I. Decision Theory: From Model to Answers

II. Empirical Risk Minimization

Pradeep Ravikumar
Co-instructor: Manuela Veloso

Machine Learning
Jan 29, 2018
Recall: Model-based ML

• Learning: From data to model
  • A model explains how the data was generated
  • E.g. given (symptoms, diseases) data, a model explains how symptoms and diseases are related

• Inference: From model to knowledge
  • Given the model, how can we answer questions relevant to us
  • E.g. given (symptom, disease) model, given some symptoms, what is the disease?
Model to Knowledge

• You know how to learn a model from data, with guarantees
• How do we go from model to knowledge?

• i.e. How do we get the answers we seek from the model?
• E.g. Recall “coin flip” example
  • The model is the Bernoulli distribution
  • The Billionaire might not care about Bernoulli distribution per se, as much as answers to questions such as:
    • Which side is more likely in the next flip?
    • If a bookie gives 3 to 5 odds on tails, should he take the bet?
Model to Knowledge: Plugin Estimates

• In most cases, the knowledge we seek is a fixed function $f(P)$ of the model $P$ of the data
  • is the coin fair: $I(p = \frac{1}{2})$?
  • does the coin have better odds than $3/5$: $I(p \geq 3/5)$
Model to Knowledge: Plugin Estimates

• In most cases, the knowledge we seek is a fixed function $f(P)$ of the distribution $P$ of the data
  • is the coin fair: $I(p = \frac{1}{2})$?
  • does the coin have better odds than $3/5$: $I(p \geq 3/5)$

• Once we learn a model, we have an estimate of the distribution of the data: $P_{\hat{\theta}}$
• So we can simply “plugin” the model for the distribution to get our answers: $f(P_{\hat{\theta}})$
• Is the coin fair: $\mathbb{I}(\theta = 1/2)$
  • Plugin Estimate: $\mathbb{I}(\hat{\theta} = 1/2)$
Specification of Knowledge

• In the previous, the specification of what knowledge we were seeking was through an explicit function of the distribution
  • E.g. is the coin fair? Does the coin have better odds than 3/5, etc.
• But such an explicit specification is not always possible
• Think of the knowledge we seek as some “decision” given the underlying data
• QUESTIONS:
  • How do we characterize such decisions?
  • What is the optimal decision we can make?
  • How do we characterize optimality?
  • Falls under decision theory
Specification of Knowledge

• **Decision theory** can be used to characterize the decision to take
  
  • Through **performance measures** (also known as **loss/utility functions**)
  
  • Whenever you encounter a task, you should automatically think about the appropriate **performance measure/loss function**
From model to knowledge: the general case

Given model parameter $\theta$, we then pick our action $a$ from a set $A$ by solving a decision-theoretic optimization problem:

$$\min_{a \in A} \ell(\theta, a).$$

Here $\ell(\theta, a)$ is the loss of taking action $a$ when model parameter is $\theta$. 
Example: Finance

• Suppose you have a model that specifies the value of apple stock, or how much the stock price will go up in next 1 hour (instead of bias of Billionaire’s coins)

• Your set of actions are how much apple stock to buy or sell

• Given the model, you have to minimize some loss function that balances risk and reward to decide the action $a \in A$ (how much stock to buy or sell).
Example: Electricity Generation

• Suppose you have a model that predicts consumer demand of electricity

• Your set of actions are based on how to schedule the generators of electrical plant

• Given the model, have to minimize some loss function to decide which and how and when of the generation of electricity
Unsupervised vs Supervised

• Learning a Bernoulli distribution as the model for a sequence of coin flips is an example of an “unsupervised learning” problem

• In a “supervised learning” problem, you have an input and an output, and the goal is to predict an output given an input

• Coin Flips: Predict coin flips given “features” about coin

• Finance: predict how much apple stock will move up given economic indicators

• Electricity Generation: predict consumer demand given past demand, weather
Supervised Learning, and Going from Model to Knowledge

• For supervised learning problems, it is natural to talk about the knowledge first, and model second

• In particular, a decision-theoretic **loss function** is part of the problem specification in supervised learning
Supervised Learning Prediction Task

Task:

Given $X \in \mathcal{X}$, predict $Y \in \mathcal{Y}$.

$\equiv$ Construct prediction rule $f : \mathcal{X} \rightarrow \mathcal{Y}$

“Lupus cell (0)”

“Healthy cell (1)”
Example: Supervised Learning Prediction Task

Task:
Given $X \in \mathcal{X}$, predict $Y \in \mathcal{Y}$.

$\equiv$ Construct prediction rule $f : \mathcal{X} \rightarrow \mathcal{Y}$

But I can always come up with a prediction rule: always say it’s not LUPUS!

“Lupus cell (0)”

“Healthy cell (1)”
Example: Supervised Learning Prediction Task

**Task:**

Given $X \in \mathcal{X}$, predict $Y \in \mathcal{Y}$.

$\equiv$ Construct **prediction rule** $f : \mathcal{X} \rightarrow \mathcal{Y}$

To complete the specification of the task, we need something more!!!

“Lupus (0)" \hspace{2in} “Healthy (1)”
Characterize Task using Performance Measures

**Performance Measure:**

\[ \text{loss}(Y, f(X)) \] - Measure of closeness between true label \(Y\) and prediction \(f(X)\)

What is the “loss” I suffer when I take decision \(f\)?

<table>
<thead>
<tr>
<th>(X)</th>
<th>(Y)</th>
<th>(f(X))</th>
<th>(\text{loss}(Y, f(X)))</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Lupus”</td>
<td>“Lupus”</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>“Healthy”</td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

\[ \text{loss}(Y, f(X)) = 1_{\{f(X) \neq Y\}} \] 0/1 loss
Characterize Task using Performance Measures

Performance Measure:

\[ \text{loss}(Y, f(X)) \] - Measure of closeness between true label \( Y \) and prediction \( f(X) \)

What is the “loss” I suffer when I take decision \( f \)?

<table>
<thead>
<tr>
<th>( X )</th>
<th>Share price, ( Y )</th>
<th>( f(X) )</th>
<th>loss(( Y, f(X) ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Past performance, trade volume etc. as of Sept 8, 2010</td>
<td>“$24.50”</td>
<td>“$24.50”</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“$26.00”</td>
<td>1?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“$26.10”</td>
<td>2?</td>
</tr>
</tbody>
</table>

\[ \text{loss}(Y, f(X)) = (f(X) - Y)^2 \quad \text{square loss} \]
Performance Measures

**Performance:**

**Measure:**

\[ \text{loss}(Y, f(X)) \] - Measure of closeness between true label \( Y \) and prediction \( f(X) \)

We don’t just want to correctly label one test sample (in this case, cell image), but most cell images \( X \in \mathcal{X} \)

Given a cell image drawn randomly from the collection of all cell images, how well does the predictor perform on average?

Risk \( R(f) \equiv \mathbb{E}_{XY} [\text{loss}(Y, f(X))] \)
Performance Measures

**Performance:**

Measure:

\[ \text{loss}(Y, f(X)) \] - Measure of closeness between true label \( Y \) and prediction \( f(X) \)

We don’t just want to correctly label one test sample (in this case, cell image), but most cell images \( X \in \mathcal{X} \)

Given a cell image drawn randomly from the collection of all cell images, how well does the predictor perform on average?

What is the “risk” of taking decision \( f \)?

\[
\text{Risk } R(f) \equiv \mathbb{E}_{X,Y} [\text{loss}(Y, f(X))] 
\]
**Performance Measures**

**Performance Measure:** Risk \( R(f) \equiv \mathbb{E}_{XY} [\text{loss}(Y, f(X))] \)

<table>
<thead>
<tr>
<th>loss ( Y, f(X) )</th>
<th>Risk ( R(f) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 1_{{f(X) \neq Y}} )</td>
<td>( P(f(X) \neq Y) )</td>
</tr>
<tr>
<td>0/1 loss</td>
<td>Probability of Error</td>
</tr>
<tr>
<td>( (f(X) - Y)^2 )</td>
<td>( \mathbb{E}[(f(X) - Y)^2] )</td>
</tr>
<tr>
<td>square loss</td>
<td>Mean Square Error</td>
</tr>
</tbody>
</table>

“Anemic cell”

Share Price “$ 24.50”
Bayes Optimal Rule

Knowledge That we seek:

Construct **prediction rule** $f^*: \mathcal{X} \rightarrow \mathcal{Y}$

$$f^*(P) = \arg \min_f \mathbb{E}_{(X,Y) \sim P}[\text{loss}(Y, f(X))]$$

Bayes optimal rule

Best possible performance:

Bayes Risk $R(f^*) \leq R(f)$ for all $f$
## Bayes Optimal Rule

Knowledge That we seek:

$$f^*(P) = \arg \min_f \mathbb{E}_{(X,Y) \sim P} [\text{loss}(Y, f(X))]$$

**Bayes optimal rule**

<table>
<thead>
<tr>
<th>loss(Y, f(X))</th>
<th>Risk R(f)</th>
<th>Bayes Optimal Rule f*(P)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1{f(X) \neq Y}</td>
<td>P(f(X) \neq Y)</td>
<td>f*(P) = \mathbb{I}(P(Y = 1</td>
</tr>
<tr>
<td>0/1 loss</td>
<td>Probability of Error</td>
<td></td>
</tr>
<tr>
<td>(f(X) - Y)^2</td>
<td>\mathbb{E}[(f(X) - Y)^2]</td>
<td>f*(P) = \mathbb{E}(Y</td>
</tr>
<tr>
<td>square loss</td>
<td>Mean Square Error</td>
<td></td>
</tr>
</tbody>
</table>
Bayes Optimal Rule

Knowledge That we seek:  
Construct **prediction rule** \( f^* : \mathcal{X} \to \mathcal{Y} \)

\[
f^*(P) = \arg \min_f \mathbb{E}_{(X,Y) \sim P}[\text{loss}(Y, f(X))]
\]

Bayes optimal rule

Best possible performance:

Bayes Risk \( R(f^*) \leq R(f) \) for all \( f \)
Model-free Methods

Knowledge That we seek:

Construct prediction rule $f^*: \mathcal{X} \to \mathcal{Y}$

$$f^*(P) = \arg\min_f \mathbb{E}_{(X,Y) \sim P}[\text{loss}(Y, f(X))]$$

Bayes optimal rule

Optimal rule is not computable
- depends on unknown distribution $P$ over $(X,Y)$!

MODEL BASED METHODS: Use a model for $P_{XY}$!

MODEL-FREE METHODS: Estimate the knowledge through some learning algorithm that does not go through a model for $P_{XY}$
Model-free Methods

\[
\{(X_i, Y_i)\}_{i=1}^{n}
\]

\( \hat{f}_n \) is a mapping from \( \mathcal{X} \rightarrow \mathcal{Y} \)

\( \hat{f}_n \left( \text{"Anemic cell"} \right) = \text{"Anemic cell"} \)

Test data \( X \)
Popular Approach for model-free ML: Empirical Risk Minimization

Knowledge That we seek:
Construct prediction rule \( f^* : \mathcal{X} \rightarrow \mathcal{Y} \)

\[
f^*(P) = \arg \min_f \mathbb{E}_{(X,Y) \sim P} [\text{loss}(Y, f(X))]
\]

Bayes optimal rule

Given \( \{X_i, Y_i\}_{i=1}^n \), learn prediction rule \( \hat{f}_n : \mathcal{X} \rightarrow \mathcal{Y} \)

Empirical Risk Minimizer:

\[
\hat{f}_n = \arg \min_f \frac{1}{n} \sum_{i=1}^n [\text{loss}(Y_i, f(X_i))]
\]

\[
\frac{1}{n} \sum_{i=1}^n [\text{loss}(Y_i, f(X_i))] \xrightarrow{\text{Law of Large Numbers}} \mathbb{E}_{XY} [\text{loss}(Y, f(X))]
\]
Empirical Risk Minimization

- Very Popular Approach in ML
- Given a loss function, and data, estimate decision function by minimizing “empirical risk”
- Typically restrict decision to lie within some restricted set
  - Could capture our prior information
  - Or just be for computational convenience

\[
\hat{f} = \arg \inf_{f \in \mathcal{F}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \text{loss}(Y_i, f(X_i)) \right\}
\]
Empirical Risk Minimization: Considerations

\[ \hat{f} = \arg \inf_{f \in \mathcal{F}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \text{loss}(Y_i, f(X_i)) \right\} \]

• **Computational Considerations**: How do we solve the above optimization problem in a computationally tractable manner?

• **Statistical Considerations**: What guarantees do I have for the empirical risk minimizer (ERM) estimator?
Statistical Considerations: Consistency and Rate of Convergence

• How does the performance of the algorithm compare with ideal performance?

\[
\text{Excess Risk} = \mathbb{E}_{D_n} [R(\hat{f}_n)] - R(f^*)
\]

• Consistent algorithm if Excess Risk \(\rightarrow 0\) as \(n \rightarrow \infty\)

• Rate of Convergence

More later ...
Computational Considerations

\[ \hat{f} = \arg \inf_{f \in \mathcal{F}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \text{loss}(Y_i, f(X_i)) \right\} \]

- Even when class of functions is simple (e.g. class of linear functions), the above optimization need not be **convex**
- This non-convexity, and consequently, computational intractability holds for 0-1 loss classification