Import pandas as pd

Import numpy as np

Import matplotlib.pyplot as plt

Import seaborn as sns

From sklearn.model\_selection import train\_test\_split

From sklearn.linear\_model import LinearRegression

From sklearn.metrics import mean\_squared\_error, r2\_score

From sklearn.preprocessing import StandardScaler

Import statsmodels.api as sm

Import pickle

# Load the dataset

url = <https://docs.google.com/spreadsheets/d/1htBZNAk96XHc7V28PD1thdqq8x1wZhqOtP5IyJzxDRM/export?format=csv>

df = pd.read\_csv(url)

# Check the first few rows of the dataset

Print(df.head())

# Preprocessing steps

# Check for missing values

Print(df.isnull().sum())

# Drop rows with missing values

Df = df.dropna()

# Check for linearity between independent and dependent variables

Sns.pairplot(df)

Plt.show()

# Check for multicollinearity

Correlation\_matrix = df.corr().round(2)

Sns.heatmap(data=correlation\_matrix, annot=True)

Plt.show()

# Check for normality of residuals

X = df.drop(‘SalePrice’, axis=1)

Y = df[‘SalePrice’]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# Check for homoscedasticity

Model = LinearRegression()

Model.fit(X\_train\_scaled, y\_train)

Y\_pred\_train = model.predict(X\_train\_scaled)

Residuals\_train = y\_train – y\_pred\_train

Plt.scatter(y\_pred\_train, residuals\_train)

Plt.xlabel(‘Predicted Values’)

Plt.ylabel(‘Residuals’)

Plt.title(‘Residual Plot’)

Plt.show()

# Check for independence of residuals

# Use statsmodels for checking assumptions

X\_train\_scaled = sm.add\_constant(X\_train\_scaled)

Model\_sm = sm.OLS(y\_train, X\_train\_scaled).fit()

Print(model\_sm.summary())

# Calculate evaluation metrics

Y\_pred\_test = model.predict(X\_test\_scaled)

Mse = mean\_squared\_error(y\_test, y\_pred\_test)

Rmse = np.sqrt(mse)

R2 = r2\_score(y\_test, y\_pred\_test)

Print(“Mean Squared Error:”, mse)

Print(“Root Mean Squared Error:”, rmse)

Print(“R-squared:”, r2)

# Save the model

With open(‘housing\_model.pkl’, ‘wb’) as f:

Pickle.dump(model, f)