Learning goals:

- Reframe loss minimization framework for modeling.
- Introduce multivariable linear models within the loss minimization framework.

Linear Regression (Reading: Ch 13)

UC Berkeley Data 100 Summer 2019
Sam Lau

(Slides adapted from Sandrine Dudoit and Joey Gonzalez)

Announcements

- HW4 due **Tuesday**
- HW5 out Tuesday, due **Friday**
- Leo was out sick! Hopefully back today.
- Today's lecture might get split into two
    - Ask lots of questions if we're going too fast

Last Time

- Draw conclusions about population using a sample through statistical estimation.
- Make estimations by picking the estimator that minimizes empirical risk / loss.
- Today:
    - Connect estimation with prediction and modeling.
    - First foray into machine learning with linear models.

Modeling

Making Predictions

- Loss minimization framework useful for predictions too!
- Suppose we have a dataset of cars and we'd like to predict fuel efficiency (miles per gallon, or mpg):

	mpg	cylinders	displacement	horsepower	...	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	...	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	...	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	...	11.0	70	usa	plymouth satellite

Models

- To make a prediction, we choose a **model**.
  - Takes input data and outputs a prediction.
- Constant model:

Prediction $\longrightarrow f_\theta(x) = \theta \longleftarrow$ **Recipe to compute the prediction**

$\uparrow$

Input data

- Simple linear model:

Two model weights

$$f_\theta(x) = \theta_1 x + \theta_0$$

The Constant Model

- Start simple: if constant model, how do we pick θ?

$$f_\theta(x) = \theta$$

- Intuition: pick θ to be close to most of the values in data

	mpg	cylinders	displacement	horsepower	...	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	...	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	...	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	...	11.0	70	usa	plymouth satellite

Model Loss

- Use x_i to denote what we use to make predictions
- Use y_i to denote what we're trying to predict
- But both x and y come from a single sample
- Idea: Pick the θ that minimizes the average loss between y in our sample and model predictions.

$$L(\theta, y_1, \ldots, y_n) = \frac{1}{n}\sum(y_i - f_\theta(x))^2$$

Constant Model Loss

$$L(\theta, y_1, \ldots, y_n) = \frac{1}{n}\sum(y_i - f_\theta(x))^2$$

Since $f_\theta(x) = \theta$ for constant model:

$$L(\theta, y_1, \ldots, y_n) = \frac{1}{n}\sum(y_i - \theta)^2$$

- Remember this expression from last lecture?
- θ = sample mean is the best model parameter.
- So, for car MPGs, we set θ = mean(mpg)

Modeling is Estimation in New Clothes

- Estimation: making best guess at population parameter
- Modeling: making predictions for population values
- Two sides of the same coin! Why?
- Modeling assumes pop values generated by parameters:

$$f_\theta^*(x) = \theta^* + \epsilon$$

- RTA: assume that data from population generated by taking a constant θ^* and adding noise ϵ.
- Estimation = Finding $\hat{\theta}$, our best estimate for θ^*
- Modeling = Using $\hat{\theta}$ to make predictions

The Modeling Pipeline

We choose what goes in the blue boxes!

Input Data

Car 1, 13 mpg
Car 2, 20 mpg
Car 3, 18 mpg

Model

$$\boxed{f_\theta(x) = \theta}$$

Model Weight(s)

$$\boxed{\theta = 15}$$

Predictions

15
15
15

Loss Function

$$\boxed{L(\theta) = (y_i - \theta)^2}$$

Loss

4
25
9

Fit a model by finding weights that minimize loss. Minimizing sample loss approximates minimizing population loss.

The Modeling Recipe

- Pick a model, pick a loss function, fit the model to sample.
- Preview of model and loss function combos:

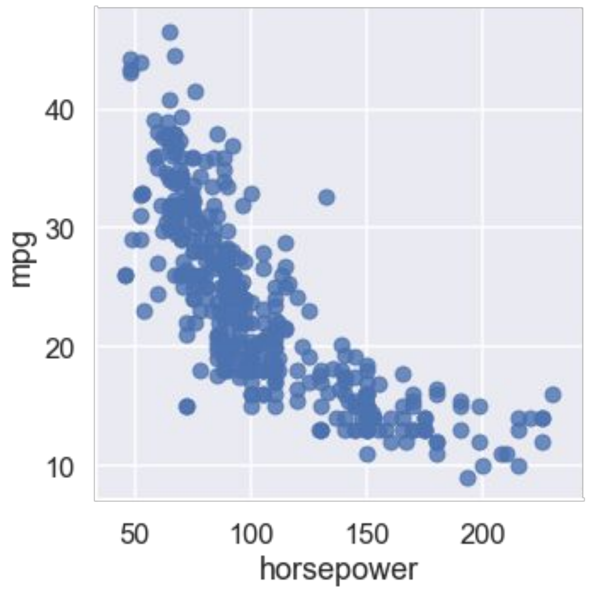
Model	Loss Function	Technique Name
$\boldsymbol{\theta} \cdot \boldsymbol{x}$	$\frac{1}{n} \sum (y_i - f_\theta(\boldsymbol{x}))^2$	Least squares linear regression
$\boldsymbol{\theta} \cdot \boldsymbol{x}$	$\frac{1}{n} \sum (y_i - f_\theta(\boldsymbol{x}))^2 + \lambda \lvert\boldsymbol{\theta}\rvert^2$	Lasso regression
$\boldsymbol{\theta} \cdot \boldsymbol{x}$	$\frac{1}{n} \sum (y_i - f_\theta(\boldsymbol{x}))^2 + \lambda \lvert\lvert\boldsymbol{\theta}\rvert\rvert_{\ell_1}$	Ridge regression
$\boldsymbol{\theta} \cdot \boldsymbol{x}$	$\frac{1}{n} \sum \lvert y_i - f_\theta(\boldsymbol{x}) \rvert$	Least absolute deviations
$\sigma(\boldsymbol{\theta} \cdot \boldsymbol{x})$	$\frac{1}{n} \sum [-y_i \ln f_\theta(\boldsymbol{x}) - (1 - y_i) \ln(1 - f_\theta(\boldsymbol{x}))]$	Logistic regression

Linear Models

Using Our Data

- If we’re trying to predict MPG, we can do better than a constant model by incorporating more information.
    - E.g. higher horsepowers have lower MPGs:

	mpg	cylinders	displacement	horsepower
0	18.0	8	307.0	130.0
1	15.0	8	350.0	165.0
2	18.0	8	318.0	150.0



Simple Linear Model

- We want our predictions to depend on the input data x.
- Simple linear model:

$$f^*_\theta(x) = \theta^*_1 x + \theta^*_0 + \epsilon \qquad \epsilon \sim N(0, \sigma^2)$$

$$f_\theta(x) = \theta_1 x + \theta_0$$

- As usual, we can minimize the loss. This time, we have two parameters.

$$\begin{aligned} L(\theta_1, \theta_0, y_1, \ldots, y_n) &= \frac{1}{n}\sum (y_i - f_\theta(x_i))^2 \\ &= \frac{1}{n}\sum (y_i - \theta_1 x_i - \theta_0)^2 \end{aligned}$$

Simple Linear Model

$$L(\theta_1, \theta_0, y_1, \ldots, y_n) = \frac{1}{n}\sum(y_i - \theta_1 x_i - \theta_0)^2$$

$$\frac{\partial}{\partial \theta_1} L = \frac{1}{n}\sum 2(y_i - \theta_1 x_i - \theta_0)(-x_i)$$

$$\frac{\partial}{\partial \theta_0} L = \frac{1}{n}\sum 2(y_i - \theta_1 x_i - \theta_0)(-1)$$

- This ends up being a lot of algebra, so we'll skip to the answer.

Skipping Ahead

Let r be the average of the products of x and y when both variables are measured in standard units.

$$\hat{\theta}_1 = r \cdot \frac{\text{SD of } y}{\text{SD of } x} \qquad \hat{\theta}_0 = \text{mean of } y - \hat{\theta}_1 \cdot \text{mean of } x$$

- Data 8 textbook has example slope/intercept calculations.
- Takeaway: Can derive these formulas by minimizing loss.
- You should know how to take the derivative but won't need to solve it.

Multivariable Linear Model

- Simple linear model uses one variable to predict:

$$f_\theta(x) = \theta_1 x + \theta_0$$

- Time to graduate from Data 8!
- Multivariable linear model uses ≥1 variable:

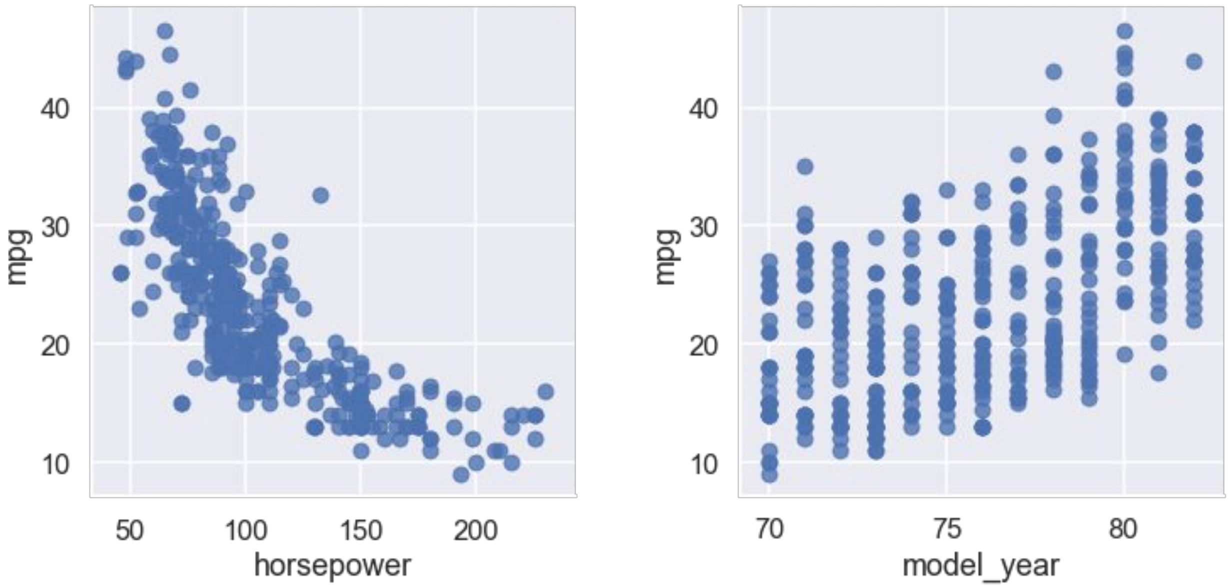
$$f_\theta(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_p x_p$$

- **x** is a vector containing one row of input data.
- IOW: Predict by combining multiple features together.

Intuition

$$f_\theta(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_p x_p$$

- Using horsepower and model year to predict mpg
  - Expect θ_1 to be negative and θ_2 to be positive. Why?



Using Matrix Multiplication

$$f_\theta(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_p x_p$$

- Many terms to write! We'll use a trick: add a column of 1s to the table:

	mpg	bias	horsepower	weight	model_year
0	18.0	1	130.0	3504	70
1	15.0	1	165.0	3693	70
2	18.0	1	150.0	3436	70

$$\boldsymbol{x} = [1, x_1, x_2, x_3]$$

$$\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \theta_3]$$

Bolded letters means vector or matrix.

- This means our model is: $f_\theta(\boldsymbol{x}) = \boldsymbol{\theta} \cdot \boldsymbol{x}$

More Notation!

$\boldsymbol{y}$: column vector of sample points to predict

$\boldsymbol{X}$: matrix of input data $(n \times p)$

	mpg	bias	horsepower	weight	model_year
0	18.0	1	130.0	3504	70
1	15.0	1	165.0	3693	70
2	18.0	1	150.0	3436	70
...	...	...	...	...	...
395	32.0	1	84.0	2295	82
396	28.0	1	79.0	2625	82
397	31.0	1	82.0	2720	82

$\boldsymbol{X}_i$: row i of input data

$\boldsymbol{x}$: row vector of input data

$\boldsymbol{\theta}$: vector of model weights

$\hat{\boldsymbol{\theta}}$: loss-minimizing model weights

Your turn: Write the matrix expression that computes a vector with a fitted linear model's predictions for **all sample points**.

Your Turn

Write the matrix expression that computes a vector with a fitted linear model's predictions for all sample points.

Prediction for one point: $\hat{\boldsymbol{\theta}} \cdot \boldsymbol{x}$

$$\text{Prediction for all points: } \hat{\boldsymbol{y}} = \begin{bmatrix} \hat{\boldsymbol{\theta}} \cdot \boldsymbol{X}_1 \\ \hat{\boldsymbol{\theta}} \cdot \boldsymbol{X}_2 \\ \vdots \\ \hat{\boldsymbol{\theta}} \cdot \boldsymbol{X}_n \end{bmatrix} = \underset{(n \times p)(p \times 1) = (n \times 1)}{\boldsymbol{X}\hat{\boldsymbol{\theta}}}$$

Your Turn

Write the matrix expression that computes the average MSE loss for all data points (this is a scalar!).

Your Turn

Write the matrix expression that computes the average MSE loss for all data points (this is a scalar!).

$$L(\boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{y}) = \frac{1}{n} \sum (y_i - \boldsymbol{X}_{\boldsymbol{i}} \cdot \boldsymbol{\theta})^2$$

$$= \frac{1}{n} ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}||^2$$

(Bonus points if you got this)

$$||\boldsymbol{v}||^2 = \boldsymbol{v} \cdot \boldsymbol{v} = \boldsymbol{v}^\top \boldsymbol{v}$$

Using matrix notation takes a lot of practice to get used to, but the results are worth it. Always check your dimensions!

Fitting a Linear Model

- How do we pick θ to minimize loss?

$$L(\boldsymbol{\theta}, \mathbf{X}, \boldsymbol{y}) = \frac{1}{n} \sum (y_i - \boldsymbol{X_i} \cdot \boldsymbol{\theta})^2$$

- Want to take partial derivatives for θ_0, θ_1, ...
- Instead, we'll take the **gradient** and set it equal to zero.
- This solves for all model weights at once!

$$\nabla_{\boldsymbol{\theta}} L = \begin{bmatrix} \frac{\partial}{\partial \theta_0}(L) \\ \frac{\partial}{\partial \theta_1}(L) \\ \vdots \\ \frac{\partial}{\partial \theta_p}(L) \end{bmatrix}$$

The Normal Equation

- Saving the setup for the Gradient Descent lecture
  - Again, you need to know how to take the gradient but not how to solve for **θ**.
- Skipping ahead to the answer:

$$\boldsymbol{X}^\top \boldsymbol{X}\hat{\boldsymbol{\theta}} = \boldsymbol{X}^\top \boldsymbol{y}$$
$$\hat{\boldsymbol{\theta}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1}\boldsymbol{X}^\top \boldsymbol{y}$$

What are the matrix shapes in these expressions?

- Expression above called normal equation
- Gives a closed-form recipe for fitting linear model

The Abnormal Equation

- In practice, it takes too long to compute this:

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

- Inverting an (n x n) matrix takes at least $O(n^2)$ time.
  - State of the art: $O(n^{2.3})$
- Takeaway: analytic solutions are elegant but are sometimes hard to find and slow.
  - Next lecture: gradient descent

Demo: Predicting MPGs

Break!

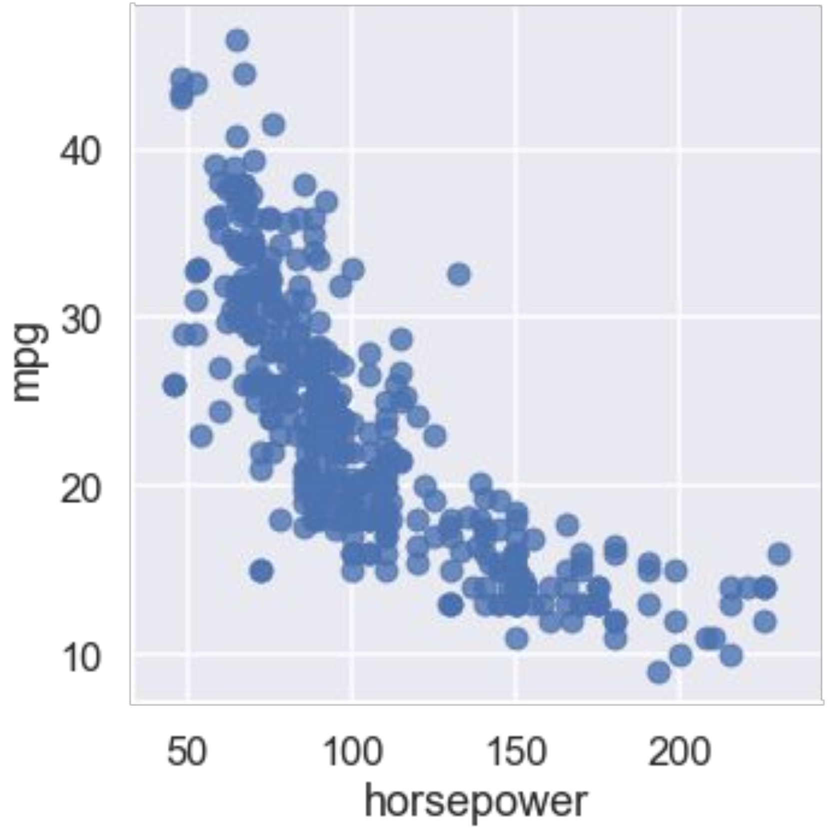
Fill out Attendance:

http://bit.ly/at-d100

Feature Engineering
(moved to Wed lecture)

Linear Models Level Up

- Horsepower and mpg have a nonlinear relationship.
- Can still use linear regression to capture this!
- **Feature engineering**: creating new features from data to give model more complexity.



Adding Features

- For now, predict MPG from horsepower alone.
- Insight: Add a new column to X with horsepower^2.

	bias	hp	hp^2
0	1	130.0	16900.0
1	1	165.0	27225.0
2	1	150.0	22500.0
...	...	...	...
395	1	84.0	7056.0
396	1	79.0	6241.0
397	1	82.0	6724.0

- Now we fit a quadratic function!

$$\begin{aligned} f_\theta(\boldsymbol{x}) &= \theta_0 + \theta_1 x_1 + \theta_2 x_2 \\ &= \theta_0 + \theta_1 \mathrm{hp} + \theta_2 \mathrm{hp}^2 \end{aligned}$$

- This is still linear in model weights θ, so we call it a linear model.

(Demo)

Polynomial Regression

- For polynomial features of degree n, usually add every possible combination of columns.
    - 4 original columns, degree 2:

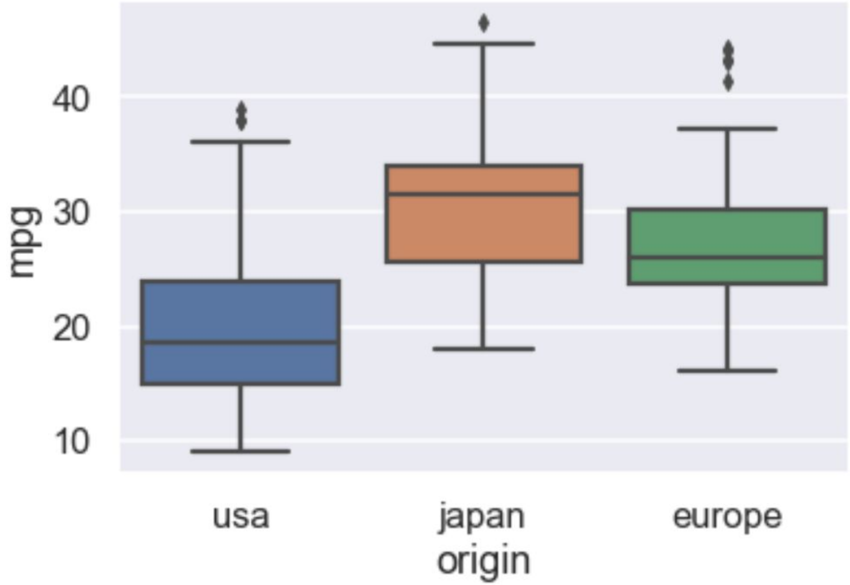
$$x_1, x_2, x_3, x_4,$$
$$x_1^2, x_2^2, x_3^2, x_4^2,$$
$$x_1x_2, x_1x_3, x_1x_4, x_2x_3, x_2x_4, x_3x_4$$

- Can end up being a lot of columns
- To cope, use kernel trick (covered in advanced courses)

Categorical Features

- Origin column is correlated with MPG. Can we use it?
- Idea: Encode categories as numbers in a smart way.
- Discuss: Why can't we just encode "usa" as 0, "japan" as 1, "europe" as 2?

```
sns.boxplot(x='origin', y='mpg', data=mpg);
```



One-Hot Encoding

- One-hot encoding makes one new column for each unique category:

origin
usa
usa
europe
...
usa
japan
japan

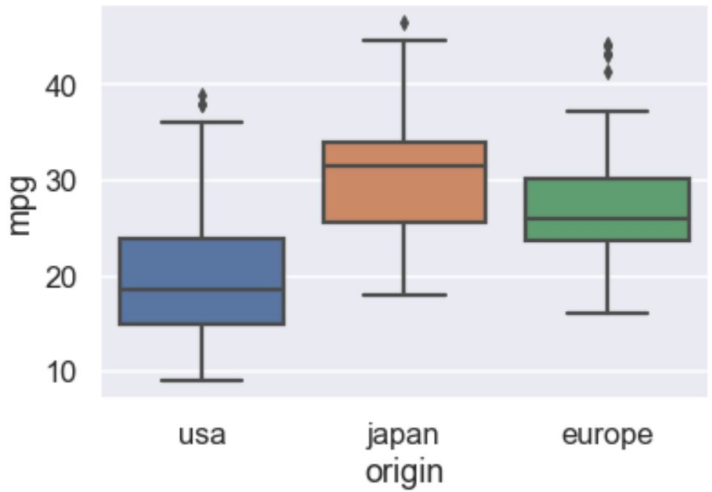
origin=usa	origin=europe	origin=japan
1	0	0
1	0	0
0	1	0
...	...	...
1	0	0
0	0	1
0	0	1

One-Hot Encoding

- What do you expect the largest weight to be?

origin=usa	origin=europe	origin=japan
1	0	0
1	0	0
0	1	0
...	...	...
1	0	0
0	0	1
0	0	1

```
sns.boxplot(x='origin', y='mpg', data=mpg);
```



- Can interpret weight as “contribution” of that category

One Hot Problem

- Problem: Adding a new column for each category makes columns of X **linearly dependent**! Why?
- One-hot columns always sum to 1:

bias		origin=usa		origin=europe		origin=japan
1		1		0		0
1		1		0		0
1		0		1		0
...	=	...	+	...	+	...
1		1		0		0
1		0		0		1
1		0		0		1

- This makes normal equations unsolvable.

$$\hat{\boldsymbol{\theta}} = \boxed{(\boldsymbol{X}^\top \boldsymbol{X})^{-1}} \boldsymbol{X}^\top \boldsymbol{y}$$

Not invertible ^

Weight Interpretation

- Invertibility isn't a problem for gradient descent, but this still affects how we interpret the model weights.
- Linearly dependent columns can "swap" weights:
  - Left: All categories matter. Right: No categories matter!

0	3	3	3
bias	**origin=usa**	**origin=europe**	**origin=japan**
1	1	0	0
1	1	0	0
1	0	1	0
...	...	...	...
1	1	0	0
1	0	0	1

=

3	0	0	0
bias	**origin=usa**	**origin=europe**	**origin=japan**
1	1	0	0
1	1	0	0
1	0	1	0
...	...	...	...
1	1	0	0
1	0	0	1

Drop it Like it's Hot

- Simple fix: Drop the last one-hot column.
- In this case, the weight for USA can be interpreted as "change in MPG between USA and Japan".

origin
usa
usa
europe
...
usa
japan

origin=usa	origin=europe
1	0
1	0
0	1
...	...
1	0
0	0

Features feat. More Features

- Feature engineering is often domain-specific:
  - Standardizing: “How many SDs away from average?”
  - Log transform: Used to fit exponential models.
  - Absolute difference: “How different is the current temperature from 70°?”
  - Binning data, then one-hot encoding: “Are we driving during morning rush hour? Evening rush hour?”
  - Date-related features: year, month, weekday
  - Image-related features: blurring, edge detection, etc.

Summary

- Modeling and estimation are closely related.
  - We can view modeling as estimation of model parameters.
- Linear models can incorporate an arbitrary number of features to make a prediction.
- Feature engineering extends linear models to generate more complex models.