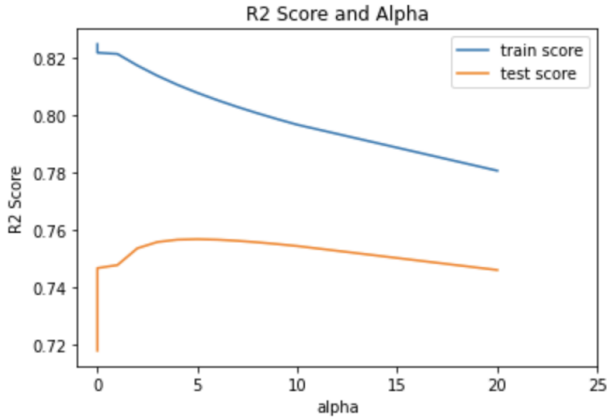
Subjective Questions - Demonstration/Validation

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?

Ans: The Optimal value of alpha for ridge is 5 and for lasso it is 0.0005




```
The optimum alpha is 5
The R2 Score of the model on the test dataset for optimum alpha is 0.7009502819383349
The MSE of the model on the test dataset for optimum alpha is 0.04988484385975613
```

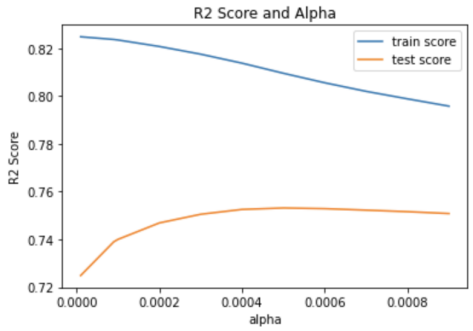
And for Lasso:

```
# cross validation
folds = 11
model_cv = GridSearchCV(estimator = lasso, param_grid = params, scoring= 'r2', cv = folds, return_train_score=True,
model_cv.fit(X_train_rfe3, y_train)

cv_results = pd.DataFrame(model_cv.cv_results_)
# plotting
plt.plot(cv_results['param_alpha'], cv_results['mean_train_score'])
plt.plot(cv_results['param_alpha'], cv_results['mean_test_score'])
plt.xlabel('alpha')
plt.ylabel('R2 Score')
plt.title("R2 Score and Alpha")
plt.legend(['train score', 'test score'], loc='upper right')
plt.show()

alpha = cv_results['param_alpha'].loc[cv_results['mean_test_score'].idxmax()]
print('The optimum alpha is',alpha)
lasso_final2 = Lasso(alpha=alpha,random_state=100)
lasso_final2.fit(X_train_rfe3,y_train)
lasso_coef2 = lasso_final2.coef_
y_test_pred = lasso_final2.predict(X_test_rfe3)
print('The R2 Score of the model on the test dataset for optimum alpha is',r2_score(y_test, y_test_pred))
print('The MSE of the model on the test dataset for optimum alpha is', mean_squared_error(y_test, y_test_pred))
```

```
Fitting 11 folds for each of 11 candidates, totalling 121 fits
```




```
The optimum alpha is 0.0005
The R2 Score of the model on the test dataset for optimum alpha is 0.7019242026257326
The MSE of the model on the test dataset for optimum alpha is 0.04972238297620295
```

After using the double value of alpha for Ridge, i.e 10, we get the R2 score to 0.69 and the following coefficients changed.

```
The R2 Score of the model on the test dataset for doubled alpha is 0.6913997394905503
The MSE of the model on the test dataset for doubled alpha is 0.05147798135499183
The most important predictor variables are as follows:
```

Out[93]:

	Ridge Doubled Alpha Co-Efficient
LotArea	0.225785
Total_porch_sf	0.224619
BsmtQual_Ex	0.154585
MasVnrType_Stone	0.131655
KitchenQual_Ex	0.129901
BsmtFullBath	0.117964
HouseStyle_2Story	0.106043
OpenPorchSF	0.103003
Neighborhood_StoneBr	0.102884
ExterQual_Ex	0.096855
MasVnrType_BrkFace	0.093342
HouseStyle_2.5Unf	0.083808
MSSubClass_70	0.080999
HouseStyle_1.5Fin	0.075827
Neighborhood_Veenker	0.075147
RoofStyle_Hip	0.071825
SaleCondition_Alloca	0.065370
HouseStyle_2.5Fin	0.059465
Condition1_PosA	0.058534
LandContour_HLS	0.058499

For Lasso, after we used 0.001, we got R2 score as 0.68 and following coefficients changed.

```
lasso_double_coef = lasso_double.coef_
y_test_pred = lasso_double.predict(X_test_rfe3)
print('The R2 Score of the model on the test dataset for doubled alpha is',r2_score(y_test, y_test_pred))
print('The MSE of the model on the test dataset for doubled alpha is', mean_squared_error(y_test, y_test_pred))
lasso_double_coeff = pd.DataFrame(np.atleast_2d(lasso_double_coef),columns=X_train_rfe3.columns)
lasso_double_coeff = lasso_double_coeff.T
lasso_double_coeff.rename(columns={0: 'Lasso Doubled Alpha Co-Efficient'},inplace=True)
lasso_double_coeff.sort_values(by=['Lasso Doubled Alpha Co-Efficient'], ascending=False,inplace=True)
print('The most important predictor variables are as follows:')
lasso_double_coeff.head(20)
```

```
The R2 Score of the model on the test dataset for doubled alpha is 0.6891077996505619
The MSE of the model on the test dataset for doubled alpha is 0.0518603026017559
The most important predictor variables are as follows:
```

)4]:

	Lasso Doubled Alpha Co-Efficient
LotArea	0.629725
Total_porch_sf	0.282125
BsmtQual_Ex	0.169981
MasVnrType_Stone	0.132529
KitchenQual_Ex	0.127909
HouseStyle_2Story	0.103650
Neighborhood_StoneBr	0.100902
BsmtFullBath	0.099962
MasVnrType_BrkFace	0.087908
ExterQual_Ex	0.084424
OpenPorchSF	0.073635
RoofStyle_Hip	0.065721
HouseStyle_2.5Unf	0.060496
MSSubClass_70	0.058176
HouseStyle_1.5Fin	0.056733
LandContour_HLS	0.048808
LotConfig_CulDSac	0.020628
Alley_Pave	0.013167
SaleCondition_Partial	0.011327
Foundation_BrkTil	0.011031

The most important predictors after the changes will be:
LotArea, Total_porch_sf, BsmtQual_Ex, MasVnrType_Stone and the rest as shown above.

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?

The optimal value of Alpha for Ridge is 5 and Lasso is 0.001

MSE for Ridge is: 0.04988484385975613
MSE for Lasso is: 0.04972238297620295

Since, MSE for Lasso is small and since it's helpful in feature reduction and selection, I would choose Lasso Regression.

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 most important predictor variables after building the model again are:
LotFrontage
Total_porch_sf
HouseStyle_2.5Unf
HouseStyle_2.5Fin
Neighbourhood_Veenker

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?

As Per, Occam's Razor— given two models that show similar 'performance' in the finite training or test
data, we should pick the one that makes fewer on the test data due to following reasons:-

- Simpler models are usually more 'generic' and are more widely applicable
- Simpler models require fewer training samples for effective training than the more complex ones and hence are easier to train.
- Simpler models are more robust.
- Complex models tend to change wildly with changes in the training data set
- Simple models have low variance, high bias and complex models have low bias, high variance
- Simpler models make more errors in the training set. Complex models lead to overfitting —
    - they work very well for the training samples, fail miserably when applied to other test samples

Therefore, to make the model more robust and generalizable, make the model simple but not simpler which will not be of any use.

Regularization can be used to make the model simpler. Regularization helps to strike the delicate balance between keeping the model simple and not making it too naive to be of any use. For regression, regularization involves adding a regularization term to the cost that adds up the absolute values or the squares of the parameters of the model.
Also, Making a model simple leads to Bias-Variance Trade-off:

- A complex model will need to change for every little change in the dataset and hence is very unstable and extremely sensitive to any changes in the training data.
- A simpler model that abstracts out some pattern followed by the data points given is unlikely to change wildly even if more points are added or removed.