DATA SCIENCE PROJECT

Analysis of Ames Housing Dataset + Predictive Modelling

by Kam Wing Sze, Cathy

NOV 10, 2022

Background

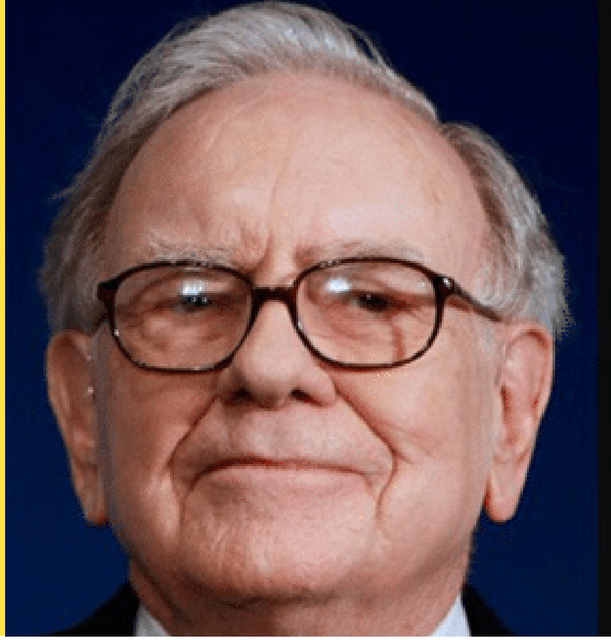
THE AMES HOUSING DATASET WAS INTRODUCED BY PROFESSOR DEAN DE COCK IN 2011 AS AN ALTERNATIVE TO THE BOSTON HOUSING DATASET

It contains 2,919 observations of housing sales in Ames, Iowa between 2006 and 2010. There are 23 nominal, 23 ordinal, 14 discrete, and 20 continuous features describing each house's size, quality, area, age, and other miscellaneous attributes.

Source: Thomas Deegan, Brandon Deniz, Hayley Caddes and John McGlynn(2018)

https://nycdatascience.com/blog/student-works/machine-learning/machine-learning-project-ames-housing-dataset/

Buffett on housing:



House prices just soared beyond - beyond reason in many places and they got financed in silly ways, and people lied about loans, all kinds of accesses entered into it. But that is what - that is the single biggest cause of why we're here.

— Warren Buffett —

Why is housing prediction significant?

Housing prices are an important **reflection of the economy**, and housing price ranges are of great interest for both buyers and sellers, including the housing agencies like Zillow and Trulia.

ACCORDING TO IJACSA...

House price prediction can help the developer **determine the selling price of a house** and the customer to **arrange the right time to purchase a house**. There are three factors that influence the price of a house which include **physical conditions**, **concept** and **location**.

Source: (IJACSA) International Journal of Advanced Computer Science and Applications Vol. 8, No. 10, 2017

Problem Statement

As a potential house buyer myself,
I want to find out ...

THIS PROJECT AIMS TO:

1. Predict the sales price for houses in Ames
2. Minimize the difference between predicted and actual sales price (RMSE/MSE)

WHERE AND WHAT IS THE DATASET?

AMES HOUSING DATASET

https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

DATA CREDIT:

Dean De Cock

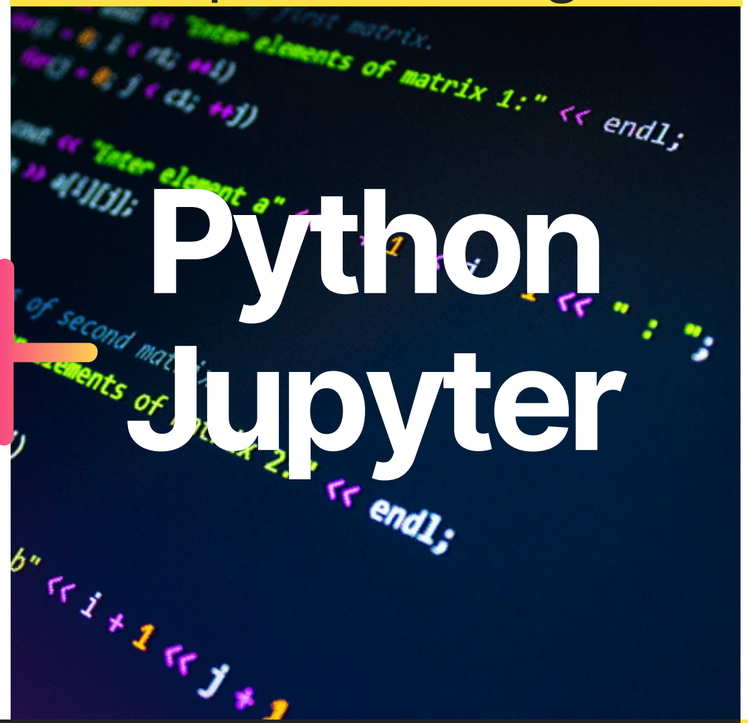
The dataset was compiled by Dean De Cock for use in data science education. It is served as a modernized and expanded version of the famous Boston Housing dataset.

What tools do I use to process and present the data?

1. Dataframe + Organization

2. Visualization

3. Codes running and open sourcing



INTRODUCTION -

DESCRIPTION OF THE DATASET

1 Target Column:

1. **SalesPrice**

the property's sale price in dollars.

LET'S TAKE A QUICK GLIMPSE OF OUR DATASET...

The dataset contains several parameters which are considered important for the housing price prediction. The feature variables and target column are listed at the sides:

79 Feature Variables:

1. **OverallQual**
2. **Gr Liv Area**
3. **Garage Area**
4. **Garage Cars**
5. **Total Bsmt SF**
6. **1st Flr SF**
7. **Year Built**
8. **Year Remod/Add**
9. **'Full Bath**
10. **Garage Yr Blt**

(Skipped the rest of the columns)

There is a total of...

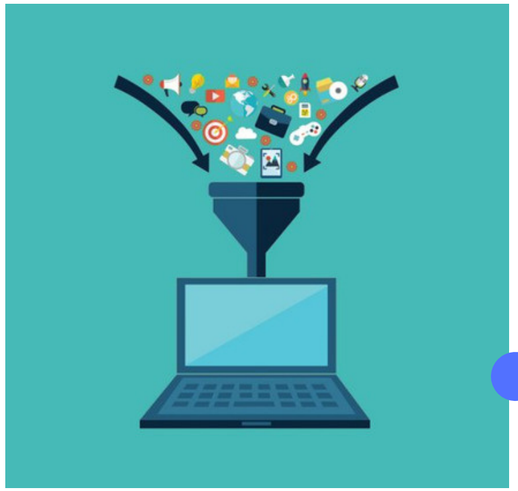
2151 ROWS OF DATA

alongside with 79 feature columns
and 1 target column

```
In [2]: # Read the data
        train = pd.read_csv('train.csv', index_col='Id')
        test = pd.read_csv('test.csv', index_col='Id')

        # Remove rows with missing target
        train = train.dropna(axis=0, subset=['SalePrice'])

        # Separate target from predictors
        X = train.drop(['SalePrice'], axis=1)
        y = train.SalePrice

In [3]: # X, y info
```

```
In [124]: train.shape

Out[124]: (2051, 80)
```

3-Step Flow of My Analysis

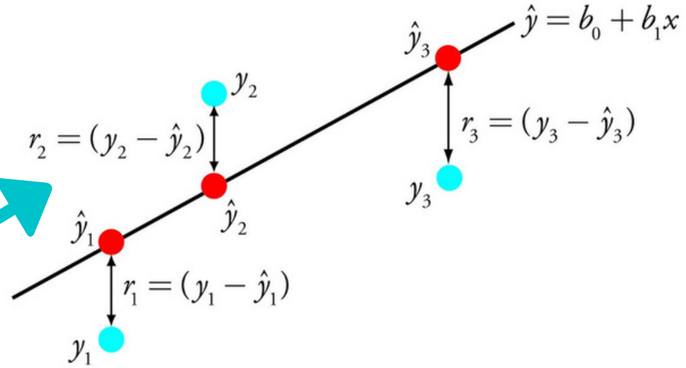


Data Mining + Preprocessing

- To identify the presence of null values in each column
- **Remove irrelevant columns** which could possibly affect the accuracy of the prediction

Data Exploration + Visualization + Feature Engineering

- To draw **insights** from the dataset
- To find out the **correlation** between the housing price and different variables (e.g. Utilities / Land Contour)



Predictive Modelling + Evaluation

- To use PolynomialFeature, standard scaler and **Ridge regression** as predictive models to predict the housing price
- To minimize the **Root Mean Square Error(RMSE)**

01

Let's start with

Data Mining + Preprocessing

Data Preprocessing - Cleaning

There are a total of 9822 missing values in the training dataset and 4171 missing values in the testing dataset.

Remove the columns with more than half missing values

In [10]:
```
# Making function so that we can reuse it in later stages as well
def show_null_values(X, test):
    # Missing values in each column of Training and Testing data
    null_values_train = X.isnull().sum()
    null_values_test = test.isnull().sum()

    # Making DataFrame for combining training and testing missing values
    null_values = pd.DataFrame(null_values_train)
    null_values['Test Data'] = null_values_test.values
    null_values.rename(columns = {0:'Train Data'}, inplace = True)

    # Showing only columns having missing values and sorting them
    null_values = null_values.loc[(null_values['Train Data']!=0) | (null_values['Test Data']!=0)]
    null_values = null_values.sort_values(by=['Train Data','Test Data'],ascending=False)

    print("Total missing values:",null_values.sum(),sep='\n')

    return null_values
```

In [11]:
```
show_null_values(X, test)
```

```
Total missing values:
Train Data    9822
Test Data     4171
dtype: int64
```

Data Preprocessing - Cleaning

Delete columns that have missing values more than half number of rows.

Alley, Pool QC, Fence and Misc Feature have the most missing values.

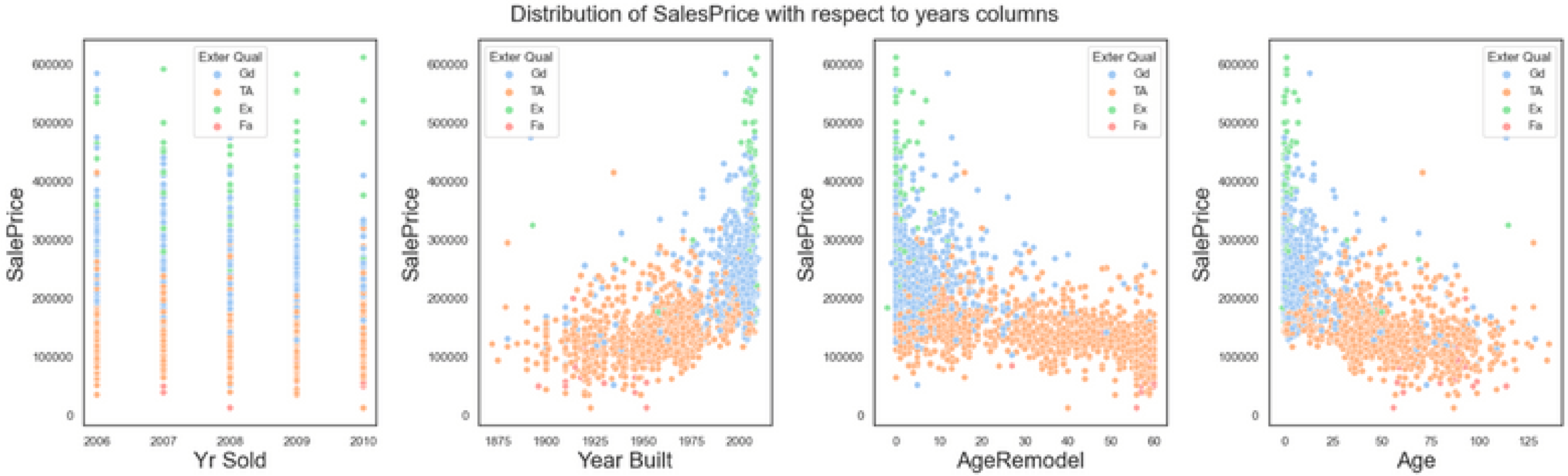
```
In [12]: # Columns with missing values in more than half number of rows
         null_cols = [col for col in X.columns if X[col].isnull().sum() > len(X)/2]
         null_cols

Out[12]: ['Alley', 'Pool QC', 'Fence', 'Misc Feature']

In [13]: X.drop(null_cols,axis=1,inplace=True)
         test.drop(null_cols,axis=1,inplace=True)
```

Let's move on to...

Data Exploration + Visualization

02

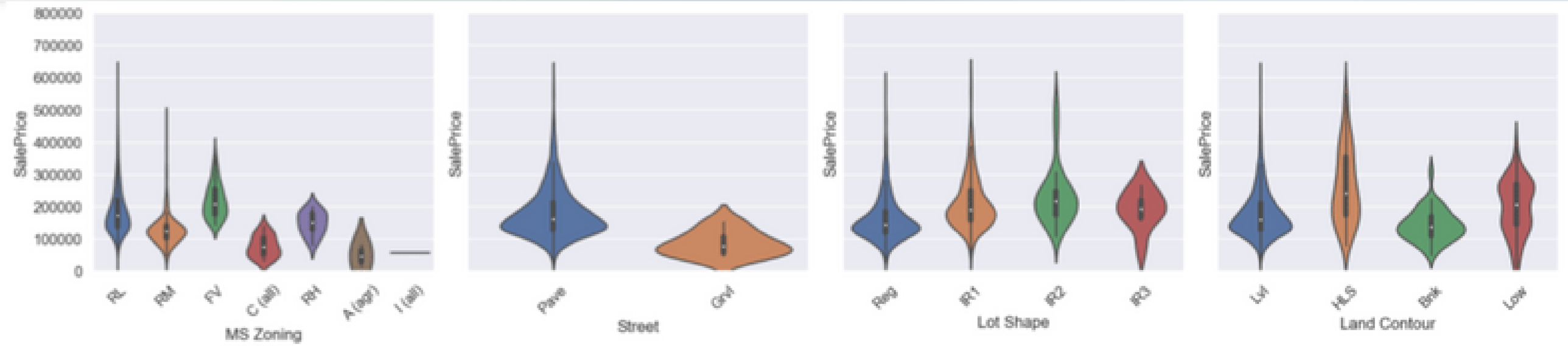
Data Visualization - SalesPrice with Years



Data mining + Preprocessing → **Data Exploration + Visualization** → Predictive Modelling + Evaluation

Data Visualization - MS Zoning, Street, Lot Shape, Land Contour

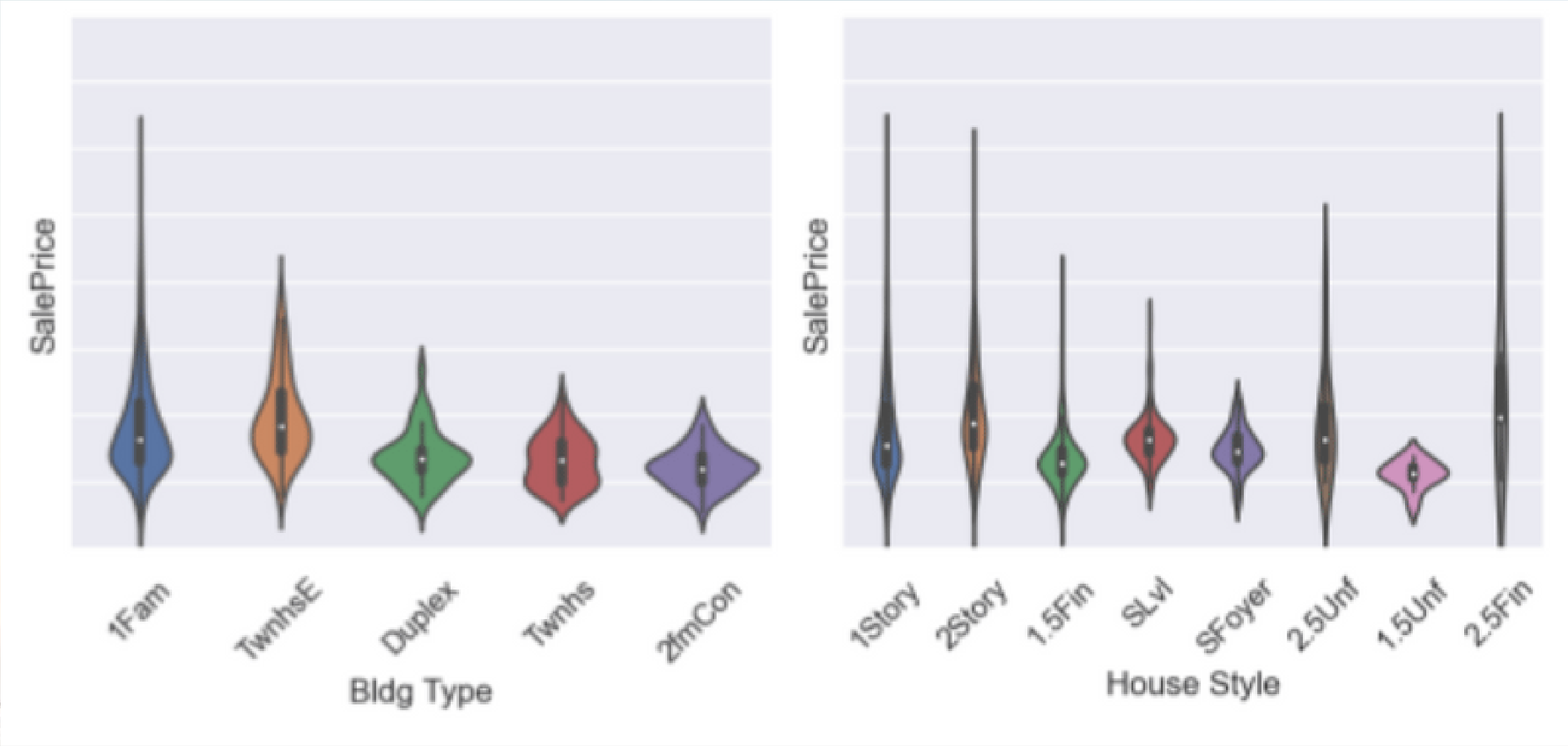
Residential Area with Low Density, Paved street, slightly irregular lot shape and hillside land contour generally resulted in higher price range.



Data mining + Preprocessing → **Data Exploration + Visualization** → Predictive Modelling + Evaluation

Data Visualization - Building Type, House Style

Single-family Detached and Townhouse End Unit tend to end with better bargains, while 1 story and 2 story housing styles appeared to be more lucrative.

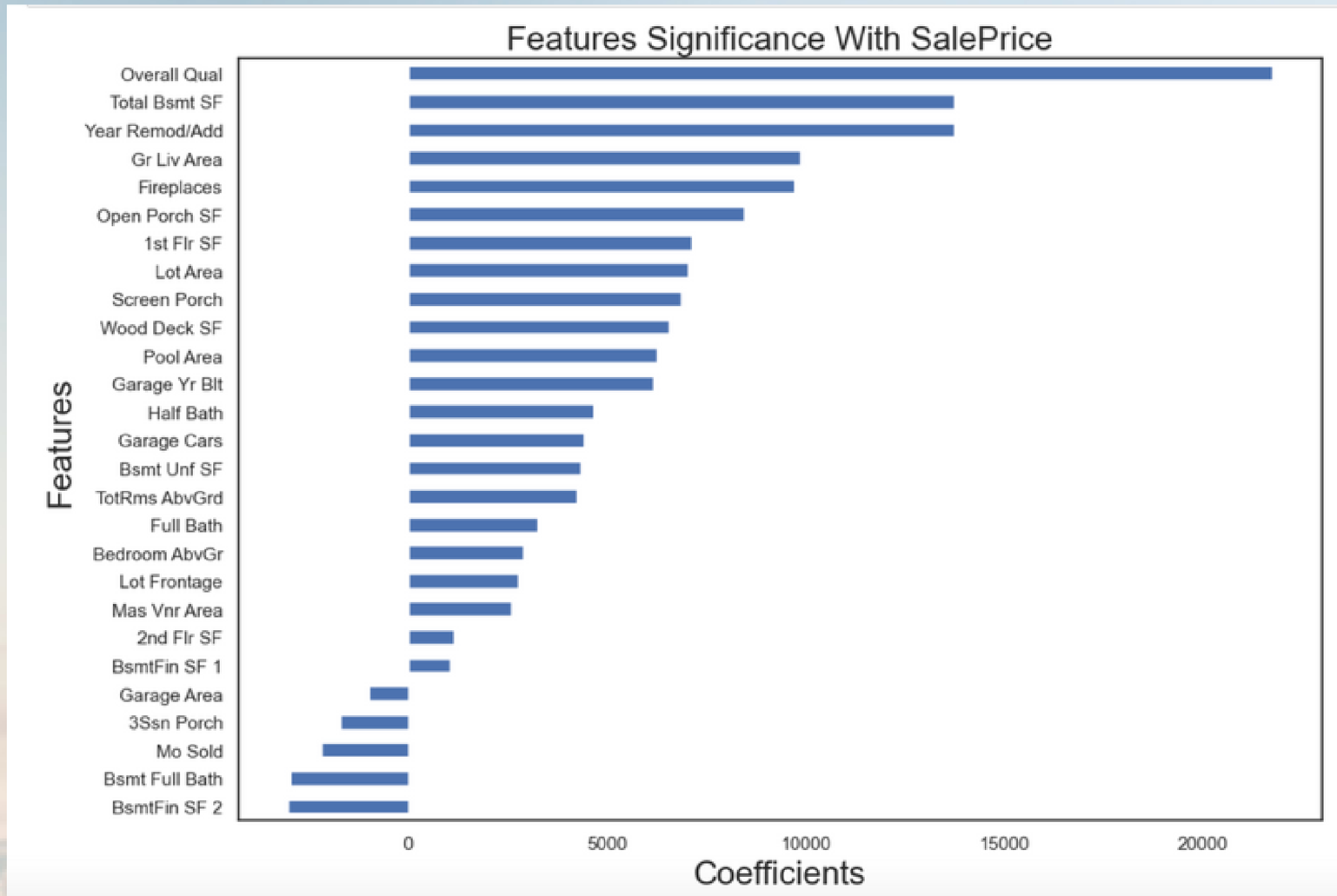


Data mining + Preprocessing → **Data Exploration + Visualization** → Predictive Modelling + Evaluation

Data Visualization - Feature Significance

Among all the different variables, interestingly, there are 5 variables that are found to be closely related to our target column:

1. **Overall Qual**
2. **Total Bsmt SF**
3. **Year Remod/Add**
4. **Gr Liv Area**
5. **Fireplaces**



Data Visualization- SalesPrice Distribution

The SalesPrice is shown in bar chart, box and line plot respectively.

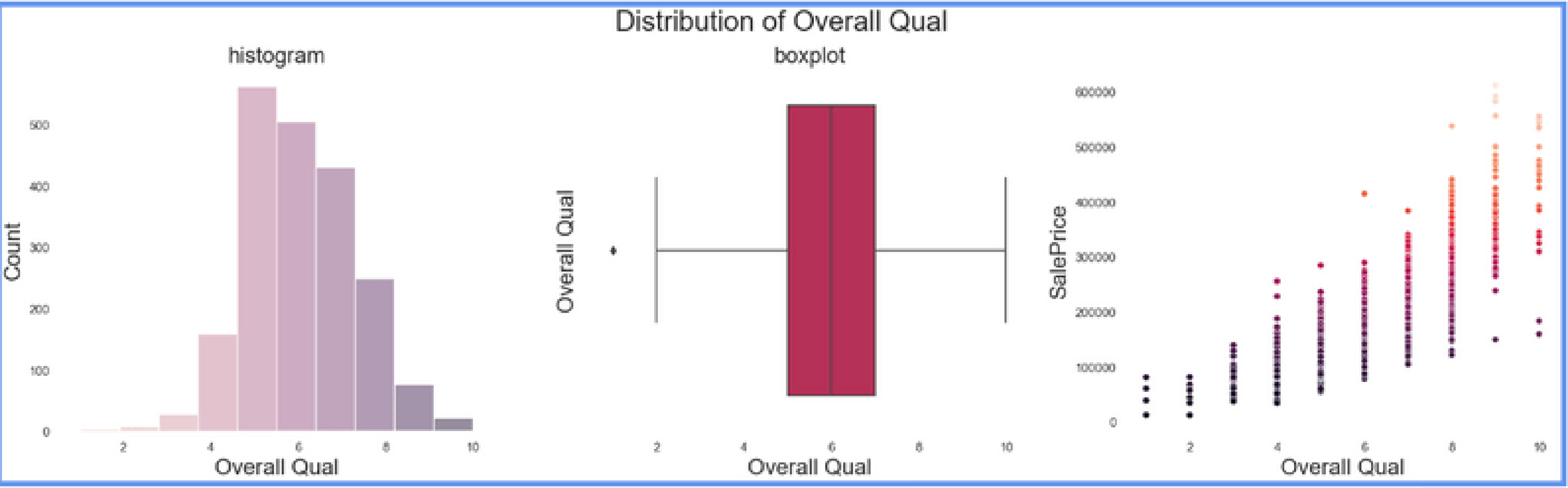
The mean lies around 150k - 200k range, while there are also some outliers ranging beyond 350k

Bivariate Analysis: SalesPrice with Overall Qual

The mean of Overall Qual is 6 out of 10.

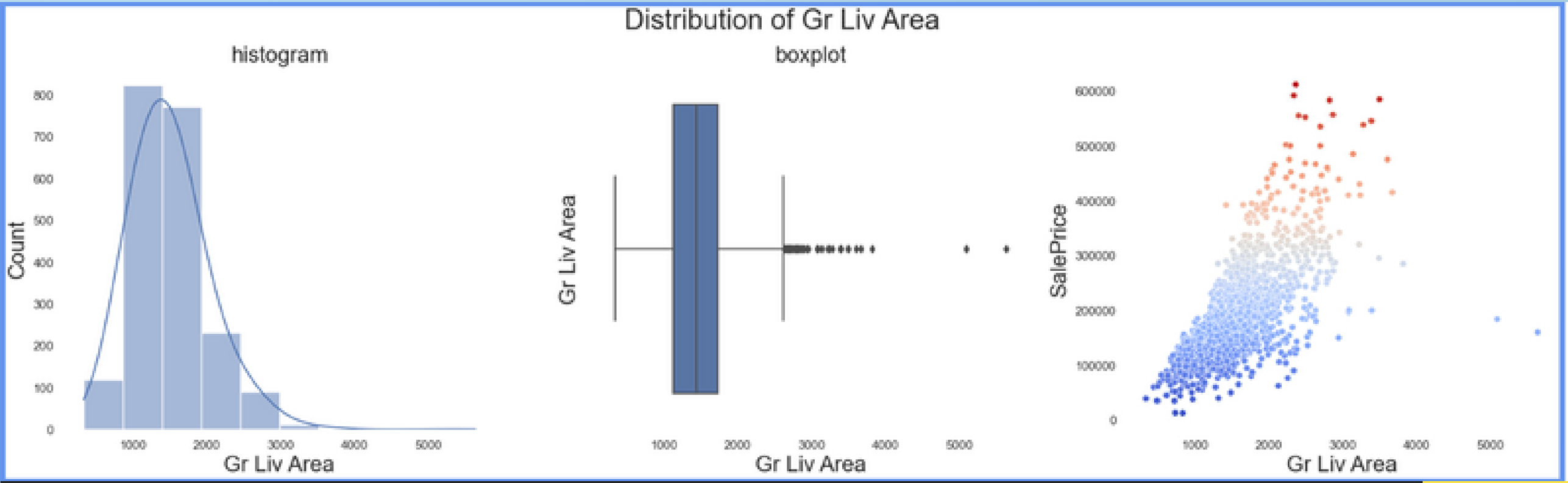
There is a strong positive correlation between SalesPrice and Overall Qual.

The **higher the overall quality** is, the **higher the salesPrice** will be.

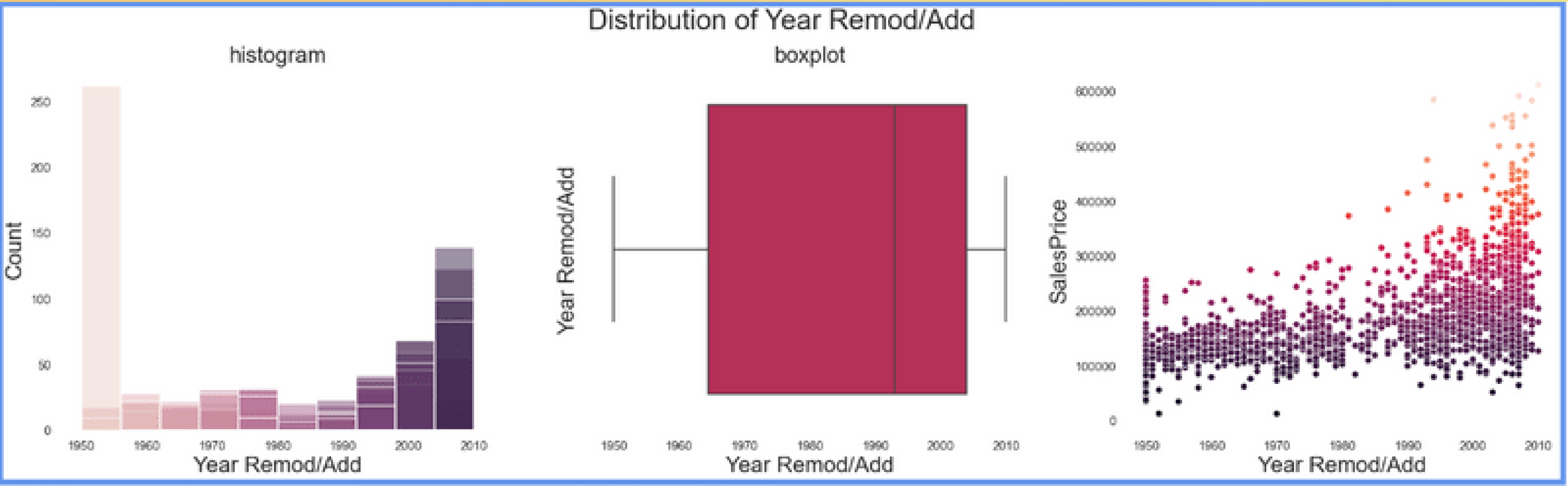


Bivariate Analysis : SalesPrice with Gr Liv Area

The mean of the Gr Liv Area is around 1500 square feet.
Same as Overall Qual, there is a **positive correlation** found between SalesPrice and Gr Liv Area.



Bivariate Analysis : SalesPrice with Year Remod/Add

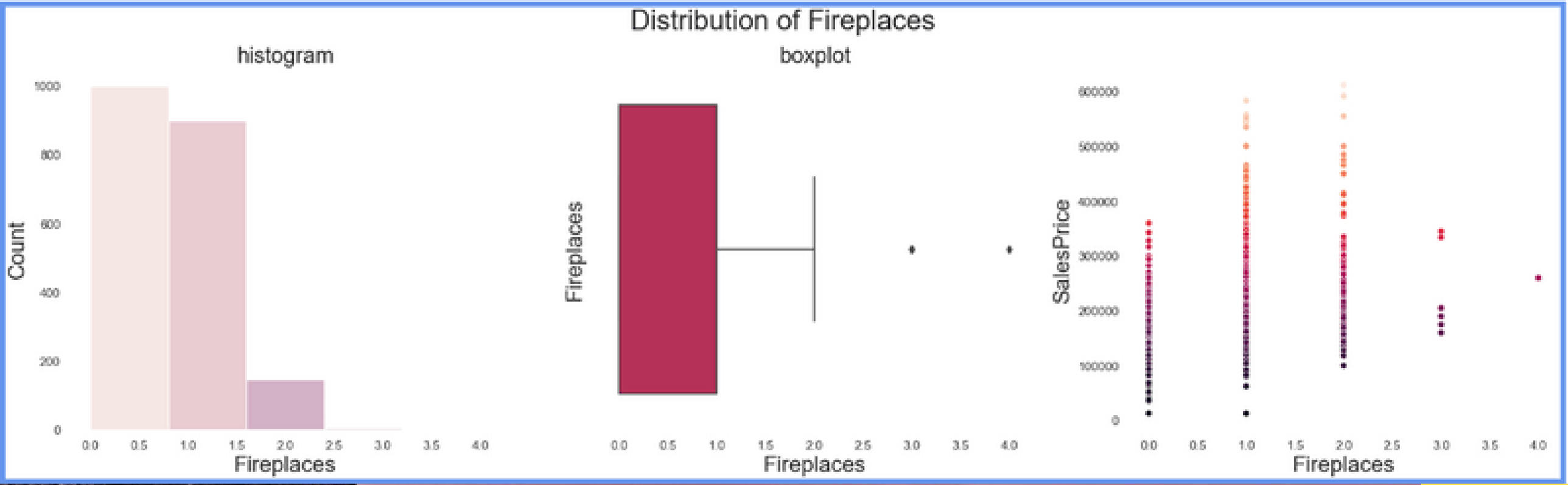


Data mining + Preprocessing → **Data Exploration + Visualization** → Predictive Modelling + Evaluation

Bivariate Analysis : SalesPrice with Fireplaces

A majority of houses don't have fireplaces. The sales prices reach to its maximum when the number of fireplaces reaches to 2.

Yet, there is a slight positive relationship with the Sales price



Data mining + Preprocessing

Data Exploration + Visualization

Predictive Modelling + Evaluation

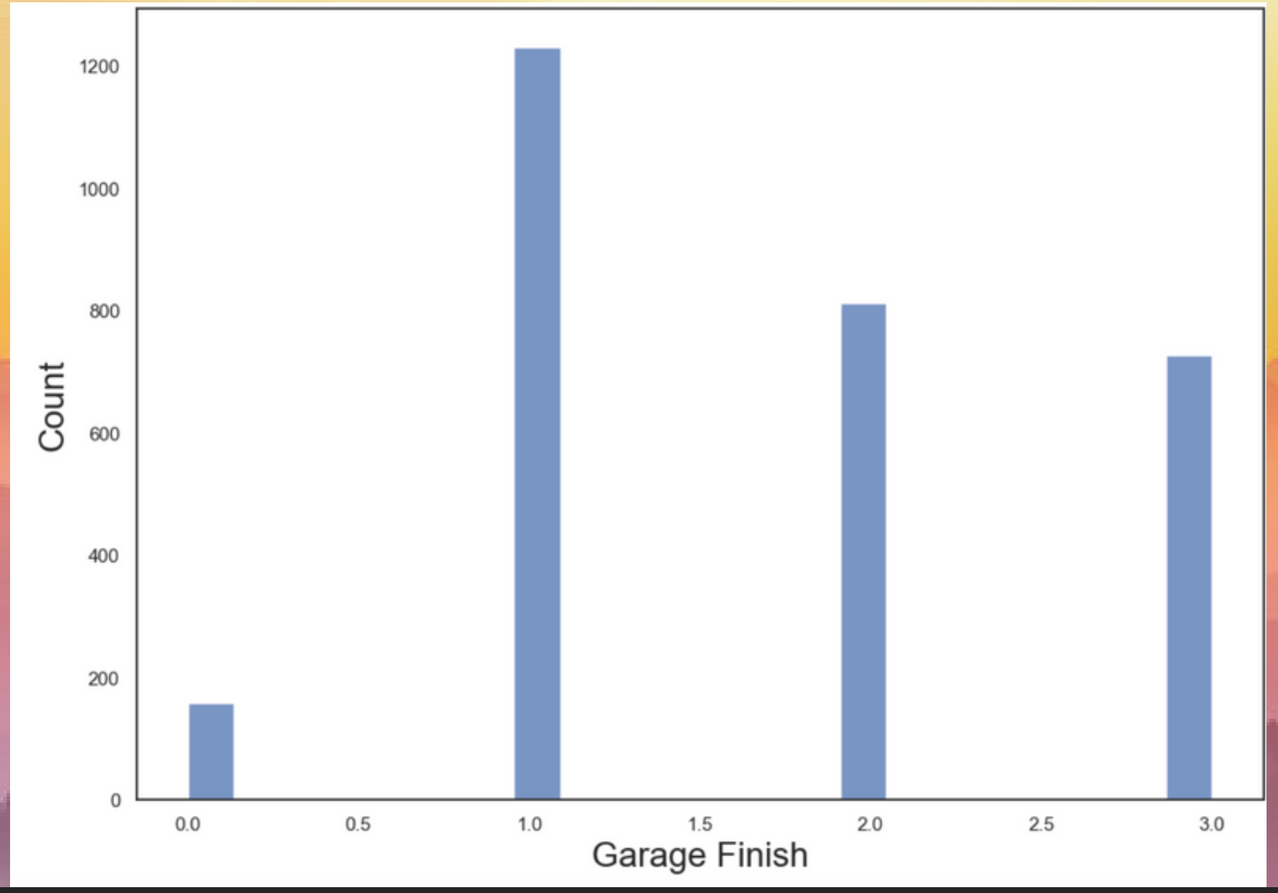
Feature Engineering - Quantify the unique values

```
{'Exter Qual': ['Gd', 'TA', 'Ex', 'Fa'],
 'Exter Cond': ['TA', 'Gd', 'Fa', 'Ex', 'Po'],
 'Bsmt Qual': ['TA', 'Gd', 'Fa', 'NA', 'Ex', 'Po'],
 'Bsmt Cond': ['TA', 'Gd', 'NA', 'Fa', 'Po', 'Ex'],
 'BsmtFin Type 1': ['GLQ', 'Unf', 'ALQ', 'Rec', 'NA', 'BLQ', 'LwQ'],
 'BsmtFin Type 2': ['Unf', 'Rec', 'NA', 'BLQ', 'GLQ', 'LwQ', 'ALQ'],
 'Heating QC': ['Ex', 'TA', 'Gd', 'Fa', 'Po'],
 'Kitchen Qual': ['Gd', 'TA', 'Fa', 'Ex', 'Po'],
 'Fireplace Qu': ['NA', 'TA', 'Gd', 'Po', 'Ex', 'Fa'],
 'Garage Finish': ['RFn', 'Unf', 'Fin', 'NA'],
 'Garage Qual': ['TA', 'Fa', 'NA', 'Gd', 'Ex', 'Po'],
 'Garage Cond': ['TA', 'Fa', 'NA', 'Po', 'Gd', 'Ex']}
```

```
# 1] Columns with similar ordered categories [Poor<Fair<Typical/Average<Good<Excellent]
ordinal_cols1 = [i for i in object_cols if ('QC' in i) or ('Qu' in i) or ('Cond' in i) and ('Condition' not in i)]
df.loc[:,ordinal_cols1] = df.loc[:,ordinal_cols1].replace(['NA', 'Po', 'Fa', 'TA', 'Gd', 'Ex'], [0,1,2,3,4,5])

# 2] Columns with similar ordered categories [No Garage/Basement<Unfinished<Rough Finished<Finished,etc]
ordinal_cols2 = ['BsmtFin Type 1', 'BsmtFin Type 2']
df.loc[:,ordinal_cols2] = df.loc[:,ordinal_cols2].replace(['NA', 'Unf', 'LwQ', 'Rec', 'BLQ', 'ALQ', 'GLQ'], [0,1,2,3,4,

# 3] Column with ordered categories [No Basement<No Exposure<Mimimum Exposure<Average Exposure<Good Exposure]
ordinal_cols3 = ['Bsmt Exposure']
df.loc[:,ordinal_cols3] = df.loc[:,ordinal_cols3].fillna('NA')
df.loc[:,ordinal_cols3] = df.loc[:,ordinal_cols3].replace(['NA', 'No', 'Mn', 'Av', 'Gd'], [0,1,2,3,4])

# 4] Column with ordered categories [Regular<Slightly irregular<Moderately Irregular<Irregular]
ordinal_cols4 = ['Lot Shape']
df.loc[:,ordinal_cols4] = df.loc[:,ordinal_cols4].replace(['Reg', 'IR1', 'IR2', 'IR3'], [0,1,2,3])
```

Feature Engineering - Example (Garage Finish)

A majority of houses have unfinished Garages, while finished garages are the second most common category.
#'Garage Finish': ['RFn' - 0 , 'Unf' - 1.0 , 'Fin' - 2.0, 'NA' - 3.0] -> Quantifying them for standard scaling later.

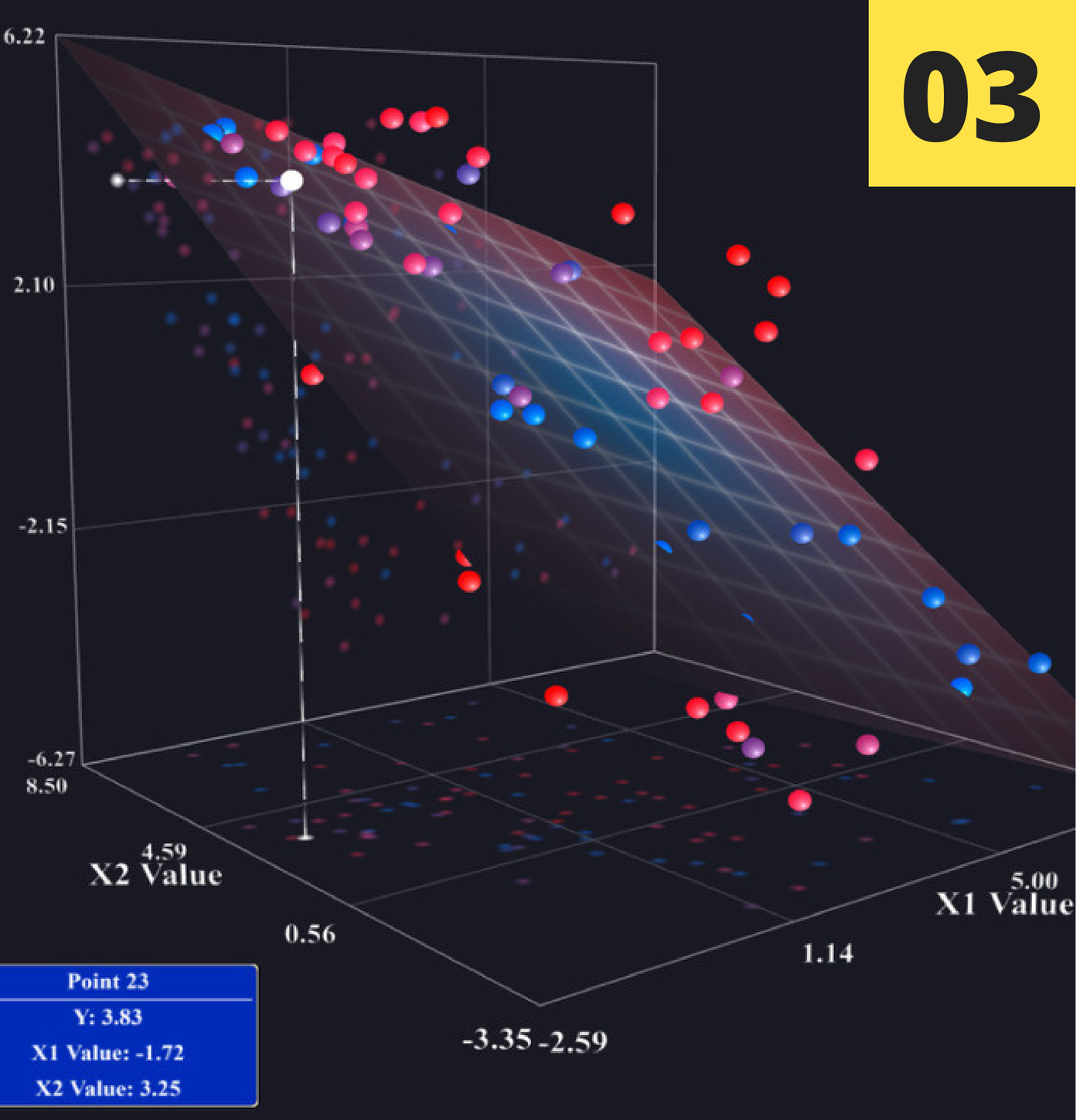


Feature Engineering - Filling in the Missing Values

4) Imputing Numerical Columns

```
In [47]: my_imputer = SimpleImputer(missing_values = np.nan, strategy ='constant', fill_value=0)

         # Fitting the data to the imputer object
         imputed_X = pd.DataFrame(my_imputer.fit_transform(X))
         imputed_X_test = pd.DataFrame(my_imputer.transform(test))
```

```
In [48]: show_null_values(X, X_test)

         Total missing values:
         Train Data     0.0
         Test Data      0.0
         dtype: float64
```



03

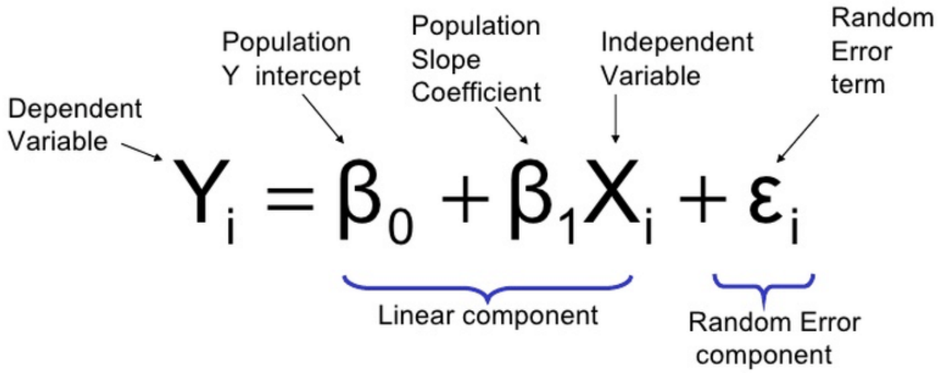
Here comes the

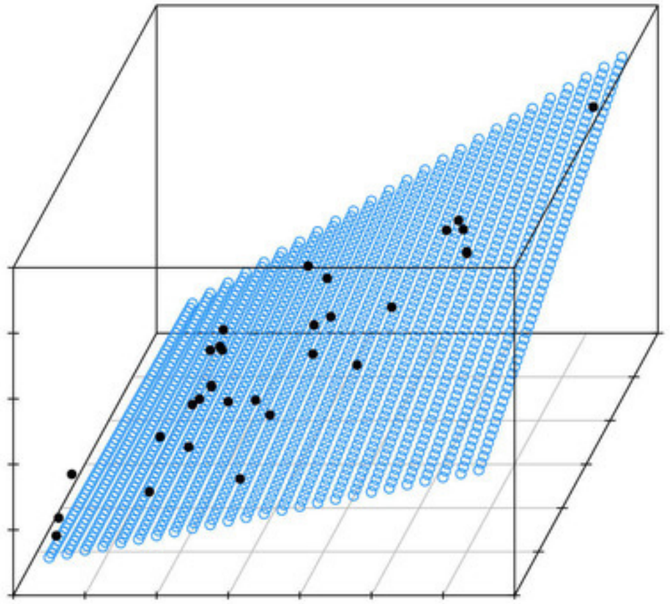
Predictive Modeling + Evaluation

Why Regression?
Will Linear Regressor be suffice?

After analyzing through the visualization, there is a linear relationship identified across different variables with the target column, which is an ideal indicator for linear regression.

Since the dataset is fairly large and numerous variables are present, a pipeline is used to apply StandardScaler, PolynomialFeatures, RFE(estimator=Ridge() and Ridge(max_iter=10000)) respectively altogether to predict the target column effectively.





Data mining + Preprocessing → Data Exploration + Visualization → **Predictive Modelling + Evaluation**

How did a mere Linear regression perform?

Linear Regression only

```
In [111]: from sklearn.linear_model import LinearRegression
          from sklearn.model_selection import train_test_split

          lr = LinearRegression()
          lr.fit(X_train, y_train)

Out[111]: LinearRegression()

In [112]: # Train score
          lr.score(X_train, y_train)

Out[112]: 0.8022489886774614

In [113]: # Test score
          lr.score(X_test, y_test)

Out[113]: 0.8517831035347828
```

Although linear regression works just fine, the score is not the most ideal. Only a 80%-85% R2 score shown in the training and testing data.

Data mining + Preprocessing → Data Exploration + Visualization → **Predictive Modelling + Evaluation**

1. Standard Scaler (Z-score)

The data obtained contains features of various dimensions and scales altogether. Different scales of the data features affect the modeling of a dataset adversely. Thus, it is necessary to Scale the data prior to modeling.

It helps standardizes a feature by subtracting the mean and then scaling to unit variance. Unit variance means dividing all the values by the standard deviation.

$$z = \frac{x - \mu}{\sigma}$$

2. Polynomial Feature

To overcome under-fitting, we need to increase the complexity of the model.

While t*his can still considered to be* ***linear model*** *as the coefficients/weights associated with the features are still linear, the curve that we are fitting is* ***quadratic*** *in nature, which can cater for multiple variables that we have.*

$$Y = \theta_0 + \theta_1 x$$

$$Y = \theta_0 + \theta_1 x + \theta_2 x^2$$

3. Recursive Feature Elimination (RFE)

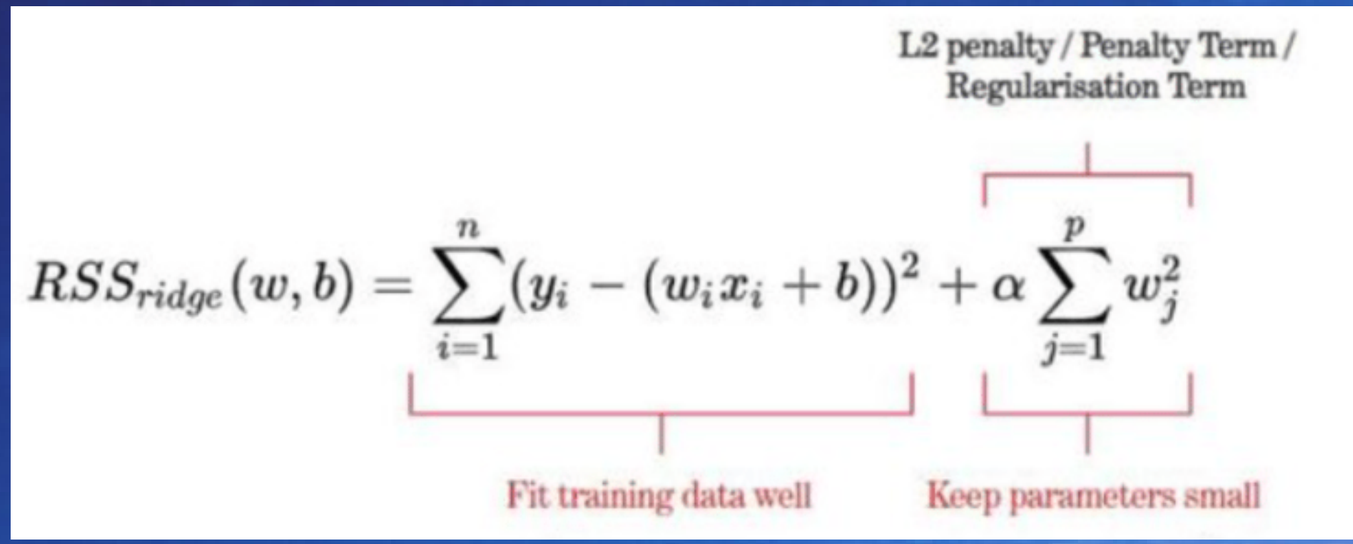
RFE is popular because it is easy to configure and use and because it is effective at selecting those features (columns) in a training dataset that are more or most relevant in predicting the target variable.

RFE works by searching for a subset of features by starting with all features in the training dataset and successfully removing features until the desired number remains.

4. Ridge Regression

Ridge Regression is a technique for analyzing multiple regression data that suffer from multicollinearity. When multicollinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far from the true value.

Therefore, this model is applied ***to reduce the standard error by adding some bias in the estimates of the regression.***



Predictive Modeling - Feature Scaling

```
features = X_y[['Overall Qual', 'Gr Liv Area', 'Garage Area',
       'Garage Cars', 'Total Bsmt SF', '1st Flr SF', 'Year Remod/Add', 'Full Bath', 'Garage Yr Blt', 'Mas Vnr Area',
       'TotRms AbvGrd', 'Fireplaces', 'BsmtFin SF 1', 'Lot Frontage',
       'Open Porch SF', 'Wood Deck SF', 'Lot Area', 'Bsmt Full Bath',
       'Half Bath', '2nd Flr SF', 'Bsmt Unf SF', 'Bedroom AbvGr',
       'Screen Porch', '3Ssn Porch', 'Mo Sold', 'Pool Area', 'BsmtFin SF 2']]

X = features
y = X_y['SalesPrice']
```

Select the feature columns to be scaled.

Import the libraries that we are going to be using.

6.3) Pipeline (Standard Scaler -> Lasso + Ridge -> Linear Regression)

```
from sklearn.preprocessing import PolynomialFeatures
from sklearn.feature_selection import RFE
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
```

```
from sklearn.linear_model import Lasso, LassoCV
from sklearn.linear_model import Ridge, RidgeCV
```

Data mining + Preprocessing → Data Exploration + Visualization → **Predictive Modelling + Evaluation**

Predictive Modeling - Spliting into training and testing data

```
X_train, X_test, y_train, y_test  = train_test_split(X, y, test_size=0.33, random_state=42)
```

Data mining + Preprocessing → Data Exploration + Visualization → **Predictive Modelling + Evaluation**

Predictive Modeling - Employ Pipeline via GridSearch

Create a pipeline to apply PolynomialFeatures, StandardScaler, RFE and Ridge.

```
In [510]: ridge_grid.fit(X_train_scaled, y_train)

Out[510]: GridSearchCV(estimator=Pipeline(steps=[('standardscaler', StandardScaler()),
                                                 ('polynomialfeatures',
                                                  PolynomialFeatures()),
                                                 ('rfe', RFE(estimator=Ridge())),
                                                 ('ridge', Ridge(max_iter=10000))]),
                       param_grid={'ridge__alpha': [0.001, 0.01, 0.1, 1.0, 10.0]})
```

Predictive Modeling - Score Evaluation

```
In [542]: ridge_grid.score(X_train_scaled, y_train)

Out[542]: 0.9431567275757873

In [543]: ridge_grid.score(X_test_scaled, y_test)

Out[543]: 0.8618800243644766
```

Predictive Modeling - Mean Squared Error Evaluation

```
]: # Training MSE
   print('train:', mean_squared_error(y_train, ridge_grid.predict(X_train_scaled), squared=False))

   # Testing RMSE
   print('test:', mean_squared_error(y_test, ridge_grid.predict(X_test_scaled), squared=False))
```

```
train: 19142.54696396386
test: 28631.799465502743
```

Predictive Modeling - Output

Outputting the predicted Y column (Sales Price) out

```
y_preds2 = ridge_grid.predict(X_test_scaled)
```

```
output2 = pd.DataFrame({'Id': X_test.index,
                        'SalePrice': y_preds2.round()})
output2.to_csv('submission_byPipeline_Scaler_Lasso_Ridge.csv', index=False)
```

```
output2.head()
```

	Id	SalePrice
0	2782	126231.0
1	2569	214346.0
2	1062	269258.0
3	287	114553.0
4	2809	196396.0

Conclusion + Recommendation

If there are customers who have tight budget in hand, they will probably end up choosing houses with the following features:

a. Two family condo
b. 1.5 unit
d. Residential area with high density
e. Gravel street
f. Low overall quality
g. 0 garage

For average buyers, they should have a mortgage and deposit that can afford a house of **150k - 200k** range. While for those who are more wealthy, they can expect paying more than **350k** to buy luxury houses.

The End

Q & A