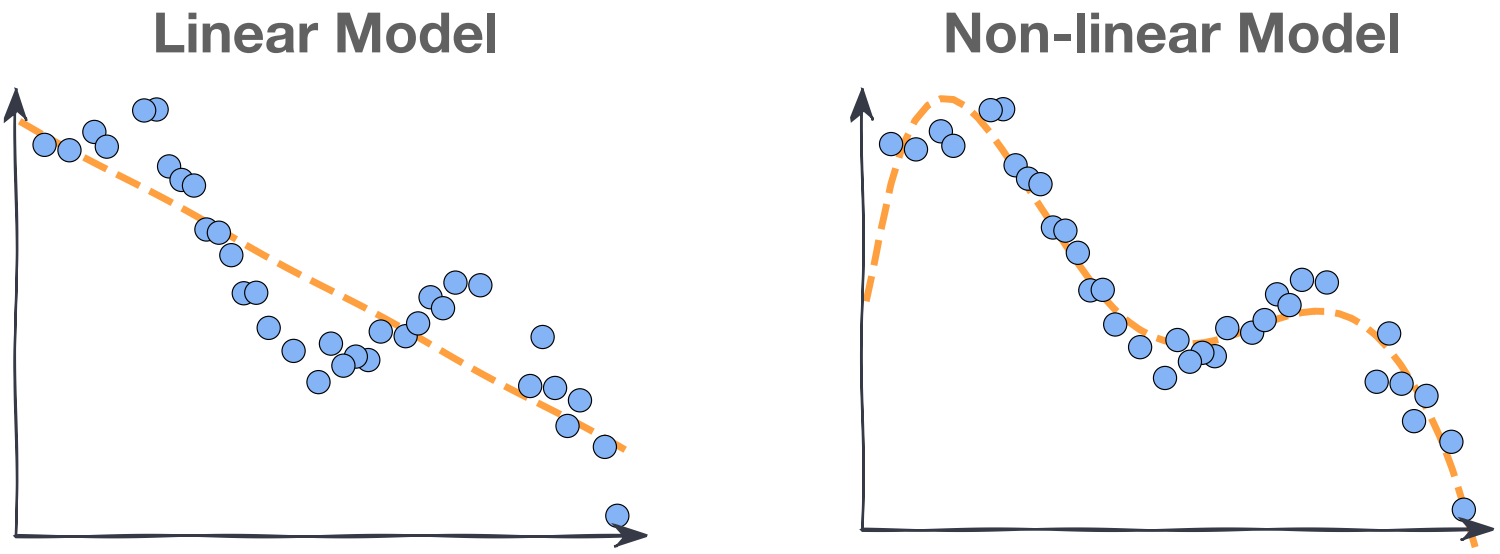
Polynomial Regression

Bookmark this page

Fitting non-linear data

Multi-linear models can fit large datasets with many predictors. However the relationship between predictor and target isn't always linear.

We want a model: $y = f_\beta(x)$ where f is a non-linear function and β is a vector of the parameters of f.



Polynomial Regression

The simplest non-linear model we can consider, for a response Y and a predictor X, is a polynomial model of degree M,

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \ldots + \beta_M x^M$$

Just as in the case of linear regression [with cross terms], polynomial regression is a special case of linear regression - we treat each x^M as a separate predictor. Thus, we can write the design matrix as:

$$X = \begin{pmatrix} 1 & x_1^1 & \ldots & x_1^M \\ 1 & x_2^1 & \ldots & x_2^M \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \ldots & x_n^M \end{pmatrix}$$

This looks a lot like multi-linear regression, where the predictors are powers of x!

Multi-Regression

$$X = \begin{pmatrix} 1 & x_{1,1} & \ldots & x_{1,J} \\ 1 & x_{2,1} & \ldots & x_{2,J} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \ldots & x_{n,J} \end{pmatrix}$$

Model Training

Given a dataset $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, to find the optimal polynomial model:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \ldots + \beta_M x^M$$

1. We transform the data by adding new predictors:

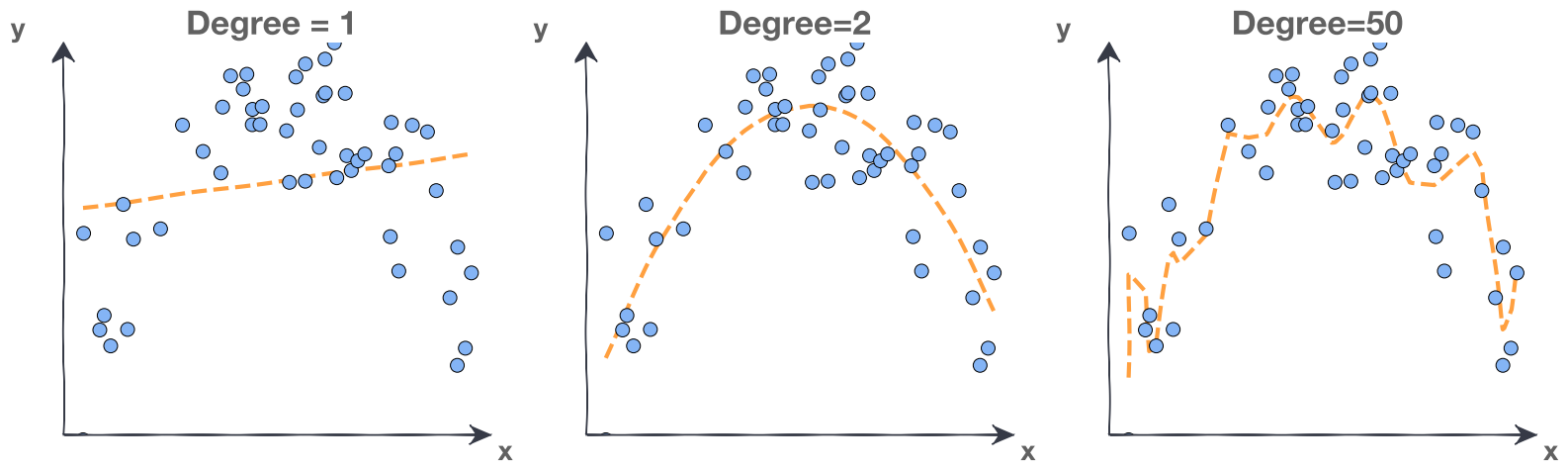
$$\tilde{x} = [1, \tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_M,]$$

2. We find the parameters (β_0, β_1) by minimizing the MSE using vector calculus as in multi-linear regression:

$$\hat{\beta} = (\tilde{X}^T \tilde{X})^{-1} \tilde{X}^T y$$

Polynomial Regression Example

Fitting a polynomial model requires choosing a degree. We want to use a degree high enough to be able to fit to the complexities of our data. However, if we use a polynomial degree too high *we will overfit to our training data*. This means that the model has fit to the noise in the training data and predictions will not generalize to new data. Consider the following plots:



Degree 1 is **underfitting**. The degree is too low, the model cannot fit the trend. The line cuts directly through the center of the data without following the curve at all. Note that using a polynomial degree of 1 is equivalent to just using simple linear regression without polynomials.

Degree 50 is **overfitting**. The degree is too high, the model fits all the noisy data points. The line is jagged as the model predictions overcompensate for small changes in the data.

Degree 2 is a good fit for this data. We want a model that fits the trend and ignores the noise. The line flows smoothly through the approximate center of the data points.

Feature Scaling

Do we need to scale out features for polynomial regression?

Linear regression, $\boldsymbol{Y = X\beta}$, is invariant under scaling. If $\boldsymbol{X}$ is multiplied by some number $\boldsymbol{\lambda}$, then β will be scaled by $\frac{1}{\lambda}$ and the $\boldsymbol{MSE}$ will be identical.

However, if the range of $\boldsymbol{X}$ is small or large, then we run into troubles. Consider a polynomial degree of 20 and the maximum or minimum value of any predictor is large or small. Those numbers to the 20th power will be problematic. They may even be bigger than the biggest number Python can handle.

It is always a good idea to scale $\boldsymbol{X}$ when considering a polynomial regression:

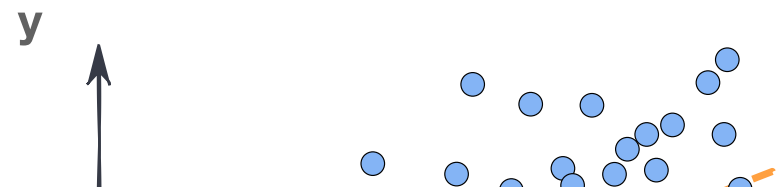
$$X^{stand} = \frac{X - \overline{X}}{\sigma_X}, X^{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

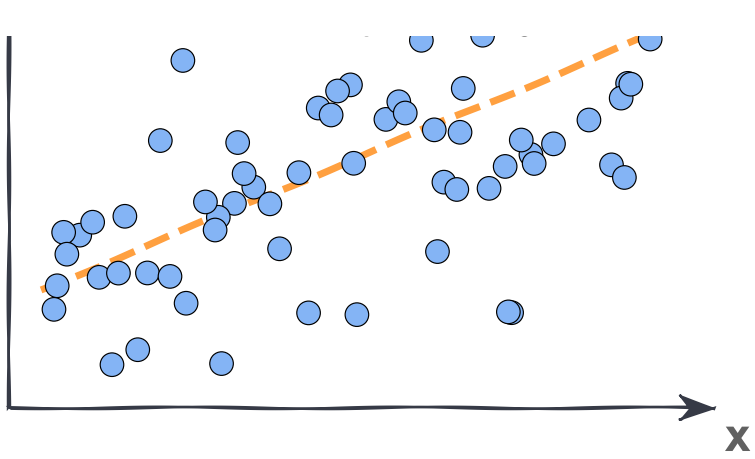
For standardization, note that we can use sklearn's StandardScaler() to do this.

For normalization, note that we can use sklearn's MinMaxScaler() to do this.

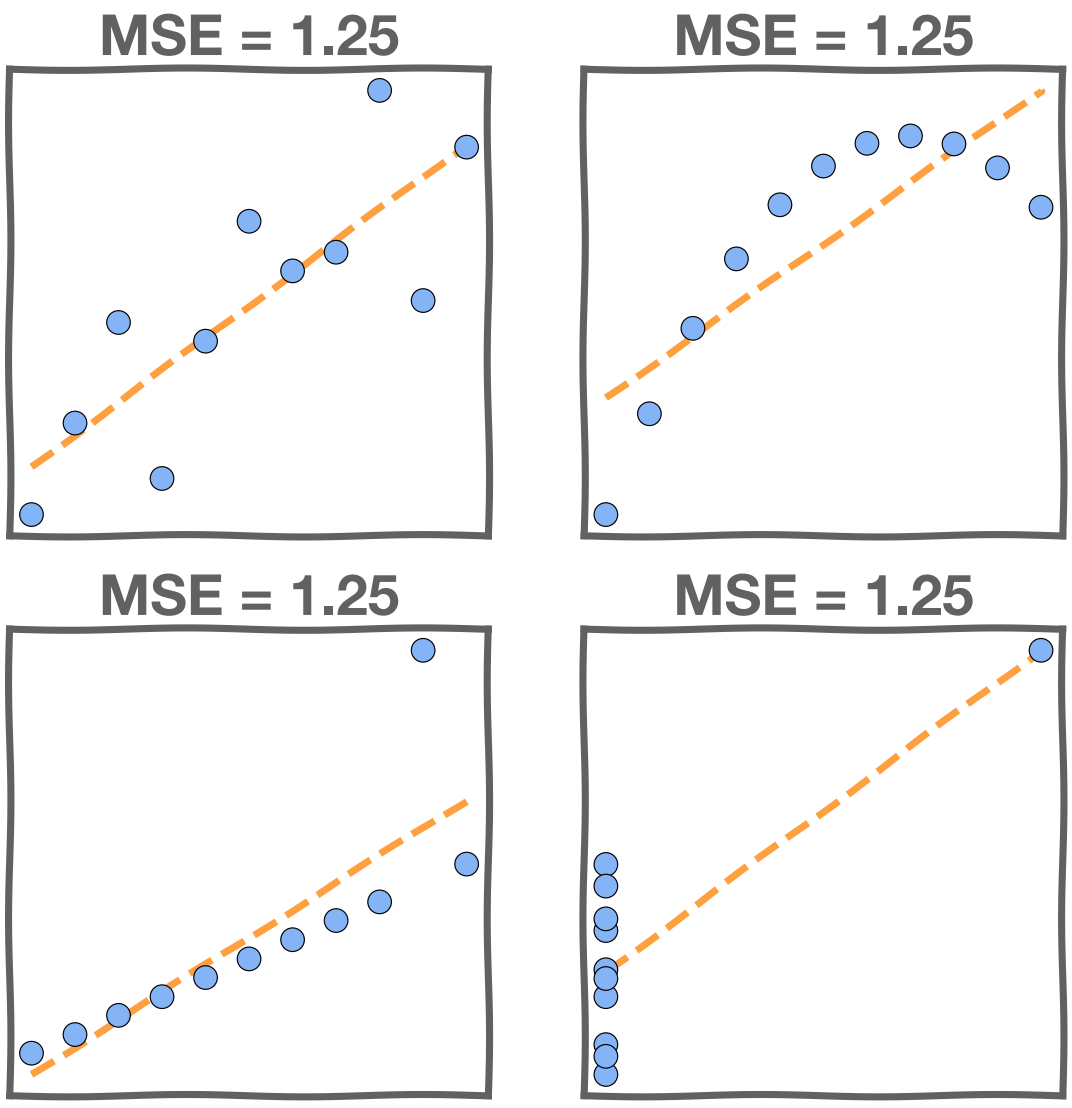
Evaluation: Training Error

Just because we found the model that minimizes the squared error it doesn't mean that it's a good model. We investigate the $\boldsymbol{R^2}$, but we also need to look at the data. For instance, in this first graph the MSE is high due to noise in the data.



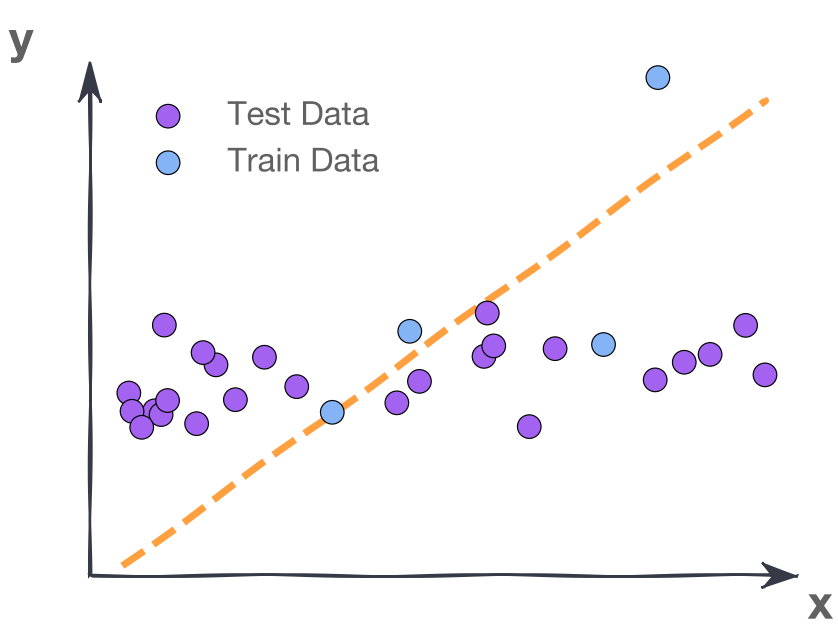


In these next graphs the MSE is high in all four models, but the models are not equal! Examining the data directly will tell us as much about our model's fit as the MSE.



Evaluation: Test Error

We need to evaluate the fitted model on our new data, data that the model did not train on. This is the **test data**.



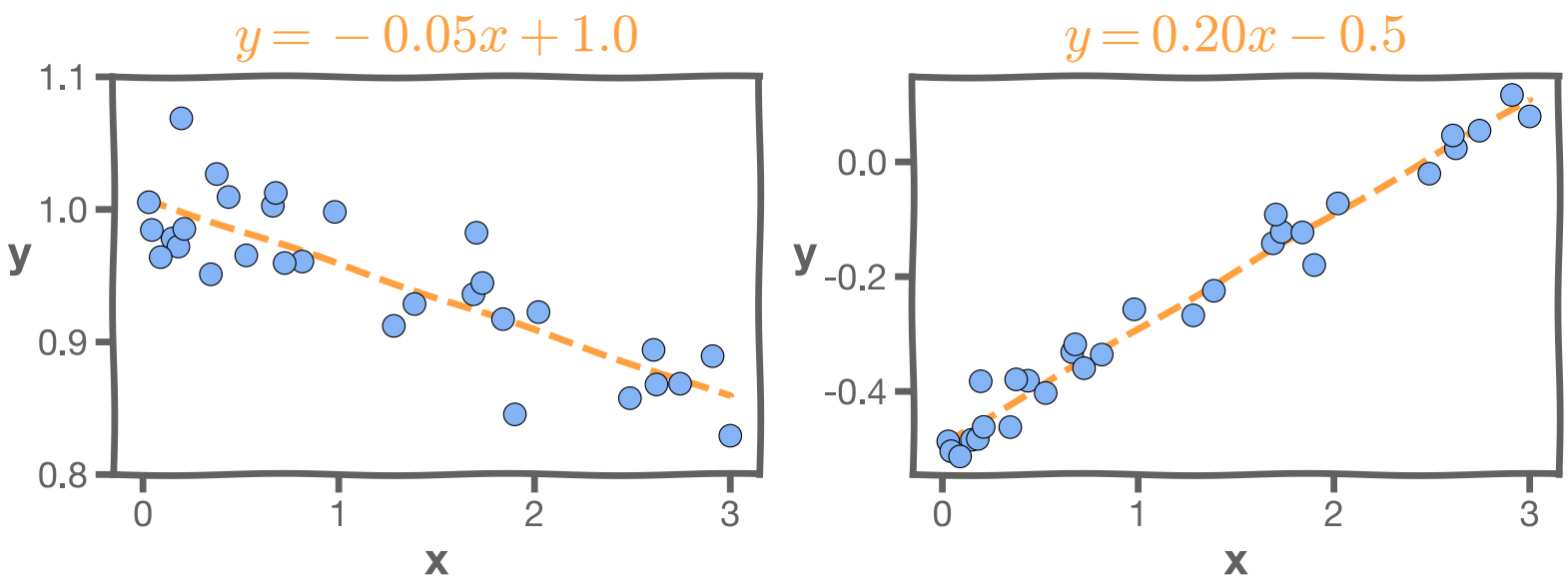
The training MSE here is 2.0 where the test MSE is 12.3. The training data contains a strange point – an outlier – which confuses the model.

Fitting to meaningless patterns in the training is called overfitting.

Evaluation: Model Interpretation

Evaluation: Model interpretation

For linear models it's important to interpret the parameters



In the left-hand graph, the MSE of the model is very small. But the slope is -0.05. That means the larger the budget the less the sales, which seems unlikely.

In the right-hand graph, the MSE is very small, but the intercept is -0.5, which means that for very small budget we will have negative sales, which is impossible.

Discussion Board (External resource)

Haga clic en Aceptar para que su nombre de usuario y dirección de correo electrónico se envíen a una aplicación de terceros.

Aceptar

‹ Previous Next ›

edX

About
Affiliates
edX for Business
Open edX
Careers
News

Legal

Terms of Service & Honor Code
Privacy Policy
Accessibility Policy
Trademark Policy
Sitemap
Cookie Policy
Your Privacy Choices

Connect

Idea Hub
Contact Us
Help Center
Security

Calculator

Media Kit

深圳市恒宇博科技有限公司 粤ICP备17044299号-2

Calculator