Philosophical Fundamentals of Machine Learning

Bruce Rushing

Purdue University

Week 2

Table of Contents

Table of Contents

Supervised Learning

The goal of supervised learning is to predict some **target** from some features. What distinguishes supervised learning from unsupervised learning is that we have 1) **labeled data**, 2) **every example has a target value**, and 3) we **reward prediction close to target**.

Supervised Learning

The goal of supervised learning is to predict some **target** from some features. What distinguishes supervised learning from unsupervised learning is that we have 1) **labeled data**, 2) **every example has a target value**, and 3) we **reward prediction close to target**.

Question: How does Machine Learning (ML) work in supervised learning?

Supervised Learning

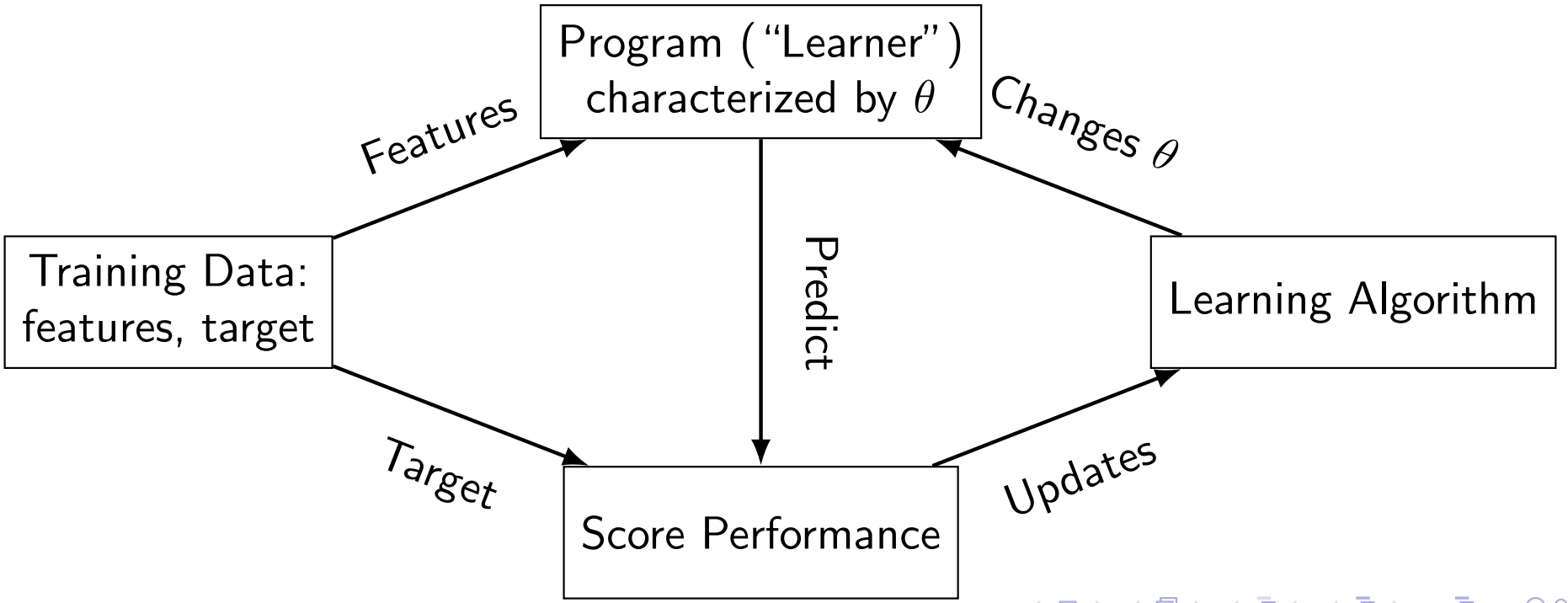
The goal of supervised learning is to predict some **target** from some features. What distinguishes supervised learning from unsupervised learning is that we have 1) **labeled data**, 2) **every example has a target value**, and 3) we **reward prediction close to target**.

Question: How does Machine Learning (ML) work in supervised learning?
Answer: ML is a type of meta-programming.

Supervised Learning

The goal of supervised learning is to predict some **target** from some features. What distinguishes supervised learning from unsupervised learning is that we have 1) **labeled data**, 2) **every example has a target value**, and 3) we **reward prediction close to target**.

Question: How does Machine Learning (ML) work in supervised learning?
Answer: ML is a type of meta-programming.



The Supervised Learning Pipeline

When building supervised learning models, the algorithm we follow is:

The Supervised Learning Pipeline

When building supervised learning models, the algorithm we follow is:

1. Evaluate data.

The Supervised Learning Pipeline

When building supervised learning models, the algorithm we follow is:

1. Evaluate data.
2. Select ML model:

The Supervised Learning Pipeline

When building supervised learning models, the algorithm we follow is:

1. Evaluate data.
2. Select ML model:
    1. Build data pipline.

The Supervised Learning Pipeline

When building supervised learning models, the algorithm we follow is:

1. Evaluate data.
2. Select ML model:
    1. Build data pipline.
    2. Build ML model.

The Supervised Learning Pipeline

When building supervised learning models, the algorithm we follow is:

1. Evaluate data.
2. Select ML model:
    1. Build data pipline.
    2. Build ML model.
    3. Train ML model.

The Supervised Learning Pipeline

When building supervised learning models, the algorithm we follow is:

1. Evaluate data.
2. Select ML model:
    1. Build data pipline.
    2. Build ML model.
    3. Train ML model.
    4. Evaluate ML model.

The Supervised Learning Pipeline

When building supervised learning models, the algorithm we follow is:

1. Evaluate data.
2. Select ML model:
    1. Build data pipline.
    2. Build ML model.
    3. Train ML model.
    4. Evaluate ML model.
3. Test ML model.

The Supervised Learning Pipeline

When building supervised learning models, the algorithm we follow is:

1. Evaluate data.
2. Select ML model:
    1. Build data pipline.
    2. Build ML model.
    3. Train ML model.
    4. Evaluate ML model.
3. Test ML model.
4. Deploy and apply ML model.

The Supervised Learning Pipeline

So how do we evaluate ML models?

Prediction as a Decision Problem

We can think of prediction as a decision problem: our actions are deciding what targets to predict given the features we have observed. So we want *criteria* for evaluating decision procedures.

Prediction as a Decision Problem

We can think of prediction as a decision problem: our actions are deciding what targets to predict given the features we have observed. So we want *criteria* for evaluating decision procedures.

There are two philosophically different approaches for evaluating decision procedures:

Prediction as a Decision Problem

We can think of prediction as a decision problem: our actions are deciding what targets to predict given the features we have observed. So we want *criteria* for evaluating decision procedures.

There are two philosophically different approaches for evaluating decision procedures:

1. **Internal**: do we do as well as we can do by our own beliefs?

Prediction as a Decision Problem

We can think of prediction as a decision problem: our actions are deciding what targets to predict given the features we have observed. So we want *criteria* for evaluating decision procedures.

There are two philosophically different approaches for evaluating decision procedures:

1. **Internal**: do we do as well as we can do by our own beliefs?
2. **External**: do we do as well as we can do reliably?

Prediction as a Decision Problem

We can think of prediction as a decision problem: our actions are deciding what targets to predict given the features we have observed. So we want *criteria* for evaluating decision procedures.

There are two philosophically different approaches for evaluating decision procedures:

1. **Internal**: do we do as well as we can do by our own beliefs?
2. **External**: do we do as well as we can do reliably?

The two methods we consider in this course are **Bayesian Statistical Decision Theory** and **Empirical Risk Minimization**.

Table of Contents

Expected Utility Theory

Example

Suppose we want to predict from some features whether a patient has cancer or not. We know that our ML model's prediction will be used to administer a drug, and we know that the drug has adverse side-effects, which can be measured by quality-adjusted life years (QALYs):

Expected Utility Theory

Example

Suppose we want to predict from some features whether a patient has cancer or not. We know that our ML model's prediction will be used to administer a drug, and we know that the drug has adverse side-effects, which can be measured by quality-adjusted life years (QALYs):

Cancer	Drugs	QALY	Relative Cost
1	1	5	5
1	0	0	10
0	1	8	2
0	0	10	0

Expected Utility Theory

Example

Suppose we want to predict from some features whether a patient has cancer or not. We know that our ML model's prediction will be used to administer a drug, and we know that the drug has adverse side-effects, which can be measured by quality-adjusted life years (QALYs):

Cancer	Drugs	QALY	Relative Cost
1	1	5	5
1	0	0	10
0	1	8	2
0	0	10	0

We can think of the QALYs as measuring the relative *utility* of our predictions. If we take our goal to minimize the relative lost utility compared to how well we could do, then we have what is called a *loss* or *cost* function. **Our goal is to minimize this loss**.

Expected Utility Theory

Recall

The expected value of a function $g(x)$ with respect to a discrete random variable $X \sim p(x)$ is given by:

Expected Utility Theory

Recall

The expected value of a function $g(x)$ with respect to a discrete random variable $X \sim p(x)$ is given by:

$$\mathbb{E}_{X \sim p(x)}[g(x)] = \sum_{x \in \mathcal{T}} g(x)p(x)$$

Expected Utility Theory

Recall

The expected value of a function $g(x)$ with respect to a discrete random variable $X \sim p(x)$ is given by:

$$\mathbb{E}_{X \sim p(x)}[g(x)] = \sum_{x \in \mathcal{T}} g(x)p(x)$$

Definition (Expected Loss)

If we let our disutility be characterized by the loss function $l : Y \to \mathbb{R}$, then we say that *quality of our prediction* is given by the expected loss with respect to the target random variable Y from $Y \sim p(y)$, where the predictions of our model characterized by $\hat{y}$:

Expected Utility Theory

Recall

The expected value of a function $g(x)$ with respect to a discrete random variable $X \sim p(x)$ is given by:

$$\mathbb{E}_{X \sim p(x)}[g(x)] = \sum_{x \in \mathcal{T}} g(x)p(x)$$

Definition (Expected Loss)

If we let our disutility be characterized by the loss function $l : Y \to \mathbb{R}$, then we say that *quality of our prediction* is given by the expected loss with respect to the target random variable Y from $Y \sim p(y)$, where the predictions of our model characterized by $\hat{y}$:

$$\mathbb{E}_{Y \sim p(y)}[l(y, \hat{y})] = \sum_{y \in Y} l(y, \hat{y})p(y)$$

Bayesian Decision Theory

Remark

Assuming we see features $\mathbf{x}$, our goal is to predict the targets, which is the posterior probability of the targets given the data $p(y|\mathbf{x})$.

Bayesian Decision Theory

Remark

Assuming we see features $\mathbf{x}$, our goal is to predict the targets, which is the posterior probability of the targets given the data $p(y|\mathbf{x})$. Our model produces predictions $\hat{y} = f(\mathbf{x})$ through some function f on the features $\mathbf{x}$. This changes our expected loss to reflect that posterior probabillity, which gives us our **posterior risk**:

Bayesian Decision Theory

Remark

Assuming we see features $\mathbf{x}$, our goal is to predict the targets, which is the posterior probability of the targets given the data $p(y|\mathbf{x})$. Our model produces predictions $\hat{y} = f(\mathbf{x})$ through some function f on the features $\mathbf{x}$. This changes our expected loss to reflect that posterior probabillity, which gives us our **posterior risk**:

$$R(\hat{y}|\mathbf{x}) := \mathbb{E}_{Y \sim p(y|\mathbf{x})}[l(y, \hat{y})] = \sum_{y \in Y} l(y, \hat{y}) p(y|\mathbf{x})$$

Bayesian Decision Theory

Remark

Assuming we see features $\mathbf{x}$, our goal is to predict the targets, which is the posterior probability of the targets given the data $p(y|\mathbf{x})$. Our model produces predictions $\hat{y} = f(\mathbf{x})$ through some function f on the features $\mathbf{x}$. This changes our expected loss to reflect that posterior probabillity, which gives us our **posterior risk**:

$$R(\hat{y}|\mathbf{x}) := \mathbb{E}_{Y \sim p(y|\mathbf{x})}[l(y, \hat{y})] = \sum_{y \in Y} l(y, \hat{y}) p(y|\mathbf{x})$$

The goal then is to find the **optimal predictor** that minimizes the expected risk:

$$\pi^*(\mathbf{x}) = \arg\min_{\hat{y} \in \hat{Y}} R(\hat{y}|\mathbf{x})$$

Example

Returning to our drug example, if we have a posterior distribution given by:

Cancer	Drugs	$l(cancer, prediction)$	$p(\cdot\|\mathbf{x})$
1	1	5	0.2
1	0	10	0.1
0	1	2	0.4
0	0	0	0.3

Example

Returning to our drug example, if we have a posterior distribution given by:

Cancer	Drugs	$l(cancer, prediction)$	$p(\cdot\|\mathbf{x})$
1	1	5	0.2
1	0	10	0.1
0	1	2	0.4
0	0	0	0.3

This leads to a posterior probability for the target *cancer* of $p(cancer|\mathbf{x}) = 0.3$ and $p(not\ cancer|\mathbf{x}) = 0.7$.

Example

Returning to our drug example, if we have a posterior distribution given by:

Cancer	Drugs	$l(cancer, prediction)$	$p(\cdot\|\mathbf{x})$
1	1	5	0.2
1	0	10	0.1
0	1	2	0.4
0	0	0	0.3

This leads to a posterior probability for the target *cancer* of $p(cancer|\mathbf{x}) = 0.3$ and $p(not\ cancer|\mathbf{x}) = 0.7$. Suppose one model predicts cancer, and we administer the drug on these features. Then our risk:

Example

Returning to our drug example, if we have a posterior distribution given by:

Cancer	Drugs	$l(cancer, prediction)$	$p(\cdot\|\mathbf{x})$
1	1	5	0.2
1	0	10	0.1
0	1	2	0.4
0	0	0	0.3

This leads to a posterior probability for the target *cancer* of $p(cancer|\mathbf{x}) = 0.3$ and $p(not\ cancer|\mathbf{x}) = 0.7$. Suppose one model predicts cancer, and we administer the drug on these features. Then our risk:

$$\begin{aligned} R(c, d) &= p(c|\mathbf{x})l(c, d) + p(not\ c|\mathbf{x})l(not\ c, d) \\ &= (0.3)(5) + (0.7)(2) = 2.9 \end{aligned}$$

Bayesian Decision Theory

Example

Cancer	Drugs	$l(cancer, prediction)$	$p(\cdot\|\mathbf{x})$
1	1	5	0.2
1	0	10	0.1
0	1	2	0.4
0	0	0	0.3

Bayesian Decision Theory

Example

Cancer	Drugs	$l(cancer, prediction)$	$p(\cdot\|\mathbf{x})$
1	1	5	0.2
1	0	10	0.1
0	1	2	0.4
0	0	0	0.3

Suppose another model doesn't predict cancer, and we don't administer the drug on these features.

Bayesian Decision Theory

Example

Cancer	Drugs	$l(cancer, prediction)$	$p(\cdot\|\mathbf{x})$
1	1	5	0.2
1	0	10	0.1
0	1	2	0.4
0	0	0	0.3

Suppose another model doesn't predict cancer, and we don't administer the drug on these features. With the same posterior probabilities *cancer* of $p(cancer|\mathbf{x}) = 0.3$ and $p(not\ cancer|\mathbf{x}) = 0.7$, our risk is:

Bayesian Decision Theory

Example

Cancer	Drugs	$l(cancer, prediction)$	$p(\cdot\|\mathbf{x})$
1	1	5	0.2
1	0	10	0.1
0	1	2	0.4
0	0	0	0.3

Suppose another model doesn't predict cancer, and we don't administer the drug on these features. With the same posterior probabilities *cancer* of $p(cancer|\mathbf{x}) = 0.3$ and $p(not\ cancer|\mathbf{x}) = 0.7$, our risk is:

$$\begin{aligned} R(c, no\ d) &= p(c|\mathbf{x})l(c, no\ d) + p(not\ c|\mathbf{x})l(not\ c, no\ d) \\ &= (0.3)(10) + (0.7)(0) = 3 \end{aligned}$$

Example

Cancer	Drugs	$l(cancer, prediction)$	$p(\cdot|\mathbf{x})$
1	1	5	0.2
1	0	10	0.1
0	1	2	0.4
0	0	0	0.3

Suppose another model doesn't predict cancer, and we don't administer the drug on these features. With the same posterior probabilities *cancer* of $p(cancer|\mathbf{x}) = 0.3$ and $p(not\ cancer|\mathbf{x}) = 0.7$, our risk is:

$$\begin{aligned} R(c, no\ d) &= p(c|\mathbf{x})l(c, no\ d) + p(not\ c|\mathbf{x})l(not\ c, no\ d) \\ &= (0.3)(10) + (0.7)(0) = 3 \end{aligned}$$

The better model is then the first model.

Bayesian Decision Theory

Remark

The expected risk is **always relative to a posterior probability**.

Bayesian Decision Theory

Remark

The expected risk is **always relative to a posterior probability**. Recall the posterior probability is a function both of the likelihoods and the prior probability:

Bayesian Decision Theory

Remark

The expected risk is **always relative to a posterior probability**. Recall the posterior probability is a function both of the likelihoods and the prior probability:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

Bayesian Decision Theory

Remark

The expected risk is **always relative to a posterior probability**. Recall the posterior probability is a function both of the likelihoods and the prior probability:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

This means that we, as the modeler, are assessing our relative performance based on our best guess about the proportion of target features given in our prior, $p(y)$. This can affect our recommendation.

Bayesian Decision Theory

Remark

The expected risk is **always relative to a posterior probability**. Recall the posterior probability is a function both of the likelihoods and the prior probability:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

This means that we, as the modeler, are assessing our relative performance based on our best guess about the proportion of target features given in our prior, $p(y)$. This can affect our recommendation.

Example

In the previous example, our assessment of the better model would change based on how prevalent we thought cancer was. For example, if we thought $p(cancer|\mathbf{x}) = 0.1$ while $p(not\ cancer|\mathbf{x}) = 0.9$, then we would recommend the second ML model

Table of Contents

Bayesian Statistical Decision Theory

In the binary classification case, the trade-off we deal with is between incorrectly predicting the target when it is absent (**false positive** or **type-1 error**) and incorrectly predicting no target when it is present (**false negative** or **type-2 error**).

Bayesian Statistical Decision Theory

In the binary classification case, the trade-off we deal with is between incorrectly predicting the target when it is absent (**false positive** or **type-1 error**) and incorrectly predicting no target when it is present (**false negative** or **type-2 error**). In contrast, we can correctly predict the target when it is present (**true positive**) and correctly predict no target when it is absent (**true negative**).

Bayesian Statistical Decision Theory

In the binary classification case, the trade-off we deal with is between incorrectly predicting the target when it is absent (**false positive** or **type-1 error**) and incorrectly predicting no target when it is present (**false negative** or **type-2 error**). In contrast, we can correctly predict the target when it is present (**true positive**) and correctly predict no target when it is absent (**true negative**). The relative rates we assing to those by our prior is given below:

probability	type
$p(\hat{y} = 1 \mid y = 0)$	false positive rate (FPR)
$p(\hat{y} = 1 \mid y = 1)$	true positive rate (TPR)
$p(\hat{y} = 0 \mid y = 1)$	false negative rate (FNR)
$p(\hat{y} = 0 \mid y = 0)$	true negative rate (TNR)

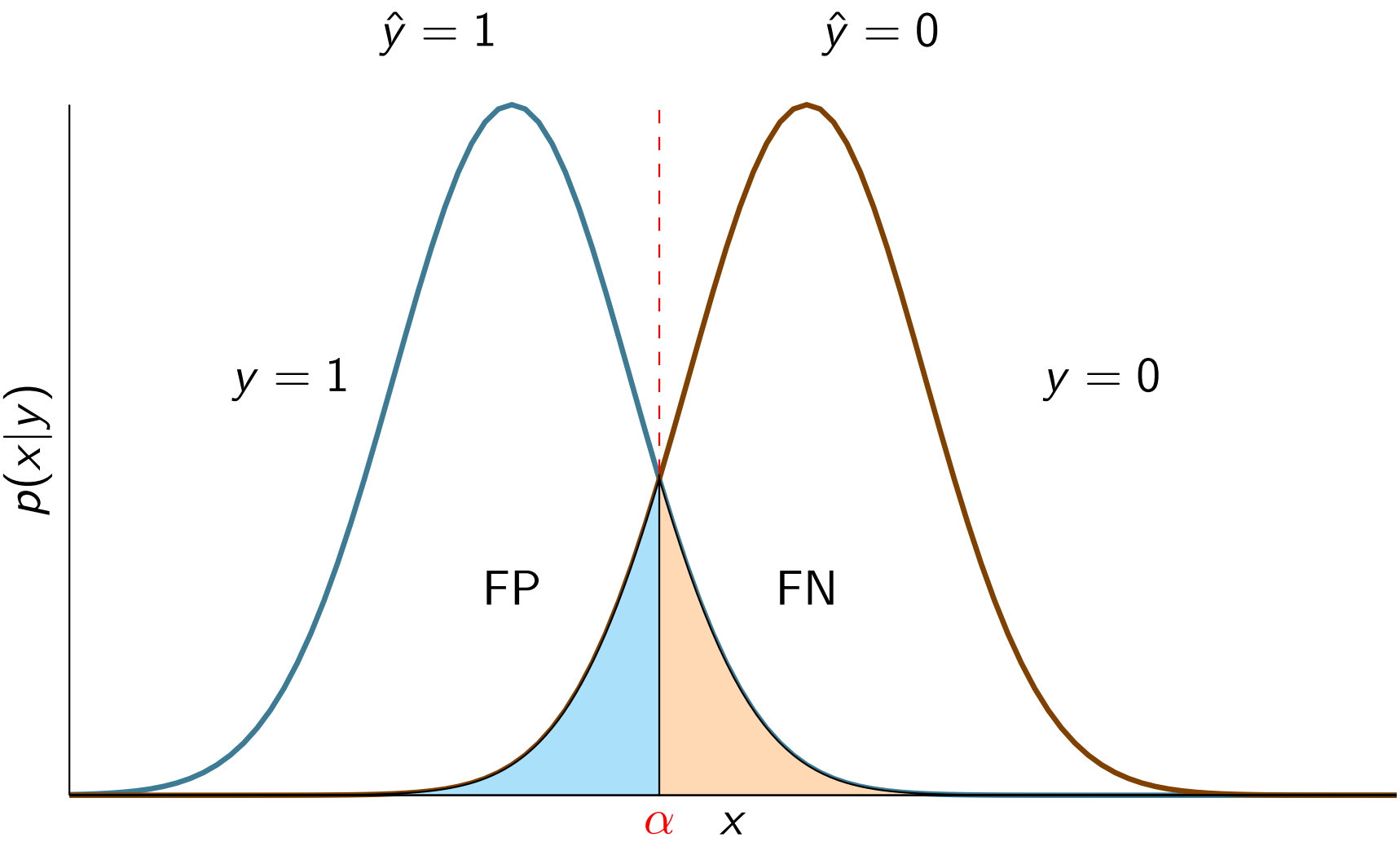
Bayesian Statistical Decision Theory

In the binary classification case, the trade-off we deal with is between incorrectly predicting the target when it is absent (**false positive** or **type-1 error**) and incorrectly predicting no target when it is present (**false negative** or **type-2 error**). In contrast, we can correctly predict the target when it is present (**true positive**) and correctly predict no target when it is absent (**true negative**). The relative rates we assing to those by our prior is given below:

probability	type
$p(\hat{y} = 1 \mid y = 0)$	false positive rate (FPR)
$p(\hat{y} = 1 \mid y = 1)$	true positive rate (TPR)
$p(\hat{y} = 0 \mid y = 1)$	false negative rate (FNR)
$p(\hat{y} = 0 \mid y = 0)$	true negative rate (TNR)

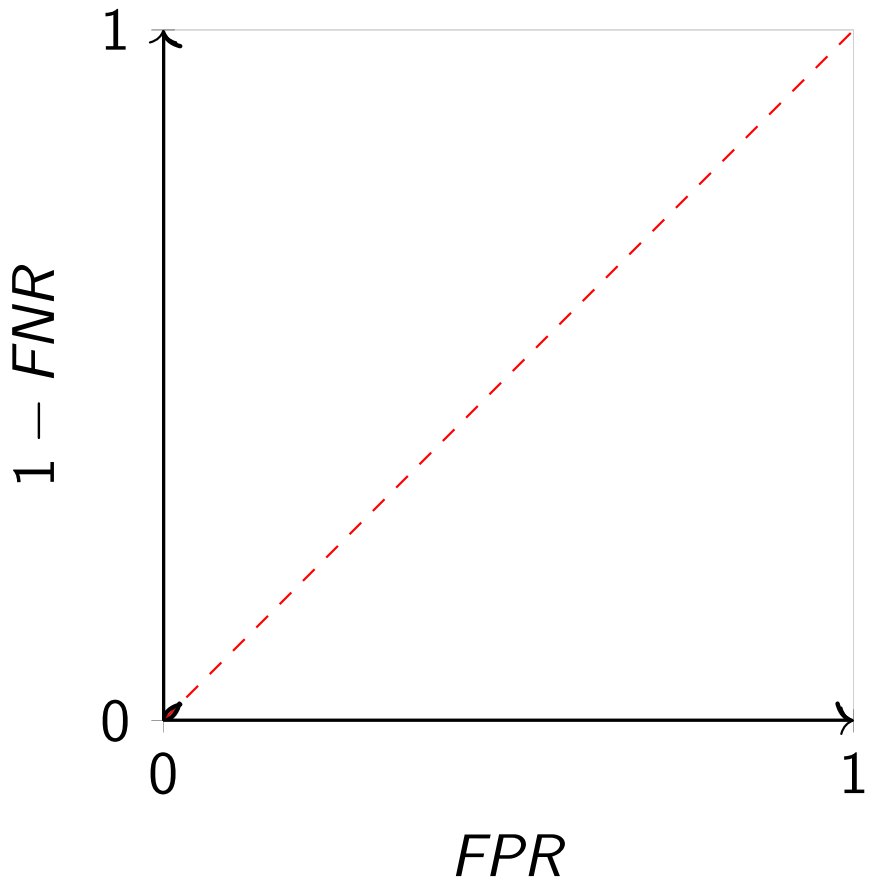
These are defined such that the TPR is $1 - FPR$ and the TNR is $1 - FPR$.

Bayesian Statistical Decision Theory



Receiver Operator Characteristic

We can plot the trade off between FPR and FNR through a **Receiver Operator Characteristic** (ROC) plot. Here the dashed red-line indicates a random guessing decision procedure.



Receiver Operator Characteristic

We can then plot the optimal Bayesian predictor in terms of our posterior probability. The area under curve (AUC) of this Bayesian predictor is as best we can do given our best beliefs. We then assess the quality of our ML algorithms by how close their AUC is to the optimal predictor:

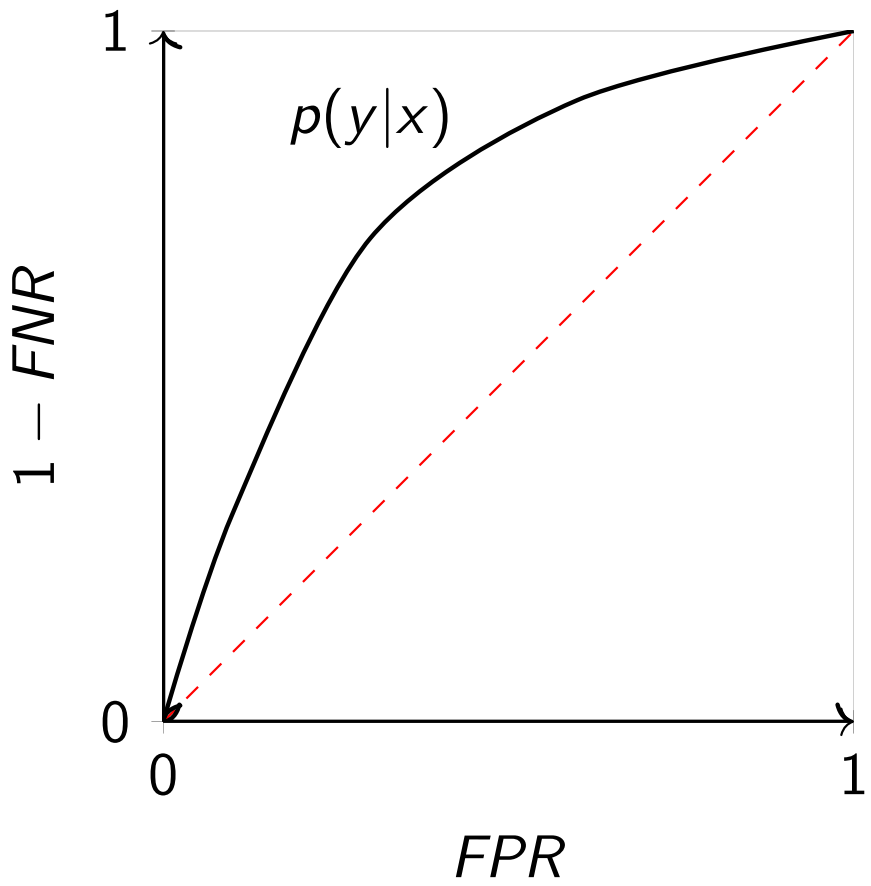


Table of Contents

Prediction as a Decision Problem

There are two philosophically different approaches for evaluating decision procedures:

Prediction as a Decision Problem

There are two philosophically different approaches for evaluating decision procedures:

1. **Internal**: do we do as well as we can do by our own beliefs?

Prediction as a Decision Problem

There are two philosophically different approaches for evaluating decision procedures:

1. **Internal**: do we do as well as we can do by our own beliefs?
2. **External**: do we do as well as we can do reliably?

Prediction as a Decision Problem

There are two philosophically different approaches for evaluating decision procedures:

1. **Internal**: do we do as well as we can do by our own beliefs?
2. **External**: do we do as well as we can do reliably?

The first is captured by **Bayesian Statistical Decision Theory**.

Prediction as a Decision Problem

There are two philosophically different approaches for evaluating decision procedures:

1. **Internal**: do we do as well as we can do by our own beliefs?
2. **External**: do we do as well as we can do reliably?

The first is captured by **Bayesian Statistical Decision Theory**.

We turn to the second now with **Empirical Risk Minimization**.

Population Risk

Definition (Population Risk)

Suppose we have some random variables of features X and targets Y sampled from some distribution $X, Y \sim p^*(\mathbf{x}, y)$.

Population Risk

Definition (Population Risk)

Suppose we have some random variables of features X and targets Y sampled from some distribution $X, Y \sim p^*(\mathbf{x}, y)$. And suppose our model makes predictions $\hat{y} = f(\mathbf{x})$, which are evaluated on loss function l. Then the *population risk* given distribution p^* is:

Population Risk

Definition (Population Risk)

Suppose we have some random variables of features X and targets Y sampled from some distribution $X, Y \sim p^*(\mathbf{x}, y)$. And suppose our model makes predictions $\hat{y} = f(\mathbf{x})$, which are evaluated on loss function l. Then the *population risk* given distribution p^* is:

$$R(f, p^*) := \mathbb{E}_{X,Y \sim p^*(\mathbf{x},y)}[l(y, f(\mathbf{x}))] = \sum_{\mathbf{x},y \in X,Y} l(y, f(\mathbf{x}))p^*(\mathbf{x}, y)$$

Population Risk

Definition (Population Risk)

Suppose we have some random variables of features X and targets Y sampled from some distribution $X, Y \sim p^*(\mathbf{x}, y)$. And suppose our model makes predictions $\hat{y} = f(\mathbf{x})$, which are evaluated on loss function l. Then the *population risk* given distribution p^* is:

$$R(f, p^*) := \mathbb{E}_{X,Y \sim p^*(\mathbf{x},y)}[l(y, f(\mathbf{x}))] = \sum_{\mathbf{x},y \in X,Y} l(y, f(\mathbf{x}))p^*(\mathbf{x}, y)$$

Remark

The population risk is very similar to the posterior risk in that it is a weighted average by some probability distribution and a loss function. However, they should not be confused because the population risk assumes to true, target distribution p^* that is "out in the world", and it weights the losses of our model by the *joint probability* of features and labels, instead of a posterior probability.

Population Risk

Example

Suppose we identify our cancer patients by whether they have IBD or not:

Population Risk

Example

Suppose we identify our cancer patients by whether they have IBD or not:

IBD	Cancer	Drugs	$l(y, \hat{y})$	$p^*(\mathbf{IBD}, Cancer)$
1	1	1	5	0.1
1	1	0	10	
1	0	1	2	0.15
1	0	0	0	
0	1	1	5	0.05
0	1	0	10	
0	0	1	2	0.7
0	0	0	0	

Population Risk

Example

Suppose we identify our cancer patients by whether they have IBD or not:

IBD	Cancer	Drugs	$l(y, \hat{y})$	$p^*(\mathbf{IBD}, Cancer)$
1	1	1	5	0.1
1	1	0	10	
1	0	1	2	0.15
1	0	0	0	
0	1	1	5	0.05
0	1	0	10	
0	0	1	2	0.7
0	0	0	0	

If we administer the drug when IBD is present, our population risk is then $(0.1)(5) + (0.15)(2) + (0.05)(10) + (0.7)(0) = 1.3$.

Population Risk

Example

Suppose we identify our cancer patients by whether they have IBD or not:

IBD	Cancer	Drugs	$l(y, \hat{y})$	$p^*(\mathbf{IBD}, Cancer)$
1	1	1	5	0.1
1	1	0	10	
1	0	1	2	0.15
1	0	0	0	
0	1	1	5	0.05
0	1	0	10	
0	0	1	2	0.7
0	0	0	0	

If we administer the drug when IBD is present, our population risk is then $(0.1)(5) + (0.15)(2) + (0.05)(10) + (0.7)(0) = 1.3$. If we do the opposite, the population risk is $(0.1)(10) + (0.15)(0) + (0.05)(5) + (0.7)(2) = 2.65$.

Empirical Risk

Remark

Our problem is that the population distribution p^* is *unknown*. So we have to approximate it using the empirical distribution from our collected data $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1,\ldots,N}$ that has N samples.

Empirical Risk

Remark

Our problem is that the population distribution p^* is *unknown*. So we have to approximate it using the empirical distribution from our collected data $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1,\ldots,N}$ that has N samples.

Definition (Empirical Risk)

Let our data $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1,\ldots,N}$ have N samples drawn *independently and identically distributed* (i.i.d.) from p^*, and we draw our features X and target Y random variables from the relative frequencies of the data, $X, Y \sim p_{\mathcal{D}}(\mathbf{x}, y)$.

Empirical Risk

Remark

Our problem is that the population distribution p^* is *unknown*. So we have to approximate it using the empirical distribution from our collected data $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1,\ldots,N}$ that has N samples.

Definition (Empirical Risk)

Let our data $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1,\ldots,N}$ have N samples drawn *independently and identically distributed* (i.i.d.) from p^*, and we draw our features X and target Y random variables from the relative frequencies of the data, $X, Y \sim p_{\mathcal{D}}(\mathbf{x}, y)$. Let our model predictions be $\hat{y} = f(\mathbf{x})$ and loss be given by l. Then the **empirical risk** is given by the average loss on the data:

Empirical Risk

Remark

Our problem is that the population distribution p^* is *unknown*. So we have to approximate it using the empirical distribution from our collected data $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1,\ldots,N}$ that has N samples.

Definition (Empirical Risk)

Let our data $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1,\ldots,N}$ have N samples drawn *independently and identically distributed* (i.i.d.) from p^*, and we draw our features X and target Y random variables from the relative frequencies of the data, $X, Y \sim p_{\mathcal{D}}(\mathbf{x}, y)$. Let our model predictions be $\hat{y} = f(\mathbf{x})$ and loss be given by l. Then the **empirical risk** is given by the average loss on the data:

$$R(f, \mathcal{D}) := \mathbb{E}_{X,Y \sim p_{\mathcal{D}}(\mathbf{x},y)}[l(y, f(\mathbf{x}))] = \frac{1}{N}\sum_{i=1}^{N} l(y^{(i)}, f(\mathbf{x}^{(i)}))$$

Empirical Risk

Example

Suppose we collected data from a fraction of oncology practices:

Empirical Risk

Example

Suppose we collected data from a fraction of oncology practices:

IBD	Cancer	Drugs	$l(y, \hat{y})$	Counts	$p_{\mathcal{D}}(\mathbf{IBD}, Cancer)$
1	1	1	5	90	0.06
1	1	0	10		
1	0	1	2	200	0.14
1	0	0	0		
0	1	1	5	30	0.02
0	1	0	10		
0	0	1	2	1100	0.78
0	0	0	0		

Empirical Risk

Example

Suppose we collected data from a fraction of oncology practices:

IBD	Cancer	Drugs	$l(y, \hat{y})$	Counts	$p_{\mathcal{D}}(\mathbf{IBD}, Cancer)$
1	1	1	5	90	0.06
1	1	0	10		
1	0	1	2	200	0.14
1	0	0	0		
0	1	1	5	30	0.02
0	1	0	10		
0	0	1	2	1100	0.78
0	0	0	0		

If we administer the drug when IBD is present, our empirical risk is then $(0.06)(5) + (0.14)(2) + (0.02)(10) + (0.78)(0) = 0.78$.

Empirical Risk

Example

Suppose we collected data from a fraction of oncology practices:

IBD	Cancer	Drugs	$l(y, \hat{y})$	Counts	$p_{\mathcal{D}}(\mathbf{IBD}, Cancer)$
1	1	1	5	90	0.06
1	1	0	10		
1	0	1	2	200	0.14
1	0	0	0		
0	1	1	5	30	0.02
0	1	0	10		
0	0	1	2	1100	0.78
0	0	0	0		

If we administer the drug when IBD is present, our empirical risk is then $(0.06)(5) + (0.14)(2) + (0.02)(10) + (0.78)(0) = 0.78$. If we do the opposite the risk is $(0.06)(10) + (0.14)(0) + (0.02)(5) + (0.78)(2) = 2.26$.

Empirical Risk Minimization

Remark

Our goal then is to choose the model that best minimizes the empirical risk.

Empirical Risk Minimization

Remark

Our goal then is to choose the model that best minimizes the empirical risk. That is we want a predictor f^*_{ERM} from our class of models $\mathcal{H}$ such that it minimizes the average loss on our dataset:

Empirical Risk Minimization

Remark

Our goal then is to choose the model that best minimizes the empirical risk. That is we want a predictor f^*_{ERM} from our class of models $\mathcal{H}$ such that it minimizes the average loss on our dataset:

$$\hat{f}_{ERM} := \arg\min_{f \in \mathcal{H}} R(f, \mathcal{D})$$

Empirical Risk Minimization

Remark

Our goal then is to choose the model that best minimizes the empirical risk. That is we want a predictor f^*_{ERM} from our class of models $\mathcal{H}$ such that it minimizes the average loss on our dataset:

$$\hat{f}_{ERM} := \arg\min_{f \in \mathcal{H}} R(f, \mathcal{D})$$

We consider the *generalization error*, ϵ, to be the difference on the population risk between our ERM predictor and the lowest possible population risk from our set of models $\mathcal{H}$:

Empirical Risk Minimization

Remark

Our goal then is to choose the model that best minimizes the empirical risk. That is we want a predictor f^*_{ERM} from our class of models $\mathcal{H}$ such that it minimizes the average loss on our dataset:

$$\hat{f}_{ERM} := \arg\min_{f \in \mathcal{H}} R(f, \mathcal{D})$$

We consider the *generalization error*, ϵ, to be the difference on the population risk between our ERM predictor and the lowest possible population risk from our set of models $\mathcal{H}$:

$$\epsilon = R(\hat{f}_{ERM}, p^*) - \min_{f \in \mathcal{H}} R(f, p^*)$$

Empirical Risk Minimization

Remark

Our goal then is to choose the model that best minimizes the empirical risk. That is we want a predictor f^*_{ERM} from our class of models $\mathcal{H}$ such that it minimizes the average loss on our dataset:

$$\hat{f}_{ERM} := \arg\min_{f \in \mathcal{H}} R(f, \mathcal{D})$$

We consider the *generalization error*, ϵ, to be the difference on the population risk between our ERM predictor and the lowest possible population risk from our set of models $\mathcal{H}$:

$$\epsilon = R(\hat{f}_{ERM}, p^*) - \min_{f \in \mathcal{H}} R(f, p^*)$$

Of course, *we don't know the population risk so we cannot calculate the generalization error directly*.

Empirical Risk Minimization

Remark

Instead, we aim to show that for certain kinds of estimation errors, we can show that our ERM predictor faithfully tracks the lowest population risk we could achieve.

Empirical Risk Minimization

Remark

Instead, we aim to show that for certain kinds of estimation errors, we can show that our ERM predictor faithfully tracks the lowest population risk we could achieve. That is we try to minimize

$$R(\hat{f}_{ERM}, p^*)$$

and so our ϵ by way of our empirical risk.

Empirical Risk Minimization

Remark

Instead, we aim to show that for certain kinds of estimation errors, we can show that our ERM predictor faithfully tracks the lowest population risk we could achieve. That is we try to minimize

$$R(\hat{f}_{ERM}, p^*)$$

and so our ϵ by way of our empirical risk. Namely, it can be shown that with respect to our sample size N, some $\delta \in (0, 1)$, and the complexity H of our class of models $\mathcal{H}$, our empirical risk upper bounds the true risk the model has on the population with probability $1 - \delta$ if we know our empirical risk samples were drawn i.i.d. from p^*:

Empirical Risk Minimization

Remark

Instead, we aim to show that for certain kinds of estimation errors, we can show that our ERM predictor faithfully tracks the lowest population risk we could achieve. That is we try to minimize

$$R(\hat{f}_{ERM}, p^*)$$

and so our ϵ by way of our empirical risk. Namely, it can be shown that with respect to our sample size N, some $\delta \in (0, 1)$, and the complexity H of our class of models $\mathcal{H}$, our empirical risk upper bounds the true risk the model has on the population with probability $1 - \delta$ if we know our empirical risk samples were drawn i.i.d. from p^*:

$$R(\hat{f}_{ERM}, p^*) \leq R(\hat{f}_{ERM}, \mathcal{D}) + \epsilon_H(N, \delta)$$

Empirical Risk Minimization

Remark

Instead, we aim to show that for certain kinds of estimation errors, we can show that our ERM predictor faithfully tracks the lowest population risk we could achieve. That is we try to minimize

$$R(\hat{f}_{ERM}, p^*)$$

and so our ϵ by way of our empirical risk. Namely, it can be shown that with respect to our sample size N, some $\delta \in (0, 1)$, and the complexity H of our class of models $\mathcal{H}$, our empirical risk upper bounds the true risk the model has on the population with probability $1 - \delta$ if we know our empirical risk samples were drawn i.i.d. from p^*:

$$R(\hat{f}_{ERM}, p^*) \leq R(\hat{f}_{ERM}, \mathcal{D}) + \epsilon_H(N, \delta)$$

The $\epsilon_H(N, \delta)$ is a risk bounding function parameterized by the complexity of our model

Empirical Risk Minimization

Remark

This is called a "what you see is what you get" guarantee.

Empirical Risk Minimization

Remark

This is called a "what you see is what you get" guarantee. Three things should be noted about it going from our drawn sample of data to the population risk:

Empirical Risk Minimization

Remark

This is called a "what you see is what you get" guarantee. Three things should be noted about it going from our drawn sample of data to the population risk:

1. *It is distribution agnostic*. Namely, it does not depend on what the true, underlying population distribution p^* happens to be.

Empirical Risk Minimization

Remark

This is called a "what you see is what you get" guarantee. Three things should be noted about it going from our drawn sample of data to the population risk:

1. *It is distribution agnostic*. Namely, it does not depend on what the true, underlying population distribution p^* happens to be.
2. *It requires our data to be drawn independently and identically distributed from the true distribution*. If we do not have that, then we lose the guarantee.

Empirical Risk Minimization

Remark

This is called a "what you see is what you get" guarantee. Three things should be noted about it going from our drawn sample of data to the population risk:

1. *It is distribution agnostic*. Namely, it does not depend on what the true, underlying population distribution p^* happens to be.
2. *It requires our data to be drawn independently and identically distributed from the true distribution*. If we do not have that, then we lose the guarantee.
3. *The less complex the class of models, the tighter the upper bound*. We don't want models that are too complex less we worry about generalization.

Empirical Risk Minimization

Remark

This is called a "what you see is what you get" guarantee. Three things should be noted about it going from our drawn sample of data to the population risk:

1. *It is distribution agnostic*. Namely, it does not depend on what the true, underlying population distribution p^* happens to be.
2. *It requires our data to be drawn independently and identically distributed from the true distribution*. If we do not have that, then we lose the guarantee.
3. *The less complex the class of models, the tighter the upper bound*. We don't want models that are too complex less we worry about generalization.

In contrast to the internalist method of evaluation, this allows ERM to be largely about just the structure of the machine learning problem. It provides us a *reliability* assurance about how our models will perform.

Table of Contents

Bias-Complexity Tradeoff

- **Bias**: how well a set of machine learning models are suited to predicting targets of a particular population from their features.

Bias-Complexity Tradeoff

- **Bias**: how well a set of machine learning models are suited to predicting targets of a particular population from their features.
- **Complexity**: how well a set of machine learning models can separate data.

Bias-Complexity Tradeoff

- **Bias**: how well a set of machine learning models are suited to predicting targets of a particular population from their features.
- **Complexity**: how well a set of machine learning models can separate data.

The bias-complexity tradeoff—not to be confused with the bias-variance tradeoff!

Bias-Complexity Tradeoff

Recall

We had our generalization error be the difference between the population risk of our ERM model and the lowest possible population risk from our model class or the *approximation error*:

Bias-Complexity Tradeoff

Recall

We had our generalization error be the difference between the population risk of our ERM model and the lowest possible population risk from our model class or the *approximation error*:

$$\epsilon = R(\hat{f}_{ERM}, p^*) - \min_{f \in \mathcal{H}} R(f, p^*)$$

Bias-Complexity Tradeoff

Recall

We had our generalization error be the difference between the population risk of our ERM model and the lowest possible population risk from our model class or the *approximation error*:

$$\epsilon = R(\hat{f}_{ERM}, p^*) - \min_{f \in \mathcal{H}} R(f, p^*)$$

The approximation error is a function of how much *inductive bias* the class of models we are selecting from have, i.e. *how good we are at the target problem*.

Bias-Complexity Tradeoff

Recall

We had our generalization error be the difference between the population risk of our ERM model and the lowest possible population risk from our model class or the *approximation error*:

$$\epsilon = R(\hat{f}_{ERM}, p^*) - \min_{f \in \mathcal{H}} R(f, p^*)$$

The approximation error is a function of how much *inductive bias* the class of models we are selecting from have, i.e. *how good we are at the target problem*.
Importantly, this error is a function of how many models we consider: if we consider more possible models, then the approximation error will go down.

Bias-Complexity Tradeoff

Recall

We had our generalization error be the difference between the population risk of our ERM model and the lowest possible population risk from our model class or the *approximation error*:

$$\epsilon = R(\hat{f}_{ERM}, p^*) - \min_{f \in \mathcal{H}} R(f, p^*)$$

The approximation error is a function of how much *inductive bias* the class of models we are selecting from have, i.e. *how good we are at the target problem*.

Importantly, this error is a function of how many models we consider: if we consider more possible models, then the approximation error will go down. Letting the number of models be a measure of complexity, then **the more complex a machine learning model, the lower the approximation error**.

Bias-Complexity Tradeoff

Theorem

Suppose we want to ensure that our generalization error is less than or equal to ϵ' for all models in class $\mathcal{H}$ with probability $1 - \delta$ as we draw data $\mathcal{D}$ i.i.d. from the population distribution p^*. Then the size of our data set, N, we would need is bounded by the size $|\mathcal{H}|$:

$$N \leq \frac{2 \log(2|\mathcal{H}|/\delta)}{\epsilon'^2}$$

Bias-Complexity Tradeoff

Theorem

Suppose we want to ensure that our generalization error is less than or equal to ϵ' for all models in class $\mathcal{H}$ with probability $1-\delta$ as we draw data $\mathcal{D}$ i.i.d. from the population distribution p^*. Then the size of our data set, N, we would need is bounded by the size $|\mathcal{H}|$:

$$N \leq \frac{2\log(2|\mathcal{H}|/\delta)}{\epsilon'^2}$$

Remark

The number of candidate models in our set $\mathcal{H}$ is a measure of complexity of our machine learning models. This means that because we can bound our generalization error ϵ by ϵ' and supposing we ensure our dataset is the right size, then our generalization error is a function of how complex our model happens to be. **The more complex our machine learning model, the higher the generalization error**.

Bias-Complexity Tradeoff

We have an intrinsic tradeoff then between how well strong our family of machine learning models are at the problem at hand and how our model we select from ERM generalizes beyond its training data.

Bias-Complexity Tradeoff

We have an intrinsic tradeoff then between how well strong our family of machine learning models are at the problem at hand and how our model we select from ERM generalizes beyond its training data.

So when using ERM, our "what you see is what you get" guarantee forces us to choose between:

Bias-Complexity Tradeoff

We have an intrinsic tradeoff then between how well strong our family of machine learning models are at the problem at hand and how our model we select from ERM generalizes beyond its training data.

So when using ERM, our "what you see is what you get" guarantee forces us to choose between:

1. Complex models good at solving the problem at hand.

Bias-Complexity Tradeoff

We have an intrinsic tradeoff then between how well strong our family of machine learning models are at the problem at hand and how our model we select from ERM generalizes beyond its training data.

So when using ERM, our "what you see is what you get" guarantee forces us to choose between:

1. Complex models good at solving the problem at hand.
2. How well we generalize from our training data.

Bias-Complexity Tradeoff

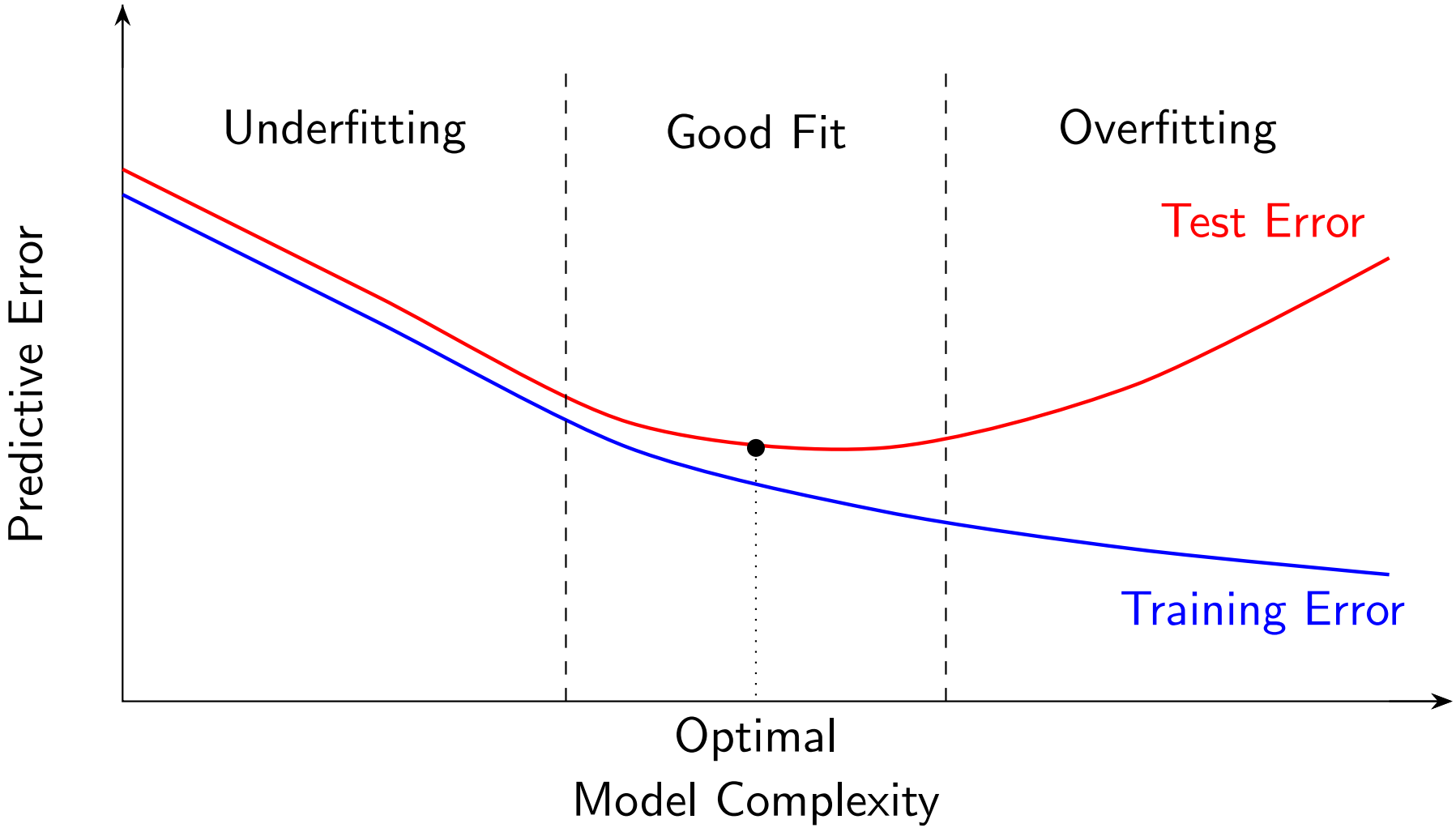
We have an intrinsic tradeoff then between how well strong our family of machine learning models are at the problem at hand and how our model we select from ERM generalizes beyond its training data.

So when using ERM, our "what you see is what you get" guarantee forces us to choose between:

1. Complex models good at solving the problem at hand.
2. How well we generalize from our training data.

This is sometimes called the tradeoff between *overfitting* and *underfitting*.

Underfitting-Overfitting



Summary

So what method should we use for evaluation?

Summary

So what method should we use for evaluation? Both (but mostly ERM)!

We use ERM to construct and select models during training. We check our generalization error and the degree to which we overfit or underfit the data by a **hold-out test set**.

Summary

So what method should we use for evaluation? Both (but mostly ERM)!

We use ERM to construct and select models during training. We check our generalization error and the degree to which we overfit or underfit the data by a **hold-out test set**. We then use AUC to measure how well we do to the optimal Bayes predictor.

Table of Contents

Naive Bayes

Recall

Let $A, B, C \in \Sigma$ for the probability space $(\Omega, \Sigma, \Pr)$. A and B are conditionally independent given C if $\Pr(A|B, C) = \Pr(A|C)$. Alternatively, we can define it as $\Pr(A, B|C) = \Pr(A|C)\Pr(B|C)$.

Naive Bayes

Recall

Let $A, B, C \in \Sigma$ for the probability space $(\Omega, \Sigma, \Pr)$. A and B are conditionally independent given C if $\Pr(A|B, C) = \Pr(A|C)$. Alternatively, we can define it as $\Pr(A, B|C) = \Pr(A|C)\Pr(B|C)$.

Naive Bayes

Consider the classification problem where we suspect that features $X = \mathbf{x}$, where x_i for $i = 1, \ldots, n$ are independent of one another conditional on the class target Y, where y_k for $k = 1, \ldots, m$. That is:

$$\Pr(X = [x_1, \ldots, x_n] | Y = y_k) = \prod_{i=1}^{n} \Pr(x_i | Y = y_k)$$

We can then use this simplifying assumption to directly learn $\Pr(y|\mathbf{x})$, by using Bayes rule.

Naive Bayes continued

The Naive Bayes classifier for target $Y = y_j$:

$$\Pr(Y = y_j | X = [x_1, \ldots, x_n]) = \frac{\prod_{i=1}^{n} \Pr(x_i | Y = y_j) \Pr(Y = y_j)}{\sum_{k=1}^{m} \prod_{i=1}^{n} \Pr(x_i | Y = y_k) \Pr(Y = y_k)}$$

Naive Bayes continued

The Naive Bayes classifier for target $Y = y_j$:

$$\Pr(Y = y_j | X = [x_1, \ldots, x_n]) = \frac{\prod_{i=1}^{n} \Pr(x_i | Y = y_j) \Pr(Y = y_j)}{\sum_{k=1}^{m} \prod_{i=1}^{n} \Pr(x_i | Y = y_k) \Pr(Y = y_k)}$$

Important: This model assumes that our features are conditionally independent of the label, which we normally think is false. However, surprisingly, it works well in most situations.

Naive Bayes

Naive Bayes continued

The Naive Bayes classifier for target $Y = y_j$:

$$\Pr(Y = y_j | X = [x_1, \ldots, x_n]) = \frac{\prod_{i=1}^{n} \Pr(x_i | Y = y_j) \Pr(Y = y_j)}{\sum_{k=1}^{m} \prod_{i=1}^{n} \Pr(x_i | Y = y_k) \Pr(Y = y_k)}$$

Important: This model assumes that our features are conditionally independent of the label, which we normally think is false. However, surprisingly, it works well in most situations.

Example

A common deployment of naive Bayes is as a span classifier in email: we treat the frequencies of certain words occurring, like "Nigerian Prince", as independent of one another conditional on whether the email is spam.

Table of Contents

k-Nearest Neighbors

k-Nearest Neighbors

Suppose we have the classification problem with features $X = \mathbf{x}$ and class targets Y. We have some distance metric $d(\mathbf{x}, \mathbf{x}')$ between our samples, which we use to compute the K closest neighbors in our dataset $\mathcal{D}$ to a sample $\mathbf{x}$, $N_K(\mathbf{x}, \mathcal{D})$. Then our posterior predictive probability given the data is:

k-Nearest Neighbors

Suppose we have the classification problem with features $X = \mathbf{x}$ and class targets Y. We have some distance metric $d(\mathbf{x}, \mathbf{x}')$ between our samples, which we use to compute the K closest neighbors in our dataset $\mathcal{D}$ to a sample $\mathbf{x}$, $N_K(\mathbf{x}, \mathcal{D})$. Then our posterior predictive probability given the data is:

$$\Pr(Y = y | X = \mathbf{x}, \mathcal{D}) = \frac{1}{K} \sum_{n \in N_K(\mathbf{x}, \mathcal{D})} \mathbb{I}(y_n = y)$$

k-Nearest Neighbors

k-Nearest Neighbors

Suppose we have the classification problem with features $X = \mathbf{x}$ and class targets Y. We have some distance metric $d(\mathbf{x}, \mathbf{x}')$ between our samples, which we use to compute the K closest neighbors in our dataset $\mathcal{D}$ to a sample $\mathbf{x}$, $N_K(\mathbf{x}, \mathcal{D})$. Then our posterior predictive probability given the data is:

$$\Pr(Y = y | X = \mathbf{x}, \mathcal{D}) = \frac{1}{K} \sum_{n \in N_K(\mathbf{x}, \mathcal{D})} \mathbb{I}(y_n = y)$$

where $\mathbb{I}$ is the indicator function that returns 1 if $y_n = y$ and 0 otherwise. That is we take an average across K-closest neighbors of those that have the target y.

k-Nearest Neighbors

k-Nearest Neighbors

Suppose we have the classification problem with features $X = \mathbf{x}$ and class targets Y. We have some distance metric $d(\mathbf{x}, \mathbf{x}')$ between our samples, which we use to compute the K closest neighbors in our dataset $\mathcal{D}$ to a sample $\mathbf{x}$, $N_K(\mathbf{x}, \mathcal{D})$. Then our posterior predictive probability given the data is:

$$\Pr(Y = y | X = \mathbf{x}, \mathcal{D}) = \frac{1}{K} \sum_{n \in N_K(\mathbf{x}, \mathcal{D})} \mathbb{I}(y_n = y)$$

where $\mathbb{I}$ is the indicator function that returns 1 if $y_n = y$ and 0 otherwise. That is we take an average across K-closest neighbors of those that have the target y.

Remark

When $K = 1$, we say our target is the same as that of the closest sample.

k-Nearest Neighbors

Definition (Euclidean Distance)

A common distance metric is *Euclidean Distance*:

$$d(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^\intercal (\mathbf{x} - \mathbf{x}')}$$

k-Nearest Neighbors

Definition (Euclidean Distance)

A common distance metric is *Euclidean Distance*:

$$d(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^\intercal (\mathbf{x} - \mathbf{x}')}$$

Example

If $\mathbf{a} = [1, 2]^\intercal, \mathbf{b} = [-6, 5]^\intercal$, measure the Euclidean distance.

k-Nearest Neighbors

Definition (Euclidean Distance)

A common distance metric is *Euclidean Distance*:

$$d(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^\intercal (\mathbf{x} - \mathbf{x}')}$$

Example

If $\mathbf{a} = [1, 2]^\intercal, \mathbf{b} = [-6, 5]^\intercal$, measure the Euclidean distance.

Remark

We can think about K as controlling the complexity of our decision boundary between classes. Unintuitively, the lower K, the *more complex* the decision boundary while the higher the K, the *less complex* the decision boundary.

k-Nearest Neighbors