**LAPORAN FINAL PROJECT PMA**



**Disusun oleh Kelompok 10:**

**Alya Resti Saraswati 5026211057**

**Maharani Putri Efendi 5026211095**

**Fahrul Ramadhan Putra 5026211118**

**DEPARTEMEN SISTEM INFORMASI**

**FAKULTAS TEKNOLOGI ELEKTRO DAN INFORMATIKA CERDAS**

**INSTITUT TEKNOLOGI SEPULUH NOPEMBER**

**2023**

**Daftar Isi**

[**I. Overview Tugas 3**](#_b53uaxjrl3dj)

[**II. Deskripsi Permasalahan atau Problem yang Diselesaikan 3**](#_8xq8mmfj14mf)

[**III. Metodologi 3**](#_9dqajgjj8r)

[A. Support Vector Regression (SVR) 3](#_rp8hngvmxge3)

[B. Recurrent Neural Network (RNN) 4](#_ovr0rt33e3n3)

[C. Long Short Term Memory (LSTM) 4](#_ea1hzk3pr2wx)

[D. Gated Recurrent Unit (GRU) 5](#_7abm0nt8s1pm)

[**IV. Hasil Eksperimen & Perbandingan 5**](#_asiqhq5fs850)

[A. Preprocessing Data 5](#_832a20raxg33)

[**1. Import Library 5**](#_sngm4nbdtfwf)

[3. Menampilkan Tipe Data 7](#_g05fkaolkva3)

[4. Pemeriksaan Missing Value 7](#_30va74cybku2)

[5. Pemeriksaan data duplikat 7](#_3g465c80dxdd)

[6. Mengubah Format Datetime dan Set Index 7](#_ew2eu4pcn7f)

[7. Mencari Outliers 8](#_bl6sm7qtjjz4)

[9. Melihat Plottingan Dataset 9](#_70nfjmlqwtrk)

[B. SVR 9](#_93njx42t3j3y)

[1. Data Split 80%-20% 9](#_eq2jrijbybzb)

[a. Splitting Data Train dan Data Test 9](#_92xyln5vl1q5)

[b. Hyperparameter Tuning 10](#_fzeva7mrjmja)

[c. Membuat dan Melatih Model SVR 11](#_c6mf5zrfp8v1)

[d. Melakukan Prediksi 12](#_6evibfs54zd6)

[e. Evaluasi Model 12](#_5kwazbsaxp9)

[f. Forecast 13](#_h2skblphffrv)

[2. Data Split 70%-30% 15](#_pejj2nq8q8ki)

[a. Splitting Data Train dan Data Test 15](#_yt1tstzc0zoo)

[b. Hyperparameter Tuning 16](#_wb8qaz9qa3r5)

[c. Membuat dan Melatih Model SVR 17](#_esaz2b2o4bu6)

[d. Melakukan Prediksi 18](#_fto3poxzpdtx)

[e. Evaluasi Matrik dan Grafik 18](#_hl6ahyn4dl77)

[f. Forecast 18](#_r4w14kdvsv19)

[3. Data Split 60%-40% 20](#_yk386snbqmih)

[a. Splitting Data Train dan Data Test 20](#_r9zfpv9bdqwq)

[b. Hyperparameter Tuning 21](#_oih7g0x5l46q)

[c. Membuat dan Melatih Model SVR 22](#_1n1h8yh6itbi)

[d. Melakukan Prediksi 23](#_ukggobl4ygr3)

[e. Evaluasi Matrik dan Grafik 23](#_lcu3z9v3ji6y)

[f. Forecast 24](#_ft8zwnupm75s)

[C. RNN 25](#_5q8o4lbxm01s)

[D. LSTM 25](#_d3u5euq3v744)

[1. Deklarasi fungsi helper 25](#_n2gknh494l8u)

[a. Split dataset 25](#_7u7qji5tsl24)

[b. Scaling dataset 25](#_wv7w9xoadb5g)

[c. Buat Timestep atau Sequencing 26](#_dicwkd21ap73)

[d. Hyperparameter Tuning 26](#_76bq5nbcx230)

[e. Training Model 27](#_9wu5dgcwn303)

[f. Prediksi 28](#_a0abn9bg3c62)

[g. Forecasting 29](#_slzax39z6n1p)

[h. Rekapitulasi Data 30](#_80kgmli7y6ig)

[i. Fungsi utama skenario 30](#_d59a4gwzdmp0)

[2. Skenario 31](#_kd6jxleas78a)

[a. 80% Train, 20% Test 31](#_c8qhc6roskmw)

[b. 70% Train, 30% Test 32](#_rjqr9onmsy8w)

[c. 60% Train, 40% Test 32](#_5b0426hvct1w)

[3. Perbandingan Hasil Setiap Skenario LSTM 32](#_hh4kse3qjj0b)

[E. GRU 33](#_to61gwvtbt30)

[1. Deklarasi fungsi helper 33](#_nftqnadzres8)

[a. Split dataset 33](#_skh8bb5w44u)

[b. Scaling dataset 33](#_b8tky5uqi0o3)

[c. Buat Timestep atau Sequencing 33](#_57ypn2o4e4u)

[d. Hyperparameter Tuning 34](#_1zt9g6milfte)

[e. Training Model 35](#_cketmv4x027e)

[f. Prediksi 35](#_xvtncprzykfp)

[g. Forecasting 35](#_btxztoblny6u)

[h. Rekapitulasi Data 36](#_g21w76tcjhpo)

[i. Fungsi utama skenario 37](#_a9yzzndbtj48)

[2. Skenario 38](#_wiz19n5w94d)

[a. 80% Train, 20% Test 38](#_1iay928u8fi4)

[b. 70% Train, 30% Test 39](#_gbuv1v4wi7x5)

[c. 60% Train, 40% Test 39](#_63cqqpctksxx)

[3. Perbandingan Hasil Setiap Skenario GRU 40](#_2duaze3vuch6)

[F. Perbandingan Hasil Metrik Semua Model 40](#_vxmopasv826)

[**V. Hasil Forecast 41**](#_qafu01as8nky)

[A. SVR 41](#_8u9ozrmxqi0i)

[B. RNN 41](#_cgpksmtqc0my)

[C. LSTM 41](#_t7h4eqnja8s3)

[D. GRU 41](#_5q9me5osikq5)

[**VI. Hasil Analisis dan Kesimpulan 41**](#_85jh0g84a9t9)

[**VII. Referensi 42**](#_azydsms129xe)

# Overview Tugas

Tugas ini bertujuan melakukan forecasting untuk 12 periode berikutnya menggunakan metode deep learning (RNN, LSTM, GRU) dan Support Vector Regression (SVR) dengan menggunakan dataset DKI Jakarta dari tahun 2014 hingga 2022. Tugas ini melibatkan tiga skenario pembagian dataset train dan test yang berbeda, yaitu 80% Train - 20% Test, 70% Train - 30% Test, dan 60% Train - 40% Test. Fokus utama dari analisis ini adalah variabel total\_visitor dengan mempertimbangkan total\_accomodation sebagai faktor yang mempengaruhi.

Untuk setiap skenario, akan dilakukan proses training dan testing menggunakan model deep learning (RNN, LSTM, GRU) dan SVR. Evaluasi forecast akan dilakukan dengan menggunakan metrik Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE), dan Mean Absolute Error (MAE) untuk memastikan keakuratan dan kinerja model. Selain itu, juga akan terdapat analisis kesimpulan mengenai model yang dibuat dengan masing-masing metode.

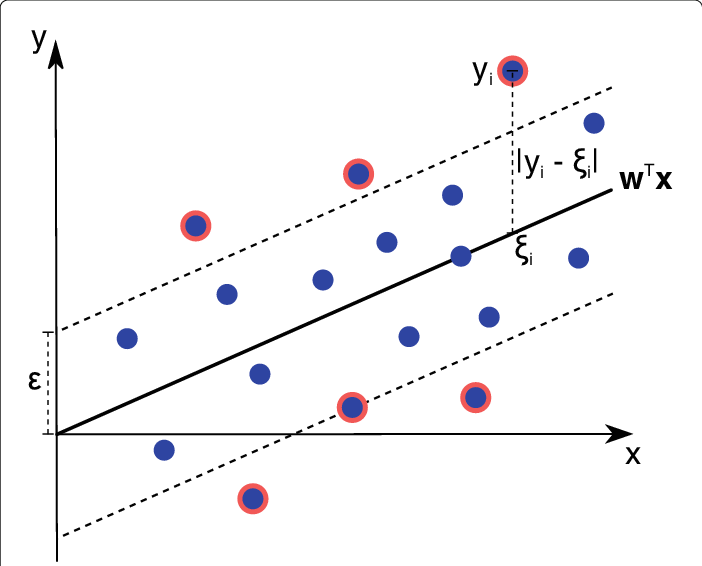
# Deskripsi Permasalahan atau Problem yang Diselesaikan

Permasalahan yang diangkat dalam Final Project ini melibatkan prediksi jumlah pengunjung ke Provinsi DKI Jakarta di masa depan. Faktor-faktor yang dapat memengaruhi jumlah pengunjung ke suatu daerah sangat kompleks, dan dalam konteks ini, penelitian akan memberikan fokus pada ketersediaan akomodasi sebagai salah satu variabel kunci. Data yang digunakan dalam penelitian ini adalah dataset provinsi Jakarta yang mencakup pertumbuhan pengunjung dan akomodasi selama sembilan tahun terakhir, dari tahun 2014 hingga 2022. Dataset tersebut terdiri dari tiga kolom, yaitu ‘datetime’, ‘total\_visitor’, dan ‘total\_accomodation’.Metode yang digunakan dalam penelitian ini adalah deep learning (RNN, LSTM, GRU) dan Support Vector Regression (SVR). Ketiga metode tersebut dipilih karena memiliki kemampuan untuk mempelajari pola data yang kompleks.Peramalan dilakukan pada tiga skenario berbeda untuk setiap metode dengan pembagian dataset train-test menggunakan rasio 80%-20%, 70%-30%, dan 60%-40%.

# Metodologi

## **Support Vector Regression (SVR)**

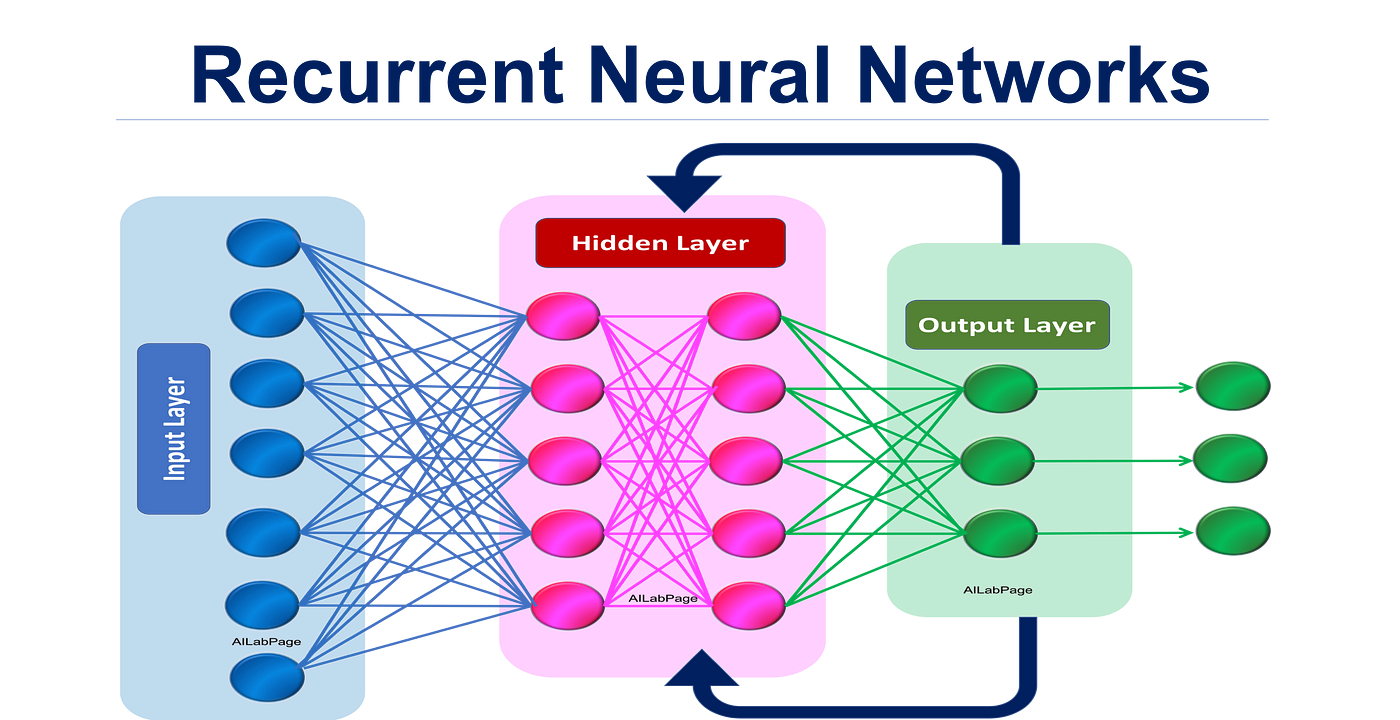
Support Vector Regression (SVR) adalah metode dalam machine learning yang digunakan untuk memprediksi nilai kontinu berdasarkan input yang diberikan. SVR bekerja dengan mencari hyperplane yang paling baik memisahkan antara data target, dengan memperbolehkan adanya kesalahan prediksi dalam batas tertentu. Meskipun hasil pencarian tidak menghasilkan penjelasan khusus mengenai SVR dalam konteks jurnal yang disebutkan, namun dari pengetahuan umum, SVR umumnya digunakan dalam konteks machine learning dan analisis data untuk memprediksi nilai kontinu berdasarkan fitur-fitur yang diberikan.



Gambar 1. Metode SVR

## **Recurrent Neural Network (RNN)**

Recurrent Neural Network (RNN) adalah jenis arsitektur jaringan saraf yang dirancang khusus untuk menangani data berurutan atau data yang memiliki dependensi temporal. RNN memiliki kemampuan untuk menyimpan dan menggunakan informasi dari data sebelumnya dalam suatu urutan, memungkinkan mereka untuk menangkap pola dan hubungan yang kompleks dalam data deret waktu. Keunikan RNN terletak pada kemampuannya mempertahankan suatu bentuk "memori" internal, yang memungkinkan pengolahan informasi kontekstual dan dinamis.



Gambar 2. Metode RNN

## 

## **Long Short Term Memory (LSTM)**

Long Short Term Memory (LSTM) merupakan salah satu pengembangan dari arsitektur Recurrent Neural Network (RNN) yang dirancang untuk mengatasi permasalahan vanishing gradient pada RNN saat memproses data sekuensial yang panjang. LSTM diusulkan sebagai solusi untuk mengatasi kesulitan dalam mempertahankan informasi dalam jangka waktu yang lama karena adanya memory cell. Arsitektur LSTM terdiri dari tiga gates, yaitu forget gate (𝑓𝑡), input gate (𝑖𝑡), dan output gate (𝑂𝑡), yang memungkinkan pengelolaan informasi dalam memory cell state. LSTM telah diterapkan dalam berbagai konteks, seperti prediksi harga saham, klasifikasi teks, dan pemodelan sekuensial lainnya

## **Gated Recurrent Unit (GRU)**

Gated Recurrent Unit (GRU) merupakan jenis arsitektur jaringan saraf rekurensial (RNN) yang dirancang untuk mengatasi permasalahan vanishing gradient dan memori panjang pada Long Short-Term Memory (LSTM). GRU memiliki struktur yang lebih sederhana dibandingkan dengan LSTM, dengan hanya menggunakan dua gates, yaitu update gate (𝑧𝑡) dan reset gate (𝑟𝑡). Meskipun GRU lebih simpel daripada LSTM, namun tetap efektif dalam banyak aplikasi, terutama ketika diperlukan efisiensi komputasi dan ketika data yang diolah tidak terlalu kompleks. GRU telah menunjukkan keberhasilannya dalam tugas-tugas seperti pengenalan suara, pemrosesan bahasa alami, dan prediksi deret waktu.

# 

Gambar 4. Metode GRU

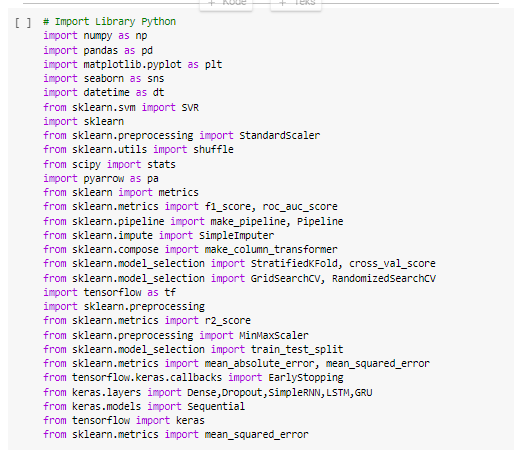
## 

# Hasil Eksperimen & Perbandingan

## Preprocessing Data

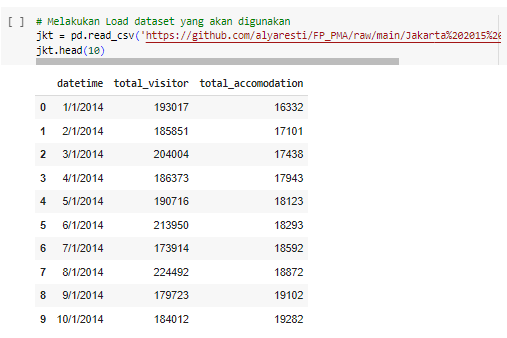
### Import Library

Untuk melakukan pemeriksaan tipe data dan analisis awal data, pertama-tama, kita perlu mengimpor library yang diperlukan dalam lingkup analisis data. Sebagai contoh, kita dapat menggunakan library seperti pandas untuk manipulasi data dan pengecekan tipe data seperti dibawah ini.

****

Gambar 5. Kode Import Libarary

1. Import Dataset

Setelah mengimpor library, langkah selanjutnya adalah membuat dataframe baru yang akan kita namai 'jkt' dan mengisi dataframe tersebut dengan data yang akan dianalisis. Fungsi `pd.read\_csv` dari Pandas digunakan untuk mengambil data dari file CSV, pada kode ini dilakukan import data set dari URL file CSV. File CSV yang dimaksud memiliki nama 'Jakarta 2015 - 2022.csv'. Data yang diambil dari URL tersebut kemudian dimuat ke dalam struktur data DataFrame dan diberi nama 'ntt'. 

Gambar 6. Kode Load Dataset

### Menampilkan Tipe Data

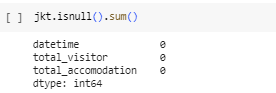
Proses ini dilakukan untuk melakukan pengecekan dan penyesuaian tipe data. Kolom datetime mempunyai tipe data yang masih belum sesuai. Fungsi `info()` memberikan gambaran umum tentang struktur dataset dengan menyajikan informasi seperti jumlah entri, tipe data dari setiap kolom, serta keberadaan nilai yang hilang. Dengan menggunakan fungsi ini, dapat dengan cepat melihat ringkasan dataset secara komprehensif, memudahkan pemahaman mengenai sifat dan karakteristik data yang sedang diolah.



Gambar 7. Kode Menampilkan Tipe Data

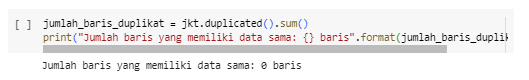
### Pemeriksaan Missing Value

Pemeriksaan *Missing Value* dilakukan karena salah satu ciri dari data yang kotor adalah data yang tidak lengkap, yaitu data yang fieldnya berisi null value maupun *unknown* (tidak diketahui). Pemeriksaan *Missing Value*, dapat dilakukan dengan menggunalam kode isnull() dan sum(). Dari hasil kode yang dijalankan, tidak ditemukan baris yang kosong (null) pada dataset. Namun, terdapat banyak field yang berisi *unknown*.



Gambar 8. Pemeriksaan Missing Value

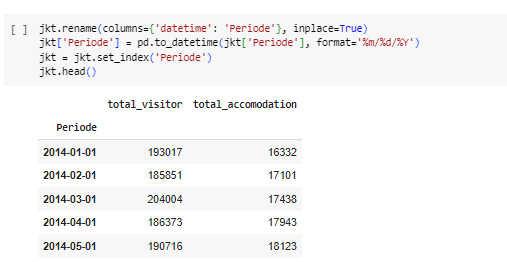
### Pemeriksaan data duplikat

Proses pemeriksaan data yang duplikat dilakukan karena data yang duplikat juga merupakan ciri data yang kotor. Untuk memeriksa apakah terdapat data duplikat dalam DataFrame, digunakan metode duplicated() dan sum() dari Pandas. Setelah kode dijalankan, tidak ditemukan data yang duplikat. 

Gambar 9. Pemeriksaan Data Duplikat

### Mengubah Format Datetime dan Set Index

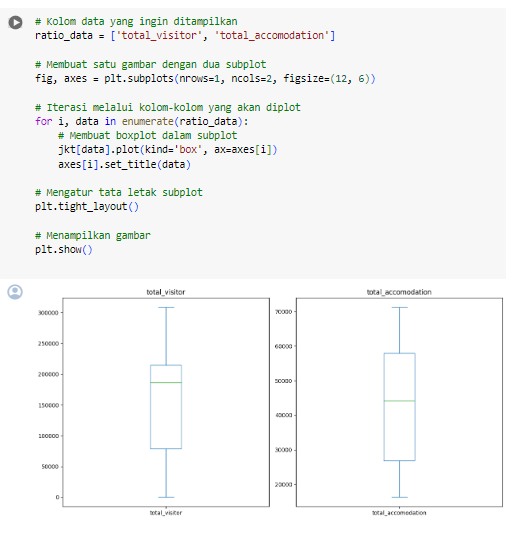
Mengubah format pada kolom datetime mejadi format M-D-Y. Hal ini dapat dilakukan dengan menggunakan kode format %m/%d/%Y.



Gambar 10. Mengubah Format DateTime dan Set Index

### Mencari Outliers

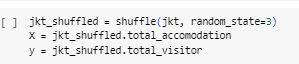
Outlier adalah data yang nilainya berbeda secara signifikan dari kesuluruhan data. Outlier memiliki dampak yang besar pada hasil analisis statistik atau pemodelan data. Untuk mendeteksi outlier kita menggunakan box plot. Namun, karena data yang digunakan adalah time series, maka tidak dilakukan penghapusan outlier



Gambar 11. Mencari Outliers

1. Shuffle Data

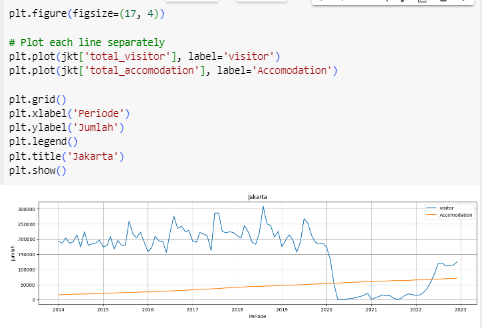
Proses ini dilakukan sebelum melakukan pemisahan data training dan testing. Ketika membagi data menjadi set pelatihan dan pengujian, shuffle data memastikan bahwa setiap kategori atau label ada di kedua set. Ini mengurangi risiko pembuatan model yang hanya terlatih pada satu kelompok tertentu dari data.



Gambar 12. Shuffle Data

### Melihat Plottingan Dataset

Dengan memplot dua variabel pada satu grafik, kita dapat dengan mudah membandingkan perubahan relatif antara jumlah pengunjung dan jumlah akomodasi. Hal ini dapat memberikan wawasan tentang sejauh mana pertumbuhan jumlah pengunjung dipengaruhi oleh ketersediaan akomodasi.



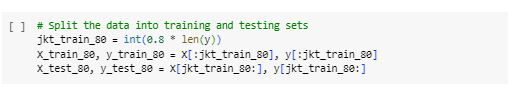
Gambar 13. Melihat Plottingan Dataset

## SVR

### Data Split 80%-20%

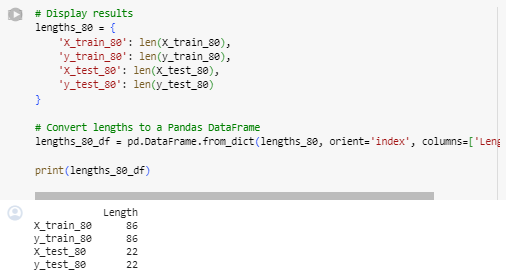
#### Splitting Data Train dan Data Test

Membagi dataset, bertujuan untuk membangun dan mengukur model secara lebih akurat, mengurangi risiko overfitting, dan memastikan bahwa model yang dihasilkan dapat diterapkan dengan baik pada situasi dunia nyata. Pertama, dalam kode di bawah ini, panjang dataset (len(y)) ditentukan, dan kemudian jumlah data yang akan digunakan sebagai data pelatihan (jkt\_train80) dihitung dengan mengambil 80% dari panjang total dataset. Setelah itu, dilakukan pemisahan antara data pelatihan dan data pengujian dengan menggunakan slicing pada array X dan y berdasarkan perhitungan sebelumnya. Sebagai contoh, X\_train80 dan y\_train80 berisi 80% pertama dari data X dan y, sementara X\_test80 dan y\_test80 berisi 20% sisanya.



Gambar 14. Split Data SVR 80%-20%

Setelah menjalankan kode di atas, diperlukan kode untuk melihat informasi berapa banyak entri yang ada dalam setiap bagian. Hal ini bertujuan untuk memastikan bahwa proses pemisahan dataset telah dilakukan dengan benar dan sesuai dengan rasio yang diinginkan (80-20).



Gambar 15. Result Split SVR 80%-20%

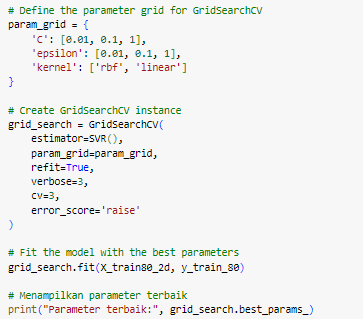
Setelah membagi dataset, dilakukan pengubahan nilai input dengan menggunakan ‘reshape(-1, 1)’, dengan untuk mengubah bentuk array atau series menjadi matriks dua dimensi (2D) dengan satu kolom. Hal ini dilakukan karena banyak model *machine learning* diharapkan menerima input dalam bentuk matriks 2D.



Gambar 16. Mengubah nilai input menjadi 2D

#### Hyperparameter Tuning

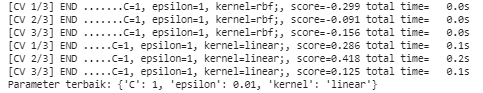
Proses ini dilakukan untuk mengidentifikasi kombinasi parameter terbaik untuk model SVR berdasarkan hasil evaluasi pada set pelatihan dengan menggunakan Grid Search Cross-Validation. Grid search adalah teknik penalaan hiperparameter di mana sejumlah nilai hiperparameter yang telah ditentukan sebelumnya diuji secara menyeluruh untuk menemukan kombinasi yang memberikan kinerja model terbaik. Pada kasus ini, nampaknya telah diterapkan pada model SVR.



Gambar 17. Kode Hyperparameter Tuning

Pertama-tama, kita menentukan `param\_grid` yang berisi serangkaian nilai untuk setiap parameter. Sebagai contoh, parameter `C` diuji pada nilai [0.01, 0.1, 1], parameter `epsilon` diuji pada nilai [0.01, 0.1, 1], dan parameter `kernel` diuji pada jenis ['linear', 'rbf']. Selanjutnya, kita menciptakan instance dari GridSearchCV menggunakan model SVR, `param\_grid` yang telah ditentukan sebelumnya, dan konfigurasi lainnya seperti `cv` (jumlah lipatan cross-validation) dan `verbose` untuk menampilkan informasi selama proses pencarian.

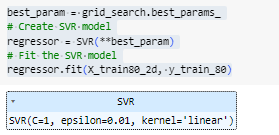
Langkah selanjutnya adalah melakukan proses fitting dengan memanggil metode `fit()` pada instance GridSearchCV. Setelah pencarian parameter selesai, kita memperoleh parameter terbaik dengan menggunakan atribut `best\_params\_` dari objek GridSearchCV. Dengan informasi mengenai parameter terbaik ini, kita dapat melakukan optimasi pada model SVR untuk data yang kita miliki. Penting dicatat bahwa GridSearchCV secara otomatis akan melatih ulang model menggunakan parameter terbaik pada seluruh data pelatihan setelah proses pencarian selesai, dengan opsi `refit=True`. Setelah menjalankan kode di atas, didapatkan kombinasi parameter terbaiknya adalah untuk parameter ‘C’ = 1, ‘epsilon’ = 0.01, ‘kernel’ = linear.



Gambar 18. Hasil Kode Hyperparameter Tuning

#### Membuat dan Melatih Model SVR

Setelah mendapat parameter terbaik, dilakukan pembuat model dan melatih model SVR. Model SVR akan dibuat menggunakan metode grid search untuk menemukan kombinasi hiperparameter optimal. Setelah proses grid search selesai, model SVR baru akan dibuat dan dilatih dengan menggunakan parameter terbaik yang telah ditemukan pada satu set data pelatihan tertentu. Variabel `best\_param` akan mengandung kombinasi parameter terbaik yang diidentifikasi selama proses grid search.



Gambar 19. Membuat dan melatih model SVR

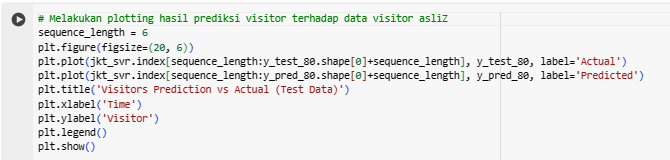
#### Melakukan Prediksi

Kode di bawah akan melakukan prediksi menggunakan model regresi yang telah dilatih sebelumnya (regressor) pada data uji (X\_test80\_2d).



Gambar 20. Prediksi SVR 80%-20%

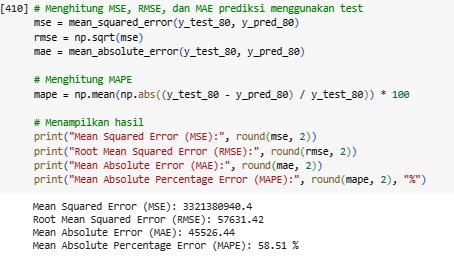
Setelah dilakukan prediksi menggunakan SVR, selanjutnya adalah menampilkan plot hasil prediksi visitor terhadap data asli visitor dengan sequence lengthnya adalah 6



gambar 21. Visualisasi prediksi visitor dan data visitor asli

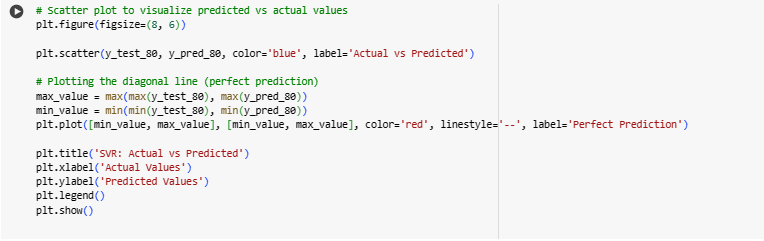
#### Evaluasi Model

Evaluasi model adalah proses penilaian dan evaluasi kinerja suatu model berdasarkan sejumlah matrik atau matriks evaluasi. Pada kode di bawah ini, dilakukan pencarian nilai MSE, MAE, RMSE, dan MAPE pada model SVR.



Gambar 22. Evaluasi Matrik Model SVR skenario 80% - 20%

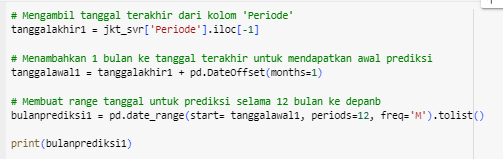
Untuk membandingkan nilai aktual (y\_test\_80) dengan nilai yang diprediksi (y\_pred\_80) dilakukan pembuatan visual menggunakan scatterplot.



Gambar 23. Kode Scatter Plot Perbandingan nilai aktual dan nilai prediksi

#### Forecast

Selanjutnya adalah menentukan periode waktu yang akan digunakan untuk melakukan prediksi selama 12 bulan ke depan setelah tanggal terakhir yang ada dalam data. Kode di bawah akan mengambil terakhir dari kolom ‘Periode’ dalam DataFrame ‘jkt\_svr’. Setelah itu, menambahkan 1 bulan ke tanggal terakhir untuk mendapatkan tanggal awal prediksi yang disimpan dalam ‘tanggalawal1’. Kemudian, membuat *range* tanggal untuk Prediksi 12 bulan ke depan.

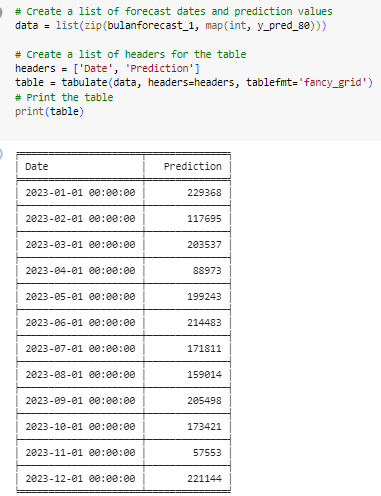
****

Gambar 24. Menentukan Periode Prediksi SVR

Kode di bawah ini akan menghasilkan DataFrame ‘gabung1’ yang berisi data aktual dan data prediksi yang telah diurutkan berdasarkan tanggal. Kode ini digunakan untuk menggabungkan data asli dengan data yang telah diprediksi dan kemudian mengurutkan DataFrame yang telah digabung berdasarkan kolom 'Periode'. Langkah ini berguna untuk visualisasi data asli dan hasil prediksi secara bersamaan.

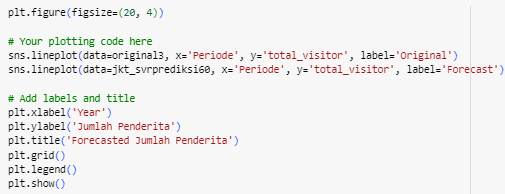
Gambar 25. Menggabungkan Data 80%-20%

Selanjutnya, dijalankan kode berikut untuk memberikan tanggal prediksi dan nilai prediksi pengunjung dalam bentuk tabel. DataFrame ‘data’ menggabungkan tanggal prediksi (bulanforecast\_1) dengan nilai prediksi (y\_pred\_80) ke dalam sebuah list. zip digunakan untuk menggabungkan dua list. Kode di bawah ini akan menampilkan hasil *forecast* 12 bulan ke depan dalam bentuk tabel.



Gambar 26. Kode Forecast SVR 80%-20% Bentuk Tabel

Selanjutnya, untuk menampilkan peramalan menggunakan grafik, dapat menggunakan kode di bawah ini. Kode ini bertujuan untuk membuat sebuah plot garis (line plot) menggunakan library ‘seaborn’ untuk memvisualisasikan data aktual dan hasil prediksi (forecast) jumlah visitor.

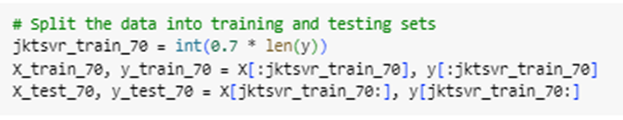


Gambar 27. Kode Forecast SVR 80%-20% Bentuk Grafik

### Data Split 70%-30%

#### Splitting Data Train dan Data Test

Membagi dataset, bertujuan untuk membangun dan mengukur model secara lebih akurat, mengurangi risiko overfitting, dan memastikan bahwa model yang dihasilkan dapat diterapkan dengan baik pada situasi dunia nyata. Pertama, dalam kode di bawah ini, panjang dataset (len(y)) ditentukan, dan kemudian jumlah data yang akan digunakan sebagai data pelatihan (jkt\_train70) dihitung dengan mengambil 70% dari panjang total dataset. Setelah itu, dilakukan pemisahan antara data pelatihan dan data pengujian dengan menggunakan slicing pada array X dan y berdasarkan perhitungan sebelumnya. Sebagai contoh, X\_train70 dan y\_train70 berisi 70% pertama dari data X dan y, sementara X\_test70 dan y\_test70 berisi 30% sisanya.



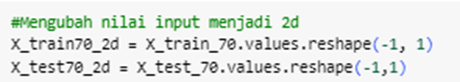
Gambar 28. Split Data SVR 70%-30%

Setelah menjalankan kode di atas, diperlukan kode untuk melihat informasi berapa banyak entri yang ada dalam setiap bagian. Hal ini bertujuan untuk memastikan bahwa proses pemisahan dataset telah dilakukan dengan benar dan sesuai dengan rasio yang diinginkan (70-30).



Gambar 29. Result Split SVR 70%-30%

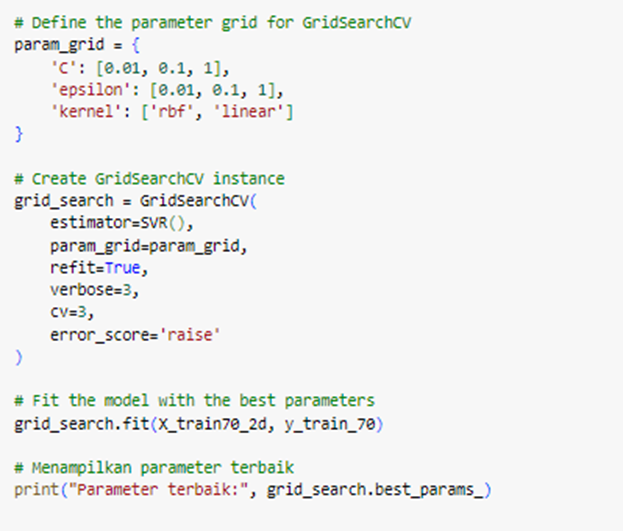
Setelah membagi dataset, dilakukan pengubahan nilai input dengan menggunakan ‘reshape(-1, 1)’, dengan untuk mengubah bentuk array atau series menjadi matriks dua dimensi (2D) dengan satu kolom. Hal ini dilakukan karena banyak model *machine learning* diharapkan menerima input dalam bentuk matriks 2D.



Gambar 30. Mengubah nilai input SVR 70%-30% menjadi 2D

#### Hyperparameter Tuning

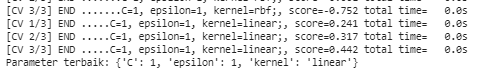
Proses ini dilakukan untuk mengidentifikasi kombinasi parameter terbaik untuk model SVR berdasarkan hasil evaluasi pada set pelatihan dengan menggunakan Grid Search Cross-Validation. Grid search adalah teknik penalaan hiperparameter di mana sejumlah nilai hiperparameter yang telah ditentukan sebelumnya diuji secara menyeluruh untuk menemukan kombinasi yang memberikan kinerja model terbaik. Pada kasus ini, nampaknya telah diterapkan pada model SVR.



Gambar 31. Kode Hyperparameter Tuning

Pertama-tama, kita menentukan `param\_grid` yang berisi serangkaian nilai untuk setiap parameter. Sebagai contoh, parameter `C` diuji pada nilai [0.01, 0.1, 1], parameter `epsilon` diuji pada nilai [0.01, 0.1, 1], dan parameter `kernel` diuji pada jenis ['linear', 'rbf']. Selanjutnya, kita menciptakan instance dari GridSearchCV menggunakan model SVR, `param\_grid` yang telah ditentukan sebelumnya, dan konfigurasi lainnya seperti `cv` (jumlah lipatan cross-validation) dan `verbose` untuk menampilkan informasi selama proses pencarian.

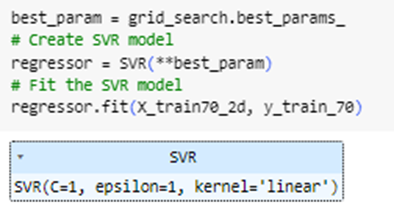
Langkah selanjutnya adalah melakukan proses fitting dengan memanggil metode `fit()` pada instance GridSearchCV. Setelah pencarian parameter selesai, kita memperoleh parameter terbaik dengan menggunakan atribut `best\_params\_` dari objek GridSearchCV. Dengan informasi mengenai parameter terbaik ini, kita dapat melakukan optimasi pada model SVR untuk data yang kita miliki. Penting dicatat bahwa GridSearchCV secara otomatis akan melatih ulang model menggunakan parameter terbaik pada seluruh data pelatihan setelah proses pencarian selesai, dengan opsi `refit=True`. Setelah menjalankan kode di atas, didapatkan kombinasi parameter terbaiknya adalah untuk parameter ‘C’ = 1, ‘epsilon’ = 0.01, ‘kernel’ = linear.



Gambar 32. Hasil Kode Hyperparameter Tuning SVR2

#### Membuat dan Melatih Model SVR

Setelah mendapat parameter terbaik, dilakukan pembuat model dan melatih model SVR. Model SVR akan dibuat menggunakan metode grid search untuk menemukan kombinasi hiperparameter optimal. Setelah proses grid search selesai, model SVR baru akan dibuat dan dilatih dengan menggunakan parameter terbaik yang telah ditemukan pada satu set data pelatihan tertentu. Variabel `best\_param` akan mengandung kombinasi parameter terbaik yang diidentifikasi selama proses grid search.



Gambar 33. Membuat dan melatih model SVR

#### Melakukan Prediksi

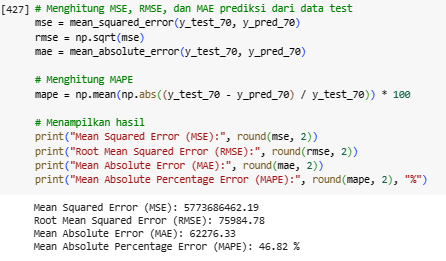
Kode di bawah akan melakukan prediksi menggunakan model regresi yang telah dilatih sebelumnya (regressor) pada data uji (X\_test70\_2d).



Gambar 34. Prediksi SVR 70%-30%

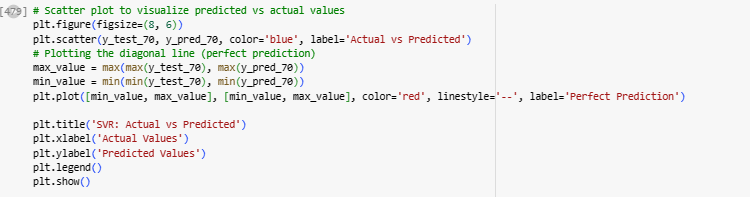
#### Evaluasi Model

Evaluasi model adalah proses penilaian dan evaluasi kinerja suatu model berdasarkan sejumlah matriks atau matriks evaluasi. Pada kode di bawah ini, dilakukan pencarian nilai MSE, MAE, RMSE, dan MAPE pada model SVR



Gambar 35. Evaluasi Matrik Model SVR skenario 70% - 30%

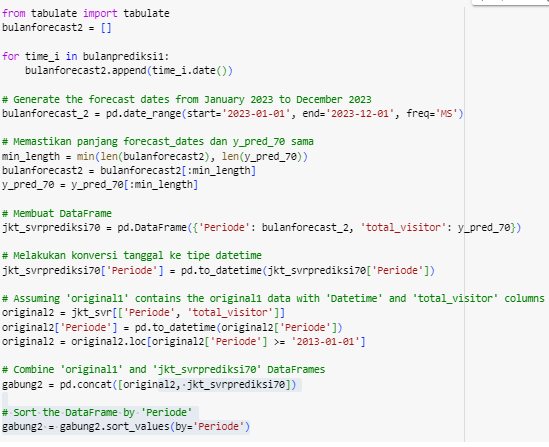
Untuk membandingkan nilai aktual (y\_test\_60) dengan nilai yang diprediksi (y\_pred\_60) dilakukan pembuatan visual menggunakan scatterplot.



Gambar 36. Scatter Plot Perbandingan nilai aktual dan nilai prediks

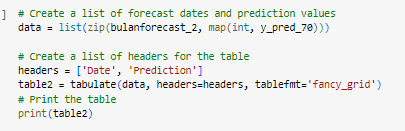
#### Forecast

Kode di bawah ini akan menghasilkan DataFrame ‘gabung2’ yang berisi data aktual dan data prediksi yang telah diurutkan berdasarkan tanggal. Kode ini digunakan untuk menggabungkan data asli dengan data yang telah diprediksi dan kemudian mengurutkan DataFrame yang telah digabung berdasarkan kolom 'Periode'. Langkah ini berguna untuk visualisasi data asli dan hasil prediksi secara bersamaan.



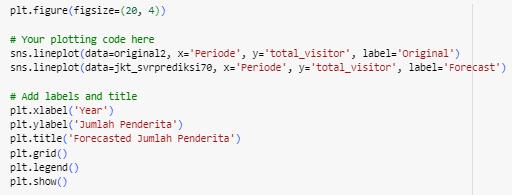
Gambar 37. Menggabungkan Data 70%-30%

Selanjutnya, dijalankan kode berikut untuk memberikan tanggal prediksi dan nilai prediksi pengunjung dalam bentuk tabel. DataFrame ‘data’ menggabungkan tanggal prediksi (bulanforecast\_2) dengan nilai prediksi (y\_pred\_70) ke dalam sebuah list. zip digunakan untuk menggabungkan dua list. Pada kode tersebut terdapat data frame ‘bulanprediksi1’, hal ini merupakan pemanggilan kode yang telah dijalankan pada gambar 24



Gambar 38. Kode Forecast SVR 70%-20%

Selanjutnya, untuk menampilkan peramalan menggunakan grafik, dapat menggunakan kode di bawah ini. Kode ini bertujuan untuk membuat sebuah plot garis (line plot) menggunakan library ‘seaborn’ untuk memvisualisasikan data aktual dan hasil prediksi (forecast) jumlah visitor.

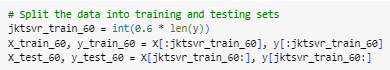


Gambar 39. Kode Forecast SVR 70%-30% Bentuk Grafik

### Data Split 60%-40%

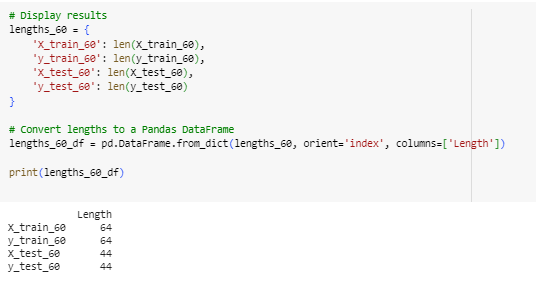
#### Splitting Data Train dan Data Test

Membagi dataset, bertujuan untuk membangun dan mengukur model secara lebih akurat, mengurangi risiko overfitting, dan memastikan bahwa model yang dihasilkan dapat diterapkan dengan baik pada situasi dunia nyata. Pertama, dalam kode di bawah ini, panjang dataset (len(y)) ditentukan, dan kemudian jumlah data yang akan digunakan sebagai data pelatihan (jkt\_train60) dihitung dengan mengambil 60% dari panjang total dataset. Setelah itu, dilakukan pemisahan antara data pelatihan dan data pengujian dengan menggunakan slicing pada array X dan y berdasarkan perhitungan sebelumnya. Sebagai contoh, X\_train60 dan y\_train60 berisi 60% pertama dari data X dan y, sementara X\_test60 dan y\_test60 berisi 40% sisanya.



Gambar 40. Split Data SVR 60%-40%

Setelah menjalankan kode di atas, diperlukan kode untuk melihat informasi berapa banyak entri yang ada dalam setiap bagian. Hal ini bertujuan untuk memastikan bahwa proses pemisahan dataset telah dilakukan dengan benar dan sesuai dengan rasio yang diinginkan (60-40).



Gambar 41. Result Split SVR 70%-30%

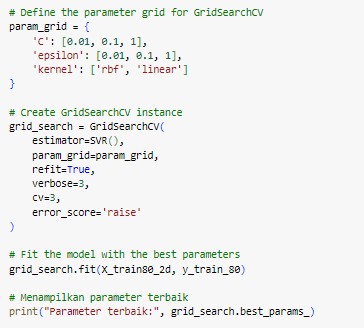
Setelah membagi dataset, dilakukan pengubahan nilai input dengan menggunakan ‘reshape(-1, 1)’, dengan untuk mengubah bentuk array atau series menjadi matriks dua dimensi (2D) dengan satu kolom. Hal ini dilakukan karena banyak model *machine learning* diharapkan menerima input dalam bentuk matriks 2D.



Gambar 42. Mengubah nilai input SVR 60%-40% menjadi 2D

#### Hyperparameter Tuning

Proses ini dilakukan untuk mengidentifikasi kombinasi parameter terbaik untuk model SVR berdasarkan hasil evaluasi pada set pelatihan dengan menggunakan Grid Search Cross-Validation. Grid search adalah teknik penalaan hiperparameter di mana sejumlah nilai hiperparameter yang telah ditentukan sebelumnya diuji secara menyeluruh untuk menemukan kombinasi yang memberikan kinerja model terbaik. Pada kasus ini, nampaknya telah diterapkan pada model SVR.



Gambar 43. Kode Hyperparameter Tuning

Pertama-tama, kita menentukan `param\_grid` yang berisi serangkaian nilai untuk setiap parameter. Sebagai contoh, parameter `C` diuji pada nilai [0.01, 0.1, 1], parameter `epsilon` diuji pada nilai [0.01, 0.1, 1], dan parameter `kernel` diuji pada jenis ['linear', 'rbf']. Selanjutnya, kita menciptakan instance dari GridSearchCV menggunakan model SVR, `param\_grid` yang telah ditentukan sebelumnya, dan konfigurasi lainnya seperti `cv` (jumlah lipatan cross-validation) dan `verbose` untuk menampilkan informasi selama proses pencarian.

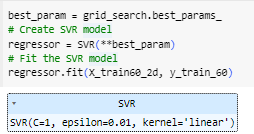
Langkah selanjutnya adalah melakukan proses fitting dengan memanggil metode `fit()` pada instance GridSearchCV. Setelah pencarian parameter selesai, kita memperoleh parameter terbaik dengan menggunakan atribut `best\_params\_` dari objek GridSearchCV. Dengan informasi mengenai parameter terbaik ini, kita dapat melakukan optimasi pada model SVR untuk data yang kita miliki. Penting dicatat bahwa GridSearchCV secara otomatis akan melatih ulang model menggunakan parameter terbaik pada seluruh data pelatihan setelah proses pencarian selesai, dengan opsi `refit=True`. Setelah menjalankan kode di atas, didapatkan kombinasi parameter terbaiknya adalah untuk parameter ‘C’ = 1, ‘epsilon’ = 0.01, ‘kernel’ = linear.



Gambar 44. Hasil Kode Hyperparameter Tuning

#### Membuat dan Melatih Model SVR

Setelah mendapat parameter terbaik, dilakukan pembuat model dan melatih model SVR. Model SVR akan dibuat menggunakan metode grid search untuk menemukan kombinasi hiperparameter optimal. Setelah proses grid search selesai, model SVR baru akan dibuat dan dilatih dengan menggunakan parameter terbaik yang telah ditemukan pada satu set data pelatihan tertentu. Variabel `best\_param` akan mengandung kombinasi parameter terbaik yang diidentifikasi selama proses grid search.



Gambar 45. Membuat dan melatih model SVR

#### Melakukan Prediksi

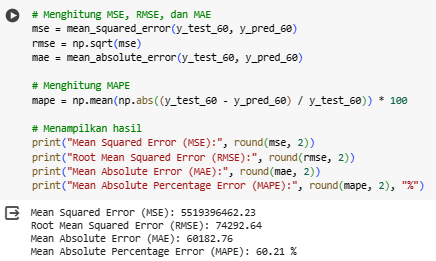
Kode di bawah akan melakukan prediksi menggunakan model regresi yang telah dilatih sebelumnya (regressor) pada data uji (X\_test60\_2d).



Gambar 46. Prediksi SVR 60%-40%

#### Evaluasi Model

Evaluasi model adalah proses penilaian dan evaluasi kinerja suatu model berdasarkan sejumlah matriks atau matriks evaluasi. Pada kode di bawah ini, dilakukan pencarian nilai MSE, MAE, RMSE, dan MAPE pada model SVR



Gambar 47. Evaluasi Matrik Model SVR skenario 60% - 40%

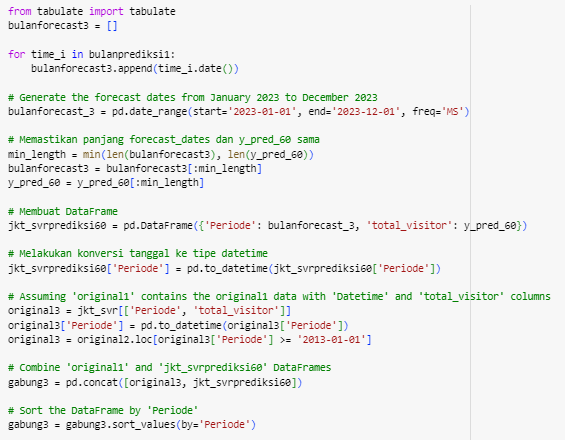
Untuk membandingkan nilai aktual (y\_test\_60) dengan nilai yang diprediksi (y\_pred\_60) dilakukan pembuatan visual menggunakan scatterplot.



Gambar 48. Scatter Plot Perbandingan nilai aktual dan nilai prediksi

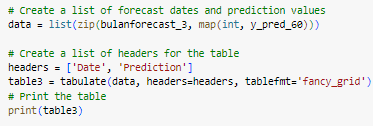
#### Forecast

Kode di bawah ini akan menghasilkan DataFrame ‘gabung3’ yang berisi data aktual dan data prediksi yang telah diurutkan berdasarkan tanggal. Kode ini digunakan untuk menggabungkan data asli dengan data yang telah diprediksi dan kemudian mengurutkan DataFrame yang telah digabung berdasarkan kolom 'Periode'. Langkah ini berguna untuk visualisasi data asli dan hasil prediksi secara bersamaan.



Gambar 49. Menggabungkan Data SVR 60%-40%

Selanjutnya, dijalankan kode berikut untuk memberikan tanggal prediksi dan nilai prediksi pengunjung dalam bentuk tabel. DataFrame ‘data’ menggabungkan tanggal prediksi (bulanforecast\_3) dengan nilai prediksi (y\_pred\_60) ke dalam sebuah list. zip digunakan untuk menggabungkan dua list. Pada kode tersebut terdapat data frame ‘bulanprediksi1’, hal ini merupakan pemanggilan kode yang telah dijalankan pada gambar 24.



Gambar 50. Kode Forecast SVR 60%-40%

Selanjutnya, untuk menampilkan peramalan menggunakan grafik, dapat menggunakan kode di bawah ini. Kode ini bertujuan untuk membuat sebuah plot garis (line plot) menggunakan library ‘seaborn’ untuk memvisualisasikan data aktual dan hasil prediksi (forecast) jumlah visitor.



Gambar 51. Kode Forecast SVR 60%-40% Bentuk Grafik

### Perbandindangan Hasil Setiap Skenario SVR

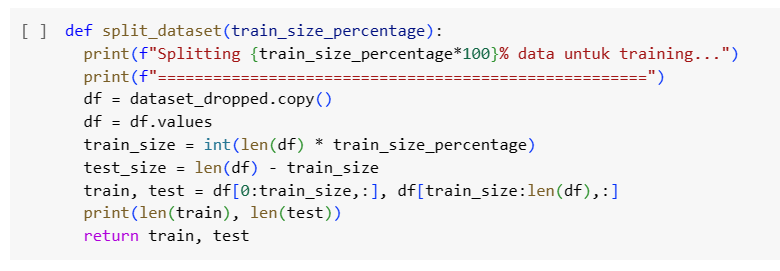
| Nilai MSE, MAE, dan MAPE | | |
| --- | --- | --- |
|  |  |  |

Tabel 1. Perbandingan Hasil Setiap Skenario SVR

## Fungsi General Untuk RNN, LSTM, dan GRU

#### Split dataset

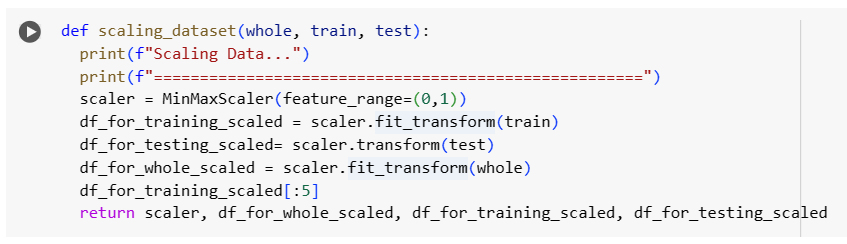
Membagi dataset, bertujuan untuk membangun dan mengukur model secara lebih akurat, mengurangi risiko overfitting, dan memastikan bahwa model yang dihasilkan dapat diterapkan dengan baik pada situasi dunia nyata. outputnya adalah dua set data terpisah yang siap digunakan untuk melatih dan menguji model machine learning



Gambar 52. *Splitting* Dataset

#### Scaling dataset

Setelah *splitting* data, perlu dilakukan *scaling* dataset yang diinginkan agar skalanya dari 0 hingga 1. *Scaling* dataset bertujuan untuk memudahkan model belajar dari data tersebut dan .memperbagus performa model yang dihasilkan



Gambar 53. *Scaling* Dataset

#### Buat Timestep atau Sequencing

Proses pembuatan *timestep* atau *sequencing* pada dataset yang telah di*-scaling* adalah langkah penting dalam mempersiapkan data untuk model time-series. Pembuatan *timestep* atau *sequencing* pada dataset yang telah di-*scaling* bertujuan untuk membantu model memahami pola dan tren dalam dataset.



Gambar 54. Buat Timestep atau Sequencing

#### Hyperparameter Tuning

Hyperparameter tuning dilakukan untuk mencari kombinasi nilai hyperparameter yang memberikan kinerja model terbaik pada data uji atau validasi. Tujuannya adalah mencapai model yang lebih baik dalam hal akurasi, generalisasi, atau kinerja keseluruhan. Kode di bawah ini melakukan pencarian parameter terbaik pada model ltsm, gru dan rnn.



Gambar 55. Hyperparameter Tuning

#### Training Model

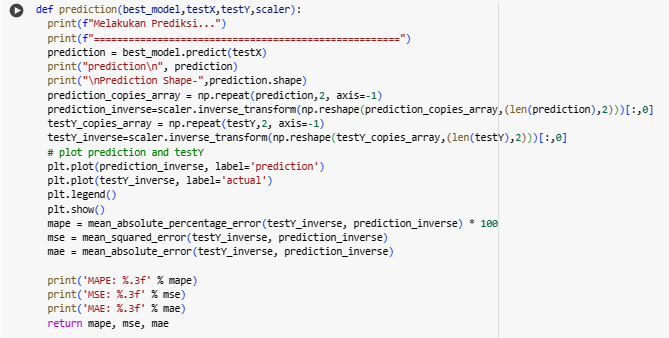
Setelah menentukan parameter terbaik dengan menggunakan fungsi, langkah selanjutnya adalah melakukan training model dengan memanfaatkan callback dari fungsi `TerminateOnBaseline`. Fungsi ini berperan sebagai callback untuk membuat *checkpoint* model pada kondisi (`val\_loss`) mencapai atau melampaui nilai terbaik. callback ini membantu mengoptimalkan pelatihan model dengan memastikan bahwa model yang akan digunakan merupakan model terbaik berdasarkan nilai *val\_loss*. Setelah training selesai, fungsi `model` mengembalikan model terbaik dan histori pelatihan untuk evaluasi performa model lebih lanjut. Selain itu, *callback* tersebut juga akan otomatis berhenti apabila telah melewati *baseline* tergantung mode *min* atau *max.*



Gambar 56. Fungsi untuk melakukan training model

#### Prediksi

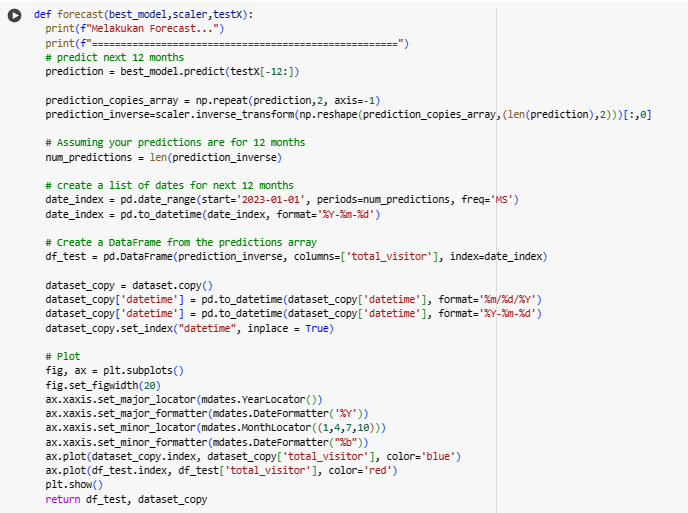
Selanjutnya adalah prediksi pada dataset menggunakan model yang sudah dilatih sebelumnya dan menampilkan visualisasi perbandingan antara hasil prediksi dan nilai aktual pada dataset uji. Visualisasi ini digunakan untuk melihat sejauh mana hasil prediksi model mendekati nilai aktual. Selain itu, dilakukan perhitungan metrik evaluasi kinerja, seperti Mean Absolute Percentage Error (MAPE), Mean Squared Error (MSE), dan Mean Absolute Error (MAE), yang memberikan gambaran lebih lanjut tentang akurasi prediksi model. Dengan informasi ini, dapat diukur seberapa baik model dalam memperkirakan nilai sesuai dengan data sebenarnya.



Gambar 57. Melakukan Prediksi, Visualisasi, dan Evaluasi Error

#### Forecasting

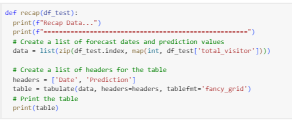
Kemudian menggunakan model yang sama dilakukan *forecasting* untuk 12 bulan berikutnya. Tujuan dari kode ini adalah memberikan visualisasi hasil peramalan model untuk12 bulan ke depan dan membandingkannya dengan data aktual.



Gambar 58. Forecasting 12 bulan kedepan

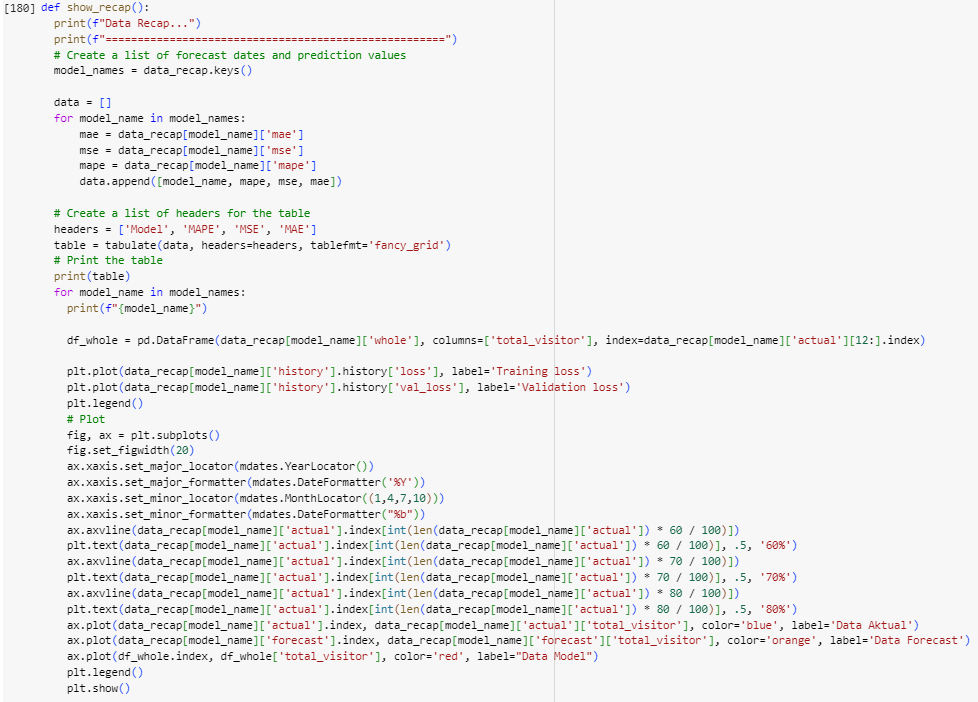
#### Rekapitulasi Data

Fungsi ini bertujuan untuk memberikan rekapitulasi hasil *modelling* untuk masing-masing model dan untuk keseluruhan akhir haisl peramalan yang telah dilakukan. Fungsi ini mencetak tabel yang menampilkan tanggal dan nilai prediksi (total\_visitor) untuk setiap tanggal hasil peramalan. Berikut adalah rekapitulasi masing-masing model:



Gambar 59. *Recap* masing-masing model

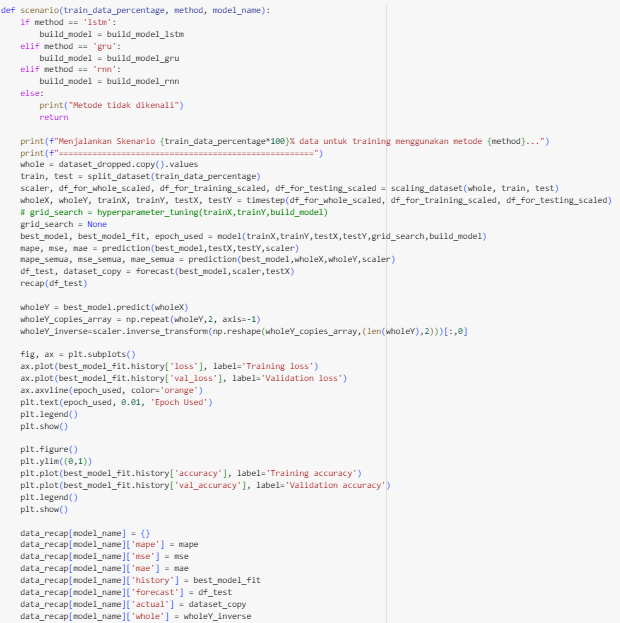
Kode di bawah ini bertujuan untuk menampilkan rekapitulasi semua model yang ada.



Gambar *60. Recap* semua model

#### Fungsi utama skenario

Kode di bawah ini merupakan kode yang mewakili suatu skenario lengkap untuk melatih, mengevaluasi, dan meramalkan berdasarkan beberapa parameter yang diberikan. Kode dibawah ini nantinya dapat digunakan untuk berbagai skenario pembagian dataste yang diinginkan.

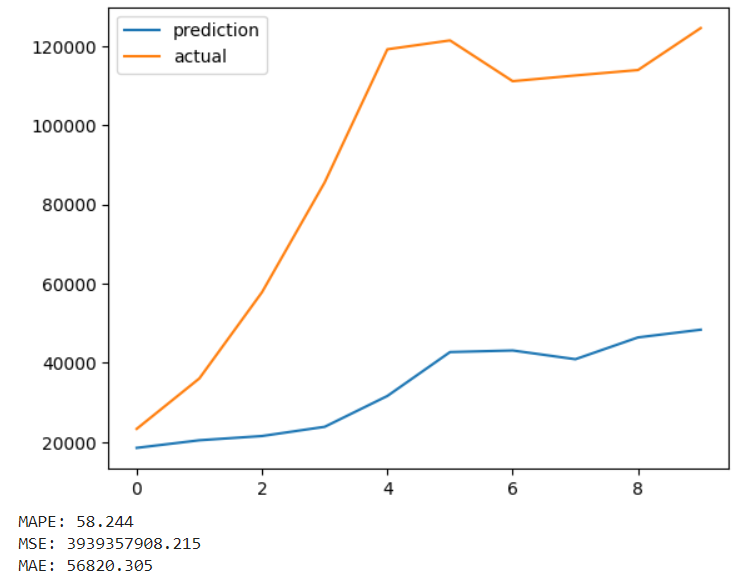


Gambar 61. Fungsi Utama untuk Semua Skenario

## RNN

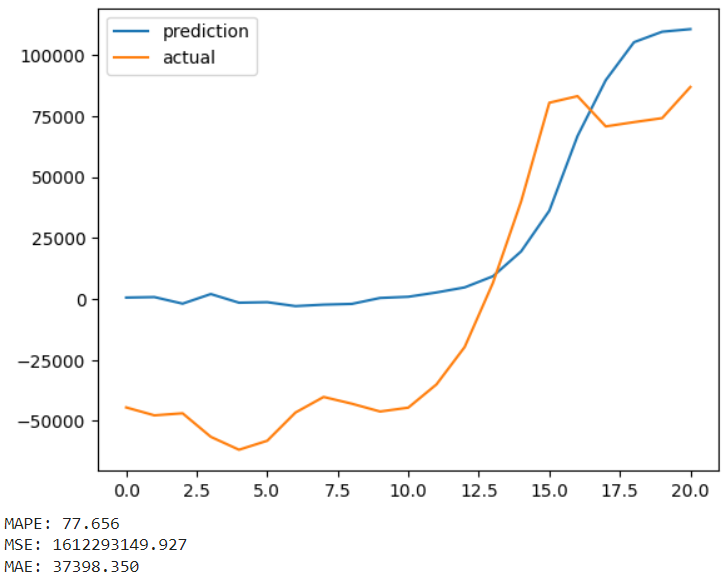
1. Hasil Skenario

#### 80% Train, 20% Test



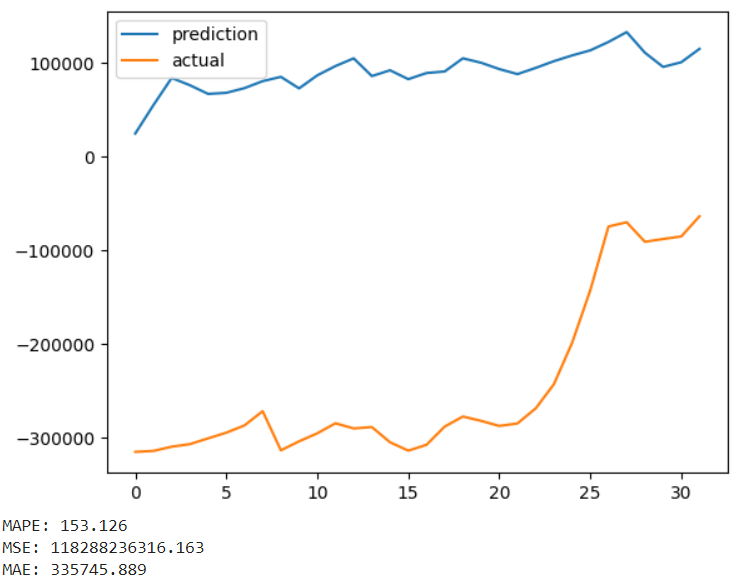
Gambar 62. Perbandingan Prediction dan Actual

#### 70% Train, 30% Test



Gambar 63. Perbandingan Prediction dan Actual

#### 60% Train, 40% Test



Gambar 64. Perbandingan Prediction dan Actual

### Perbandingan Hasil Setiap Skenario RNN

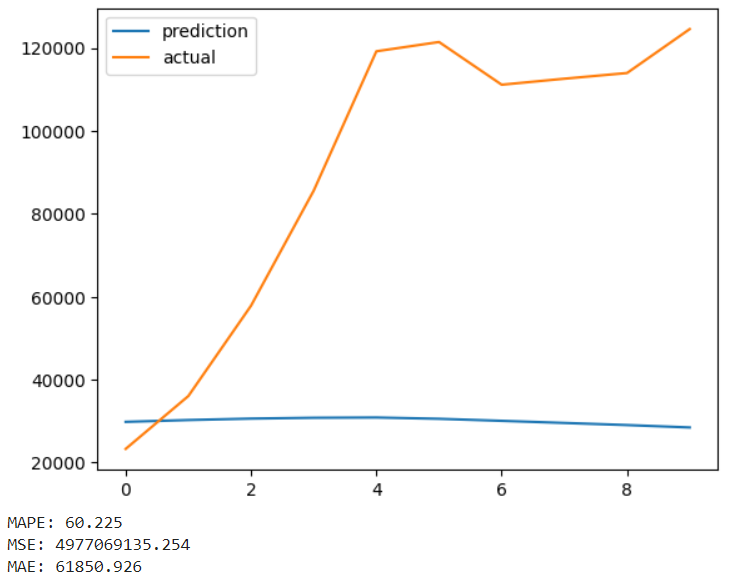
| * Nilai MSE, MAE, dan MAPE | | |
| --- | --- | --- |
| Skenario 80% Train | Skenario 70% Train | Skenario 60% Train |
| * Visualisasi Trainning Loss | | |
| Skenario 80% Train | Skenario 70% Train | Skenario 60% Train |

Tabel 2. Perbandingan Hasil Setiap Skenario RNN

## LSTM

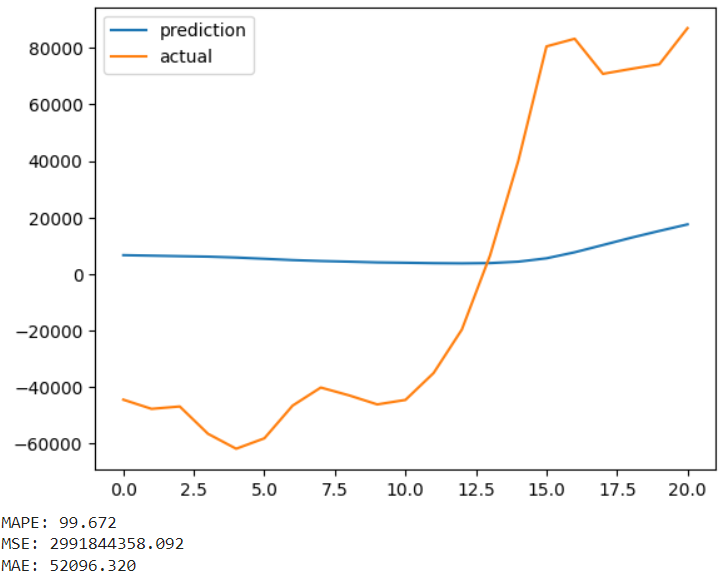
1. Hasil Skenario

#### 80% Train, 20% Test



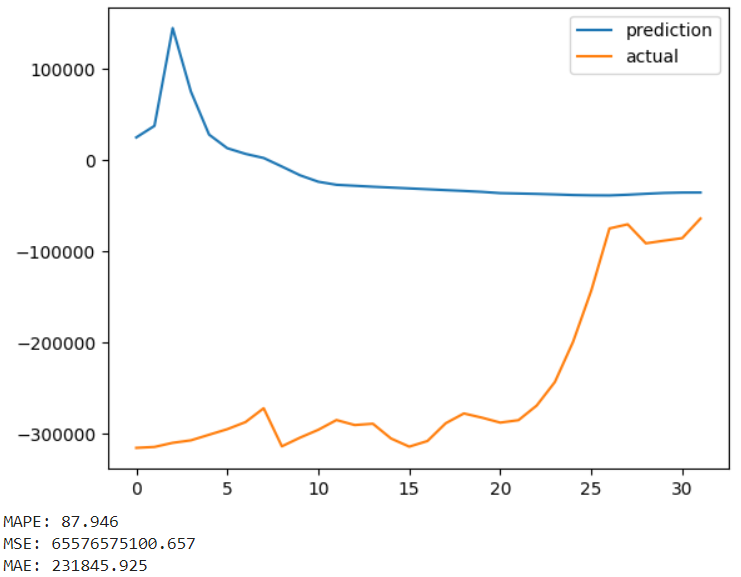
Gambar 65. Perbandingan Prediction dan Actual

#### 70% Train, 30% Test



Gambar 66. Perbandingan Prediction dan Actual

#### 60% Train, 40% Test



Gambar 67. Perbandingan Prediction dan Actual

1. Perbandingan Hasil Setiap Skenario LSTM

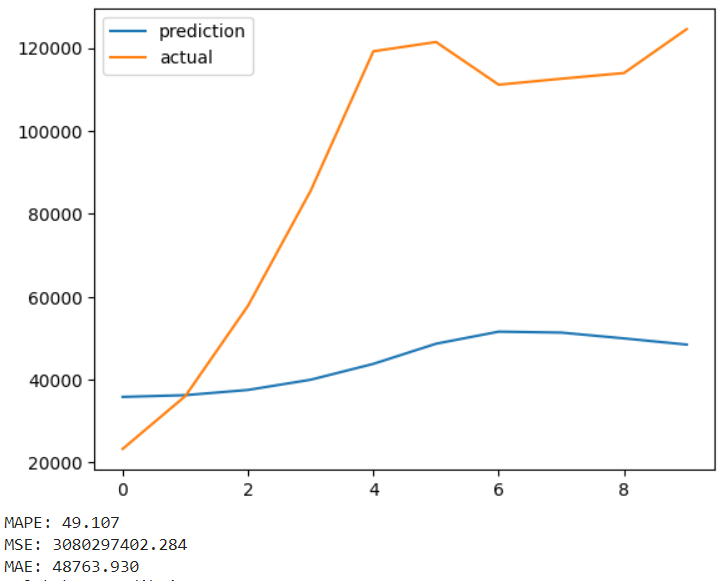
| * Nilai MSE, MAE, dan MAPE | | |
| --- | --- | --- |
| Skenario 80% Train | Skenario 70% Train | Skenario 60% Train |
| * Visualisasi Training Loss | | |
| Skenario 80% Train | Skenario 70% Train | Skenario 60% Train |

Tabel 3. Perbandingan Hasil Setiap Skenario LSTM

## GRU

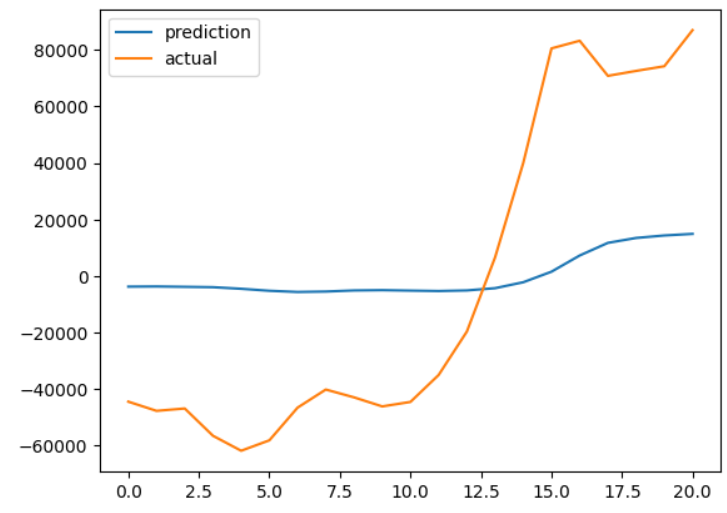
1. Hasil Skenario

#### 80% Train, 20% Test



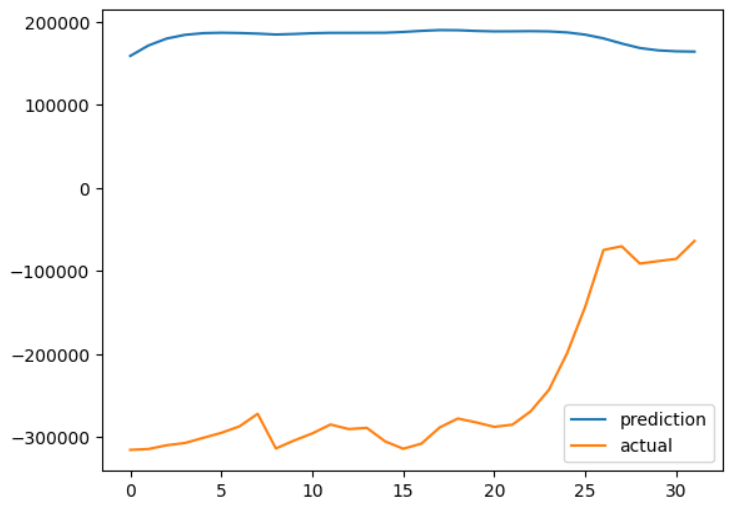
Gambar 68. Perbandingan Prediction dan Actual

#### 70% Train, 30% Test



Gambar 69. Perbandingan Prediction dan Actual

#### 60% Train, 40% Test



Gambar 70. Perbandingan Prediction dan Actual

1. Perbandingan Hasil Setiap Skenario GRU

| * Nilai MSE, MAE, dan MAPE | | |
| --- | --- | --- |
| Skenario 80% Train | Skenario 70% Train | Skenario 60% Train |
| * Visualisasi Training Loss | | |
| Skenario 80% Train | Skenario 70% Train | Skenario 60% Train |

Tabel 4. Perbandingan Hasil Setiap Skenario GRU

## Perbandingan Hasil Metrik Semua Model

Di bawah ini merupakan perbandingan dari evaluasi Forecast. Mean Absolute Percentage Error (MAPE), merupakan pengukuran persentase rata-rata kesalahan prediksi dibandingkan dengan nilai sebenarnya. Mean Squared Error (MSE) merupakan metrik pengukuran rata-rata dari kuadrat kesalahan prediksi. Mean Absolute Error (MAE) mengukur rata-rata dari nilai absolut dari kesalahan prediksi.

| Model | MAPE (%) | MSE | MAE |
| --- | --- | --- | --- |
| SVR - 80:20 | 58.51 | 3321380940.4 | 45526.44 |
| SVR - 70:30 | 46.82 | 5773686462.19 | 62276.33 |
| SVR - 60:40 | 60.21 | 5519396462.23 | 60182.76 |
| RNN - 80:20 | 39.4021 | 1.58012e+09 | 33640.1 |
| RNN - 70:30 | 77.6562 | 1.61229e+09 | 37398.4 |
| RNN - 60:40 | 153.126 | 1.18288e+11 | 335746 |
| LSTM - 80:20 | 67.6346 | 3.62077e+09 | 50605.8 |
| LSTM - 70:30 | 148.085 | 4.51619e+09 | 62417.2 |
| LSTM - 60:40 | 102.48 | 7.40563e+10 | 254206 |
| GRU - 80:20 | 57.7628 | 3.87333e+09 | 51855 |
| GRU - 70:30 | 92.3682 | 2.5032e+09 | 346976.1 |
| GRU - 60:40 | 195.444 | 1.91292e+11 | 427527 |

Tabel 5. Perbandingan Hasil Metrik

# Hasil Forecast

## **SVR**

| * Perbandingan antara Data Aktual dengan Nilai yang diprediksi | | |
| --- | --- | --- |
| 80% Train -20% Test | 70% Train - 30% Test | 60% Train - 40% Test |
| * Visualisasi Hasil Prediksi Model SVR untuk 12 Periode Berikutnya | | |
| 80% Train -20% Test | 70% Train - 30% Test | 60% Train - 40% Test |
| * Tabel Data Hasil Prediksi | | |
| 80% Train - 20% Test | 70% Train - 30% Test | 60% Train - 40% Test |

Tabel 6. Hasil Forecast 3 Skenario SVR

## **RNN**

| * Perbandingan antara Data Aktual dengan Nilai yang diprediksi | | |
| --- | --- | --- |
| 80% Train - 20% Test | 70% Train - 30% Test | 60% Train - 40% Test |
| * Visualisasi Hasil Prediksi Model SVR untuk 12 Periode Berikutnya | | |
| 80% Train - 20% Test | 70% Train - 30% Test | 60% Train - 40% Test |
| * Tabel Data Hasil Prediksi | | |
| 80% Train -20% Test | 70% Train - 30% Test | 60% Train - 40% Test |

Tabel 7. Hasil Forecast 3 Skenario RNN

## **LSTM**

| * Perbandingan antara Data Aktual dengan Nilai yang diprediksi | | |
| --- | --- | --- |
| 80% Train - 20% Test | 70% Train - 30% Test | 60% Train - 40% Test |
| * Visualisasi Hasil Prediksi Model SVR untuk 12 Periode Berikutnya | | |
| 80% Train - 20% Test | 70% Train - 30% Test | 60% Train - 40% Test |
| * Tabel Data Hasil Prediksi | | |
|  |  |  |

Tabel 8. Hasil Forecast 3 Skenario LSTM

## **GRU**

| * Perbandingan antara Data Aktual dengan Nilai yang diprediksi | | |
| --- | --- | --- |
| 80% Train - 20% Test | 70% Train - 30% Test | 60% Train - 40% Test |
| * Visualisasi Hasil Prediksi Model SVR untuk 12 Periode Berikutnya | | |
| 80% Train - 20% Test | 70% Train - 30% Test | 60% Train - 40% Test |
| * Tabel Data Hasil Prediksi | | |
| 80% Train - 20% Test | 70% Train - 30% Test | 60% Train - 40% Test |

Tabel 9. Hasil Forecast 3 Skenario GRU

# Hasil Analisis dan Kesimpulan

1. Hasil Analisis

Dalam percobaan ini, dilakukan uji coba terhadap dua jenis model, yakni Support Vector Regression (SVR) dan deep learning menggunakan arsitektur Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), dan Gated Recurrent Unit (GRU). Dataset yang digunakan adalah data time series mengenai jumlah pengunjung dan kapasitas akomodasi di Provinsi Jakarta. Berikut ini hasil analisis per-model :

1. SVR

Berdasarkan hasil evaluasi matriks pada tabel 5, terlihat bahwa skenario 2 (Data split 70% - 30%) memiliki nilai MAPE yang paling rendah sebesar 46.82%, yang menunjukkan akurasi prediksi yang lebih baik dibandingkan dengan skenario lainnya. Oleh karena itu, skenario 2 dengan menggunakan model SVR terlihat paling sesuai di antara ketiga skenario tersebut.

1. RNN

Berdasarkan hasil evaluasi matriks pada tabel 5, terlihat bahwa skenario 1 (Data split 80% - 20%) memiliki nilai MAPE yang paling rendah yaitu sebesar 39.40%, yang menunjukkan akurasi prediksi yang lebih baik dibandingkan dengan skenario lainnya. Oleh karena itu, skenario 1 dengan menggunakan model SVR terlihat paling sesuai di antara ketiga skenario tersebut.

1. LSTM

Berdasarkan hasil evaluasi matriks pada tabel 5, terlihat bahwa skenario 1 (Data split 80% - 20%) memiliki nilai MAPE yang paling rendah sebesar 67.63% , yang menunjukkan akurasi prediksi yang lebih baik dibandingkan dengan skenario lainnya. Oleh karena itu, skenario 1 dengan menggunakan model SVR terlihat paling sesuai di antara ketiga skenario tersebut.

1. GRU

Berdasarkan hasil evaluasi matriks pada tabel 5, terlihat bahwa skenario 1 (Data split 80% - 20%) memiliki nilai MAPE yang paling rendah sebesar 57.76%, yang menunjukkan akurasi prediksi yang lebih baik dibandingkan dengan skenario lainnya. Oleh karena itu, skenario 1 dengan menggunakan model SVR terlihat paling sesuai di antara ketiga skenario tersebut.

1. Kesimpulan

Berdasarkan 4 metode yang telah digunakan dalam 3 skenario yang berbeda dapat diambil kesimpulan pada masing-masing skenario :

* Metode SVR terbaik menggunakan skenario 2 karena memiliki mape sebesar 46.82%
* Metode RNN terbaik menggunakan skenario 1 karena memiliki mape sebesar 39.4%
* Metode LSTM terbaik menggunakan skenario 1 karena memiliki mape sebesar 67.63%
* Metode GRU terbaik menggunakan skenario 1 karena memiliki mape sebesar 57.76%

Berdasarkan poin diatas maka didapatkan metode RNN paling optimal digunakan pada dataset Jakarta 2015 - 2022. Berdasarkan hasil evaluasi matriks pada tabel 5, selain RNN metode yang optimaladalah metodeSVR**.** Pada eksperimen ini, RNN terbukti menjadi metode yang paling efektif dalam menyelesaikan tugas yang diberikan. Hal itu dapat terjadi, karena model RNN cocok untuk menangani data yang memiliki hubungan ketergantungan waktu atau urutan serta lebih sederhana dan ringan secara komputasi. Dalam ketiga skenario tersebut, ketiga metode deep learning paling baik pada skenario 80% Data Training dan 20% Data Splitting. Hal ini dapat terjadi karena dengan menggunakan 80% dari data sebagai data pelatihan, model memiliki lebih banyak data untuk "belajar". Dalam beberapa kasus, lebih banyak data pelatihan dapat meningkatkan kapabilitas model untuk menangkap pola yang lebih kompleks dan umum.

Meskipun demikian, perlu dicatat bahwa metode selain RNN juga memiliki potensi menjadi yang terbaik dalam skenario tertentu. Keefektifan suatu metode sangat tergantung pada konteks dan karakteristik khusus dari data atau tugas yang dihadapi. Sehingga, meskipun RNN mendominasi dalam percobaan ini, ada kemungkinan metode lain dapat memberikan hasil yang optimal dalam skenario dan penggunaan parameter yang berbeda.

# Referensi

1. Saadah, S.,Fakhira Zahra Z., Hasna Haifa Z. (2021). Support vector regression (SVR) Dalam Memprediksi Harga Minyak Kelapa Sawit di Indonesia Dan Nilai Tukar Mata uang EUR/USD. *Journal of Computer Science and Informatics Engineering (J-Cosine)*, *5*(1), 85–92. https://doi.org/10.29303/jcosine.v5i1.403
2. Stasi, C. (2016) ‘Non-invasive assessment of liver fibrosis: Between prediction/prevention of outcomes and cost-effectiveness’, *World Journal of Gastroenterology*, 22(4), p. 1711. doi:10.3748/wjg.v22.i4.1711.
3. T. S. (2023). An Introduction to Support Vector Regression (SVR). Retrieved from <https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2>
4. Koehrsen, W. (2018). Recurrent Neural Networks by Example in Python. Retrieved from <https://towardsdatascience.com/recurrent-neural-networks-by-example-in-python-ffd204f99470>
5. Girsang, A. S. (2019). Long Short Term Memory (LSTM). Retrieved from https://mti.binus.ac.id/2019/12/02/long-short-term-memory-lstm/
6. Team, A., & Bunga. (2022). Apa Itu Long Short Term Memory Network (LSTM)? Retrieved from <https://algorit.ma/blog/lstm-network-adalah-2022/>
7. Papers with Code - GRU Explained. (n.d.). Retrieved from https://paperswithcode.com/metho