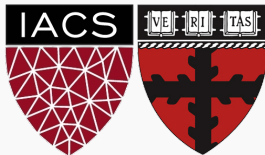# Lecture #4: Introduction to Regression

Data Science 1
CS 109A, STAT 121A, AC 209A, E-109A

Pavlos Protopapas   Kevin Rader
Margo Levine   Rahul Dave

Lecture Outline

Announcements

Data

Statistical Modeling

Regression vs. Classification

Error, Loss Functions

Model I: k-Nearest Neighbors

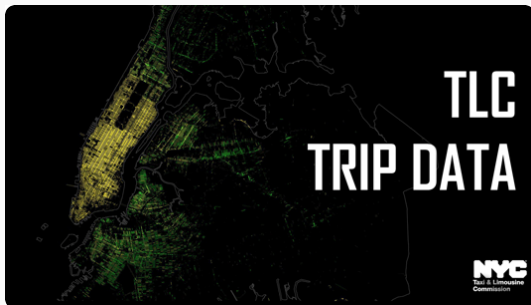Model II: Linear Regression

Evaluating Model Performance

Comparison of Two Models

Announcements

Announcements

1. Work in pairs but not submitting together? Add the name of your partner (only one) in the notebook .
2. HW1 due on Wednesday 11:59pm.
3. Make groups now.
4. A-sections start on Wednesday.
5. HW2 will be released on Wednesday 11:58pm.

Data

The yellow and green taxi trip records include fields capturing pick-up and drop-off dates/times, pick-up and drop-off locations, trip distances, itemized fares, rate types, payment types, and driver-reported passenger counts.
The data used were collected and provided to the NYC Taxi and Limousine Commission (TLC).

More details on the data can be found here:

http://www.nyc.gov/html/tlc/html/about/trip_
record_data.shtml

Notebook:
https:
//github.com/cs109/a-2017/blob/master/Lectures/
Lecture4-IntroRegression/Lecture4_Notebook.ipynb

Statistical Modeling

Predicting a Variable

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

Examples:

- ▶ Predicting the amount of view 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.

There is an asymmetry in many of these problems: the variable we'd like to predict may be more difficult to measure, is more important than the other(s), or may be directly or indirectly influenced by the values of the other variable(s).

Thus, we'd like to define two categories of variables: variables whose value we want to predict and variables whose values we use to make our prediction.

Definition

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

- the variable we'd like to predict the **outcome** or **response variable**; typically, we denote this variable by Y and the individual measurements y_i.

- the variables we use in making the predictions the **features** or **predictor variables**; 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 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 $\widehat{f}$.

Prediction vs. Estimation

For some problems, what's important is obtaining $\widehat{f}$, our estimate of f. These are called *inference* 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 the response variable, we denote the *predicted value* by $\widehat{y}_i$,

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

For some problems, we don't care about the specific form $\widehat{f}$, we just want to make our prediction $\widehat{y}_i$ as close to the observed value y_i as possible. These are called *prediction problems*.

We'll see that some algorithms are better suited for inference and others for prediction.

Regression vs. Classification

Outcome Variables

There are two main types of prediction problems we will see this semester:

- *Regression problems* are ones with a quantitative response variable.
  Example: Predicting the number of taxicab pick-ups in New York.

- *Classification problems* are ones with a categorical response variable.
  Example: Predicting whether or not a Netflix user will like a particular movie.

This distinction is important, as each type of problem may require it's own specialized algorithms along with metrics measuring effectiveness.

Error, Loss Functions

Which of the following linear models is the best? How do you know?
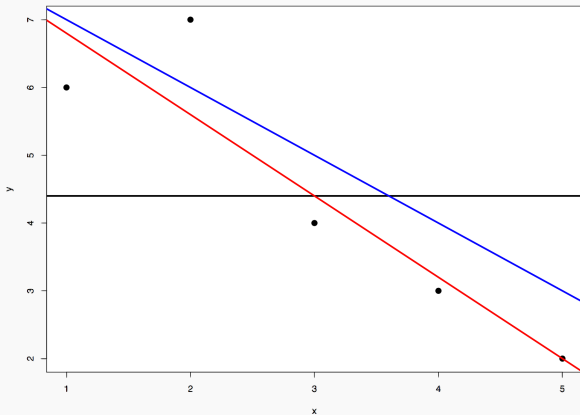
Loss functions are used to choose a suitable estimate $\widehat{f}$ of f.

A statistical modeling approach is often an algorithm that:

- assumes some mathematical form for f, and hence for $\widehat{f}$,
- then chooses values for the unknown parameters of $\widehat{f}$ so that the loss function is minimized on the set of observations

Error & Loss Functions

In order to quantify how well a model performs, we define a *loss* or *error function*.

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

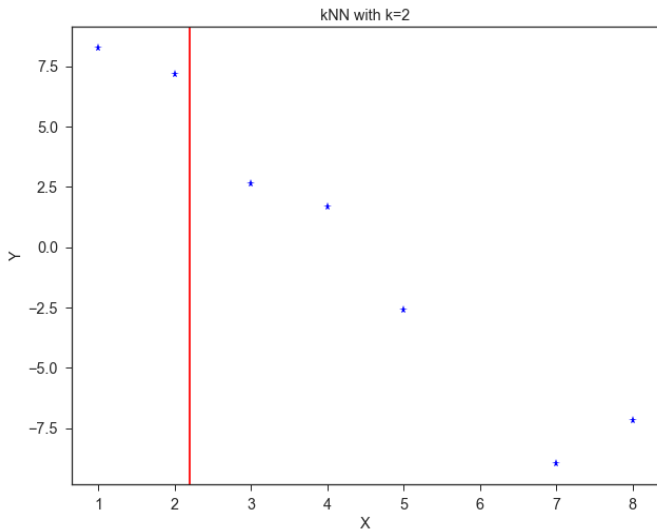$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \widehat{y_i})^2$$

The quantity $|y_i - \widehat{y_i}|$ is called a *residual* and measures the error at the i-th prediction.
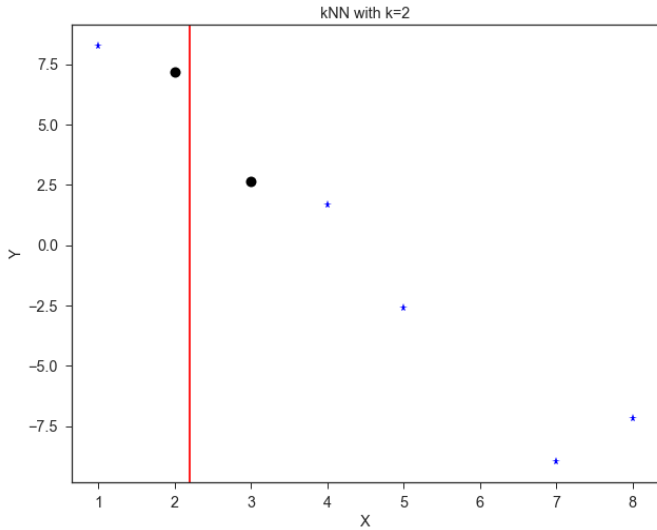
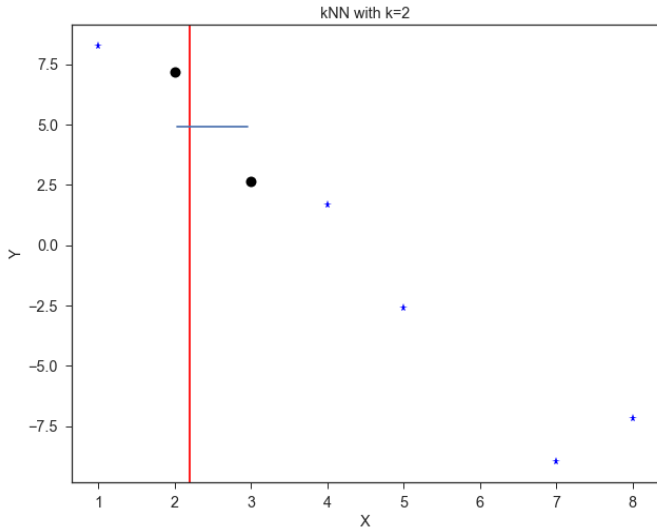**Caution:** The MSE is by no means the only valid (or the best) loss function!

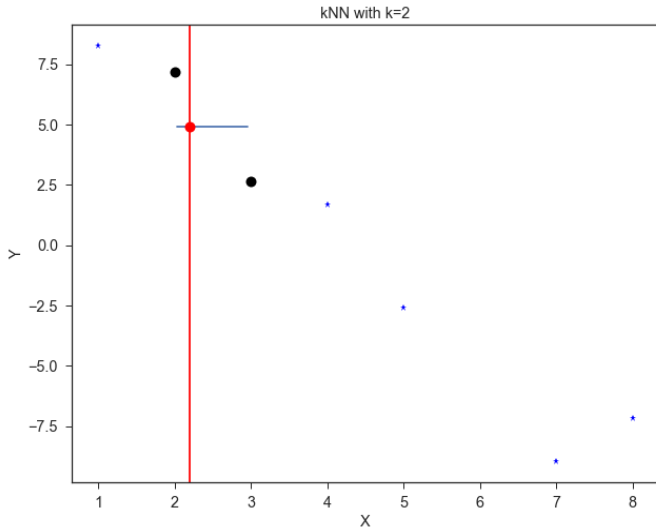**Question:** What would be an intuitive loss function for predicting categorical outcomes?

Model I: k-Nearest Neighbors

kNN with k=2

kNN with k=2

kNN with k=2

kNN with k=2

20

kNN with k=2
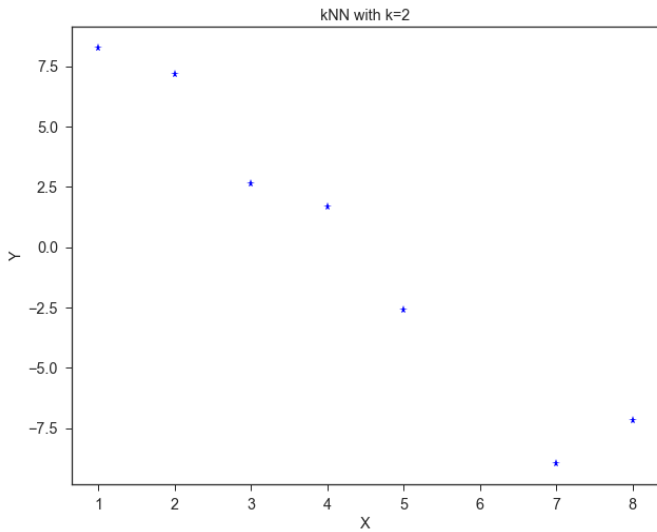
k-Nearest Neighbors

The k-**Nearest Neighbor (kNN) model** is an intuitive way to predict a quantitative response variable:

to predict a response for a set of observed predictor values, we use the responses of other observations most similar to it!

Note: this strategy can also be applied in classification to predict a categorical variable. We will encounter kNN again later in the semester in the context of classification.
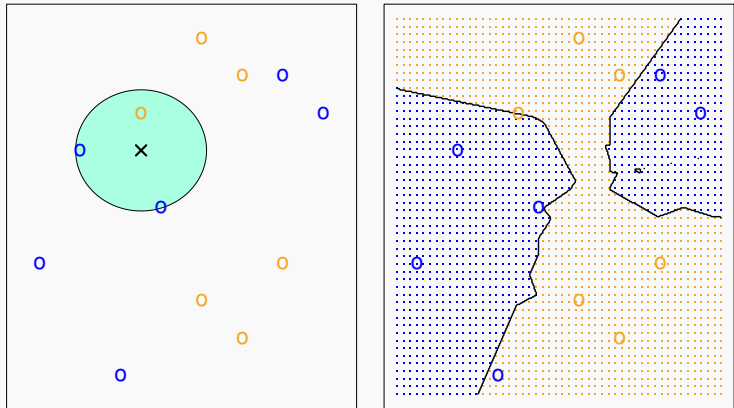
k-Nearest Neighbors

Fixed a value of k. The predicted response for the i-th observation is the average of the observed response of the k-closest observations

$$\widehat{y}_i = \frac{1}{k} \sum_{i=1}^{k} y_{n_i}$$

where $\{X_{n_1}, \ldots, X_{n_k}\}$ are the k observations most similar to X_i (similar refers to a notion of distance between predictors).

kNN Regression: A Simple Example

Suppose you have 5 observations of taxi cab pick ups in New York City, the response is the average cab fare (in units of \$10), and the predictor is time of day (in hours after 7am):

X	1	2	3	4	5
Y	6	7	4	3	2

We calculate the predicted number of pickups using kNN for $k = 2$:

$$X = 1 \qquad \widehat{y}_1 = \frac{1}{2}\left(7 + 4\right) = 5.5$$

kNN Regression: A Simple Example

Suppose you have 5 observations of taxi cab pick ups in New York City, the response is the average cab fare (in units of $10), and the predictor is time of day (in hours after 7am):

X	1	2	3	4	5
Y	6	7	4	3	2

We calculate the predicted number of pickups using kNN for $k = 2$:

$$X = 2 \qquad \widehat{y}_2 = \frac{1}{2}(6 + 4) = 5.0$$

kNN Regression: A Simple Example

Suppose you have 5 observations of taxi cab pick ups in New York City, the response is the average cab fare (in units of $10), and the predictor is time of day (in hours after 7am):

X	1	2	3	4	5
Y	6	7	4	3	2

We calculate the predicted number of pickups using kNN for $k = 2$:

$$\widehat{Y} = (5.5, 5.0, 5.0, 3.0, 3.5)$$

kNN Regression: A Simple Example

Suppose you have 5 observations of taxi cab pick ups in New York City, the response is the average cab fare (in units of $10), and the predictor is time of day (in hours after 7am):

X	1	2	3	4	5
Y	6	7	4	3	2

We calculate the predicted number of pickups using kNN for $k = 2$:

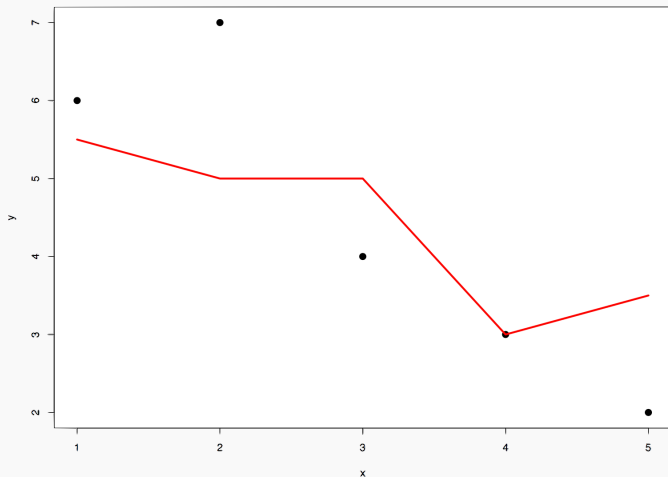$$\widehat{Y} = (5.5, 5.0, 5.0, 3.0, 3.5)$$

The MSE given our predictions is

$$MSE = \frac{1}{5} \left[(6 - 5.5)^2 + (7 - 5.0)^2 + \ldots + (3.5 - 2)^2 \right] = 1.5$$
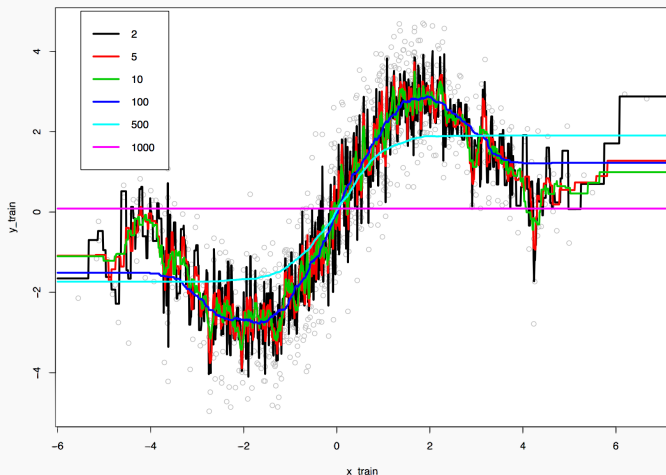
On average, our predictions are off by $15.

kNN Regression: A Simple Example

We plot the observed responses along with predicted responses for comparison:

But what value of k should we choose? What would our predicted responses look like if k is very small? What if k is large (e.g. $k = n$)?

kNN with Multiple Predictors

In our simple example, we used absolute value to measure the distance between the predictors in two different observations, $|x_i - x_j|$.

When we have multiple predictors in each observation, we need a notion of distance between two **sets** of predictor values. Typically, we use Euclidean distance:

$$d(x_i - x_j) = \sqrt{(x_{i,1} - x_{j,1})^2 + \ldots + (x_{i,p} - x_{j,p})^2}$$
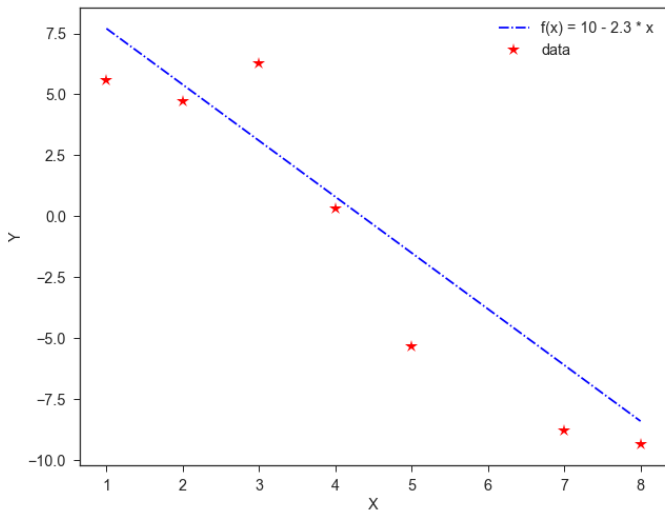
Caution: when using Euclidean distance, the scale (or units) of measurement for the predictors matter! Predictors with large values, comparatively, will dominate the distance measurement.

Model II: Linear Regression
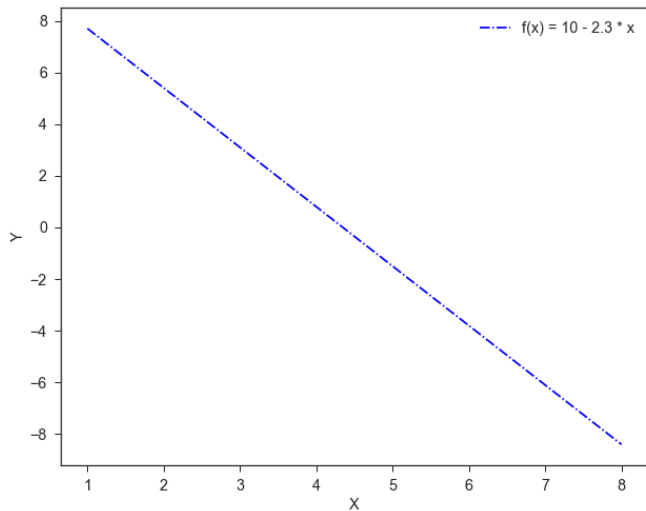
$f(x) = 10 - 2.3 * x$
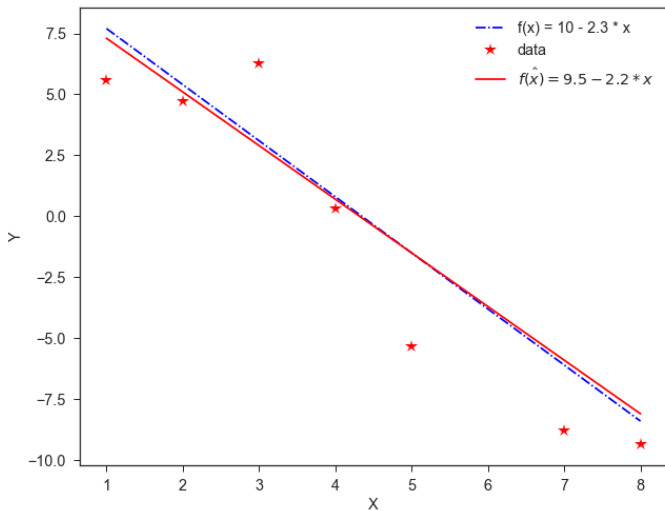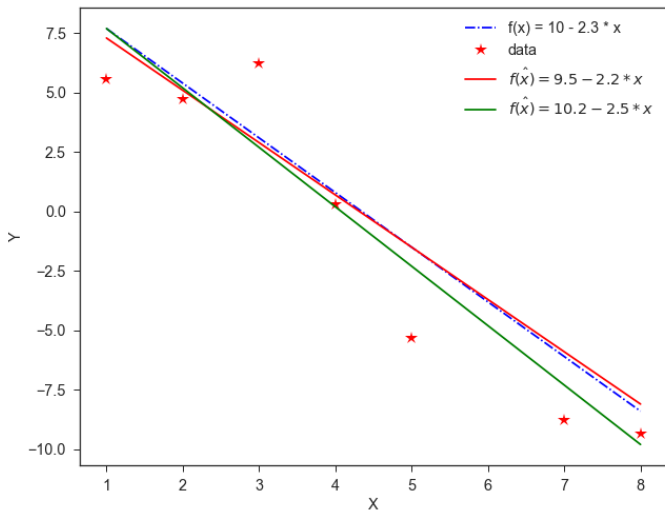
Note that in building our kNN model for prediction, we did not compute a closed form for $\widehat{f}$, our estimate of the function, f, relating predictor to response.

Alternatively, if each observation has only one predictor, we can build a model by first assuming a simple form for f (and hence $\widehat{f}$), say a *linear form*,

$$Y = f(X) + \epsilon = \beta_1 X + \beta_0 + \epsilon.$$

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

If our statistical model is

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

then it follows that our estimate is

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

where $\widehat{\beta_1}$ and $\widehat{\beta_0}$ are estimates of β_1 and β_0, respectively, that we compute using observations.

Recall that our intuition says to choose $\widehat{\beta_1}$ and $\widehat{\beta_0}$ in order to minimize the predictive errors made by our model, i.e. minimize our loss function.

Inference for Linear Regression

Again we use MSE as our loss function,

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

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

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

Now, taking the partial derivatives of L and finding the global minimum will give us explicit formulae for $\widehat{\beta_0}, \widehat{\beta_1}$,

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

where $\overline{y}$ and $\overline{x}$ are sample means. The line $\widehat{Y} = \widehat{\beta_1} X + \widehat{\beta_0}$ is called the **regression line**.

Linear Regression: A Simple Example

Recall our simple example from before, where we observe the average cab fare in NYC using the time of day,

X	1	2	3	4	5
Y	6	7	4	3	2

By our formula, we compute the regression line to be

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

Using this model, we can generate predicted responses:
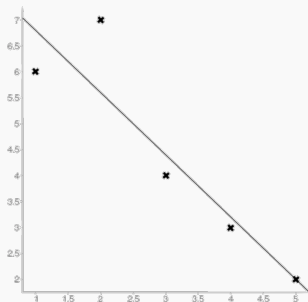
$$\widehat{Y} = (6.8, 5.6, 4.4, 3.2, 2.0)$$

Let's graph our linear model against the observations.

Linear Regression: A Simple Example

Why doesn't our line fit the observations exactly? There are two possibilities:

- f is not a linear function
- the difference between prediction and observation is due to the noise term in $Y = f(X) + \epsilon$.

Regardless of the form of f, the presence of the random term ϵ means that the predictions made using $\widehat{f}$ will never exactly match the observations.

Question: Is it possible to measure how confidently $\widehat{\beta}_0$, $\widehat{\beta}_1$ approximate the true parameters of f?

Evaluating Model Performance

We interpret the ϵ term in our observation

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

to be noise introduced by random variations in natural systems or imprecisions of our scientific instruments.

We call ϵ the measurement error or **irreducible error**. Since even predictions made with the actual function f will not match observed values of Y.

Due to ϵ, every time we measure the response Y for a fix value of X we will obtain a different observation, and hence a different estimate of β_0 and β_1.

Measurement vs. Sampling Error

Again due to ϵ, if we make only a few observations, the noise in the observed values of Y will have a large impact on our estimate of β_0 and β_1.

If we make many observations, the noise in the observed values of Y will 'cancel out' - noise that bias some observations towards higher values will be canceled with noise that bias other observations towards lower values.

This feels intuitively true but requires some assumptions on ϵ and a formal justification - or at least an example.

Measurement vs. Sampling Error

In summary, the variations in $\widehat{\beta}_0$, $\widehat{\beta}_1$ (estimates of β_0 and β_1 respectively) are affected by

- (**Measurement**) $\text{Var}[\epsilon]$, the variance (the scale of the variation) in the noise, ϵ

- (**Sampling**) n, the number of observation we make

The variances of $\widehat{\beta}_0$, $\widehat{\beta}_1$ are also called *standard errors*, which we will see in the next lecture.

Bootstrapping for Estimating Sampling Error

With some assumption on ϵ, we can compute the variances or standard errors of $\widehat{\beta}_0$ and $\widehat{\beta}_1$ analytically.

The standard errors can also be estimated empirically through *bootstrapping*.

Definition

Bootstrapping is the practice of estimating properties of an estimator by measuring those properties by, for example, sampling from the observed data.

For example, we can compute $\widehat{\beta}_0$ and $\widehat{\beta}_1$ multiple times by randomly sampling from our data set. We then use the variance of our multiple estimates to approximate the true variance of $\widehat{\beta}_0$ and $\widehat{\beta}_1$.

Training vs. Testing Sets

One more way to evaluate our model is to use it to predict the responses for predictors that we did not use to build our model.

Typically, after collecting a set of observations of predictor and response, we split the data into a *training set* and a *testing set*.

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.

Caution: In order to maintain the integrity of the final test, you should use your test data once and must not use the results to inform changes you make to the model.

Comparison of Two Models

Linear regression is an example of *a parametric model*, that is, it is a model with a fixed form and a fixed number of parameters that does not depend on the number of observations in the training set.

kNN is an example of a *non-parametric model*, that is, it is a model whose structure depends on the data. The set of parameters of the kNN model is the entire training set.

In particular, the number of parameters in kNN depends on the number of observations in the training set.

kNN vs. Linear Regression

So which model is better? Rather than answer this question, let's define 'better'.

To compare two models, we can consider any combination of the following criteria (and possibly more):

- ▶ Which model gives less predictive error, with respect to a loss function?
- ▶ Which model takes less space to store?
- ▶ Which model takes less time to train (perform inference)?
- ▶ Which model takes less time to make a prediction?

Bibliography

1. Bolelli, L., Ertekin, S., and Giles, C. L. *Topic and trend detection in text collections using latent dirichlet allocation.* In European Conference on Information Retrieval (2009), Springer, pp. 776-780.
2. Chen, W., Wang, Y., and Yang, S. *Efficient influence maximization in social networks. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (2009)*, ACM, pp. 199-208.
3. Chong, W., Blei, D., and Li, F.-F. *Simultaneous image classification and annotation. In Computer Vision and Pattern Recognition*, 2009. CVPR 2009. IEEE Conference on (2009), IEEE, pp. 1903-1910.
4. Du, L., Ren, L., Carin, L., and Dunson, D. B. *A bayesian model for simultaneous image clustering, annotation and object segmentation.* In Advances in neural information processing systems (2009), pp. 486-494.
5. Elango, P. K., and Jayaraman, K. *Clustering images using the latent dirichlet allocation model.*
6. Feng, Y., and Lapata, M. *Topic models for image annotation and text illustration.* In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (2010), Association for Computational Linguistics, pp. 831-839.
7. Hannah, L. A., and Wallach, H. M. *Summarizing topics: From word lists to phrases.*
8. Lu, R., and Yang, Q. *Trend analysis of news topics on twitter.* International Journal of Machine Learning and Computing 2, 3 (2012), 327.