**What is the problem?**

Let’s get started to make a prediction on our first machine learning algorithm with a rich dataset on housing prices from Ames, Iowa. Each row in the dataset describes the properties of a single house as well as the amount it was sold for. In this concept, we'll build models that predict the final sale price of a house based on its other attributes. The original data set contains 82 features and 2930 data points. You can read more about this dataset [here](http://ww2.amstat.org/publications/jse/v19n3/decock.pdf).

However, for the purpose of understanding how Linear Regression works, we will specifically, work on the following features of the house

* ExterQual
* AllFlrsSF
* GrLivArea
* SimplOverallCond
* GarageArea
* TotRmsAbvGrd
* LotFrontage

**Brief explanation of the dataset & features**

* ExterQual (Ordinal): Evaluates the quality of the material on the exterior

5: Excellent 4: Good 3: Average/Typical 2: Fair 1: Poor

* AllFlrsSF(Continuous): Total square feet for 1st and 2nd floor combined
* GrLivArea (Continuous): Above grade (ground) living area square feet
* SimplOverallCond (Ordinal): Rates the overall condition of the house

1: Bad 2: Average 3: Good

* Garage Area (Continuous): Size of garage in square feet
* TotRmsAbvGrd (Nominal): Total rooms above grade (does not include bathrooms)
* LotFrontage (Continuous): Linear feet of street-connected to property

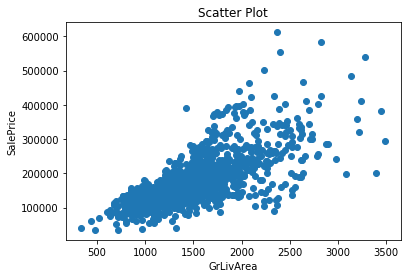
**What we want as an outcome?**

Using the set of some basic attributes that are related to the price of the house, predict the sale price for a new house using Linear Regression.

**Need for Linear Regression**

**Intuition for Linear regression**

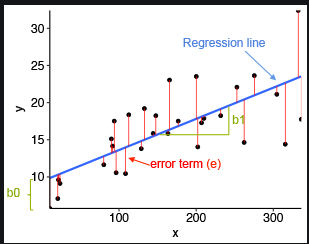
Let's start with what information we have: The main goal is to build a machine learning model that can predict the selling price of the house given some of its features like GrLivArea, Garage Area etc. If you do a scatter plot with SalePrice which is the target variable and GrLivArea, a feature of the house, we might get something similar to the following:

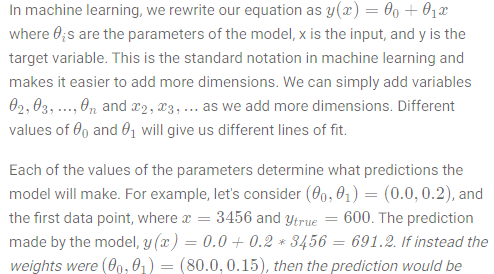


**Know Your Linear Regression**

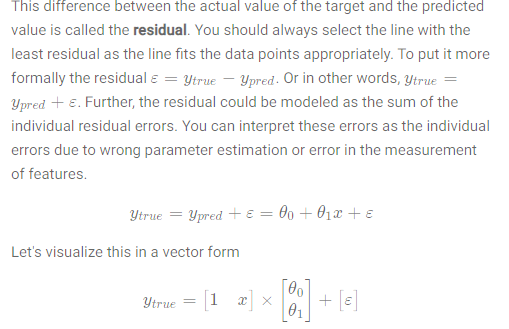
In simple linear regression, we establish a relationship between the target variable and input variables by fitting a line, known as the regression line.

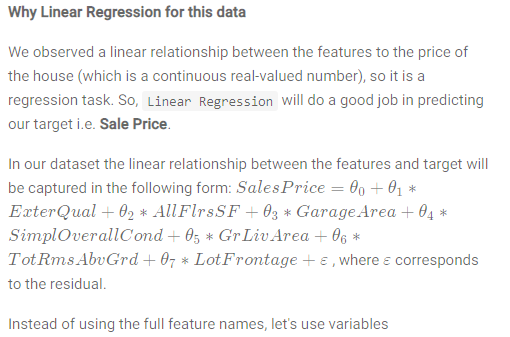
In general, a line can be represented by linear equation y = mx + b Where y is the dependent variable, x is the independent variable, m is the slope, b is the intercept.

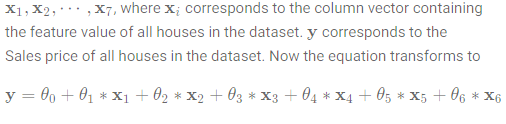


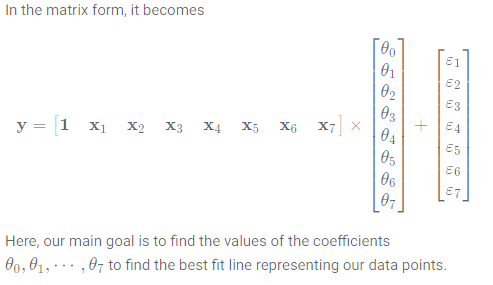








.



Assumptions of Linear Regression

#### Linear Relationship

According to this assumption, the relationship between response (Dependent Variables) and feature variables (Independent Variables) should be linear.

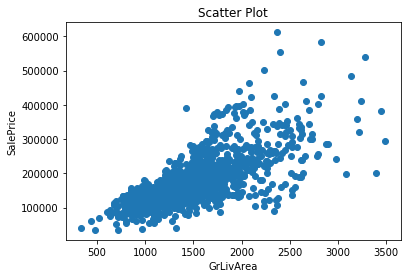
**Why it is important?**

* Linear regression only captures the linear relationship, as it is trying to fit a linear model to the data.

**How to validate it?**

* The linearity assumption can be tested using scatter plots.

In the scatter plot for SalePrice vs GrLivArea, you can clearly see that a linear pattern is evident here i.e. as the value of GrLivArea increases the SalePrice also increases and vice-versa.



#### Little or No Multicollinearity Assumption

**What is multicollinearity?**

It is assumed that there is little or no multicollinearity in the data. But what do we mean by multicollinearity? Well, multicollinearity occurs when **independent variables in a regression model are correlated**. This correlation is a problem because independent variables should be independent. If the degree of correlation between variables is high enough, it can cause problems when you fit the model and interpret the results.

**Why sweat over multicollinearity?**

The interpretation of a regression coefficient is that it represents the mean change in the dependent variable for each unit change in an independent variable when you hold all of the other independent variables constant. However, when independent variables are correlated, changes in one variable, in turn, shifts another variable/variable. The stronger the correlation, the more difficult it is to change one variable without changing another. It becomes difficult for the model to estimate the relationship between each independent variable and the dependent variable independently because the independent variables tend to change in unison.

**Effects of multicollinearity**

* It results in unstable parameter estimates which makes it very difficult to assess the effect of independent variables.
* Weakens the statistical power of the regression model

**How to validate it?**

* Multicollinearity occurs when the features (or independent variables) are not independent of each other. Pair plots of features help validate.
* You can also calculate the correlation coefficient (Pearson or Spearman) to figure out which features are correlated.

**Treating multicollinearity**

* Remove some of the highly correlated independent variables.
* Linearly combine the independent variables, such as adding them together.

# Ordinary Least Squares

The ordinary least squares (OLS) approach to regression allows us to estimate the parameters of a linear model. The goal of this method is to determine the linear model that minimizes the sum of the squared errors between the observations in a dataset and those predicted by the model.

**Discrete example to understand OLS**

Let's take an example to clarify the understanding of the above calculation. For simplicity, we will consider only one feature i.e. GarageArea as our feature vector X and we have 'SalePrice' as our target vector y. Hence, we have data points as follows

|  |  |
| --- | --- |
| **GarageArea (X)** | **SalePrice (y)** |
| 548 | 208500 |
| 460 | 181500 |
| 608 | 223500 |
| 642 | 140000 |
| 836 | 250000 |
| 480 | 143000 |
| 636 | 307000 |
| 484 | 200000 |
| 468 | 129900 |
| 205 | 118000 |

Let us first calculate the mean of X and y

*x*ˉ=548+460+608+642+836+480+636+484+468+205​/10 =536

*y*ˉ=10208500+181500+223500+140000+250000+143000+307000+20000+129900+11800​/10=190140

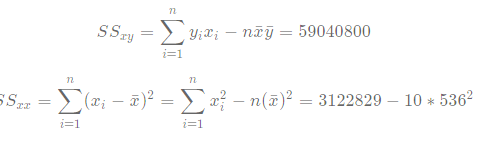
We now need to calculate ∑*y*∗*x*

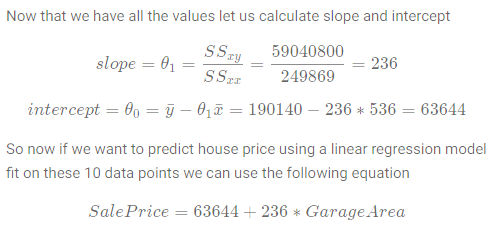
∑*y*∗*x= (208500*548) + (181500*460) + … + (118000*205) = 1078191200

Next we need to calculate ∑*x*2



Now we will calculate the sum of cross deviations and the sum of squared deviations





**Model building with scikit-learn**

Now that you have gone through the mathematical concept of Ordinary Least Squares let us see how we can implement the same in Python using sklearn library. The steps involved for building a model will be as follows:

* Train the model using sklearn
* Test it on the test set using sklearn

The code snippet for model building with scikit learn is described below:

*# import packages*

**from** sklearn.linear\_model **import** LinearRegression

*# instantiate linear regression model*

linreg = LinearRegression()

*# fit model on training data*

linreg.fit(X\_train, y\_train)

*# make predictions on test data*

pred = linreg.predict(X\_test)

**Evaluation : Mean Absolute Error**

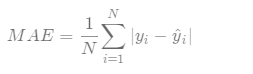
Now that you have fitted your model on training data, it is time to test it on unseen data. You also need to have some kind of measure to quantify the performance of the model. This measure is captured by what we call the error metrics and there are many different forms depending on the problem statement.

We have a regression problem at hand and the different types of error metrics that can be used are:

* Mean Absolute Error (MAE)
* Root Mean Squared Error (RMSE)
* R-Squared

**Mean Absolute Error**

So we already know what a residual is. To recap, it is the difference between our prediction and the true value. Mean absolute error is nothing but the average of absolute values of these residuals. We can write a simple formula for Mean Absolute Error (MAE) as follows.



**Calculating MAE with scikit-learn**

Scikit-learn provides a very easy way to calculate MAE. The code snippet is shown below:

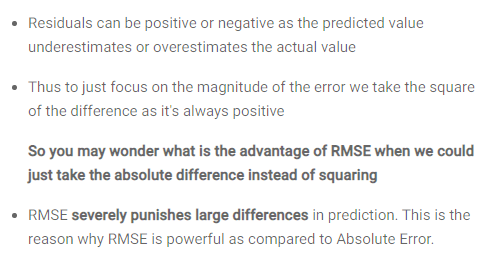
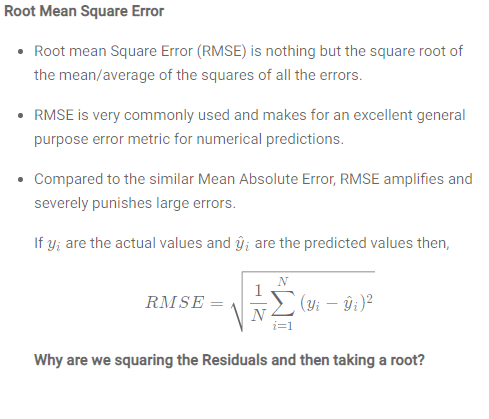
*# import packages*

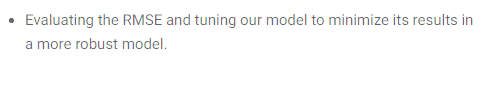
**from** sklearn.metrics **import** mean\_absolute\_error

*# MAE calculation*

mae = mean\_absolute\_error(y\_test, y\_pred)

The variable mae gives us the Mean Absolute Error for our predictions y\_pred and true target y\_test.





**What is R-Squared?**

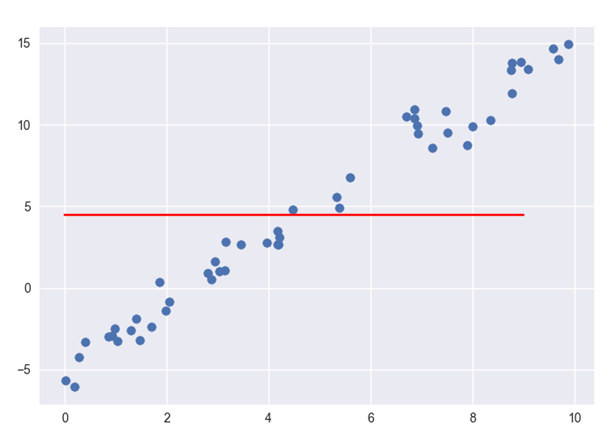
R-squared is a statistical measure of how close the data are to the fitted regression line i.e. **it measures the goodness of fit of a straight line**. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.

It is a **measure of the proportion of variability in the response that is explained by the regression model.**



R-squared is always between 0 and 100%:

1. 0 % indicates that the model explains none of the variability of the response data around its mean.



1. 100% indicates that the model explains all the variability of the response data around its mean.

