LA SERENA DATA SCIENCE SCHOOL

Regression

The basics of statistical inference

Matthew J. Graham
Center for Data-Driven Discovery/ZTF, Caltech
mjg@caltech.edu
(with material from Pavlos Protopapas and Guillermo Cabrera)

August 20, 2018

CENTER FOR DATA-DRIVEN DISCOVERY

Outline

- Statistical Modelling
- Error and Loss Functions
- Evaluating Model Performance
- Linear Regression
- Multiple Linear Regression (with interaction terms)
- Polynomial Regression
- Robust Regression
- Model Selection
- Regularization
- Heteroscedastic Errors
- Logistic Regression

Predicting a variable

Let's imagine a scenario where we'd like to predict one variable using another (or a set of other) variables:

Examples:

- Predicting the amount of views a YouTube video will get next week based on video length, the date it was posted, previous number of views, etc.
- Predicting which movies a Netflix user will rate highly based on their previous movie ratings, demographic data, etc.
- Predicting the expected cab fare in New York City based on time of year, location of pickup, weather conditions, etc.
- Predicting the redshift of a galaxy based on its magnitudes and colors in different passband filters

Outcome vs. predictor values

Definition

Suppose we are observing *p+1* number variables and we are making *n* sets of observations. We call:

- the **outcome** or **response variable** is the variable we'd like to predict; typically, we denote this variable by Y and the individual measurements y_i
- the **features** or **predictor variables** are the variables we use in making the predictions; typically, we denote these variables by $X = (X_1,\ldots,X_p)$ and the individual measurements $x_{i,j}$.

Note: i indexes the observation $(i=1,2,\ldots,n)$ and j indexes the value of the j-th predictor variable $(j=1,2,\ldots,p)$

True vs. statistical model

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

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

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

- **Inference** problems:

What's important is obtaining $\hat{f}$, our estimate of f.

- **Prediction** problems

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

$$\widehat{y_i} = \hat{f}(x_{i,1}, \ldots, x_{i,p})$$

We don't care about the specific form of $\hat{f}$, we just want to make our prediction $\widehat{y}_i$ as close to the observed value y_i as possible.

Error and loss functions

A **loss** or **error function** quantifies how well a model performs.

The **Mean Squared Error (MSE)** is a common loss function for quantitative outcomes:

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

The quantity $|y_i - \widehat{y}_i|$ is called a **residual** and measures the error at the *i*th prediction.

Assuming some mathematical form for f (and $\hat{f}$), values are chosen for the unknown parameters of $\hat{f}$ so that the loss function is minimized on the set of observations.

Other loss functions

- **Maximum absolute deviation**

$$\max_i |y_i - \hat{y}_i|$$

- **Sum of absolute deviations (L$_1$-norm)**

$$\sum_i |y_i - \hat{y}_i|$$

- **Sum of squared errors (L$_2$-norm)**

$$\sum_i (y_i - \hat{y}_i)^2$$

Others used in classification: hinge, logistic

Linear regression

If each observation has only one predictor, we can build a model by first assuming a simple **linear** form for f (and $\hat{f}$):

$$Y = f(X) + \epsilon = \beta_1^{\text{true}} X + \beta_0^{\text{true}} + \epsilon$$

Remember ϵ is the random quantity or **noise** by which observed values of Y differ from the rule $f(X)$.

Our estimate is then:

$$\hat{Y} = \hat{f}(X) = \hat{\beta}_1 \, X + \hat{\beta}_0$$

where $\widehat{\beta_1}$ and $\widehat{\beta_0}$ are estimates of β_1 and β_0 computed from our observations.

Which models are linear?

$$Y = \beta_0 e^{-X} + \epsilon$$

$$Y = \beta_0 + \beta_1 \cos X + \beta_2 \sin X + \epsilon$$

$$Y = \left(\frac{X}{\beta_0}\right)^{-\beta_1} + \epsilon$$

$$Y = \begin{cases} \beta_0 + \beta_1 X : X < x_0 \\ \beta_2 + \beta_3 X : X \geq x_0 \end{cases}$$

Which models are linear?

$Y = \beta_0 e^{-X} + \epsilon$	YES
$Y = \beta_0 + \beta_1 \cos X + \beta_2 \sin X + \epsilon$	YES
$Y = \left(\frac{X}{\beta_0}\right)^{-\beta_1} + \epsilon$	NO
$Y = \begin{cases} \beta_0 + \beta_1 X : X < x_0 \\ \beta_2 + \beta_3 X : X \geq x_0 \end{cases}$	NO

$$\boldsymbol{f(x \mid \theta) = \sum_{p=1}^{k} \theta_p g_p(x)}$$

Inference for linear regression

Assume MSE for our loss function:

$$L(\beta_0, \beta_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2 = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - (\beta_1 X + \beta_0)\right)^2$$

then the optimal values for $\widehat{\beta_1}$ and $\widehat{\beta_0}$ should be:

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

Now taking partial derivatives of *L* and finding their global minima gives:

$$\widehat{\beta_1} = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2} \qquad \widehat{\beta_0} = \bar{y} - \widehat{\beta_1}\bar{x}$$

Linear regression: a simple example

Let’s assume a simple data set, the average cab fare in NYC at different times of day:

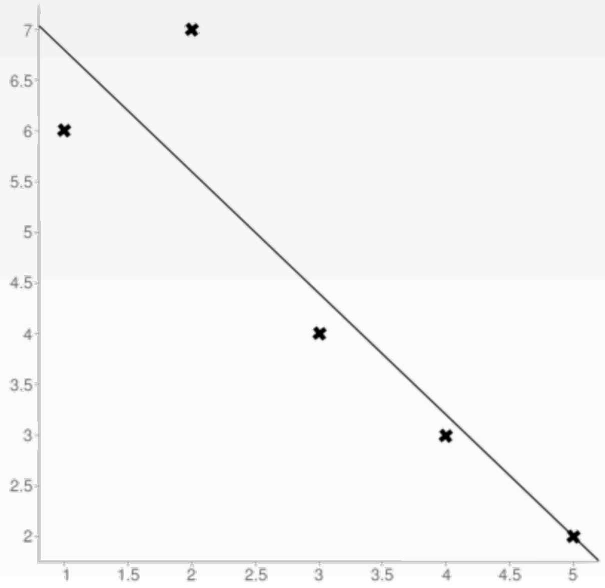
Time (X)	1	2	3	4	5
Fare (Y)	6	7	4	3	2

By our formula, we compute the regression line to be:

$$\hat{Y} = -1.2X + 8$$

and the predicted responses:

$$\hat{Y} = (6.8, 5.6, 4.4, 3.2, 2.0)$$



Training vs. Testing Sets

One way to evaluate our model is to use it to predict the responses for predictors that we did not use to build our model. This involves splitting data into a **training set** and a **testing set** after collecting a set of observations of predictor and response.

We use the training set to build a model and use the testing set to perform a final evaluation of the model, simulating model performance in real-time usage.

Note: To maintain the integrity of the final test, the test data should only be used once and the results should not be used to inform changes made to the model.

Measurement vs. sampling error

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

The measurement error or **irreducible error** ϵ is noise introduced by random variations in natural systems or imprecisions of our scientific instruments.

Variations in $\widehat{\beta_0}$ and $\widehat{\beta_1}$ are affected by:

- Var(ϵ) – the variance in the noise (**measurement)**
- n, the number of observations (**sampling**)

The variances of $\widehat{\beta_0}$, $\widehat{\beta_1}$ are also called **standard errors**.

Estimating sampling errors

- **Analytically**

If we know the variance σ^2 of the noise ϵ, we can compute $\mathrm{SE}(\widehat{\beta_0})$, $\mathrm{SE}(\widehat{\beta_1})$ analytically:

$$SE(\widehat{\beta_0}) = \sigma\sqrt{\frac{1}{n} + \frac{\bar{x}^2}{\sum_i (x_i - \bar{x})^2}} \qquad SE(\widehat{\beta_1}) = \frac{\sigma}{\sqrt{\sum_i (x_i - \bar{x})^2}}$$

- **Bootstrapping**

Estimate properties of an estimator by measuring those properties by randomly sampling from observed data

- **Empirically**

Assume residuals $\epsilon_i = y_i - \hat{y}_i$ and $\epsilon_j = y_j - \hat{y}_j$ are uncorrelated for i≠j and $\epsilon \sim \mathcal{N}(0, \sigma^2)$:

$$\sigma \approx \sqrt{\frac{n \cdot MSE}{n-2}} = \sqrt{\frac{\sum_i (y_i - \hat{y}_i)^2}{n-2}}$$

Confidence intervals

Definition

A *n*% **confidence interval** of an estimate $\hat{X}$ is the range of values such that the true value of X is contained in this interval with *n* percent probability.

For linear regression, the 95% confidence interval for $\widehat{\beta_0}, \widehat{\beta_1}$ can be approximated using their standard errors:

$$\widehat{\beta_k} = \widehat{\beta_k} \pm 2SE\left(\widehat{\beta_k}\right)$$

for k = 0,1.

Residual analysis

In estimating the variance of ϵ, we assumed that:

- the residuals $\epsilon_i = y_i - \hat{y_i}$ were uncorrelated
- normally distributed with zero mean and fixed variance.

These assumptions need to be verified using the data. In *residual analysis,* we typically create two types of plots:

- a plot of ϵ_i vs. x_i - this allows us to compare the distribution of noise at different values of x_i

- a histogram of ϵ_i - this allows us to explore the distribution of the noise independent of x_i

Model fitness: R^2

While loss functions measure the predictive errors made by a model, we are also interested in the ability of our models to capture interesting features or variations in the data.

The **explained variance** or R^2 is the ratio of the variation of the model and the variation in the data. The explained variance of a regression line is given by:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} |y_i - \bar{y_i}|^2}{\sum_{i=1}^{n} |\hat{y_i} - \bar{y_i}|^2}$$

For a regression line, this gives:

$$0 \leq R^2 \leq 1$$

Model fitness: information criteria

Information criteria are a set of metrics which measure the fit of a model to observations given the number of parameters used in the model.

Two such criteria are **Aiken's Information Criterion** and **Bayes Information Criterion**:

$$RSS = \sum_i (y_i - \hat{y}_i)$$

$$AIC \approx n \cdot \ln\left(\frac{RSS}{n}\right) + 2J$$

$$BIC \approx n \cdot \ln\left(\frac{RSS}{n}\right) + J \cdot \ln(n)$$

The smaller the AIC or BIC value, the better the model.

Cross validation: leave-one-out

Given a data set $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$, where each $\boldsymbol{X}_i = \left(x_{i,1}, \ldots, x_{i,J}\right)$ contains J number of features.

To ensure that every observation in the data set is included in at least one training set and at least one validation set, we create training/validation splits using the **leave one out** method:

- Validation set: $\{\boldsymbol{X}_i\}$
- Training set: $\boldsymbol{X}_{-i} := \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{i-1}, \boldsymbol{X}_{i+1}, \ldots, \boldsymbol{X}_n\}$

for $i = 1, \ldots, n$. We fit the model on each training set, denoted $\widehat{f_{\boldsymbol{X}_{-i}}}$, and evaluate it on the corresponding validation set, $\widehat{f_{\boldsymbol{X}_{-i}}}(\boldsymbol{X}_i)$. The **cross validation score** is the performance of the model averaged across all validation sets:

$$CV(Model) = \frac{1}{n} L\left(\widehat{f_{\boldsymbol{X}_{-i}}}(\boldsymbol{X}_i)\right)$$

Cross validation: K-fold

Rather than creating *n* number of training/validation splits, each time leaving one data point for the validation set, we can include more data in the validation set using **K-fold validation**:

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

We fit the model for each training set, denoted $\widehat{f_{C_{-i}}}$, and evaluate it on the corresponding validation set, $\widehat{f_{C_{-i}}}(C_i)$. The cross validation score is the performance modelled :

$$CV(Model) = \frac{1}{n} L\left(\widehat{f_{\boldsymbol{C}_{-i}}}(\boldsymbol{C}_i)\right)$$

Robust regression

Often a data set has a small fraction of points with very large residuals from the regression line. If scientific knowledge of the identity of discordant points is not available, **robust** techniques can be used to reduce the influence of such outliers.

The **breakdown point** of an estimator is the proportion of arbitrarily large data values that can be handled before giving an arbitrarily large estimator value (sample mean has a breakdown point of 1/n).

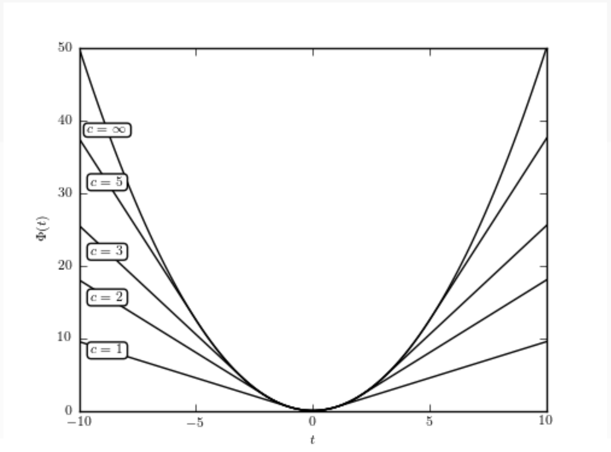
M estimators modify the underlying likelihood estimator to be less sensitive than the classic L_2 norm: for any function ψ, any solution $\widehat{\beta_M}$ is an M-estimator:

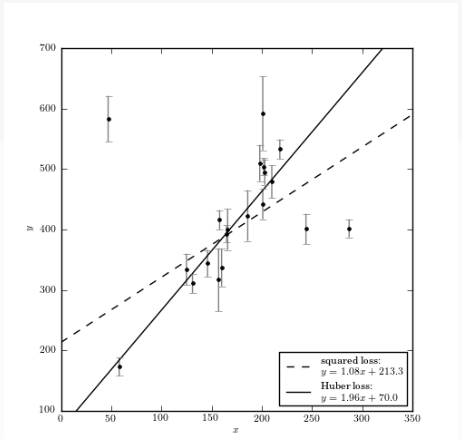
$$\sum_{i=1}^{n} \psi(y_i - \widehat{\beta_M} x_i) x_i = 0$$

Robust regression: Huber loss

Ideally the M estimator increases less than the square of the residual and has a unique minimum at zero. One common example is the **Huber estimator** which minimizes:

$$L_{Huber} = \begin{cases} \frac{1}{2}(y_i - \widehat{y}_i)^2 \text{ for } |y_i - \widehat{y}_i| \le c \\ c|y_i - \widehat{y}_i| - \frac{1}{2}c^2 \text{ for } |y_i - \widehat{y}_i| \ge c \end{cases}$$





Ivezic et al. 2014

Multiple linear regression

It is unlikely that any response variable *Y* depends solely on one predictor *x*. Rather, we expect *Y* is a function of multiple predictors $f(X_1,\ldots,X_J)$. We can still assume a linear form, though:

$$y = f(X_1, \ldots, X_J) + \epsilon = \beta_0 + \beta_1 x_1 + \cdots + \beta_J x_J + \epsilon$$

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

$$\hat{y} = \hat{f}(X_1, \ldots, X_J) = \widehat{\beta_0} + \widehat{\beta_1} x_1 + \cdots + \widehat{\beta_J} x_J$$

Multiple linear regression in vectors

Given a set of observations

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

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

$$\boldsymbol{Y}=\begin{pmatrix}y_1\\ \vdots\\ y_J\end{pmatrix},\qquad \boldsymbol{X}=\begin{pmatrix}1 & \cdots & x_{1,J}\\ \vdots & \ddots & \vdots\\ 1 & \cdots & x_{n,J}\end{pmatrix},\qquad \boldsymbol{\beta}=\begin{pmatrix}\beta_1\\ \vdots\\ \beta_J\end{pmatrix}$$

The MSE can then be expressed as:

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

and minimizing it yields:

$$\hat{\beta}=(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}=\underset{\boldsymbol{\beta}}{\operatorname{argmin}}\,MSE\,(\boldsymbol{\beta})$$

Interaction terms

Now suppose that there are interactions between predictors:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$$

The term $\beta_3 x_1 x_2$ is called the **interaction term**.

$$\boldsymbol{Y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \boldsymbol{X} = \begin{pmatrix} 1 & x_{1,1} & x_{1,2} & x_{1,1}x_{1,2} \\ 1 & x_{2,1} & x_{2,2} & x_{2,1}x_{2,2} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n,1} & x_{n,2} & x_{n,1}x_{n,2} \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix}$$

Minimizing the MSE again gives:

$$\hat{\beta} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{Y} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}}\, MSE\,(\boldsymbol{\beta})$$

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*:

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

However, we can treat as linear regression with each x^m as a separate predictor. Thus:

$$\boldsymbol{Y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \boldsymbol{X} = \begin{pmatrix} 1 & x_1^1 & \dots & x_1^M \\ 1 & x_2^1 & \dots & x_2^M \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^M \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_M \end{pmatrix}$$

Minimizing the MSE again gives:

$$\hat{\beta} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{Y} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}}\, MSE\,(\boldsymbol{\beta})$$

Model selection

This is application of a principled method to determine the complexity of a model:

- Choosing a subset of predictors
- Choosing the degree of a polynomial model

It typically consists of the following steps:

- Split the training set into two subsets: training and validation
- Multiple models are fitted on the training set and each model is evaulated on the validation set
- The model with the best validation performance is selected
- The selected model is evaluated one last time on the testing set

Overfitting

Overfitting is the phenomenon wherein the model is unnecessarily complex. Portions of the model are actually capturing the random noise in the observation rather than the relationship between predictor(s) and response. This causes the model to lose predictive power with new data.

Overfitting 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 coefficient values are too extreme

A sign of overfitting may be a high training R^2 or low MSE and unexpectedly poor testing performance.

Theil-Sen estimator

"The most popular nonparametric technique for estimating a linear trend"

The slope of the regression line, β_{TS}, is given by median of the $\frac{n(n-1)}{2}$ slopes defined by all pairs of data points:

$$\beta_{ij} = \frac{Y_i - Y_j}{X_i - X_j}$$

The intercept is given by the median of $Y_i - \beta_{TS} X_i$.

Confidence intervals can be determined from the middle 95% of slopes from sampled pairs of points (n ~ 600).

Regularization

The idea of regularization revolves around modifying the loss function, *L*; in particular, we add a **regularization term** that penalizes some specific properties of the model parameters:

$$L_{reg}(\beta) = L(\beta) + \lambda R(\beta)$$

where λ is a scalar that gives the weight (or importance) of the regularization term.

Fitting the model using the modified loss function L_{reg} would result in model parameters with desirable properties (specified by *R*).

LASSO Regression

Suppose we want to discourage extreme values in model parameters; we then need to choose a regularization term that penalizes parameter magnitudes. With MSE, a regularized loss function is:

$$L_{LASSO}(\beta) = \frac{1}{n}\sum_{i=1}^{n} |y_i - \boldsymbol{\beta}^T \boldsymbol{x}_i|^2 + \lambda \sum_{j=1}^{J} |\beta_j|$$

Note that $\sum_{j=1}^{J} |\beta_j|$ is the L$_1$ norm of the vector β

$$\sum_{j=1}^{J} |\beta_j| = \|\boldsymbol{\beta}\|_1$$

L_{LASSO} is often called the loss function for **L$_1$ regularization** and finding model parameters that minimize it is called **LASSO regression**.

Ridge Regression

Alternatively, we can choose a regularization term that penalizes the squares of the parameter magnitudes:

$$L_{Ridge}(\beta) = \frac{1}{n}\sum_{i=1}^{n} |y_i - \boldsymbol{\beta}^T \boldsymbol{x}_i|^2 + \lambda \sum_{j=1}^{J} \beta_j^2$$

Note that $\sum_{j=1}^{J} \beta_j^2$ is the L_2 norm of the vector β

$$\sum_{j=1}^{J} \beta_j^2 = \|\boldsymbol{\beta}\|_2^2$$

L_{Ridge} is often called the loss function for **L_2 regularization** and finding model parameters that minimize it is called **ridge regression**.

Choosing λ

We can see that in both ridge and LASSO regression, the larger our choice of the **regularization parameter** λ, the more heavily we penalize large values in β:

- if λ is close to zero, we recover the MSE, i.e., ridge and LASSO regression is just ordinary regression

- If λ is sufficiently large, the MSE term in the regularized loss function will be insignificant and the regularization term will force β_{Ridge} and β_{LASSO} to be close to zero

The recommendation is to select λ using cross validation.

Heteroscedastic errors

So far we have been assuming that our errors are identically distributed or **homoscedastic**: $\epsilon \sim \mathcal{N}(0, \sigma^2)$. In astronomy, it is common to have errors that are independent but not identically distributed (**heteroscedastic**): $\epsilon_i \sim \mathcal{N}\left(0, \sigma_i^2\right)$ due to night-to-night variations, say. The effect of this is to introduce **weighting** of data points in estimators.

Recall that the matrix solution for linear regression is:

$$\hat{\beta} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{Y}$$

This becomes:

$$\hat{\beta} = (\boldsymbol{X}^T \boldsymbol{C}^{-1} \boldsymbol{X})^{-1} (\boldsymbol{X}^T \boldsymbol{C}^{-1} \boldsymbol{Y})$$

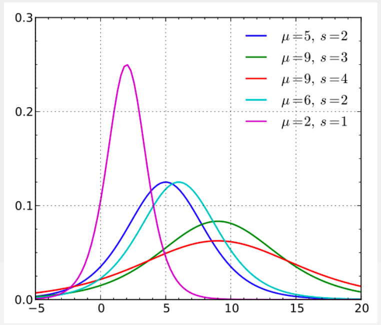
where

$$\boldsymbol{C} = \begin{pmatrix} \sigma_1^2 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_n^2 \end{pmatrix}$$

Logistic regression

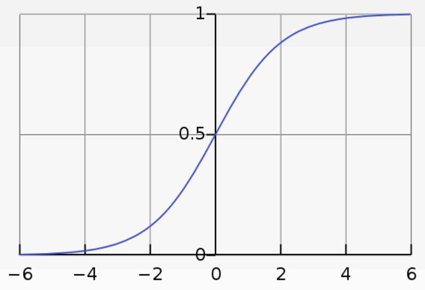
What happens if our response variable is **categorical**?

$$y = \begin{cases} 1, & \beta_0 + \beta_1 x + \epsilon > 0 \\ 0, & \text{otherwise} \end{cases}$$



ϵ is an error distributed by the logistic distribution

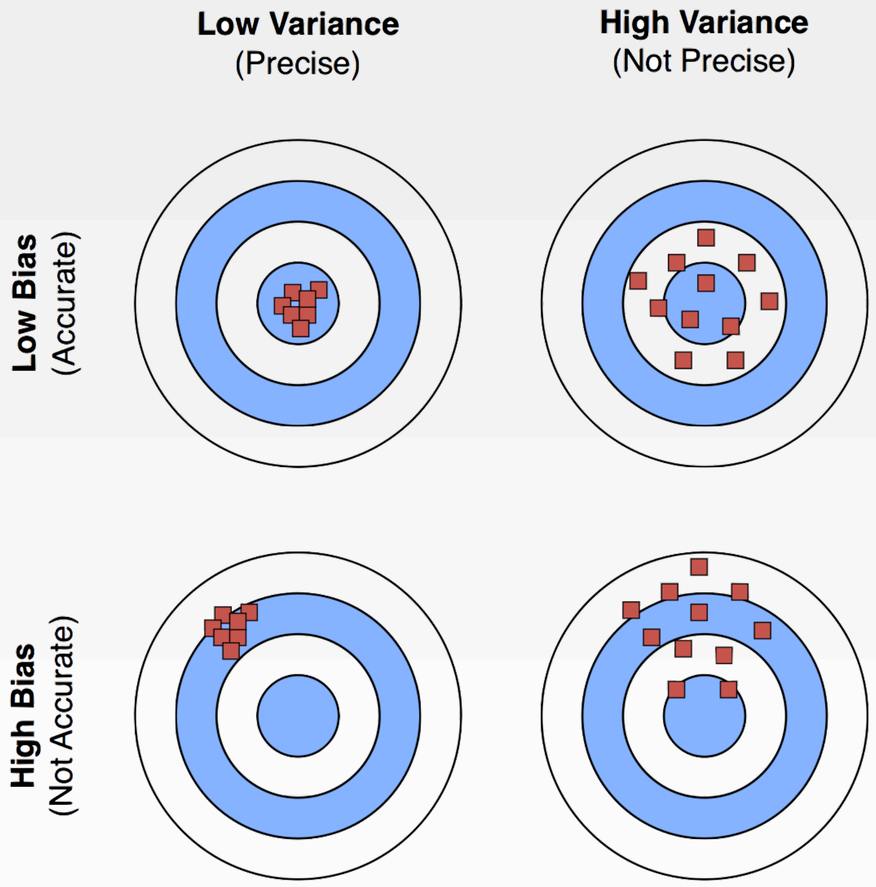
$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

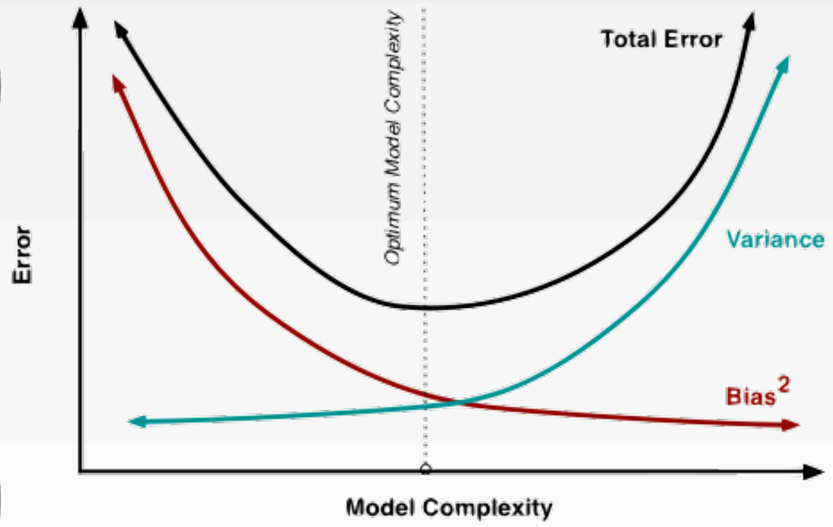


No closed form solution so requires an iterative solution.

Bias and variance

Using a single validation set leaves open the possibility of overfitting


Low Variance
(Precise)
High Variance
(Not Precise)
Low Bias
(Accurate)
High Bias
(Not Accurate)


Error
Optimum Model Complexity
Total Error
Variance
Bias^2
Model Complexity