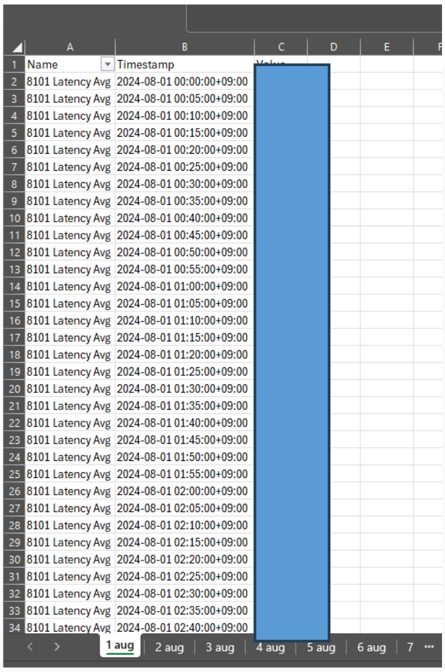




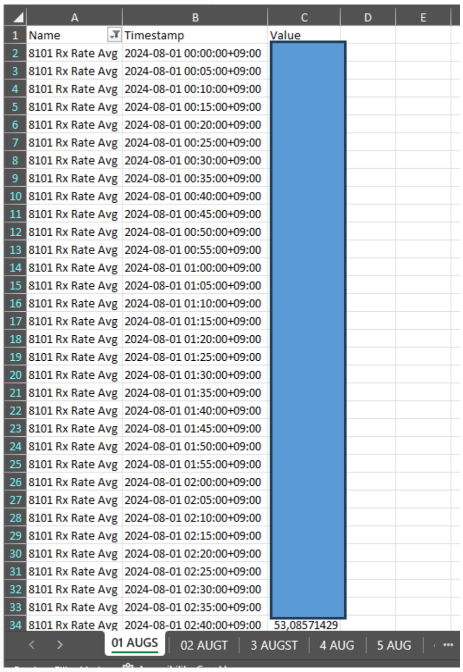
(a)Data Estimated movement lhd: id, timestamp, asset_id, speed, distance, asset_name (kiri) ; (b) Data location Stat LHD: id, asset_id, altitude, heading, speed, timestamp, system_id, system_name, asset_name, location_latitude, location_longitude (kanan)



	A	B	C
1	Name	Timestamp	Value
2	8101 RSSI Avg	2024-08-01 00:00:00+09:00	
3	8101 RSSI Avg	2024-08-01 00:05:00+09:00	
4	8101 RSSI Avg	2024-08-01 00:10:00+09:00	
5	8101 RSSI Avg	2024-08-01 00:15:00+09:00	
6	8101 RSSI Avg	2024-08-01 00:20:00+09:00	
7	8101 RSSI Avg	2024-08-01 00:25:00+09:00	
8	8101 RSSI Avg	2024-08-01 00:30:00+09:00	
9	8101 RSSI Avg	2024-08-01 00:35:00+09:00	
10	8101 RSSI Avg	2024-08-01 00:40:00+09:00	
11	8101 RSSI Avg	2024-08-01 00:45:00+09:00	
12	8101 RSSI Avg	2024-08-01 00:50:00+09:00	
13	8101 RSSI Avg	2024-08-01 00:55:00+09:00	
14	8101 RSSI Avg	2024-08-01 01:00:00+09:00	
15	8101 RSSI Avg	2024-08-01 01:05:00+09:00	
16	8101 RSSI Avg	2024-08-01 01:10:00+09:00	
17	8101 RSSI Avg	2024-08-01 01:15:00+09:00	
18	8101 RSSI Avg	2024-08-01 01:20:00+09:00	
19	8101 RSSI Avg	2024-08-01 01:25:00+09:00	
20	8101 RSSI Avg	2024-08-01 01:30:00+09:00	
21	8101 RSSI Avg	2024-08-01 01:35:00+09:00	
22	8101 RSSI Avg	2024-08-01 01:40:00+09:00	
23	8101 RSSI Avg	2024-08-01 01:45:00+09:00	
24	8101 RSSI Avg	2024-08-01 01:50:00+09:00	
25	8101 RSSI Avg	2024-08-01 01:55:00+09:00	
26	8101 RSSI Avg	2024-08-01 02:00:00+09:00	
27	8101 RSSI Avg	2024-08-01 02:05:00+09:00	
28	8101 RSSI Avg	2024-08-01 02:10:00+09:00	
29	8101 RSSI Avg	2024-08-01 02:15:00+09:00	
30	8101 RSSI Avg	2024-08-01 02:20:00+09:00	
31	8101 RSSI Avg	2024-08-01 02:25:00+09:00	
32	8101 RSSI Avg	2024-08-01 02:30:00+09:00	
33	8101 RSSI Avg	2024-08-01 02:35:00+09:00	
34	8101 RSSI Avg	2024-08-01 02:40:00+09:00	
35	8101 RSSI Avg	2024-08-01 02:45:00+09:00	

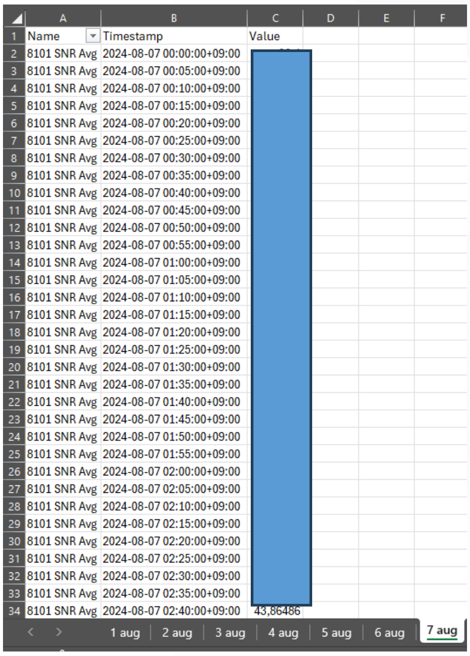


	A	B	C
1	Name	Timestamp	Value
2	8101 Latency Avg	2024-08-01 00:00:00+09:00	
3	8101 Latency Avg	2024-08-01 00:05:00+09:00	
4	8101 Latency Avg	2024-08-01 00:10:00+09:00	
5	8101 Latency Avg	2024-08-01 00:15:00+09:00	
6	8101 Latency Avg	2024-08-01 00:20:00+09:00	
7	8101 Latency Avg	2024-08-01 00:25:00+09:00	
8	8101 Latency Avg	2024-08-01 00:30:00+09:00	
9	8101 Latency Avg	2024-08-01 00:35:00+09:00	
10	8101 Latency Avg	2024-08-01 00:40:00+09:00	
11	8101 Latency Avg	2024-08-01 00:45:00+09:00	
12	8101 Latency Avg	2024-08-01 00:50:00+09:00	
13	8101 Latency Avg	2024-08-01 00:55:00+09:00	
14	8101 Latency Avg	2024-08-01 01:00:00+09:00	
15	8101 Latency Avg	2024-08-01 01:05:00+09:00	
16	8101 Latency Avg	2024-08-01 01:10:00+09:00	
17	8101 Latency Avg	2024-08-01 01:15:00+09:00	
18	8101 Latency Avg	2024-08-01 01:20:00+09:00	
19	8101 Latency Avg	2024-08-01 01:25:00+09:00	
20	8101 Latency Avg	2024-08-01 01:30:00+09:00	
21	8101 Latency Avg	2024-08-01 01:35:00+09:00	
22	8101 Latency Avg	2024-08-01 01:40:00+09:00	
23	8101 Latency Avg	2024-08-01 01:45:00+09:00	
24	8101 Latency Avg	2024-08-01 01:50:00+09:00	
25	8101 Latency Avg	2024-08-01 01:55:00+09:00	
26	8101 Latency Avg	2024-08-01 02:00:00+09:00	
27	8101 Latency Avg	2024-08-01 02:05:00+09:00	
28	8101 Latency Avg	2024-08-01 02:10:00+09:00	
29	8101 Latency Avg	2024-08-01 02:15:00+09:00	
30	8101 Latency Avg	2024-08-01 02:20:00+09:00	
31	8101 Latency Avg	2024-08-01 02:25:00+09:00	
32	8101 Latency Avg	2024-08-01 02:30:00+09:00	
33	8101 Latency Avg	2024-08-01 02:35:00+09:00	
34	8101 Latency Avg	2024-08-01 02:40:00+09:00	

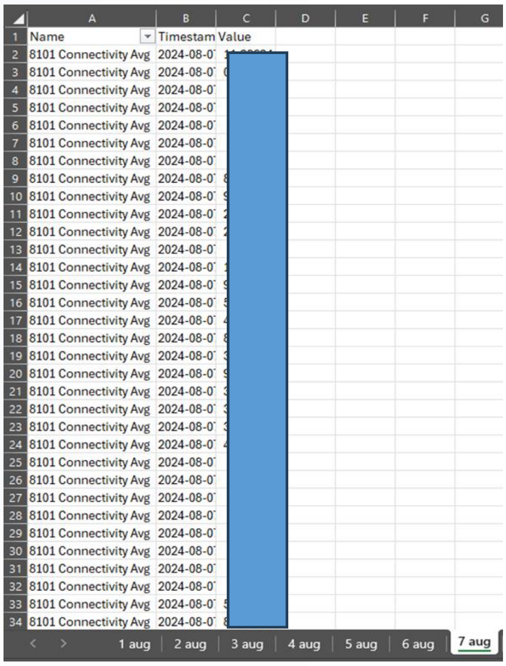


	A	B	C
1	Name	Timestamp	Value
2	8101 Rx Rate Avg	2024-08-01 00:00:00+09:00	
3	8101 Rx Rate Avg	2024-08-01 00:05:00+09:00	
4	8101 Rx Rate Avg	2024-08-01 00:10:00+09:00	
5	8101 Rx Rate Avg	2024-08-01 00:15:00+09:00	
6	8101 Rx Rate Avg	2024-08-01 00:20:00+09:00	
7	8101 Rx Rate Avg	2024-08-01 00:25:00+09:00	
8	8101 Rx Rate Avg	2024-08-01 00:30:00+09:00	
9	8101 Rx Rate Avg	2024-08-01 00:35:00+09:00	
10	8101 Rx Rate Avg	2024-08-01 00:40:00+09:00	
11	8101 Rx Rate Avg	2024-08-01 00:45:00+09:00	
12	8101 Rx Rate Avg	2024-08-01 00:50:00+09:00	
13	8101 Rx Rate Avg	2024-08-01 00:55:00+09:00	
14	8101 Rx Rate Avg	2024-08-01 01:00:00+09:00	
15	8101 Rx Rate Avg	2024-08-01 01:05:00+09:00	
16	8101 Rx Rate Avg	2024-08-01 01:10:00+09:00	
17	8101 Rx Rate Avg	2024-08-01 01:15:00+09:00	
18	8101 Rx Rate Avg	2024-08-01 01:20:00+09:00	
19	8101 Rx Rate Avg	2024-08-01 01:25:00+09:00	
20	8101 Rx Rate Avg	2024-08-01 01:30:00+09:00	
21	8101 Rx Rate Avg	2024-08-01 01:35:00+09:00	
22	8101 Rx Rate Avg	2024-08-01 01:40:00+09:00	
23	8101 Rx Rate Avg	2024-08-01 01:45:00+09:00	
24	8101 Rx Rate Avg	2024-08-01 01:50:00+09:00	
25	8101 Rx Rate Avg	2024-08-01 01:55:00+09:00	
26	8101 Rx Rate Avg	2024-08-01 02:00:00+09:00	
27	8101 Rx Rate Avg	2024-08-01 02:05:00+09:00	
28	8101 Rx Rate Avg	2024-08-01 02:10:00+09:00	
29	8101 Rx Rate Avg	2024-08-01 02:15:00+09:00	
30	8101 Rx Rate Avg	2024-08-01 02:20:00+09:00	
31	8101 Rx Rate Avg	2024-08-01 02:25:00+09:00	
32	8101 Rx Rate Avg	2024-08-01 02:30:00+09:00	
33	8101 Rx Rate Avg	2024-08-01 02:35:00+09:00	
34	8101 Rx Rate Avg	2024-08-01 02:40:00+09:00	53,08571429

(c) Data RSSI(dBm) avg (kiri); (d) Data Latency (ms) avg (tengah) ; (e) Data Rate avg (kanan)



	A	B	C
1	Name	Timestamp	Value
2	8101 SNR Avg	2024-08-07 00:00:00+09:00	
3	8101 SNR Avg	2024-08-07 00:05:00+09:00	
4	8101 SNR Avg	2024-08-07 00:10:00+09:00	
5	8101 SNR Avg	2024-08-07 00:15:00+09:00	
6	8101 SNR Avg	2024-08-07 00:20:00+09:00	
7	8101 SNR Avg	2024-08-07 00:25:00+09:00	
8	8101 SNR Avg	2024-08-07 00:30:00+09:00	
9	8101 SNR Avg	2024-08-07 00:35:00+09:00	
10	8101 SNR Avg	2024-08-07 00:40:00+09:00	
11	8101 SNR Avg	2024-08-07 00:45:00+09:00	
12	8101 SNR Avg	2024-08-07 00:50:00+09:00	
13	8101 SNR Avg	2024-08-07 00:55:00+09:00	
14	8101 SNR Avg	2024-08-07 01:00:00+09:00	
15	8101 SNR Avg	2024-08-07 01:05:00+09:00	
16	8101 SNR Avg	2024-08-07 01:10:00+09:00	
17	8101 SNR Avg	2024-08-07 01:15:00+09:00	
18	8101 SNR Avg	2024-08-07 01:20:00+09:00	
19	8101 SNR Avg	2024-08-07 01:25:00+09:00	
20	8101 SNR Avg	2024-08-07 01:30:00+09:00	
21	8101 SNR Avg	2024-08-07 01:35:00+09:00	
22	8101 SNR Avg	2024-08-07 01:40:00+09:00	
23	8101 SNR Avg	2024-08-07 01:45:00+09:00	
24	8101 SNR Avg	2024-08-07 01:50:00+09:00	
25	8101 SNR Avg	2024-08-07 01:55:00+09:00	
26	8101 SNR Avg	2024-08-07 02:00:00+09:00	
27	8101 SNR Avg	2024-08-07 02:05:00+09:00	
28	8101 SNR Avg	2024-08-07 02:10:00+09:00	
29	8101 SNR Avg	2024-08-07 02:15:00+09:00	
30	8101 SNR Avg	2024-08-07 02:20:00+09:00	
31	8101 SNR Avg	2024-08-07 02:25:00+09:00	
32	8101 SNR Avg	2024-08-07 02:30:00+09:00	
33	8101 SNR Avg	2024-08-07 02:35:00+09:00	
34	8101 SNR Avg	2024-08-07 02:40:00+09:00	43,86486



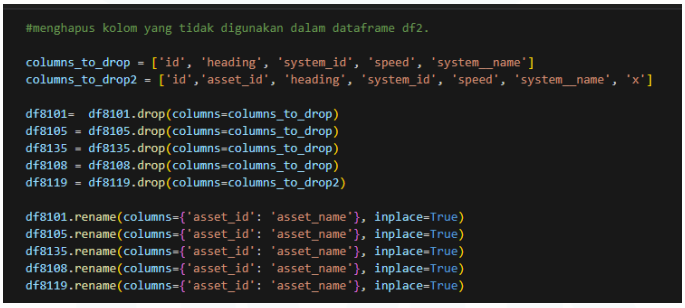
	A	B	C
1	Name	Timestam	Value
2	8101 Connectivity Avg	2024-08-0	
3	8101 Connectivity Avg	2024-08-0	
4	8101 Connectivity Avg	2024-08-0	
5	8101 Connectivity Avg	2024-08-0	
6	8101 Connectivity Avg	2024-08-0	
7	8101 Connectivity Avg	2024-08-0	
8	8101 Connectivity Avg	2024-08-0	
9	8101 Connectivity Avg	2024-08-0	
10	8101 Connectivity Avg	2024-08-0	
11	8101 Connectivity Avg	2024-08-0	
12	8101 Connectivity Avg	2024-08-0	
13	8101 Connectivity Avg	2024-08-0	
14	8101 Connectivity Avg	2024-08-0	
15	8101 Connectivity Avg	2024-08-0	
16	8101 Connectivity Avg	2024-08-0	
17	8101 Connectivity Avg	2024-08-0	
18	8101 Connectivity Avg	2024-08-0	
19	8101 Connectivity Avg	2024-08-0	
20	8101 Connectivity Avg	2024-08-0	
21	8101 Connectivity Avg	2024-08-0	
22	8101 Connectivity Avg	2024-08-0	
23	8101 Connectivity Avg	2024-08-0	
24	8101 Connectivity Avg	2024-08-0	
25	8101 Connectivity Avg	2024-08-0	
26	8101 Connectivity Avg	2024-08-0	
27	8101 Connectivity Avg	2024-08-0	
28	8101 Connectivity Avg	2024-08-0	
29	8101 Connectivity Avg	2024-08-0	
30	8101 Connectivity Avg	2024-08-0	
31	8101 Connectivity Avg	2024-08-0	
32	8101 Connectivity Avg	2024-08-0	
33	8101 Connectivity Avg	2024-08-0	
34	8101 Connectivity Avg	2024-08-0	

(f) Data SNR (dB) avg (g) Data Connectivity (%)

Gambar 3. 4 Tampilan data setelah di import dari FTP

Tabel 3. 3 Indikator Performa Jaringan dari Cisco[5]

Parameter	level		
	Optimal	**Average**	**Bad**
RSSI	-30dBm sampai -67 dBm	-67 dBm sampai -75 dBm	< -75 dBm
SNR	> 30 dBm	20 dB sampai 30 dB	< 20 dBm
Rate	> 54 Mbps	20 Mbps sampai 54 Mbps	< 20 Mbps
Latency	< 10 ms	10 ms sampai 30 ms	> 30 ms



```
#menghapus kolom yang tidak digunakan dalam dataframe df2.

columns_to_drop = ['id', 'heading', 'system_id', 'speed', 'system__name']
columns_to_drop2 = ['id','asset_id', 'heading', 'system_id', 'speed', 'system__name', 'x']

df8101=  df8101.drop(columns=columns_to_drop)
df8105 = df8105.drop(columns=columns_to_drop)
df8135 = df8135.drop(columns=columns_to_drop)
df8108 = df8108.drop(columns=columns_to_drop)
df8119 = df8119.drop(columns=columns_to_drop2)

df8101.rename(columns={'asset_id': 'asset_name'}, inplace=True)
df8105.rename(columns={'asset_id': 'asset_name'}, inplace=True)
df8135.rename(columns={'asset_id': 'asset_name'}, inplace=True)
df8108.rename(columns={'asset_id': 'asset_name'}, inplace=True)
df8119.rename(columns={'asset_id': 'asset_name'}, inplace=True)
```

Gambar 3. 5 Kode untuk menghapus kolom yang tidak digunakan

```
#pisahkan antara data longitude dan latitude dari database agar menjadi kolom terpisah
df01 [['latitude of CAP (DD)','longitude of CAP(DD)']]= df01['Location of CAP'].str.split(',', expand=True)
df05 [['latitude of CAP (DD)','longitude of CAP(DD)']]= df05['Location of CAP'].str.split(',', expand=True)
df35 [['latitude of CAP (DD)','longitude of CAP(DD)']]= df35['Location of CAP'].str.split(',', expand=True)
df08 [['latitude of CAP (DD)','longitude of CAP(DD)']]= df08['Location of CAP'].str.split(',', expand=True)
df19 [['latitude of CAP (DD)','longitude of CAP(DD)']]= df19['Location of CAP'].str.split(',', expand=True)
```

Gambar 3. 6 Kode untuk membagi satu kolom menjadi dua kolom

```
df01['speed-lhd movement(km/h)'] = pd.to_numeric(df01['speed-lhd movement(km/h)'], errors='coerce')
df01['distance-lhd-movement (km)'] = pd.to_numeric(df01['distance-lhd-movement (km)'], errors='coerce')



df05['speed-lhd movement(km/h)'] = pd.to_numeric(df05['speed-lhd movement(km/h)'], errors='coerce')
df05['distance-lhd-movement (km)'] = pd.to_numeric(df05['distance-lhd-movement (km)'], errors='coerce')

df08['speed-lhd movement(km/h)'] = pd.to_numeric(df08['speed-lhd movement(km/h)'], errors='coerce')
df08['distance-lhd-movement (km)'] = pd.to_numeric(df08['distance-lhd-movement (km)'], errors='coerce')

df19['speed-lhd movement(km/h)'] = pd.to_numeric(df19['speed-lhd movement(km/h)'], errors='coerce')
df19['distance-lhd-movement (km)'] = pd.to_numeric(df19['distance-lhd-movement (km)'], errors='coerce')

# Convert relevant columns to numeric types (if applicable)
df35['speed-lhd movement(km/h)'] = pd.to_numeric(df35['speed-lhd movement(km/h)'], errors='coerce')
df35['distance-lhd-movement (km)'] = pd.to_numeric(df35['distance-lhd-movement (km)'], errors='coerce')
df35['latitude of CAP (DD)'] = pd.to_numeric(df35['latitude of CAP (DD)'], errors='coerce')
df35['longitude of CAP(DD)'] = pd.to_numeric(df35['longitude of CAP(DD)'], errors='coerce')
```

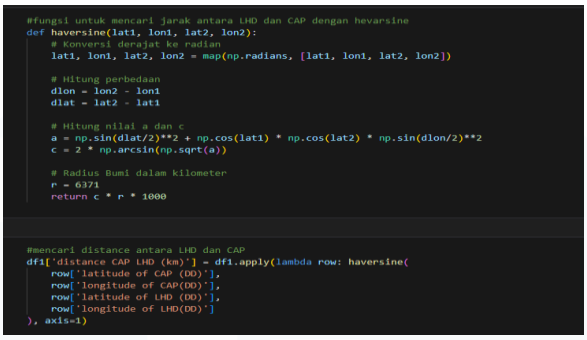
Gambar 3. 7 Kode untuk mengubah tipe data dari object ke numerik

```
df8101['timestamp'] = pd.to_datetime(df8101['timestamp'], errors = 'coerce')
df8101.set_index('timestamp', inplace=True)

df8105['timestamp'] = pd.to_datetime(df8105['timestamp'], errors = 'coerce')
df8105.set_index('timestamp', inplace=True)


df8135['timestamp'] = pd.to_datetime(df8135['timestamp'], errors = 'coerce')
df8135.set_index('timestamp', inplace=True)


df8108['timestamp'] = pd.to_datetime(df8108['timestamp'], errors = 'coerce')
df8108.set_index('timestamp', inplace=True)


df8119['timestamp'] = pd.to_datetime(df8119['timestamp'], errors = 'coerce')
df8119.set_index('timestamp', inplace=True)
```

```
df8101 = df8101.resample('5min').agg({
    'latitude of LHD (DD)': 'first' ,
    'longitude of LHD(DD)':'first' ,
    'altitude of LHD(m)': 'first',
    'asset__name': 'first'
})
df8101 = df8101[df8101.index.minute % 5 == 0]



df8105 = df8105.resample('5min').agg({
'latitude of LHD (DD)': 'first' ,
    'longitude of LHD(DD)':'first' ,
    'altitude of LHD(m)': 'first',
    'asset__name': 'first'
})
df8105 = df8105[df8105.index.minute % 5 == 0]


df8135 = df8135.resample('5min').agg({
'latitude of LHD (DD)': 'first' ,
    'longitude of LHD(DD)':'first' ,
    'altitude of LHD(m)': 'first',
    'asset__name': 'first'
})
df8135 = df8135[df8135.index.minute % 5 == 0]
```

(a) (b)

Gambar 3. 8 Preprocessing kode - (a) Mengubah tipe data kolom timestamp menjadi datetime; (b) Resampling data menjadi per 5 menit.



Gambar 3. 9 kode untuk mencari jarak antara *loader* dan CAP dengan rumus *haversine*

```
#standarisasi dataset
scaler = RobustScaler()
features_scaled = scaler.fit_transform(df)
```

Gambar 3. 10 Inisialisasi scaler dengan Robust Scaler

```
# Menggunakan DataFrame df yang sudah ada
features = ['altitude of LHD(m)', 'speed-lhd movement(m/s) avg', 'distance CAP LHD (m)']
target = 'SNR Avg (dB)'  # Fokus pada target SNR
```

Gambar 3. 11 Menentukan features dan target

```
# Membagi data menjadi training dan testing dengan perbandingan 70:30
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

Gambar 3. 12 Data Split menjadi data train dan data test

```
# Inisialisasi model regresi Random Forest
rf_model = RandomForestRegressor(random_state=42)
```

Gambar 3. 13 Inisialisasi model

```python
# Hyperparameter tuning untuk Random Forest
rf_param_grid = {
    'n_estimators': [50, 100, 200],
    'max_features': ['auto', 'sqrt'],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10]
}

# Melakukan Grid Search untuk menemukan hyperparameter terbaik
grid_search = GridSearchCV(rf_model, rf_param_grid, cv=5, scoring='r2')
grid_search.fit(X_train_scaled, y_train)
best_rf_model = grid_search.best_estimator_
```

Gambar 3. 14 Hyperparameter Tuning

```python
# Mencari rentang optimal untuk Connectivity Avg antara 90% dan 100%
optimal_conditions = df[(df[target] >= 90) & (df[target] <= 100)][features]
print("\nRentang Optimal untuk Connectivity Avg antara 90% dan 100%:")
print(optimal_conditions.describe())
```

Gambar 3. 15 Mencari rentang optimal dengan Boolean condition

- **Analisis dan Hasil Project Magang**

Tabel 3. 4 Analisis dan Hasil Project Magang

No	Analisis prediktif	Hasil yang diperoleh
1	Performa model	Hasil antara skenario 1 dan skenario 2 memiliki perbedaan yang cukup signifikan, kemungkinan karena pola data pada setiap loader berbeda sehingga model sulit untuk menangkap pola yang relevan ketika training pada dataset gabungan. Selain itu, model cenderung 'belajar' pola spesifik pada masing-masing dataset loader, jadi ketika training model menggunakan data gabungan kelima

		loader, model mungkin tidak dapat menangkap pola umum yang tidak berlaku untuk semua loader. Pola yang berbeda pada masing-masing dataset loader kemungkinan karena kondisi setiap hari pada loader berbeda-beda, Misalnya status *standby*, *productive time*, dan *unplanned down* dari tiap loader yang berbeda setiap harinya. Berdasarkan hasil training dan testing model yang telah dilakukan dalam 2 skenario, diperoleh bahwa skenario 2 dengan dataset loader 8101 yang memiliki nilai R2 score paling baik diantara skenario lainnya. Hasil R2 score untuk skenario 2 dengan dataset 8101 bisa dilihat pada Tabel 3.6. Oleh karena itu, untuk prediksi dan mencari rentang optimal antara loader dan CAP dilakukan dengan hasil training model menggunakan dataset loader 8101. Berdasarkan hasil R2 score, model paling baik dalam memprediksi parameter SNR avg (dB) dengan nilai 0.81 atau 81% diikuti dengan RSSI avg (dBm), SNR avg (dB), Tx rate avg (Mbit/s), dan Rx rate avg (Mbit/s), masuk dalam kategori 'Very Strong'. Sedangkan untuk Connectivity avg

		(%) dan Latency (ms) memiliki nilai R2 paling kecil masuk dalam kategori '*Fair*' dan '*Weak*', Hal ini dapat terjadi mungkin karena rentang data untuk atribut Connectivity dan Latency cenderung memiliki perbedaan nilai min dan max dalam data yang cukup besar.
2	Perbandingan Aktual dan Prediksi	Berdasarkan dari hasil perbandingan antara data aktual dan data prediksi, model bisa dan sangat baik dalam memprediksi parameter SNR avg, RSSI avg, Rx rate dan Tx rate. Sedangkan untuk Connectivity avg, model sebenarnya dapat memprediksi cukup baik, namun sulit untuk memprediksi ketika memprediksi Connectivity avg 100%. Hasil dari perbandingan aktual dan prediksi bisa dilihat pada Tabel 3.7
3	Mencari rentang optimal	Berdasarkan hasil preediksi lalu dilakukan *boolean indexing* atau *conditional filtering* untuk setiap *target* terhadap *features* lihat hasilnya pada Tabel 3.8. Diperoleh jarak minimum antara loader dan CAP yaitu 35 meter dan maximumnya adalah 67.81 meter. Selain itu, rata-rata perpindahan loader harus sekitar 0.39 m/s sampai 1.46 m/s agar

		Connectivity, Latency, RSSI, SNR, Tx rate, dan RX rate tetap stabil ketika loader melakukan *roaming*. Rangkuman rentang optimal *features* lihat Tabel 3.9

Skenario 1

Pada skenario ini, training dan testing model dilakukan pada dataset yang merupakan gabungan dari lima loader.

Dengan dataset: datacombine (loader 8101, 8105, 8108, 8119, dan 8135)

Tabel 3. 5 Evaluasi metrik hasil training skenario 1

Target	R2 Score
Connectivity avg (%)	0.19
Latency Avg (ms)	0.18
RSSI avg (dBm)	0.48
SNR avg (db)	0.48
Tx rate avg (Mbit/s)	0.54
Rx rate avg (Mbit/s)	0.51

Skenario 2

Pada skenario ini, training dan testing model dilakukan pada dataset per loader.

Dengan dataset: loader 8101

Tabel 3. 6 Evaluasi metrik hasil training skenario 2

Target	R2 Score
Connectivity avg (%)	**0.32**
Latency Avg (ms)	**0.19**
RSSI avg (dBm)	**0.80**
SNR avg (db)	**0.81**
Tx rate avg (Mbit/s)	**0.78**
Rx rate avg (Mbit/s)	**0.76**

Tabel 3. 7 Perbandingan dataset actual vs prediksi

No	*Target*	Perbandingan Aktual vs Prediksi
1	Connectivity Avg (%)	Hasil pada target Connectivity Avg (%): Actual Predicted 234 96.093750 85.632511 110 100.000000 41.164547 249 95.795796 86.412550 9 94.225146 84.794836 93 100.000000 43.110716
2	Latency Avg (ms)	Hasil pada target Latency Avg (ms): Actual Predicted 234 17.874821 26.596071 110 9.889771 54.039816 249 15.568574 28.461034 9 4.917835 11.882929 93 2.943506 3.101872
3	RSSI Avg (dBm)	Hasil pada target RSSI Avg (dBm): Actual Predicted 234 -63.868421 -64.231396 110 -58.270270 -67.230706 249 -64.594595 -66.243753 9 -48.783784 -54.082438 93 -43.142857 -44.642280

4	SNR Avg (dB)	Hasil pada target SNR Avg (dB): Actual Predicted 234 31.131579 31.035205 110 36.729730 28.567725 249 30.405405 29.125337 9 46.216216 40.494331 93 52.621622 51.301287
5	Rx Rate Avg (Mbit/s)	Hasil pada target Rx Rate Avg (Mbit/s): Actual Predicted 234 77.473684 75.259017 110 67.189189 66.043315 249 73.864865 61.725483 9 93.270270 106.412534 93 143.621622 128.601564
6	Tx Rate Avg (Mbt/s)	Hasil pada target Tx Rate Avg (Mbit/s): Actual Predicted 234 62.605263 66.317616 110 61.000000 56.584624 249 63.324324 57.534765 9 61.378378 70.321506 93 114.594595 100.194921

Tabel 3. 8 Hasil *Boolean Indexing* setiap target

No	***Target***	**Hasil *Boolean Indexing***
1	Connectivity Avg (%)	Rentang Optimal untuk Connectivity Avg antara 90% dan 100%: altitude of LHD(m) speed-lhd movement(m/s) avg distance CAP LHD (m) count 110.000000 110.000000 110.000000 mean 2825.558636 1.464585 67.806408 std 1.682258 0.908513 42.434702 min 2820.000000 0.004033 3.174499 25% 2824.500000 0.589956 32.056814 50% 2825.500000 1.675502 62.563335 75% 2826.500000 2.264457 95.915135 max 2829.800000 3.096027 176.987866 • Altitude of LHD (m): 2825.55 m. • Speed-lhd movement (m/s): 1.46 m/s. • Distance CAP-LHD (m): 67.81 m.
2	Latency avg (ms)	Rentang Optimal untuk Latency Avg < 10 ms: altitude of LHD(m) speed-lhd movement(m/s) avg distance CAP LHD (m) count 153.000000 153.000000 153.000000 mean 2826.680065 0.390705 35.156154 std 2.232124 0.653011 28.704371 min 2819.000000 0.031254 4.618807 25% 2825.500000 0.078467 17.450746 50% 2826.500000 0.120400 30.849908 75% 2829.000000 0.222997 34.113377 max 2829.800000 3.096027 176.987866 • Altitude of LHD (m): 2826.03 m.

<table>
<tr><td></td><td></td><td>

- Speed-lhd movement (m/s): 0.39 m/s.
- Distance CAP-LHD (m): 35.15 m.

</td></tr>
<tr><td>3</td><td>RSSI avg (dBm)</td><td>

Rentang Optimal untuk RSSI Avg antara -30 dBm dan -67 dBm:

	altitude of LHD(m)	speed-lhd movement(m/s) avg	distance CAP LHD (m)
count	234.000000	234.000000	234.000000
mean	2826.222436	0.861806	46.551292
std	2.176533	0.903510	38.624128
min	2819.000000	0.004033	1.076432
25%	2824.812500	0.107679	19.436380
50%	2826.250000	0.326931	31.867500
75%	2827.750000	1.619760	59.348726
max	2829.800000	3.096027	176.987866

- Altitude of LHD (m): 2826.22 m.
- Speed-lhd movement (m/s): 0.86 m/s.
- Distance CAP-LHD (m): 46.55 m

</td></tr>
<tr><td>4</td><td>SNR avg (dB)</td><td>

Rentang Optimal untuk SNR Avg > 30 dB:

	altitude of LHD(m)	speed-lhd movement(m/s) avg	distance CAP LHD (m)
count	208.000000	208.000000	208.000000
mean	2826.394471	0.747492	41.009423
std	2.167086	0.866272	35.073704
min	2820.000000	0.004033	1.076432
25%	2825.250000	0.097953	17.497406
50%	2826.500000	0.164618	31.657981
75%	2827.750000	1.283931	45.890141
max	2829.800000	3.096027	176.987866

- Altitude of LHD (m): 2826.39 m.
- Speed-lhd movement (m/s): 0.75 m/s.
- Distance CAP-LHD (m): 41 m

</td></tr>
<tr><td>5</td><td>Rx rate avg (Mbit/s)</td><td>

Rentang Optimal untuk Rx Rate Avg > 54 Mbit/s:

	altitude of LHD(m)	speed-lhd movement(m/s) avg	distance CAP LHD (m)
count	286.000000	286.000000	286.000000
mean	2826.040210	0.947023	56.776309
std	2.059558	0.911862	44.358145
min	2819.000000	0.004033	1.076432
25%	2824.500000	0.114649	24.128124
50%	2826.000000	0.598821	34.113377
75%	2827.500000	1.720620	91.176959
max	2829.800000	3.096027	176.987866

- Altitude of LHD (m): **2826.04 m.**
- Speed-lhd movement (m/s): **0.94 m/s**.
- Distance CAP-LHD (m): **56.77 m**

</td></tr>
<tr><td>6</td><td>Tx rate avg (Mbit/s)</td><td>

Rentang Optimal untuk Tx Rate Avg > 54 Mbit/s:

	altitude of LHD(m)	speed-lhd movement(m/s) avg	distance CAP LHD (m)
count	251.000000	251.000000	251.000000
mean	2826.127291	0.899896	55.216771
std	2.110100	0.913923	44.377626
min	2820.000000	0.004033	1.076432
25%	2824.700000	0.112855	23.142400
50%	2826.000000	0.449891	34.001232
75%	2827.500000	1.696280	92.938695
max	2829.800000	3.096027	176.987866

- Altitude of LHD (m): **2826.12 m.**
- Speed-lhd movement (m/s): **0.89 m/s**.
- Distance CAP-LHD (m): **55.21 m**

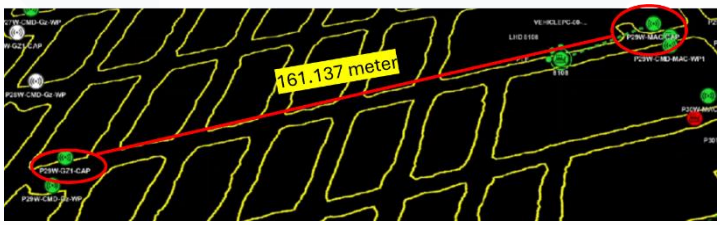
</td></tr>
</table>

Tabel 3. 9 Rangkuman range optimal

Features	Range Optimal
Distance lhd-cap (m)	**35 m - 67.81 m**
Altitude of LHD (m)	**3835.55 m - 2826.30 m**
Speed LHD movement (m/s)	**0.39 m/s - 1.46 m/s**

- **Perbandingan kondisi eksisting vs hasil prediksi**

  Jika dibandingkan dengan existing kondisi pada panel GRSPCN-GBCEL-P29W-GZ-AP1 ke GRSPCN-GBCEL-MAC-AP1 jarak antar panel adalah 0.161357 km atau 161.137 meter lihat Gambar 3.16.



Gambar 3. 16 Panel P29W dan MAC di Underground Mine GBC

Jika dilihat pada panel GRSPCN-GBCEL-P29W-GZ-AP1 dan GRSPCN-GBCEL-MAC-AP1 jarak cukup jauh antara keduanya, sehingga ketika loader disekitar kedua panel tersebut melakukan *roaming* tentu mengalami konektivitas yang tidak stabil.

3.3 Kendala yang Ditemukan

Kendala yang dihadapi selama melakukan project prediksi rentang optimal antara *loader* dan Cisco Access Points (CAP) di *underground mine* ini adalah:

1. Mengumpulkan data: Dalam proses pengumpulan data penulis mengalami kendala yaitu kesulitan dalam mengumpulkan data untuk beberapa parameter *target* yaitu RSSI avg, SNR avg, Connectivity avg. Latency Avg, dan Rate Avg. Hal ini karena, akses akun FTP yang diberikan kepada penulis hanya *read-only* sehingga jika ingin menarik data dari website untuk per 5 menit tidak bisa dilakukan dari akun penulis.
2. Data kurang lengkap: Pada projet ini, penulis juga kesulitan dalam mengumpulkan data jarak antara *loader* dan CAP, karena di FTP tidak ada data tentang jarak antara CAP secara langsung, Namun hanya terdapat data *latitude* dan *longitude* dari kedua titik baik *loader* maupun CAP. Namun hal tersebut harus dilakukan secara manual untuk mengumpulkan data sehingga cukup memakan waktu.
3. Pemilihan model: pertama dalam project ini penulis menggunakan model *k-means clustering* , namun performanya kurang baik.

3.4 Solusi atas Kendala yang Ditemukan

Adapun solusi untuk menghadapi kendala yang ditemukan antara lain:

1. Meminta tolong bantuan dari rekan kantor yang memiliki akses bukan *read-only* sehingga bisa menarik data dari FTP untuk parameter-parameter yang dibutuhkan penulis.
2. Mencari jarak menggunakan data *latitude* dan *longitude.* Kemudian, menggunakan rumus *haversine* untuk mencari jarak antara dua titik, dalam hal ini jarak antara *loader* dan CAP.

3. Saya membandingkan beberapa model regresi, termasuk regresi linier, regresi ridge, dan regresi hutan acak. Model terbaik yang penulis temukan adalah model *random forest regression*