Subjective Assignment:

Q1. What is the optimal value of alpha for ridge and lasso regression? What will be the changes in the model if you choose double the value of alpha for both ridge and lasso? What will be the most important predictor variables after the change is implemented?

Answer:

Optimal Alpha Values:
The optimal value of alpha (lambda) for Ridge regression is 100, and for Lasso regression is 10, as determined by cross-validation.

Changes with Double Alpha:
When you double the value of alpha for both Ridge and Lasso regression, the regularization strength will increase. This will likely lead to smaller coefficient values in both cases. For Ridge, all coefficients will be shrunk, while for Lasso, some coefficients may be reduced to exactly zero, leading to feature elimination.

Most Important Predictor Variables after Change:
After implementing the change and doubling the value of alpha, you would need to examine the coefficients of the models to identify the most important predictor variables. Features with larger absolute coefficients in the regularized models are considered more important.

```
# Double the value of alpha for Ridge and Lasso
doubled_alpha_for_ridge = 200
doubled_alpha_for_lasso = 20

# Ridge Regression with doubled alpha
ridge_model_doubled_alpha = Ridge(alpha=doubled_alpha_for_ridge)
ridge_model_doubled_alpha.fit(X_train_scaled, y_train)
ridge_predictions_doubled_alpha =
ridge_model_doubled_alpha.predict(X_test_scaled)
ridge_mse_doubled_alpha = mean_squared_error(y_test,
ridge_predictions_doubled_alpha)
ridge_r2_doubled_alpha = r2_score(y_test, ridge_predictions_doubled_alpha)
```

```
# Lasso Regression with doubled alpha
lasso_model_doubled_alpha = Lasso(alpha=doubled_alpha_for_lasso,
max_iter=lasso_grid.best_params_['max_iter'])
lasso_model_doubled_alpha.fit(X_train_scaled, y_train)
lasso_predictions_doubled_alpha =
lasso_model_doubled_alpha.predict(X_test_scaled)
lasso_mse_doubled_alpha = mean_squared_error(y_test,
lasso_predictions_doubled_alpha)
lasso_r2_doubled_alpha = r2_score(y_test, lasso_predictions_doubled_alpha)

# Model evaluation and interpretation for models with doubled alpha
print("\nModel Performance with Doubled Alpha:")
print("Ridge Regression (Doubled Alpha):")
print("RMSE:", ridge_mse_doubled_alpha ** 0.5)
print("R-squared:", ridge_r2_doubled_alpha)
print("\nLasso Regression (Doubled Alpha):")
print("RMSE:", lasso_mse_doubled_alpha ** 0.5)
print("R-squared:", lasso_r2_doubled_alpha)

# Analyze coefficients and feature significance for models with doubled
alpha
ridge_coeff_abs_doubled_alpha = abs(ridge_model_doubled_alpha.coef_)
lasso_coeff_abs_doubled_alpha = abs(lasso_model_doubled_alpha.coef_)

significant_features_ridge_doubled_alpha = [feature for feature,
coefficient in zip(X.columns, ridge_coeff_abs_doubled_alpha) if
coefficient > 0]
significant_features_lasso_doubled_alpha = [feature for feature,
coefficient in zip(X.columns, lasso_coeff_abs_doubled_alpha) if
coefficient > 0]

print("\nSignificant Features from Ridge Regression (Doubled Alpha):")
print(significant_features_ridge_doubled_alpha)

print("\nSignificant Features from Lasso Regression (Doubled Alpha):")
print(significant_features_lasso_doubled_alpha)

# Decide which model to choose based on R-squared and significant features
```

```python
if ridge_r2 > lasso_r2 and len(significant_features_ridge) >
len(significant_features_lasso):
chosen_model = "Ridge"
elif lasso_r2 > ridge_r2 and len(significant_features_lasso) >
len(significant_features_ridge):
chosen_model = "Lasso"
else:
chosen_model = "No clear winner, consider both"

print("\nChosen Model:", chosen_model)

# Evaluate how well the selected variables describe the price of a house
print("\nModel Descriptiveness:")
print("Ridge MSE:", ridge_mse)
print("Ridge R-squared:", ridge_r2)
print("Lasso MSE:", lasso_mse)
print("Lasso R-squared:", lasso_r2)

# Emphasize the significance of the modeling for management's
decision-making
print("\nSignificance for Management:")
if chosen_model != "No clear winner, consider both":
print(f"The {chosen_model} model provides insights into how house prices
vary with key variables.")
else:
print("Both models offer valuable insights. Consider the variables
identified by each.")

# Plot significant features for Ridge and Lasso
plt.figure(figsize=(12, 6))
plt.barh(range(len(significant_features_ridge)),
ridge_coeff_abs[ridge_coeff_abs > 0], color='blue', align='center')
plt.yticks(range(len(significant_features_ridge)),
significant_features_ridge)
plt.xlabel('Coefficient Magnitude')
plt.title('Significant Features from Ridge Regression')
plt.gca().invert_yaxis() # Invert y-axis to have the highest coefficient
at the top
plt.show()
```

```
plt.figure(figsize=(12, 6))
plt.barh(range(len(significant_features_lasso)),
lasso_coeff_abs[lasso_coeff_abs > 0], color='green', align='center')
plt.yticks(range(len(significant_features_lasso)),
significant_features_lasso)
plt.xlabel('Coefficient Magnitude')
plt.title('Significant Features from Lasso Regression')
plt.gca().invert_yaxis() # Invert y-axis to have the highest coefficient
at the top
plt.show()

# Plot the model coefficients during training for Ridge
plt.figure(figsize=(12, 6))
plt.plot(ridge_model.coef_.T, label='Ridge Coefficients')
plt.xlabel('Iterations')
plt.ylabel('Coefficients')
plt.title('Ridge Model Coefficients During Training')
plt.legend()
plt.show()

# Plot the model coefficients during training for Lasso
plt.figure(figsize=(12, 6))
plt.plot(lasso_model.coef_.T, label='Lasso Coefficients')
plt.xlabel('Iterations')
plt.ylabel('Coefficients')
plt.title('Lasso Model Coefficients During Training')
plt.legend()
plt.show()
```

Model Performance with Doubled Alpha:
Ridge Regression (Doubled Alpha):
RMSE: 33155.73541972794
R-squared: 0.8566810900176642

Lasso Regression (Doubled Alpha):
RMSE: 36582.333528244584
R-squared: 0.8255266998791047

Significant Features from Ridge Regression (Doubled Alpha):

['Id', 'MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond', 'YearBuilt',
'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF',
'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'BsmtFullBath',
'BsmtHalfBath', 'FullBath', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr',
'TotRmsAbvGrd', 'Fireplaces', 'GarageYrBlt', 'GarageCars', 'GarageArea',
'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch', 'ScreenPorch',
'PoolArea', 'MiscVal', 'MoSold', 'YrSold', 'MSZoning_FV', 'MSZoning_RH',
'MSZoning_RL', 'MSZoning_RM', 'Street_Pave', 'Alley_Grvl', 'Alley_Pave',
'LotShape_IR2', 'LotShape_IR3', 'LotShape_Reg', 'LandContour_HLS',
'LandContour_Low', 'LandContour_Lvl', 'Utilities_NoSeWa', 'LotConfig_CulDSac',
'LotConfig_FR2', 'LotConfig_FR3', 'LotConfig_Inside', 'LandSlope_Mod',
'LandSlope_Sev', 'Neighborhood_Blueste', 'Neighborhood_BrDale',
'Neighborhood_BrkSide', 'Neighborhood_ClearCr', 'Neighborhood_CollgCr',
'Neighborhood_Crawfor', 'Neighborhood_Edwards', 'Neighborhood_Gilbert',
'Neighborhood_IDOTRR', 'Neighborhood_MeadowV', 'Neighborhood_Mitchel',
'Neighborhood_NAmes', 'Neighborhood_NPkVill', 'Neighborhood_NWAmes',
'Neighborhood_NoRidge', 'Neighborhood_NridgHt', 'Neighborhood_OldTown',
'Neighborhood_SWISU', 'Neighborhood_Sawyer', 'Neighborhood_SawyerW',
'Neighborhood_Somerst', 'Neighborhood_StoneBr', 'Neighborhood_Timber',
'Neighborhood_Veenker', 'Condition1_Feedr', 'Condition1_Norm', 'Condition1_PosA',
'Condition1_PosN', 'Condition1_RRAe', 'Condition1_RRAn', 'Condition1_RRNe',
'Condition1_RRNn', 'Condition2_Feedr', 'Condition2_Norm', 'Condition2_PosA',
'Condition2_PosN', 'Condition2_RRAe', 'Condition2_RRAn', 'Condition2_RRNn',
'BldgType_2fmCon', 'BldgType_Duplex', 'BldgType_Twnhs', 'BldgType_TwnhsE',
'HouseStyle_1.5Unf', 'HouseStyle_1Story', 'HouseStyle_2.5Fin', 'HouseStyle_2.5Unf',
'HouseStyle_2Story', 'HouseStyle_SFoyer', 'HouseStyle_SLvl', 'RoofStyle_Gable',
'RoofStyle_Gambrel', 'RoofStyle_Hip', 'RoofStyle_Mansard', 'RoofStyle_Shed',
'RoofMatl_CompShg', 'RoofMatl_Metal', 'RoofMatl_Roll', 'RoofMatl_Tar&Grv',
'RoofMatl_WdShake', 'RoofMatl_WdShngl', 'Exterior1st_AsphShn',
'Exterior1st_BrkComm', 'Exterior1st_BrkFace', 'Exterior1st_CBlock',
'Exterior1st_CemntBd', 'Exterior1st_HdBoard', 'Exterior1st_ImStucc',
'Exterior1st_MetalSd', 'Exterior1st_Plywood', 'Exterior1st_Stone',
'Exterior1st_Stucco', 'Exterior1st_VinylSd', 'Exterior1st_Wd Sdng',
'Exterior1st_WdShing', 'Exterior2nd_AsphShn', 'Exterior2nd_Brk Cmn',
'Exterior2nd_BrkFace', 'Exterior2nd_CBlock', 'Exterior2nd_CmentBd',
'Exterior2nd_HdBoard', 'Exterior2nd_ImStucc', 'Exterior2nd_MetalSd',

'Exterior2nd_Other', 'Exterior2nd_Plywood', 'Exterior2nd_Stone', 'Exterior2nd_Stucco', 'Exterior2nd_VinylSd', 'Exterior2nd_Wd Sdng', 'Exterior2nd_Wd Shng', 'MasVnrType_BrkCmn', 'MasVnrType_BrkFace', 'MasVnrType_Stone', 'ExterQual_Fa', 'ExterQual_Gd', 'ExterQual_TA', 'ExterCond_Fa', 'ExterCond_Gd', 'ExterCond_Po', 'ExterCond_TA', 'Foundation_CBlock', 'Foundation_PConc', 'Foundation_Slab', 'Foundation_Stone', 'Foundation_Wood', 'BsmtQual_Ex', 'BsmtQual_Fa', 'BsmtQual_Gd', 'BsmtQual_TA', 'BsmtCond_Fa', 'BsmtCond_Gd', 'BsmtCond_Po', 'BsmtCond_TA', 'BsmtExposure_Av', 'BsmtExposure_Gd', 'BsmtExposure_Mn', 'BsmtExposure_No', 'BsmtFinType1_ALQ', 'BsmtFinType1_BLQ', 'BsmtFinType1_GLQ', 'BsmtFinType1_LwQ', 'BsmtFinType1_Rec', 'BsmtFinType1_Unf', 'BsmtFinType2_ALQ', 'BsmtFinType2_BLQ', 'BsmtFinType2_GLQ', 'BsmtFinType2_LwQ', 'BsmtFinType2_Rec', 'BsmtFinType2_Unf', 'Heating_GasA', 'Heating_GasW', 'Heating_Grav', 'Heating_OthW', 'Heating_Wall', 'HeatingQC_Fa', 'HeatingQC_Gd', 'HeatingQC_Po', 'HeatingQC_TA', 'CentralAir_Y', 'Electrical_FuseA', 'Electrical_FuseF', 'Electrical_FuseP', 'Electrical_SBrkr', 'KitchenQual_Fa', 'KitchenQual_Gd', 'KitchenQual_TA', 'Functional_Maj2', 'Functional_Min1', 'Functional_Min2', 'Functional_Mod', 'Functional_Sev', 'Functional_Typ', 'FireplaceQu_Ex', 'FireplaceQu_Fa', 'FireplaceQu_Gd', 'FireplaceQu_Po', 'FireplaceQu_TA', 'GarageType_2Types', 'GarageType_Attchd', 'GarageType_Basment', 'GarageType_BuiltIn', 'GarageType_CarPort', 'GarageType_Detchd', 'GarageFinish_Fin', 'GarageFinish_RFn', 'GarageFinish_Unf', 'GarageQual_Ex', 'GarageQual_Fa', 'GarageQual_Gd', 'GarageQual_Po', 'GarageQual_TA', 'GarageCond_Ex', 'GarageCond_Fa', 'GarageCond_Gd', 'GarageCond_Po', 'GarageCond_TA', 'PavedDrive_P', 'PavedDrive_Y', 'PoolQC_Ex', 'PoolQC_Fa', 'PoolQC_Gd', 'Fence_GdPrv', 'Fence_GdWo', 'Fence_MnPrv', 'Fence_MnWw', 'MiscFeature_Gar2', 'MiscFeature_Othr', 'MiscFeature_Shed', 'MiscFeature_TenC', 'SaleType_CWD', 'SaleType_Con', 'SaleType_ConLD', 'SaleType_ConLI', 'SaleType_ConLw', 'SaleType_New', 'SaleType_Oth', 'SaleType_WD', 'SaleCondition_AdjLand', 'SaleCondition_Alloca', 'SaleCondition_Family', 'SaleCondition_Normal', 'SaleCondition_Partial']

Significant Features from Lasso Regression (Doubled Alpha):
['Id', 'MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'TotalBsmtSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',

'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces', 'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MoSold', 'YrSold', 'MSZoning_FV', 'MSZoning_RH', 'MSZoning_RL', 'MSZoning_RM', 'Street_Pave', 'Alley_Grvl', 'Alley_Pave', 'LotShape_IR2', 'LotShape_IR3', 'LotShape_Reg', 'LandContour_HLS', 'LandContour_Low', 'LandContour_Lvl', 'Utilities_NoSeWa', 'LotConfig_CulDSac', 'LotConfig_FR2', 'LotConfig_FR3', 'LotConfig_Inside', 'LandSlope_Mod', 'LandSlope_Sev', 'Neighborhood_BrDale', 'Neighborhood_BrkSide', 'Neighborhood_ClearCr', 'Neighborhood_CollgCr', 'Neighborhood_Crawfor', 'Neighborhood_Edwards', 'Neighborhood_Gilbert', 'Neighborhood_IDOTRR', 'Neighborhood_MeadowV', 'Neighborhood_Mitchel', 'Neighborhood_NAmes', 'Neighborhood_NPkVill', 'Neighborhood_NWAmes', 'Neighborhood_NoRidge', 'Neighborhood_NridgHt', 'Neighborhood_OldTown', 'Neighborhood_SWISU', 'Neighborhood_Sawyer', 'Neighborhood_SawyerW', 'Neighborhood_Somerst', 'Neighborhood_StoneBr', 'Neighborhood_Timber', 'Neighborhood_Veenker', 'Condition1_Feedr', 'Condition1_Norm', 'Condition1_PosA', 'Condition1_PosN', 'Condition1_RRAe', 'Condition1_RRAn', 'Condition1_RRNe', 'Condition1_RRNn', 'Condition2_Feedr', 'Condition2_Norm', 'Condition2_PosA', 'Condition2_PosN', 'Condition2_RRAe', 'Condition2_RRAn', 'Condition2_RRNn', 'BldgType_2fmCon', 'BldgType_Duplex', 'BldgType_Twnhs', 'BldgType_TwnhsE', 'HouseStyle_1.5Unf', 'HouseStyle_1Story', 'HouseStyle_2.5Fin', 'HouseStyle_2.5Unf', 'HouseStyle_2Story', 'HouseStyle_SFoyer', 'HouseStyle_SLvl', 'RoofStyle_Gambrel', 'RoofStyle_Hip', 'RoofStyle_Mansard', 'RoofStyle_Shed', 'RoofMatl_Metal', 'RoofMatl_Tar&Grv', 'RoofMatl_WdShngl', 'Exterior1st_BrkComm', 'Exterior1st_BrkFace', 'Exterior1st_CBlock', 'Exterior1st_HdBoard', 'Exterior1st_ImStucc', 'Exterior1st_Plywood', 'Exterior1st_Stone', 'Exterior1st_Stucco', 'Exterior1st_VinylSd', 'Exterior1st_Wd Sdng', 'Exterior1st_WdShing', 'Exterior2nd_AsphShn', 'Exterior2nd_Brk Cmn', 'Exterior2nd_BrkFace', 'Exterior2nd_CBlock', 'Exterior2nd_CmentBd', 'Exterior2nd_HdBoard', 'Exterior2nd_ImStucc', 'Exterior2nd_MetalSd', 'Exterior2nd_Other', 'Exterior2nd_Plywood', 'Exterior2nd_Stone', 'Exterior2nd_Stucco', 'Exterior2nd_VinylSd', 'Exterior2nd_Wd Sdng', 'Exterior2nd_Wd Shng', 'MasVnrType_BrkCmn', 'MasVnrType_BrkFace', 'ExterQual_Fa', 'ExterQual_Gd', 'ExterQual_TA', 'ExterCond_Gd', 'ExterCond_Po', 'Foundation_CBlock', 'Foundation_PConc', 'Foundation_Slab', 'Foundation_Stone', 'Foundation_Wood', 'BsmtQual_Ex', 'BsmtQual_Gd', 'BsmtQual_TA', 'BsmtCond_Fa', 'BsmtCond_Gd', 'BsmtCond_Po', 'BsmtCond_TA', 'BsmtExposure_Gd', 'BsmtExposure_Mn', 'BsmtExposure_No', 'BsmtFinType1_ALQ', 'BsmtFinType1_BLQ', 'BsmtFinType1_GLQ', 'BsmtFinType1_LwQ', 'BsmtFinType1_Rec', 'BsmtFinType2_ALQ', 'BsmtFinType2_BLQ', 'BsmtFinType2_LwQ', 'BsmtFinType2_Rec', 'BsmtFinType2_Unf', 'Heating_GasW', 'Heating_Grav', 'Heating_OthW', 'Heating_Wall', 'HeatingQC_Fa', 'HeatingQC_Gd', 'HeatingQC_Po', 'HeatingQC_TA', 'CentralAir_Y', 'Electrical_FuseF', 'Electrical_FuseP', 'Electrical_SBrkr', 'KitchenQual_Fa', 'KitchenQual_Gd', 'KitchenQual_TA', 'Functional_Maj2', 'Functional_Min2', 'Functional_Mod', 'Functional_Sev', 'Functional_Typ', 'FireplaceQu_Ex', 'FireplaceQu_Fa', 'FireplaceQu_Gd', 'FireplaceQu_TA', 'GarageType_2Types', 'GarageType_Attchd', 'GarageType_Basment', 'GarageType_BuiltIn', 'GarageType_CarPort', 'GarageFinish_RFn', 'GarageFinish_Unf', 'GarageQual_Ex', 'GarageQual_Fa', 'GarageQual_Gd', 'GarageQual_Po', 'GarageCond_Ex', 'GarageCond_Fa', 'GarageCond_Gd', 'GarageCond_Po', 'PavedDrive_P', 'PavedDrive_Y', 'PoolQC_Ex', 'PoolQC_Fa', 'PoolQC_Gd', 'Fence_GdPrv', 'Fence_GdWo', 'Fence_MnPrv', 'Fence_MnWw', 'MiscFeature_Gar2', 'MiscFeature_Othr', 'MiscFeature_Shed', 'MiscFeature_TenC', 'SaleType_CWD', 'SaleType_Con', 'SaleType_ConLD', 'SaleType_ConLI', 'SaleType_ConLw', 'SaleType_New', 'SaleType_Oth', 'SaleType_WD', 'SaleCondition_AdjLand', 'SaleCondition_Alloca', 'SaleCondition_Family', 'SaleCondition_Normal', 'SaleCondition_Partial']

Chosen Model: Ridge

Model Descriptiveness:
Ridge MSE: 1102283933.6316578
Ridge R-squared: 0.8562924308748676
Lasso MSE: 1367962588.9054093
Lasso R-squared: 0.821655226654686

Significance for Management:
The Ridge model provides insights into how house prices vary with key variables.

Q2. You have determined the optimal value of lambda for ridge and lasso regression during the assignment. Now, which one will you choose to apply and why?

Answer:

Based on the provided code and output, you have determined the optimal values of lambda (alpha) for Ridge and Lasso regression using cross-validation. The optimal lambda values are 100 for Ridge regression and 10 for Lasso regression. You then applied both Ridge and Lasso regression models to the dataset and evaluated their performance. Here's a summary of your findings and the model choice:

1. Performance Evaluation:

Ridge Regression:
- Root Mean Squared Error (RMSE): 33200.66
- R-squared (R2) value: 0.8563

Lasso Regression:
- RMSE: 36985.98
- R2 value: 0.8217

2. Significant Features:
- Ridge Regression: You have identified a substantial number of significant features using Ridge regression.
- Lasso Regression: Lasso regression has also highlighted a set of significant features.

3. Model Choice:
- Based on the comparison of R-squared values and the number of significant features, you have chosen the Ridge model. The Ridge model outperformed the Lasso model in terms of R-squared, and it identified more significant features.

4. Model Descriptiveness:
- The Ridge model's R-squared value of 0.8563 indicates that approximately 85.63% of the variance in the target variable (SalePrice) can be explained by the features in the model.

5. Significance for Management:

- I emphasize the significance of the Ridge model for management's decision-making. The Ridge model provides insights into how house prices vary with key variables, offering valuable information that can be used to make informed decisions.

Overall, I have chosen the Ridge regression model due to its better performance in terms of R-squared and the number of significant features. The insights from this model can be valuable for understanding the relationships between the input features and house prices, which can aid in making data-driven decisions in real estate management or related fields.

Q3. After building the model, you realised that the five most important predictor variables in the lasso model are not available in the incoming data. You will now have to create another model excluding the five most important predictor variables. Which are the five most important predictor variables now?

Answer :

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error

# Load data
data = pd.read_csv('train.csv')

# Handle missing values (replace with 0 for simplicity)
data.fillna(0, inplace=True)

# Convert categorical variables using one-hot encoding
categorical_cols = data.select_dtypes(include=['object']).columns
data_encoded = pd.get_dummies(data, columns=categorical_cols,
drop_first=True)

# Split data
```

```
X = data_encoded.drop('SalePrice', axis=1)
y = data_encoded['SalePrice']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Standardize features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Train Lasso model to identify important features
lasso_model = Lasso(alpha=10) # Use the optimal alpha identified
previously
lasso_model.fit(X_train_scaled, y_train)

# Get the indices of the top 5 important features
top_5_indices = lasso_model.coef_.argsort()[-5:][::-1]

# Get the names of the top 5 important features
top_5_features = X.columns[top_5_indices]

print("Top 5 most important predictor variables from Lasso model:")
print(top_5_features)

# Exclude the top 5 features from the dataset
X_train_without_top_5 = X_train_scaled[:, ~top_5_indices]
X_test_without_top_5 = X_test_scaled[:, ~top_5_indices]

# Train a new Lasso model without the top 5 features
lasso_model_without_top_5 = Lasso(alpha=10) # Use the optimal alpha
identified previously
lasso_model_without_top_5.fit(X_train_without_top_5, y_train)

# Predict and evaluate the model
lasso_predictions_without_top_5 =
lasso_model_without_top_5.predict(X_test_without_top_5)
lasso_mse_without_top_5 = mean_squared_error(y_test,
lasso_predictions_without_top_5)

print("\nLasso Regression without top 5 features:")
```

```
print("RMSE:", lasso_mse_without_top_5 ** 0.5)
```

```
Top 5 most important predictor variables from Lasso model:
Index(['PoolArea', 'RoofMatl_CompShg', 'RoofMatl_Tar&Grv', '2ndFlrSF',
       'RoofMatl_WdShngl'],
      dtype='object')

Lasso Regression without top 5 features:
RMSE: 86920.36052855481
```

Q4: How can you make sure that a model is robust and generalisable? What are the implications of the same for the accuracy of the model and why?

Answer:

Based on the results of your analysis, you have determined the optimal values of lambda (alpha) for Ridge and Lasso regression. The optimal lambda for Ridge is 100, and for Lasso, it's 10. After evaluating both models, you've found that the Ridge model has a lower RMSE (Root Mean Squared Error) and a higher R-squared value compared to the Lasso model. Additionally, the Ridge model has identified more significant features. Based on these results, you have chosen the Ridge model as the preferred model for making predictions on house prices.

The reason for choosing Ridge over Lasso in this case is mainly because Ridge performed better in terms of RMSE and R-squared. A lower RMSE indicates that the Ridge model's predictions are closer to the actual target values, and a higher R-squared suggests that the Ridge model explains more of the variance in the target variable.

```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.linear_model import Ridge
from sklearn.preprocessing import StandardScaler

# Load and preprocess data
data = pd.read_csv('train.csv')
```

```
data.fillna(0, inplace=True)
categorical_cols = data.select_dtypes(include=['object']).columns
data_encoded = pd.get_dummies(data, columns=categorical_cols,
drop_first=True)
X = data_encoded.drop('SalePrice', axis=1)
y = data_encoded['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)

# Ridge Regression with Cross-Validation
ridge_model = Ridge(alpha=100)
ridge_cv_scores = cross_val_score(ridge_model, X_train_scaled, y_train,
cv=5, scoring='neg_mean_squared_error')
ridge_rmse_scores = (-ridge_cv_scores)**0.5
ridge_mean_rmse = ridge_rmse_scores.mean()

# Evaluate on the test set
ridge_model.fit(X_train_scaled, y_train)
ridge_test_predictions = ridge_model.predict(X_test_scaled)
ridge_test_rmse = mean_squared_error(y_test, ridge_test_predictions,
squared=False)

print("Ridge Cross-Validation RMSE:", ridge_mean_rmse)
print("Ridge Test RMSE:", ridge_test_rmse)
```

Ridge Cross-Validation RMSE: 36927.4995638409
Ridge Test RMSE: 33200.66164448621