Assignment 4

Welcome to assignment #4! In this assignment we'll continue with feature engineering.

Youll see that a lot of good feature engineering depends on understanding your data. The exploratory data analysis skills we've been developing will be important here.

Note: You need to answer all questions in complete sentences. Failure to do so will result in a grit or zero marks.

```
!pip install matplotlib-venn -q gdc
import numpy as np
import pandas as pd
import matplotlib as mpl

%matplotlib inline
%matplotlib inline
sns.set_theme(style='ticks', font_scale = 1.5,
              palette='deep', color_codes=True)

# plot-size figure, figsize=(W, H), figsize=(1, 1)
plt.rcParams['figure.figsize'] = (8, 6)
plt.rcParams['axes.grid'] = True
```

```
# The following function randomly generates a set of values IN ANY ORDER (!!)
import random
import numpy as np
rng = np.random.seed(42)
# Lambda expression generate returned a set of values, IN ANY ORDER!
def lambda_generator(n, minVal=0, maxVal=10):
    return lambda x: (maxVal - minVal) * np.random.rand(int(n)) + minVal
[f(i) for i in range(5)] # [ 7.4940304 , 3.9173794 , 6.5404673 , 6.3948917 , 2.7259261]
```

```
# Load in some data
!wget -q -O data.csv -gc
"https://raw.githubusercontent.com/dsc-courses/dsc-data-files/master/data/kc_house_data.csv?raw=1"
```

```
inverted-block
```

Q1. Plot SalePrice and the 3 explanatory variables in a scatter matrix. (1 mark) You can see more of this here.

```
# Load data and look at explanatory variables that we plot below:
df = df[['SalePrice', 'YearBuilt', 'LotSqFootage', 'TotRmsAbvGrd']]
print(df.shape)
df.head(2)
```

```
# A scatter matrix:
pd.plotting.scatter_matrix(df, figsize=(7, 7), s=1);
```

Q2. Examine the histogram of SalePrice. Describe the distribution. Show how you would transform this to a normal distribution. (2 marks)

- The distribution of SalePrice shows right-skewed and the standardized normal distribution seems great. The SalePrice is measured in what are called standard deviations. The observations are within two standard deviations that is known as the Empirical Rule.
- SalePrice seem up to about 800,000
- Log transformed SalePrice (ln SalePrice) seem approximately fit to a gaw nor. 0.5
- The standardized SalePrice (std_SalePrice) seem fit around 2.5 to 2.5 standard deviations.

```
# Histogram of SalePrice:
sns.histplot(data=df, x='SalePrice', figsize=(7, 7));
```

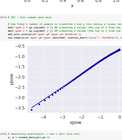
```
# A function if it is a normal distribution:
# Log transformation of SalePrice:
df['ln_SalePrice'] = np.log(df['SalePrice'])
df['std_SalePrice'] = (df['SalePrice'] - df['SalePrice'].mean()) / df['SalePrice'].std()

display(df[['SalePrice', 'ln_SalePrice', 'std_SalePrice']].sample(2, random_state=42).
        sort_values(by='std_SalePrice', ascending=False).
        round(2).
        transpose()
        .rename(columns={'SalePrice': 'Original', 'ln_SalePrice': 'Log', 'std_SalePrice': 'Standardized'}))
```

Q3. Make a separate scatterplot showing SalePrice and LotArea how would you describe the relationship? (1 mark)

You can read more about scatterplots in seaborn here.

```
# Another plot showing SalePrice and LotArea:
sns.scatplot(data=df, x='LotArea', y='SalePrice', aspect=1.5, figsize=(7, 7), title = 'The relationship between SalePrice and LotArea')
```



- The relationship between SalePrice and LotArea has a weak linear relationship (uncorrelated) and the relationship has been clustered in SalePrice between approximately 150,000 and 200,000.

Q4. Is it possible to quantify the **relationship** between these two variables? If so, explain what value you found and the meaning of the measurement. (1 mark)

```
# Quantify relationship:
# Correlation between SalePrice and LotArea:
print(quantify_relationship(df[['SalePrice', 'LotArea']]))
```

```
                 SalePrice  LotArea
SalePrice         1.000000 0.263843
LotArea           0.263843 1.000000
```

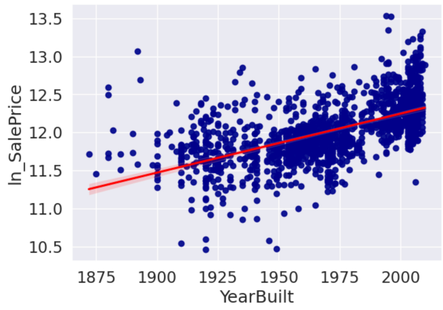
- Yes, the LotArea and SalePrice variables can quantify the relationship because the scatter plot shows a weak positive linear relationship and indicates a correlation value of about 0.26.

```
# Check if normal:
df['ln_LotArea'] = np.log(df['LotArea'])
df.hist(['LotArea', 'ln_LotArea'])
# Log transformation:
df['ln_SalePrice'] = np.log(df['SalePrice'])
df.hist(['SalePrice', 'ln_SalePrice'])
# Log transformation of SalePrice and LotArea:
df.hist(['ln_SalePrice', 'ln_LotArea'])
```

```
array([[<Axis_1>, <Axis_2>]]
       [<Axis_3>, <Axis_4>]]
       [<Axis_5>, <Axis_6>]]
       [<Axis_7>, <Axis_8>]]
       [<Axis_9>, <Axis_10>]]
       [<Axis_11>, <Axis_12>]]
       [<Axis_13>, <Axis_14>]]
       [<Axis_15>, <Axis_16>]]
       [<Axis_17>, <Axis_18>]]
       [<Axis_19>, <Axis_20>]]
       [<Axis_21>, <Axis_22>]]
       [<Axis_23>, <Axis_24>]]
       [<Axis_25>, <Axis_26>]]
       [<Axis_27>, <Axis_28>]]
       [<Axis_29>, <Axis_30>]]
       [<Axis_31>, <Axis_32>]]
       [<Axis_33>, <Axis_34>]]
       [<Axis_35>, <Axis_36>]]
       [<Axis_37>, <Axis_38>]]
       [<Axis_39>, <Axis_40>]]
       [<Axis_41>,
```

```
regression_data.plot.scatter(x='YearBuilt', y='ln_SalePrice');
sns.regplot(x='YearBuilt',y='ln_SalePrice',data=regression_data, scatter_kws={"color": "DarkBlue"}, line_kws={"color": "red"});
```

```
WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in case its length matches with *x* & *y*.  Please use the *c
               ln_SalePrice  YearBuilt
ln_SalePrice       1.00000    0.58657
YearBuilt          0.58657    1.00000
```

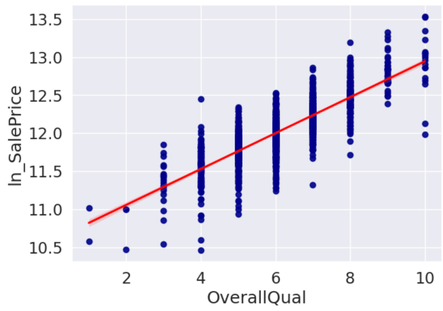


- The log-transformed value of SalePrice and YearBuilt shows a positive linear relationship in which the correlation coefficient r is about 0.587.

```
[221] coefficients2 = regression_data[["ln_SalePrice","OverallQual"]]
      print(coefficients2.corr())

      regression_data.plot.scatter(x='OverallQual', y='ln_SalePrice');
      sns.regplot(x='OverallQual',y='ln_SalePrice',data=regression_data, scatter_kws={"color": "DarkBlue"}, line_kws={"color": "red"});
```

```
WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in case its length matches with *x* & *y*.  Please use the *c
               ln_SalePrice  OverallQual
ln_SalePrice       1.000000     0.817184
OverallQual        0.817184     1.000000
```



- The log-transformed value of SalePrice and OverallQual shows a strong linear relationship in which the correlation coefficient r is about 0.812.

Q7(b) Create a regression model using untransformed (i.e, not log transformed) SalePrice as the dependent variable and the same explanatory variables from part (a). Compare the R^2 value from the model you created with transformed SalePrice and this model with an untransformed SalePrice.

What does the comparison suggest to you? (1 mark)

Note: There is no need to explain the output from this model.

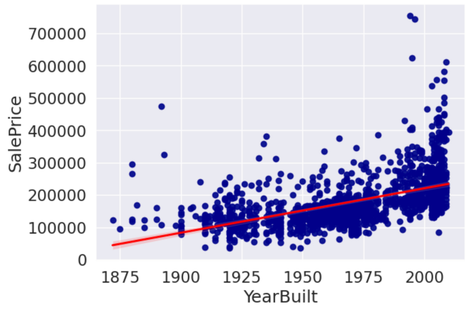
```
[222] #Create a regression model using untransformed (i.e, not log transformed) SalePrice

      regression_data2 = df[["SalePrice", "YearBuilt", "OverallQual", "BldgType"]]

      coefficients3 = regression_data2[["SalePrice","YearBuilt"]]
      print(coefficients3.corr())

      regression_data2.plot.scatter(x='YearBuilt', y='SalePrice');
      sns.regplot(x='YearBuilt',y='SalePrice',data=regression_data2, scatter_kws={"color": "DarkBlue"}, line_kws={"color": "red"});
```

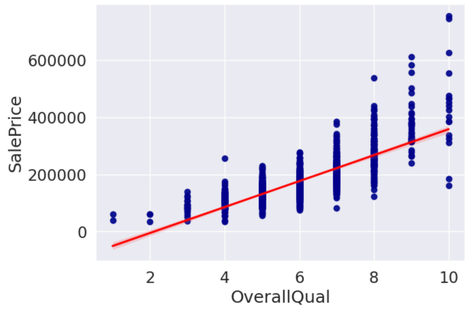
```
WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in case its length matches with *x* & *y*.  Please use the *c
           SalePrice  YearBuilt
SalePrice   1.000000   0.522897
YearBuilt   0.522897   1.000000
```




```
coefficients4 = regression_data2[["SalePrice","OverallQual"]]
print(coefficients4.corr())

regression_data2.plot.scatter(x='OverallQual', y='SalePrice');
sns.regplot(x='OverallQual',y='SalePrice',data=regression_data2, scatter_kws={"color": "DarkBlue"}, line_kws={"color": "red"});
```

```
WARNING:matplotlib.axes._axes:*c* argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in case its length matches with *x* & *y*.  Please use the *c
             SalePrice  OverallQual
SalePrice     1.000000     0.790982
OverallQual   0.790982     1.000000
```



- Compare the R^2 value from the model that created with transformed SalePrice and this model with an untransformed SalePrice, it shows that the low strength and direction of the linear association between two variables also indicates a low correlation coefficient r than the R^2 value from the model that created with transformed SalePrice.

Colab paid products - Cancel contracts here
0s completed at 3:00 PM