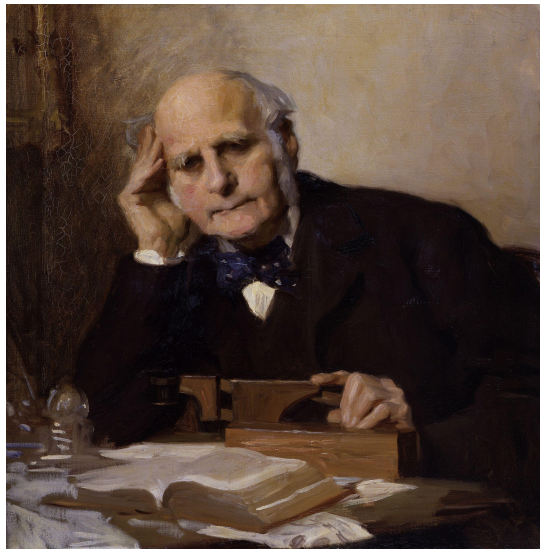
Linear Regression

Introduction

Introduction



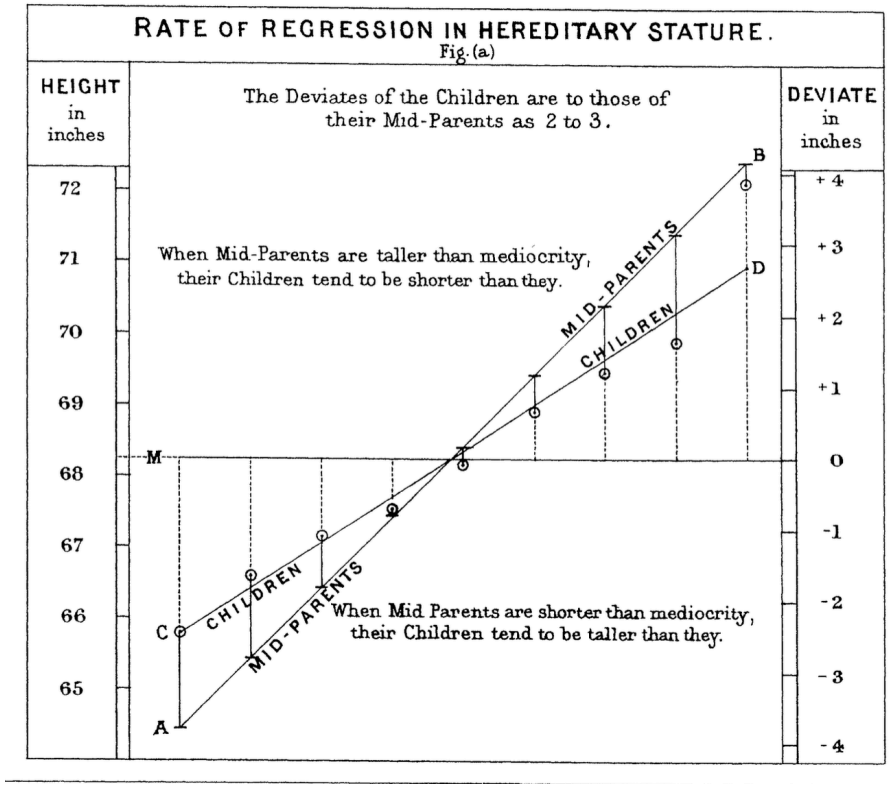
Sir Francis Galton by Charles Wellington Furse

ANTHROPOLOGICAL MISCELLANEA.

REGRESSION *towards* MEDIOCRITY *in* HEREDITARY STATURE.
By FRANCIS GALTON, F.R.S., &c.

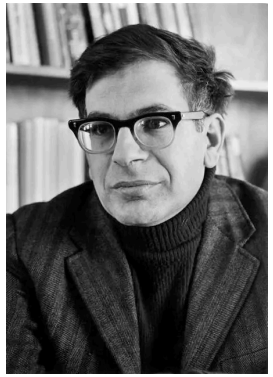
In 1886 Francis Galton published his observations about the height of children compared to their parents.

Regression to the Mean



Defined mid-parent as the average of the heights of the parents.

Praise versus Punishment



Daniel Kahneman

Firing Coaches

NFL Coaches

Reference

Is it a good strategy to fire a coach after a bad season?

> bad season -> fire and replace coach -> better season next year

This applies to many situations

> “I had so many colds last year, and then I started herbal remedies, and I have much fewer colds this year.”

> “My investments were doing quite bad last year, so switched investment managers, and now it is doing much better.”

Model Types

The most common form of machine learning is **regression**, which means constructing an equation that describes the relationships among variables.

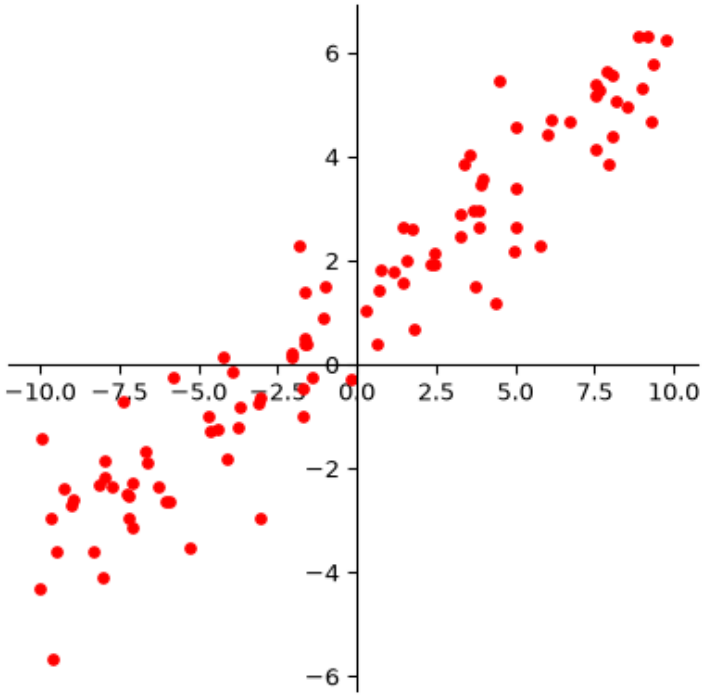
It is a form of supervised learning:

- whereas **classification** deals with predicting categorical features (labels or classes),
- **regression** deals with predicting continuous features (real values).

Fit to Linear Models

For example, we may look at these points and decide to model them using a line.

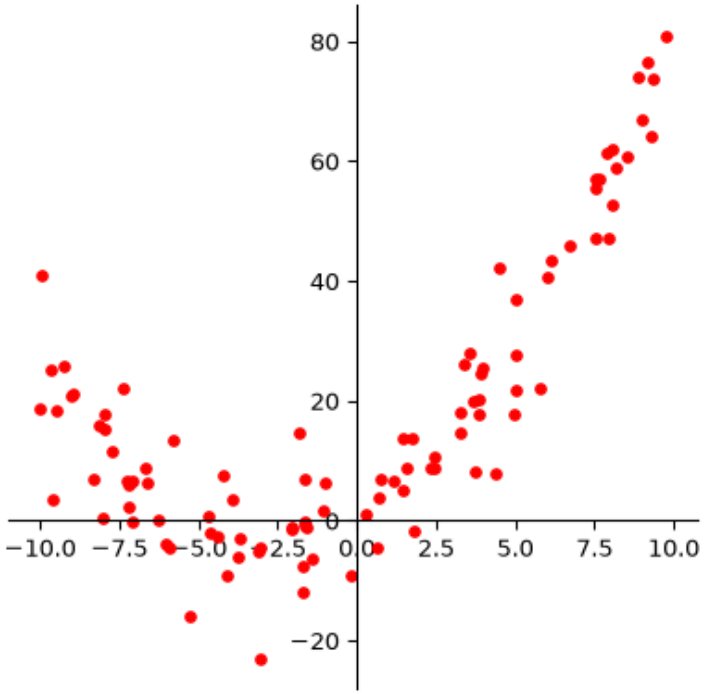
▶ Code



Fit to Quadratic Models

We may look at these points and decide to model them using a quadratic function.

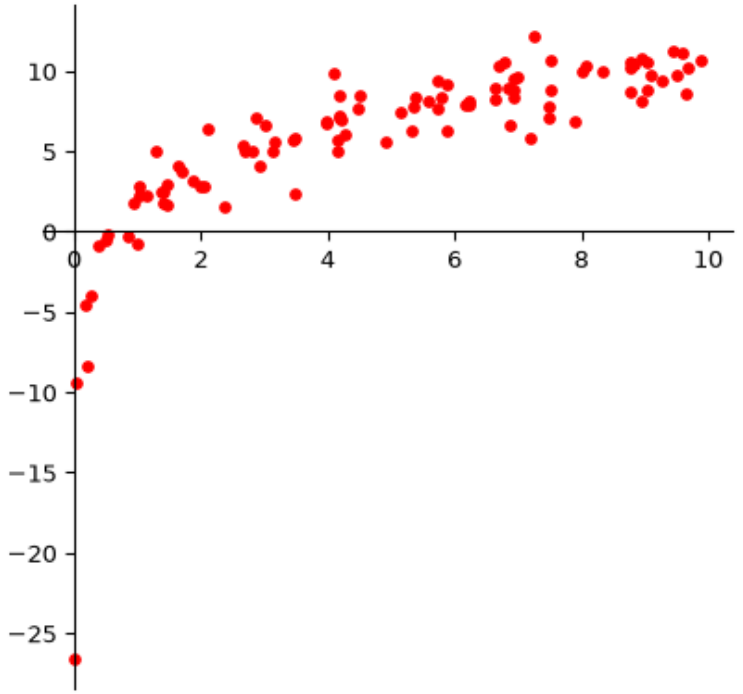
▶ Code



Fit to Logarithmic Models

And we may look at these points and decide to model them using a logarithmic function.

▶ Code



Approach

Clearly, none of these datasets agrees perfectly with the proposed model.

So the question arises:

How do we find the **best** linear function (or quadratic function, or logarithmic function) given the data?

Framework

Framework

This problem has been studied extensively in the field of statistics.

Certain terminology is used:

- Some values are referred to as “independent,” and
- Some values are referred to as “dependent.”

Framework cont.

The basic regression task is:

- given a set of independent variables
- and the associated dependent variables,
- estimate the parameters of a model (such as a line, parabola, etc) that describes how the dependent variables are related to the independent variables.

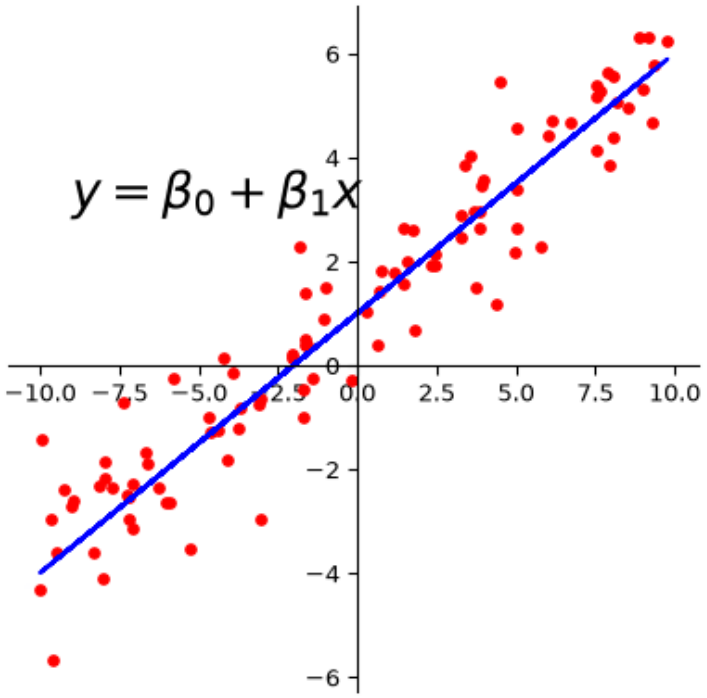
Framework cont.

▶ Code

The **independent variables** are collected into a matrix X, sometimes called the **design matrix.**

The dependent variables are collected into an **observation** vector $\mathbf{y}$.

The parameters of the model (for any kind of model) are collected into a **parameter** vector β.



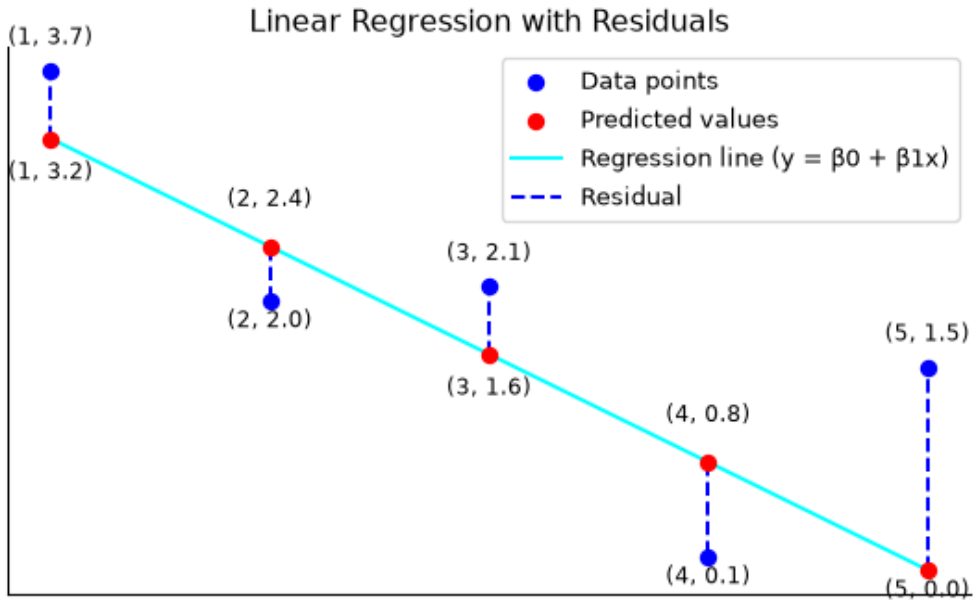
Least-Squares Lines

Least-Squares Lines cont.

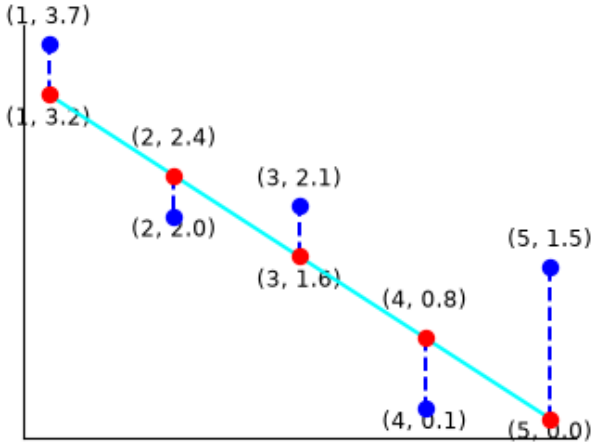
Suppose we have a line $y = \beta_0 + \beta_1 x$.

For each data point (x_j, y_j), there is a point $(x_j, \beta_0 + \beta_1 x_j)$ that is the point on the line with the same x-coordinate.

▶ Code



Least-Squares Lines cont.



We call

- y_j the **observed** value of y and
- $\beta_0 + \beta_1 x_j$ the **predicted** y-value.

A least-squares problem

If the data points were on the line, the parameters β_0 and β_1 would satisfy the equations

$$\begin{aligned}
\beta_0 + \beta_1 x_1 &= y_1, \\
\beta_0 + \beta_1 x_2 &= y_2, \\
\beta_0 + \beta_1 x_3 &= y_3, \\
&\vdots \\
\beta_0 + \beta_1 x_n &= y_n.
\end{aligned}$$

We can write this system as

$$X\beta = \mathbf{y},$$

where

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}.$$

Of course, if the data points don't actually lie exactly on a line,

… then there are no parameters β_0, β_1 for which the predicted y-values in $X\beta$ equal the observed y-values in $\mathbf{y}$,

… and $X\beta = \mathbf{y}$ has no solution.

Now, since the data doesn't fall exactly on a line, we have decided to seek the β that minimizes the sum of squared residuals, i.e.,

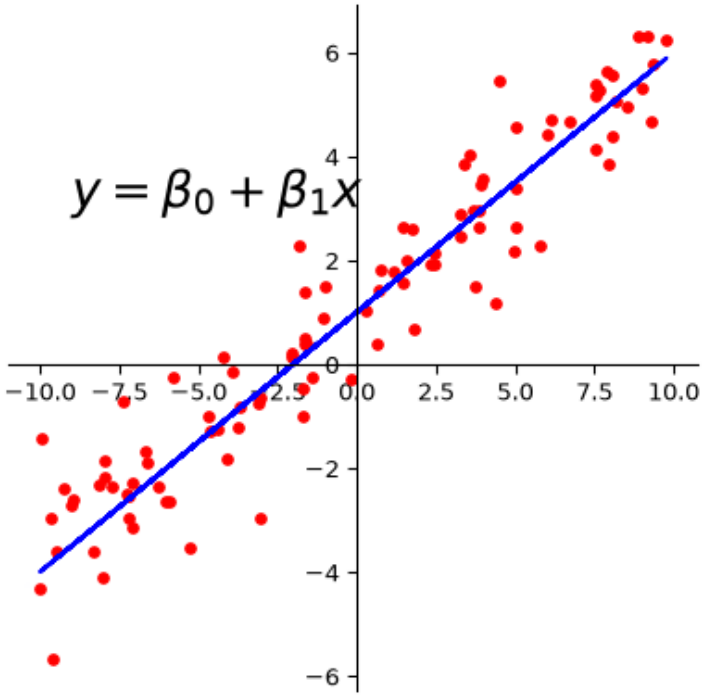
$$\sum_i (\beta_0 + \beta_1 x_i - y_i)^2 = \|X\beta - \mathbf{y}\|^2.$$

This is key:

> **Important**
>
> **The sum of squares of the residuals** is **exactly** the **square of the distance between the vectors $X\beta$ and $\mathbf{y}$**.

Computing the least-squares solution of $X\beta = \mathbf{y}$ is equivalent to finding the β that determines the least-squares line.

▶ Code



Now, to obtain the least-squares line, find the least-squares solution to $X\beta = \mathbf{y}$.

From linear algebra we know that the least squares solution of $X\beta = \mathbf{y}$ is given by the solution of the **normal equations**:

$$X^T X\beta = X^T \mathbf{y},$$

because $X\beta - \mathbf{y}$ is normal (e.g. orthogonal) to the column space of X.

The General Linear Model

The General Linear Model

Another way that the inconsistent (.i.e. no exact solution) linear system is often written is to collect all the residuals into a **residual vector.**

Then an exact equation is

$$y = X\beta + \epsilon,$$

where ϵ is the **residual vector.**

Any equation of this form is referred to as a **linear model.**

In this formulation, the goal is to find the β so as to minimize the **norm** of ϵ, i.e., $\|\epsilon\|$.

Least-Squares Fitting of Other Models

In model fitting, the parameters of the model are what is unknown.

A central question for us is whether the model is *linear* in its parameters.

For example, is this model linear in its parameters?

$$y = \beta_0 e^{-\beta_1 x}$$

For a model that is linear in its parameters, an observation is a linear combination of (arbitrary) known functions.

In other words, a model that is linear in its parameters is

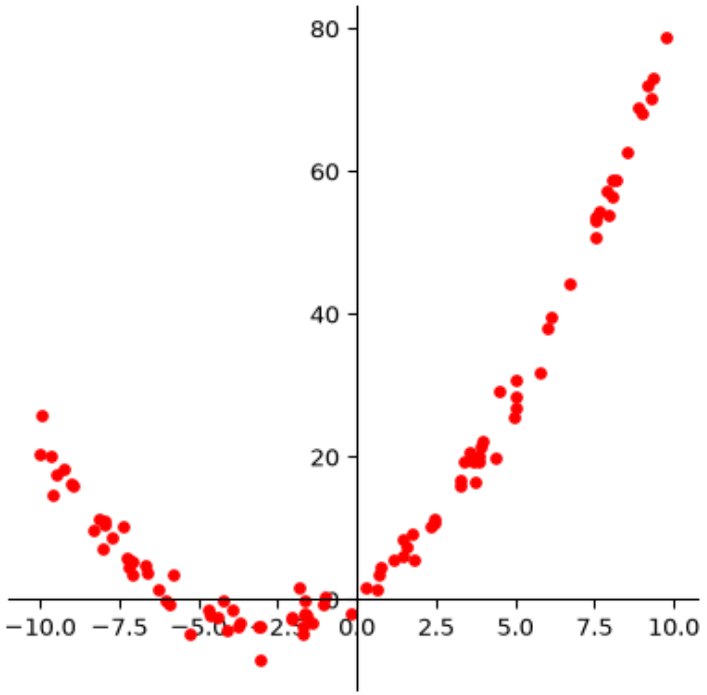
$$y = \beta_0 f_0(x) + \beta_1 f_1(x) + \cdots + \beta_n f_n(x)$$

where $f_0, \ldots, f_n$ are known functions and $\beta_0, \ldots, \beta_n$ are parameters.

Example

Suppose data points $(x_1, y_1), \ldots, (x_n, y_n)$ appear to lie along some sort of parabola instead of a straight line.

▶ Code



As a result, we wish to approximate the data by an equation of the form

$$y = \beta_0 + \beta_1 x + \beta_2 x^2.$$

Let's describe the linear model that produces a "least squares fit" of the data by the equation.

Solution

The ideal relationship is $y = \beta_0 + \beta_1 x + \beta_2 x^2$.

Suppose the actual values of the parameters are $\beta_0, \beta_1, \beta_2$.
Then the coordinates of the first data point satisfy the equation

$$y_1 = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \epsilon_1,$$

where ϵ_1 is the residual error between the observed value y_1 and the predicted y-value.

Each data point determines a similar equation:

$$\begin{aligned} y_1 &= \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \epsilon_1, \\ y_2 &= \beta_0 + \beta_1 x_2 + \beta_2 x_2^2 + \epsilon_2, \\ &\vdots \\ y_n &= \beta_0 + \beta_1 x_n + \beta_2 x_n^2 + \epsilon_n. \end{aligned}$$

Clearly, this system can be written as $\mathbf{y} = X\beta + \epsilon$.

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

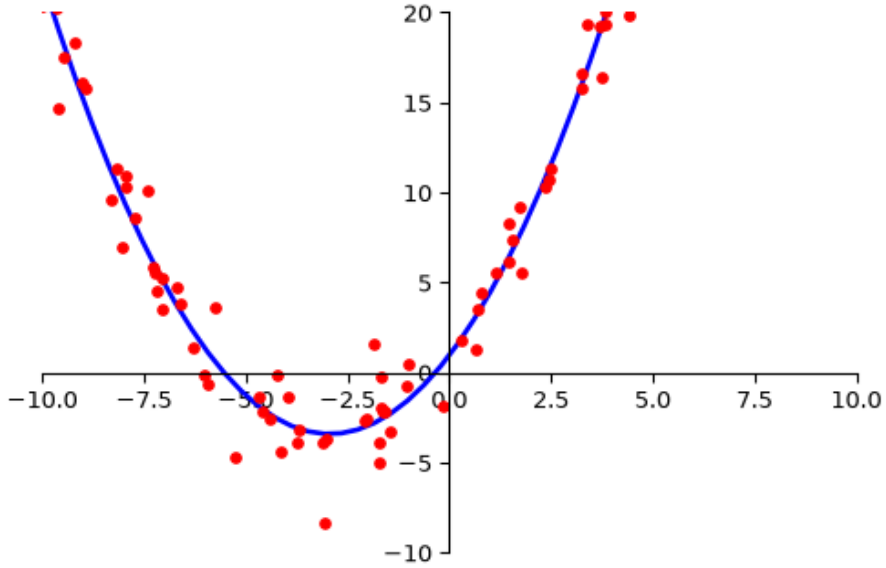
Let's solve for β.

```
m = np.shape(xquad)[0]
X = np.array([np.ones(m), xquad, xquad**2]).T

# Solve for beta
beta = np.linalg.inv(X.T @ X) @ X.T @ yquad
```

And plot the results.

▶ Code



Now let's try a different model.

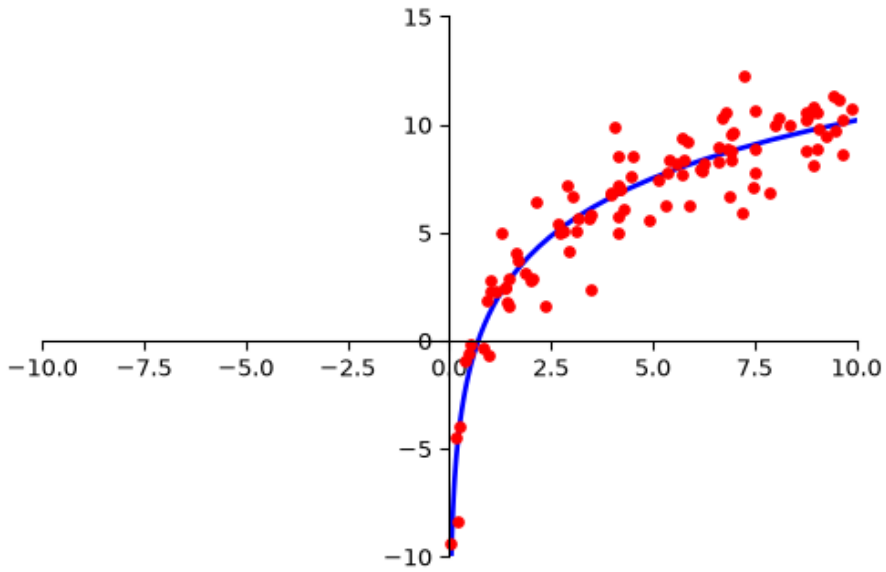
$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & log(x_1) \\ 1 & log(x_2) \\ \vdots & \vdots \\ 1 & log(x_n) \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

```
m = np.shape(xlog)[0]
X = np.array([np.ones(m), np.log(xlog)]).T

# Solve for beta
beta = np.linalg.inv(X.T @ X) @ X.T @ ylog
```

And plot the results.

▶ Code



Multiple Regression

Suppose an experiment involves two independent variables – say, u and v, – and one dependent variable, y.

A simple equation for predicting y from u and v has the form

$$y = \beta_0 + \beta_1 u + \beta_2 v.$$

Since there is more than one independent variable, this is called **multiple regression.**

Multiple Regression, cont.

A more general prediction equation might have the form

$$y = \beta_0 + \beta_1 u + \beta_2 v + \beta_3 u^2 + \beta_4 uv + \beta_5 v^2.$$

A least squares fit to equations like this is called a **trend surface.**

In general, a linear model will arise whenever y is to be predicted by an equation of the form

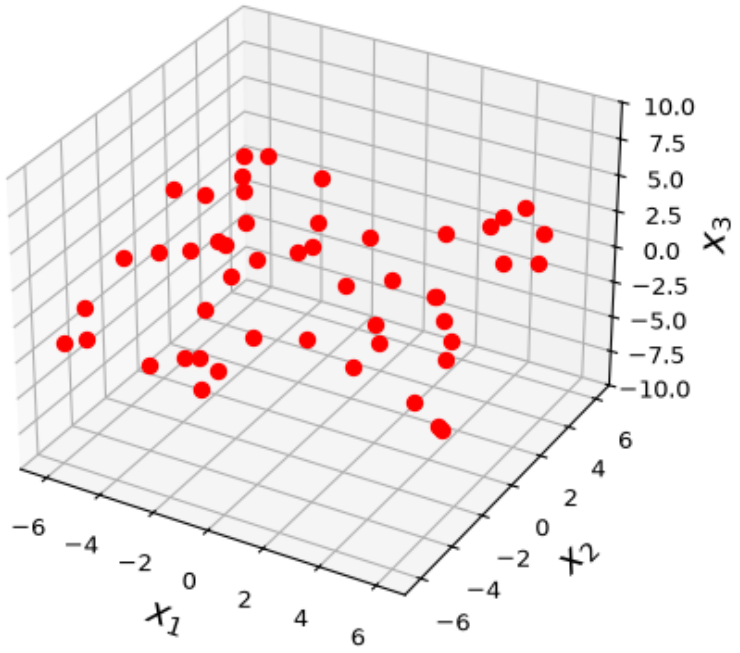
$$y = \beta_0 f_0(u, v) + \beta_1 f_1(u, v) + \cdots + \beta_n f_n(u, v),$$

with $f_0, \ldots, f_n$ any sort of known functions and $\beta_0, \ldots, \beta_n$ unknown weights.

Example

Let's take an example. Here are a set of points in $\mathbb{R}^3$:

▶ Code



Example

In geography, local models of terrain are constructed from data

$$(u_1, v_1, z_1), \dots, (u_n, v_n, z_n),$$

where u_j, v_j, and z_j are latitude, longitude, and altitude, respectively.

Solution

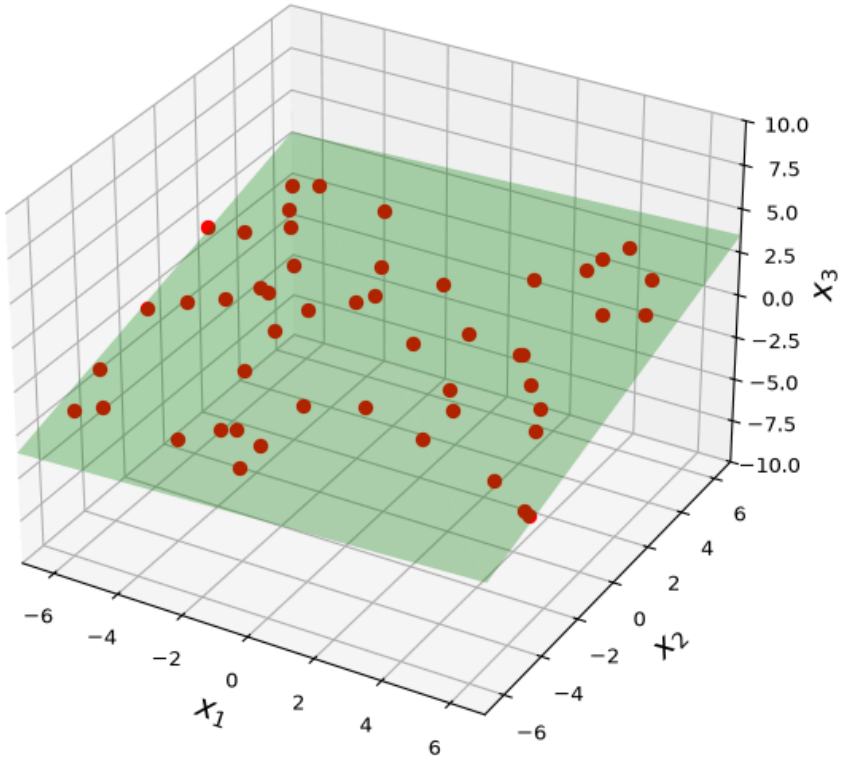
We expect the data to satisfy these equations:

$$\begin{aligned} y_1 &= \beta_0 + \beta_1 u_1 + \beta_2 v_1 + \beta_3 z_1 + \epsilon_1, \\ y_2 &= \beta_0 + \beta_1 u_2 + \beta_2 v_2 + \beta_3 z_2 + \epsilon_2, \\ &\vdots \\ y_n &= \beta_0 + \beta_1 u_n + \beta_2 v_n + \beta_3 z_n + \epsilon_n. \end{aligned}$$

This system has the matrix for $\mathbf{y} = X\beta + \epsilon$, where

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}, \quad X = \begin{bmatrix} 1 & u_1 & v_1 & z_1 \\ 1 & u_2 & v_2 & z_2 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & u_n & v_n & z_n \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}, \quad \epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}.$$

▶ Code



This example shows that the linear model for multiple regression has the same form as the model for the simple regression in the earlier examples.

We can see that the general principle is the same across all the different kinds of linear models.

Measuring the fit of a regression model: R^2

Given any X and $\mathbf{y}$, the above algorithm will produce an output $\hat{\beta}$.

But how do we know whether the data is in fact well described by the model?

Measuring the fit, cont.

For any given n, we can equally work with just

$$\sum_{i=1}^{n} (y_i - \overline{y})^2 ,$$

which is called the **Total Sum of Squares** (TSS).

Total Sum of Squares

Now to measure the quality of fit of a model, we break TSS down into two components.

For any given $\mathbf{x}_i$, the prediction made by the model is $\hat{y}_i = \mathbf{x}_i^T \beta$.

We can break the total sum of squares into two parts:

- the residual ϵ is $y_i - \hat{y}_i$, and
- the part that the model "explains" is $\hat{y}_i - \overline{y}$.

RSS and ESS

Se we define the Residual Sum of Squares (RSS) as:

$$\mathrm{RSS} = \sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2,$$

and the Explained Sum of Squares (ESS) as:

$$\mathrm{ESS} = \sum_{i=1}^{n} \left(\hat{y}_i - \overline{y}\right)^2,$$

One can show that the total sum of squares is exactly equal to the sum of squares of the residuals plus the sum of squares of the explained part.

In other words:

$$\mathrm{TSS} = \mathrm{RSS} + \mathrm{ESS}.$$

R^2, cont.

Now, a good fit is one in which the model explains a large part of the variance of $\mathbf{y}$.

So the measure of fit R^2 is defined as:

$$R^2 = \frac{\text{ESS}}{\text{TSS}} = 1 - \frac{\text{RSS}}{\text{TSS}} = 1 - \frac{\sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n} \left(y_i - \overline{y}\right)^2}.$$

As a result, $0 \leq R^2 \leq 1$.

R^2, cont.

The closer the value of R^2 is to 1 the better the fit of the regression:

- small values of RSS imply that the residuals are small and therefore we have a better fit.

R^2 is called the **coefficient of determination.**

It tells us “how well does the model predict the data?”

Warning

Do **not** confuse R^2 with Pearson's r, which is the **correlation coefficient.**

(To make matters worse, sometimes people talk about r^2… very confusing!)

OLS in Practice

OLS in Practice

First, we'll look at a test case on synthetic data. We'll use Scikit-Learn's `make_regression` function.

```
from sklearn import datasets
X, y = datasets.make_regression(n_samples=100, n_features=20, n_informative=5, bias=0.1, noise=30, random_sta
```

And then use the `statsmodels` package to fit the model.

```
import statsmodels.api as sm

# Add a constant (intercept) to the design matrix
#X_with_intercept = sm.add_constant(X)
#model = sm.OLS(y, X_with_intercept)

model = sm.OLS(y, X)
results = model.fit()
```

And print the summary of the results.

▶ Code

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared (uncentered):                   0.969
Model:                            OLS   Adj. R-squared (uncentered):              0.961
Method:                 Least Squares   F-statistic:                              123.8
Date:                Mon, 31 Aug 2026   Prob (F-statistic):                    1.03e-51
Time:                        12:04:07   Log-Likelihood:                         -468.30
No. Observations:                 100   AIC:                                      976.6
Df Residuals:                      80   BIC:                                      1029.
Df Model:                          20
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
x1            12.5673      3.471      3.620      0.001       5.659      19.476
x2            -3.8321      2.818     -1.360      0.178      -9.440       1.776
x3            -2.4197      3.466     -0.698      0.487      -9.316       4.477
x4             2.0143      3.086      0.653      0.516      -4.127       8.155
x5            -2.6256      3.445     -0.762      0.448      -9.481       4.230
x6             0.7894      3.159      0.250      0.803      -5.497       7.076
```

The R^2 value is very good. We can see that the linear model does a very good job of predicting the observations y_i.

Note

Aside: Summary Statistics

Here is what all the statistics in the summary table mean:

1. **R-squared (Uncentered):**

- Definition: Measures the proportion of the variation in the dependent variable that is explained by the model, without subtracting the mean of the dependent variable.
- Formula: $R^2(\text{uncentered}) = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum y_i^2}$

where y_i is the actual value, and $\hat{y}_i$ is the predicted value from the regression.

- Interpretation: Higher values (closer to 1) indicate a better fit, but this version of R^2 can sometimes be misleading because it doesn't account for the mean.

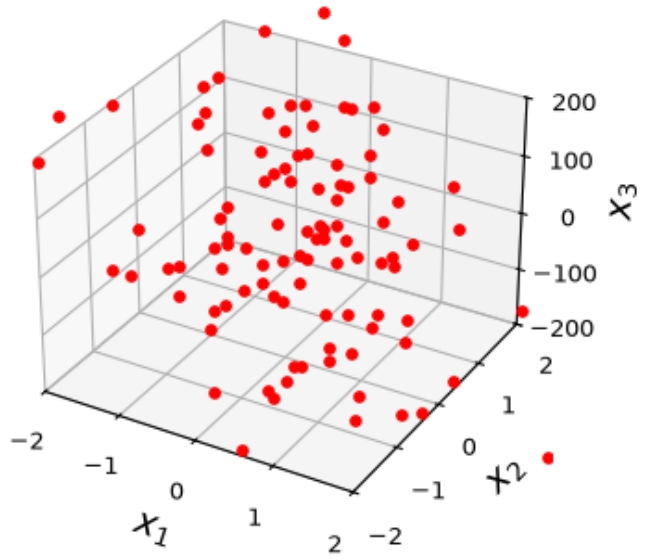
2. **Adjusted R-squared (Uncentered):**

- Definition: Adjusts the uncentered R^2 to account for the number of predictors in the model, preventing overfitting by penalizing for adding more variables.

Back to the example.

Some of the independent variables may not contribute to the accuracy of the prediction.

▶ Code



Note that each parameter of an independent variable has an associated confidence interval.

If a coefficient is not distinguishable from zero, then we cannot assume that there is any relationship between the independent variable and the observations.

In other words, if the confidence interval for the parameter includes zero, the associated independent variable may not have any predictive value.

▶ Code

```
Confidence Intervals: [[  5.65891465  19.47559281]
 [ -9.44032559   1.77614877]
 [ -9.31636359   4.47701749]
 [ -4.12661379   8.15524508]
 [ -9.4808662    4.22965424]
 [ -5.49698033   7.07574692]
 [-10.22359973   4.08684835]
 [ 83.74738375  96.52928603]
 [ -6.77896356   6.75226985]
 [  8.80365396  21.73126149]
 [ -6.86882065   6.4194618 ]
 [ -6.97868351   7.1332267 ]
 [ -6.71228582   6.2218515 ]
 [ 82.96557061  97.07028228]
 [ -5.74782503   9.08465366]
 [ -1.06173893   9.85081724]
 [  2.02753258  15.5561241 ]
 [ 66.56165458  80.19256546]
 [ -8.90825108   5.0804296 ]
 [ -7.85545335   5.21424811]]
```

Find the independent variables that are not significant.

▶ Code

```
array([False,  True,  True,  True,  True,  True,
True, False,  True,
       False,  True,  True,  True, False,  True,
True, False, False,
        True,  True])
```

Let's look at the shape of significant variables.

▶ Code

```
(100, 6)
```

By eliminating independent variables that are not significant, we help avoid overfitting.

▶ Code

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared (uncentered):                   0.965
Model:                            OLS   Adj. R-squared (uncentered):              0.963
Method:                 Least Squares   F-statistic:                              437.1
Date:                Mon, 31 Aug 2026   Prob (F-statistic):                    2.38e-66
Time:                        12:04:07   Log-Likelihood:                         -473.32
No. Observations:                 100   AIC:                                      958.6
Df Residuals:                      94   BIC:                                      974.3
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
x1            11.9350      3.162      3.775      0.000       5.657      18.213
x2            90.5841      2.705     33.486      0.000      85.213      95.955
x3            14.3652      2.924      4.913      0.000       8.560      20.170
x4            90.5586      3.289     27.535      0.000      84.028      97.089
x5             8.3185      3.028      2.747      0.007       2.307      14.330
x6            71.9119      3.104     23.169      0.000      65.749      78.075
```

Real Data: House Prices in Ames, Iowa

Let’s see how powerful multiple regression can be on a real-world example.

A typical application of linear models is predicting house prices.

Linear models have been used for this problem for decades, and when a municipality does a value assessment on your house, they typically use a linear model.

We can consider various measurable attributes of a house (its “features”) as the independent variables, and the most recent sale price of the house as the dependent variable.

For our case study, we will use the features:

- Lot Area (sq ft),
- Gross Living Area (sq ft),
- Number of Fireplaces,
- Number of Full Baths,
- Number of Half Baths,
- Garage Area (sq ft),
- Basement Area (sq ft)

So our design matrix will have 8 columns (including the constant for the intercept):

$$X\beta = \mathbf{y},$$

and it will have one row for each house in the data set, with y the sale price of the house.

Ames Housing Data

We will use data from housing sales in Ames, Iowa from 2006 to 2009:

Ames Iowa

Tim Kiser (w:User:Malepheasant) CC BY-SA 2.5 via Wikimedia Commons

▶ Code

▶ Code

	LotArea	GrLivArea	Fireplaces	FullBath	HalfBath	GarageArea	TotalBsmtSI
0	8450	1710	0	2	1	548	856
1	9600	1262	1	2	0	460	1262
2	11250	1786	1	2	1	608	920
3	9550	1717	1	1	0	642	756
4	14260	2198	1	2	1	836	1145

Some things to note here:

- House prices are in dollars
- Areas are in square feet
- Rooms are in counts

Do we have scaling concerns here?

No, because each feature will get its own β, which will correct for the scaling differences between different units of measure.

▶ Code

> **Note**
>
> Note that removing the intercept will cause the R^2 to go up, which is counter-intuitive. The reason is explained here] – but amounts to the fact that the formula for R2 with/without an intercept is different.

Ames Housing Data, cont.

Let's split the data into training and test sets.

▶ Code

Fit the model to the training data.

▶ Code

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.759
Model:                            OLS   Adj. R-squared:                  0.757
Method:                 Least Squares   F-statistic:                     325.5
Date:                Mon, 31 Aug 2026   Prob (F-statistic):          1.74e-218
Time:                        12:04:07   Log-Likelihood:                -8746.5
No. Observations:                 730   AIC:                         1.751e+04
Df Residuals:                     722   BIC:                         1.755e+04
Df Model:                           7
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -4.285e+04   5350.784     -8.007      0.000   -5.34e+04   -3.23e+04
LotArea        0.2361      0.131      1.798      0.073      -0.022       0.494
GrLivArea      48.0865      4.459     10.783      0.000      39.332      56.841
Fireplaces   1.089e+04   2596.751      4.192      0.000    5787.515     1.6e+04
FullBath      1.49e+04   3528.456      4.224      0.000    7977.691    2.18e+04
```

We see that we have:

- β_0: Intercept of -$42,850
- β_1: Marginal value of one square foot of Lot Area: $0.23
  - but **NOTICE** - this coefficient is not statistically different from zero!
- β_2: Marginal value of one square foot of Gross Living Area: $48
- β_3: Marginal value of one additional fireplace: $10,890
- β_4: Marginal value of one additional full bath: $14,900
- β_5: Marginal value of one additional half bath: $15,600
- β_6: Marginal value of one square foot of Garage Area: $99
- β_7: Marginal value of one square foot of Basement Area: $62

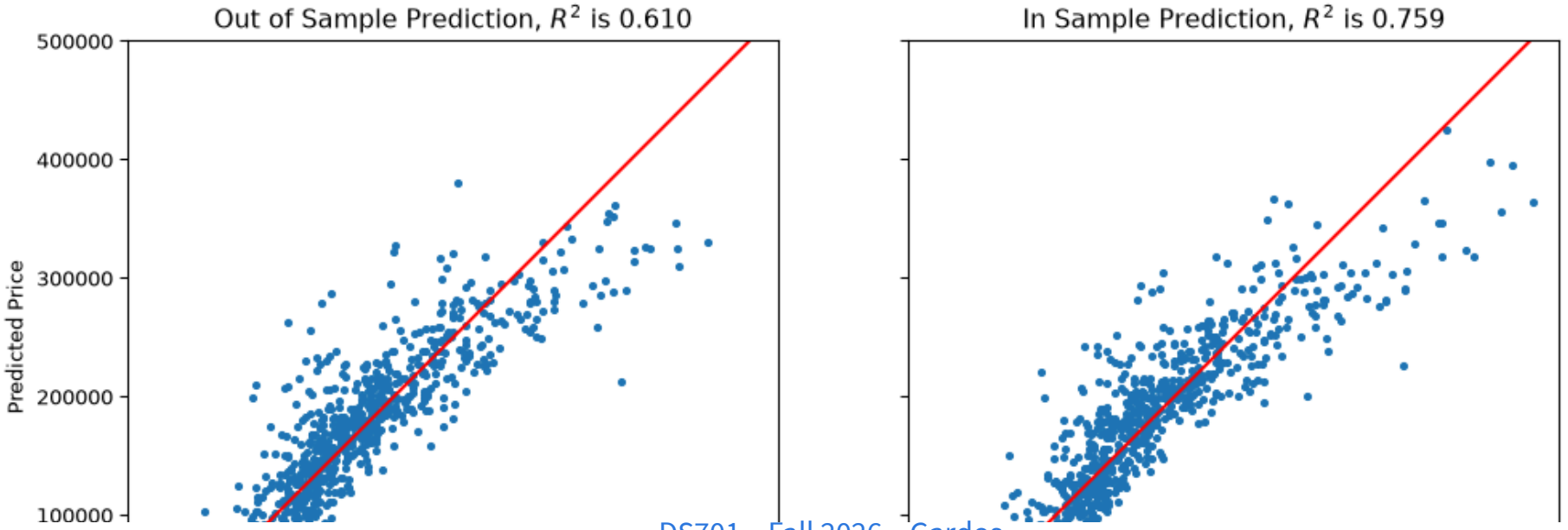
Is our model doing a good job?

There are many statistics for testing this question, but we'll just look at the predictions versus the ground truth.

For each house we compute its predicted sale value according to our model:

$$\hat{\mathbf{y}} = X\hat{\beta}$$

▶ Code



Recap

- Linear models are a powerful tool for prediction and inference
- The normal equations provide a simple way to compute the model coefficients
- The R^2 statistic provides a simple measure of fit
- Careful use of the model is required to avoid overfitting