LAPORAN EVALUASI TENGAH SEMESTER (ETS) BIG DATA ANALYSIS

House Price Prediction (Regression)

1. Identitas Mahasiswa

- Nama**:** ILHAM NURHAKIM
- NIM**:** 20123001
- CASE**:** C
- LINK GITHUB**:** https://github.com/ilhamst

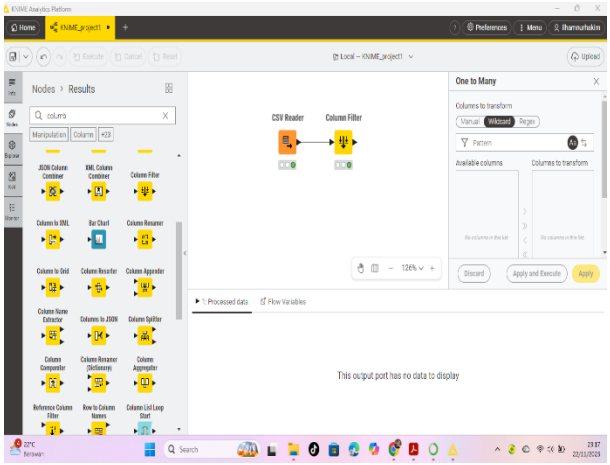
2. Informasi Dataset & Deskripsi Fitur

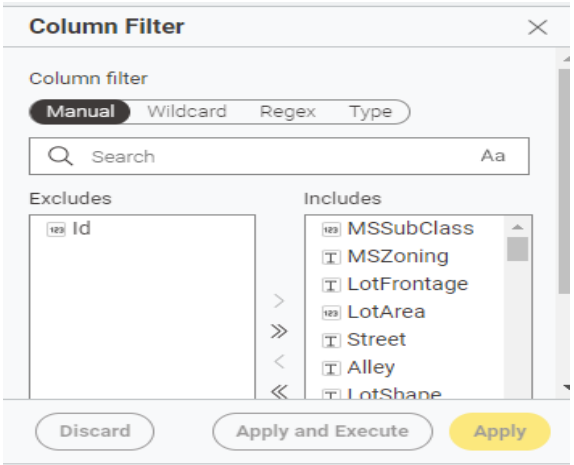
Tujuan : untuk membandingkan kinerja dua model Regresi (Linear Regression dan Decision Tree Regressor) dalam *House Price Prediction* (SalePrice) menggunakan *dataset* Ames, Iowa.

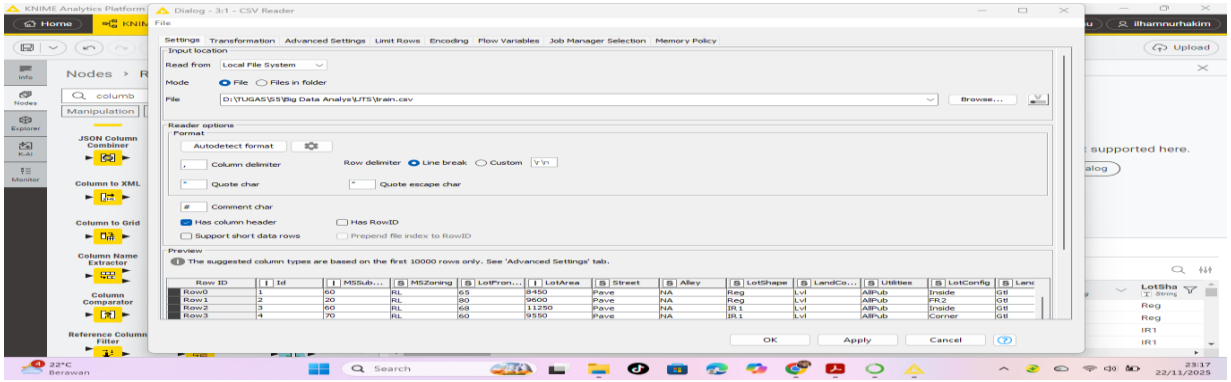
Target Variabel: SalePrice (Harga Jual Rumah), yang merupakan variabel Numerik/Float.

Langkah *Preprocessing* Awal (Umum):

1. Reader & Filter**:** Data dibaca menggunakan *CSV Reader* dan kolom Id yang tidak relevan dihapus menggunakan *Column Filter*.

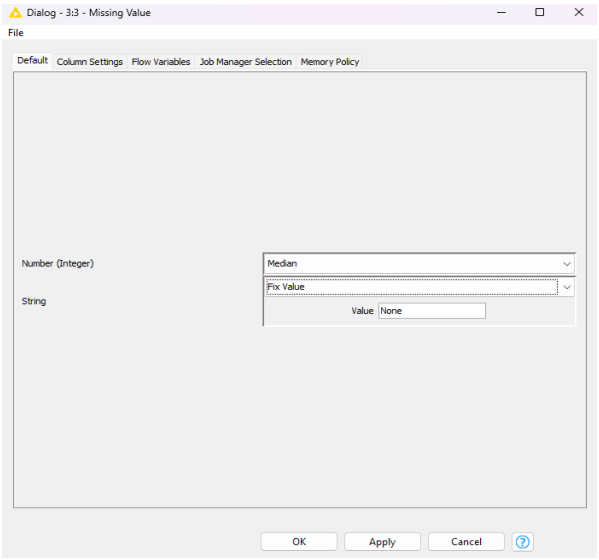




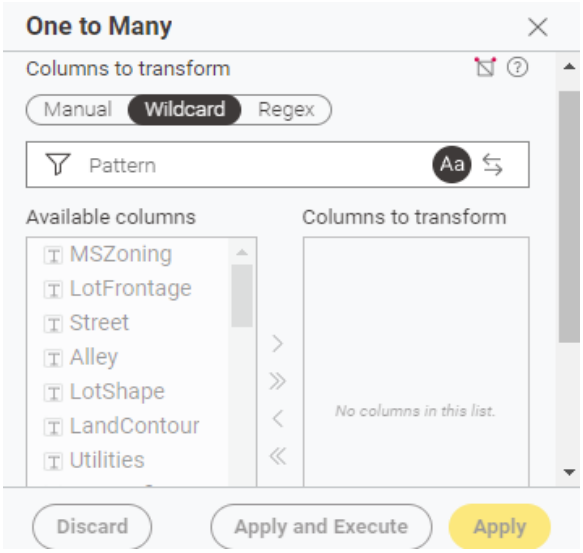


2. Imputasi (*Missing Value*): Nilai hilang diatasi menggunakan Node *Missing Value*.

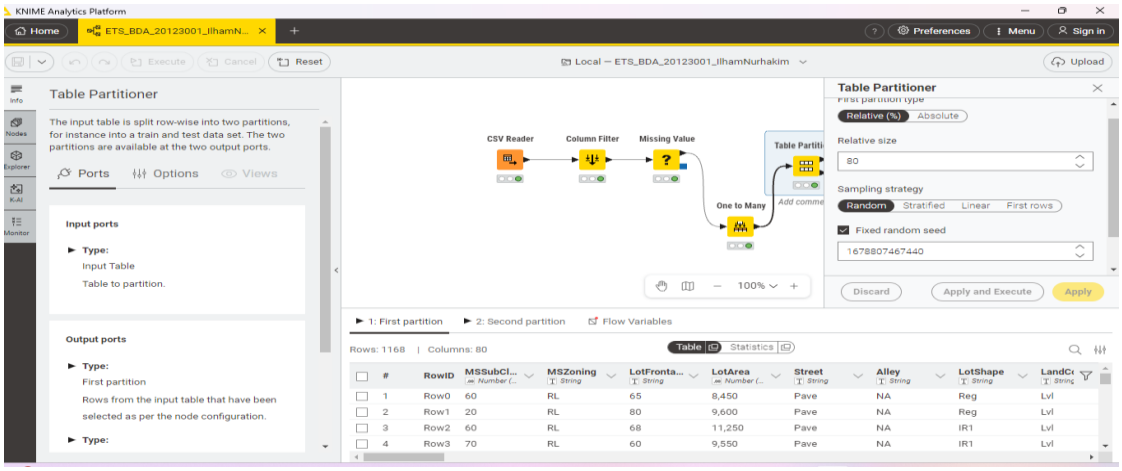
- Kolom Numerik (Integer) diisi dengan nilai *Median*.
- Kolom String diisi dengan *Fix Value: None*.



3. Encoding: Kolom kategorikal (bertipe String) diubah menjadi fitur biner (One-Hot Encoded) menggunakan Node One to Many.



4. Split Data: Data dibagi menjadi 80% Training Set dan 20% Test Set menggunakan Table Partitioner. Parameter Fixed random seed digunakan untuk memastikan hasil *split* yang sama setiap kali dieksekusi.



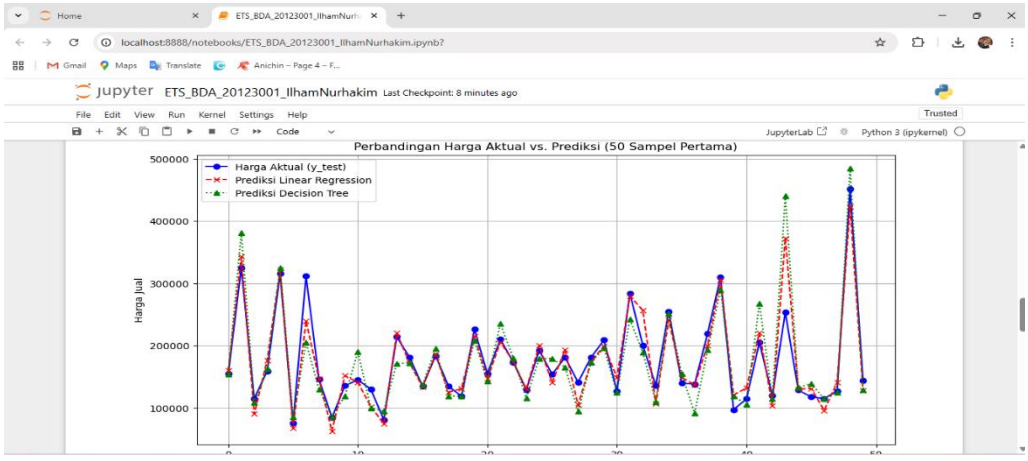
3. Python Modeling (Jupyter Notebook)

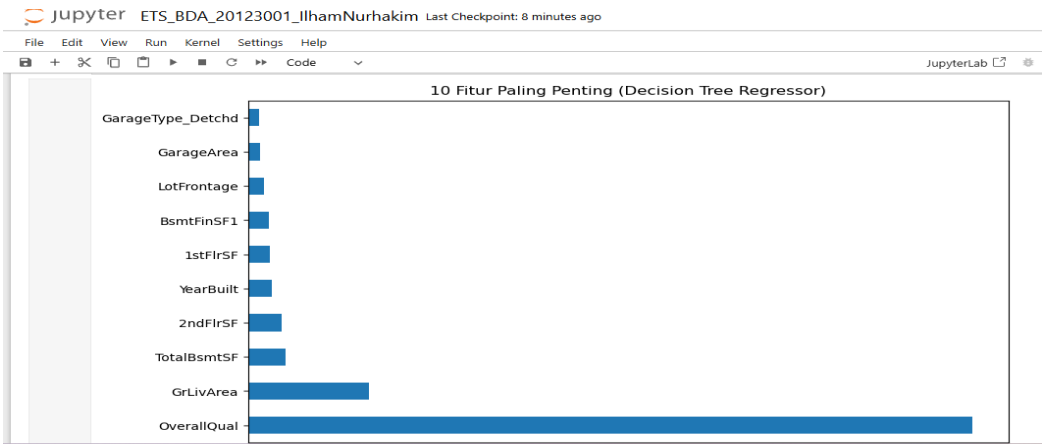
Bagian ini menyajikan hasil pemodelan Linear Regression dan Decision Tree yang diimplementasikan menggunakan Python (Jupyter Notebook).

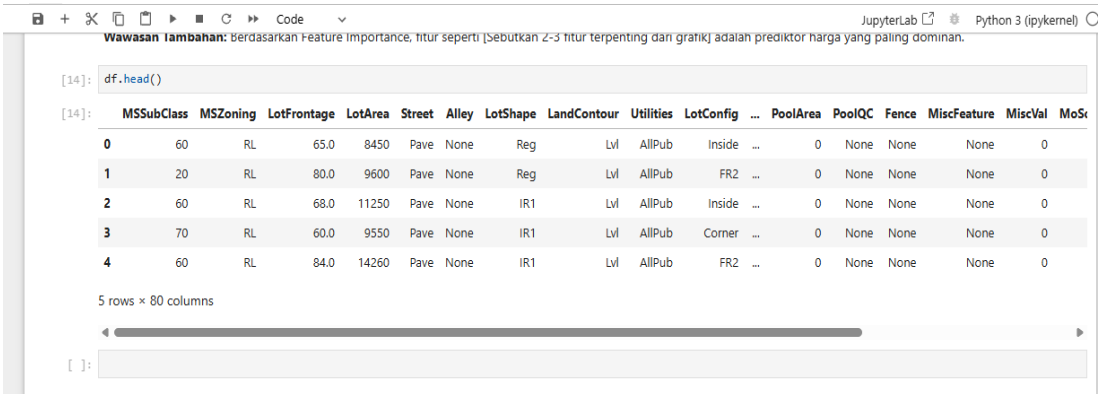
```
# C.4 & C.5. Model Pembanding: Decision Tree Regressor
dt_reg_model = DecisionTreeRegressor(random_state=42)
dt_reg_model.fit(X_train, y_train)
y_pred_dt = dt_reg_model.predict(X_test)

# C.6. Evaluasi Pembanding (RMSE)
rmse_dt = mean_squared_error(y_test, y_pred_dt, squared=False)
print(f"2. RMSE Decision Tree Regressor (Pembanding): ${rmse_dt:,.2f}") # C.7 Tampilkan Hasil
```

```
Data Split Selesai. Siap untuk Modeling.
1. RMSE Linear Regression (Baseline): $83,125.52
2. RMSE Decision Tree Regressor (Pembanding): $45,162.76
```

HASIL RUNNING :







Model	root mean ε (RMSE) (Python)	Keterangan
Linear Regression (Model Baseline)	$83,126.52$	Menjadi model dasar (baseline) untuk perbandingan.
Decision Tree Regressor (Model Pembanding)	$45,162.76$	Memberikan *error* lebih rendah daripada Linear Regression di Python.

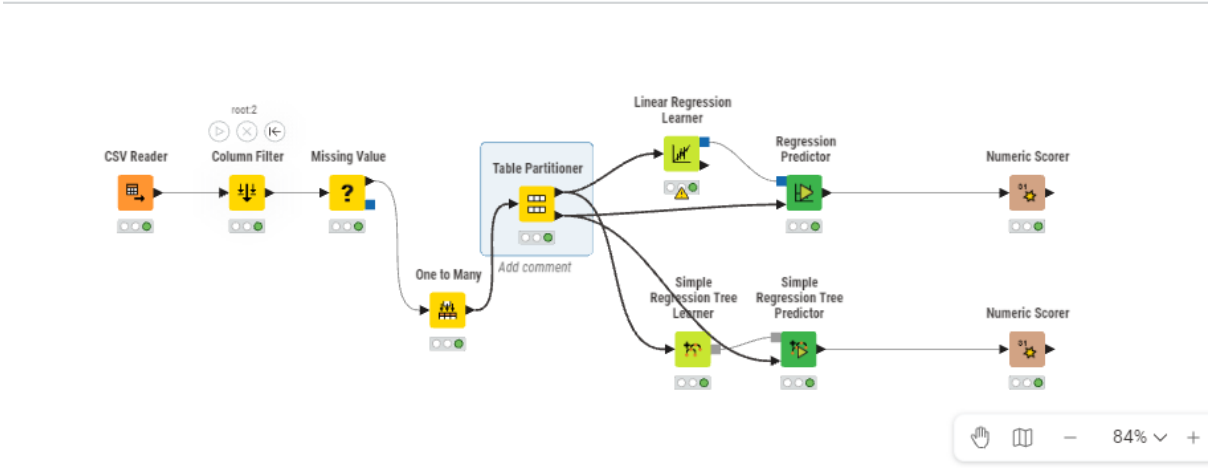
Dalam implementasi Python ini, Decision Tree Regressor menghasilkan RMSE yang lebih rendah, menjadikannya model terbaik untuk lingkungan Python.

4. KNIME Workflow

Workflow KNIME dibangun berdasarkan urutan Node Reader - Preprocessing - Model Learner - Predictor - Evaluator. Dua jalur pemodelan paralel (Linear Regression dan Decision Tree) dibuat dari node Table Partitioner.

- Skema Workflow KNIME

Workflow utama yang mencakup semua langkah hingga evaluasi akhir:

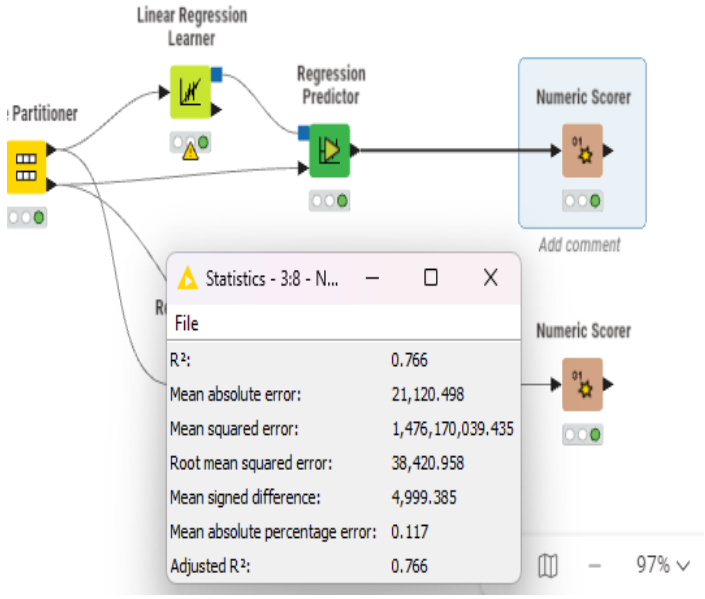


Terdapat aliran cabang pada table partitioner sehingga dapat menghasilkan 2 nilai akhir dari linear dan decision tree.

- Evaluasi Linear Regression (LR)

1. Learner & Predictor: Node *Linear Regression Learner* dilatih pada Training Set dan hasilnya diumpankan ke *Regression Predictor.*
2. *Evaluator: Numeric Scorer* menghitung metrik performa.

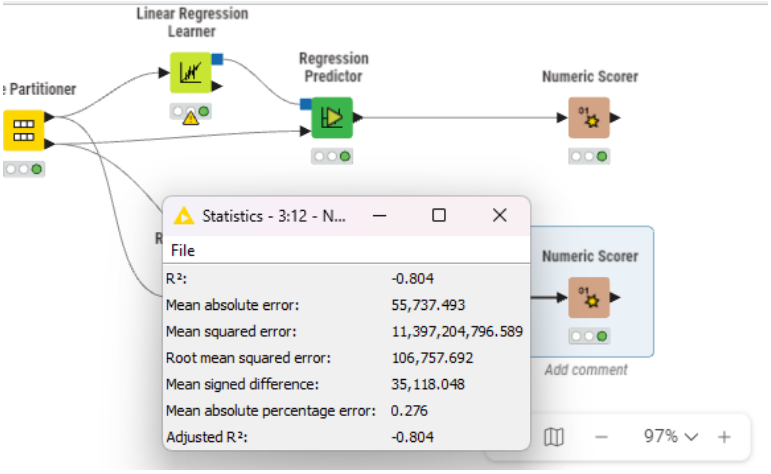
Hasil Linear Regression (KNIME):



Evaluasi Decision Tree (DT)

1. Learner & Predictor: Node *Simple Regression Tree Learner* (bukan *Decision Tree Learner* biasa) dilatih dan diprediksi menggunakan *Simple Regression Tree Predictor.*
2. Evaluator: Numeric Scorer menghitung metrik performa.

Hasil Decision Tree (KNIME):



Perbandingan Model

Model	RMSE (KNIME)	RMSE (Python)
Linear Regression	38,420.958	83,126.52
Decision Tree Regressor	106,757.692	45,162.76

5. Kesimpulan

1. Model Terbaik Berdasarkan KNIME: Model Linear Regression adalah model terbaik, dengan *error* rata-rata prediksi (RMSE) sebesar 38.421. Hal ini menunjukkan bahwa hubungan antara fitur-fitur rumah dan harga jual memiliki karakter yang lebih *linear* di data yang diproses oleh KNIME.
2. Alasan: Untuk memvalidasi model yang optimal, diperlukan sinkronisasi penuh pada semua langkah *preprocessing*, terutama pada nilai random state saat membagi data, untuk memastikan kedua *platform* bekerja pada Test Set yang identik.