CS M148 –

Data Science Fundamentals

Lecture #4: Multi and poly regression, Model selection

Baharan Mirzasoleiman

UCLA Computer Science

jupyter nbconvert --to webpdf --allow-chromium-download <path>

Announcements

Waitlist

- The waitlist is enrolled!
- PTEs: please stay after the class

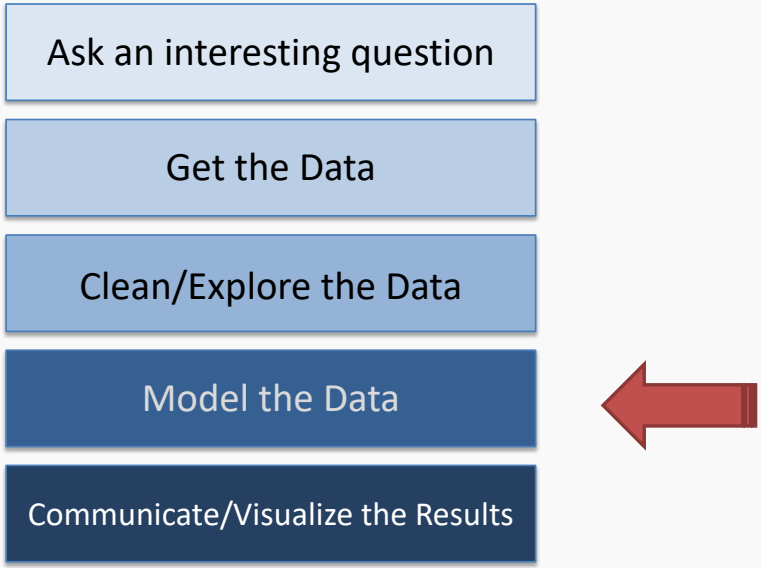
Project 1 is posted

- Due Wed Jan 19, 2pm

Let’s quickly review what we saw last time

Ready to Model the Data!

The Data Science Process



Previous Lecture Review

Part A: Statistical Modeling

k-Nearest Neighbors (kNN)

Part B: Model Fitness

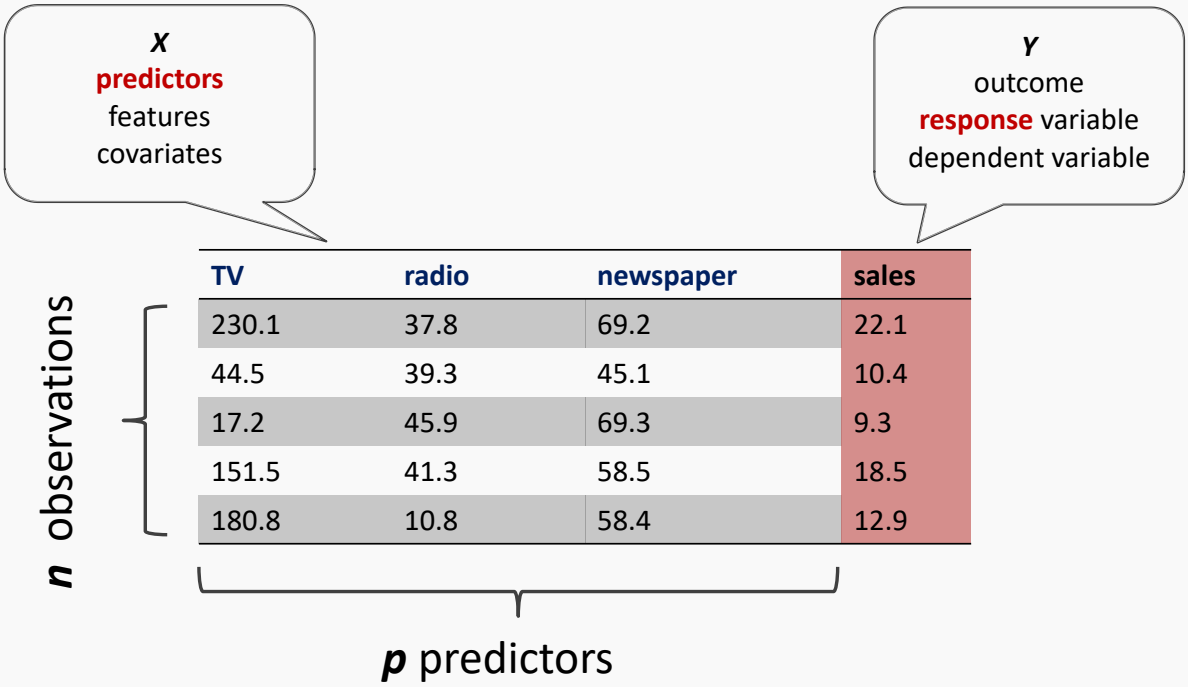
How does the model perform predicting?

Part B: Comparison of Two Models

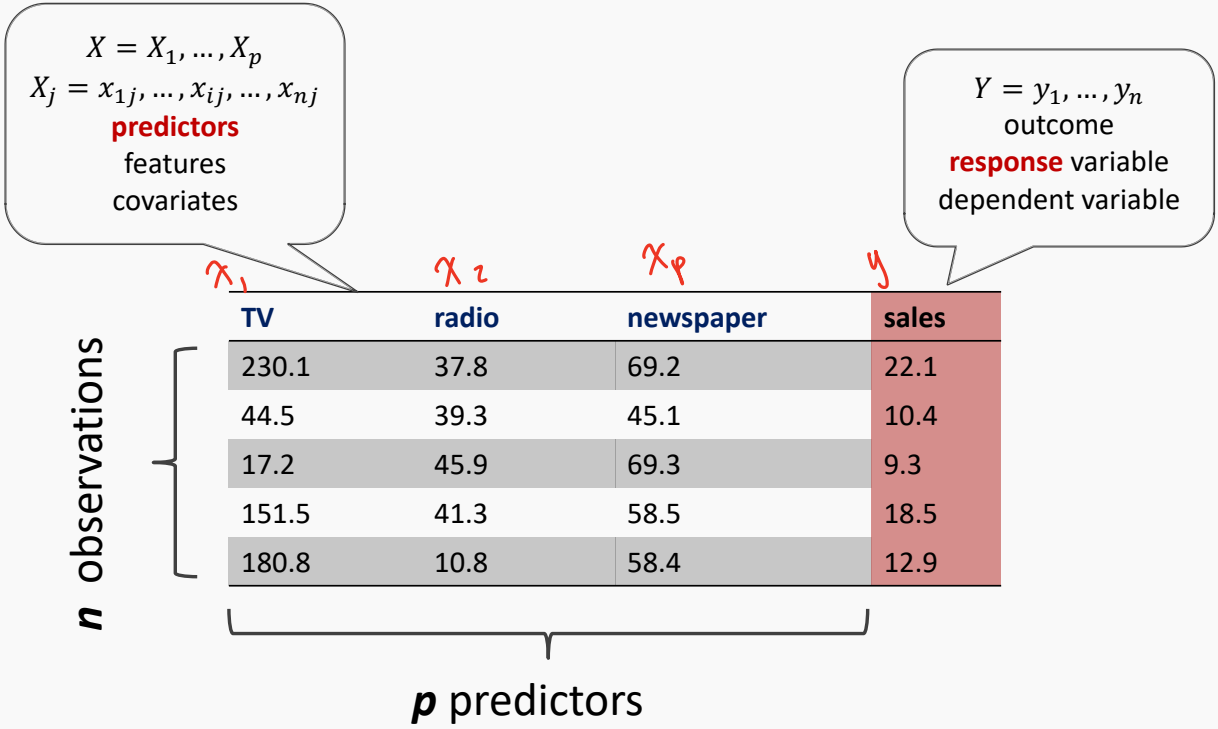
How do we choose from two different models?

Part C: Linear Models

Response vs. Predictor Variables



Response vs. Predictor Variables



Statistical Model

True vs. Statistical Model

We will assume that the response variable, Y, relates to the predictors, X, through some unknown function expressed generally as:

$$Y = f(X) + \varepsilon$$

Here, f is the unknown function expressing an underlying rule for relating Y to X, ε is the random amount (unrelated to X) that Y differs from the rule $f(X)$.

A ***statistical model*** is any algorithm that estimates f. We denote the estimated function as $\hat{f}$.

Prediction vs. Estimation

For some problems, what's important is obtaining $\hat{f}$, our estimate of f. These are called ***inference*** problems.

When we use a set of measurements, $(x_{i,1}, \ldots, x_{i,p})$ to predict a value for the response variable, we denote the ***predicted*** value by:

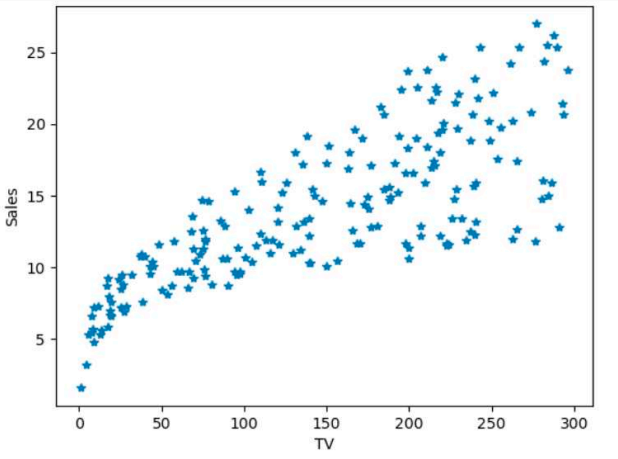
$$\hat{y}_i = \hat{f}(x_{i,1}, \ldots, x_{i,p}).$$

For some problems, we don't care about the specific form of $\hat{f}$, we just want to make our predictions $\hat{y}$'s as close to the observed values y's as possible. These are called ***prediction problems.***

Example: predicting sales

Motivation: Predict Sales

Build a model to **predict** sales based on TV budget

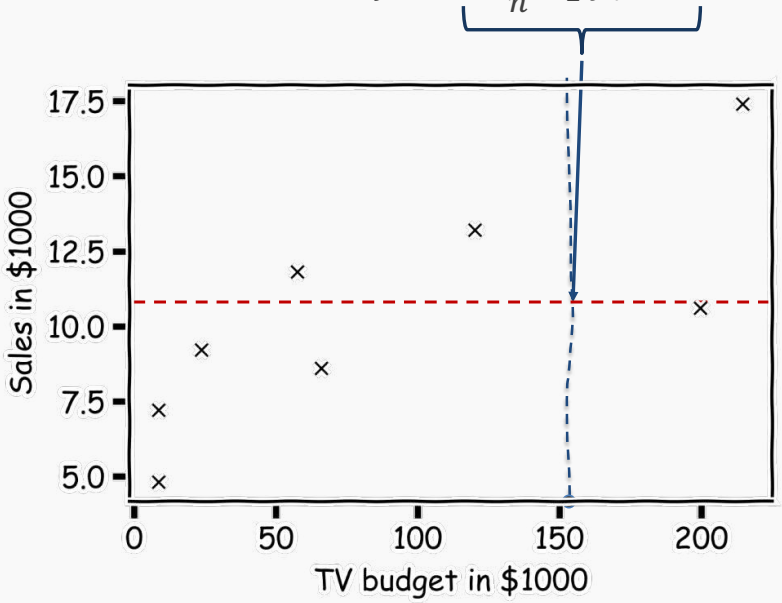


The response, y, is the sales

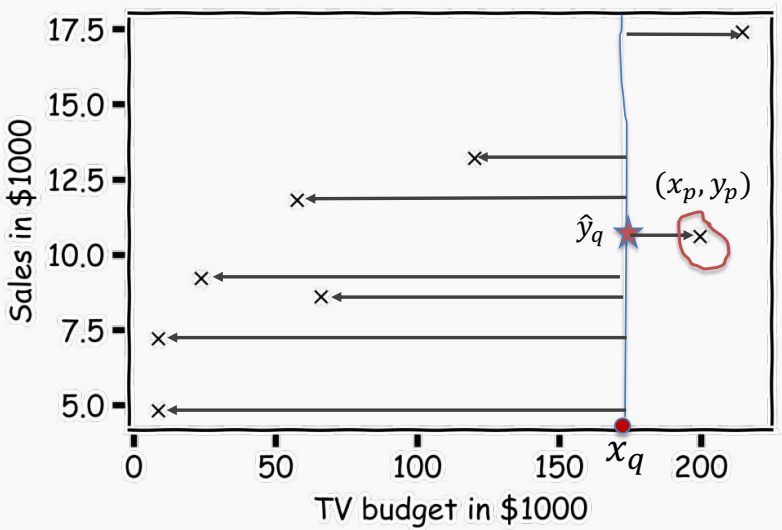
The predictor, x, is TV budget

Statistical Model

Simple idea is to take the mean of all *y's*, $\hat{f}(x) = \frac{1}{n}\sum_1^n y_i$



Simple Prediction Model



What is $\hat{y}_q$ at some x_q ?

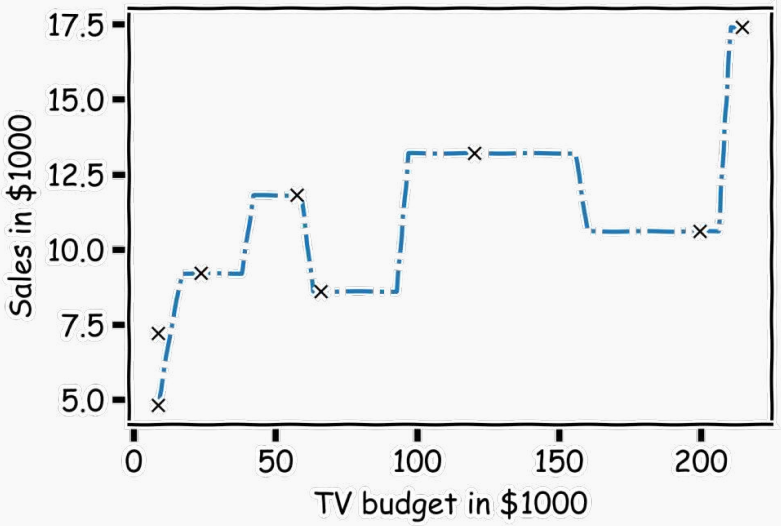
Find distances to all other points $D(x_q, x_i)$

Find the nearest neighbor, (x_p, y_p)

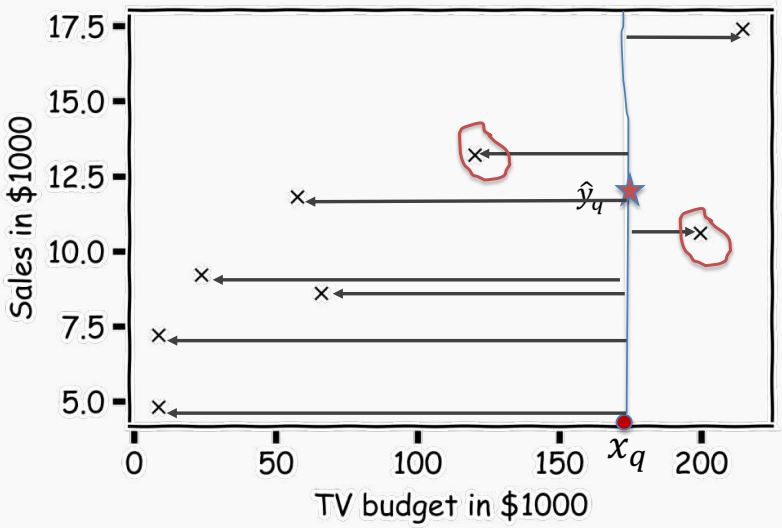
Predict $\hat{y}_q = y_p$

Simple Prediction Model

Do the same for "all" $x's$



Extend the Prediction Model



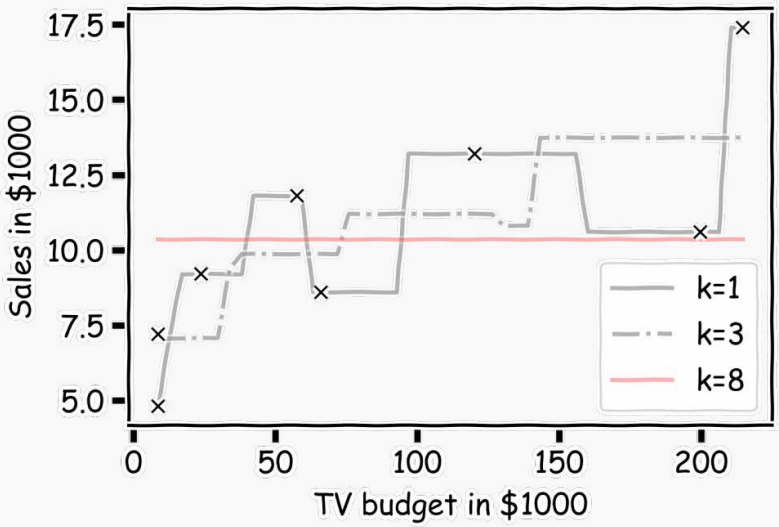
What is $\hat{y}_q$ at some x_q ?

Find distances to all other points $D(x_q, x_i)$

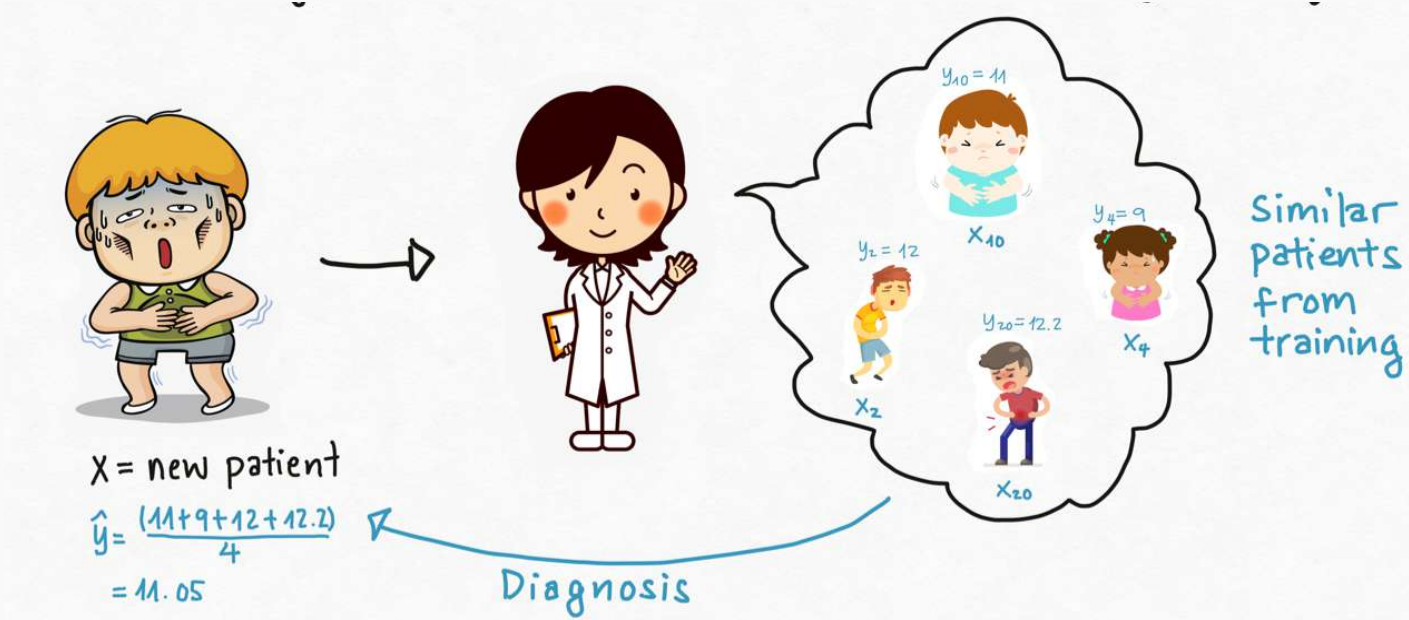
Find the k-nearest neighbors, $x_{q_1}, \ldots, x_{q_k}$

Predict $\hat{y}_q = \frac{1}{k}\sum_i^k y_{q_i}$

Simple Prediction Models



k-Nearest Neighbors – kNN



k-Nearest Neighbors – kNN

The very human way of decision making by similar examples. kNN is a non-parametric learning algorithm.

The k-Nearest Neighbor Algorithm:

Given a dataset $D = \{(x^{(1)}, y^{(1)}), \dots, (x^{(N)}, y^{(N)})\}$. For every new X:

1. Find the k-number of observations in D most similar to X:

$$\{(x^{(n_1)}, y^{(n_1)}), \dots, (x^{(n_k)}, y^{(n_k)})\}$$

   These are called the k-nearest neighbors of x

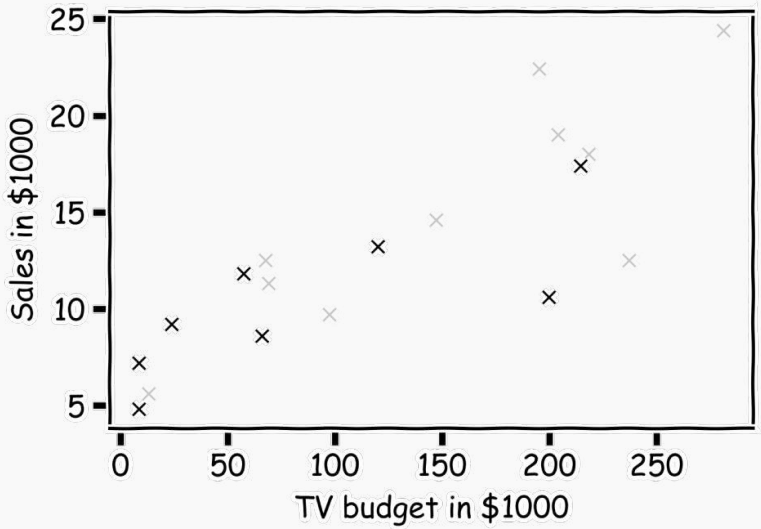
2. Average the output of the k-nearest neighbors of x

$$\hat{y} = \frac{1}{K} \sum_{k=1}^{K} y^{(n_k)}$$

Error Evaluation

Error Evaluation

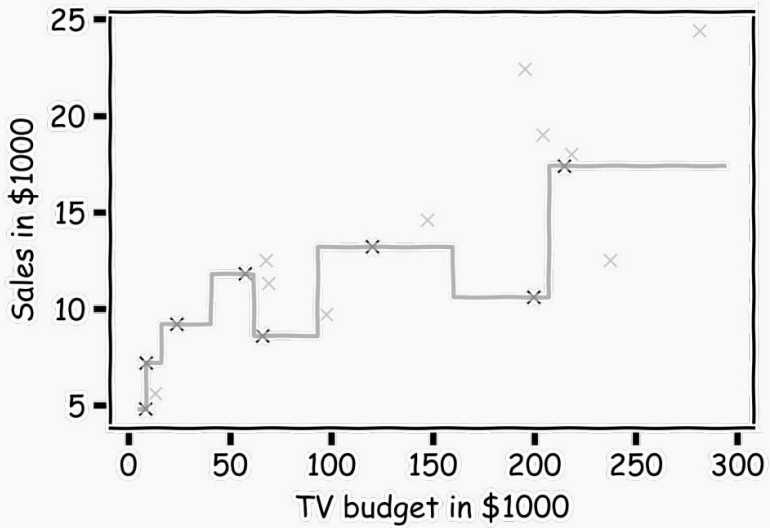
Hide some of the data from the model. This is called **train-test** split.



We use the train set to estimate $\hat{y}$, and the test set to evaluate the model.

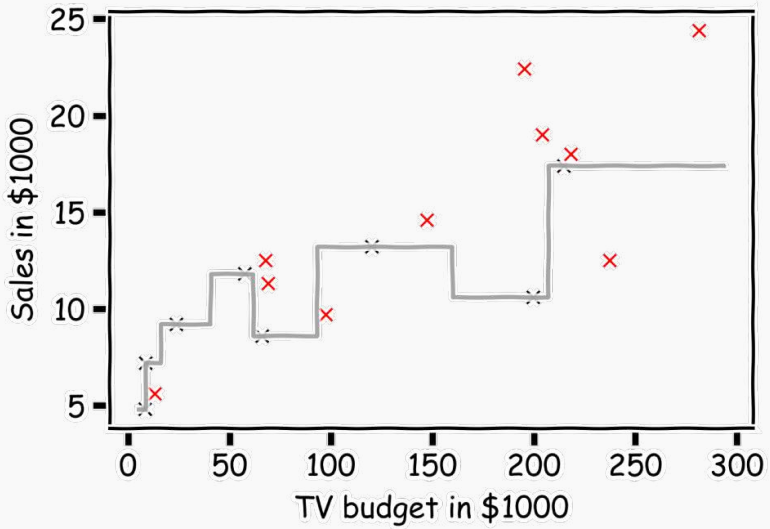
Error Evaluation

Estimate $\hat{y}$ for *k=1* .



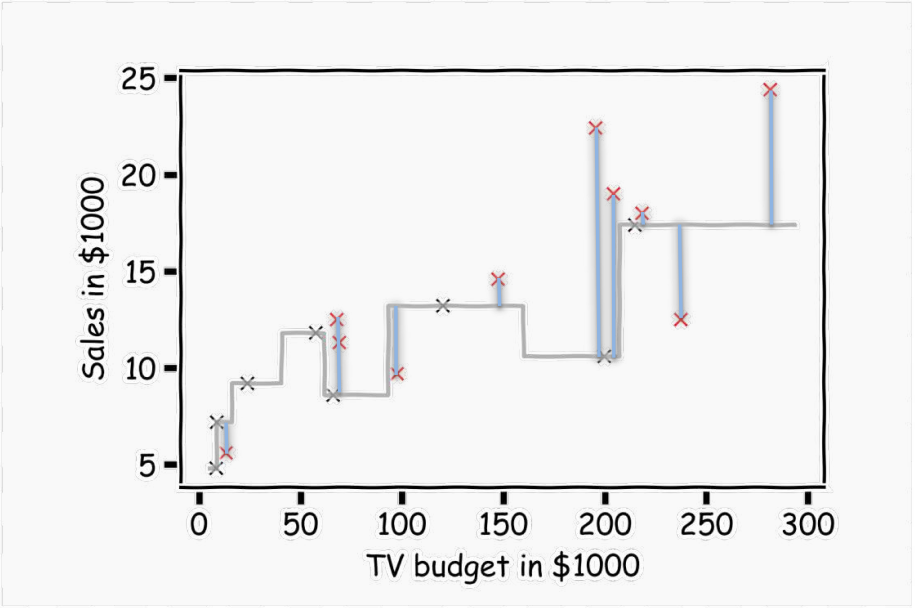
Error Evaluation

Now, we look at the data we have not used, the **test data** (red crosses).



Error Evaluation

Calculate the **residuals** $(y_i - \hat{y}_i)$.



Error Evaluation

In order to quantify how well a model performs, we aggregate the errors and we call that the ***loss*** or ***error*** *or* ***cost function.***

A common loss function for quantitative outcomes is the **Mean Squared Error (MSE):**

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2$$

Note: Loss and cost function refer to the same thing. Cost usually refers to the total loss where loss refers to a single training point.

Error Evaluation

Caution: The MSE is by no means the only valid (or the best) loss function!

1. Max Absolute Error
2. Mean Absolute Error
3. Mean Squared Error

We will motivate MSE when we introduce probabilistic modeling.

Note: The square **R**oot of the **M**ean of the **S**quared **E**rrors (RMSE) is also commonly used.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2}$$

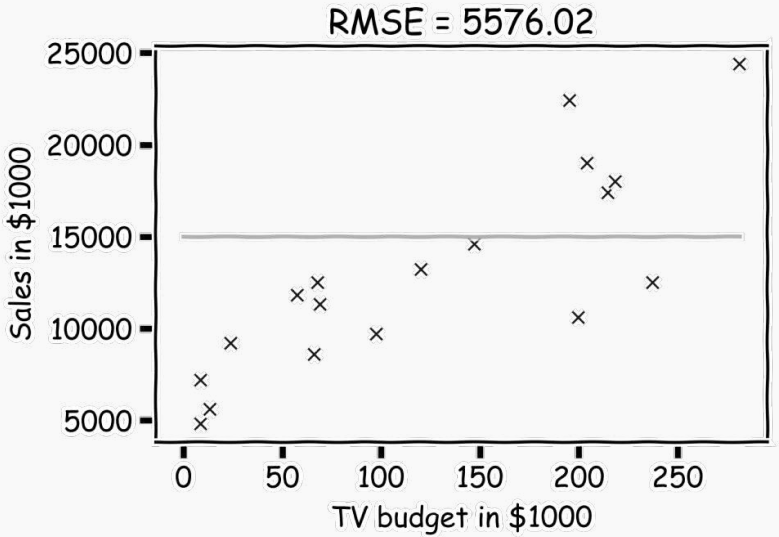
Model Comparison

Compare the RMSEs

Model Fitness

Model fitness

It is better if we compare it to something.



We will use the simplest model:

$$\hat{y} = \bar{y} = \frac{1}{n}\sum_i y_i$$

as the worst possible model and

$$\hat{y_i} = y_i$$

as the best possible model.

R-squared

$$R^2 = 1 - \frac{\sum_i (\hat{y}_i - y_i)^2}{\sum_i (\bar{y} - y_i)^2}$$

- If our model is as good as the mean value, $\bar{y}$, then $R^2 = 0$
- If our model is perfect then $R^2 = 1$
- R^2 can be negative if the model is worst than the average. This can happen when we evaluate the model on the test set.

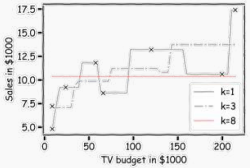
Linear Models

Why do we like them?

Interpretation!

Linear Models

Note that in building our kNN model for prediction, we did not compute a closed form for $\hat{f}$.



What if we ask the question:

"how much more sales do we expect if we double the TV advertising budget?"

Alternatively, we can build a model by first assuming a simple form of f:

$$f(x) = \beta_0 + \beta_1 X$$

… then it follows that our estimate is:

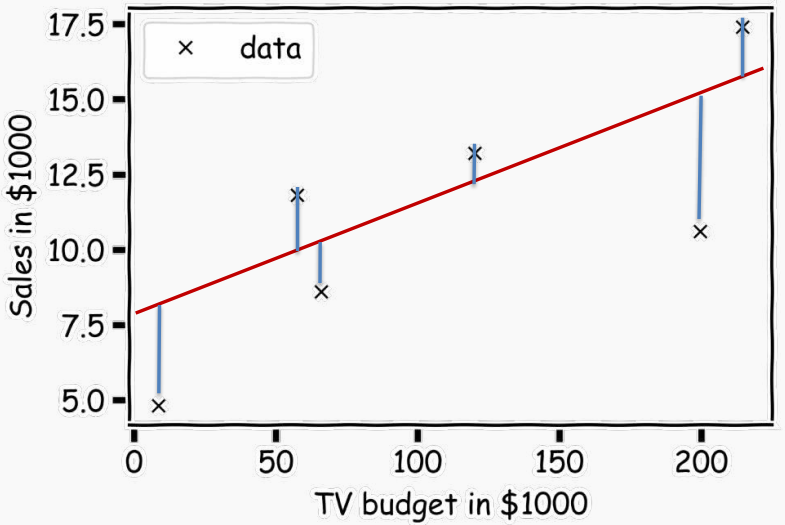
$$\widehat{Y} = \widehat{f}(X) = \widehat{\beta_1} X + \widehat{\beta_0}$$

where $\hat{\beta}_1$ and $\hat{\beta}_0$ are **estimates** of β_1 and β_0 respectively, that we compute using observations.

Estimate of the regression coefficients (cont)

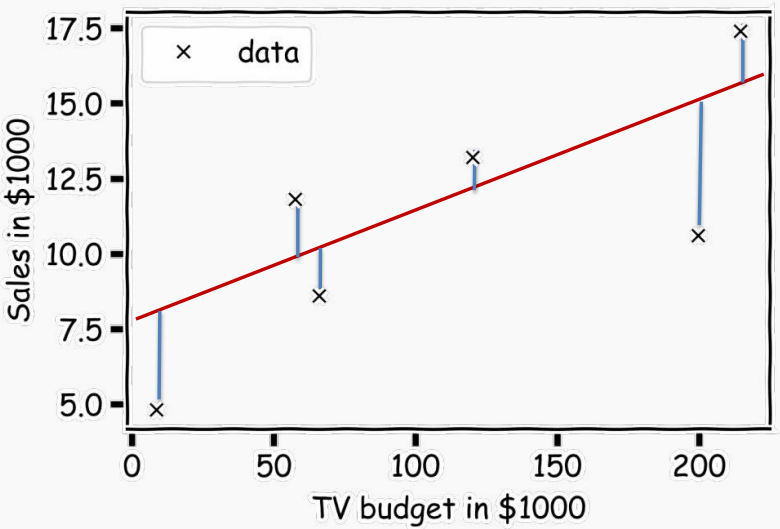
Question: Which line is the best?

For each observation (x_n, y_n), the absolute residual is $r_i = |y_i - \hat{y}_i|$.



Loss Function: Aggregate Residuals

How do we aggregate residuals across the entire dateset?



1. Max Absolute Error
2. Mean Absolute Error
3. Mean Squared Error

Estimate of the regression coefficients (cont)

Again we use MSE as our **loss function**,

test y

predictor for our model

$$L(\beta_0, \beta_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}[y_i - (\beta_1 X + \beta_0)]^2 .$$

We choose $\hat{\beta}_1$ and $\hat{\beta}_0$ in order to minimize the predictive errors made by our model, i.e. minimize our loss function.

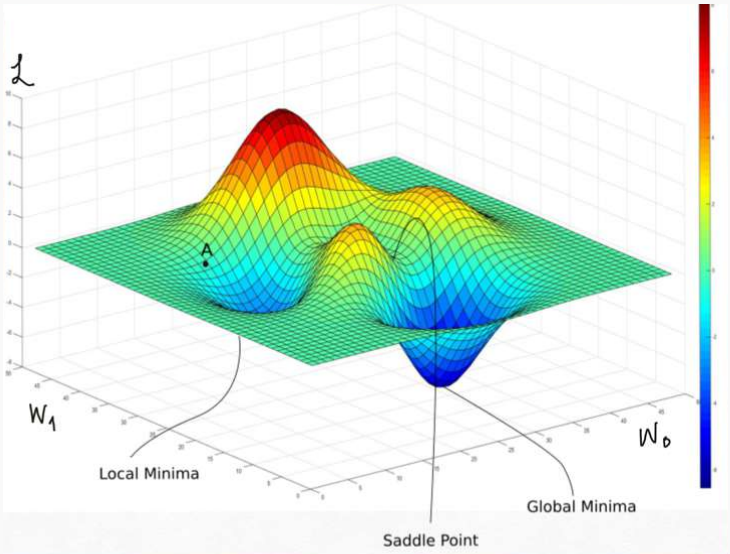
Then the optimal values for $\hat{\beta}_0$ and $\hat{\beta}_1$ should be:

$$\widehat{\beta_0}, \widehat{\beta_1} = \underset{\beta_0, \beta_1}{\operatorname{argmin}} L(\beta_0, \beta_1).$$

WE CALL THIS **FITTING** OR **TRAINING** THE MODEL

Optimization

How does one minimize a loss function?



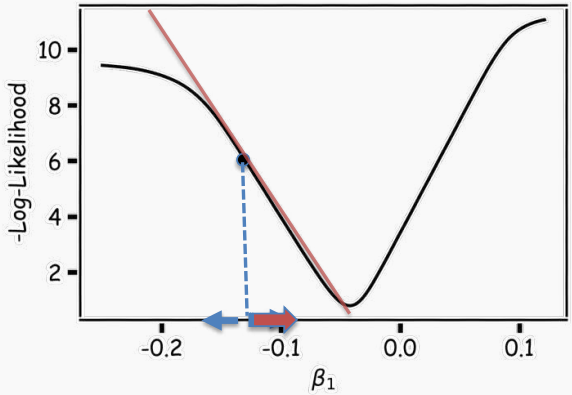
The global minima or maxima of $L(\beta_0, \beta_1)$ must occur at a point where the gradient (slope)

$$\nabla L = \left[\frac{\partial L}{\partial \beta_0}, \frac{\partial L}{\partial \beta_1}\right] = 0$$

- Brute Force: Try every combination
- Exact: Solve the above equation
- Greedy Algorithm: Gradient Descent

works for small studies

Gradient Descent



- Start from a random point
  1. Determine which direction to go to reduce the loss (left or right)
  2. Compute the slope of the function at this point and step to the right if slope is negative or step to the left if slope is positive
  3. Goto to #1

Estimate of the regression coefficients: analytical solution

Take the gradient of the loss function and find the values of $\hat{\beta}_0$ and $\hat{\beta}_1$ where the gradient is zero: $\nabla L = \left[\frac{\partial L}{\partial \beta_0}, \frac{\partial L}{\partial \beta_1}\right]$=0

This does not usually yield to a close form solution. However for linear regression this procedure gives us explicit formulae for $\hat{\beta}_0$ and $\hat{\beta}_1$:

$$\hat{\beta}_1 = \frac{\sum_i (x_i - \overline{x})(y_i - \overline{y})}{\sum_i (x_i - \overline{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

where $\bar{y}$ and $\bar{x}$ are sample means.

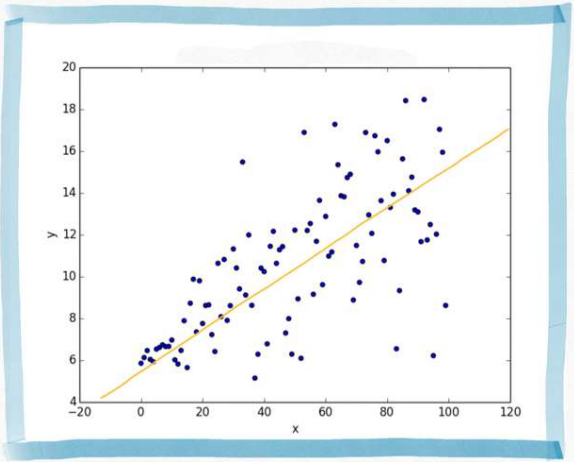
The line:

$$\widehat{Y} = \widehat{\beta}_1 X + \widehat{\beta}_0$$

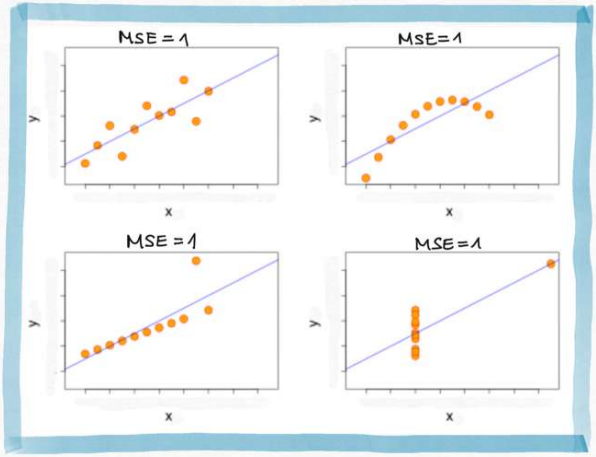
is called the **regression line**.

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 R2 but also:



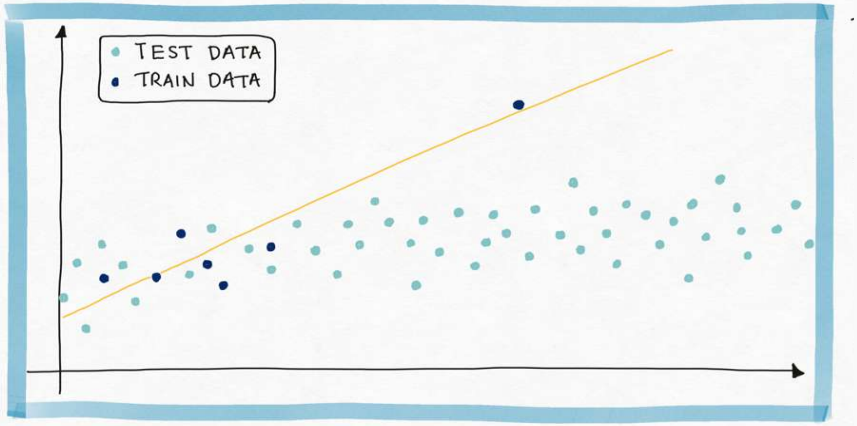
The MSE is high due to noise in the data.



The MSE is high in all four models but the models are not equal.

Evaluation: Test Error

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



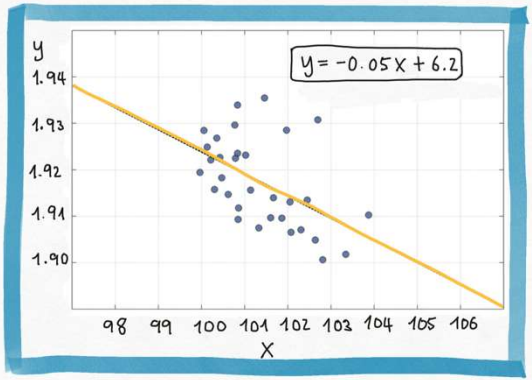
The training MSE here is 2.0 where the test MSE is 12.3.

The training data contains a strange point – an outlier – which confuses the model.

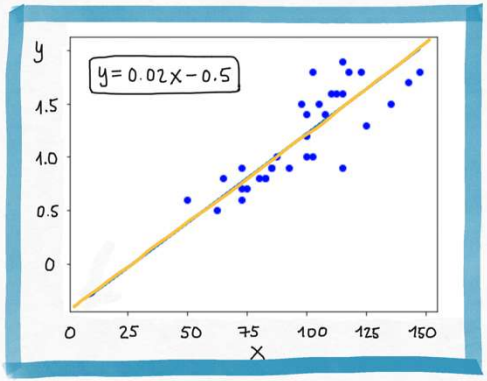
Fitting to meaningless patterns in the training is called **overfitting.**

Evaluation: Model Interpretation

For linear models it's important to interpret the parameters



The MSE of this model is very small. But the slope is -0.05. That means the larger the budget the less the sales.



The MSE is very small but the intercept is -0.5 which means that for very small budget we will have negative sales.

Multi & Poly Regression

Previous Lecture Review

Part A: Multi Regression

How do we interpret the model?

Residual analysis: is the linear model good enough?

Part B: Poly Regression

What if the relationship between predictor and target isn't linear?

Part C: Model Selection

How do we decide on the complexity of the model?

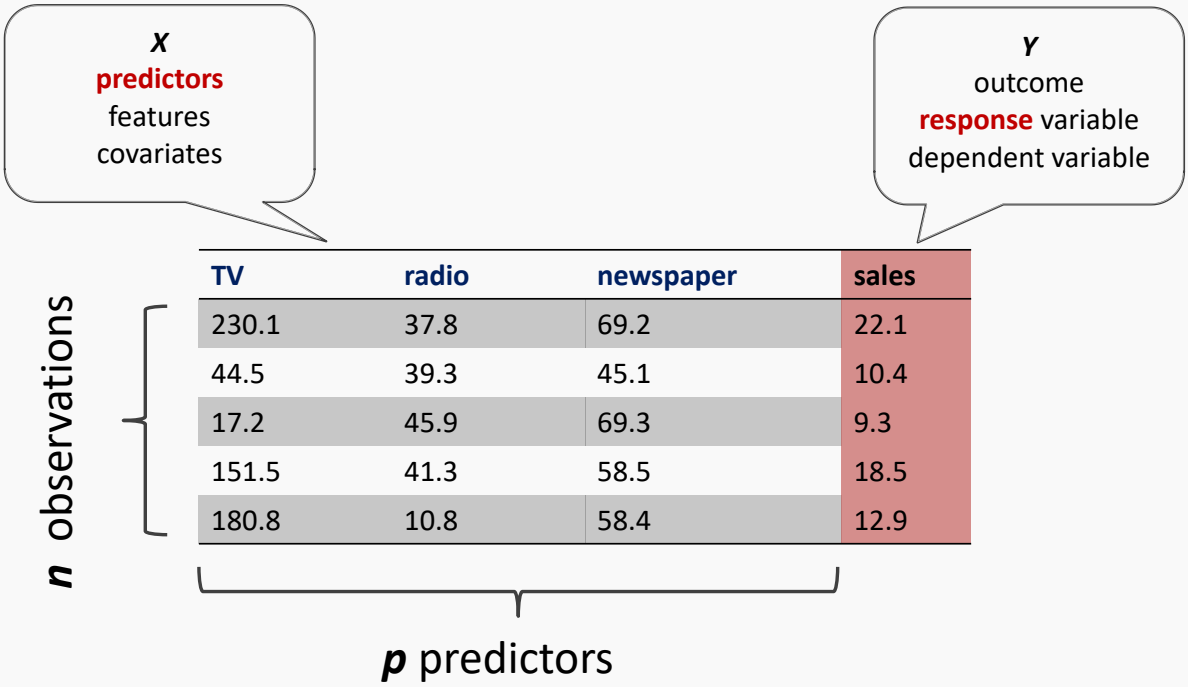
Multiple Linear Regression

If you have to guess someone's height, would you rather be told

- Their weight, only
- Their weight and gender
- Their weight, gender, and income
- Their weight, gender, income, and favorite number

Of course, you'd always want as much data about a person as possible. Even though height and favorite number may not be strongly related, at worst you could just ignore the information on favorite number. We want our models to be able to take in lots of data as they make their predictions.

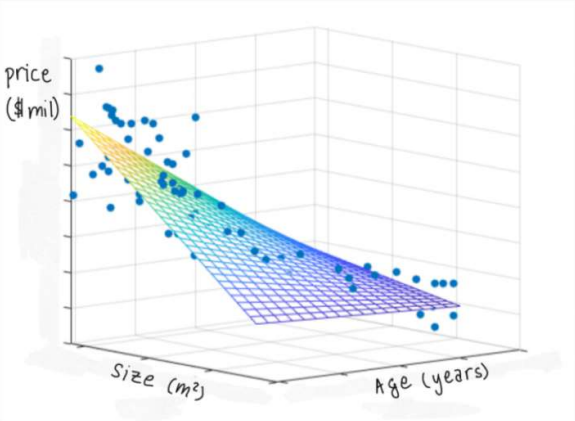
Response vs. Predictor Variables



Multilinear Models

In practice, it is unlikely that any response variable Y depends solely on one predictor *x*. Rather, we expect that is a function of multiple predictors $f(X_1, \dots, X_J)$. Using the notation we introduced in last lecture,

$$Y = y_1, \dots, y_n, \quad X = X_1, \dots, X_J \text{ and } X_j = x_{1j}, \dots, x_{ij}, \dots, x_{nj}$$



In this case, we can still assume a simple form for f -a multilinear form:

$$Y = f(X_1, \dots, X_J) + \epsilon = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_J X_J$$

Hence, $\hat{f}$, has the form

$$\hat{Y} = \hat{f}(X_1, \dots, X_J) + \epsilon = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \dots + \hat{\beta}_J X_J$$

Multiple Linear Regression

Given a set of observations,

$$\{(x_{1,1}, \ldots, x_{1,J}, y_1), \ldots (x_{n,1}, \ldots, x_{n,J}, y_n)\},$$

$y = f(x)$

$\downarrow$

$\beta_0 + \beta_1 x_1$

the data and the model can be expressed in vector/matrix notation,

1 predictor

x_1

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_y \end{pmatrix}, \qquad \mathbf{X} = \begin{pmatrix} 1 & x_{1,1} & \ldots & x_{1,J} \\ 1 & x_{2,1} & \ldots & x_{2,J} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \ldots & x_{n,J} \end{pmatrix}, \qquad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_J \end{pmatrix},$$

J predictors

Multilinear Model, example

For our data

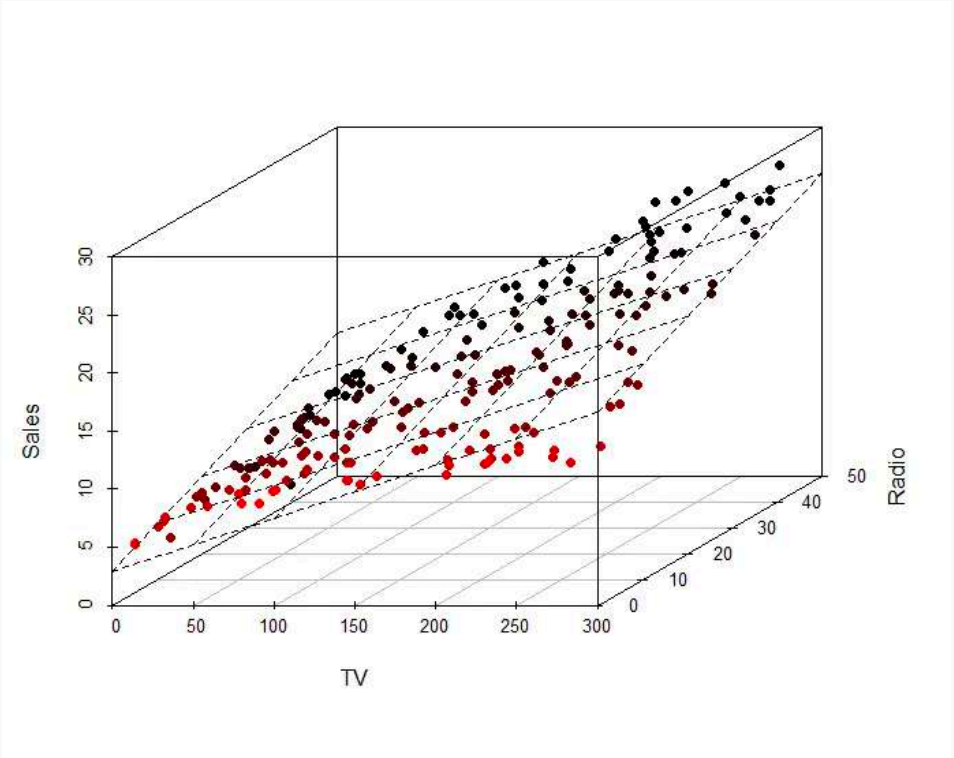
$$\text{Sales} = \beta_0 + \beta_1 \times TV + \beta_2 \times Radio + \beta_3 \times Newspaper + \epsilon$$

In linear algebra notation

$$\boldsymbol{Y} = \begin{pmatrix} Sales_1 \\ \vdots \\ Sales_n \end{pmatrix}, \boldsymbol{X} = \begin{pmatrix} 1 & TV_1 & Radio_1 & News_1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & TV_n. & Radio_n & News_n \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_3 \end{pmatrix}$$

$$Sales_1 = 1 \quad TV_1 \quad Radio_1 \quad News_1 \times \begin{matrix} \beta_0 \\ \vdots \\ \beta_3 \end{matrix}$$

Multilinear Model, example



Multiple Linear Regression

The model takes a simple algebraic form:

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon$$

Thus, the MSE can be expressed in vector notation as

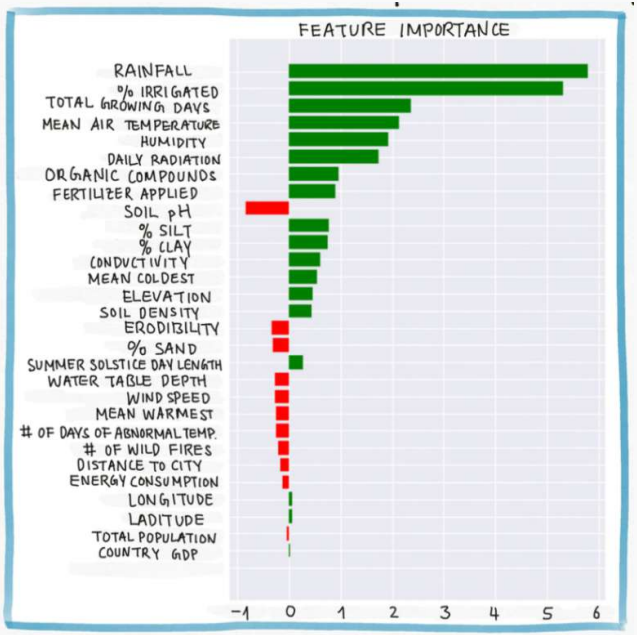
$$\mathrm{MSE}(\beta) = \frac{1}{\mathrm{n}} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|^2$$

Minimizing the MSE using vector calculus yields,

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{Y} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}}\ \mathrm{MSE}(\boldsymbol{\beta}).$$

Interpreting multi-linear regression

For linear models, it is easy to interpret the model parameters.



When we have a large number of predictors: $X_1, \dots, X_J$ there will be a large number of model parameters, $\beta_1, \dots, \beta_J$.

Looking at the values of β's is impractical, so we visualize these values in a feature importance graph.

The feature importance graph shows which predictors has the most impact on the model's prediction.

Qualitative Predictors

So far, we have assumed that all variables are quantitative. But in practice, often some predictors are **qualitative**.

Example: The Credit data set contains information about balance, age, cards, education, income, limit , and rating for a number of potential customers.

binary

one-hot [1 0 0] [0 1 0]

Income	Limit	Rating	Cards	Age	Education	Gender	Student	Married	Ethnicity	Balance
14.890	3606	283	2	34	11	Male	No	Yes	Caucasian	333
106.02	6645	483	3	82	15	Female	Yes	Yes	Asian	903
104.59	7075	514	4	71	11	Male	No	No	Asian	580
148.92	9504	681	3	36	11	Female	No	No	Asian	964
55.882	4897	357	2	68	16	Male	No	Yes	Caucasian	331

Qualitative Predictors

If the predictor takes only two values, then we create an **indicator** or **dummy variable** that takes on two possible numerical values.

For example for the gender, we create a new variable:

$$x_i = \begin{cases} 1 & \text{if } i\,\text{th person is female} \\ 0 & \text{if } i\,\text{th person is male} \end{cases}$$

We then use this variable as a predictor in the regression equation.

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i\,\text{th person is female} \\ \beta_0 + \epsilon_i & \text{if } i\,\text{th person is male} \end{cases}$$

Qualitative Predictors

Question: What is interpretation of β_0 and β_1?

β_1 = avg. diff. b/w females & males

Qualitative Predictors

Question: What is interpretation of β_0 and β_1?

- β_0 is the average credit card balance among males,
- $\beta_0 + \beta_1$ is the average credit card balance among females,
- and β_1 the average difference in credit card balance between females and males.

Example: Calculate β_0 and β_1 for the Credit data.

You should find $\beta_0 \sim \$509, \beta_1 \sim \$19$

More than two levels: One hot encoding

Often, the qualitative predictor takes more than two values (e.g. ethnicity in the credit data).

In this situation, a single dummy variable cannot represent all possible values.

We create additional dummy variable as:

$$x_{i,1} = \begin{cases} 1 & \text{if } i\,\text{th person is Asian} \\ 0 & \text{if } i\,\text{th person is not Asian} \end{cases}$$

$$x_{i,2} = \begin{cases} 1 & \text{if } i\,\text{th person is Caucasian} \\ 0 & \text{if } i\,\text{th person is not Caucasian} \end{cases}$$

More than two levels: One hot encoding

We then use these variables as predictors, the regression equation becomes:

$$y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i\text{th person is Asian} \\ \beta_0 + \beta_2 + \epsilon_i & \text{if } i\text{th person is Caucasian} \\ \beta_0 + \epsilon_i & \text{if } i\text{th person is AfricanAmerican} \end{cases}$$

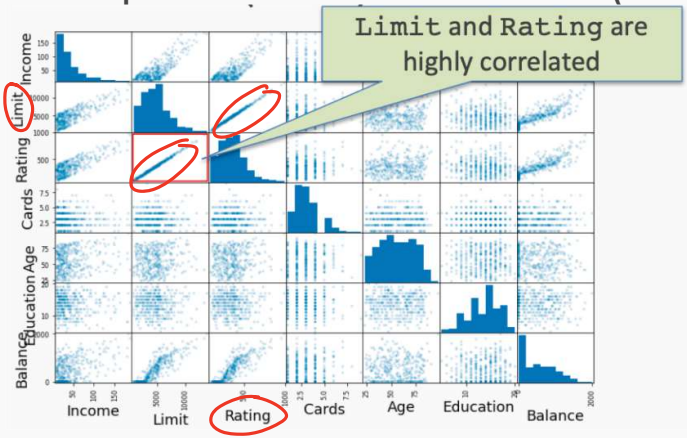
Question: What is the interpretation of $\beta_0, \beta_1, \beta_2$?

β_0 : avg. balance for A.A.

β_2 : diff. btw avg. balance btw A.A. & Caucasian

Collinearity

Collinearity and multicollinearity refers to the case in which two or more predictors are correlated (related). predictor will be good



	Columns	Coefficients
0	Income	-7.802001
1	Limit	0.193077
2	Rating	1.102269
3	Cards	17.923274
4	Age	-0.634677
5	Education	-1.115028
6	Gender	10.406651
7	Student	426.469192
8	Married	-7.019100

	Columns	Coefficients
0	Income	-7.770915
1	Rating	3.976119
2	Cards	4.031215
3	Age	-0.669308
4	Education	-0.375954
5	Gender	10.368840
6	Student	417.417484
7	Married	-13.265344

The regression coefficients are not uniquely determined. In turn it hurts the interpretability of the model as then the regression coefficients are not unique and have influences from other features.

Both limit and rating have positive coefficients, but it is hard to understand if the balance is higher because of the rating or is it because of the limit? If we remove limit then we achieve almost the same model performance but the coefficients change.

Beyond linearity

So far we assumed:

- linear relationship between X and Y
- the residuals $r_i = y_i - \hat{y}_i$ were uncorrelated (taking the average of the square residuals to calculate the MSE implicitly assumed uncorrelated residuals).

These assumptions need to be verified using the data and **visually inspecting the residuals**.

Residual Analysis

If the correct model is not linear then,

$$y = \beta_0 + \beta_1 x + \phi(x) + \varepsilon$$

our model assuming linear relationship is:

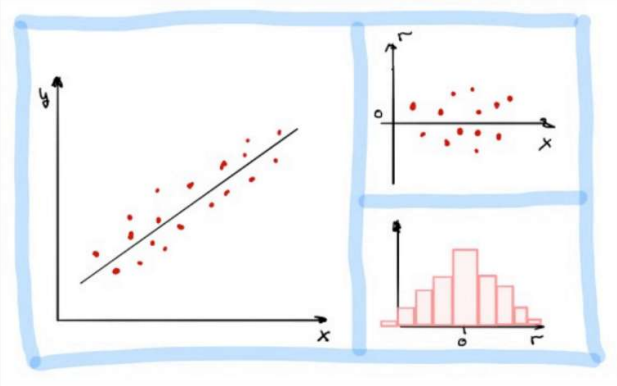
$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

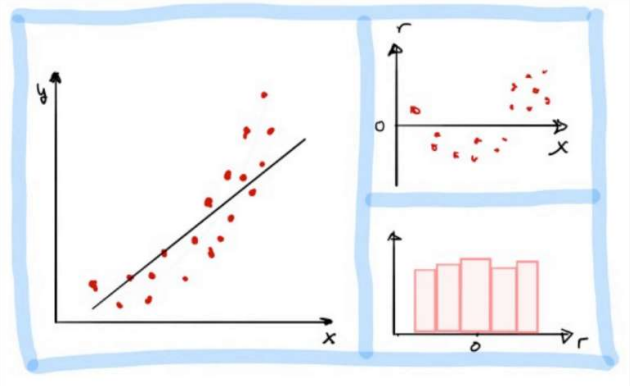
Then the residuals, $r = y - \hat{y} = \varepsilon + \phi(x)$, are not independent of x

In residual analysis, we typically create two types of plots:

1. a plot of r_i with respect to x_i or $\hat{y}_i$. This allows us to compare the distribution of the noise at different values of x_i or $\hat{y}_i$.
2. a histogram of r_i. This allows us to explore the distribution of the noise independent of x_i or $\hat{y}_i$.

Residual Analysis





Linear assumption is correct. There is no obvious relationship between residuals and x. Histogram of residuals is symmetric and normally distributed.

Linear assumption is incorrect. There is an obvious relationship between residuals and x. Histogram of residuals is symmetric but not normally distributed.

Note: For multi-regression, we plot the residuals vs predicted $\hat{y}$, since there are too many x's and that could wash out the relationship.

Beyond linearity

We also assumed that the average effect on sales of a one-unit increase in TV is always β_1, regardless of the amount spent on radio.

Synergy effect or **interaction effect** states that when an increase on the radio budget affects the effectiveness of the TV spending on sales.

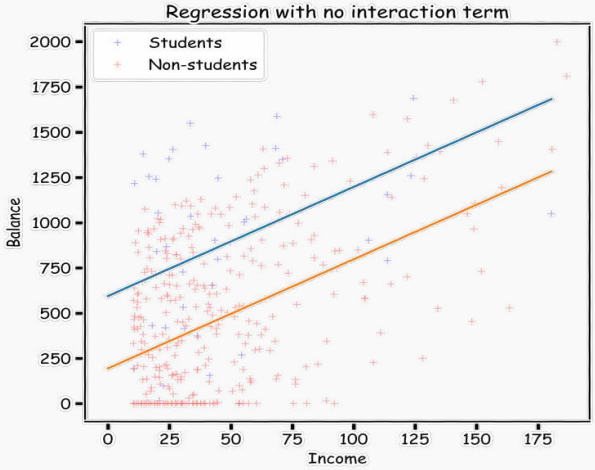
We change

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon$$

To

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \boxed{\beta_3 X_1 X_2} + \epsilon$$

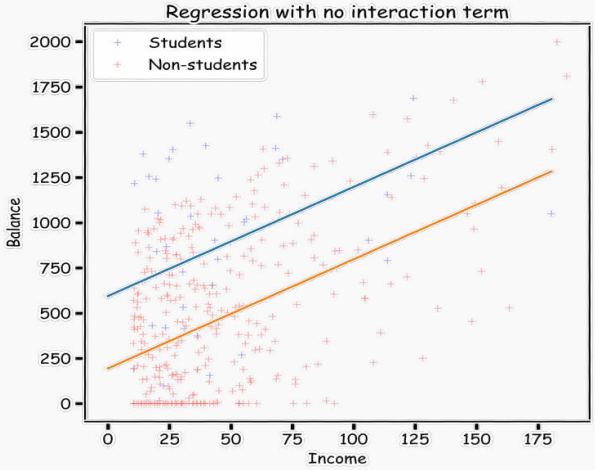
What does it mean?

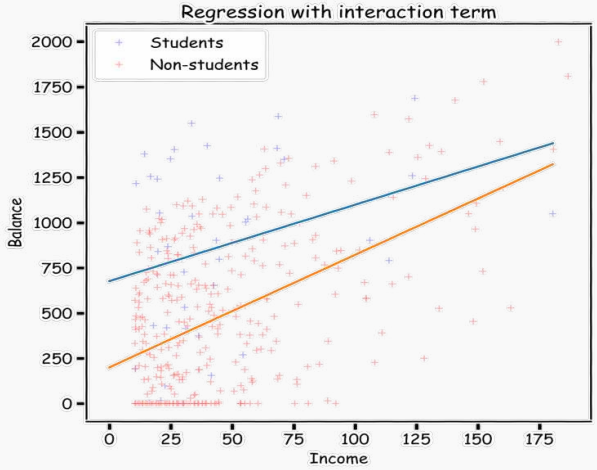


$$x_{Student} = \begin{cases} 0 & Balance = \beta_0 + \beta_1 \times Income. \\ 1 & Balance = (\beta_0 + \beta_2) + (\beta_1) \times Income. \end{cases}$$

$\beta_0 + \beta_1 \times \text{income} + \beta_2 \times \text{student} + \beta_3 \times \text{income} \times \text{student}$

What does it mean?





$$x_{Student} = \begin{cases} 0 & Balance = \beta_0 + \beta_1 \times Income. \\ 1 & Balance = (\beta_0 + \beta_2) + (\beta_1) \times Income. \end{cases}$$

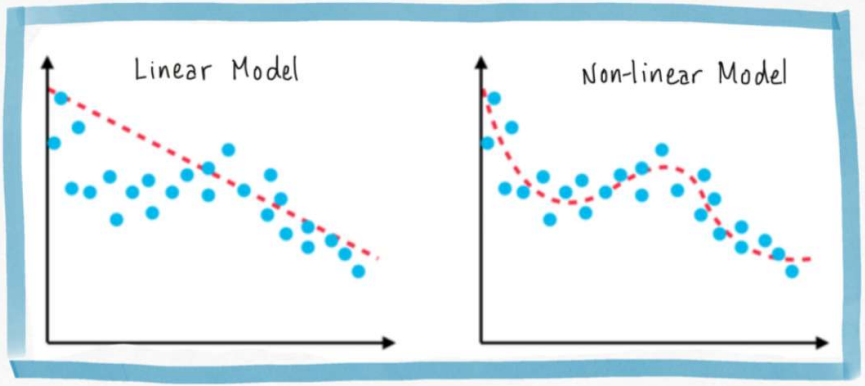
$$x_{Student} = \begin{cases} 0 & Balance = \beta_0 + \beta_1 \times Income. \\ 1 & Balance = (\beta_0 + \beta_2) + (\beta_1 + \beta_3) \times Income \end{cases}$$

Too many predictors, collinearity and too many interaction terms leads to **OVERFITTING**!

Polynomial Regression

Fitting non-linear data

Multi-linear models can fit large datasets with many predictors. But 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

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \qquad \mathbf{X} = \begin{pmatrix} 1 & x_1^1 & \ldots & x_1^M \\ 1 & x_2^1 & \ldots & x_2^M \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \ldots & x_n^M \end{pmatrix}, \qquad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_M \end{pmatrix}.$$

Polynomial Regression

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

Multi-Regression

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \quad \mathbf{X} = \begin{pmatrix} 1 & x_{1,1} & \dots & x_{1,J} \\ 1 & x_{2,1} & \dots & x_{2,J} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \dots & x_{n,J} \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_J \end{pmatrix},$$

Poly-Regression

$$\mathbf{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \quad \mathbf{X} = \begin{pmatrix} 1 & x_1^1 & \dots & x_1^M \\ 1 & x_2^1 & \dots & x_2^M \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^M \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_M \end{pmatrix}.$$

Model Training

Give a dataset $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, we find the optimal polynomial model:

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

1. We transform the data by adding new predictors:

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

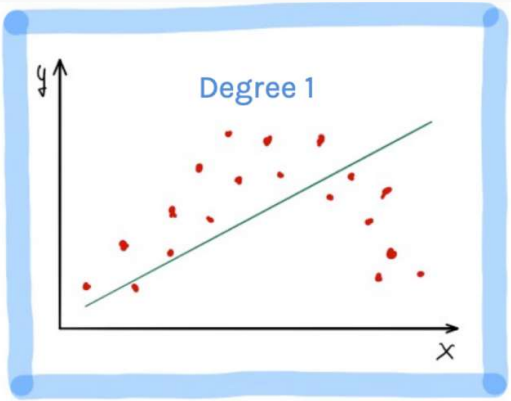
where $\tilde{x}_k = x^k$

2. Fit the parameters by minimizing the MSE using vector calculus. As in multi-linear regression:

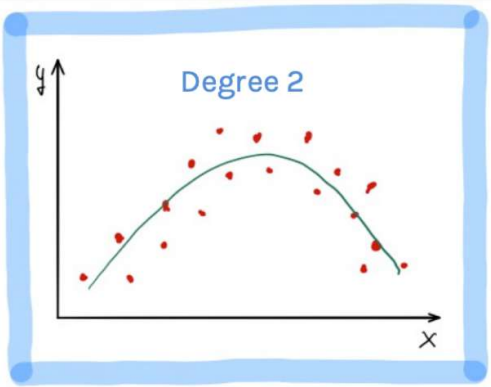
$$\widehat{\boldsymbol{\beta}} = \left(\tilde{\boldsymbol{X}}^T \tilde{\boldsymbol{X}}\right)^{-1} \tilde{\boldsymbol{X}}^T \boldsymbol{y}$$

Polynomial Regression (cont)

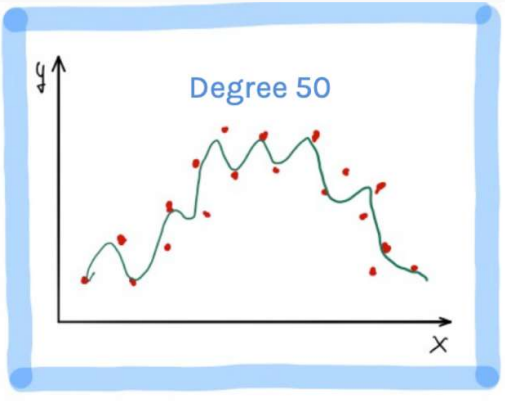
Fitting a polynomial model requires choosing a degree.



Underfitting: when the degree is too low, the model cannot fit the trend.

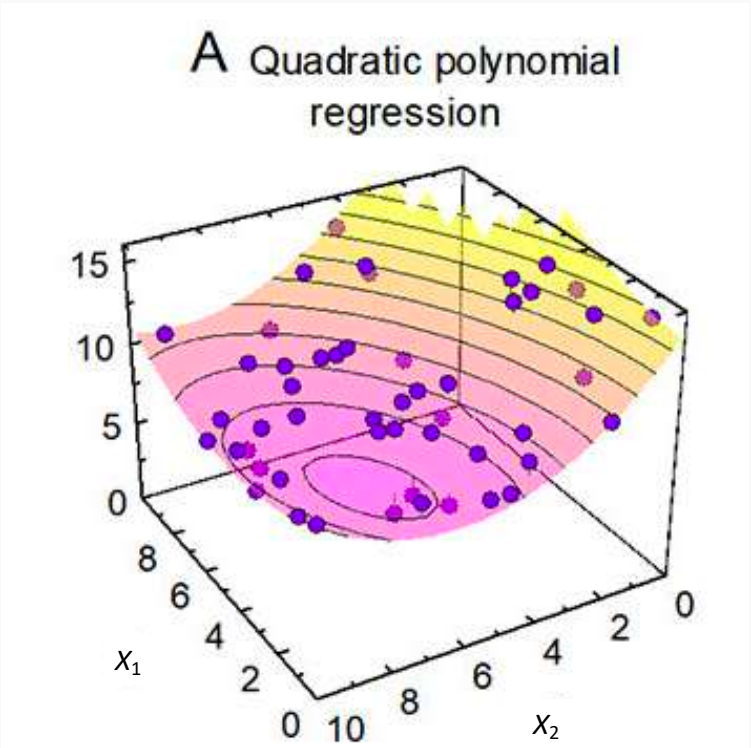


We want a model that fits the trend and ignores the noise.



Overfitting: when the degree is too high, the model fits all the noisy data points.

Poly-Regression



Feature Scaling

Do we need to scale out features for polynomial regression?

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

However if the range of X is low 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.

It is always a good idea to scale X when considering polynomial regression:

$$X^{norm} = \frac{X - \bar{X}}{\sigma_X}$$

Note: sklearn's StandardScaler() can do this.

High degree of polynomial
leads to **OVERFITTING**!

Model Selection

Model Selection

Model selection is the application of a principled method to determine the complexity of the model, e.g. choosing a subset of predictors, choosing the degree of the polynomial model etc.

A strong motivation for performing model selection is to avoid overfitting, which we saw can happen when:

- there are too many predictors:
  - the feature space has high dimensionality
  - the polynomial degree is too high
  - too many cross terms are considered
- the coefficients values are too **extreme (we have not seen this yet)**

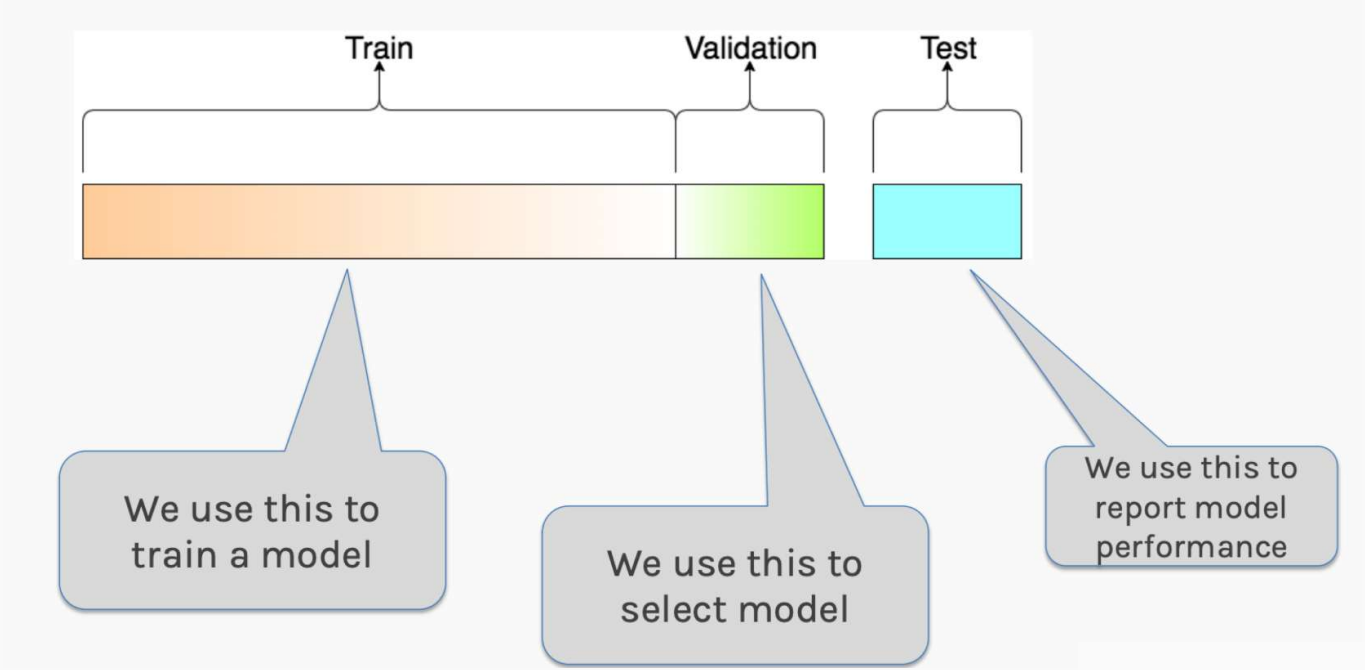
Generalization Error

We know to evaluate the model on both train and test data, because models that do well on training data may do poorly on new data (overfitting).

The ability of models to do well on new data is called generalization.

The goal of model selection is to choose the model that generalizes the best.

Train-Validation-Test

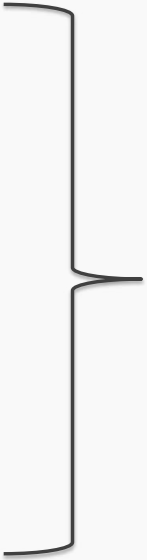


Model Selection

Question: How many different models when considering J predictors (only linear terms) do we have?

Example: 3 predictors (X_1, X_2, X_3)

- Models with 0 predictor:
  - M0:
- Models with 1 predictor:
  - M1: X_1
  - M2: X_2
  - M3: X_3
- Models with 2 predictors:
  - M4: $\{X_1, X_2\}$
  - M5: $\{X_2, X_3\}$
  - M6: $\{X_3, X_1\}$
- Models with 3 predictors:
  - M7: $\{X_1, X_2, X_3\}$



2^J Models

Stepwise Variable Selection and Cross Validation

Selecting optimal subsets of predictors (including choosing the degree of polynomial models) through:

- stepwise variable selection - iteratively building an optimal subset of predictors by optimizing a fixed model evaluation metric each time.
- validation - selecting an optimal model by evaluating each model on validation set.

Stepwise Variable Selection: Forward method

In **forward selection**, we find an 'optimal' set of predictors by iteratively building up our set.

1. Start with the empty set P0, construct the null model M0.

2. For $k = 1, \dots, J$:

2.1 Let M_{k-1} be the model constructed from the best set of $k-1$ predictors, P_{k-1}.

2.2 Select the predictor X_{n_k}, not in P_{k-1}, so that the model constructed from $P_k = X_{n_k} \cup P_{k-1}$ optimizes a fixed metric (this can be p-value, F-stat; validation MSE, R^2, or AIC/BIC on training set).

2.3 Let M_k denote the model constructed from the optimal P_k.

3. Select the model M amongst $\{M_0, M_1, \dots, M_J\}$ that optimizes a fixed metric (this can be validation MSE, R^2; or AIC/BIC on training set)

Stepwise Variable Selection Computational Complexity

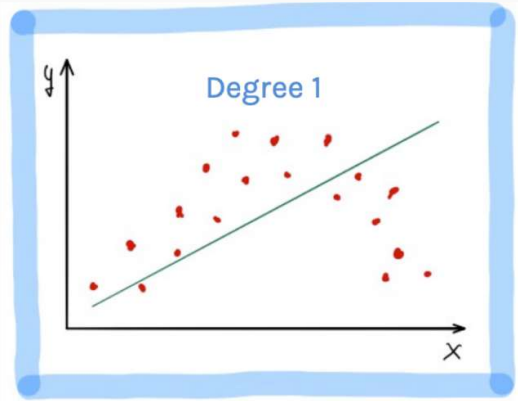
How many models did we evaluate?

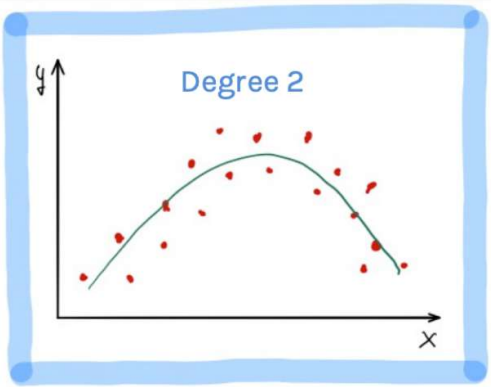
- 1st step, ***J* Models**
- 2nd step, *J-1* **Models** (add 1 predictor out of *J-1* possible)
- 3rd step, *J-2* **Models** (add 1 predictor out of *J-2* possible)
- …

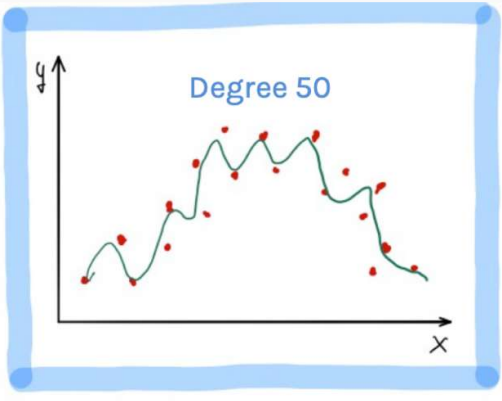
$$O(J^2) \ll 2^J \text{ for large } J$$

Choosing the degree of the polynomial model

Fitting a polynomial model requires choosing a degree.





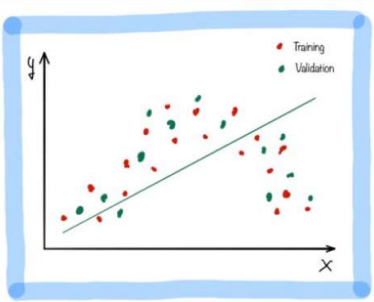


Underfitting: when the degree is too low, the model cannot fit the trend.

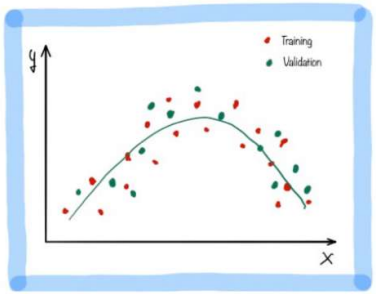
We want a model that fits the trend and ignores the noise.

Overfitting: when the degree is too high, the model fits all the noisy data points.

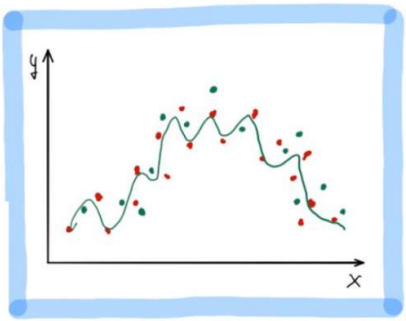
Underfitting: train and validation error is high.

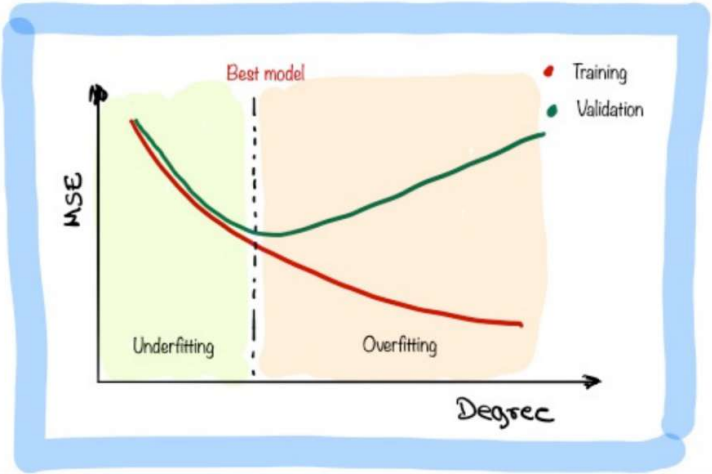


Best model: validation error is minimum.



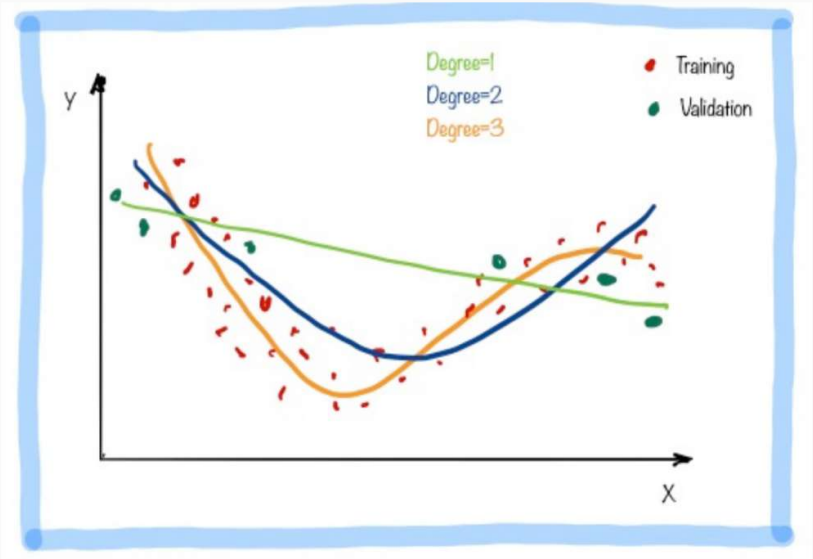
Overfitting: train error is low, validation error is high.





Cross Validation: Motivation

Using a single validation set to select amongst multiple models can be problematic - **there is the possibility of overfitting to the validation set**



It is obvious that degree=3 is the correct model but the validation set by chance favors the linear model.

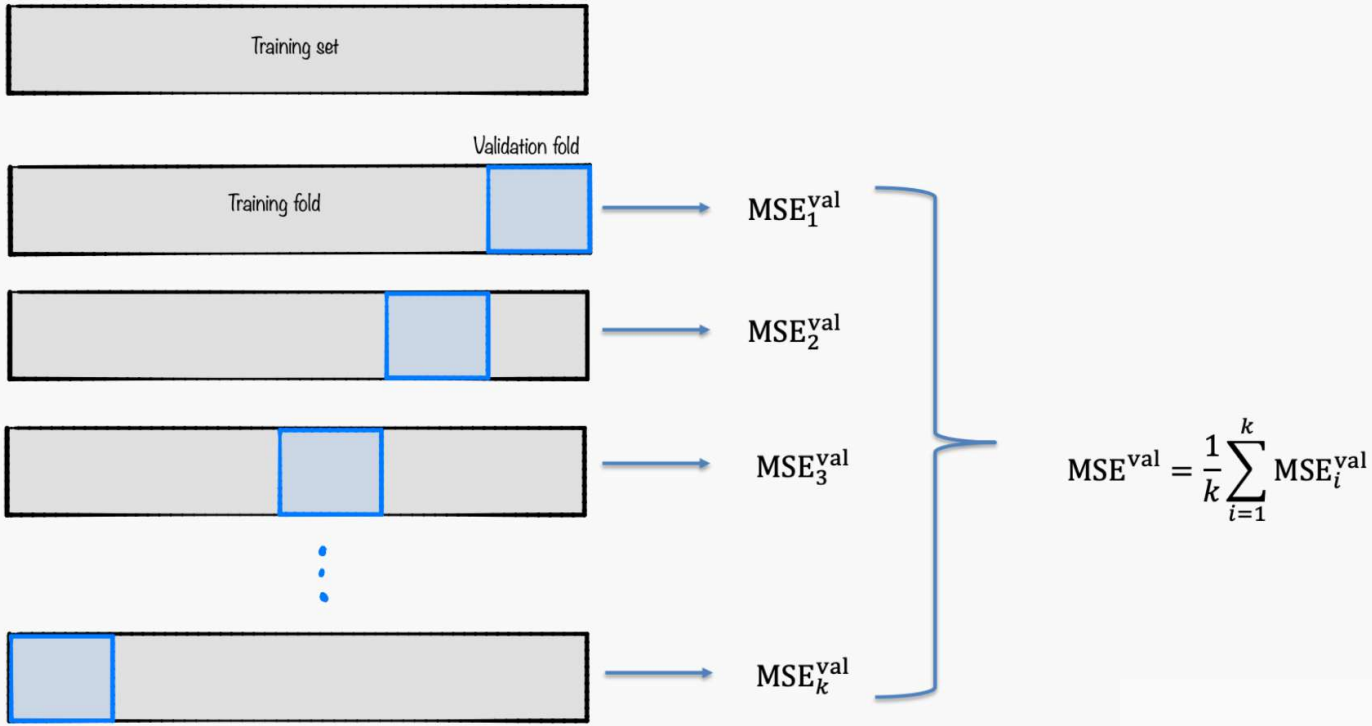
Cross Validation: Motivation

Using a single validation set to select amongst multiple models can be problematic - **there is the possibility of overfitting to the validation set**.

One solution to the problems raised by using a single validation set is to evaluate each model on **multiple** validation sets and average the validation performance.

One can randomly split the training set into training and validation multiple times **but** randomly creating these sets can create the scenario where important features of the data never appear in our random draws.

Cross Validation



K-Fold Cross Validation

Given a data set $\{X_1, \dots, X_n\}$, where each $\{X_1, \dots, X_n\}$ contains *J* features.

To ensure that every observation in the dataset is included in at least one training set and at least one validation set we use the **K-fold validation**:

- split the data into *K* uniformly sized chunks, $\{C_1, \dots, C_K\}$
- we create *K* number of training/validation splits, using one of the *K* chunks for validation and the rest for training.

We fit the model on each training set, denoted $\hat{f}_{C_{-i}}$, and evaluate it on the corresponding validation set, $\hat{f}_{C_{-i}}(C_i)$. The ***cross validation is the performance*** of the model averaged across all validation sets:

$$CV(\text{Model}) = \frac{1}{K}\sum_{i=1}^{K} L(\hat{f}_{C_{-i}}(C_i))$$

where *L* is a loss function.

Leave-One-Out

*Or using the **leave one out*** method:

- validation set: $\{X_i\}$
- training set: $X_{-i} = \{X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n\}$

for $i = 1, \dots, n$:

We fit the model on each training set, denoted $\hat{f}_{X_{-i}}$, and evaluate it on the corresponding validation set, $\hat{f}_{X_{-i}}(X_i)$.

The ***cross validation score*** is the performance of the model averaged across all validation sets:

$$CV(\text{Model}) = \frac{1}{n}\sum_{i=1}^{n} L(\hat{f}_{X_{-i}}(X_i))$$

where *L* is a loss function.