**Regression Models**

1. **Logistic Regression**

<https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression>

<https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html>

1. Ordinary Least Squares

* from **sklearn** import **linear\_model**
* reg = linear\_model.LinearRegression() : creating a linear regression object
* OR
* from sklearn.linear\_model import LinearRegression
* reg.fit([[0, 0], [1, 1], [2, 2]], [0, 1, 2]) : training the model using training sets
* reg.coef\_

**This equation minimizes the sum of the squared residuals**

1. Ridge Regression

Ordinary least squares overfit the data and have high variance, so here we introduce a line of best fit that does not fit the training data as wee. This introduces a small bias into the line of best fit and in return we get a significant drop in variance

Here the linear equation minimizes the sum of the squared residuals AND ( this adds a penalty to the least squares method and the lambda determines how severe the penalty is )

The larger we make lambda the more the slope **gets close to zero** and our prediction is very sensitive to the X parameter.

To decide which lambda to use we try a bunch of values for lambda and use cross validation, typically 10-fold cross validation, to determine which one results in the lowest variance

RidgeCV implements ridge regression with built-in cross-validation of the alpha parameter

* linreg = linear\_model.RidgeCV()

1. Lasso Regression

Similar to ridge regression but has a very important difference ;

- minimizes the sum of the squared residuals AND lambda\* the modulus of the slope .

When increasing lambda, **the slope gets smaller until the slope equals 0**

It is better when most variables are useless as it excludes useless variables effectively reducing the number of features upon which the given solution is dependent