Question 1

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?

```
optimal_ridge_alpha = ridge_cv_model.best_params_['alpha']
md("Optimal value of alpha for ridge regression: {} ".format(optimal_ridge_alpha
```

Optimal value of alpha for ridge regression: 2.0

```
optimal_lasso_alpha = lasso_cv_model.best_params_['alpha']
md("Optimal value of alpha for lasso regression: {}".format(optimal_lasso_alpha)
```

Optimal value of alpha for lasso regression: 0.0001

```
ridge_model = Ridge(alpha=optimal_ridge_alpha*2)
print('Training Ridge Model with double the optimal alpha.')
model, scores = fit_evalaute_one_model(ridge_model, X_train_rfe, X_test_rfe, y_t
print(f'Results: {scores}')
```

```
Training Ridge Model with double the optimal alpha.
Results: {'model': 'Ridge', 'training_r2': 0.8178398491686983, 'testing_r2': 0.80
59768525422892}
```

```
lasso_model = Lasso(alpha=optimal_lasso_alpha*2)
print('Training Lasso Model with double the optimal alpha.')
model, scores = fit_evalaute_one_model(lasso_model, X_train_rfe, X_test_rfe, y_t
print(f'Results: {scores}')
```

```
Training Lasso Model with double the optimal alpha.
Results: {'model': 'Lasso', 'training_r2': 0.8276976693557124, 'testing_r2': 0.81
91657988139053}
```

We can see that the testing r2 score of both lasso and ridge decreases when the optimum alpha value is doubled.

Testing R2 score of ridge decreased from `0.87` to `0.86` while testing R2 score of lasso decreased from `0.88` to `0.87`

Question 2

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?

After finding the optimal values of alpha for both RIDGE and LASSSO, we compare both their best scores using `model.best_score_` and pick which ever gives the highest value. In our case, the best score for lasso is *0.8806* and the best score for ridge is *0.8705*.

Moreover the difference in R2 Score between training and testing is not too much and also it can help in feature selection.

Hence we pick LASSO as it gives a better score compared to RIDGE on the testing data.

Question 3

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?

The top 5 Lasso selected features that define a high `SalePrice` are:

1. **`GrLivArea`**
2. **`OverallQual`**
3. **`GarageCars`**
4. **`OverallCond`**
5. **`LotArea`**

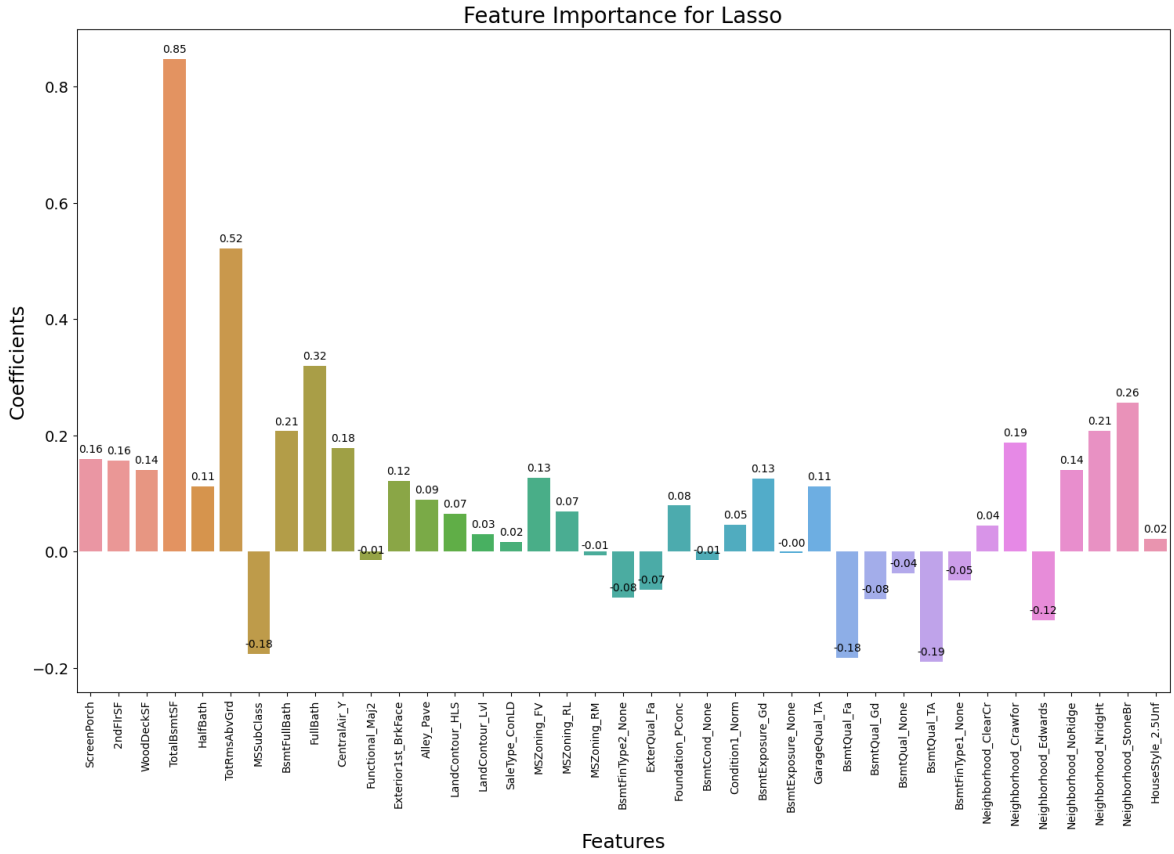
```
# Build lasso model without the top features
X_train_rfe = X_train_rfe.drop(columns=['GrLivArea','OverallQual','GarageCars','
X_test_rfe = X_test_rfe.drop(columns=['GrLivArea','OverallQual','GarageCars','Ov
```

```
lasso_model_new_variables = Lasso(alpha = 0.001)
print('Training Lasso Model with optimal alpha......')
print()
model,scores = fit_evalaute_one_model(lasso_model_new_variables,X_train_rfe, X_t
print(f'Results: {scores}')
```

```
Training Lasso Model with optimal alpha......

Results: {'model': 'Lasso', 'training_r2': 0.8108408600020477, 'testing_r2': 0.82
01890496993058}
```

```
lasso_model_new_variables = interpret_model(lasso_model_new_variables)
# taking into consideration the negative values as well.
lasso_model_new_variables['Coef'] = lasso_model_new_variables['Coef'].abs()
lasso_model_new_variables.sort_values(by='Coef',ascending=False).head(10)
```



	Feature	Coef
0	constant	11.117
5	TotalBsmtSF	0.847
8	TotRmsAbvGrd	0.522
11	FullBath	0.320
59	Neighborhood_StoneBr	0.256
58	Neighborhood_NridgHt	0.208
10	BsmtFullBath	0.207
52	BsmtQual_TA	0.190
55	Neighborhood_Crawfor	0.187
49	BsmtQual_Fa	0.183

The 5 most important features that define a high `SalePrice` **now** are:

1. **`TotalBsmtSF`**
2. **`TotRmsAbvGrd`**
3. **`FullBath`**
4. **`Neighborhood_StoneBr`**
5. **`Neighborhood_NridgHt`**

Question 4

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?

When working with advanced linear regression techniques like Lasso and Ridge, there are several strategies you can employ to achieve robustness and generalizability:

- **Feature Selection and Regularization:** Lasso and Ridge regression are regularization techniques that help prevent overfitting by adding a penalty term to the linear regression loss function.
- **Cross-Validation:** Use techniques like k-fold cross-validation just as used in this project to assess the model's performance on multiple subsets of the data.
- **Scaling:** Scaling features can help prevent the dominance of certain features due to their different scales. Standardizing features ensures that the regularization terms in Lasso and Ridge are applied consistently across all features, promoting better convergence and generalization.

The **implications** of robustness and generalizability for the accuracy of a model are that a robust model is less likely to make mistakes due to noise or outliers, and a generalizable model is more likely to make accurate predictions on new data.

For example, if a model is not robust, it may make inaccurate predictions if the data it is tested on contains noise or outliers. Similarly, if a model is not generalizable, it may make inaccurate predictions on new data that is different from the training data.

Therefore, it is important to make sure that a model is both robust and generalizable in order to **maximize** its accuracy.