

# PREDIKSI CURAH HUJAN BERDASARKAN DATA CUACA HARIAN DAN DATA CUACA PER JAM MENGGUNAKAN ALGORITMA CATBOOST

Tim Biomatika

Ang, Johan Nicholas & Habiburrohman

---

## 1. Pendahuluan

Laporan ini dibuat untuk memenuhi salah satu komponen penilaian pada babak penyisihan cabang kompetisi Datavidia pada Arkavidia 2023. Analisis dan pemodelan dilakukan dengan menggunakan data cuaca yang disediakan di laman Kaggle oleh panitia Datavidia 2023 ([Weather Forecasting - Datavidia | Kaggle](#)). Seluruh alur analisis dan pemodelan dilakukan melalui media Google Collaboratory menggunakan bahasa pemrograman Python beserta *modules* dan *libraries* yang tersedia di dalamnya.

## 2. Preprocessing

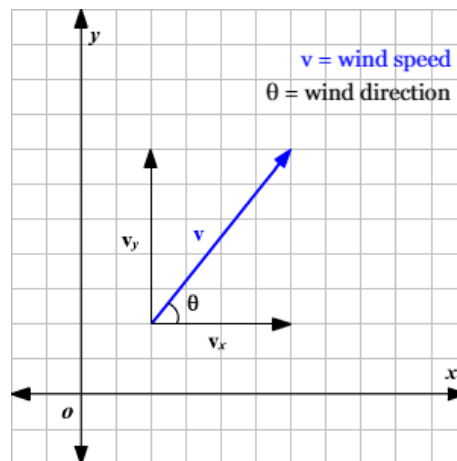
*Dataset* terdiri dari empat file, yakni `train.csv`, `train_hourly.csv`, `test.csv`, dan `test_hourly.csv`. Untuk data latih, `train.csv` memiliki 13258 baris dan 16 fitur, sementara `train_hourly.csv` memiliki 318192 baris dan 32 fitur. Untuk data uji, `test.csv` memiliki 4972 baris dan 16 fitur, sementara `test_hourly.csv` memiliki 119328 baris dan 32 fitur.

Pertama-tama, dilakukan pemeriksaan *missing values* pada data latih. Ditemukan bahwa fitur target ‘rain\_sum (mm)’ mengandung *missing values* sebanyak 60 baris. Fitur target berperan sebagai variabel dependen yang membedakan data latih dan data uji. Oleh karena itu, *missing values* pada fitur target dihapus.

Melalui observasi lebih lanjut dari fitur ‘time’, diperoleh baris data di `train_hourly.csv` yang berkorespondensi dengan *missing values* pada fitur target di `train.csv`, yakni mulai tanggal 23-12-2022 dan seterusnya. Dengan demikian, baris yang berkorespondensi tersebut dihapus pula.

Pada fitur ‘winddirection\_10m\_dominant (°)’ terdapat *missing values* sebanyak 406 baris. Imputasi rerata tidak bisa dilakukan secara langsung karena satuannya berupa derajat yang siklusnya berulang setiap 360 kali. Oleh karena itu, dilibatkan vektor supaya hasil imputasi tetap valid.

*Missing values* fitur tersebut diisi dengan rerata arah resultan vektor dari baris tak kosong yang dikelompokkan berdasarkan kota. Resultan vektor dapat dihitung jika diketahui arah vektor (*direction*) dan panjang vektor (*magnitude*). Pada kasus ini, fitur ‘windspeed\_10m\_dominant (°)’ dijadikan sebagai arah vektor, sedangkan fitur ‘windspeed\_10m\_max (km/h)’ dijadikan sebagai panjang vektor. Dengan penjumlahan vektor, diperoleh resultan vektor untuk setiap kota dan digunakan arah resultan untuk menghitung nilai rerata yang akan dijadikan sebagai nilai imputasi.

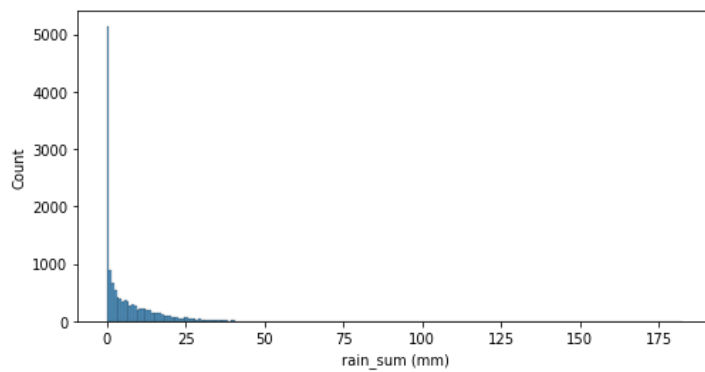


**Gambar 1.1 Ilustrasi Resultan Vektor Angin**

Pada `train_hourly.csv` terdapat *missing values* pada fitur ‘winddirection\_10m’ dan ‘winddirection\_100m (°)’. Dengan cara yang sama, *missing values* pada kedua fitur diimputasi dengan rerata arah resultan vektor dari baris data tak kosong yang dikelompokkan berdasarkan kota. Adapun arah dan panjang vektor untuk masing-masing kasus adalah ‘winddirection\_10m (°)’ dan ‘windspeed\_10m (km/h)’ serta ‘winddirection\_100m (°)’ dan ‘windspeed\_100m (km/h)’.

### 3. Exploratory Data Analysis

Pertama, dilakukan analisis fitur target ‘rain\_sum (mm)’ untuk mengetahui distribusi data.



**Gambar 2.1 Distribusi Fitur Target**

Dari grafik di atas, terlihat bahwa target tidak berdistribusi normal, melainkan memiliki kemencengan (*skewness*) yang sangat positif, yakni 2.996. Hal ini ditandai dengan data yang terpusat pada nilai yang kecil dan mean = 6.3 yang lebih besar daripada median = 2.2.

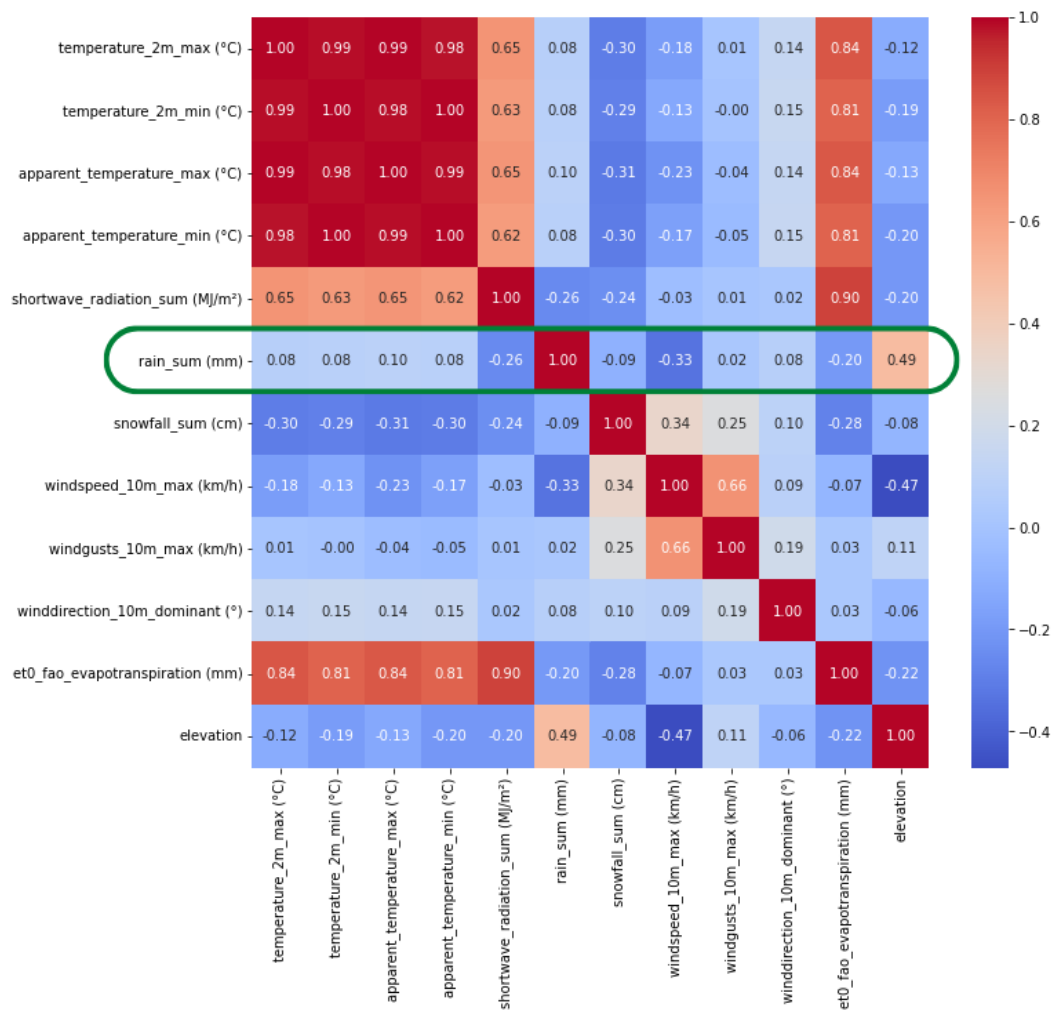
Hal ini sejalan dengan hasil analisis kuartil yang menunjukkan bahwa terdapat interval yang besar antara nilai Q3 dan maksimum. Dengan kata lain, terdapat pencilan dengan nilai yang jauh lebih besar dari nilai pusat persebaran.

min	0.000000
25%	0.000000
50%	2.200000
75%	9.300000
max	182.700000

**Gambar 2.2 Statistik Lima Serangkai Fitur Target**

Umumnya, pencilan bersifat buruk sebab model pembelajaran mesin tidak dapat bekerja dengan baik dengan keberadaan pencilan. Namun, jika memperhatikan konteks data, keberadaan pencilan masuk akal sebab pada faktanya sangat mungkin terjadi hujan deras atau ekstrem di waktu-waktu tertentu, khususnya saat musim hujan. Dengan kata lain, keberadaan pencilan pada fitur target bersifat alami dan tidak perlu dihapus.

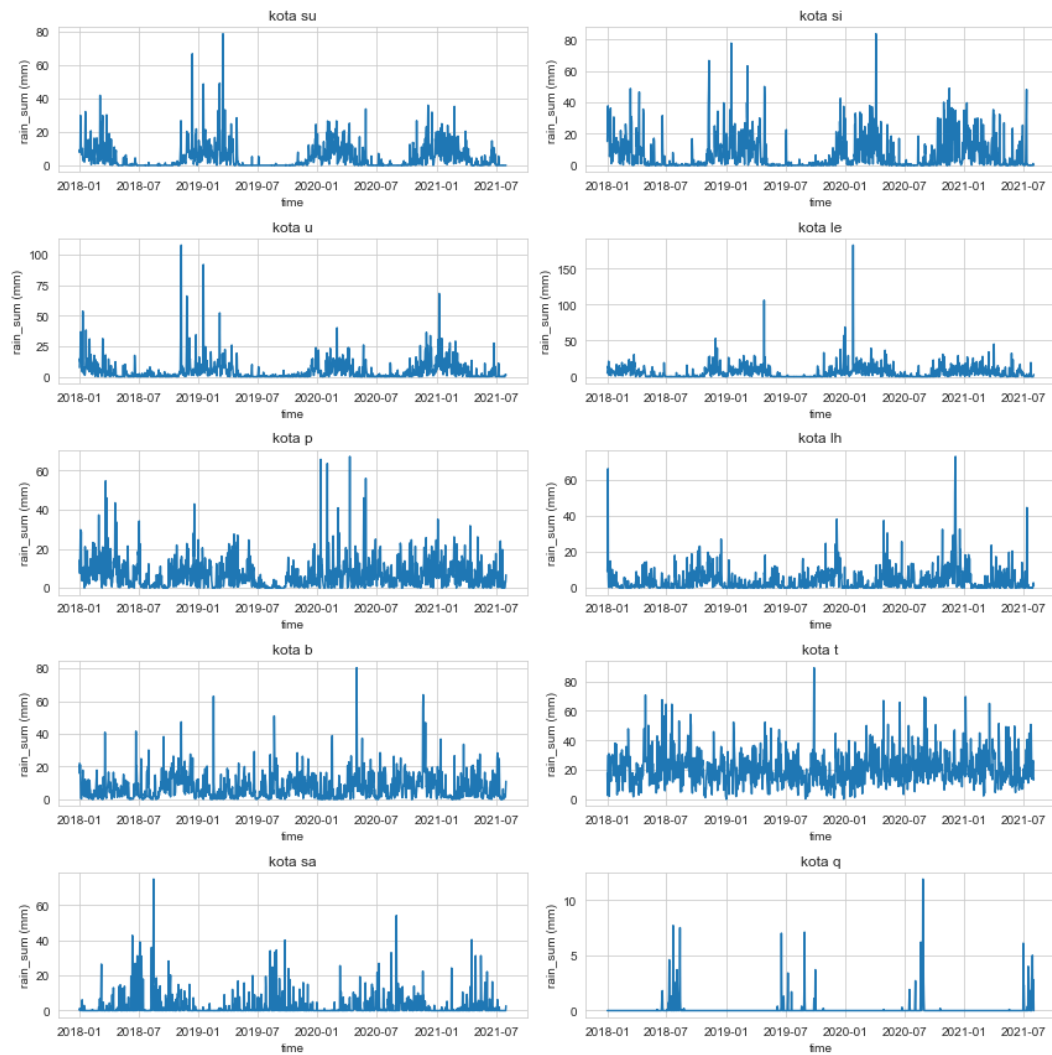
Selanjutnya, dilakukan analisis korelasi untuk mengetahui hubungan linear antar fitur pada *dataset*, khususnya fitur target 'rain\_sum (mm)'.



**Gambar 2.3 Korelasi Antar Fitur**

Berdasarkan Gambar 2.3, dapat dilihat bahwa korelasi antara fitur independen dan dependen yang paling tinggi adalah 0.49 yang termasuk lemah. Selain itu, terdapat beberapa *multicollinearity* atau korelasi kuat antara fitur-fitur independen. Dengan demikian, disimpulkan bahwa penggunaan model linear tidak cocok untuk memprediksi curah hujan dari *dataset* ini.

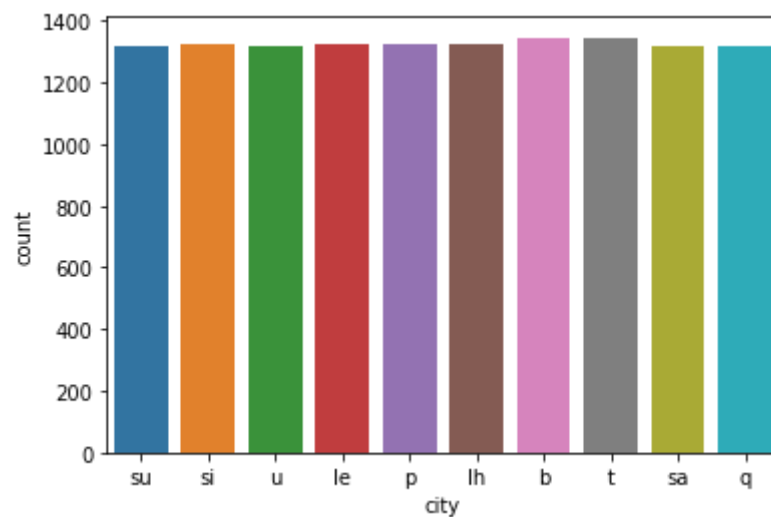
Kemudian, dilakukan analisis *time series* fitur target berdasarkan masing-masing kota untuk melihat keterkaitan pola hujan terhadap waktu. Hipotesis awal yakni curah hujan dipengaruhi oleh musim sehingga ada bulan-bulan tertentu dengan curah hujan tinggi. Sebaliknya, ada bulan-bulan tertentu pula dengan curah hujan rendah. Grafik yang memaparkan analisis dapat dilihat pada Gambar 2.4.



**Gambar 2.4 Grafik Time Series Curah Hujan Setiap Kota**

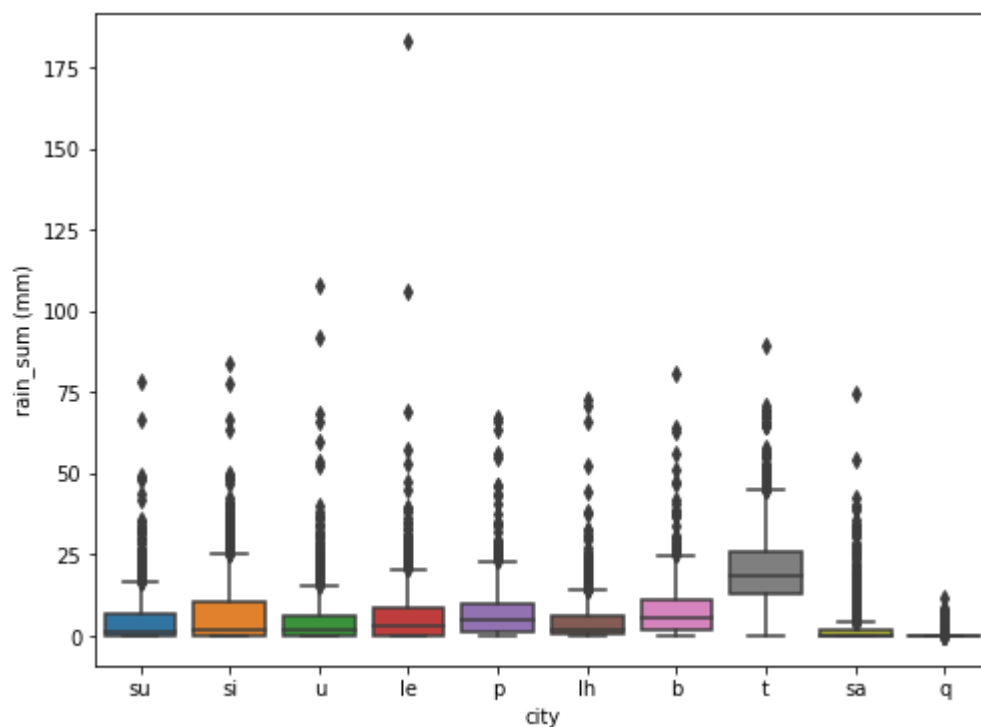
Untuk beberapa kota, seperti kota sa dan q, pola curah hujan terhadap waktu dapat terlihat cukup jelas. Namun, untuk beberapa kota lainnya, seperti kota b dan t, fluktuasi curah hujan sangat tinggi dan kurang terlihat polanya seolah-olah hujan terjadi sepanjang waktu. Dengan demikian, disimpulkan bahwa fitur waktu berpotensi bermanfaat dalam pemodelan untuk memprediksi curah hujan di kota-kota tertentu yang curah hujannya dipengaruhi oleh waktu.

Selanjutnya dilakukan analisis terhadap fitur categorical, yaitu ‘city’. Pertama, diperiksa terlebih dahulu frekuensi masing-masing kelas. Terlihat pada Gambar 2.5 bahwa masing-masing kelas pada fitur ‘city’ memiliki frekuensi yang serupa. Dengan demikian, tidak perlu dikhawatirkan terjadi ketimpangan atau *imbalance* kelas yang berdampak buruk pada model.



**Gambar 2.5 Grafik Time Series Curah Hujan Berdasarkan Tiap Kota**

Selanjutnya dilakukan analisis bivariat untuk mengetahui hubungan antara masing-masing kelas 'city' terhadap fitur target yang dapat dilihat pada Gambar 2.6.

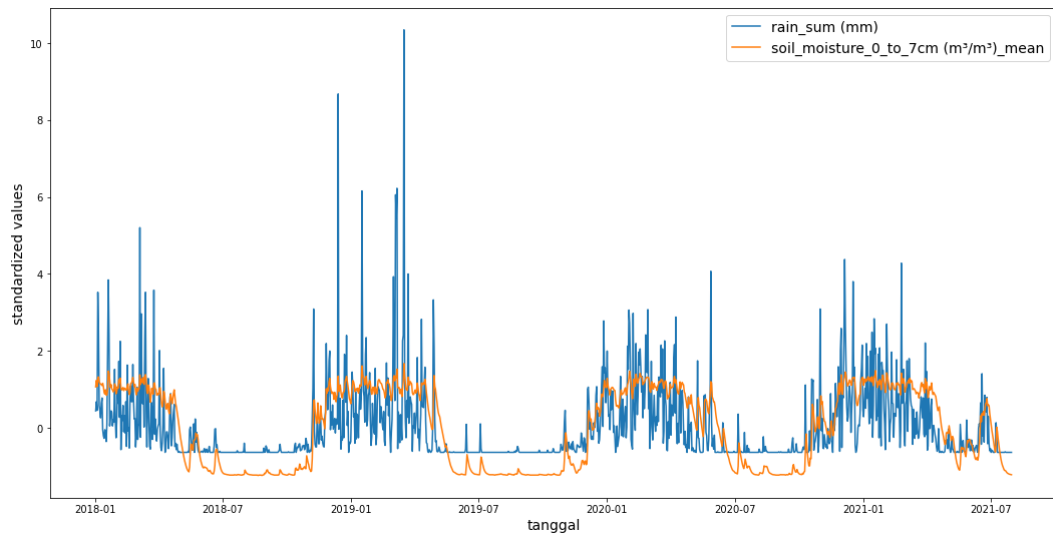


**Gambar 2.6 Grafik Boxplot Curah Hujan Berdasarkan Tiap Kota**

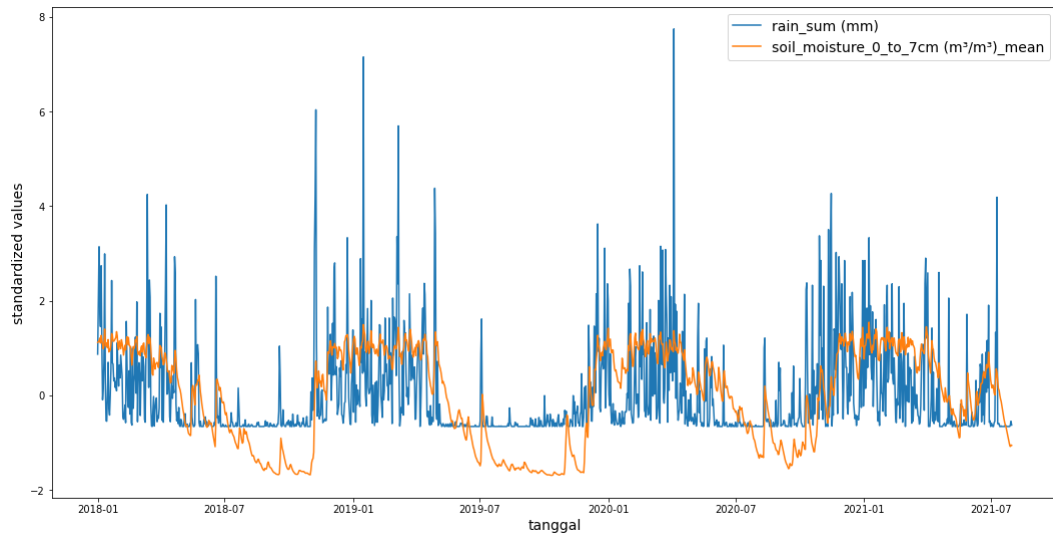
Dari grafik tersebut, disimpulkan bahwa kota t relatif memiliki curah hujan yang paling tinggi di antara seluruh kota. Sebaliknya, kota q relatif memiliki curah hujan yang paling rendah di antara seluruh kota. Melalui grafik tersebut,

dapat dilihat pula bahwa curah hujan memiliki banyak pencilan terlepas dari kota apapun itu.

Kemudian, dilakukan analisis bivariat antara fitur pada data per jam dan fitur target. Pada kasus ini, diambil nilai rerata fitur dari data per jam dalam melakukan *group by* data per jam menjadi data harian. Dari analisis ini, ditemukan beberapa fitur independen yang berpengaruh terhadap fitur target.



**Gambar 2.7 Grafik Time Series Fitur Terkait Kota si**



**Gambar 2.8 Grafik Time Series Fitur Terkait Kota su**

#### 4. Feature Engineering Tahap 1

Pertama, diciptakan fitur baru berupa durasi sinar matahari dalam satuan jam dari pengurangan fitur 'sunrise (iso8601)' dan 'sunset (iso8601)'.

**Tabel 3.1 Penciptaan Fitur ‘sun\_duration (hrs)’**

<b>sunrise (iso8601)</b>	<b>sunset (iso8601)</b>	<b>sun_duration (hrs)</b>
2018-01-01 05:15:00	2018-01-01 17:49:00	12.566667
2018-01-02 05:15:00	2018-01-02 17:50:00	12.583333
2018-01-03 05:16:00	2018-01-03 17:50:00	12.566667
2018-01-04 05:16:00	2018-01-04 17:50:00	12.566667
2018-01-05 05:17:00	2018-01-05 17:51:00	12.566667

Setelah ditelusuri lebih lanjut, ternyata terdapat beberapa data jam yang tidak valid pada kota q seperti Gambar 3.2. Hal ini menghasilkan durasi sebesar 0 jam yang jelas mustahil. Dengan demikian, nilai 0 jam diganti dengan nilai rerata durasi pada kota q sehingga diperoleh nilai yang lebih masuk akal.

**Tabel 3.2 Cuplikan Data Jam yang Tidak Valid pada Kota q**


<b>sunrise (iso8601)</b>	<b>sunset (iso8601)</b>	<b>city</b>
1970-01-01T07:00	1970-01-01T07:00	q
1970-01-01T07:00	1970-01-01T07:00	q
1970-01-01T07:00	1970-01-01T07:00	q
1970-01-01T07:00	1970-01-01T07:00	q
1970-01-01T07:00	1970-01-01T07:00	q

Kedua, dilakukan *ekstraksi* fitur ‘date’ untuk mendapatkan fitur bulan dan pekan (*week of month*). Hal ini dilakukan atas dasar hasil EDA yang menunjukkan bahwa waktu cenderung berpengaruh terhadap curah hujan. Selain itu, kondisi cuaca pada umumnya bersifat musiman atau *seasonal* yang berarti bergantung terhadap waktu.

Ketiga, dilakukan penggabungan *dataset* antara data cuaca harian dan data cuaca per jam sehingga diperoleh fitur-fitur yang lebih bervariasi. Hal ini dicapai dengan metode *group by* berdasarkan tanggal dan kota pada data cuaca per jam serta mengambil nilai *maximum*, *minimum*, *standard deviation* (std), dan *skewness* dari fitur-fitur independen.



	time	city	temperature_2m (°C)			
0	2018-01-01T00:00	su	25.0			
1	2018-01-01T01:00	su	25.2			
2	2018-01-01T02:00	su	24.9			
3	2018-01-01T03:00	su	25.1			
4	2018-01-01T04:00	su	24.8			

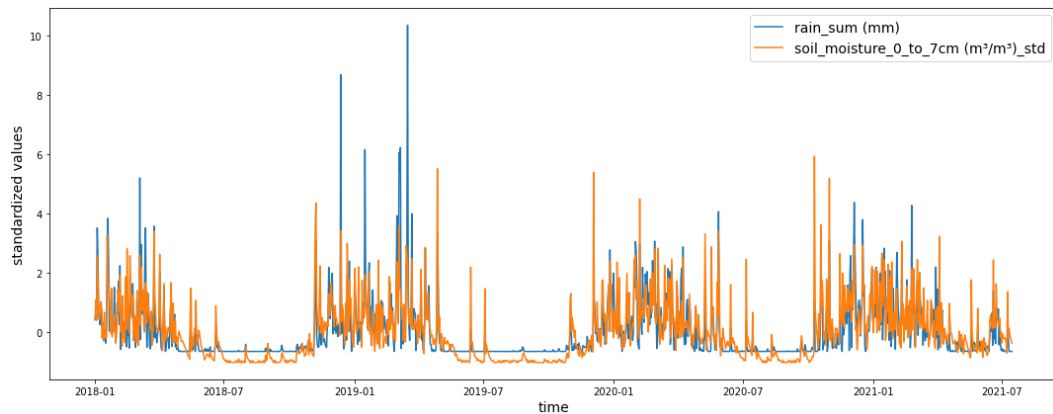
	date	city	temperature_2m (°C)_max	temperature_2m (°C)_min	temperature_2m (°C)_std	temperature_2m (°C)_skw
0	2018-01-01	su	25.2	25.0	0.158114	0.0

**Gambar 3.3 Ilustrasi Group By Data Per Jam serta Pengambilan Nilai Max, Min, Standard Deviation, dan Skewness**

Pengambilan nilai *max* dan *min* fitur dilakukan atas dasar fitur-fitur yang digunakan pada data cuaca harian, yang mana terdapat fitur seperti ‘*apparent\_temperature\_max (°C)*’ dan ‘*apparent\_temperature\_min (°C)*’ yang mengimplikasikan bahwa nilai *max* dan *min* memiliki informasi yang berdampak.

Tidak diambil nilai rerata karena nilai *max* dan *min* memberikan informasi lebih banyak dibandingkan dengan nilai tunggal rata-rata. Terlebih lagi, nantinya akan digunakan nilai standar deviasi yang menyiratkan informasi nilai rata-rata sehingga keberadaan nilai rata-rata cenderung bersifat *redundant*.

Pengambilan nilai *standard deviation* berangkat dari hipotesis bahwa data pada fitur independen cenderung lebih terdispersi saat terjadi hujan pada hari tersebut. Sebaliknya, variansi data pada fitur independen cenderung rendah saat tidak terjadi hujan pada hari tersebut. Hipotesis tersebut divalidasi melalui analisis yang menunjukkan bahwa data dengan *standard deviation* fitur yang lebih tinggi cenderung memiliki curah hujan yang lebih tinggi pula.



**Gambar 3.4 Grafik Hubungan antara Nilai Std Fitur 'soil\_moisture\_0\_to\_7cm (m<sup>3</sup>/m<sup>3</sup>)' dan Curah Hujan di Kota Su**

Pengambilan nilai *skewness* dilakukan agar model dapat menangkap ketidaksimetrisan data untuk dijadikan pertimbangan dalam melakukan prediksi. Hal ini bermula dari analisis bahwa saat hujan terjadi, distribusi suatu fitur akan cenderung terkonsentrasi pada salah satu sisi dan kurvanya akan menceng.

Langkah terakhir dari tahap ini adalah melakukan *encoding* yang bertujuan untuk mentransformasi tipe data fitur 'city' menjadi numerik sehingga dapat diinterpretasikan oleh model. Digunakan metode *mean encoding* yang memetakan tiap kelas 'city' dengan nilai rerata fitur targetnya. Dipilih metode ini karena angka hasil *encoding* akan mengandung makna atau informasi yang mungkin bermanfaat dalam prediksi. Selain itu, *mean encoding* tidak menambah dimensi fitur sehingga *computational cost* akan tetap sama.

## 5. Modeling and Validation Tahap 1

Dalam tahap *modeling*, dilakukan eksperimen untuk menentukan model yang paling akurat dalam memprediksi curah hujan. Model-model yang digunakan dibatasi berupa model berbasis *gradient boosting* dengan alasan performanya yang sudah terbukti sangat efektif. Selain itu, berdasarkan pengalaman sebelumnya, performa *gradient boosting* secara umum lebih unggul.

Sebenarnya, dalam *task* regresi dapat pula digunakan model linear. Namun, keberadaan *multicollinearity* serta ketiadaan hubungan linear yang kuat antara fitur independen dan dependen mengakibatkan model linear kurang efektif pada kasus ini. Oleh karena itu, diputuskan untuk tidak dipilih model linear.

Untuk validasi, digunakan metrik *mean-squared error* (MSE) yang memiliki formula sebagai berikut.

$$MSE = \sum_{i=1}^n \frac{(\hat{y}_i - y_i)^2}{n}.$$

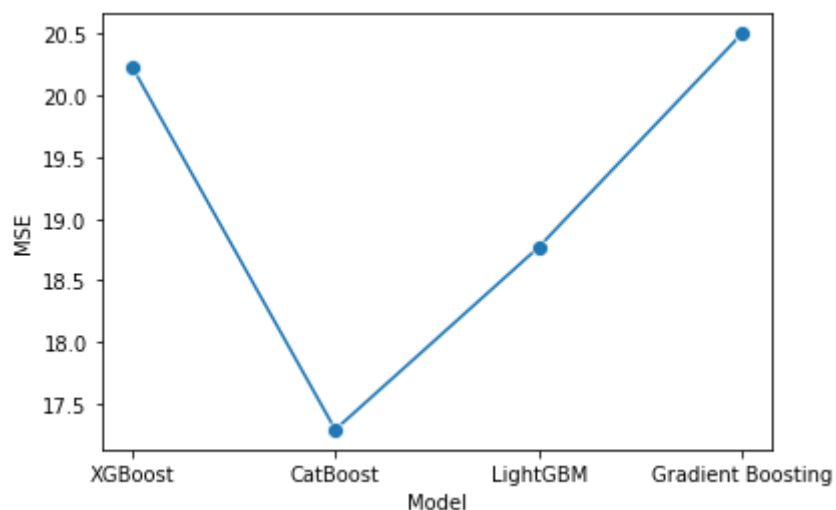
Keterangan:  $n$  = frekuensi data,  $i$  = urutan data,  $\hat{y}$  = nilai prediksi,  $y$  = nilai aktual.

Prinsipnya adalah selisih antara nilai prediksi dan nilai aktual dihitung, kemudian dikuadratkan dan dicari reratanya sehingga diperoleh nilai MSE.

Setelah itu, validasi dilakukan dengan metode *K-Fold cross validation* yang bekerja dengan prinsip membagi data secara acak menjadi  $k$  partisi dan melakukan prediksi terhadap masing-masing partisi secara terpisah. Pada kasus ini, digunakan  $k = 3$ .

**Tabel 4.1 Hasil Validasi**

Model	MSE
XGBoost	20.233
CatBoost	17.292
LightGBM	18.770
Scikit-Learn Gradient Boosting	20.500



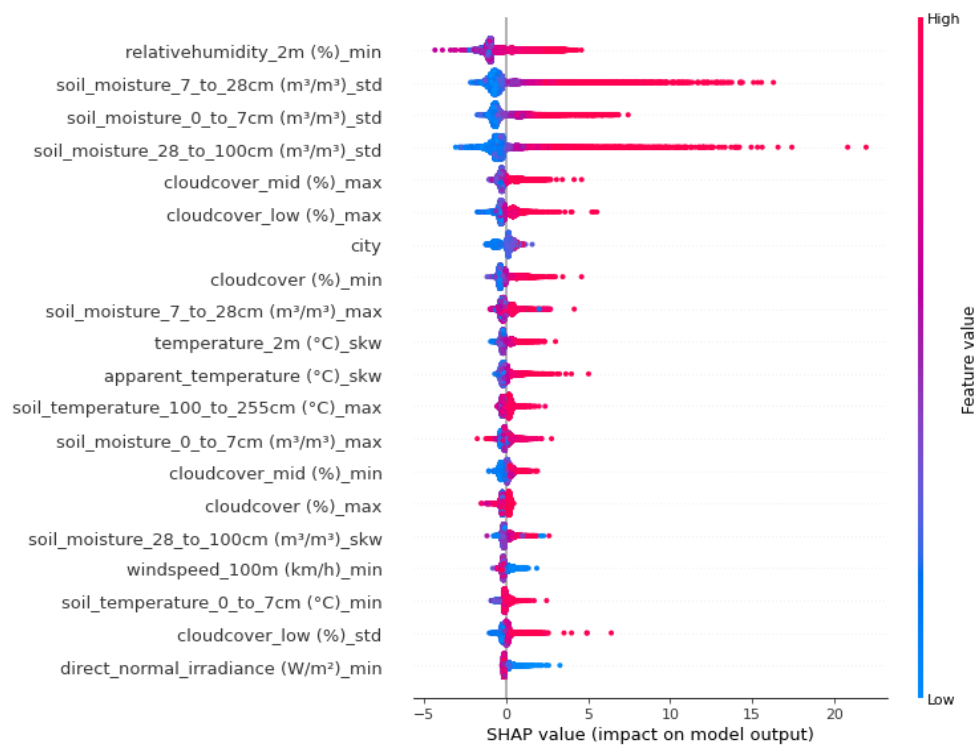
**Gambar 4.1 Grafik Garis MSE Hasil Validasi Tiap Model**

Hasil evaluasi menunjukkan bahwa model CatBoost menghasilkan MSE yang terkecil. Artinya, model CatBoost memiliki performa terbaik.

## 6. Feature Engineering Tahap 2

Sebelum dilakukan *feature engineering* tahap kedua, diperiksa tingkat kepentingan fitur yang bertujuan untuk mengetahui seberapa berguna suatu fitur dalam memprediksi variabel target.

Pada kasus ini, digunakan metode SHAP (*SHapley Additive exPlanations*) sebagai representasi tingkat kepentingan fitur. Secara spesifik, SHAP memperhitungkan seberapa besar pengaruh setiap nilai dari suatu fitur terhadap setiap prediksi yang dilakukan.



**Gambar 5.1 Dua Puluh Fitur dengan Tingkat Kepentingan Tertinggi**

Setelah diperoleh urutan fitur berdasarkan tingkat kepentingan, dilakukan ekstraksi fitur yang bertujuan untuk mengeliminasi fitur yang kurang berpengaruh. Dengan demikian, diperoleh fitur-fitur yang esensial yang berdampak pada peningkatan akurasi prediksi serta *computational cost* yang lebih rendah.

Ekstraksi fitur dilakukan dengan metode *Recursive Feature Elimination* (RFE). Prinsip dari metode ini yaitu dilakukan validasi secara berulang yang diiringi dengan pengurangan sebuah fitur pada setiap perulangannya. Skor validasi akan ditinjau untuk menentukan jumlah fitur yang optimal. Ilustrasi pada Gambar 5.2 mengilustrasikan bagaimana metode ini bekerja.

Validasi ke-1 → Digunakan n fitur dengan tingkat kepentingan tertinggi  
 Validasi ke-2 → Digunakan n-1 fitur dengan tingkat kepentingan tertinggi  
 Validasi ke-3 → Digunakan n-2 fitur dengan tingkat kepentingan tertinggi  
 ...  
 Validasi ke-n → Digunakan 1 fitur dengan tingkat kepentingan tertinggi

ket: n adalah total jumlah fitur independen

**Gambar 5.2 Ilustrasi Cara Kerja Recursive Feature Elimination**

Hasil *Recursive Feature Elimination* (RFE) Menunjukkan bahwa 37 fitur dengan tingkat kepentingan tertinggi menghasilkan skor validasi yang terbaik pula. Dengan demikian, untuk seterusnya akan digunakan 37 fitur dengan tingkat kepentingan tertinggi tersebut.

**Tabel 5.1 Cuplikan Hasil Recursive Feature Elimination**

	Top n-features	MSE
0	37	16.701
1	53	16.799
2	56	16.799
3	50	16.839
4	60	16.839

## 7. Modeling and Validation Tahap 2

Pada tahap ini, dilakukan *hyperparameter tuning* terhadap model terbaik dengan menggunakan *library* bernama Hyperopt yang menerapkan algoritma optimasi Bayesian dalam mencari kombinasi parameter terbaik.

*Hyperparameter tuning* diawali dengan mendefinisikan ruang sampel atau daerah jelajah parameter. Kemudian, Hyperopt melakukan validasi dengan mengambil nilai random ruang sampel. Adapun nilai MSE hasil validasi dijadikan sebagai acuan atau pertimbangan untuk menentukan nilai suatu parameter pada validasi selanjutnya. Hal ini berlangsung sebanyak 200 kali iterasi dengan *return*

*value* terakhir berupa parameter terbaik yang menghasilkan nilai MSE terkecil. Daerah jelajah dan hasil *hyperparameter tuning* pada model CatBoost dapat dilihat pada Tabel 6.1.

**Tabel 6.1 Parameter dan Hasil Hyperparameter Tuning oleh Hyperopt**

Parameter	Daerah Jelajah/Ruang Sampel	Nilai
learning_rate	hp.loguniform('learning_rate', np.log(np.e**-5), np.log(1))	0.0608
random_strength	hp.quniform('random_strength', 1, 20, 1)	12
l2_leaf_reg	hp.loguniform('l2_leaf_reg', np.log(1), np.log(10))	3.72266
bagging_temperature	hp.uniform('bagging_temperature', 0, 1)	0.29013
iterations	hp.quniform('iterations', 200, 2000, 100)	1900

Hasil evaluasi model CatBoost sebelum dan sesudah dilakukan *hyperparameter tuning* ditunjukkan pada Tabel 6.2. Hasil evaluasi menunjukkan bahwa penyesuaian parameter memberikan skor yang lebih baik dari sebelumnya.

**Tabel 6.2 Hasil Evaluasi Sebelum dan Sesudah Hyperparameter Tuning**

Kondisi	Model	MSE
Sebelum <i>tuning</i>	CatBoost	16.701
Sesudah <i>tuning</i>	CatBoost	16.102

## 8. Kesimpulan

Berdasarkan analisis dan pemodelan yang telah dilakukan, dapat ditarik kesimpulan sebagai berikut.

- Data cuaca yang telah dianalisis dengan metode-metode statistik menghasilkan informasi mengenai karakteristik dan hubungan antar fitur.
- Data cuaca yang telah melewati tahap *preprocessing* memiliki kualitas yang lebih baik.
- Dilakukan *feature engineering* sehingga diperoleh informasi tambahan yang membantu meningkatkan performa model dalam memprediksi curah hujan.
- Lewat modeling dan validasi, ditemukan bahwa model terbaik adalah CatBoost yang memiliki nilai MSE sebesar 17.292.
- Seleksi fitur dan *hyperparameter tuning* meningkatkan performa model dalam memprediksi curah hujan, yang dibuktikan dengan nilai MSE sebesar 16.102.