Lecture 6: Machine learning
• PS2 due Weds.
• Section this week: ML tutorial
• Reminder: GSIs can’t check answers.
• Szeliski 5.1
• Optional: Goodfellow Deep Learning “ML Basics”
Last week: hole filling

Source: A. Efros
[Hays and Efros “Scene Completion Using Millions of Photographs”, 2007.]
Simple inpainting [Efros & Leung 1999]
How would a human solve this?
2 Million Flickr Images

Source: A. Efros
... 200 scene matches

Source: A. Efros
Why does this work?
Nearest neighbors from a collection of 20 thousand images

Source: A. Efros
Nearest neighbors from a collection of 2 million images

Source: A. Efros
“Unreasonable Effectiveness of Data”

[Halevy, Norvig, Pereira 2009]

Parts of our world can be explained by elegant mathematics
  physics, chemistry, astronomy, etc.

But much cannot
  psychology, economics, genetics, etc.

Enter The Data!

“For many tasks, once we have a billion or so examples, we essentially have a closed set that represents (or at least approximates) what we need…”

Source: A. Efros
Learning

Data $\rightarrow$ Learner $\rightarrow$ Model

Inference

Input $\rightarrow$ Model $\rightarrow$ Output

Source: Isola, Torralba, Freeman
Learning from examples

(aka supervised learning)

Training data

\[
\begin{align*}
\{x_1, y_1\} \\
\{x_2, y_2\} \\
\{x_3, y_3\} \\
\ldots 
\end{align*}
\]

\[
\text{Learner} \quad \rightarrow \quad f : X \rightarrow Y
\]

Source: Isola, Torralba, Freeman
Learning from examples
(aka *supervised learning*)

Training data

\[
\begin{align*}
X & \quad Y \\
\{ & \\
& \text{“bowling alley”} \\
& \text{“desert”} \\
& \vdots
\end{align*}
\]

\[
\rightarrow
\]

\[
\text{Learner}
\]

\[
\rightarrow f : X \rightarrow Y
\]

Source: Isola, Torralba, Freeman
Learning from examples

(aka *supervised learning*)

Training data

\[
\begin{align*}
X &= \{ \text{images 1, 2, ...} \} \\
Y &= \{ \text{corresponding labels} \}
\end{align*}
\]

$\rightarrow$

Learner

$\rightarrow$

$f : X \rightarrow Y$

Source: Isola, Torralba, Freeman
Case study #1: Nearest neighbor
Nearest neighbor

Some feature ideas:
- tiny image
- normalized tiny image
- edge orientation histogram

Feature vector $x_q$

We’ll discuss better options later in course.
Nearest neighbor

Input image

Take label from:
\[ \text{argmin}_i \| x_q - x_i \| \]

\( x_q \)

\( X \)

\{ “bowling alley” \}

\{ “beach” \}

\{ “desert” \}

\( Y \)

K-nearest neighbor:
Take most common label in \( k \) closest examples
$k = 1$

$k = 15$

Source: [Hastie et al. “Elements of Statistical Learning”, 2001]
How do we choose $k$?
How do we choose k?

Choose hyperparameters like k, feature space, similarity function, etc.

Pool of nearest neighbors

Training Set

Validation Set

Test Set

% accuracy

choose best k = 3

Measures generalization. Ideally only test once at end!
Can work well when datasets are really big

Input image  Colorized output  Nearest Neighbors

[Torralba et al. “80 million tiny images”, 2008]
Nearest Neighbor

• Hard to define similarity metric

\[ || \quad - \quad || \quad = \quad ? \]

• Pays attention to all features equally
• Can be very slow: $O(nd)$
Case study #2: Linear least squares
Training data

$$\{x_i, y_i\}_{i=1}^N$$

Test query

Source: Isola, Torralba, Freeman
The relationship between $X$ and $Y$ is roughly linear: 

$$y \approx \theta_1 x + \theta_0$$

**Hypothesis space**

The relationship between $X$ and $Y$ is roughly linear: 

$$y \approx \theta_1 x + \theta_0$$

Source: Isola, Torralba, Freeman
Search for the parameters, \( \theta = \{\theta_0, \theta_1\} \), that best fit the data.

\[
f_\theta(x) = \theta_1 x + \theta_0
\]

Best fit in what sense?

Source: Isola, Torralba, Freeman
Search for the parameters, \( \theta = \{\theta_0, \theta_1\} \), that best fit the data.

\[
f_\theta(x) = \theta_1 x + \theta_0
\]

Best fit in what sense?

The least-squares objective (aka loss) says the best fit is the function that minimizes the squared error between predictions and target values:

\[
\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2 \quad \hat{y} \equiv f_\theta(x)
\]

Source: Isola, Torralba, Freeman
Search for the **parameters**, \( \theta = \{\theta_0, \theta_1\} \), that best fit the data.

\[
f_\theta(x) = \theta_1 x + \theta_0
\]

Best fit in what sense?

The least-squares **objective** (aka **loss**) says the best fit is the function that minimizes the squared error between predictions and target values:

\[
\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2 \quad \hat{y} \equiv f_\theta(x)
\]
Complete learning problem:

\[
\theta^* = \arg \min_{\theta} \sum_{i=1}^{N} (f_{\theta}(x_i) - y_i)^2
\]

\[
= \arg \min_{\theta} \sum_{i=1}^{N} (\theta_1 x_i + \theta_0 - y_i)^2
\]
Training data

\[ \{x_i, y_i\}_{i=1}^{N} \]

Test query

Source: Isola, Torralba, Freeman
Training data

$$\{x_i, y_i\}_{i=1}^{N}$$

Test query

$$f_\theta$$

$$x' \rightarrow f_\theta \rightarrow \hat{y}'$$

Source: Isola, Torralba, Freeman
How to minimize the objective w.r.t. $\theta$?

Recall:

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} (\theta_1 x_i + \theta_0 - y_i)^2$$

Learning problem

$$J(\theta) = \sum_{I=1}^{N} (\theta_1 x_i + \theta_0 - y_i)^2$$

$$= (y - X\theta)^T(y - X\theta)$$

$$\theta^* = \arg\min_{\theta} J(\theta)$$

$$\frac{\partial J(\theta)}{\partial \theta} = 0$$

$$\frac{\partial J(\theta)}{\partial \theta} = 2(X^TX\theta - X^Ty)$$

Solution

$$2(X^TX\theta^* - X^Ty) = 0$$

$$X^TX\theta^* = X^Ty$$

$$\theta^* = (X^TX)^{-1}X^Ty$$

Source: Isola, Torralba, Freeman
Empirical Risk Minimization
(formalization of supervised learning)

\[ \theta^* = \arg \min_{\theta} \sum_{i=1}^{N} \left( \theta_1 x_i + \theta_0 - y_i \right)^2 \]

Linear least squares learning problem

Source: Isola, Torralba, Freeman
Empirical Risk Minimization
(formalization of supervised learning)

Objective function
(loss)

\[ f^* = \arg \min_{f \in \mathcal{F}} \sum_{i=1}^{N} \mathcal{L}(f(x_i), y_i) \]

Hypothesis space
Training data

Source: Isola, Torralba, Freeman
The Problem of Generalization
Linear regression

\[ f_{\theta}(x) = \theta_0 + \theta_1 x \]

Source: Isola, Torralba, Freeman
Linear regression

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

Training data

Source: Isola, Torralba, Freeman
Polynomial regression

$f_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$

$f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k$

K-th degree polynomial regression

Source: Isola, Torralba, Freeman
Training data

\[ \{x_i, y_i\}_{i=1}^N \]
Training data

Let \( \{x_i, y_i\}_{i=1}^N \)

Test data

True data-generating process

\( \rho_{\text{data}} \)

Source: Isola, Torralba, Freeman
True data-generating process

\[ \{ x_{i}^{(\text{train})}, y_{i}^{(\text{train})} \}_{i=1}^{N} \sim P_{\text{data}} \]

\[ \{ x_{i}^{(\text{test})}, y_{i}^{(\text{test})} \}_{i=1}^{M} \sim P_{\text{data}} \]

Source: Isola, Torralba, Freeman
Training objective:

\[
\sum_{i=1}^{N} (f_{\theta}(x_{i}^{\text{train}}) - y_{i}^{\text{train}})^2
\]

Test time evaluation:

\[
\sum_{i=1}^{M} (f_{\theta}(x_{i}^{\text{test}}) - y_{i}^{\text{test}})^2
\]

Source: Isola, Torralba, Freeman
Training objective:

\[
\sum_{i=1}^{N} (f_{\theta}(x_i^{\text{train}}) - y_i^{\text{train}})^2
\]

True objective:

\[
\mathbb{E}_{\{x, y\} \sim p_{\text{data}}} [(f_{\theta}(x) - y)^2]
\]
What happens as we add more basis functions?

$$f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k$$

Source: Isola, Torralba, Freeman
What happens as we add more basis functions?

\[ f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k \]

Source: Isola, Torralba, Freeman
What happens as we add more basis functions?

\[ f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k \]

Source: Isola, Torralba, Freeman
What happens as we add more basis functions?

\[ f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k \]

Source: Isola, Torralba, Freeman
What happens as we add more basis functions?

\[
f_{\theta}(x) = \sum_{k=0}^{K} \theta_k x^k
\]

Source: Isola, Torralba, Freeman
What happens as we add more basis functions?

\[ f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k \]

Source: Isola, Torralba, Freeman
What happens as we add more basis functions?

\[ f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k \]

Source: Isola, Torralba, Freeman
What happens as we add more basis functions?

\[ f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k \]
What happens as we add more basis functions?

\[ f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k \]

Source: Isola, Torralba, Freeman
What happens as we add more basis functions?

\[ f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k \]

Source: Isola, Torralba, Freeman
What happens as we add more basis functions?

$$f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k$$

This phenomenon is called **overfitting**.

It occurs when we have too high capacity a model, e.g., too many free parameters, too few data points to pin these parameters down.

Source: Isola, Torralba, Freeman
When the model does not have the capacity to capture the true function, we call this \textbf{underfitting}.

An underfit model will have high \textbf{error} on the training points. This error is known as \textbf{approximation error}.

Source: Isola, Torralba, Freeman
The true function is a quadratic, so a quadratic model ($K=2$) fits well.
Now we have zero approximation error — the curve passes exactly through each training point.

But we have high **generalization error**, reflected in the gap between the true function and the fit line. We want to do well on *novel* queries, which will be sampled from the green curve (plus noise).

Source: Isola, Torralba, Freeman
Underfitting

\[ K = 1 \]

- High error on train set
- High error on test set

Appropriate model

\[ K = 2 \]

- Low error on train set
- Low error on test set

Overfitting

\[ K = 10 \]

- Lowest error on train set
- High error on test set

Source: Isola, Torralba, Freeman
We need to control the **capacity** of the model (e.g., use the appropriate number of free parameters).

The capacity may be defined as the number of hypotheses under consideration in the hypothesis space.

Complex models with many free parameters have high capacity.

Simple models have low capacity.

Source: Isola, Torralba, Freeman
Training error versus generalization error

[“Deep Learning”, Goodfellow et al.]

Source: Isola, Torralba, Freeman
How do we know if we are underfitting or overfitting?

**Cross validation**: measure prediction error on validation set

Validation data \( \{ x^{(\text{val})}_i, y^{(\text{val})}_i \} \)
Fitting a model

Underfitting?
  1. add more parameters (more features, more layers, etc.)

Overfitting?
  1. remove parameters
  2. add **regularizers**
Regularization

Empirical risk minimization:

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} \mathcal{L}(f_{\theta}(x_i), y_i) + R(\theta)$$
Regularized least squares

\[ f_\theta(x) = \sum_{k=0}^{K} \theta_k x^k \]

\[ R(\theta) = \lambda \| \theta \|^2 \]

Only use polynomial terms if you really need them! Most terms should be zero.

*ridge regression*, a.k.a., *Tikhonov regularization*
\[
\theta^* = \arg \min_{\theta} \sum_{i=1}^{N} \mathcal{L}(f_{\theta}(x_i), y_i) + \lambda \|\theta\|_2^2
\]

Source: Isola, Torralba, Freeman
\[ \theta^* = \arg \min_{\theta} \sum_{i=1}^{N} \mathcal{L}(f_\theta(x_i), y_i) + \lambda \|\theta\|^2_2 \]

Source: Isola, Torralba, Freeman
Underfitting

\[ K = 1 \]

Simple model
Doesn’t fit the training data

Appropriate model

\[ K = 2 \]

Simple model
Fits the training data

Overfitting

\[ K = 10 \]

Complex model
Fits the training data

Source: Isola, Torralba, Freeman
Image classification

image $x$ \rightarrow \text{Classifier} \rightarrow \text{“Fish”}

label $y$

Source: Isola, Torralba, Freeman
Image classification

Source: Isola, Torralba, Freeman
Image classification

image $x$ \rightarrow \text{Classifier} \rightarrow \text{“Fish”}

Source: Isola, Torralba, Freeman
Image classification

: image $x$ → Classifier → “Duck”

label $y$

Source: Isola, Torralba, Freeman
Training data

$x_i$ \hspace{1cm} $y_i$

\{ 
\begin{align*}
& \text{"Fish"}, \\
& \text{"Grizzly"}, \\
& \text{"Chameleon"}, \\
& \vdots
\end{align*}
\}

$\text{arg min}_{f \in \mathcal{F}} \sum_{i=1}^{N} \mathcal{L}(f(x_i), y_i)$

$X$

$y$

"Fish"

Source: Isola, Torralba, Freeman
How to represent class labels?

Training data

\[ \begin{align*}
\{ & \text{"Fish"}, \{ & \text{"Grizzly"}, \{ & \text{"Chameleon"}, \ldots
\end{align*} \]

One-hot vector

\[ \begin{align*}
\{ & [0,0,1], \{ & [0,1,0], \{ & [1,0,0], \ldots
\end{align*} \]

Source: Isola, Torralba, Freeman
What should the loss be?

0-1 loss (number of misclassifications)

\[ \mathcal{L}(\hat{y}, y) = \mathbb{1}(\hat{y} = y) \]  

← discrete; hard to optimize!

Least squares approximation (predict 1 for true class, 0 for others)

\[ \mathcal{L}(\hat{y}, y) = \sum_{k=1}^{K} (y_k - \hat{y}_k)^2 \]  

← easy to optimize, but crude approximation

Cross entropy

\[ \mathcal{L}(\hat{y}, y) = H(y, \hat{y}) = -\sum_{k=1}^{K} y_k \log \hat{y}_k \]  

← easy to optimize, good approximation

Adapted from: Isola, Torralba, Frey
Ground truth label $y$

$x$

$[0,0,0,0,0,1,0,0,...]$
Source: Isola, Torralba, Freeman
Prediction $\hat{y}$

$f_\theta : X \to \mathbb{R}^K$

dolphins

Ground truth label $y$

dolphins
cats
grizzly bears
angel fish
chameleons
clown fish
iguana
elephant

Source: Isola, Torralba, Freeman
$\mathbf{x} \xrightarrow{f} \mathbf{y} \xrightarrow{\hat{y}} \mathbf{y}$

$\mathbf{f}_\theta : \mathcal{X} \rightarrow \mathbb{R}^K$

Ground truth label $y$

Prediction $\hat{y}$

Loss

$H(y, \hat{y}) = -\sum_{k=1}^{K} y_k \log \hat{y}_k$

Source: Isola, Torralba, Freeman
\[ f_\theta : X \rightarrow \mathbb{R}^K \]

**Prediction** \( \hat{y} \)

**Ground truth label** \( y \)

**Loss**

\[ H(y, \hat{y}) = -\sum_{k=1}^{K} y_k \log \hat{y}_k \]

Source: Isola, Torralba, Freeman
**Softmax regression** (a.k.a. multinomial logistic regression)

\[ f_\theta : X \to \mathbb{R}^K \]

\[ z = f_\theta(x) \quad \text{← logits: vector of K scores, one for each class} \]

\[ \hat{y} = \text{softmax}(z) \quad \text{← squash into a non-negative vector that sums to 1} \]

\[ \hat{y}_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \]

Source: Isola, Torralba, Freeman
Next lecture: more linear models