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?

Answer:
The optimal value of lambda for ridge is 0.01

The optimal value of lambda for lasso regression is 0.0001

The top 5 features are the top 5 predictor variables for this model.

```
final.sort_values(by='Coeffs', ascending=False).head(15)
```

	Features	Coeffs
10	GrLivArea	0.500826
2	YearBuilt	0.120109
13	GarageArea	0.098145
7	1stFlrSF	0.089926
22	Neighborhood_Crawfor	0.059734
3	BsmtFinSF1	0.054160
14	WoodDeckSF	0.050776
43	Functional_Typ	0.044558
17	MSZoning_ResiLow	0.040623
51	Exterior1st_BrkFace	0.040261
24	Neighborhood_Somerst	0.034571
66	RoofMatl_WdShngl	0.032875
20	LandSlope_Sev	0.026592
25	Neighborhood_StoneBr	0.026340
72	GarageQual_Gd	0.022935

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?
Answer:
The optimal value of lambda for ridge regression is 0.01 and the optimal value of lambda for lasso regression is 0.0001. But as Lasso has its test and train accuracy closer, we will opt for Lasso model for our 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?
Answer:

After deleting the top 5 predictor values as shown below, we get the next feature models below that:

```
final.sort_values(by='Coeffs', ascending=False).head(15)
```

	Features	Coeffs
10	GrLivArea	0.500826
2	YearBuilt	0.120109
13	GarageArea	0.098145
7	1stFlrSF	0.089926
22	Neighborhood_Crawfor	0.059734
3	BsmtFinSF1	0.054160
14	WoodDeckSF	0.050776
43	Functional_Typ	0.044558
17	MSZoning_ResiLow	0.040623
51	Exterior1st_BrkFace	0.040261
24	Neighborhood_Somerst	0.034571
66	RoofMatl_WdShngl	0.032875
20	LandSlope_Sev	0.026592
25	Neighborhood_StoneBr	0.026340
72	GarageQual_Gd	0.022935

After deleting the top 5 predictor variables we get below predictor variables:

```
final.sort_values(by='Coeffs', ascending=False).head(15)
```

	Features	Coeffs
5	TotalBsmtSF	0.717965
61	RoofMatl_WdShngl	0.256351
60	RoofMatl_WdShake	0.252055
56	RoofMatl_Membran	0.233516
59	RoofMatl_Tar&Grv	0.216183
6	2ndFlrSF	0.211142
55	RoofMatl_CompShg	0.202811
57	RoofMatl_Metal	0.183743
1	LotArea	0.140694
2	BsmtFinSF1	0.089136
58	RoofMatl_Roll	0.077872
14	MSZoning_ResiLow	0.066092
44	BsmtFinType1_NA	0.064770
9	GarageYrBlt	0.061468

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?

Answer:
We can make sure our model is more robust and generalisable, by ensuring that the model is as simple as possible but also provide good prediction of the target field.

We can do that by cleaning our data of any outliers and null values and building a simple model on this data which gives a test score for r-squared similar to that of the train set.

That is there is no overfitting and much complexity in the data.