Kuis Statistika dan Probabilitas

Uji Regresi Linear Data Sekunder Dari Kaggle.Com

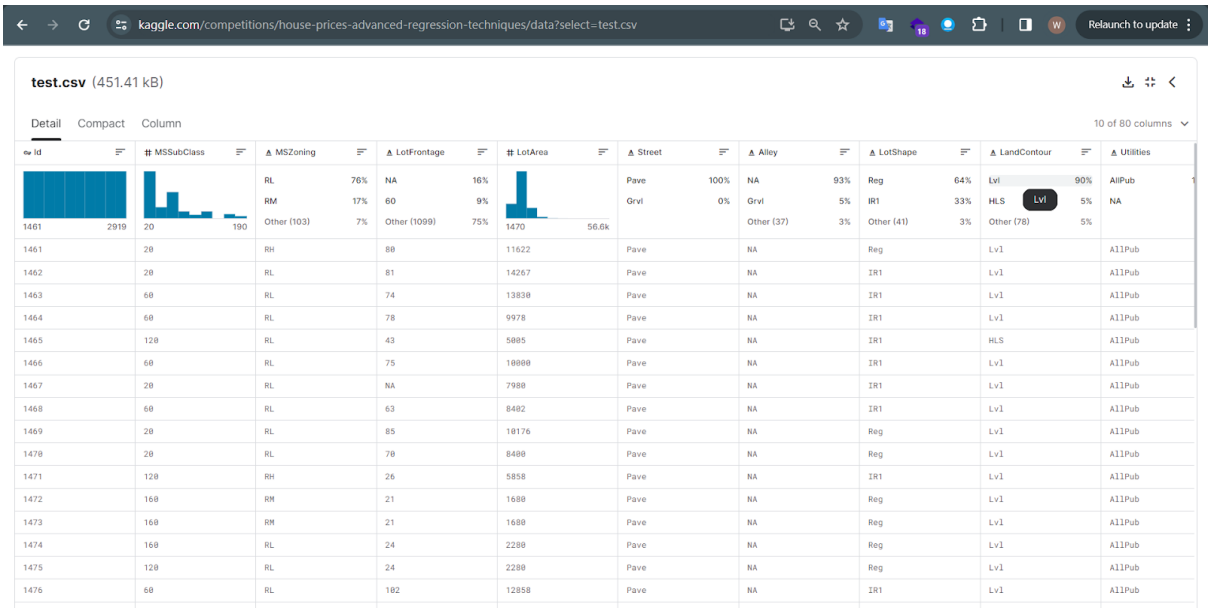
Nama : Wempy Aditya Wiryawan
Nim : 202210370311058
Kelas : 3A Statistika dan Probabilitas

INSTRUKSI KUIS

1. Mencari data kaggle minimal 1000 data dan minimal variabel NUMERIK 10. Data setiap individu WAJIB BERBEDA
2. Mengerjakan basic analysis regresi menggunakan python dan excel
    a. Uji Asumsi klasik regresi: normalitas, heteroskedastisitas, autokorelasi (semua pakai python dan masing2 diinterpretasi signifikan atau tidaknya)
    b. Pemodelan Regresi
        i. MANUAL: menggunakan excel dengan cara mengambil 100 data dari 1000 data yang dipunya + 5 variabel NUMERIK. Dianalisis menggunakan excel pakai rumus (X'X)inverse. (X'Y).
        ii. PYTHON
            1. Menggunakan excel dengan cara mengambil 100 data dari 1000 data yang dipunya + 5 variabel NUMERIK. Dianalisis menggunakan python. Koefisien regresi dibuktikan sama persis dengan manual di excel
            2. Menggunakan keseluruhan data 1000 data + 10 variabel NUMERIK. Dianalisis menggunakan python
    c. Uji Parsial Regresi: menguji signifikansi masing-masing koefisien regresi dan diinterpretasi. (Menggunakan python)
    d. Uji Simultan Regresi: menguji signifikansi keseluruhan model regresi dan diinterpretasi (menggunakan python)
    e. Uji Kebaikan Model menggunakan R2 (r square) (menggunakan python)
3. Dikumpulkan maksimal Hari Kamis tanggal 11 Januari 2024 jam 23.59 di Google Classroom
4. Dikumpulkan dalam bentuk
    a. Laporan dalam bentuk PDF (interpretasi keseluruhan makalah dan termasuk ss excel)
    b. PPT
    c. Link YouTube untuk pengumpulan video yang sudah dipresentasikan (wajib terlihat wajah)
    d. Source code
    e. Dikumpulkan INDIVIDU (Nama dopant - NIM Belakang)

PENYELESAIAN

1. **Dataset dari kaggle,com**
   source url :
   https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/data



Dataset ini menyediakan informasi kaya tentang berbagai aspek properti, memungkinkan analisis yang mendalam untuk memprediksi harga jual berdasarkan fitur-fitur tersebut.

- SalePrice (Harga Jual):
  Variabel target yang menyimpan harga jual properti dalam dolar.

Informasi Umum tentang Properti:

- MSSubClass: Kelas bangunan.
- MSZoning: Klasifikasi zona umum properti.
- LotFrontage: Panjang linear jalan yang terhubung dengan properti.
- LotArea: Luas tanah dalam kaki persegi.
- Street: Jenis akses jalan.
- Alley: Jenis akses gang.
- LotShape: Bentuk umum properti.
- LandContour: Keturunan tanah.
- Utilities: Jenis utilitas yang tersedia.
- LotConfig: Konfigurasi lot.

Informasi Bangunan dan Kondisi Umum:

- OverallQual: Kualitas dan penyelesaian umum.
- OverallCond: Peringkat kondisi umum.
- YearBuilt: Tahun konstruksi asli.
- YearRemodAdd: Tahun renovasi.
- RoofStyle: Jenis atap.
- RoofMatl: Material atap.
- Exterior1st: Penutup eksterior pertama.
- Exterior2nd: Penutup eksterior kedua.

Informasi tentang Material dan Fitur Eksterior:

- MasVnrType: Tipe veneer masonry.
- MasVnrArea: Luas veneer masonry dalam kaki persegi.
- ExterQual dan ExterCond: Kualitas dan kondisi material eksterior.
- Foundation: Jenis dasar.

Informasi Basement:

- BsmtQual, BsmtCond, BsmtExposure: Kualitas, kondisi, dan paparan basement.
- BsmtFinType1, BsmtFinSF1, BsmtFinType2, BsmtFinSF2, BsmtUnfSF: Jenis dan luas area basement.

Informasi Tentang Sistem Pemanas dan Listrik:

- Heating, HeatingQC, CentralAir, Electrical: Jenis pemanas, kualitas pemanas, AC sentral, sistem listrik.

Informasi Lantai:

- 1stFlrSF, 2ndFlrSF, LowQualFinSF, GrLivArea: Luas lantai tingkat pertama, kedua, tingkat rendah, dan total luas area lantai tinggal di atas tanah.

Informasi Kamar Mandi dan Kamar Tidur:

- BsmtFullBath, BsmtHalfBath, FullBath, HalfBath, Bedroom, Kitchen, TotRmsAbvGrd: Jumlah kamar mandi dan tidur.

Informasi Tambahan tentang Properti:

- Functional, Fireplaces, GarageType, GarageYrBlt, GarageFinish, GarageCars, GarageArea: Informasi tentang fungsionalitas, jumlah perapian, dan garasi.

Informasi Tentang Eksterior dan Fasilitas Tambahan:

- WoodDeckSF, OpenPorchSF, EnclosedPorch, 3SsnPorch, ScreenPorch, PoolArea: Luas area dek kayu, teras terbuka, teras tertutup, dan area kolam renang.

Kualitas dan Kondisi Tambahan:

- PoolQC, Fence, MiscFeature, MiscVal: Kualitas kolam renang, pagar, fitur tambahan lainnya, dan nilai fitur tambahan.

Informasi Penjualan:

- MoSold, YrSold, SaleType, SaleCondition: Bulan dan tahun penjualan, jenis penjualan, dan kondisi penjualan.

Sebelum dilakukan analisis data maka sebelumnya akan dilakukan data cleaning dengan tujuan untuk menghilangkan missing value pada data, dan mengambil data di bagian variabel numeriknya saja, berikut adalah code python untuk melakukan cleaning data :

```
import pandas as pd

# Membaca data dari CSV
data = pd.read_csv('train.csv')  # Gantilah 'nama_file.csv' dengan
nama file CSV yang sesuai

# Membersihkan data dari nilai yang hilang
data_cleaned = data.dropna()

# Menyaring hanya variabel numerik
```

```
data_numerical = data.select_dtypes(include=['number'])

# Menyimpan data yang telah dibersihkan ke dalam file CSV
data_numerical.to_csv('data_cleaned.csv', index=False)

# Menampilkan informasi sebelum dan setelah pembersihan
print(f"Jumlah baris sebelum pembersihan: {len(data)}")
print(f"Jumlah baris setelah pembersihan: {len(data_numerical)}")

# Menampilkan beberapa baris pertama dari data yang telah
dibersihkan
print("Beberapa baris pertama dari data yang telah dibersihkan:")
print(data_numerical.head())
```

2. **Mengerjakan basic analysis regresi**
   a. **Uji Asumsi klasik regresi**
      i. **Uji Normalisasi**

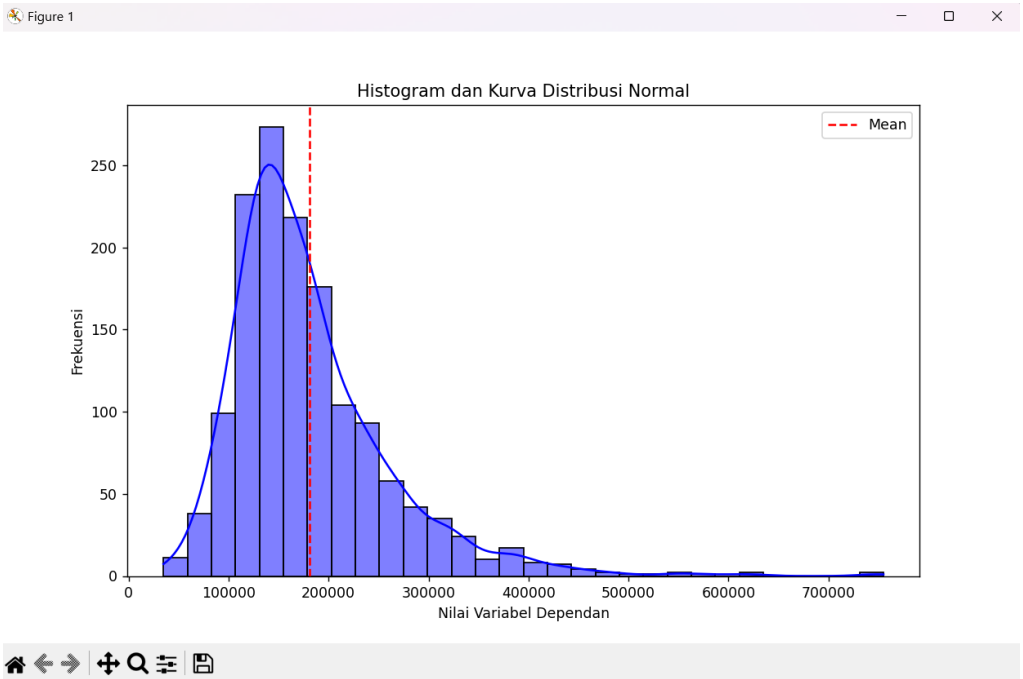
      Uji normalitas adalah suatu prosedur statistik yang digunakan untuk menguji apakah suatu sampel data atau residu dari model regresi memiliki distribusi normal. Distribusi normal (atau sering disebut distribusi Gaussian) adalah distribusi simetris yang membentuk lonceng.

      Source code python :

```
import pandas as pd
from scipy.stats import shapiro
import matplotlib.pyplot as plt
import seaborn as sns

# Membaca data dari CSV
data = pd.read_csv('./train.csv')

# Mengekstrak variabel dependen
y_variable = data['SalePrice']

# Uji Normalitas dengan Shapiro-Wilk
stat, p_value = shapiro(y_variable)

# Menampilkan hasil uji normalitas
print(f'Statistik Uji: {stat}')
print(f'P-value: {p_value}')

# Membuat histogram dan kurva distribusi normal
plt.figure(figsize=(10, 6))

# Histogram
```

```
sns.histplot(y_variable, kde=True, color='blue', bins=30)
plt.title('Histogram dan Kurva Distribusi Normal')
plt.xlabel('Nilai Variabel Dependan')
plt.ylabel('Frekuensi')

# Menambahkan kurva distribusi normal
plt.axvline(x=y_variable.mean(), color='red', linestyle='--',
label='Mean')
plt.legend()

# Menampilkan plot
plt.show()

# Interpretasi hasil uji normalitas
alpha = 0.05
if p_value > alpha:
    print('Tidak ada bukti yang cukup untuk menolak H0,
sehingga data terdistribusi normal.')
else:
    print('Terdapat bukti yang cukup untuk menolak H0, sehingga
data tidak terdistribusi normal.')
```

Dan berikut adalah hasil nya




```
Statistik Uji: 0.869672954082489
P-value: 3.2072044604461286e-33
Terdapat bukti yang cukup untuk menolak H0, sehingga data
tidak terdistribusi normal.
```

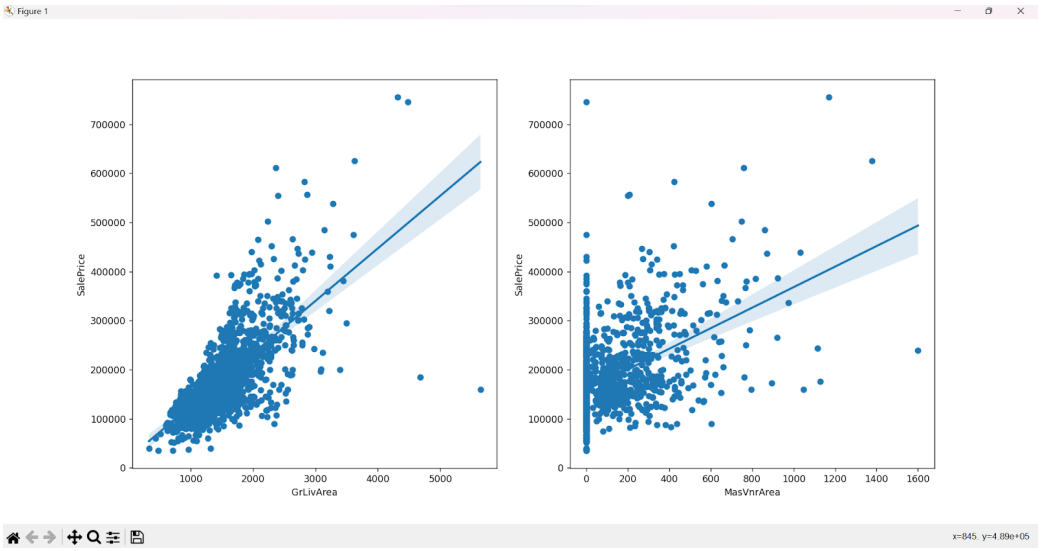
kurva distribusi memiliki skewness positif, artinya ekor distribusi lebih panjang di sisi kanan dan ekor pendek di sisi kiri. Puncak distribusi dan nilai mean akan cenderung ke arah kiri, sementara sebagian besar data berada di sebelah kanan.

ii. Uji Heteroskedastisitas

Uji heteroskedastisitas bertujuan untuk mengidentifikasi apakah variabilitas (heteroskedastisitas) dari residu dalam model regresi tidak konstan. Dengan kata lain, uji ini memeriksa apakah varians residual berubah sepanjang rentang nilai independen.

```
import pandas as pd
import statsmodels.api as sm
import seaborn as sns
import matplotlib.pyplot as plt


train = pd.read_csv('train.csv')

## Plot sizing.
fig, (ax1, ax2) = plt.subplots(figsize = (12,8),
ncols=2,sharey=False)
## Scatter plotting for SalePrice and GrLivArea.
sns.scatterplot( x = train.GrLivArea, y = train.SalePrice,
ax=ax1)
## Putting a regression line.
sns.regplot(x=train.GrLivArea, y=train.SalePrice, ax=ax1)

## Scatter plotting for SalePrice and MasVnrArea.
sns.scatterplot(x = train.MasVnrArea,y = train.SalePrice,
ax=ax2)
## regression line for MasVnrArea and SalePrice.
sns.regplot(x=train.MasVnrArea, y=train.SalePrice, ax=ax2)
plt.show()
```

Dan berikut adalah hasilnya



Grafik yang diberikan menunjukkan perhitungan wajar properti berdasarkan atribut properti. Grafik ini memperlihatkan persebaran atribut Sale Price dan GrLivArea secara horisontal dan masing-masing pada grafik vertikal masing-masing. Grafik ini juga menampilkan distribusi jumlah masing-masing unit dan tahapan properti dari unit sampel yang digunakan dalam perhitungan wajar properti.

Sale Price mencakup jangkauan harga rumah dari $700,000 hingga $100,000 dengan tahapan yang dihitung pada basis 100,000. Sale Price dan GrLivArea secara vertikal ditampilkan berdasarkan frekuensi tingkat, yang mengindikasikan jumlah unit yang memiliki ukuran yang sama atau harga yang sama.

Pada grafik lainnya, persebaran Sale Price terhadap GrLivArea ditampilkan secara horizontal. Sale Price adalah nilai tertinggi di kiri kanan grafik, yang mengindikasikan harga tertinggi untuk ukuran tertinggi, sementara harga terendah ditampilkan di kanan kiri grafik, yang mengindikasikan harga terendah untuk ukuran terkecil.

Distribusi jumlah unit berdasarkan Sale Price ditampilkan dalam grafik horizontal. Pada grafik ini, unit dengan harga yang sama diagregasikan bersama. Contohnya, harga antara $100,000 hingga $200,000 diasumsikan untuk semua unit yang termasuk dalam rentang ini.

```
import pandas as pd
import statsmodels.api as sm
from statsmodels.stats.diagnostic import het_white
import seaborn as sns
import matplotlib.pyplot as plt

data = pd.read_csv('data_cleaned1.csv')
y_variable = data['SalePrice']
X_variables = data[['GrLivArea', 'GarageArea', 'LotArea',
'MasVnrArea', 'TotalBsmtSF']]
```

```
X_variables = sm.add_constant(X_variables)

model = sm.OLS(y_variable, X_variables).fit()

# Melakukan uji White (Breusch-Pagan) untuk heteroskedastisitas
het_white_stat, het_white_p_value, het_white_fstat,
het_white_p_value_fstat = het_white(model.resid, X_variables)
print(f'White Test Statistic: {het_white_stat}')
print(f'White Test P-value: {het_white_p_value}')

# Visualisasi
plt.figure(figsize=(10, 6))
sns.scatterplot(x=model.fittedvalues, y=model.resid)
plt.title('Residual Plot untuk Deteksi Heteroskedastisitas')
plt.xlabel('Nilai Prediksi')
plt.ylabel('Residuals')
plt.show()

# Interpretasi hasil uji heteroskedastisitas
alpha = 0.05
if het_white_p_value < alpha:
    print('Terdapat bukti yang cukup untuk menolak H0, sehingga
data menunjukkan tanda-tanda heteroskedastisitas.')
else:
    print('Tidak ada bukti yang cukup untuk menolak H0,
sehingga tidak terdapat tanda-tanda heteroskedastisitas pada
data.')
```

Output

```
White Test Statistic: 1031.7205551362495
White Test P-value: 6.689327373714455e-206
Terdapat bukti yang cukup untuk menolak H0, sehingga data
menunjukkan tanda-tanda heteroskedastisitas.
```

White Test Statistic: 1031.72
Nilai uji statistik White yang tinggi menunjukkan bahwa residual pada model regresi memiliki variasi yang tidak konstan (heteroskedastisitas).
White Test P-value: ~0.00 (e-206)
P-value yang sangat rendah menunjukkan bahwa terdapat cukup bukti statistik untuk menolak hipotesis nol (H0). Oleh karena itu, kita dapat menyimpulkan bahwa data menunjukkan tanda-tanda heteroskedastisitas.
Residual Plot:
Melalui visualisasi residual plot, kita dapat melihat bahwa ada pola dalam sebaran residu, menunjukkan variasi yang tidak konstan di seluruh rentang nilai prediksi.

iii. Uji Autokorelasi

Uji autokorelasi adalah suatu metode statistik yang digunakan untuk mengevaluasi apakah ada ketergantungan antara nilai-nilai suatu variabel dengan nilai-nilai yang sama variabel pada periode waktu sebelumnya (autokorelasi). Uji ini umumnya digunakan dalam konteks analisis deret waktu atau analisis regresi untuk memeriksa apakah terdapat pola tertentu dalam residu model regresi.

Salah satu uji autokorelasi yang umum digunakan adalah Uji Durbin-Watson. Uji ini menguji apakah terdapat autokorelasi positif atau negatif pada residu model regresi. Statistik Durbin-Watson memiliki nilai antara 0 dan 4, dan interpretasinya sebagai berikut:

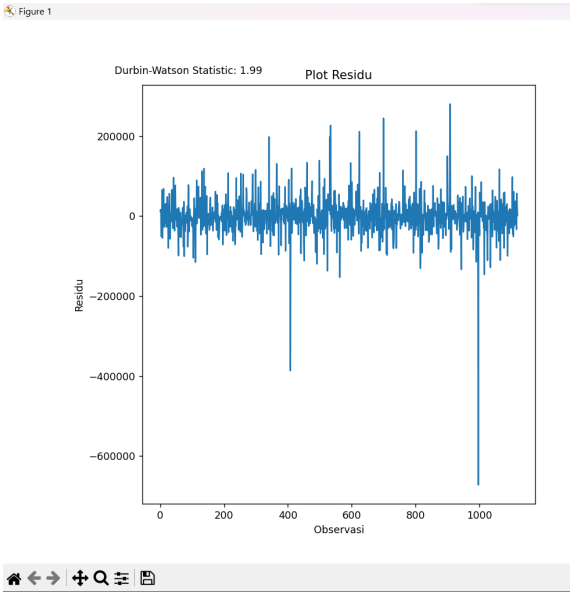
- Nilai mendekati 0: Indikasi autokorelasi positif yang kuat.
- Nilai mendekati 4: Indikasi autokorelasi negatif yang kuat.
- Nilai mendekati 2: Tidak ada indikasi autokorelasi yang signifikan.

source code python uji Durbin-Watson

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
import matplotlib.pyplot as plt

# Membaca data dari file CSV
data = pd.read_csv('data_cleaned1.csv')  # Gantilah
'nama_file.csv' dengan nama file CSV Anda

# Menentukan variabel independen dan dependen
# Gantilah 'variabel_dependen' dan 'variabel_independen'
sesuai dengan nama variabel dalam dataset Anda
variabel_dependen = data['SalePrice']
variabel_independen = data[['GrLivArea', 'GarageArea',
'LotArea', 'MasVnrArea', 'TotalBsmtSF']]

# Menambahkan konstanta ke variabel independen
X = sm.add_constant(variabel_independen)

# Membuat model regresi
model = sm.OLS(variabel_dependen, X).fit()

# Menghitung residu (kesalahan)
residuals = model.resid

# Membuat plot residu
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.plot(residuals)
plt.title('Plot Residu')
```

```
plt.xlabel('Observasi')
plt.ylabel('Residu')

# Melakukan uji Durbin-Watson
durbin_watson_statistic =
sm.stats.stattools.durbin_watson(residuals)

# Menampilkan hasil uji
plt.subplot(1, 2, 2)
plt.text(0.1, 0.9, f'Durbin-Watson Statistic:
{durbin_watson_statistic:.2f}',
transform=plt.gcf().transFigure)
plt.axis('off')

# Menampilkan plot dan hasil uji
plt.show()

# Interpretasi hasil uji
if durbin_watson_statistic < 1.5:
    print('Indikasi positif autokorelasi (residuals cenderung
positif terkorelasi).')
elif durbin_watson_statistic > 2.5:
    print('Indikasi negatif autokorelasi (residuals cenderung
negatif terkorelasi).')
else:
    print('Tidak ada indikasi autokorelasi yang signifikan.')
```

Dan berikut adalah hasilnya



Dalam kasus ini, nilai 1.99 mendekati 2, yang mengindikasikan bahwa tidak ada bukti yang cukup untuk menyimpulkan adanya autokorelasi yang signifikan pada residu model regresi. Oleh karena itu, dari nilai statistik Durbin-Watson ini, kita dapat menganggap bahwa tidak ada autokorelasi yang signifikan dalam residu model regresi

b. Pemodelan Regresi

i. Manual

menggunakan excel dengan cara mengambil 100 data dari 1000 data yang dipunya + 5 variabel NUMERIK

GrLivArea	GarageArea	LotArea	MasVnrArea	TotalBsmtSF	MSSubClass	WoodDeckSF	OpenPorchSF	1stFlrSF	2ndFlrSF
1710	548	8450	196	856	60	0	61	856	854
1262	460	9600	0	1262	20	298	0	1262	0
1786	608	11250	162	920	60	0	42	920	866
1717	642	9550	0	756	70	0	35	961	756
2198	836	14260	350	1145	60	192	84	1145	1053
1362	480	14115	0	796	50	40	30	796	566
1694	636	10084	186	1686	20	255	57	1694	0
1774	468	6120	0	952	50	90	0	1022	752
1077	205	7420	0	991	190	0	4	1077	0
1040	384	11200	0	1040	20	0	0	1040	0
2324	736	11924	286	1175	60	147	21	1182	1142
1494	840	10652	306	1494	20	160	33	1494	0
854	576	6120	0	832	45	48	112	854	0
1296	516	10791	0	0	90	0	0	1296	0
1114	576	13695	0	1114	20	0	102	1114	0
1339	294	7560	0	1029	20	0	0	1339	0
2376	853	14215	380	1158	60	240	154	1158	1218
1108	280	7449	0	637	45	0	0	1108	0
1795	534	9742	281	1777	20	171	159	1795	0
1060	572	4224	0	1040	120	100	110	1060	0
1600	890	14230	640	1566	20	0	56	1600	0
900	576	7200	0	900	20	222	32	900	0
1704	772	11478	200	1704	20	0	50	1704	0
1600	319	16321	0	1484	20	288	258	1600	0
520	240	6324	0	520	30	49	0	520	0
1317	250	8500	0	649	70	0	54	649	668
1234	484	11049	0	1234	20	0	30	1234	0
1700	447	10552	0	1398	20	0	38	1700	0

SUMMARY OUTPUT

Regression Statistics

Multiple R	0,896056791
R Square	0,802917772
Adjusted R Square	0,792434675
Standard Error	34807,79035
Observations	100

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	5	4,63985E+11	92797091790	76,59165552	1,28932E-31
Residual	94	1,13889E+11	1211582269		
Total	99	5,77874E+11			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95,0%*	*Upper 95,0%*
Intercept	-54362,01792	13578,09847	-4,003654711	0,000124521	-81321,65073	-27402,4	-81321,7	-27402,4
X Variable 1	69,52826652	8,544444767	8,137248051	1,6439E-12	52,56307114	86,49346	52,56307	86,49346
X Variable 2	123,7489945	25,52048539	4,849006303	4,8986E-06	73,07747254	174,4205	73,07747	174,4205
X Variable 3	1,827060791	0,803631573	2,273505489	0,025272059	0,231431465	3,42269	0,231431	3,42269
X Variable 4	30,11165568	16,97779126	1,773590876	0,079369362	-3,598146867	63,82146	-3,59815	63,82146
X Variable 5	44,18825269	10,71479825	4,124039638	8,03145E-05	22,9137696	65,46274	22,91377	65,46274

ii. Python

1. Koefisien Regresi dengan 100 data

berikut code jika diambil 100 data secara random + 5 variabel numerik

```
import pandas as pd
import numpy as np

# Ambil 100 data dan 5 variabel numerik dari DataFrame
data = pd.read_csv('train.csv')
selected_data = data.sample(n=100)
selected_variables = selected_data[['GrLivArea',
'GarageArea', 'LotArea', 'MasVnrArea', 'TotalBsmtSF']]

# Hitung matriks X
X = np.c_[np.ones(selected_data.shape[0]),
selected_variables]

# Hitung transpos dari matriks X
X_transpose = X.T

# Hitung matriks X'X dan X'Y
XtX = np.dot(X_transpose, X)
XtY = np.dot(X_transpose, selected_data['SalePrice'])

# Hitung invers dari matriks X'X
XtX_inv = np.linalg.inv(XtX)
```

```
# Hitung koefisien regresi
coefficients = np.dot(XtX_inv, XtY)

print("Koefisien Regresi:")
print(coefficients)
```

Dan berikut adalah hasil dari koefisien regresi

```
[-1.84218936e+04  5.04127718e+01  1.02220512e+02
1.57093918e+00 1.53586501e+02  4.08279950e+01]
```

Berikut jika diambil 100 data teratas saja

```
import pandas as pd
import numpy as np

# Ambil 100 data dan 5 variabel numerik dari DataFrame
data = pd.read_csv('data_cleaned1.csv')
selected_data = data.head(100)  # Gantilah 'your_data'
dengan nama DataFrame Anda
selected_variables = selected_data[['GrLivArea',
'GarageArea', 'LotArea', 'MasVnrArea', 'TotalBsmtSF']]

# Hitung matriks X
X = np.c_[np.ones(selected_data.shape[0]),
selected_variables]

# Hitung transpos dari matriks X
X_transpose = X.T

# Hitung matriks X'X dan X'Y
XtX = np.dot(X_transpose, X)
XtY = np.dot(X_transpose, selected_data['SalePrice'])

# Hitung invers dari matriks X'X
XtX_inv = np.linalg.inv(XtX)

# Hitung koefisien regresi
coefficients = np.dot(XtX_inv, XtY)

print("Koefisien Regresi:")
print(coefficients)
```

Hasilnya sebagai berikut

```
Koefisien Regresi:
[-5.43620179e+04  6.95282665e+01  1.23748995e+02
1.82706079e+00 3.01116557e+01  4.41882527e+01]
```

Intercept (Konstanta): -54,362
Nilai ini menunjukkan perkiraan harga jual ("SalePrice") ketika semua variabel independen diatur ke nilai 0. Dalam konteks ini, interpretasi konstan mungkin tidak memiliki makna praktis.
GrLivArea (Luas Area Hidup di Atas Tanah): 69.53
Setiap peningkatan satu unit dalam luas area hidup di atas tanah dikaitkan dengan peningkatan sekitar $69.53 dalam harga jual, jika variabel lain tetap.
GarageArea (Luas Area Garasi): 123.75
Setiap peningkatan satu unit dalam luas area garasi dikaitkan dengan peningkatan sekitar $123.75 dalam harga jual, jika variabel lain tetap.
LotArea (Luas Tanah): 1.83
Setiap peningkatan satu unit dalam luas tanah dikaitkan dengan peningkatan sekitar $1.83 dalam harga jual, jika variabel lain tetap.
MasVnrArea (Luas Area Material Veneer): 30.11
Setiap peningkatan satu unit dalam luas area material veneer dikaitkan dengan peningkatan sekitar $30.11 dalam harga jual, jika variabel lain tetap.
TotalBsmtSF (Total Luas Area Basement): 44.19
Setiap peningkatan satu unit dalam total luas area basement dikaitkan dengan peningkatan sekitar $44.19 dalam harga jual, jika variabel lain tetap.

2. Koefisien Regresi dengan 1000 data dan 10 variabel numerik

```
import pandas as pd
import numpy as np

# Ambil 100 data dan 5 variabel numerik dari DataFrame
data = pd.read_csv('data_cleaned1.csv')
selected_data = data.sample(n=1000)
selected_variables = selected_data[['GrLivArea',
'GarageArea', 'LotArea', 'MasVnrArea', 'EnclosedPorch',
'TotalBsmtSF', 'WoodDeckSF', 'OpenPorchSF', '1stFlrSF',
'2ndFlrSF']]

# Hitung matriks X
X = np.c_[np.ones(selected_data.shape[0]),
selected_variables]

# Hitung transpos dari matriks X
X_transpose = X.T

# Hitung matriks X'X dan X'Y
XtX = np.dot(X_transpose, X)
XtY = np.dot(X_transpose, selected_data['SalePrice'])
```

```
# Hitung invers dari matriks X'X
XtX_inv = np.linalg.inv(XtX)

# Hitung koefisien regresi
coefficients = np.dot(XtX_inv, XtY)

print("Koefisien Regresi:")
print(coefficients)
```

Dan berikut adalah hasilnya :

```
Koefisien Regresi:
[-1.13699349e+04 -5.93743032e+01  9.30842017e+01
3.70697808e-01
  5.23267618e+01 -7.93659189e+01  4.15743846e+01
7.62321433e+01
  3.30966066e+01  1.16177242e+02  1.21200263e+02]
```

- Intercept (Konstanta): Saat semua variabel independen di-set ke nilai 0, nilai SalePrice diharapkan berada pada -11,369.93.
- GrLivArea (Luas Area Hidup): Dengan setiap peningkatan satu unit dalam GrLivArea, SalePrice diharapkan turun sekitar 59.37 unit.
- GarageArea (Luas Area Garasi): Dengan setiap peningkatan satu unit dalam GarageArea, SalePrice diharapkan naik sekitar 93.08 unit.
- LotArea (Luas Area Lot): Dengan setiap peningkatan satu unit dalam LotArea, SalePrice diharapkan naik sekitar 0.37 unit.
- MasVnrArea (Luas Area Veneer Masonry): Dengan setiap peningkatan satu unit dalam MasVnrArea, SalePrice diharapkan naik sekitar 52.33 unit.
- EnclosedPorch (Luas Area Teras Tertutup): Dengan setiap peningkatan satu unit dalam EnclosedPorch, SalePrice diharapkan turun sekitar 79.37 unit.
- TotalBsmtSF (Luas Total Basement): Dengan setiap peningkatan satu unit dalam TotalBsmtSF, SalePrice diharapkan naik sekitar 41.57 unit.
- WoodDeckSF (Luas Area Wood Deck): Dengan setiap peningkatan satu unit dalam WoodDeckSF, SalePrice diharapkan naik sekitar 76.23 unit.
- OpenPorchSF (Luas Area Open Porch): Dengan setiap peningkatan satu unit dalam OpenPorchSF, SalePrice diharapkan naik sekitar 33.10 unit.
- 1stFlrSF (Luas Lantai 1): Dengan setiap peningkatan satu unit dalam 1stFlrSF, SalePrice diharapkan naik sekitar 116.18 unit.
- 2ndFlrSF (Luas Lantai 2): Dengan setiap peningkatan satu unit dalam 2ndFlrSF, SalePrice diharapkan naik sekitar 121.20 unit.

c. Uji Parsial Regresi

Uji Parsial Regresi dilakukan untuk menguji signifikansi masing-masing koefisien regresi. Dalam konteks regresi linear, uji ini dapat dilakukan dengan menggunakan uji t-statistik untuk masing-masing koefisien regresi. Berikut adalah contoh kode Python untuk melakukan Uji Parsial Regresi:

```
import pandas as pd
import statsmodels.api as sm

# Baca data dari CSV
data = pd.read_csv('data_cleaned1.csv')  # Gantilah
'nama_file.csv' dengan nama file CSV yang sesuai

# Pilih variabel independen (X) dan variabel dependen (Y)
X = data[['GrLivArea', 'GarageArea', 'LotArea', 'MasVnrArea',
'TotalBsmtSF']]  # Gantilah dengan variabel independen yang sesuai
Y = data['SalePrice']  # Gantilah dengan variabel dependen yang
sesuai

# Tambahkan kolom bias (intercept) ke matriks X
X = sm.add_constant(X)

# Buat model regresi
model = sm.OLS(Y, X).fit()

# Hitung t-statistik dan p-value untuk masing-masing koefisien
t_statistic = model.tvalues
p_values = model.pvalues

# Tampilkan hasil uji parsial regresi
results = pd.DataFrame({'Koefisien': model.params, 't-Statistic':
t_statistic, 'P-Value': p_values})
print(results)
```

Dan berikut adalah hasilnya

```
               Koefisien  t-Statistic       P-Value
const       -24154.369877    -4.556237  5.781502e-06
GrLivArea       65.571508    18.915838  1.987758e-69
GarageArea     101.824422    10.710023  1.518483e-25
LotArea          0.377556     1.972514  4.879770e-02
MasVnrArea      57.766919     6.579589  7.241091e-11
TotalBsmtSF     44.509115    10.843862  4.067206e-26
```

const (Intercept):
Interpretasi: Intercept atau konstanta model regresi. T-statistik yang signifikan menunjukkan bahwa intercept memiliki dampak yang signifikan pada variabel dependen.
GrLivArea:
Interpretasi: Setiap unit peningkatan dalam GrLivArea dihubungkan dengan peningkatan sebesar 65.57 unit dalam variabel dependen. T-statistik yang tinggi

dan p-value yang sangat rendah menunjukkan bahwa GrLivArea secara signifikan mempengaruhi variabel dependen.
GarageArea:
Interpretasi: Setiap unit peningkatan dalam GarageArea dihubungkan dengan peningkatan sebesar 101.82 unit dalam variabel dependen. T-statistik yang tinggi dan p-value yang sangat rendah menunjukkan bahwa GarageArea secara signifikan mempengaruhi variabel dependen.
LotArea:
Interpretasi: Setiap unit peningkatan dalam LotArea dihubungkan dengan peningkatan sebesar 0.38 unit dalam variabel dependen. Meskipun t-statistik cukup tinggi, nilai p-value yang cukup besar (0.05) menunjukkan bahwa ada sedikit ketidakpastian terkait signifikansi LotArea terhadap variabel dependen.
MasVnrArea:
Interpretasi: Setiap unit peningkatan dalam MasVnrArea dihubungkan dengan peningkatan sebesar 57.77 unit dalam variabel dependen. T-statistik yang tinggi dan p-value yang sangat rendah menunjukkan bahwa MasVnrArea secara signifikan mempengaruhi variabel dependen.
TotalBsmtSF:
Interpretasi: Setiap unit peningkatan dalam TotalBsmtSF dihubungkan dengan peningkatan sebesar 44.51 unit dalam variabel dependen. T-statistik yang tinggi dan p-value yang sangat rendah menunjukkan bahwa TotalBsmtSF secara signifikan mempengaruhi variabel dependen.
Kesimpulannya, variabel GrLivArea, GarageArea, MasVnrArea, dan TotalBsmtSF memiliki dampak signifikan terhadap variabel dependen, sementara LotArea memiliki dampak yang kurang signifikan. P-value yang rendah menunjukkan bahwa koefisien regresi untuk variabel-variabel tersebut tidak muncul secara kebetulan.

d. Uji Simultan Regresi

Uji Simultan Regresi, juga dikenal sebagai uji F-statistik, digunakan untuk menguji signifikansi keseluruhan dari model regresi. Pada uji ini, hipotesis nol (H0) adalah bahwa semua koefisien regresi (kecuali intercept) adalah nol, yang berarti tidak ada hubungan linear antara variabel independen dan variabel dependen secara keseluruhan. Hipotesis alternatif (H1) adalah setidaknya satu koefisien regresi tidak sama dengan nol, menunjukkan adanya hubungan linear antara setidaknya satu variabel independen dan variabel dependen.

Berikut adalah contoh kode Python untuk melakukan Uji Simultan Regresi menggunakan library statsmodels:

```
import pandas as pd
import statsmodels.api as sm

# Baca data dari CSV
data = pd.read_csv('data_cleaned1.csv')  # Gantilah
'nama_file.csv' dengan nama file CSV yang sesuai

# Pilih variabel independen (X) dan variabel dependen (Y)
```

```
X = data[['GrLivArea', 'GarageArea', 'LotArea', 'MasVnrArea',
'TotalBsmtSF']]  # Gantilah dengan variabel independen yang
sesuai
Y = data['SalePrice']  # Gantilah dengan variabel dependen yang
sesuai

# Tambahkan kolom bias (intercept) ke matriks X
X = sm.add_constant(X)

# Buat model regresi
model = sm.OLS(Y, X).fit()

# Hitung uji F-statistik dan p-value
f_statistic = model.fvalue
p_value = model.f_pvalue

# Tampilkan hasil uji simultan regresi
print(f"Uji Simultan Regresi (F-statistik): {f_statistic}")
print(f"P-Value: {p_value}")

# Interpretasi
if p_value < 0.05:
    print("Keseluruhan model regresi signifikan secara
statistik.")
else:
    print("Tidak cukup bukti untuk menolak hipotesis nol. Model
regresi mungkin tidak signifikan secara keseluruhan.")
```

Hasilnya ketika program dijalankan

```
Uji Simultan Regresi (F-statistik): 432.36968645433524
P-Value: 5.19596271515203e-258
Keseluruhan model regresi signifikan secara statistik.
```

Hasil uji simultan regresi menunjukkan bahwa nilai F-statistik yang tinggi dan nilai p-value yang sangat rendah (melebihi ambang signifikansi 0.05) mengindikasikan bahwa keseluruhan model regresi adalah signifikan secara statistik.

Terdapat bukti yang cukup untuk menolak hipotesis nol bahwa tidak ada hubungan antara variabel independen dan variabel dependen secara keseluruhan.

Model regresi secara statistik signifikan dan memiliki kemampuan untuk menjelaskan variabilitas dalam variabel dependen dengan menggunakan variabel independen yang dipilih.

e. Uji Kebaikan Model

Uji Kebaikan Model menggunakan R-squared (R^2) memberikan informasi tentang seberapa baik model regresi dapat menjelaskan variasi dalam data. Nilai R^2 berkisar antara 0 dan 1, dan semakin tinggi nilainya, semakin baik model regresi dapat menjelaskan variasi dalam variabel dependen.

```
import pandas as pd
import statsmodels.api as sm

# Baca data dari CSV
data = pd.read_csv('data_cleaned1.csv')  # Gantilah
'nama_file.csv' dengan nama file CSV yang sesuai

# Pilih variabel independen (X) dan variabel dependen (Y)
X = data[['GrLivArea', 'GarageArea', 'LotArea', 'MasVnrArea',
'TotalBsmtSF']]  # Gantilah dengan variabel independen yang sesuai
Y = data['SalePrice']  # Gantilah dengan variabel dependen yang
sesuai

# Tambahkan kolom bias (intercept) ke matriks X
X = sm.add_constant(X)

# Buat model regresi
model = sm.OLS(Y, X).fit()

# Dapatkan nilai R²
r_squared = model.rsquared

# Tampilkan nilai R²
print(f"Nilai R-squared (R²): {r_squared}")
```

Dan berikut adalah hasilnya ketika di jalankan

```
Nilai R-squared (R²): 0.6597340331584924
```

R-squared (R^2) adalah suatu metrik evaluasi kebaikan model yang mengukur sejauh mana variabilitas dalam variabel dependen dapat dijelaskan oleh model regresi. Dalam konteks ini:
R-squared = 0.6597: Ini berarti sekitar 65.97% dari variasi dalam variabel dependen ("SalePrice") dapat dijelaskan oleh variabel independen yang terpilih dalam model regresi.
Semakin tinggi nilai R-squared, semakin baik model mampu menjelaskan variasi dalam data. Dalam kasus ini, sekitar 66% variasi harga jual ("SalePrice") dijelaskan oleh kombinasi variabel independen yang dipilih.