EECS 445 – Lecture 10
Decision trees + Ensembles

Professor Maggie Makar

Outline

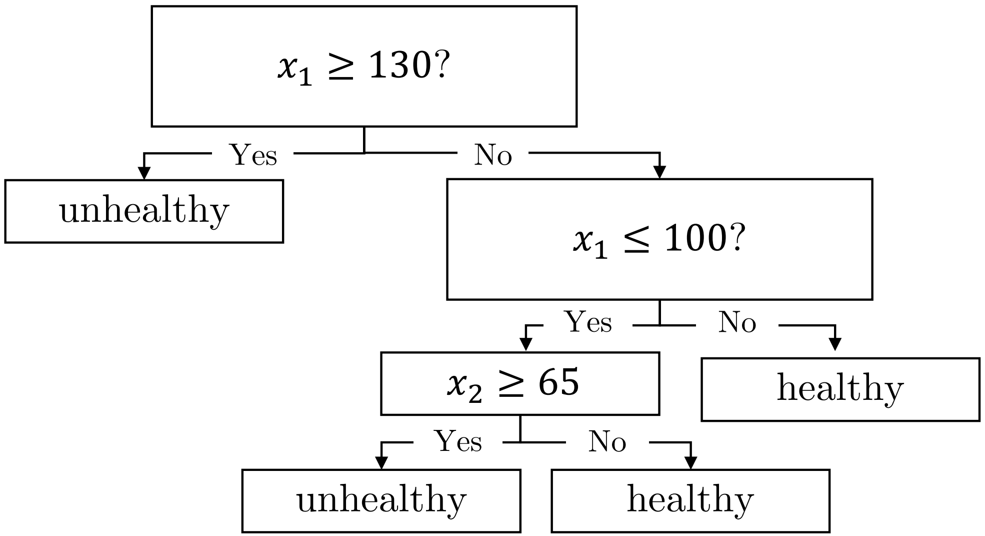
- Recap:
  - What are decision trees?
  - How do we train DTs? (part 1)

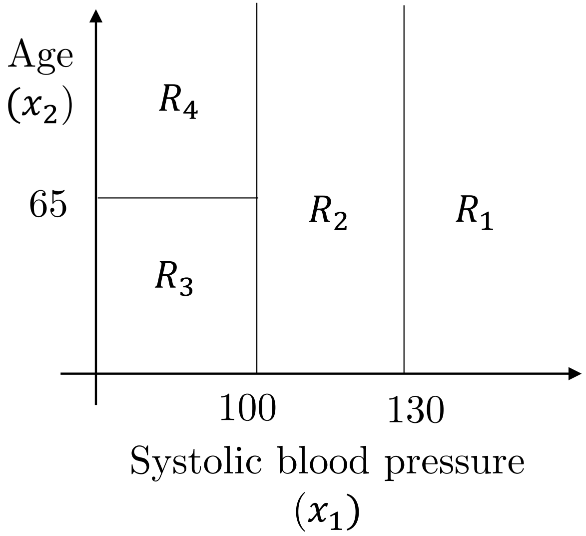
- How do we train DTs? (part 2)
  - Measuring uncertainty
  - The algorithm
  - Bias Variance trade-offs
  - Ensemble methods

Decision trees

μ_m = majority label in region R_m

$$f(\bar{x}) = \sum_{m}^{M} \mu_m [\![\bar{x} \in R_m]\!]$$





Training decision trees

$\{\mu_m\}_{m=1}^{M}$ $\{R_m\}_{m=1}^{M}$

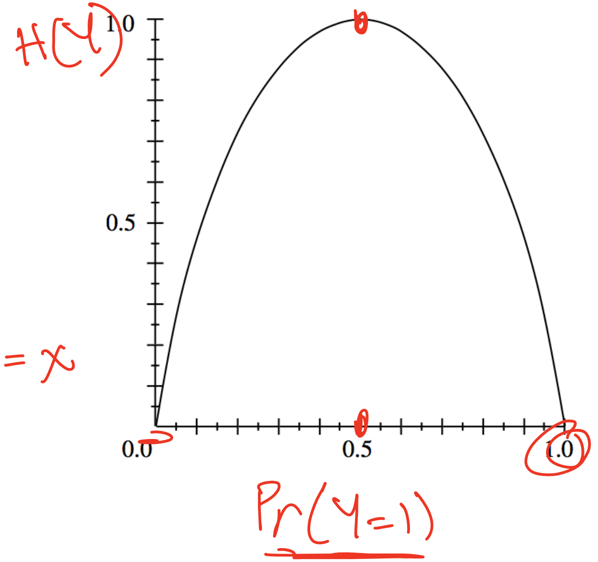
- No closed form solution, gradient descent does not work
- Brute-force is too computationally expensive
- Will use greedy approach:
  - Evaluate one split at a time
  - Pick splits that minimize the uncertainty in the label
    - Measure uncertainty using Shannon entropy

Entropy and conditional entropy

Measure of uncertainty in the value of Y

- Entropy:

$$H(Y) = -\sum_{k=1}^{K} p(Y = y_k) \log_2 p(Y = y_k)$$



- The entropy of Y conditioned on $X = x$:

$$H(Y \mid X = x) = -\sum_{k=1}^{K} p(Y = y_k \mid X = x) \log_2 p(Y = y_k \mid X = x)$$

How uncertain am I about Y if I know that X=x

- Conditional entropy:

$$H(Y \mid X) = \sum_{x} p(X = x) H(Y \mid X = x)$$

How uncertain am I about Y when I learn the value of X

Information Gain (aka Mutual Information)

- Information gain (IG):
  - Decrease in entropy (uncertainty) in Y after knowing the value of X

$$IG(Y, X) = H(Y) - H(Y \mid X)$$

$\underline{IG(Y, X_1) =}$

$H(Y) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} \approx 0.97$

$H(Y \mid X_1) = P(X_1 = \text{Sunny})\, H(Y \mid X_1 = \text{sunny})$
$\quad + P(X_1 = \text{cloudy})\, H(Y \mid X_1 = \text{cloudy})$

$= \frac{2}{5}(1) + \frac{3}{5}\left[-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3}\right] \approx 0.95$

$H(Y) - H(Y \mid X) \approx 0.97 - 0.95 = 0.02$

x_1	x_2	y
Sunny	Raining	0
Sunny	Dry	1
Cloudy	Raining	0
Cloudy	Dry	0
Cloudy	Dry	1

Marginal and conditional entropy

- Check your understanding:
  - When does $H(Y \mid X) = 0$?
  - When does $H(Y \mid X) = H(Y)$? X is indep to Y.
  - What is the IG if X, Y are independent?

TL;DPA:

1. Getting the true optimal tree ~~problem~~ is hard

2. Forget optimal, we'll shoot for a close-to-optimal approach that is *greedy*

3. Our approach evaluates splits based on measures of uncertainty

Training decision trees: the algorithm

1. Start with an empty tree
2. Split on the best feature and split value
3. Recurse until a stopping criterion is met

Step #2: Split on the best feature and split value

Consider all possible splits...

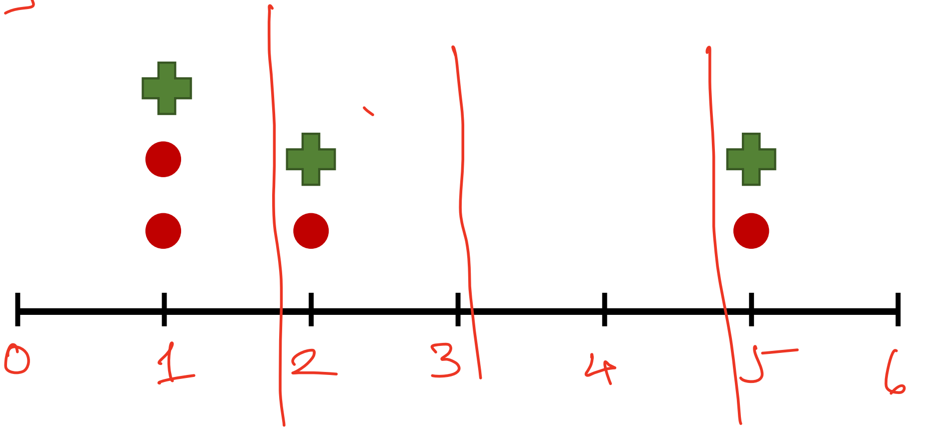
- Continuous variable (e.g., age)

Age	Specie	Wears Tin-foil Hat	Plotting World Domination
1	Cat	Yes	Yes
5	Dog	No	No
1	Snake	Yes	Yes
2	Cat	No	Yes
5	Snake	Yes	Yes
2	Snake	No	No
1	Dog	Yes	No

$$H(Y \mid [\![Age < 5]\!]) = P(Age<5)\, H(Y \mid Age < 5) + P(Age \geq 5)\, H(Y \mid Age \geq 5)$$

$$= \frac{5}{7}\left[-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right] + \frac{2}{7}[1]$$

$$H(Y \mid [\![Age < 4]\!]) = H(Y \mid [\![Age < 5]\!])$$



Step #2: Split on the best feature and split value

Consider all possible splits...

- Categorical variable (e.g., Specie)

Method 1: Binary trees.

Specie = Cat — Y / N

Specie = Dog — Y / N

Specie = Snake — Y / N

$$H(Y \mid [Sp = Cat]) = P(Sp = C)\, H(Y \mid Sp = Cat) + P(Sp \neq C)\, H(Y \mid Sp \neq cat)$$

Age	Specie	Wears Tin-foil Hat	Plotting World Domination
1	Cat	Yes	Yes
5	Dog	No	No
1	Snake	Yes	Yes
2	Cat	No	Yes
5	Snake	Yes	Yes
2	Snake	No	No
1	Dog	Yes	No

Step #2: Split on the best feature and split value

Consider all possible splits...

- Categorical variable (e.g., Specie)

Method 2: Non bin trees!

Age	Specie	Wears Tin-foil Hat	Plotting World Domination
1	Cat	Yes	Yes
5	Dog	No	No
1	Snake	Yes	Yes
2	Cat	No	Yes
5	Snake	Yes	Yes
2	Snake	No	No
1	Dog	Yes	No

$$H(Y \mid \text{Specie}) = P(Sp=\text{cat})\,H(Y \mid Sp=\text{cat}) + P(Sp=\text{Dog})\,H(Y \mid Sp=\text{Dog}) + P(Sp=\text{Snake})\,H(Y \mid Sp=\text{Snake}).$$

Step #2: Split on the best feature and split value

Consider all possible splits...

- Binary variable (e.g., Tinfoil hat)

Tinfoil hat

Y N

$H(Y \mid TF\ hat) = H(Y \mid [TF\ hat = Yes])$

Age	Specie	Wears Tinfoil Hat	Plotting World Domination
1	Cat	Yes	Yes
5	Dog	No	No
1	Snake	Yes	Yes
2	Cat	No	Yes
5	Snake	Yes	Yes
2	Snake	No	No
1	Dog	Yes	No

Step #2: Split on the best feature and split value

- Best feature that minimizes entropy vs. maximize IG

Min Entropy

$$X^* = \arg\min_X H(Y|X)$$

Max IG

$$X^* = \arg\max_X IG(Y,X)$$
$$= \arg\max_X H(Y) - H(Y|X)$$
$$= \arg\max_X - H(Y|X)$$
$$= \arg\min_X H(Y|X)$$

Step #3: Recurse until a stopping criteria is met

What's a good stopping criteria?

1 When all data have the same label (assuming this is possible)

Outlook	Temperature	Humidity	Wind	PlayTennis
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes

Learning decision trees: algorithm stopping criteria

What's a good stopping criteria?

1 When all data have the same label (assuming this is possible)

2 If all data have identical features (no further splits possible)

Outlook	Temperature	Humidity	Wind	PlayTennis
Sunny	Hot	High	Strong	No
Sunny	Hot	High	Strong	No
Sunny	Hot	High	Strong	Yes

Step # 3: Recurse until a stopping criteria is met

1. When all records have the same label (assumes this is possible)
2. If all records have identical features (no further splits possible)
3. If all features have zero IG

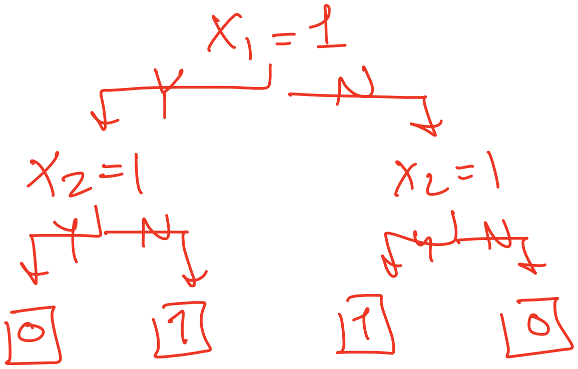
Why is it shortsighted to stop when IG = 0?

$H(Y) = 1$

$H(Y|X_1) = 1$

$H(Y|X_2) = 1$

Accuracy = 50%



accuracy = 100%

x_1	x_2	y
1	1	0
1	0	1
0	1	1
0	0	0

Learning decision trees: pseudocode

```
BuildTree(DS)
    if (y(i) == y) for all examples in DS
            return y
    elseif (x(i) == x) for all examples in DS
            return majority label
    else
        xs,ts = argmin_{x,t} H(y|[[x > t]])
        DSg = {examples in DS where xs ≥ ts}
        BuildTree(DSg)
        DSl = {examples in DS where xs < ts}
        BuildTree(DSl)
```

Learning decision trees: pseudocode

```
BuildTree(DS)
    if (y(i) == y) for all examples in DS
            return y
    elseif (x(i) == x) for all examples in DS
            return majority label
    else
        xs,ts = argmin_{x,t} H(y|[[x > t]])
        DSg = {examples in DS where xs ≥ ts}
        BuildTree(DSg)
        DSl = {examples in DS where xs < ts}
        BuildTree(DSl)
```

stopping criteria 1

Outlook	Temperature	Humidity	Wind	PlayTennis
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes

Learning decision trees: pseudocode

```
BuildTree(DS)
    if (y(i) == y) for all examples in DS
            return y
    elseif (x(i) == x) for all examples in DS
            return majority label
    else
        xs,ts = argminx,t H(y|[[x > t]])
        DSg = {examples in DS where xs ≥ ts}
        BuildTree(DSg)
        DSl = {examples in DS where xs < ts}
        BuildTree(DSl)
```

stopping criteria 2

Outlook	Temperature	Humidity	Wind	PlayTennis
Sunny	Hot	High	Strong	No
Sunny	Hot	High	Strong	No
Sunny	Hot	High	Strong	Yes

Learning decision trees: pseudocode

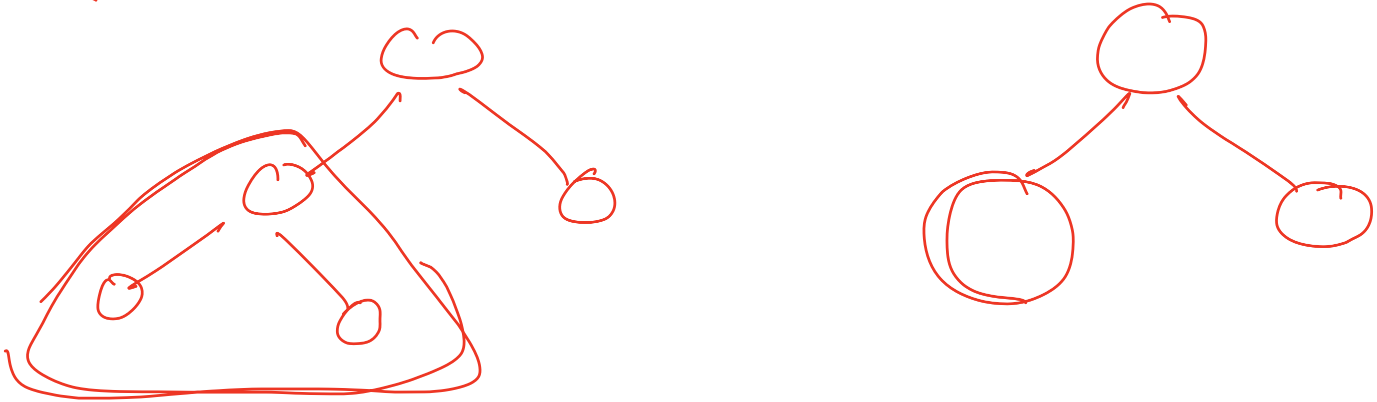
```
BuildTree(DS)
    if (y^(i) == y) for all examples in DS
            return y
    elseif (x^(i) == x) for all examples in DS
            return majority label
    else
        x_s,t_s = argmin_x,t H(y|⟦x > t⟧)
        DS_g = {examples in DS where x_s ≥ t_s}
        BuildTree(DS_g)
        DS_l = {examples in DS where x_s < t_s}
        BuildTree(DS_l)
```

Regularization

Shouldn't stop growing the tree when we stop seeing reductions in IG → prone to overfitting.

Methods to prevent overfitting

1. Set a max depth
2. Grow full tree then prune (e.g., weakest link pruning)



Example: is your roommate good or evil? (Your turn)

Find the best first split

	Height (cm)	Mask	Cape	Evil
Batman	180	T	T	0
Robin	179	T	T	0
Alfred	175	F	F	0
Penguin	179	F	F	1
Catwoman	165	T	F	1
Joker	180	F	F	1

Example: is your roommate good or evil? (Your turn)

$$H(Y|[\![Height > 165]\!]) = \frac{5}{6}\big(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\big) + \frac{1}{6}(0)$$

$$H(Y|[\![Height > 175]\!]) = \frac{2}{3}(1) + \frac{1}{3}(1)$$

$$H(Y|[\![Height > 179]\!]) = \frac{1}{3}(1) + \frac{2}{3}(1)$$

$$H(Y|Cape) = \frac{1}{3}(0) + \frac{2}{3}\big(-\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4}\big)$$

$$H(Y|Mask) = \frac{1}{2}\big(-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3}\big) + \frac{1}{2}\big(-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3}\big)$$

	Height (cm)	Mask	Cape	Evil
Batman	180	T	T	0
Robin	179	T	T	0
Alfred	175	F	F	0
Penguin	179	F	F	1
Catwoman	165	T	F	1
Joker	180	F	F	1

TL;DPA: We walked through the decision tree algorithm, how to make splits and the relevant stopping criteria

A problem with decision trees

$S_n^{(1)} =$

x_1	x_2	x_3	x_4	x_5	y
1	1	0	0	0	1
1	1	0	0	0	0
1	0	1	1	0	1
1	1	0	0	1	1
0	0	1	0	1	0
1	1	1	0	0	1
0	0	0	1	1	1
1	1	0	0	0	1
1	1	0	1	1	0
0	0	0	0	1	1
1	0	1	1	0	0
1	1	0	0	1	1
1	0	1	1	0	1
1	1	0	1	1	0
0	0	0	1	0	1
0	1	0	0	0	0
0	1	1	0	1	1
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	0	0	1

$S_n^{(2)} =$

x_1	x_2	x_3	x_4	x_5	y
1	1	0	0	0	1
1	1	0	0	0	0
1	0	1	1	0	1
1	1	0	0	1	1
0	0	1	0	1	0
1	1	1	0	0	0
0	0	0	1	1	1
1	1	0	0	0	1
1	1	0	1	1	0
0	0	0	0	1	1
1	0	1	1	0	0
1	1	0	0	1	1
1	0	1	1	0	1
1	1	0	1	1	0
0	0	0	1	0	1
0	1	0	0	0	0
0	1	1	0	1	1
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	0	0	1

A problem with decision trees

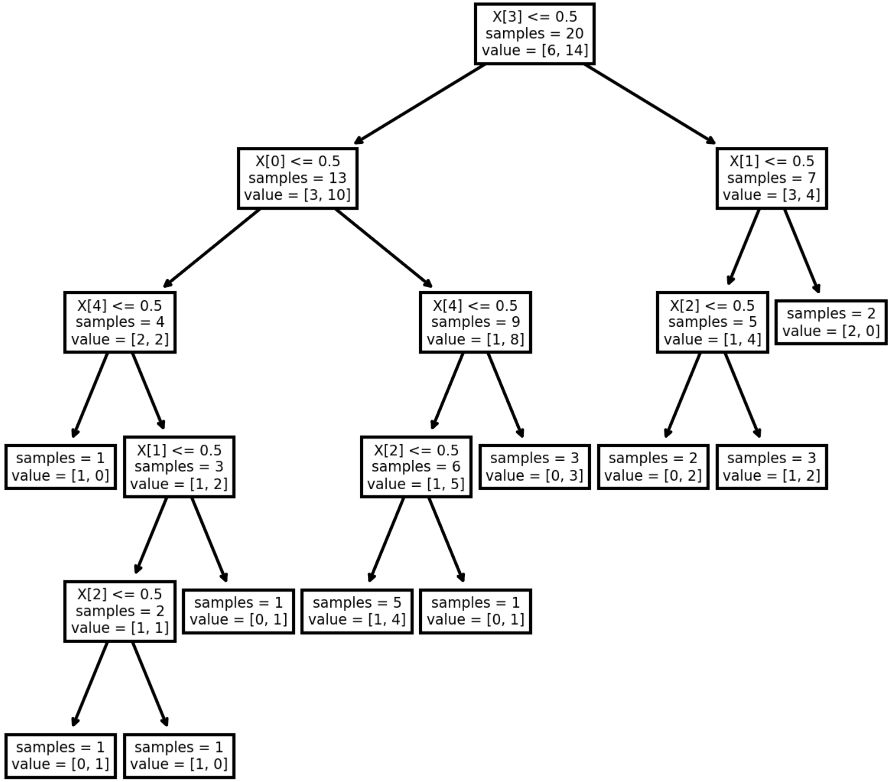
$S_n^{(1)} =$

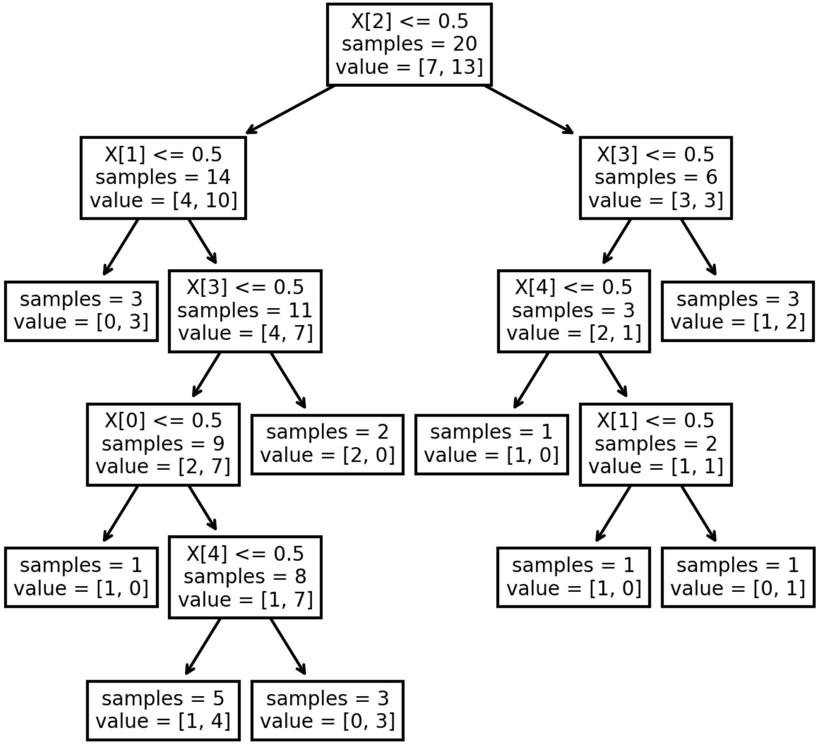
x_1	x_2	x_3	x_4	x_5	y
1	1	0	0	0	1
1	1	0	0	0	0
1	0	1	1	0	1
1	1	0	0	1	1
0	0	1	0	1	0
1	1	1	0	0	$\boxed{1}$
0	0	0	1	1	1
1	1	0	0	0	1
1	1	0	1	1	0
0	0	0	0	1	1
1	0	1	1	0	0
1	1	0	0	1	1
1	0	1	1	0	1
1	1	0	1	1	0
0	0	0	1	0	1
0	1	0	0	0	0
0	1	1	0	1	1
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	0	0	1

$S_n^{(2)} =$

x_1	x_2	x_3	x_4	x_5	y
1	1	0	0	0	1
1	1	0	0	0	0
1	0	1	1	0	1
1	1	0	0	1	1
0	0	1	0	1	0
1	1	1	0	0	$\boxed{0}$
0	0	0	1	1	1
1	1	0	0	0	1
1	1	0	1	1	0
0	0	0	0	1	1
1	0	1	1	0	0
1	1	0	0	1	1
1	0	1	1	0	1
1	1	0	1	1	0
0	0	0	1	0	1
0	1	0	0	0	0
0	1	1	0	1	1
1	1	0	0	0	1
1	1	0	0	1	1
1	1	0	0	0	1

A problem with trees





Bias variance trade-off and generalization error

- Remember: our goal is to minimize generalization error
- What are sources of generalization error?
  1. Bias (structural error)
  2. Variance (estimation error)
  3. Irreducible noise

Bias variance trade-off and generalization error

- Remember: our goal is to minimize generalization error
- What are sources of generalization error?
    1. Bias (structural error)
    2. Variance (estimation error)
    3. Irreducible noise

- Two ways to understand the bias variance trade-off:
    1. An analogy
    2. Using images

Bias variance trade-off: an analogy

A modified game of Marco Polo:

- **Objective**: give an estimate of where your friend, Jane, is
- **Setting:** Multiple rounds
- **Rules:**
  - Jane can't change location
  - Player from round i can't give information to player from round j
- **Data:** your friend's voice
- **Hypothesis space**: the room
- **Noise:** loud dish washer, echoes

Bias variance trade-off in images

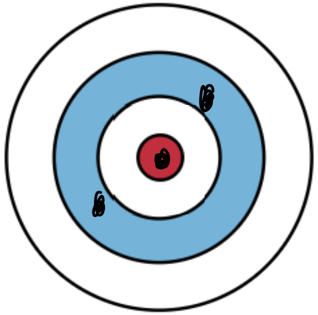


Image from http://scott.fortmann-roe.com/docs/BiasVariance.html

Bias variance trade-off in images

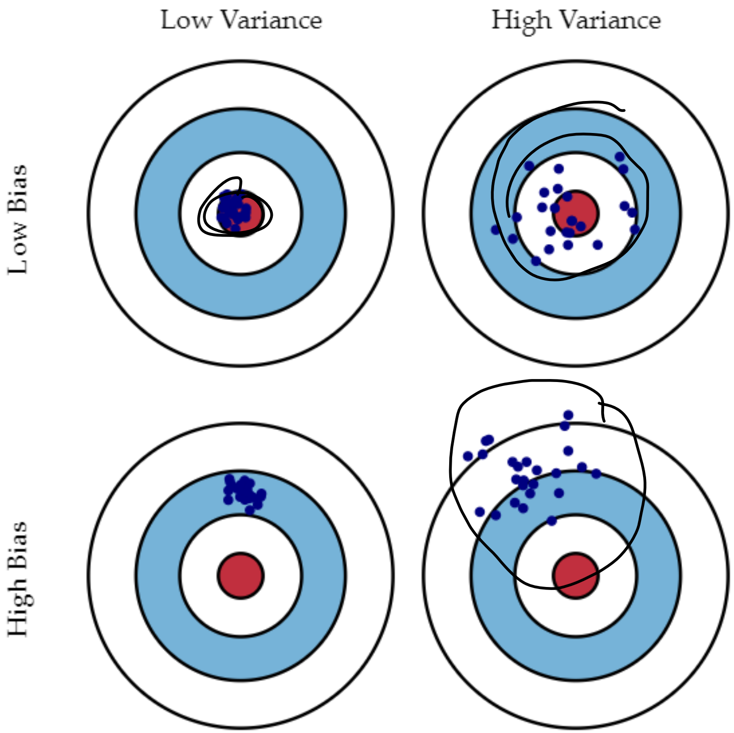


Image from http://scott.fortmann-roe.com/docs/BiasVariance.html

Bias variance trade-off in images

$$R_n(\bar{\theta}) = \frac{1}{n}\sum_i (\bar{\theta}\bar{x}_i + b - y_i)^2 + \lambda\|\theta\|_2^2$$

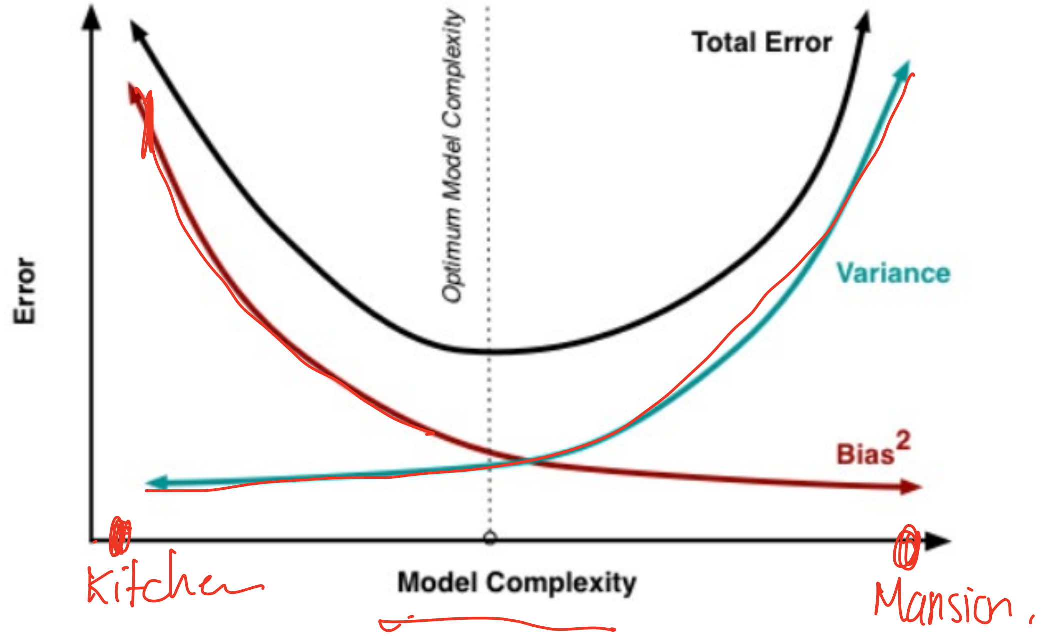


Image from http://scott.fortmann-roe.com/docs/BiasVariance.html

Ensemble methods

- Idea: Create a set of weak/base models whose individual decisions are combined in some way
- Which models? Typically trees & more commonly classification
- Main advantage: Reduces variance without increasing bias
- Describes a set of approaches that differ in training and combination methods
- Two main types of ensembles
  - Bagging
    - "Vanilla" bagging
    - Random Forests
  - Boosting (Adaboost)

TL;DPA:

1. Review of bias variance trade-offs

2. Decision trees are high variance models

3. Ensemble methods can reduce the variance of decision trees without increasing bias (that's magical & unlike what we've seen so far)

Ensemble methods: bagging

n/2

$S_n =$

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

n/2

Ensemble methods: bagging

- Bootstrap sampling: sample n data points with replacement. Do that B times

$S_n^{(1)}$

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$f^{(1)}(\bar{X})$

$S_n^{(2)}$

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D14	Rain	Mild	High	Strong	No

$f^{(2)}(\bar{X})$

...

$S_n^{(B)}$

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D14	Rain	Mild	High	Strong	No

$f^{(B)}(\bar{X})$

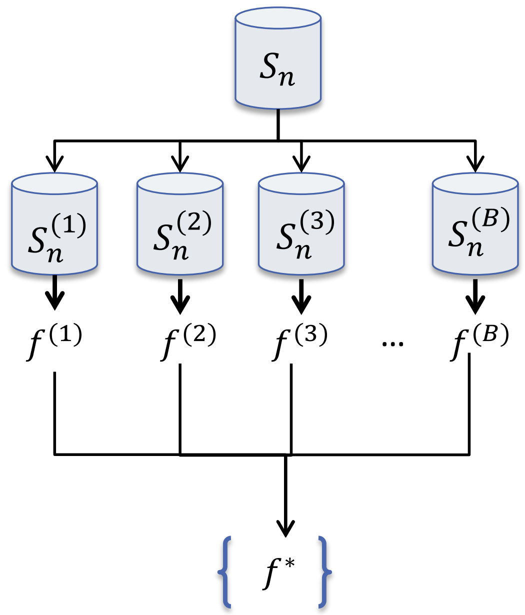
Ensemble methods: bagging

Bagging = **B**ootstrap **agg**regat**ing**

Algorithm:

1. Sample n points B times with replacement
2. Build B decision trees using each of the B bootstrap replicates
3. Aggregate their prediction

$$f(\bar{x}) = \underbrace{\arg\max_{y} \sum_{b=1}^{B} [\![f^{(b)} = y]\!]}$$

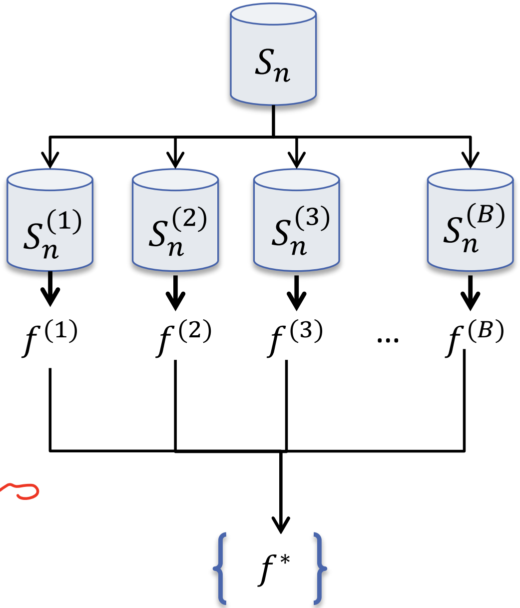


Ensemble methods: bagging

- Bagging = **B**ootstrap **agg**regat**ing**

Assumptions:

- Each decision tree has a misclassification rate better than 50%
- Classifiers are independent create predictions that are uncorrelated.



If assumptions are satisfied:
As B → ∞, misclassification rate → 0

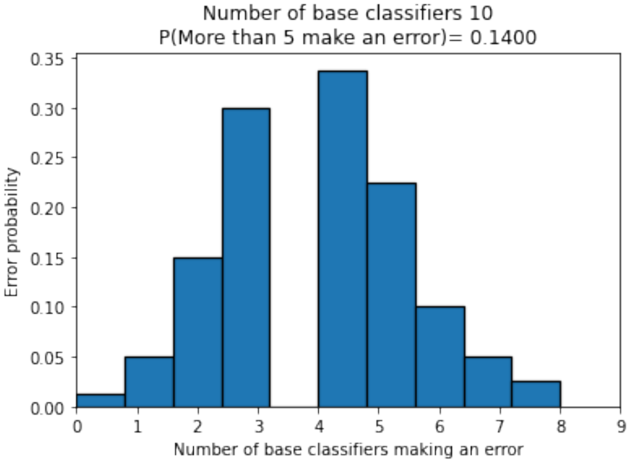
Why does bagging work?

Demo

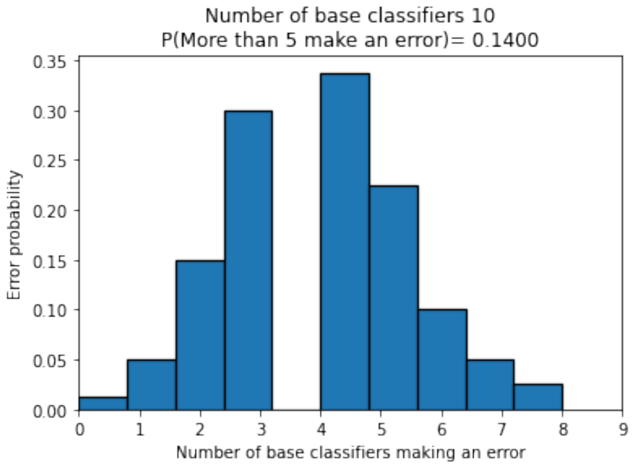
..that you can run!

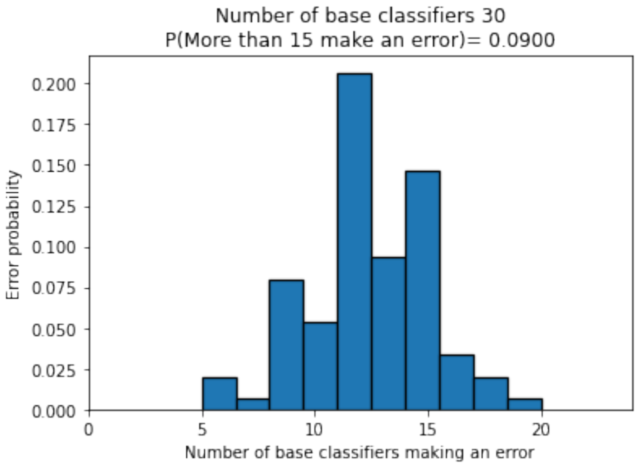
https://colab.research.google.com/drive/1xbrNNEmd9URcP6b1pFtF-iirfmH4KJXn?usp=sharing

Bagged classifiers in action
Base classifier error rate $= 0.4$

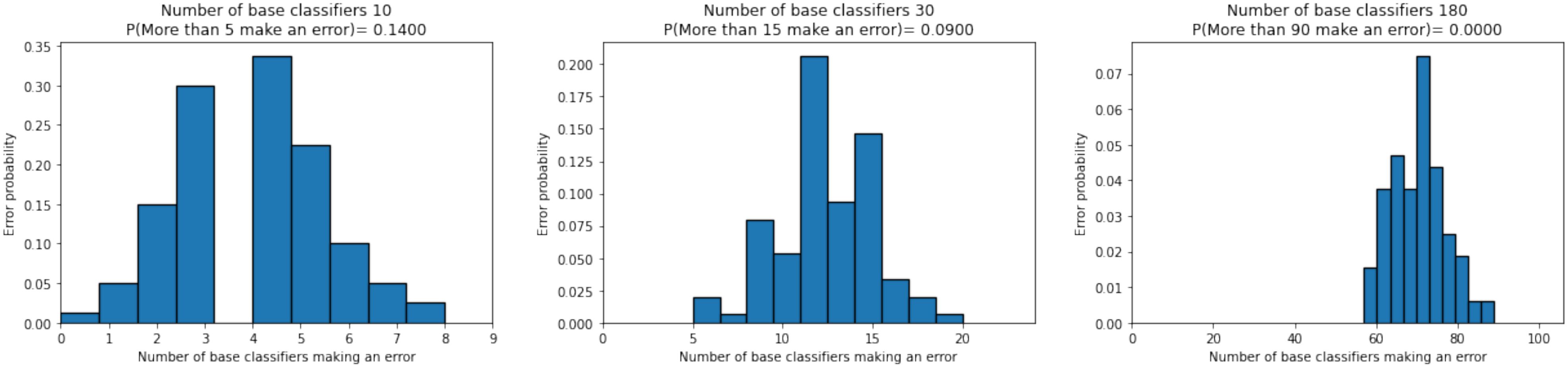


Bagged classifiers in action
Base classifier error rate = 0.4

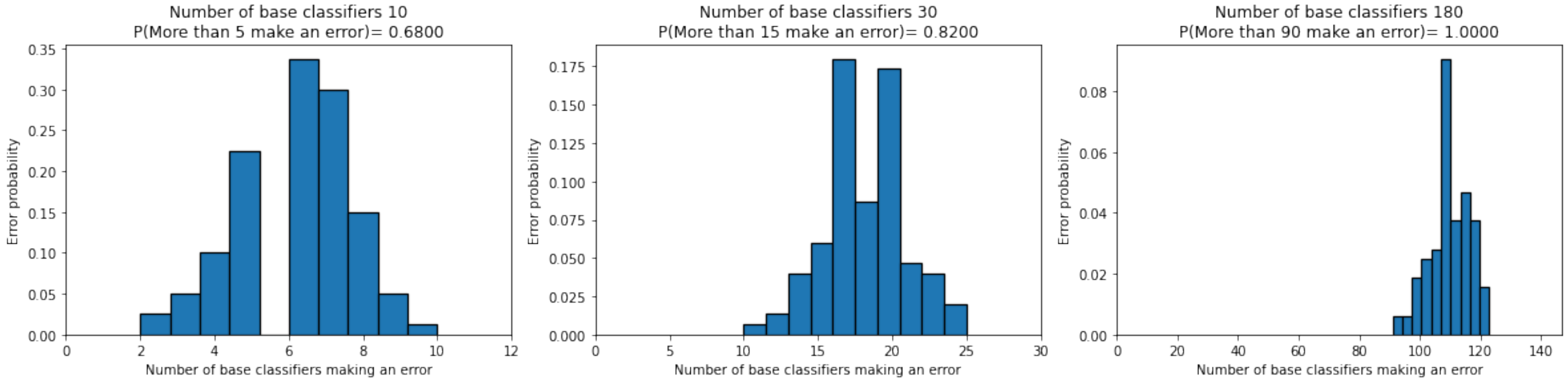




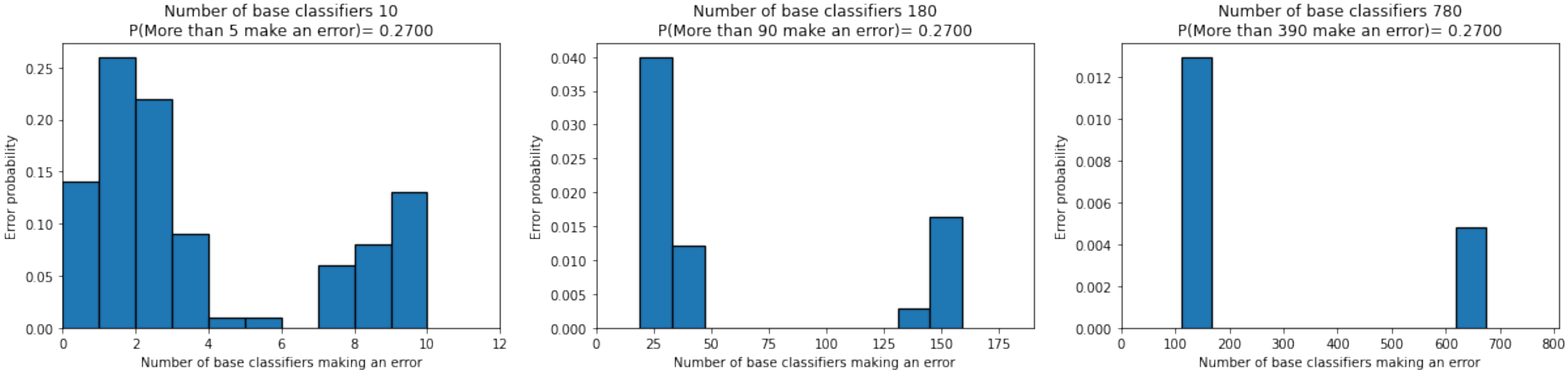
Bagged classifiers in action
Base classifier error rate = 0.4



Bagged classifiers in action
Base classifier error rate = 0.6



Bagged classifiers in action
Correlated base classifier with error rate = 0.3



Ensemble methods: bagging

- Bootstrap sampling: sample n data points with replacement. Do that K times

$$S_n^{(1)}$$

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$$S_n^{(2)}$$

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D14	Rain	Mild	High	Strong	No

...

$$S_n^{(k)}$$

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D14	Rain	Mild	High	Strong	No

Ensemble methods: Random Forests

Aim: decorrelate predictions from different classifiers

1. Bootstrap sampling
2. At each node, best split is chosen from random subset of $m < d$ features

TL;DPA:
We looked at 2 kinds of bagged models:

1. Vanilla bagging: bootstrap sampling + fitting a different tree on each sample

2. Random forests: Vanilla bagging + additional shenanigans

$$\operatorname{argmax}\Big(\underbrace{\sum_i (1-y_i)[[x_i \in R_m]]}_{}, \underbrace{\sum_i y_i [[x_i \in R_m]]}_{} \Big)$$

$$u_m = \frac{1}{n_m} \sum_i y_i [[x_i \in R_m]] > 0.5$$

Additional slides

Regularization: Weakest link pruning

- By Breiman et. al., 1984
- Same as cost-complexity pruning
- Algorithm:
  - Start with the full tree, T_0
  - For each subtree:
    - Replace subtree with a single node to obtain new tree T_k
    - Compute

$$\alpha_k = \frac{\mathrm{Err}(T_k) - \mathrm{Err}(T_0)}{|T_0| - |T_k|}$$

  - Pick T_k with the minimum α_k. In case of ties, pick T_k that prunes the least leaves (i.e., has the largest $|T_k|$)

Regularization: Weakest link pruning

Algorithm:

- Start with the full tree, T_0
- For each subtree:
  - Replace subtree with a single node to obtain new tree T_k
  - Compute
$$\alpha_k = \frac{\mathrm{Err}(T_k) - \mathrm{Err}(T_0)}{|T_0| - |T_k|}$$
- Pick T_k with the minimum α_k. In case of ties, pick T_k that prunes the least leaves (i.e., has the largest $|T_k|$)