Let $X \sim \mathcal{N}(\mu, \Sigma)$ where $X \in \mathbb{R}^d$

1. Let $Y = AX + b$. For what $\tilde{\mu}, \tilde{\Sigma}$ is $Y \sim \mathcal{N}(\tilde{\mu}, \tilde{\Sigma})$

2. Suppose I can generate independent Gaussians $Z \sim \mathcal{N}(0, 1)$ (e.g., `numpy.random.randn`). How can I use this to generate $X$?
Regularization in Linear Regression

Recall Least Squares: 
\[ \hat{w}_{LS} = \arg \min_w \sum_{i=1}^{n} (y_i - x_i^T w)^2 \]
\[ = \arg \min_w (y - Xw)^T (y - Xw) \]
when \((X^T X)^{-1}\) exists….
\[ = (X^T X)^{-1} X^T y \]
Regularization in Linear Regression

Recall Least Squares:

$$\hat{w}_{LS} = \arg \min_w \sum_{i=1}^{n} (y_i - x_i^T w)^2$$

$$= \arg \min_w (y - Xw)^T (y - Xw)$$

In general:

$$= \arg \min_w w^T (X^T X)w - 2y^T X w$$
Regularization in Linear Regression

Recall Least Squares: \( \hat{w}_{LS} = \arg \min_w \sum_{i=1}^{n} (y_i - x_i^T w)^2 \)

\[ = \arg \min_w (y - Xw)^T (y - Xw) \]

\[ = \arg \min_w w^T (X^T X)w - 2y^T Xw \]

In general:

\[ (y_1 - x_1^T w)^2 + (y_2 - x_2^T w)^2 + \cdots + (y_n - x_n^T w)^2 = \sum_{i=1}^{n} (y_i - x_i^T w)^2 \]

What if \( x_i \in \mathbb{R}^d \) and \( d > n \)?
Regularization in Linear Regression

Recall Least Squares: \[ \hat{w}_{LS} = \arg \min_w \sum_{i=1}^{n} (y_i - x_i^T w)^2 \]

When \( x_i \in \mathbb{R}^d \) and \( d > n \) the objective function is flat in some directions:
Regularization in Linear Regression

Recall Least Squares: \[ \hat{w}_{LS} = \arg \min_w \sum_{i=1}^{n} (y_i - x_i^T w)^2 \]

When \( x_i \in \mathbb{R}^d \) and \( d > n \) the objective function is flat in some directions:

Implies optimal solution is underconstrained and unstable due to lack of curvature:

- small changes in training data result in large changes in solution
- often the magnitudes of \( w \) are “very large”

Regularization imposes “simpler” solutions by a “complexity” penalty
Ridge Regression

- Old Least squares objective:

\[
\hat{w}_{LS} = \arg \min_w \sum_{i=1}^{n} (y_i - x_i^T w)^2
\]

- Ridge Regression objective:

\[
\hat{w}_{ridge} = \arg \min_w \sum_{i=1}^{n} (y_i - x_i^T w)^2 + \lambda \|w\|_2^2
\]
Minimizing the Ridge Regression Objective

\[ \hat{w}_{ridge} = \arg \min_w \sum_{i=1}^{n} (y_i - (x_i^T w + b))^2 + \lambda \| w \|_2^2 \]

\[ = \arg \min_w \| y - (Xw + 1b) \|_2^2 + \lambda \| w \|_2^2 \]
Shrinkage Properties

\[ \hat{w}_{ridge} = (X^T X + \lambda I)^{-1} X^T y \]

- If orthonormal features/basis: \( X^T X = I \)
Ridge Regression: Effect of Regularization

\[ \hat{w}_{ridge} = \arg \min_w \sum_{i=1}^{n} (y_i - (x_i^T w + b))^2 + \lambda ||w||^2 \]

- Solution is indexed by the regularization parameter \( \lambda \)
- Larger \( \lambda \)
  - Smaller \( \lambda \)
  - As \( \lambda \to 0 \)
  - As \( \lambda \to \infty \)
Ridge Regression: Effect of Regularization

\[ D \overset{i.i.d.}{\sim} P_{XY} \]

\[ \hat{w}_D^{(\lambda)} = \arg \min_w \frac{1}{|D|} \sum_{(x_i, y_i) \in D} (y_i - x_i^T w)^2 + \lambda \|w\|_2^2 \]

**TRAIN error:**

\[
\frac{1}{|D|} \sum_{(x_i, y_i) \in D} (y_i - x_i^T \hat{w}_D^{(\lambda)})^2
\]

**TRUE error:**

\[
\mathbb{E}[(Y - X^T \hat{w}_D^{(\lambda)})^2]
\]

**TEST error:**

\[ T \overset{i.i.d.}{\sim} P_{XY} \]

\[
\frac{1}{|T|} \sum_{(x_i, y_i) \in D} (y_i - x_i^T \hat{w}_D^{(\lambda)})^2
\]

**Important:** \( D \cap T = \emptyset \)
Ridge Regression: Effect of Regularization

\[ \mathcal{D} \overset{i.i.d.}{\sim} P_{XY} \]

\[ \hat{w}_{D,\text{ridge}}^{(\lambda)} = \arg\min_w \frac{1}{|D|} \sum_{(x_i,y_i) \in D} (y_i - x_i^T w)^2 + \lambda \|w\|^2 \]

**TRAIN error:**

\[ \frac{1}{|D|} \sum_{(x_i,y_i) \in D} (y_i - x_i^T \hat{w}_{D,\text{ridge}}^{(\lambda)})^2 \]

**TRUE error:**

\[ \mathbb{E}[(Y - X^T \hat{w}_{D,\text{ridge}}^{(\lambda)})^2] \]

**TEST error:**

\[ \mathcal{T} \overset{i.i.d.}{\sim} P_{XY} \]

\[ \frac{1}{|T|} \sum_{(x_i,y_i) \in D} (y_i - x_i^T \hat{w}_{D,\text{ridge}}^{(\lambda)})^2 \]

Each line is i.i.d. draw of \( D \) or \( T \)

**Important:** \( D \cap T = \emptyset \)

©2017 Kevin Jamieson
Ridge Coefficient Path

- Typical approach: select $\lambda$ using cross validation, up next

From Kevin Murphy textbook
What you need to know…

- Regularization
  - Penalizes for complex models
- Ridge regression
  - $L_2$ penalized least-squares regression
  - Regularization parameter trades off model complexity with training error
Cross-Validation

Machine Learning – CSE546
Kevin Jamieson
University of Washington

October 10, 2016
How... How... How???????

- *How do we pick the regularization constant* $\lambda$...
- *How do we pick the number of basis functions*...

- We could use the test data, but...
How... How... How???????

- How do we pick the regularization constant $\lambda$...
- How do we pick the number of basis functions...

- We could use the test data, but...

Consider a validation set with 1 example:
- $D$ – training data
- $D\setminus j$ – training data with $j$th data point $(x_j, y_j)$ moved to validation set

Learn classifier $f_{D\setminus j}$ with $D\setminus j$ dataset

Estimate true error as squared error on predicting $y_j$:
- Unbiased estimate of $\text{error}_{\text{true}}(f_{D\setminus j})$
(LOO) Leave-one-out cross validation

- Consider a validation set with 1 example:
  - $D$ – training data
  - $D \setminus j$ – training data with $j$th data point $(x_j, y_j)$ moved to validation set

- Learn classifier $f_{D\setminus j}$ with $D \setminus j$ dataset

- Estimate true error as squared error on predicting $y_j$:
  - Unbiased estimate of $\text{error}_{\text{true}}(f_{D\setminus j})$

- **LOO cross validation**: Average over all data points $j$:
  - For each data point you leave out, learn a new classifier $f_{D\setminus j}$
  - Estimate error as:
  $$\text{error}_{\text{LOO}} = \frac{1}{n} \sum_{j=1}^{n} (y_j - f_{D \setminus j}(x_j))^2$$
LOO cross validation is (almost) unbiased estimate of true error of $h_D$!

- When computing **LOOCV** error, we only use $N-1$ data points
  - So it’s not estimate of true error of learning with $N$ data points
  - Usually pessimistic, though – learning with less data typically gives worse answer

- LOO is almost unbiased! Use LOO error for model selection!!!
  - E.g., picking $\lambda$
Computational cost of LOO

- Suppose you have 100,000 data points
- You implemented a great version of your learning algorithm
  - Learns in only 1 second
- Computing LOO will take about 1 day!!!
Use $k$-fold cross validation

- Randomly divide training data into $k$ equal parts
  - $D_1, \ldots, D_k$
- For each $i$
  - Learn classifier $f_{D \setminus D_i}$ using data point not in $D_i$
  - Estimate error of $f_{D \setminus D_i}$ on validation set $D_i$:
    \[
    \text{error}_{D_i} = \frac{1}{|D_i|} \sum_{(x_j, y_j) \in D_i} (y_j - f_{D \setminus D_i}(x_j))^2
    \]

©2017 Kevin Jamieson
Use $k$-fold cross validation

- Randomly divide training data into $k$ equal parts
  - $D_1, \ldots, D_k$
- For each $i$
  - Learn classifier $f_{D \setminus D_i}$ using data point not in $D_i$
  - Estimate error of $f_{D \setminus D_i}$ on validation set $D_i$:
    $$\text{error}_{D_i} = \frac{1}{|D_i|} \sum_{(x_j, y_j) \in D_i} (y_j - f_{D \setminus D_i}(x_j))^2$$
- $k$-fold cross validation error is average over data splits:
  $$\text{error}_{k\text{-fold}} = \frac{1}{k} \sum_{i=1}^{k} \text{error}_{D_i}$$

- $k$-fold cross validation properties:
  - Much faster to compute than LOO
  - More (pessimistically) biased – using much less data, only $n(k-1)/k$
  - Usually, $k = 10$
Recap

- Given a dataset, begin by splitting into

- **Model selection**: Use k-fold cross-validation on **TRAIN** to train predictor and choose magic parameters such as $\lambda$

- **Model assessment**: Use **TEST** to assess the accuracy of the model you output
  - Never ever ever ever ever train or choose parameters based on the test data
Example

- Given 10,000-dimensional data and \( n \) examples, we pick a subset of 50 dimensions that have the highest correlation with labels in the training set:

\[
\begin{align*}
&\text{50 indices } j \text{ that have largest} \\
&\frac{\left| \sum_{i=1}^{n} x_{i,j} y_i \right|}{\sqrt{\sum_{i=1}^{n} x_{i,j}^2}}
\end{align*}
\]

- After picking our 50 features, we then use CV to train ridge regression with regularization \( \lambda \)

- What’s wrong with this procedure?
Bootstrap

Machine Learning – CSE546
Kevin Jamieson
University of Washington

October 10, 2016
Limitations of CV

- An 80/20 split throws out a relatively large amount of data if only have, say, 20 examples.
- Test error is informative, but how accurate is this number? (e.g., 3/5 heads vs. 30/50)
- How do I get confidence intervals on statistics like the median or variance of a distribution?
- Instead of the error for the entire dataset, what if I want to study the error for a particular example $x$?
Limitations of CV

- An 80/20 split throws out a relatively large amount of data if only have, say, 20 examples.
- Test error is informative, but how accurate is this number? (e.g., 3/5 heads vs. 30/50)
- How do I get confidence intervals on statistics like the median or variance of a distribution?
- Instead of the error for the entire dataset, what if I want to study the error for a particular example $x$?


“The most important innovation in statistics of the last 40 years”
— famous ML researcher and statistician, 2015
Bootstrap: basic idea

Given dataset drawn iid samples with CDF $F_Z$:

$$D = \{z_1, \ldots, z_n\} \overset{i.i.d.}{\sim} F_Z$$

We compute a statistic of the data to get: $\hat{\theta} = t(D)$
Bootstrap: basic idea

Given dataset drawn iid samples with CDF $F_Z$:

$$\mathcal{D} = \{z_1, \ldots, z_n\} \overset{i.i.d.}{\sim} F_Z$$

We compute a statistic of the data to get: $\hat{\theta} = t(\mathcal{D})$

For b=1,...,B define the $b$th **bootstrapped** dataset as drawing $n$ samples **with replacement** from $\mathcal{D}$

$$\mathcal{D}^*b = \{z_1^*b, \ldots, z_n^*b\} \overset{i.i.d.}{\sim} \widehat{F}_{Z,n}$$

and the $b$th bootstrapped statistic as: $\theta^*b = t(\mathcal{D}^*b)$
Bootstrap: basic idea

Given dataset drawn iid samples with CDF $F_Z$:

$$
\mathcal{D} = \{z_1, \ldots, z_n\} \overset{i.i.d.}{\sim} F_Z 
\quad \hat{\theta} = t(\mathcal{D})
$$

For $b=1,\ldots,B$, samples sampled with replacement from $\mathcal{D}$

$$
\mathcal{D}^*b = \{z_1^*b, \ldots, z_n^*b\} \overset{i.i.d.}{\sim} \hat{F}_{Z,n} 
\quad \theta^*b = t(\mathcal{D}^*b)
$$
Bootstrap: basic idea

Given dataset drawn iid samples with CDF $F_Z$:

$$D = \{z_1, \ldots, z_n\} \overset{i.i.d.}{\sim} F_Z \quad \hat{\theta} = t(D)$$

For $b=1,\ldots,B$, samples sampled with replacement from $D$

$$D^* = \{z_1^b, \ldots, z_n^b\} \overset{i.i.d.}{\sim} F_{Z,n} \quad \theta^b = t(D^*)$$

$$\sup_{x} |\hat{F}_n(x) - F(x)| \to 0 \quad \text{as} \quad n \to \infty$$
Common applications of the bootstrap:
- Estimate parameters that escape simple analysis like the variance or median of an estimate
- Confidence intervals
- Estimates of error for a particular example:

\[ D, \hat{\theta}, \theta^{*b} \text{ for } b = 1, \ldots, 10 \]

95% confidence interval

Figures from Hastie et al
Advantages:
• Bootstrap is **very** generally applicable. Build a confidence interval around *anything*
• **Very** simple to use
• Appears to give meaningful results even when the amount of data is very small
• Very strong **asymptotic theory** (as num. examples goes to infinity)
Takeaways

Advantages:
• Bootstrap is very generally applicable. Build a confidence interval around anything
• Very simple to use
• Appears to give meaningful results even when the amount of data is very small
• Very strong asymptotic theory (as num. examples goes to infinity)

Disadvantages
• Very few meaningful finite-sample guarantees
• Potentially computationally intensive
• Reliability relies on test statistic and rate of convergence of empirical CDF to true CDF, which is unknown
• Poor performance on “extreme statistics” (e.g., the max)

Not perfect, but better than nothing.
Recap

- Learning is...
  - Collect some data
    - E.g., housing info and sale price
  - Randomly split dataset into TRAIN, VAL, and TEST
    - E.g., 80%, 10%, and 10%, respectively
  - Choose a hypothesis class or model
    - E.g., linear with non-linear transformations
  - Choose a loss function
    - E.g., least squares with ridge regression penalty on TRAIN
  - Choose an optimization procedure
    - E.g., set derivative to zero to obtain estimator, cross-validation on VAL to pick num. features and amount of regularization
  - Justifying the accuracy of the estimate
    - E.g., report TEST error with Bootstrap confidence interval