I. A description of how to handle the missing values in your code and report the result.

1. Description

The characteristics of missing values in the data set are as follows:

1. There are three features that have missing value problems: OverallQual, OverallCond, YearBuilt. All of these features are available as numeric data types.
2. The rate of missing data is less than 6% of data set.
3. The type of missing data is missing completely at random.

Solutions to handle the missing values:

Solution 1: delete rows with missing values because

1. The rate of missing data is less than 6% of data set.
2. The type of missing data is missing completely at random.

Step 1: Convert datatype of non-numeric columns to numeric, all missing values will be converted to NaN.

```
# List columns to convert
cols = ['YearBuilt', 'OverallQual', 'OverallCond']

# Convert datatype in training set
training_set[cols] = training_set[cols].apply(pd.to_numeric, errors='coerce', axis=1)

# Convert datatype in test set
test_set[cols] = test_set[cols].apply(pd.to_numeric, errors='coerce', axis=1)
```

Step 2: Remove rows that contains na values

```
# Drop rows with missing value values
training_set_solution_1 = training_set.dropna(axis=0)
test_set_solution_1 = test_set.dropna(axis=0)
```

Solution 2: fill missing values based on other columns because

1. Column **YearBuilt**:
   - The YearBuilt column is related to the YearRemodAdd column.
   - If there was no remodelling or additions, YearRemodAdd equals YearBuilt.

   => For the values of the YearBuilt column that are missing, the value will be replaced with the value of the YeaRemodAdd column.

2. Columns **OverallQual** and **OverallCond**:
   - The values in the two columns OverallQual and OverallCond are not too big difference.
   - There are no cases where both columns contain missing values

   => Replacing the missing value of the OverallQual column with the value in the OverallCond column and vice versa

Step 1: using numpy.where method to replace missing values

Create imputation function to replace missing value by using numpy.where

```
def imputation(data_set):
    data_set['YearBuilt'] = np.where(
        data_set['YearBuilt'].isnull(),
        data_set['YearRemodAdd'],
        data_set['YearBuilt'])

    data_set['OverallQual'] = np.where(
        data_set['OverallQual'].isnull(),
        data_set['OverallCond'],
        data_set['OverallQual'])

    data_set['OverallCond'] = np.where(
        data_set['OverallCond'].isnull(),
        data_set['OverallQual'],
        data_set['OverallCond'])
```

2. Report the result:

RMSE of both solutions

Solution 1		Solution 2	
Training RMSE	**Cross Validation RMSE**	**Training RMSE**	**Cross Validation RMSE**
29290.372731	30645.350614	29774.731629	31012.044177

The RMSE of both solutions is not too different. However, RMSE cross validation of solution 1 is smaller than solution 2 so solution 1 is the better method in this case.

II. A description of the regression technique you used and report the results.

1. Linear regression

It is a simple linear model that assumes a linear relationship between the input variables and the single output variable. This model fits a linear model with coefficients $w = (w_1, w_2, \dots, w_p)$ in order to minimize the residual sum of squares between the observed targets and the predicted targets (ordinary least squares).

In my assignment, I used Linear regression in a pipeline with a standard scaler to transform data to have a mean value of 0 and standard deviation of 1. I also use default hyperarameters.

```
# Create pipeline for training model with scaler and linear regression
pl_linear = Pipeline ([
    ('scaler', StandardScaler()),
    ('linear', LinearRegression())
])
```

2. Ridge regression

Ridge regression is similar to linear regression. This model solves a regression model where the loss function is the linear least squares function and regularization is given by the l2-norm. It selects coefficients such that they are kept as small as possible, so that features will have minimal impact on the output variable.

I used Ridge regression in a pipeline with the same standard scaler and default hyperparameters.

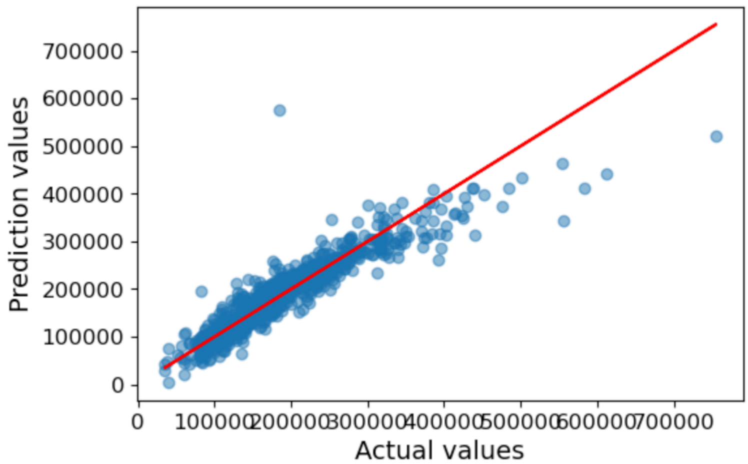
```
from sklearn.linear_model import Ridge

# Create a pipeline using Ridge model
pl_ridge = Pipeline ([
    ('scaler',StandardScaler()),
    ('ridge', Ridge())
])
```

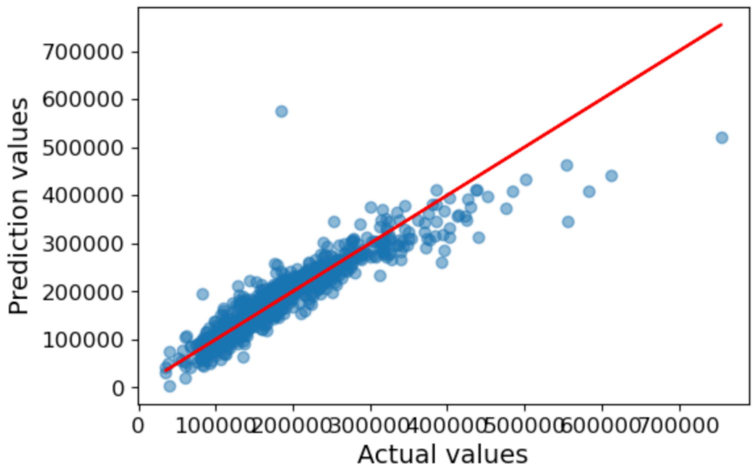
3. Result
This is the result of 2 models that applied the better solution in solving missing values (delete rows with missing values)

Model	RMSE training	RMSE cross validation	RMSE test
Linear regression	29290.3727306562	30645.3506137612	48476.6978515388
Ridge regression	29281.1985325225	30589.2340222668	48602.8228916531

Visualisation of the fit for linear regression



Visualisation of the fit for ridge regression



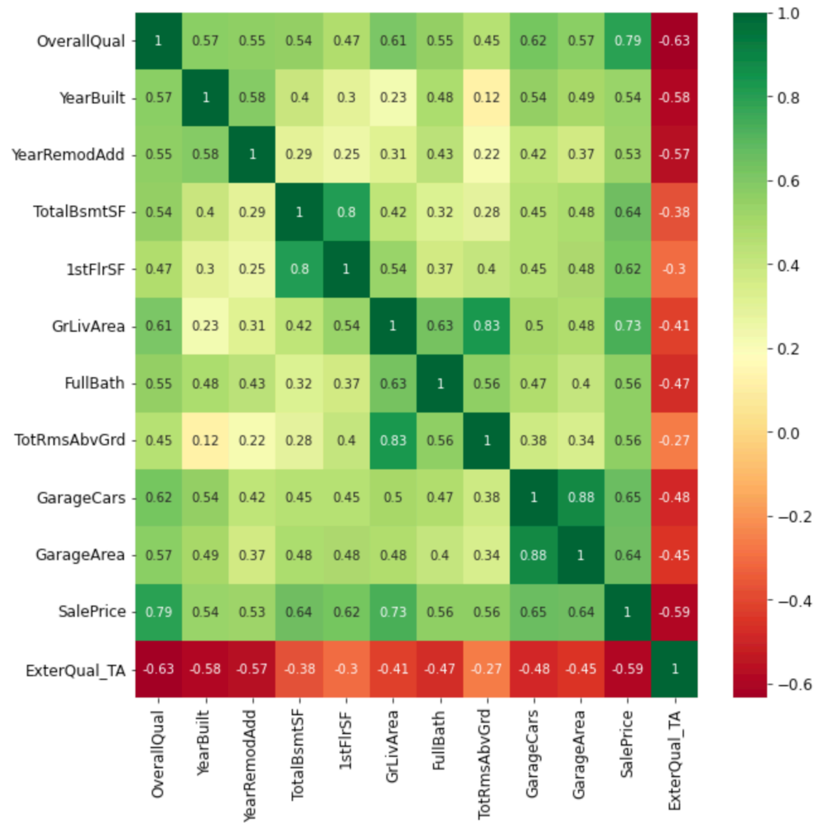
Overall, the results of both models are not too different. Ridge models have slightly smaller rmse results in the cross validation.

III. A description of the feature selection you applied and report the results.

1. According to correlation matrix and heatmap, select features that:

- have absolute value of correlation to target that are above 0.5.
- If 2 features are correlated to each other, remove one of features.
  - **GarageCars** and **GarageArea** are correlated with each other => remove GarageArea (keeping 'GarageCars' since its correlation with 'SalePrice' is higher).
  - **GrLivArea** and **TotRmsAbvGrd** are correlated with each other => remove TotRmsAbvGrd (keeping 'GrLivArea' since its correlation with 'SalePrice' is higher).
  - **TotalBsmtSF** and **1stFlrSF** are correlated with each other => remove 1stFlrSF (keeping 'TotalBsmtSF' since its correlation with 'SalePrice' is higher).

	Correlation to the target	Abs of correlation
SalePrice	1.000000	1.000000
OverallQual	0.792206	0.792206
GrLivArea	0.727463	0.727463
GarageCars	0.654479	0.654479
TotalBsmtSF	0.643649	0.643649
GarageArea	0.642608	0.642608
1stFlrSF	0.621144	0.621144
ExterQual_TA	-0.593505	0.593505
FullBath	0.561537	0.561537
TotRmsAbvGrd	0.559885	0.559885
YearBuilt	0.536462	0.536462
YearRemodAdd	0.525192	0.525192



Choose features:

1. SalePrice
2. OverallQual
3. GrLivArea
4. GarageCars
5. TotalBsmtSF
6. ExterQual_TA
7. FullBath
8. YearBuilt
9. YearRemodAdd

Result using linear regression model:

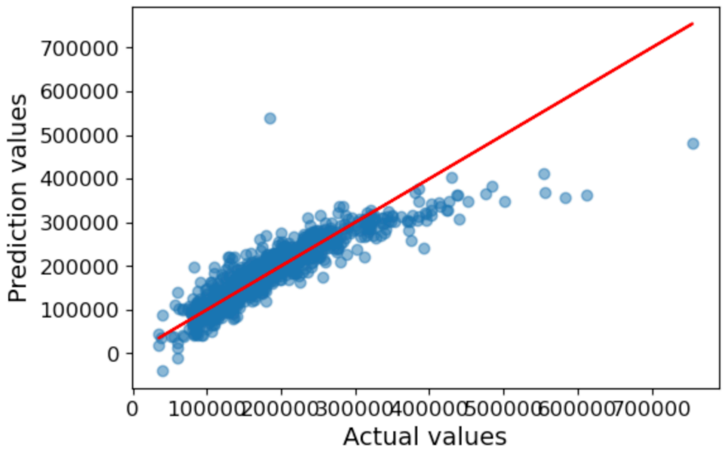
Features selection	RMSE training	RMSE cross validation	RMSE test
Full feature set	29290.3727306562	30645.3506137612	48476.6978515388
Subset of the features	34859.7791943402	34886.6430054037	49465.3712610298

This selection of subset of features has a larger rmse than using the full feature set

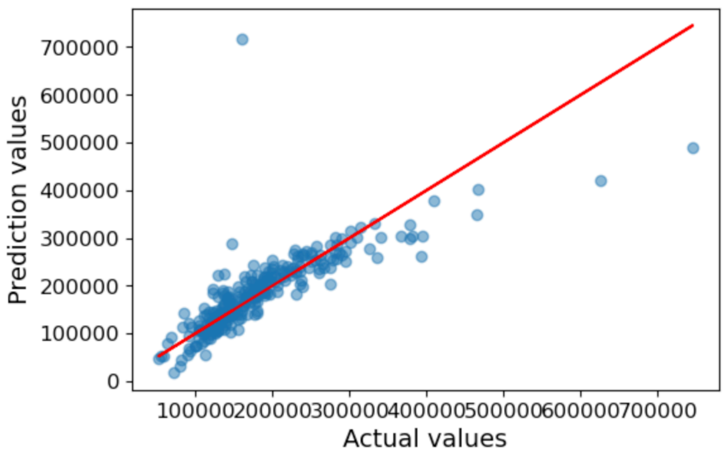
Visualization for a subset of the features

```
Training and testing with a subset of the features

Training and validation
Shape of training set: (1102, 9)
Shape of test set: (277, 9)
RMSE training: 34859.779194340204
Cross validation validation scores: [33375.74575948 30263.84837844 45151.98126947 34641.04986784
 31000.58975178]
Cross validation validation mean RMSE: 34886.64300540368
```



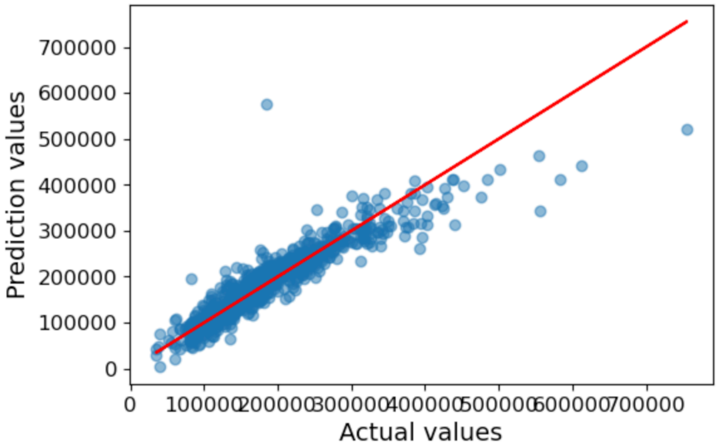

```
Evaluate performance
Shape of training set: (1102, 9)
Shape of test set: (277, 9)
RMSE test: 49465.37126102984
```



Visualization for full feature set

```
Training and testing with full feature set

Training and validation
Shape of training set: (1102, 59)
Shape of test set: (277, 59)
RMSE training: 29290.37273065618
Cross validation validation scores: [28305.48007486 24850.41530111 43689.23346094 28816.75613881
 27564.8680931 ]
Cross validation validation mean RMSE: 30645.35061376122
```




```
Evaluate performance
Shape of training set: (1102, 59)
Shape of test set: (277, 59)
RMSE test: 48476.697851538825
```