• Homework 3: Due **Mar 19 (Friday)**.
  • Keep track of your late days
  • Utilize the office hours early. Don’t wait till the last day

• Exam 1: **Mar 23 (Tuesday)**
  • Duration: 75+5 Minutes
  • Content: LFD Chapters 1 to 5
  • Time: during lecture time (exceptions: students who have informed me last week)
  • Format: Gradescope online exam + Zoom (with camera on)
  • Information access during exam:
    • Allowed: Textbook, slides, hardcopy materials (e.g., your own notes)
    • Not allowed: search for information online during exam, talk to any other persons

• Other notes
  • **Follow Piazza announcements**
  • Practice questions are now on Gradescope
  • **This Thursday lecture will be a review lecture**
Recap
Decision Tree
**Decision Tree Hypothesis**

**Pros**
- **Easy to interpret** (interpretability is getting attention and is important in some domains)
- **Can handle multi-type data** (Numerical, categorical, ...)
- **Easy to implement** (Bunch of if-else rules)

**Cons**
- Generally speaking, **bad generalization**
- **VC dimension is infinity**
- **High variance** (small change of data leads to very different hypothesis)
- **Easily overfit**

**Why we care?**
- One of the classical model
- Building block for other models (e.g., random forest)

**Credit Card Approval Example**

![Credit Card Approval Decision Tree](attachment:image.png)
Learning Decision Tree from Data

• Given dataset $D$, how to learn a decision tree hypothesis?

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1</td>
<td>+1</td>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td>+1</td>
<td>+1</td>
<td>-1</td>
<td>+1</td>
</tr>
<tr>
<td>+1</td>
<td>-1</td>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td>+1</td>
<td>-1</td>
<td>-1</td>
<td>+1</td>
</tr>
<tr>
<td>-1</td>
<td>+1</td>
<td>+1</td>
<td>+1</td>
</tr>
<tr>
<td>-1</td>
<td>+1</td>
<td>-1</td>
<td>+1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>+1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

• Potential approach:
  • Empirical risk minimization
  • Find $g = \arg\min_{h \in H} E_{in}(h)$

• Multiple decision trees with zero $E_{in}$

How to avoid overfitting?
Learning Decision Tree from Data

• Conceptual intuition to deal with overfitting
  • Regularization: Constrain $H$

• Informally,
  \[
  \text{minimize } E_{in}(\overline{w}) \\
  \text{subject to } \text{size}(\text{tree}) \leq C
  \]

• This optimization is generally computationally intractable.
• Most decision tree learning algorithms rely on \textit{heuristics} to approximate the goal.
Greedy-Based Decision Tree Algorithm

• Greedily, recursively, choose the next feature to split

• DecisionTreeLearn($D$): Input a dataset $D$, output a decision tree hypothesis
  • Create a root node
  • If termination conditions are met
    • return a single node tree with leaf prediction based on $D$
  • Else: Greedily find a feature $A$ to split according to split criteria
    • For each possible value $v_i$ of $A$
      • Let $D_i$ be the dataset containing data with value $v_i$ for feature $A$
      • Create a subtree DecisionTreeLearn($D_i$) that being the child of root

• Most decision tree learning algorithms follow this template, but with different choices of heuristics
ID3: Using Information Gain as Selection Criteria

- Information gain of choosing feature $A$ to split
  - $Gain(D, A) = H(D) - \sum_i \frac{|D_i|}{|D|} H(D_i)$ [The amount of decrease in entropy]
- ID3: Choose the split that maximize $Gain(D, A)$

DecisionTreeLearn($D$)
- Create a root node $r$
- If termination conditions are met
  return a single node tree with leaf prediction based on
- Else: Greedily find a feature $A$ to split according to split criteria
  For each possible value $v_i$ of $A$
    - Let $D_i$ be the dataset containing data with value $v_i$ for feature $A$
    - Create a subtree DecisionTreeLearn($D_i$) that being the child of root $r$

Notations:
- $H(D)$: Entropy of $D$
- $|D|$ is the number of points in $D$

- ID3 termination conditions
  - If all labels are the same
  - If all features are the same
  - If dataset is empty
- ID3 leaf predictions
  - Most common labels (majority voting)
- ID3 split criteria
  - Information gain
Illustration of “High Variance” of Decision Trees

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain</td>
<td>Before</td>
<td>Both</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>Rain</td>
<td>During</td>
<td>Both</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>Rain</td>
<td>During</td>
<td>Both</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>Rain</td>
<td>After</td>
<td>Backpack</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>Rain</td>
<td>After</td>
<td>Backpack</td>
<td>Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>Rain</td>
<td>After</td>
<td>Lunchbox</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>No Rain</td>
<td>Before</td>
<td>Backpack</td>
<td>Tired</td>
<td>Bike</td>
</tr>
<tr>
<td>No Rain</td>
<td>Before</td>
<td>Lunchbox</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>No Rain</td>
<td>Before</td>
<td>Lunchbox</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>No Rain</td>
<td>During</td>
<td>Backpack</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>No Rain</td>
<td>During</td>
<td>Both</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Backpack</td>
<td>Not Tired</td>
<td>Bike</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Backpack</td>
<td>Tired</td>
<td>Bike</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Both</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Both</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Lunchbox</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain</td>
<td>Before</td>
<td>Both</td>
<td>Not Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>Rain</td>
<td>During</td>
<td>Both</td>
<td>Not Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>Rain</td>
<td>During</td>
<td>Both</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>Rain</td>
<td>After</td>
<td>Backpack</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>Rain</td>
<td>After</td>
<td>Backpack</td>
<td>Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>Rain</td>
<td>After</td>
<td>Lunchbox</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>No Rain</td>
<td>Before</td>
<td>Backpack</td>
<td>Tired</td>
<td>Bike</td>
</tr>
<tr>
<td>No Rain</td>
<td>Before</td>
<td>Lunchbox</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>No Rain</td>
<td>Before</td>
<td>Lunchbox</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>No Rain</td>
<td>During</td>
<td>Backpack</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>No Rain</td>
<td>During</td>
<td>Both</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Backpack</td>
<td>Not Tired</td>
<td>Bike</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Backpack</td>
<td>Tired</td>
<td>Bike</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Both</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Both</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>No Rain</td>
<td>After</td>
<td>Lunchbox</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
</tbody>
</table>
Illustration of "High Variance" of Decision Trees

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain</td>
<td>Before</td>
<td>Both</td>
<td>Tired</td>
<td>Drive</td>
</tr>
<tr>
<td>Rain</td>
<td>During</td>
<td>Both</td>
<td>Not Tired</td>
<td>Metro</td>
</tr>
</tbody>
</table>

High variance: A small deviation of data would lead to very different learned hypothesis
**Decision Tree Hypothesis**

- **Pros**
  - Easy to interpret (interpretability is getting attention and is important in some domains)
  - Can handle multi-type data (Numerical, categorical, ...)
  - Easy to implement (Bunch of if-else rules)

- **Cons**
  - Generally speaking, bad generalization
  - VC dimension is infinity
  - High variance (small change of data leads to very different hypothesis)
  - Easily overfit

- **Why we care?**
  - One of the classical model
  - Building block for other models

---

**Credit Card Approval Example**

```
<table>
<thead>
<tr>
<th>Annual Income</th>
<th>have debt?</th>
</tr>
</thead>
<tbody>
<tr>
<td>≥ 100k</td>
<td>&gt; 20k</td>
</tr>
<tr>
<td>Approve</td>
<td>Approve</td>
</tr>
<tr>
<td>Deny</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>Deny</td>
</tr>
</tbody>
</table>
```

Today’s Lecture

The notes are not intended to be comprehensive. They should be accompanied by lectures and/or textbook. Let me know if you spot errors.
Ensemble Learning
Ensemble Learning

- Assume we are given a set of learned hypothesis
  - \( g_1, g_2, \ldots, g_M \)

- What can we do?
  - Select the best one: use validation for model selection
  - What if all of them are not good enough?

- Can we **aggregate** them?
Aggregation

• Given a set of weak learners $g_1, \ldots, g_M$, how to output a stronger learner that has better performance?

• Uniform aggregation
  • Regression (average): $\bar{g}(\vec{x}) = \frac{1}{M} \sum_{m=1}^{M} g_m(\vec{x})$
  • Classification (majority vote): $\bar{g}(\vec{x}) = \text{sign} \left( \frac{1}{M} \sum_{m=1}^{M} g_m(\vec{x}) \right)$

• Weighted aggregation
  • Regression (average): $\bar{g}(\vec{x}) = \frac{1}{M} \sum_{m=1}^{M} \alpha_m g_m(\vec{x})$
  • Classification (majority vote): $\bar{g}(\vec{x}) = \text{sign} \left( \frac{1}{M} \sum_{m=1}^{M} \alpha_m g_m(\vec{x}) \right)$

• Stacking (won’t talk about this in this course)
  • Take the prediction of $g_1$ to $g_m$ as input features, train another model on top of that

Mathematically, majority voting and average is similar with +1/-1 labels
Is Aggregation a Good Idea?

• Some illustrative examples
Is Aggregation a Good Idea?

• Some illustrative examples
Is Aggregation a Good Idea?

• Maybe
  • If the hypothesis is diverse, and “on average” they seem good
  • (If you take humans as weak learners, this is almost democracy)

• Question:
  • How do we find a set of hypothesis that are diverse and “on average” good
  • How do we aggregate the set of hypothesis

• Ensemble learning
  • Bagging – Random Forest (This lecture)
  • Boosting – AdaBoost (Next lecture)
Diverse Weak Learners

• One common way to construct weak learners is via decision trees

• Fully grown decision trees
  • High variance
  • Low bias

• Decision stump (One-depth decision trees, split on only one attribute)
  • Low variance
  • High bias

• We will discuss how to construct diverse weak learners next
  • Hint: Randomization
Bagging

Bootstrapped Aggregating
Review: Bias-Variance Decomposition

- \( f \): sine function, \( H: h(x) = ax + b, N=2 \)

- Observations
  - The variance of each learned hypothesis is high
  - The variance of “average” of them (\( \bar{g}(\hat{x}) \)) is lower

- Can we apply similar intuitions?
  - Generate a lot of high-variance but low bias weak learners
  - Aggregate them using uniform aggregation

For each dataset, learn a hypothesis.

Draw many datasets, learn many hypothesis

Take average.

We only have one dataset in practice!
Bootstrapping

• Intuition:
  • Use the dataset $D$ we have to approximate the data distribution
  • Sample (with replacement) from $D$ to create bootstrapped datasets

• Bootstrapping:
  • Let $D = \{(\tilde{x}_1, y_1), \ldots, (\tilde{x}_N, y_N)\}$ be the dataset we have
  • Repeatedly uniformly sample $N$ points from $D$ with replacement
    • The number of sampled points doesn’t have to be $N$, but it’s a reasonable/common choice.
  • Obtain many bootstrapped datasets
    • $\tilde{D}^{(1)} = \{(x_1, y_1), (x_1, y_1), (x_4, y_4), \ldots\}$
    • $\ldots$
    • $\tilde{D}^{(M)}$
Bagging - Bootstrapped Aggregating

• Bootstrap $M$ datasets $\{\tilde{D}^m\}$ and learn a hypothesis from each of them

• Aggregate the learned hypothesis (assume we are doing classification)
Bagging - Bootstrapped Aggregating

• Bootstrap $M$ datasets $\{\tilde{D}^{(m)}\}$ and learn a hypothesis from each of them

\[
\begin{align*}
\tilde{D}^{(1)} & \rightarrow g_1 \\
\tilde{D}^{(2)} & \rightarrow g_2 \\
\vdots & \\
\tilde{D}^{(M)} & \rightarrow g_M
\end{align*}
\]

• Aggregate the learned hypothesis (assume we are doing classification)

\[
G(\tilde{x}) = \tilde{g}(\tilde{x}) = \text{sign} \left( \frac{1}{M} \sum_{m=1}^{M} g_m(\tilde{x}) \right)
\]
Why/When Bagging Might Be Helpful?

• What we know from statistics
  • Consider $M$ independent random variables $x_1, x_2, \ldots, x_M$ with variance $\sigma^2$
  • The variance of $\frac{1}{M} \sum_{m=1}^{M} x_m$ is $\frac{\sigma^2}{M}$

• If you have “weak learners” that have high variance but low bias
  • Bagging helps reduce the variance and maintain low bias
  • From bias-variance decomposition, this implies a strong learner
Exercise:
Given a dataset $D$ with $N$ points. Consider we bootstrap a dataset $\tilde{D}^{(m)}$ by sampling $N$ points with replacement from $D$, what’s the probability that a given point $(\tilde{x}_n, y_n)$ is not in $\tilde{D}^{(m)}$?
Out-Of-Bag (OOB) Error
Probability for a Point to be Out of Bag

• Consider we bootstrap a dataset $\widetilde{D}^{(m)}$ by sampling $N$ points from $D$, what's the probability that a given point $(\tilde{x}_n, y_n)$ is not in $\widetilde{D}^{(m)}$.

$$
(1 - \frac{1}{N})^N = \left(\frac{1}{1 + \frac{N}{N-1}}\right)^N
\approx \frac{1}{e} \approx 0.36 \text{ when } N \to \infty
$$

When $N$ is large, for each bootstrapped dataset $\widetilde{D}^{(m)}$, a significant proportion of points in $D$ is not included.

• A point that is not in $\widetilde{D}^{(m)}$ is not involved in training $g_m$
  • Can we utilize it to validate the performance of $g_m$?
  • Yes, but we care about the overall performance, not just the performance of $g_m$...
### Out-Of-Bag (OOB) Error

<table>
<thead>
<tr>
<th></th>
<th>$\tilde{D}^{(1)}$</th>
<th>$\tilde{D}^{(2)}$</th>
<th>$\tilde{D}^{(3)}$</th>
<th>$\tilde{D}^{(4)}$</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\tilde{x}_1, y_1)$</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>...</td>
</tr>
<tr>
<td>$(\tilde{x}_2, y_2)$</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$(\tilde{x}_N, y_N)$</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>...</td>
</tr>
</tbody>
</table>

Whether a point is in a bootstrapped dataset

- Recall that we learn $g_1, ..., g_M$ using $\tilde{D}^{(1)}, ..., \tilde{D}^{(M)}$

- Which set of hypothesis can $(\tilde{x}_1, y_1)$ be used for validation?
Out-Of-Bag (OOB) Error

<table>
<thead>
<tr>
<th>( \hat{x}_1, y_1 )</th>
<th>( \hat{x}_2, y_2 )</th>
<th>...</th>
<th>( \hat{x}_N, y_N )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

\[ \bar{D}(1) \quad \bar{D}(2) \quad \bar{D}(3) \quad \bar{D}(4) \quad ... \]

\( (\hat{x}_1, y_1) \): the aggregation of hypothesis that \( \hat{x}_n \) is OOB of
- \( G^-_1 = \text{aggregate}(g_3, g_4, ...) \)
- \( G^-_2 = \text{aggregate}(g_2, g_3, g_4, ...) \)
- \( G^-_N = \text{aggregate}(g_1, ..., ...) \)

**Agggregate:**
- Majority voting for classification
- Average for regression

**Error:**
- Binary error for classification
- Squared error for regression

\[ E_{OOB}(G) = \frac{1}{N} \sum_{n=1}^{N} \text{error}(G^-_n(\hat{x}_n), y_n) \]

Whether a point is in a bootstrapped dataset
Out-Of-Bag (OOB) Error

\[ E_{OOB}(G) = \frac{1}{N} \sum_{n=1}^{N} \text{error}(G_n^-(\hat{x}_n), y_n) \]

• Bagging provided an intrinsic mechanism for us to perform validation
  • We don’t need to split the dataset into training and validation

• Practical issues (you might face this in HW4)
  • What if some \( \hat{x}_n \) appears in all bootstrapped datasets?
    • The probability of this happening is small when the number of bags \( M \) is large
  • Let \( S \) be the set of points that is out of bag for at least one bootstrapped dataset
    • \( E_{OOB}(G) = \frac{1}{|S|} \sum_{(\hat{x}_n, y_n) \in S} \text{error}(G_n^-(\hat{x}_n), y_n) \)
Random Forest
What We Have Learned

Bagging:
A method to generate and aggregate many high-variance weak learners into a stronger one.

Decision tree:
Various nice properties
Bad generalization
- Due to high variance

Random Forest:
1. Construct many random trees
2. Aggregate the random trees
Random Forest

• Construct many random trees
  • Bootstrapping datasets and learn a max-depth tree for each of them
  • Other randomizations (not required in HW4)
    • When choosing split features, choose from a random subset (instead of all features)
    • Randomly project features (similar to non-linear transformation) for each tree

• Aggregate the random trees
  • Classification: Majority vote $\bar{g}(\hat{x}) = \text{sign}\left(\frac{1}{M} \sum_{m=1}^{M} g_m(\hat{x})\right)$
  • Regression: Average $\bar{g}(\hat{x}) = \frac{1}{M} \sum_{m=1}^{M} g_m(\hat{x})$
Questions?

• Note that in HW4, you will be asked to implement Bagging Decision Tree and calculate the OOB errors.

• Make sure you know the definitions/algorithm well.
Brief Discussion on Feature Importance

• Not all features are equally important
  • Some features could be redundant -- (birth year, age)
  • Some features might be irrelevant -- feature: name, label: prob of heart attack

• How do we know which features are more important?
  • Linear models:
    • The size of the weight is a proxy for feature importance
    • Applying L1 regularization is one way to reduce the number of features.
  • Decision tree:
    • The feature closer to the root is probably more important
  • Random forest:
    • Average "information gain" of all trees is a proxy for feature importance

• See LFD e-Chap 9.2 for more discussion on feature selection