Mathematical techniques in data science

Lecture 8: Hypothesis spaces and loss functions
Cross-validation

Progress of projects

- First meeting (03/20)
  Project description (04/03)
- Second meeting (Week of 04/18)
- Third meeting (Week of 05/03)
- Final presentation (05/14-16)
- Final report due (05/21)

Where are we?

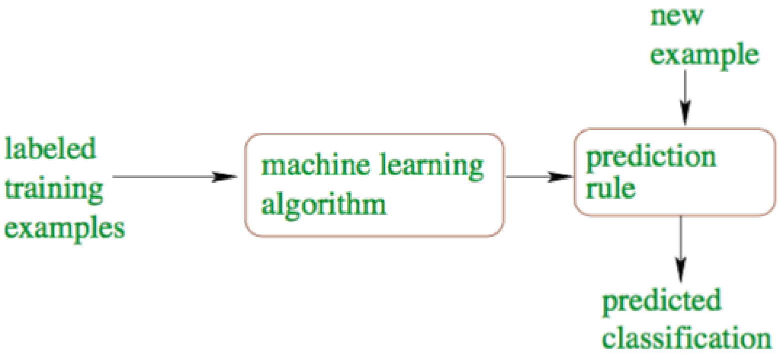
- Algorithms
  - Intros to classification
  - Overfitting and underfitting
  - Nearest neighbors
  - Logistic regression
  - Feed-forward neural networks
  - Convolutional neural networks
- Codings
  - Numpy, matplotlib, sklearn
  - Reading sklearn documentations
  - Pre-process inputs (i.e., numpy.shape())
  - Data simulations (by hand or using built-in functions in sklearn)
  - Data splitting
  - Train models; making prediction; evaluate models

What's next?

- Mathematical techniques in data sciences
  - A short introduction to statistical learning theory
  - Random forests — boosting and bootstrapping
  - SVM – the kernel trick
  - Linear regression – regularization and feature selection
- Algorithms and learning contexts
  - PCA and Manifold learning
  - Clustering
  - Selected topics

A short introduction to statistical learning

Diagram of a typical supervised learning problem



Supervised learning: learning a function that maps an input to an output based on example input-output pairs

Supervised learning: standard setting

- Given: a sequence of label data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ sampled (independently and identically) from an unknown distribution $P_{X,Y}$
- Goal: predict the label of new samples (as accurately as possible)

Example

- MNIST dataset

- Each image as a vector in $x \in \mathbb{R}^{784}$ and the label as a scalar $y \in \{0, 1, \ldots, 9\}$
- Goal: learn to identify/predict digits (as accurately as possible)

Supervised learning: standard setting

- Given: a sequence of label data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ sampled (independently and identically) from an unknown distribution $P_{X,Y}$
- Goal: predict the label of new samples (as accurately as possible)
- Question:
  - How to make predictions?
  - What do you mean by "as accurately as possible?"

Hypothesis space

- Given: a sequence of label data $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ sampled (independently and identically) from an unknown distribution $P_{X,Y}$
- Goal: a learning algorithm seeks a function $h : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ is the input space and $\mathcal{Y}$ is the output space
- The function h is an element of some space of possible functions $\mathcal{H}$, usually called the *hypothesis space*
- Usually, this hypothesis space can be indexed by some parameters (often specified by a model or a learning algorithm)

Hypothesis space: logistic regression

- Two classes: 0 and 1
- $x \in \mathbb{R}^d$
- Probability model

$$p_{w,b}(x) = \frac{1}{1 + e^{-w^T x - b}}$$

- Prediction rule $h_{w,b}(x)$
  - If $p_{w,b}(x) > 0.5$, predict $h_{w,b}(x) = 1$
  - If $p_{w,b}(x) \leq 0.5$, predict $h_{w,b}(x) = 0$
- Hypothesis space

$$\mathcal{H} = \{h_{w,b} : w \in \mathbb{R}^d, b \in \mathbb{R}\}$$

Loss function

- The function h is an element of some space of possible functions $\mathcal{H}$, usually called the *hypothesis space*
- In order to measure how well a function fits the data, a *loss function*

$$L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^{\geq 0}$$

  is defined

Loss function: examples

- In order to measure how well a function fits the data, a *loss function*

$$L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^{\geq 0}$$

  is defined
- For regression:

$$L(h(x), y) = [h(x) - y]^2$$

- For classification: the 0-1 loss and the binary-cross-entropy loss

$$L(h(x), y) = \begin{cases} 0, & \text{if} \quad h(x) = y \\ 1 & \text{otherwise} \end{cases}$$

$$L(p(x), y) = -y \ \log(p(x)) - (1 - y) \ \log(1 - p(x))$$

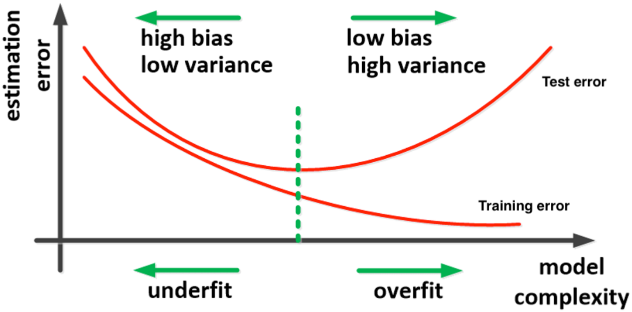
Loss function

- The function h is an element of some space of possible functions $\mathcal{H}$, usually called the *hypothesis space*
- In order to measure how well a function fits the data, a *loss function*

  $$L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^{\geq 0}$$

  is defined
- It is straightforward that we want to have a hypothesis with minimal loss
- Question: minimal loss on which dataset?

Underfiting/Overfitting



Training error vs. test error

+ Code + Text

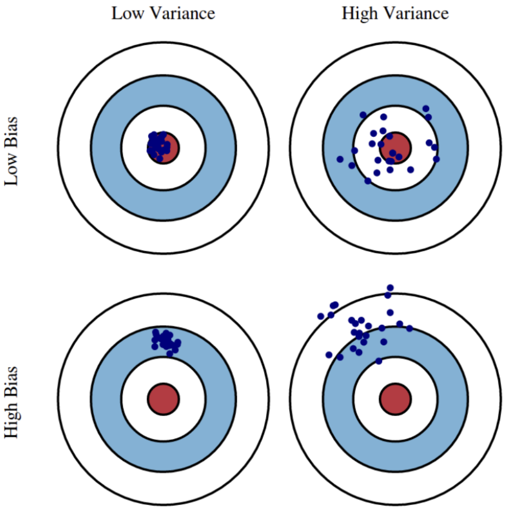
```
1563/1563 [==============================] - 7s 5ms/step - loss: 0.0821 - accuracy: 0.9754
Epoch 25/30
1563/1563 [==============================] - 6s 4ms/step - loss: 0.0741 - accuracy: 0.9774
Epoch 26/30
1563/1563 [==============================] - 7s 4ms/step - loss: 0.0830 - accuracy: 0.9759
Epoch 27/30
1563/1563 [==============================] - 6s 4ms/step - loss: 0.0778 - accuracy: 0.9768
Epoch 28/30
1563/1563 [==============================] - 7s 5ms/step - loss: 0.0740 - accuracy: 0.9783
Epoch 29/30
1563/1563 [==============================] - 7s 4ms/step - loss: 0.0741 - accuracy: 0.9787
Epoch 30/30
1563/1563 [==============================] - 7s 4ms/step - loss: 0.0679 - accuracy: 0.9796
{'loss': [1.3642792701721191, 0.9498528242111206, 0.7482982873916626, 0.5679550766944885, 0.4281372
```

7. Evaluate the trained model on test set

```
score = model.evaluate(X_test, Y_test, verbose=0)
print("Test loss:", score[0])
print("Test accuracy:", score[1])
```

```
Test loss: 4.181140899658203
Test accuracy: 0.6317999958892004
```

Variance-bias trade-off



Risk function

- Assumption: The future samples will be obtained from the same distribution $P_{X,Y}$ of the training data
- With a pre-defined loss function, the risk function is defined as

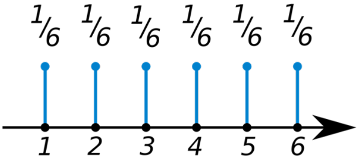
$$R(h) = E_{(X,Y)\sim P}[L(h(X), Y)]$$

- The "optimal hypothesis", denoted by h^* in this lecture, is the minimizer over $\mathcal{H}$ of the risk function

$$h^* = \arg\min_{h\in\mathcal{H}} R(h)$$

Review: Probability

Discrete random variable



- Probability of an event A:

$$P(A) = \sum_{x \in A} P(x)$$

Example: $P(\{\text{X is even}\}) = P(2) + P(4) + P(6) = 1/2$

- Sometimes we write $P(X = x)$ for $P(x)$, for example, $P(X = 2) = P(2)$.

Continuous random variable

- Sample space is continuous (real values)
- Characterized by a density function P:
  - $P(x) \geq 0$ for all $x \in \mathbb{R}$
  - $\int_{-\infty}^{\infty} P(x)\ dx = 1$
  - For any fixed constant a, b,

$$P(a \leq X \leq b) = \int_a^b P(x)\ dx$$

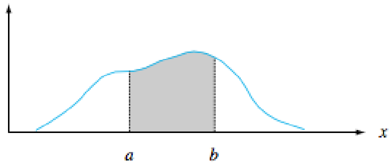


Figure 4.2 $P(a \leq X \leq b)$ = the area under the density curve between a and b

Expectation of random variables

- Expectation (expected value or mean) of a discrete random variable X:

$$E[X] = \sum_x xP(x) = \sum_{i=1}^{n} x_i P(x_i)$$

- For continuous variables:

$$E[X] = \int_x xP(x)dx$$

- Can be used for functions:

$$E[g(X)] = \sum_x g(x)P(x)$$

or

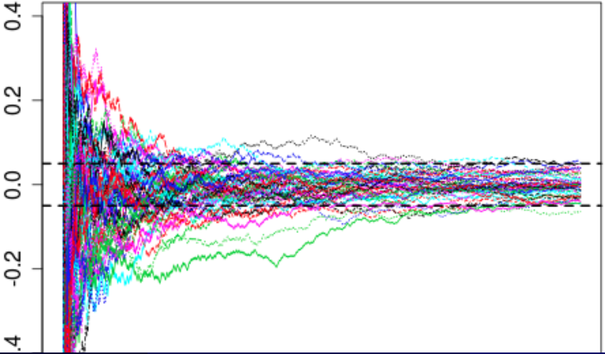
$$E[g(X)] = \int_x g(x)P(x)dx$$

Law of large numbers

THEOREM If $X_1, X_2, \ldots, X_n$ is a random sample from a distribution with mean μ and v
ance σ^2, then $\overline{X}$ converges to μ

a. In mean square $E[(\overline{X} - \mu)^2] \rightarrow 0$ as $n \rightarrow \infty$

b. In probability $P(|\overline{X} - \mu| \geq \varepsilon) \rightarrow 0$ as $n \rightarrow \infty$



Empirical risk

Empirical risk

- Since P is unknown, the simplest approach is to approximate the risk function by the empirical risk

$$R_n(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(x_i), y_i)$$

- Rationale: The law of large number – If the random variables $Z_1, Z_2, \ldots, Z_n$ are drawn independently from the same distribution P_Z, then

$$\frac{Z_1 + Z_2 + \ldots Z_n}{n} \approx E[Z]$$

ERM

- Empirical risk minimizer (ERM): minimizer of the empirical risk function

$$R_n(h) = \frac{1}{n}\sum_{i=1}^{n} L(h(x_i), y_i)$$

- The risk function is defined as

$$R(h) = E_{(X,Y)\sim P}[L(h(X), Y)]$$

- Rationale: $R_n(h) \approx R(h)$
- In this lecture, we use the notation $\hat{h}_n$ to denote the ERM
- We hope that

$$R(\hat{h}_n) \approx R(h^*)$$

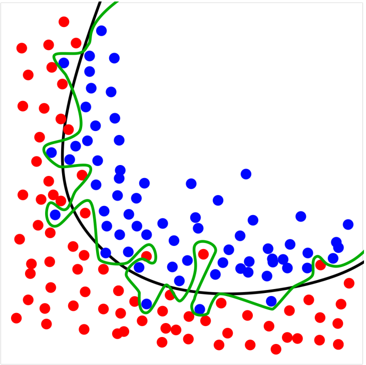
- Note: $\hat{h}_n$ is random, while h^* is a fixed hypothesis

Failure of ERM

We hope that

$$R(\hat{h}_n) \approx R(h^*),$$

but in general, this might not be true if the hypothesis space $\mathcal{H}$ is too large



Failure of ERM

- We hope that
$$R(\hat{h}_n) \approx R(h^*),$$
but in general, this might not be true if the hypothesis space $\mathcal{H}$ is too large
- Question: What does "too large" mean?
- We need to be able to quantify/control the difference between $R(\hat{h}_n)$ and $R(h^*)$

K-fold cross-validation

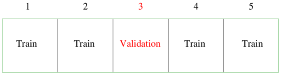
K-fold cross-validation:

Split data into K equal (or almost equal) parts/folds at random.
for each parameter λ_i **do**
  for $j = 1, \ldots, K$ **do**
    Fit model on data with fold j removed.
    Test model on remaining fold $\to$ j-th test error.
  end for
  Compute average test errors for parameter λ_i.
end for
Pick parameter with smallest average error.

K-fold cross-validation

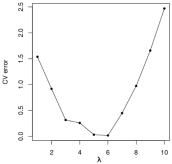
More precisely,

- Split data into K folds $F_1, \dots, F_K$.



- Let $L(y, \hat{y})$ be a *loss function*. For example, $L(y, \hat{y}) = \|y - \hat{y}\|_2^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2$.
- Let $f_\lambda^{-k}(\mathbf{x})$ be the model fitted on all, but the k-th fold.
- Let

$$CV(\lambda) := \frac{1}{n} \sum_{k=1}^{n} \sum_{i \in F_k} L(y_i, f_\lambda^{-i}(\mathbf{x}_i))$$



- Pick λ among a *relevant* set of parameters

$$\hat{\lambda} = \underset{\lambda \in \{\lambda_1, \dots, \lambda_m\}}{\operatorname{argmin}} CV(\lambda)$$

K-fold cross-validation