Laporan Code Challenge 2025
Tim Semoga Berkah

Disusun Oleh:

Richard Edgar Gonassis Telkom University Surabaya

Steven Andre Gonassis Telkom University Surabaya

I. Latar Belakang Masalah

Mobilitas perkotaan merupakan salah satu elemen penting dalam *smart city.* Pertumbuhan penduduk menyebabkan kota-kota di dunia menghadapi tantangan besar dalam mengelola transportasi publik, kemacetan lalu lintas, hingga kebutuhan mobilitas masyarakat. Tanpa perencanaan mobilitas yang efektif, mobilitasnya menjadi tidak efisien juga, sehingga menyebabkan kualitas hidup masyarakat yang tidak maksimal.

Berdasarkan masalah tersebut, pengumpulan, analisis, hingga pengolahan data, menjadi sangat penting. Dengan memanfaatkan AI, kota dapat memperoleh wawasan prediktif yang berguna untuk mengambil keputusan berbasis data dalam mengoptimalkan layanan publik. Kompetisi CODE Challenge yang bertema *Smart Society* menjadi simulasi nyata untuk membuat model AI yang bisa memprediksi jumlah perjalanan harian (*trips_thousands*) dari data yang diberikan, seperti: demografi, aktivitas urban, cuaca, dan infrastruktur transportasi.

II. Tujuan dan Manfaat

1. Tujuan

   Tujuan dari partisipasi dalam kompetisi ini adalah:

   a. Membangun Model AI Prediktif
   b. Mengintegrasikan Berbagai Sumber Data
   c. Menghasilkan Solusi Format *Submission* yang *Valid*
   d. Mensimulasikan Pengambilan Keputusan di *Smart City*

2. Manfaat

   a. Mengembangkan *Hard Skill*
   b. Memahami Konteks Perencanaan *Smart City*

III. Explonatory Data Analysis (EDA)

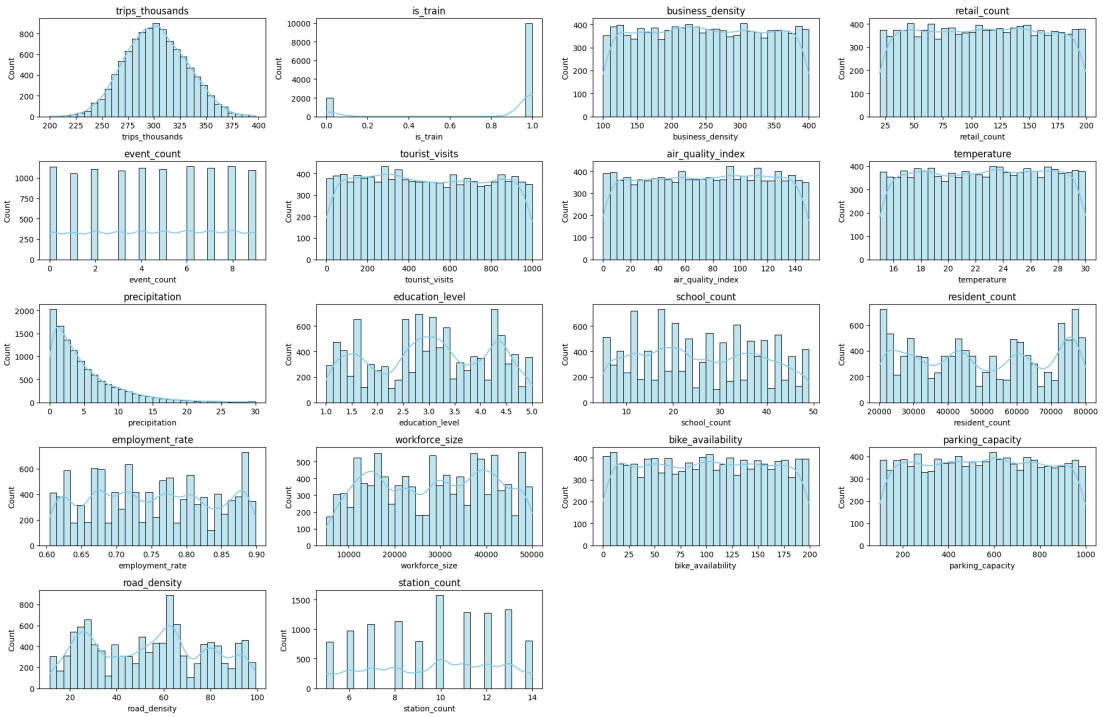
1. Penggabungan Dataset

   Langkah awal dalam EDA kami dilakukan dengan menggabungkan semua data yang diberikan baik dari data train maupun data test. Data digabungkan menggunakan kolom *'date'* dan atau *'zone_id'*. Semua data yang digabung menggunakan 'date', selalu kami ubah dulu ke dalam format tanggal sebelum digabung.

   a. Aktivitas Urban: semua data digabungkan menggunakan *'date'* dan *'zone_id'*.
   b. Cuaca: semua data digabungkan menggunakan *'date'* dan *'zone_id'*.
   c. Demografi: semua data di demografi digabungkan menggunakan '*zone_id'*, karena tidak memiliki tanggal.
   d. Infrastruktur Transportasi:
      i. *Bike_sharing* dan *parking_data* digabungkan menggunakan *'zone_id'* dan '*date'*.
      ii. Road_network dan transit_stations digabungkan menggunakan *'zone_id'* saja.

2. Cek Distribusi Data

   Setelah melakukan penggabungan dataset, dihasilkan *df_merged*. Kami mengecek distribusi data untuk menentukan apakah datanya itu berdistribusi normal atau skewed.



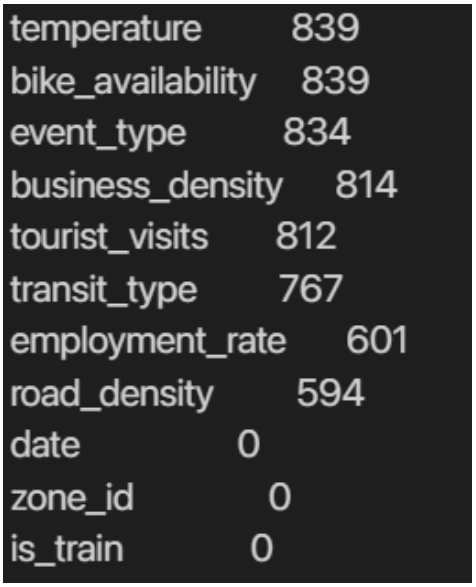
Gambar 1: Distribusi Data

Berdasarkan Gambar 1, ada beberapa insight yang bisa diambil, yaitu:

1. Target variabel, *trips_thousands*, berbentuk mendekati distribusi normal, dengan rata-rata sekitar 300. Distribusi ini relatif simetris, sehingga memudahkan penerapan model regresi.
2. Aktivitas urban seperti *business_density, retail_count, event_count, dan tourist_visits* distribusinya mendekati seragam, artinya datanya cukup merata antar zona. *Event_count* menunjukkan distribusi integer terbatas, yang cukup umum.
3. Cuaca seperti *temperature* dan *air_quality_index* punya distribusi yang relatif merata. Tapi, precipitation bersifat *right-skewed*, dengan sebagian besar nilai di bawah 10 mm. Artinya, hujan deras cukup jarang terjadi dan merupakan outlier.
4. Pada demografi, *education_level* menunjukkan distribusi multimodal, artinya tingkat pendidikan tiap zona bervariasi. *Resident_count* menunjukkan penyebaran yang luas. *Employment_rate* antara 60%-90% dengan variasi yang cukup tinggi. Kemudian, *workforce_size* juga menunjukkan distribusi yang cukup merata.
5. Infrastruktur Transportasi, yaitu *bike_availability* dan *parking_capacity* menunjukkan distribusi yang cukup merata, tapi *station_count* dan *road_density* cukup bervariasi.

Kesimpulannya, kami punya normalitas pada target, kemudian variasi antar-zona bisa digunakan di model untuk generalisasi.

3. Missing and Duplicate Data Handling
    a. Missing Values



Gambar di atas menunjukkan jumlah *missing values* masing-masing fitur. *Handling missing values* nya, kami pertama pisahkan fitur kategorikal dan numerik. Untuk numerik, semua diisi menggunakan median kecuali *trips_thousands*, dibiarkan kosong karena belum diketahui (target yang harus diprediksi). Median dipilih karena paling robust terhadap outlier. Pada nilai kategorikal, strategi pengisian *missing values* nya menggunakan modus untuk menghindari kategori baru yang tidak alami, sederhana, dan efisien.

b. Duplicate Data

Tidak ada data duplikat di dataset ini.

4. Data Encoding

Pada dataset ini, fitur kategorikal harus dikonversi ke format numerik agar dapat diproses oleh algoritma machine learning. Model KNN tidak dapat menangani data non-numerik secara langsung karena menggunakan pengukuran jarak antar fitur. Jadi, butuh encoding supaya data kategorikal tetap dapat berkontribusi dalam proses pembelajaran model.

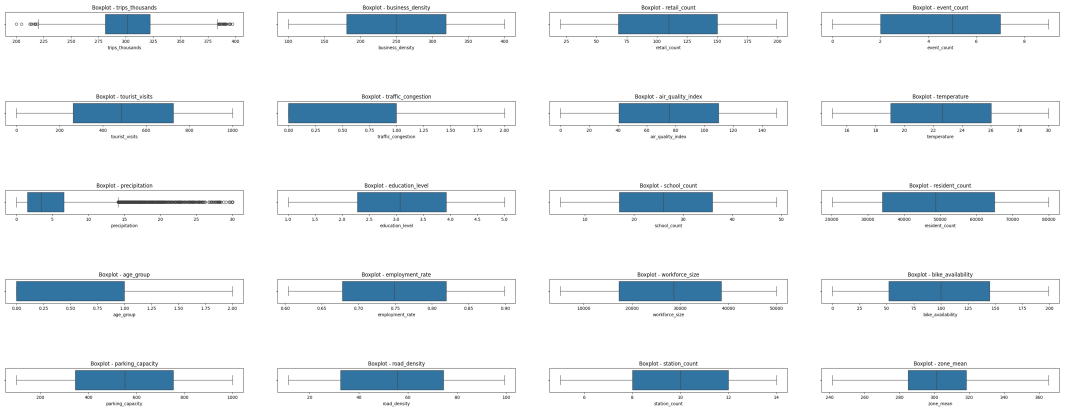
Kami menggunakan dua jenis encoding di dataset ini, yaitu *Ordinal Encoding* (*Manual Mapping*) dan *One-Hot Encoding*. Ordinal encoding digunakan untuk fitur kategorikal yang punya urutan bermakna, seperti *age_group* yang usianya diklasifikasikan sebagai *Youth(Y), Adult (A),* dan *Senior(S)*. Sedangkan *One-Hot Encoding* digunakan untuk fitur kategorikal nominal, yang tidak punya urutan bermakna, seperti *event_type, commute_preference,* dan *transit_type.*

Kami juga melakukan standarisasi huruf kapital dan menambahkan *zone_mean,* rata-rata trips tiap zona, untuk menggantikan zone_id di dataset. Ini dilakukan untuk mendapatkan korelasi antara zona dan fitur-fitur lain. *Zone_mean* dilakukan dengan cara mengelompokkan seluruh data berdasarkan masing-masing *zone_id,* kemudian dihitung rata-rata *trips_thousands* nya. *Zone_id* dan *zone_mean* masing-masing punya 200 *unique values*. Maka, dapat disimpulkan bahwa tiap

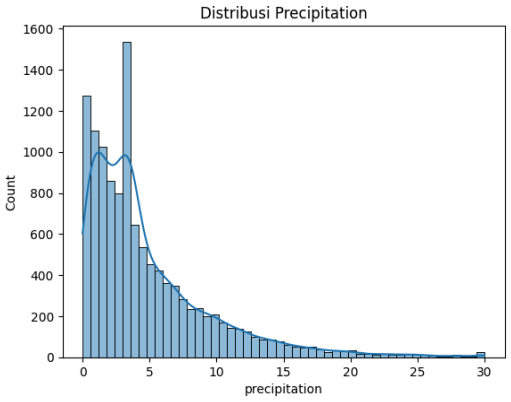
zone_id punya *zone_mean* yang berbeda-beda. Sehingga kami menambahkan *zone_mean* ke *df_merged*.

5. Outlier Handling

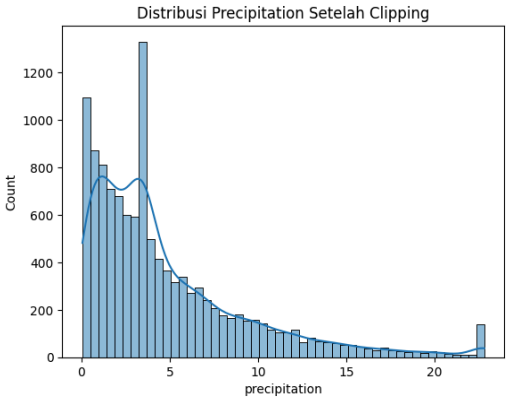
Sebaran data menggunakan boxplot yaitu seperti ini:



Berdasarkan *boxplot* di atas, maka ada 2 fitur yang punya *outlier*, yaitu *precipitation* dan *trips_thousands*. *Outlier* pada *trips_thousands* tidak dihilangkan, karena mencerminkan kondisi nyata dan bukan *noise*. Selain itu, *outlier* tetap dipertahankan untuk generalisasi model. *Outlier* yang perlu di-*handle* adalah untuk fitur *precipitation*.



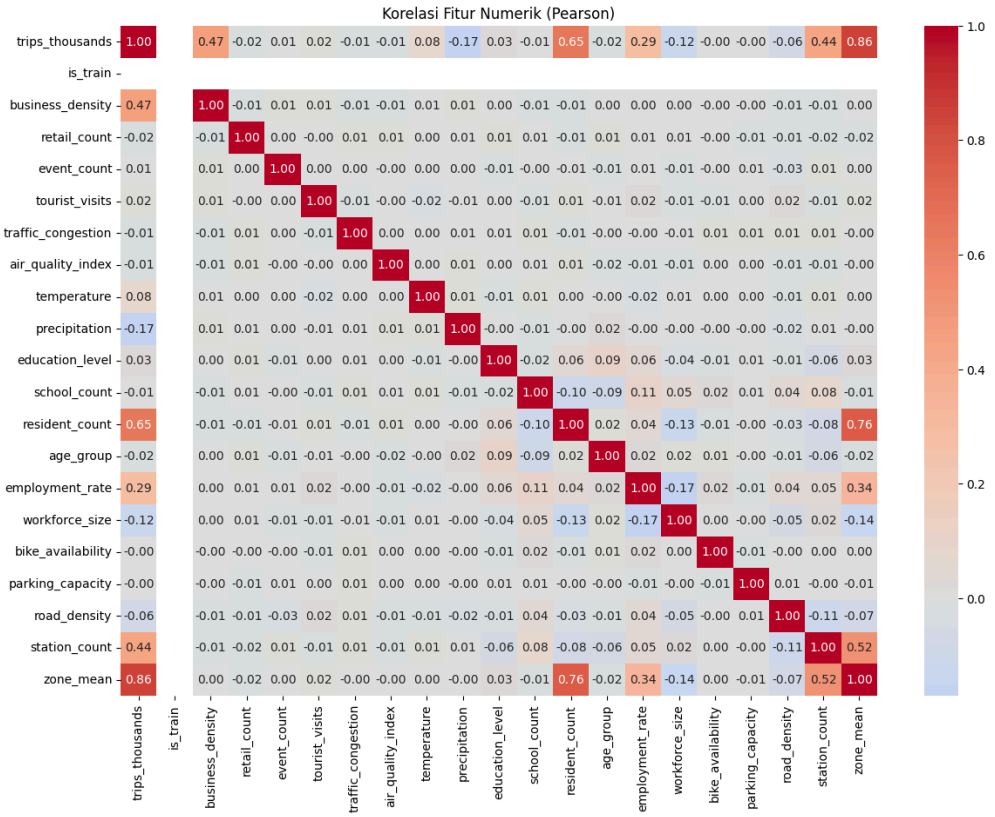
Gambar 2: Distribusi Awal Precipitation



Gambar 3: Distribusi Precipitation Setelah Clipping

Outlier pada *precipitation* di-*handle* dengan melakukan *clipping,* yaitu membatasi nilai ekstrem menjadi batas atas tertentu. Hasilnya, nilai maksimum pada fitur *precipitation* dapat dibatasi menjadi 22,8. Metode *clipping* dipilih untuk membangun model yang stabil dan akurat. Dengan *clipping,* kami membatasi pengaruh berlebih dari nilai-nilai ekstrem yang dapat menyebabkan model menjadi tidak stabil atau overfit pada data anomali. Berbeda dengan metode *dropping*, *clipping* memastikan tidak ada kehilangan data observasi, sehingga informasi penting dari fitur lain pada baris yang sama tetap terjaga. Pendekatan ini dapat mencegah distorsi skala.

6. Correlation



Gambar 4: Heatmap Korelasi Pearson

Gambar 4 menunjukkan korelasi antar fitur numerik, dengan warna merah menunjukkan korelasi positif, biru menunjukkan korelasi negatif, dengan intensitas warna menunjukkan kekuatan korelasi (dari -1 hingga +1). Beberapa insight utama terhadap *trips_thousands:*

a. *Resident_count* punya korelasi +0.65, artinya semakin tinggi populasi, perjalanan cenderung lebih banyak

b. Business_density punya korelasi +0.47, artinya aktivitas ekonomi padat mendorong peningkatan mobilitas

c. Station_count punya korelasi +0.44, artinya banyaknya stasiun transit berhubungan dengan mobilitas.

d. Precipitation punya korelasi -0.17, artinya curah hujan sedikit berpengaruh pada mobilitas, mungkin karena orang menghindari hujan.

IV. Metodologi dan Modeling

Tujuan utama dari metodologi ini adalah membangun model kecerdasan buatan yang dapat memprediksi *trips_thousands* secara akurat berdasarkan data multivariabel yang kompleks, meliputi demografi, cuaca, infrastruktur, dan aktivitas urban. Untuk menangani kompleksitas ini, kami menggunakan kombinasi *feature engineering Broad Learning System (BLS)* dan model prediksi *K-Nearest Neighbors (KNN)*. Pendekatan ini bertujuan untuk mengidentifikasi hubungan non-linear antar fitur tanpa perlu *training* neural network.

Alur metodologi kami, yaitu:

a. Data preprocessing

1. Imputasi nilai numerik dengan median
2. Imputasi nilai kategorikal dengan modus

3. *Encoding* menggunakan manual encoding dan *One-Hot-Encoding.*
4. *Clipping* pada *precipitation*

b. Standarisasi fitur.
   1. Semua fitur numerik dinormalisasi menggunakan *StandardScaler* untuk menyamakan skala semua fitur.

c. Rekayasa fitur menggunakan BLS
   1. *Polynomial feature expansion*, yaitu fitur ditransformasikan ke bentuk polinomial derajat 2 untuk menangkap interaksi antar fitur
   2. *Feature mapping node*, untuk membentuk representasi non-linier dari fitur polinomial melalui *random projection* dan aktivasi *tanh*.
   3. *Enhancement node*, layer tambahan BLS untuk memperluas kompleksitas representasi fitur dengan transformasi non-linear lanjutan.
   4. Gabungan semua fitur awal, polinomial, dan representasi non-linier digunakan sebagai input akhir untuk model.

d. Model building dengan KNN Regressor

   Model regresi yang digunakan adalah KNN dengan parameter: *n_neighbors* = 5 dan *weights = 'uniform'* sebagai *baseline model*. Kami memilih KNN karena mampu bekerja baik dalam situasi di mana distribusi data tidak diketahui dan pola relasi antar fitur bersifat lokal.

e. Evaluasi Model

   Evaluasi Model menggunakan RMSE (Root Mean Squared Error) untuk menguji akurasi prediksi model.

f. Hyperparameter Tuning

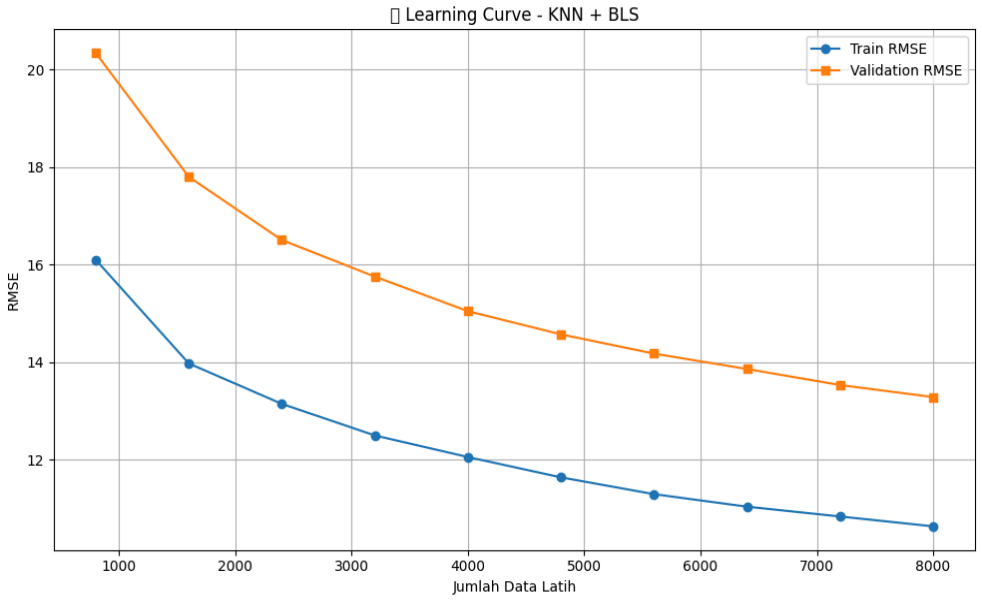
   Pada model ini, kami tidak menggunakan feature importance, tapi menggunakan parameter tuning karena KNN bersifat non-parametrik dan *instance-based*, sehingga tidak memiliki parameter bobot internal seperti *tree-based model* dan prediksi didasarkan pada kedekatan data bukan hasil pembelajaran fitur tertentu. Meskipun sebenarnya bisa menggunakan *permutation importance*, tapi itu terlalu mahal secara komputasi. Sehingga fokus pada parameter tuning digunakan untuk mengoptimalkan model.

   Hyperparameter tuning dilakukan menggunakan *grid search* karena sederhana dan akurat. Selain itu, *grid search* cocok untuk model dengan parameter terbatas seperti KNN.

V. Analisis Hasil

1. Evaluasi Model

Baseline model kami punya RMSE (*Root Mean Squared Error*) 13.1322. Model kami juga menunjukkan performa yang cukup baik dalam melakukan generalisasi pada data baru dan tidak terlalu *overfit*. Pada Gambar 5, dapat dilihat bahwa semakin banyak data yang digunakan untuk melatih model, RMSE nya semakin menurun tidak hanya pada *data train*, tapi juga untuk *data validation.*



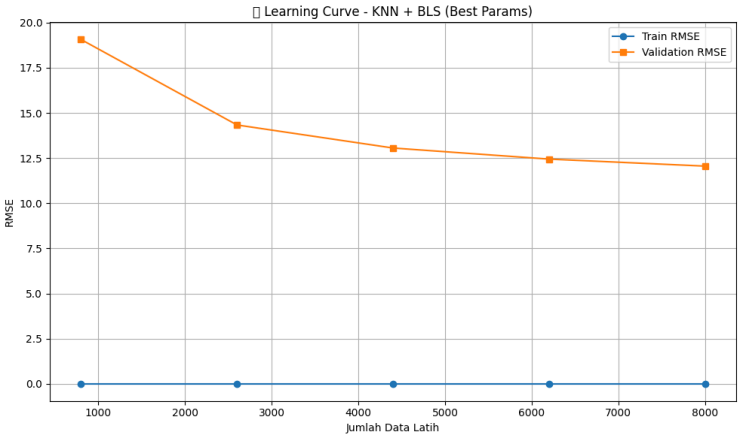
Gambar 5: Learning Curve KNN + BLS

2. Hasil Tuning dan Best Params

Gambar 6: Parameter Tuning

Pemilihan nilai *n_neighbors* dalam *grid search* dilakukan untuk uji generalisasi model KNN terhadap variasi lokal data dan untuk menghindari *overfitting* maupun *underfitting*. Nilai-nilai yang digunakan mencakup rentang kecil hingga besar untuk mengevaluasi performa model dalam berbagai tingkat kompleksitas.

Proses ini menghasilkan parameter terbaik yaitu: *Best Params: {'metric': 'manhattan', 'n_neighbors': 7, 'weights': 'distance'}* dengan RMSE 12.06. Namun, model malah *overfit* setelah di-*tuning*. Dari gambar 7, dapat dilihat bahwa model terlalu *fit* untuk *data train* tapi gagal generalisasi di data baru.



Gambar 7: Learning Curve KNN Best Parameter

VI. Kesimpulan dan Rekomendasi

1. Kesimpulan

Dalam kompetisi Code Challenge, kami membuat sebuah model AI untuk memprediksi jumlah perjalanan harian (*trips_thousands*) di berbagai zona kota fiktif dengan memanfaatkan data multivariabel dari berbagai domain, yaitu demografi, cuaca, aktivitas urban, dan infrastruktur transportasi.

Metodologi yang digunakan menggabungkan pendekatan *feature engineering Broad Learning System (BLS)* dengan model prediktif *K-Nearest Neighbors (KNN)*. Pendekatan ini dipilih karena mampu menangkap hubungan non-linear antar fitur dengan efisiensi komputasi yang lebih baik dibandingkan *deep learning*.

Model baseline (KNN dengan *n_neighbors=5, weights='uniform'*, dan *metric='minkowski'*) memberikan hasil yang cukup baik dengan RMSE 13.1322 pada data validasi. Meskipun dilakukan *hyperparameter tuning* menggunakan *Grid Search*, model terbaik yang dihasilkan justru *overfitting*. Oleh karena itu, kami memilih untuk mempertahankan *model baseline* yang lebih stabil dan memiliki kemampuan generalisasi yang lebih baik terhadap data baru.

2. Rekomendasi

Kami menggunakan *baseline model* untuk submission karena model hasil tuning gagal generalisasi pada data baru dan baseline model lebih stabil serta performa yang cukup baik. Pengembangan selanjutnya dapat dilakukan *ensemble learning* atau dilakukan *dimensionality reduction* untuk menekan *noise* dari hasil *feature expansion.* Selain itu, jika waktu dan sumber daya mencukupi, penggunaan model deep learning, seperti MLP, bisa dieksplorasi untuk menangkap pola non-linear lebih dalam dibandingkan KNN. Dengan pendekatan yang efisien, sistematis, dan masuk akal secara komputasi, model ini telah menunjukkan performa yang cukup sebagai fondasi awal sistem prediksi mobilitas.