Classification Trees

Prof Wells

STA 295: Stat Learning

April 18th, 2024

Outline

- Introduction to Decision Trees
- Discuss Theory and Algorithm for Decision Trees
- Describe the Pruning Algorithm as means of improving RMSE
- Implement Decision Trees in R

Section 1

Classification Trees

Classification Trees

Classification trees are very similar to regression trees, except the terminal nodes predict levels of a categorical variable, rather than values of a quantitative variable

Classification Trees

Classification trees are very similar to regression trees, except the terminal nodes predict levels of a categorical variable, rather than values of a quantitative variable

- To *grow* a classification tree, we need to make cuts based on a metric other than RSS (why?)

Classification Trees

Classification trees are very similar to regression trees, except the terminal nodes predict levels of a categorical variable, rather than values of a quantitative variable

- To *grow* a classification tree, we need to make cuts based on a metric other than RSS (why?)
- For each split candidate, we average the value of the metric on the two proposed subregions, and select the split that minimizes the average value of the metric.

Classification Trees

Classification trees are very similar to regression trees, except the terminal nodes predict levels of a categorical variable, rather than values of a quantitative variable

- To *grow* a classification tree, we need to make cuts based on a metric other than RSS (why?)
- For each split candidate, we average the value of the metric on the two proposed subregions, and select the split that minimizes the average value of the metric.
- The most natural choice is to use *Classification Error Rate E* (i.e. proportion of obs. in region not in most common class)

$$E = 1 - \max_k(p_k) \quad \text{where } \hat{p}_k = \text{prop. obs.in class k}$$

Classification Trees

Classification trees are very similar to regression trees, except the terminal nodes predict levels of a categorical variable, rather than values of a quantitative variable

- To *grow* a classification tree, we need to make cuts based on a metric other than RSS (why?)
- For each split candidate, we average the value of the metric on the two proposed subregions, and select the split that minimizes the average value of the metric.
- The most natural choice is to use *Classification Error Rate E* (i.e. proportion of obs. in region not in most common class)

$$E = 1 - \mathrm{max}_k(p_k) \quad \text{where } \hat{p}_k = \text{prop. obs.in class k}$$

  - This is the proportion of observations misclassified, if we were to always classify using the most frequent class

Classification Error Splits

- Suppose we have 100 observations in 3 classes A, B and C with the following counts:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Classification Error Splits

- Suppose we have 100 observations in 3 classes A, B and C with the following counts:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

$$E = 1 - \max p_k = 1 - 0.5 = 0.5$$

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$$E = 0.25$$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$$E = 0.5$$

Classification Error Splits

- Suppose we have 100 observations in 3 classes A, B and C with the following counts:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

$$E = 1 - \max p_k = 1 - 0.5 = 0.5$$

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$$E = 0.25$$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$$E = 0.5$$

- The overall error on the split is the weighted average of error on each region:

Classification Error Splits

- Suppose we have 100 observations in 3 classes A, B and C with the following counts:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

$$E = 1 - \max p_k = 1 - 0.5 = 0.5$$

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$$E = 0.25$$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$$E = 0.5$$

- The overall error on the split is the weighted average of error on each region:

$$E_{\text{avg}} = 0.6 \cdot 0.25 + 0.4 \cdot 0.5 = 0.35$$

Classification Error Splits

- Suppose we have 100 observations in 3 classes A, B and C with the following counts:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

$$E = 1 - \max p_k = 1 - 0.5 = 0.5$$

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$$E = 0.25$$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$$E = 0.5$$

- The overall error on the split is the weighted average of error on each region:

$$E_{\text{avg}} = 0.6 \cdot 0.25 + 0.4 \cdot 0.5 = 0.35$$

- Unfortunately, E tends to be too insensitive to increases in node purity (i.e. a proposed cut can increase node purity, while E remains constant)

Alternative Metrics: Gini Index

- The *Gini index* G for a region with a total of K classes:

$$G = \sum_{i=k}^{K} \hat{p}_k(1 - \hat{p}_k) \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

Alternative Metrics: Gini Index

- The *Gini index* G for a region with a total of K classes:

$$G = \sum_{i=k}^{K} \hat{p}_k(1 - \hat{p}_k) \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - G measures the probability that a random element would be incorrectly labeled if it was labeled randomly according proportions for labels in the region.

Alternative Metrics: Gini Index

- The *Gini index* G for a region with a total of K classes:

$$G = \sum_{i=k}^{K} \hat{p}_k(1 - \hat{p}_k) \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - G measures the probability that a random element would be incorrectly labeled if it was labeled randomly according proportions for labels in the region.
  - G is always between 0 and $1 - \frac{1}{k}$

Alternative Metrics: Gini Index

- The *Gini index G* for a region with a total of K classes:

$$G = \sum_{i=k}^{K} \hat{p}_k(1 - \hat{p}_k) \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - G measures the probability that a random element would be incorrectly labeled if it was labeled randomly according proportions for labels in the region.
  - G is always between 0 and $1 - \frac{1}{k}$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)

Alternative Metrics: Gini Index

- The *Gini index* G for a region with a total of K classes:

$$G = \sum_{i=k}^{K} \hat{p}_k(1 - \hat{p}_k) \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - G measures the probability that a random element would be incorrectly labeled if it was labeled randomly according proportions for labels in the region.
  - G is always between 0 and $1 - \frac{1}{k}$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)
  - While it is maximized when all $\hat{p}_k$ are equal (high impurity)

Alternative Metrics: Gini Index

- The *Gini index* G for a region with a total of K classes:

$$G = \sum_{i=k}^{K} \hat{p}_k(1 - \hat{p}_k) \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - G measures the probability that a random element would be incorrectly labeled if it was labeled randomly according proportions for labels in the region.
  - G is always between 0 and $1 - \frac{1}{k}$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)
  - While it is maximized when all $\hat{p}_k$ are equal (high impurity)

- Suppose we have 100 observations in 3 classes A, B and C with the following counts:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Alternative Metrics: Gini Index

- The *Gini index* G for a region with a total of K classes:

$$G = \sum_{i=k}^{K} \hat{p}_k(1 - \hat{p}_k) \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - G measures the probability that a random element would be incorrectly labeled if it was labeled randomly according proportions for labels in the region.
  - G is always between 0 and $1 - \frac{1}{k}$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)
  - While it is maximized when all $\hat{p}_k$ are equal (high impurity)

- Suppose we have 100 observations in 3 classes A, B and C with the following counts:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

$$G = \sum_{k=1}^{3} \hat{p}_k(1 - \hat{p}_k) = 0.5(1 - 0.5) + 0.3(1 - 0.3) + 0.2(1 - .2) = 0.62$$

Alternative Metrics: Gini Index

- The *Gini index* G for a region with a total of K classes:

$$G = \sum_{i=k}^{K} \hat{p}_k(1 - \hat{p}_k) \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - G measures the probability that a random element would be incorrectly labeled if it was labeled randomly according proportions for labels in the region.
  - G is always between 0 and $1 - \frac{1}{k}$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)
  - While it is maximized when all $\hat{p}_k$ are equal (high impurity)

- Suppose we have 100 observations in 3 classes A, B and C with the following counts:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

$$G = \sum_{k=1}^{3} \hat{p}_k(1 - \hat{p}_k) = 0.5(1 - 0.5) + 0.3(1 - 0.3) + 0.2(1 - .2) = 0.62$$

- As $G = 0.62$ is close to max of $1 - \frac{1}{3} = 0.67$, then region has high impurity.

Gini Index Splits

- Consider the same 100 observations on 3 classes with $G = 0.62$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Gini Index Splits

- Consider the same 100 observations on 3 classes with $G = 0.62$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$G = 0.4822$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$G = 0.59375$

Gini Index Splits

- Consider the same 100 observations on 3 classes with $G = 0.62$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$G = 0.4822$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$G = 0.59375$

- Overall error rate after split:

Gini Index Splits

- Consider the same 100 observations on 3 classes with $G = 0.62$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$G = 0.4822$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$G = 0.59375$

- Overall error rate after split:

$$G_{\text{avg}} = 0.6 \cdot 0.4822 + 0.4 \cdot 0.59375 = 0.52682$$

Gini Index Splits

- Consider the same 100 observations on 3 classes with $G = 0.62$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$$G = 0.4822$$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$$G = 0.59375$$

- Overall error rate after split:

$$G_{\text{avg}} = 0.6 \cdot 0.4822 + 0.4 \cdot 0.59375 = 0.52682$$

  - Since the new average Gini index is less than the Gini index for the original region, the proposed cut reduces node impurity.

Gini Index Splits

- Consider the same 100 observations on 3 classes with $G = 0.62$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$$G = 0.4822$$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$$G = 0.59375$$

- Overall error rate after split:

$$G_{\text{avg}} = 0.6 \cdot 0.4822 + 0.4 \cdot 0.59375 = 0.52682$$

  - Since the new average Gini index is less than the Gini index for the original region, the proposed cut reduces node impurity.

- Is it the greatest increase in node purity? It depends on the relationship between predictors and response (and therefore, what cuts are allowed)

Alternative Metrics: Entropy

- The *information* or *entropy* D for a region with a total of K classes:

$$D = -\sum_{k=1}^{K} \hat{p}_k \log_2 \hat{p}_k \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

Alternative Metrics: Entropy

- The *information* or *entropy* D for a region with a total of K classes:

$$D = -\sum_{k=1}^{K} \hat{p}_k \log_2 \hat{p}_k \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - D measures the average amount of information gained by learning the true class of an observation, given you already know the probabilities that it belongs to each class

Alternative Metrics: Entropy

- The *information* or *entropy* D for a region with a total of K classes:

$$D = -\sum_{k=1}^{K} \hat{p}_k \log_2 \hat{p}_k \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - D measures the average amount of information gained by learning the true class of an observation, given you already know the probabilities that it belongs to each class
  - D is always between 0 and $\log_2 k$

Alternative Metrics: Entropy

- The *information* or *entropy* D for a region with a total of K classes:

$$D = -\sum_{k=1}^{K} \hat{p}_k \log_2 \hat{p}_k \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - D measures the average amount of information gained by learning the true class of an observation, given you already know the probabilities that it belongs to each class
  - D is always between 0 and $\log_2 k$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)

Alternative Metrics: Entropy

- The *information* or *entropy* D for a region with a total of K classes:

$$D = -\sum_{k=1}^{K} \hat{p}_k \log_2 \hat{p}_k \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - D measures the average amount of information gained by learning the true class of an observation, given you already know the probabilities that it belongs to each class
  - D is always between 0 and $\log_2 k$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)
  - While it is maximized when all $\hat{p}_k$ are equal (high impurity)

Alternative Metrics: Entropy

- The *information* or *entropy* D for a region with a total of K classes:

$$D = -\sum_{k=1}^{K} \hat{p}_k \log_2 \hat{p}_k \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - D measures the average amount of information gained by learning the true class of an observation, given you already know the probabilities that it belongs to each class
  - D is always between 0 and $\log_2 k$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)
  - While it is maximized when all $\hat{p}_k$ are equal (high impurity)

- Suppose we have the same 100 observations in 3 classes

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Alternative Metrics: Entropy

- The *information* or *entropy* D for a region with a total of K classes:

$$D = -\sum_{k=1}^{K} \hat{p}_k \log_2 \hat{p}_k \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - D measures the average amount of information gained by learning the true class of an observation, given you already know the probabilities that it belongs to each class
  - D is always between 0 and $\log_2 k$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)
  - While it is maximized when all $\hat{p}_k$ are equal (high impurity)

- Suppose we have the same 100 observations in 3 classes

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

$$D = -\sum_{k=1}^{3} \hat{p}_k \log_2 \hat{p}_k = -0.5(-1) + 0.3(-1.7) + 0.2(-2.3) = 1.49$$

Alternative Metrics: Entropy

- The *information* or *entropy* D for a region with a total of K classes:

$$D = -\sum_{k=1}^{K} \hat{p}_k \log_2 \hat{p}_k \quad \text{where } \hat{p}_k = \text{prop. obs. in class k}$$

  - D measures the average amount of information gained by learning the true class of an observation, given you already know the probabilities that it belongs to each class
  - D is always between 0 and $\log_2 k$
  - It is small when all $\hat{p}_k$ are close to 0 or 1 (high purity)
  - While it is maximized when all $\hat{p}_k$ are equal (high impurity)

- Suppose we have the same 100 observations in 3 classes

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

$$D = -\sum_{k=1}^{3} \hat{p}_k \log_2 \hat{p}_k = -0.5(-1) + 0.3(-1.7) + 0.2(-2.3) = 1.49$$

- As $D = 1.49$ is close to max of $\log_2 3 = 1.58$, then region has high impurity.

Entropy Splits

- Consider the same 100 observations on 3 classes with $D = 1.49$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Entropy Splits

- Consider the same 100 observations on 3 classes with $D = 1.49$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$D = 0.99$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$D = 1.41$

Entropy Splits

- Consider the same 100 observations on 3 classes with $D = 1.49$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$D = 0.99$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$D = 1.41$

- Overall error after split:

Entropy Splits

- Consider the same 100 observations on 3 classes with $D = 1.49$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$D = 0.99$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$D = 1.41$

- Overall error after split:

$$D_{\text{avg}} = 0.6 \cdot 0.99 + 0.4 \cdot 1.41 = 1.158$$

Entropy Splits

- Consider the same 100 observations on 3 classes with $D = 1.49$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$D = 0.99$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$D = 1.41$

- Overall error after split:

$$D_{\mathrm{avg}} = 0.6 \cdot 0.99 + 0.4 \cdot 1.41 = 1.158$$

  - The new entropy is less than the old one, so the proposed split decreases impurity

Entropy Splits

- Consider the same 100 observations on 3 classes with $D = 1.49$:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

- Suppose we cut our region into two subregions with the following counts

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

$D = 0.99$

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

$D = 1.41$

- Overall error after split:

$$D_{\text{avg}} = 0.6 \cdot 0.99 + 0.4 \cdot 1.41 = 1.158$$

  - The new entropy is less than the old one, so the proposed split decreases impurity

- Is it the greatest increase in node purity? It depends on the relationship between predictors and response (and therefore, what cuts are allowed)

Comparison of Metrics

- Consider the same 100 observations on 3 classes, which are to be cut into two regions:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

Comparison of Metrics

- Consider the same 100 observations on 3 classes, which are to be cut into two regions:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

- Comparing the values of the metrics before and after the split:

Comparison of Metrics

- Consider the same 100 observations on 3 classes, which are to be cut into two regions:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

- Comparing the values of the metrics before and after the split:

Metric	Large Region	Sub-region 1	Sub-region 2	Average in Sub-regions
Error	0.5	0.25	0.5	.35
Gini	0.62	0.48	0.59	0.52
Entropy	1.49	.99	1.41	1.16

Comparison of Metrics

- Consider the same 100 observations on 3 classes, which are to be cut into two regions:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

- Comparing the values of the metrics before and after the split:

Metric	Large Region	Sub-region 1	Sub-region 2	Average in Sub-regions
Error	0.5	0.25	0.5	.35
Gini	0.62	0.48	0.59	0.52
Entropy	1.49	.99	1.41	1.16

- Metrics differ in how much better Region 2 is than the larger region. They also differ in how much better Region 1 is than region 2.

Comparison of Metrics

- Consider the same 100 observations on 3 classes, which are to be cut into two regions:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

- Comparing the values of the metrics before and after the split:

Metric	Large Region	Sub-region 1	Sub-region 2	Average in Sub-regions
Error	0.5	0.25	0.5	.35
Gini	0.62	0.48	0.59	0.52
Entropy	1.49	.99	1.41	1.16

- Metrics differ in how much better Region 2 is than the larger region. They also differ in how much better Region 1 is than region 2.
  - Therefore, each metric will tend to favor making different cuts.

Comparison of Metrics

- Consider the same 100 observations on 3 classes, which are to be cut into two regions:

Class	A	B	C
n	50	30	20
$\hat{p}$	0.5	0.3	0.2

Region 1

Class	A	B	C	total
n	45	10	5	60
$\hat{p}$	0.75	0.67	0.08	1.0

Region 2

Class	A	B	C	total
n	5	20	15	40
$\hat{p}$	0.125	0.5	0.375	1.0

- Comparing the values of the metrics before and after the split:

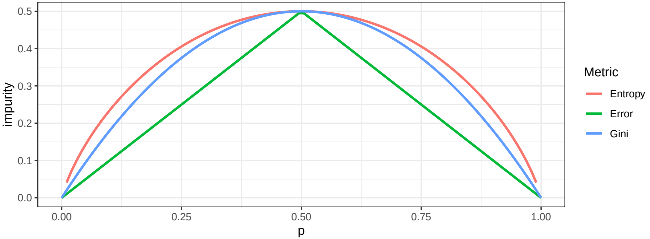
Metric	Large Region	Sub-region 1	Sub-region 2	Average in Sub-regions
Error	0.5	0.25	0.5	.35
Gini	0.62	0.48	0.59	0.52
Entropy	1.49	.99	1.41	1.16

- Metrics differ in how much better Region 2 is than the larger region. They also differ in how much better Region 1 is than region 2.
  - Therefore, each metric will tend to favor making different cuts.

- Overall, Gini index and Entropy tend to make more accurate models that Error rate. But neither Gini nor Entropy is consistently better than the other.

Graphical Comparison of Metrics (Optional)

- The following plot shows the size of the metric as a function of the proportion of observations in a single class, for binary class problems. Values of p close to 0 or 1 indicate high class purity.



- The closer the curve is to the upper-left and upper-right corners, the more sensitive the metric is to class purity.

Dealing with Categorical Variables

Both regression and classification trees can easily hand either quantitative or binary categorical variables.

Dealing with Categorical Variables

Both regression and classification trees can easily hand either quantitative or binary categorical variables.

- But with some modification, trees can also be used with multi-level categorical variables.

Dealing with Categorical Variables

Both regression and classification trees can easily hand either quantitative or binary categorical variables.

- But with some modification, trees can also be used with multi-level categorical variables.
- To do so, we recode all multilevel categorical variables as a sequence of dummy binary variables. Then proceed as usual.

Dealing with Categorical Variables

Both regression and classification trees can easily hand either quantitative or binary categorical variables.

- But with some modification, trees can also be used with multi-level categorical variables.
- To do so, we recode all multilevel categorical variables as a sequence of dummy binary variables. Then proceed as usual.
- But this conversion has a significant downside! The algorithm is biased toward making early splits on categorical variables with many levels.

Dealing with Categorical Variables

Both regression and classification trees can easily hand either quantitative or binary categorical variables.

- But with some modification, trees can also be used with multi-level categorical variables.
- To do so, we recode all multilevel categorical variables as a sequence of dummy binary variables. Then proceed as usual.
- But this conversion has a significant downside! The algorithm is biased toward making early splits on categorical variables with many levels.
    - Since trees are already prone to high variance, this additional bias can lead to unwanted increases in MSE.

Dealing with Categorical Variables

Both regression and classification trees can easily hand either quantitative or binary categorical variables.

- But with some modification, trees can also be used with multi-level categorical variables.
- To do so, we recode all multilevel categorical variables as a sequence of dummy binary variables. Then proceed as usual.
- But this conversion has a significant downside! The algorithm is biased toward making early splits on categorical variables with many levels.
  - Since trees are already prone to high variance, this additional bias can lead to unwanted increases in MSE.
- The “simple” fix is to lump together levels before building a tree, using domain knowledge

Dealing with Categorical Variables

Both regression and classification trees can easily hand either quantitative or binary categorical variables.

- But with some modification, trees can also be used with multi-level categorical variables.
- To do so, we recode all multilevel categorical variables as a sequence of dummy binary variables. Then proceed as usual.
- But this conversion has a significant downside! The algorithm is biased toward making early splits on categorical variables with many levels.
  - Since trees are already prone to high variance, this additional bias can lead to unwanted increases in MSE.
- The "simple" fix is to lump together levels before building a tree, using domain knowledge
- An alternative is to allow the model algorithm to lump together values as necessary at each node (order levels in increasing frequency, then make appropriate cut)

Dealing with Categorical Variables

Both regression and classification trees can easily hand either quantitative or binary categorical variables.

- But with some modification, trees can also be used with multi-level categorical variables.
- To do so, we recode all multilevel categorical variables as a sequence of dummy binary variables. Then proceed as usual.
- But this conversion has a significant downside! The algorithm is biased toward making early splits on categorical variables with many levels.
  - Since trees are already prone to high variance, this additional bias can lead to unwanted increases in MSE.
- The "simple" fix is to lump together levels before building a tree, using domain knowledge
- An alternative is to allow the model algorithm to lump together values as necessary at each node (order levels in increasing frequency, then make appropriate cut)
  - But this generally leads to less interpretable models

Section 2

Classification Trees in R

Mushroom Hunting

Mushroom Hunting

Can I eat this?

Mushrooms

- The `mushrooms` data set contains information on `edibility` and 22 other features on 8124 samples of Mushrooms. We'll do a 80-20 training-test split.

Mushrooms

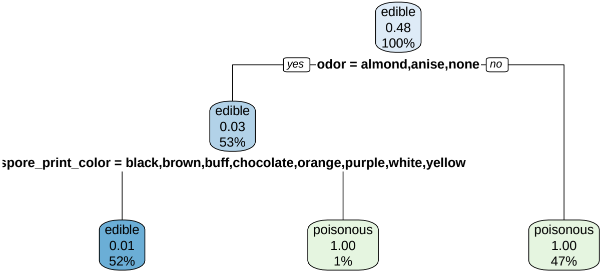
- The `mushrooms` data set contains information on `edibility` and 22 other features on 8124 samples of Mushrooms. We'll do a 80-20 training-test split.

```
## Rows: 6,498
## Columns: 23
## $ edibility                <fct> edible, edible, edible, edible, edible, edibl~
## $ cap_shape                <fct> convex, bell, convex, convex, bell, bell, bel~
## $ cap_surface              <fct> scaly, scaly, scaly, smooth, scaly, smooth, s~
## $ cap_color                <fct> yellow, white, gray, yellow, white, white, ye~
## $ bruises                  <fct> yes, yes, no, yes, yes, yes, yes, yes, yes, y~
## $ odor                     <fct> almond, anise, none, almond, almond, anise, a~
## $ gill_attachement         <fct> free, free, free, free, free, free, free, fre~
## $ gill_spacing             <fct> close, close, crowded, close, close, close, c~
## $ gill_size                <fct> broad, broad, broad, broad, broad, broad, bro~
## $ gill_color               <fct> black, brown, black, brown, gray, brown, gray~
## $ stalk_shape              <fct> enlarging, enlarging, tapering, enlarging, en~
## $ stalk_root               <fct> club, club, equal, club, club, club, club, cl~
## $ stalk_surface_above_ring <fct> smooth, smooth, smooth, smooth, smooth, smoot~
## $ stalk_surface_below_ring <fct> smooth, smooth, smooth, smooth, smooth, smoot~
## $ stalk_color_above_ring   <fct> purple, purple, purple, purple, purple, purpl~
## $ stalk_color_below_ring   <fct> purple, purple, purple, purple, purple, purpl~
## $ veil_type                <fct> partial, partial, partial, partial, partial, ~
## $ veil_color               <fct> white, white, white, white, white, white, whi~
## $ ring_number              <fct> one, one, one, one, one, one, one, one, one, ~
## $ ring_type                <fct> pendant, pendant, evanescent, pendant, pendan~
## $ spore_print_color        <fct> brown, brown, brown, black, black, brown, bla~
## $ population               <fct> numerous, numerous, abundant, numerous, numer~
## $ habitat                  <fct> grasses, meadows, grasses, grasses, meadows, ~
```

Implementing classfication trees in R

As with regression trees, we use the `rpart` package.

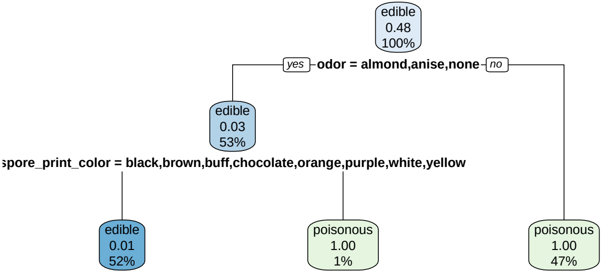
```
library(rpart)
library(rpart.plot)
mushroom_tree<-rpart(edibility ~ ., data = mushrooms_train)
rpart.plot(mushroom_tree)
```



Implementing classfication trees in R

As with regression trees, we use the rpart package.

```
library(rpart)
library(rpart.plot)
mushroom_tree<-rpart(edibility ~ ., data = mushrooms_train)
rpart.plot(mushroom_tree)
```

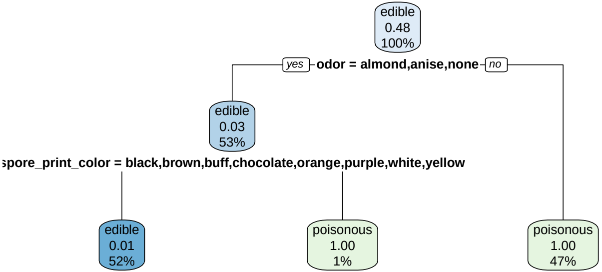


- In each node, the title is the most prominent class, the 2nd number is the proportion of obs. in the node of the target class, and the 3rd number is the overall proportion of observations in the node.

Implementing classfication trees in R

As with regression trees, we use the rpart package.

```
library(rpart)
library(rpart.plot)
mushroom_tree<-rpart(edibility ~ ., data = mushrooms_train)
rpart.plot(mushroom_tree)
```



- In each node, the title is the most prominent class, the 2nd number is the proportion of obs. in the node of the target class, and the 3rd number is the overall proportion of observations in the node.
- The default parameters created data with relatively few terminal nodes. And it seems

Implementing classfication trees in R

By default, `rpart` uses Gini index for splitting

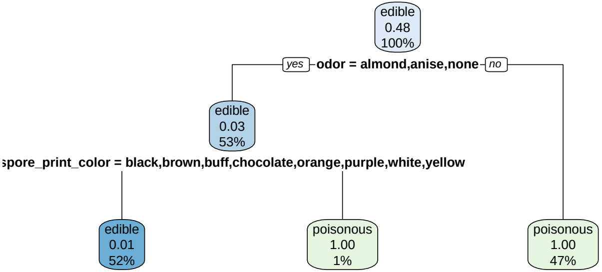
- But we can instead split using Entropy (Information) by adding a `parms` input:

Implementing classfication trees in R

By default, rpart uses Gini index for splitting

- But we can instead split using Entropy (Information) by adding a parms input:

```
library(rpart)
library(rpart.plot)
mushroom_tree<-rpart(edibility ~ ., data = mushrooms_train, parms = list(split = "information"))
rpart.plot(mushroom_tree)
```

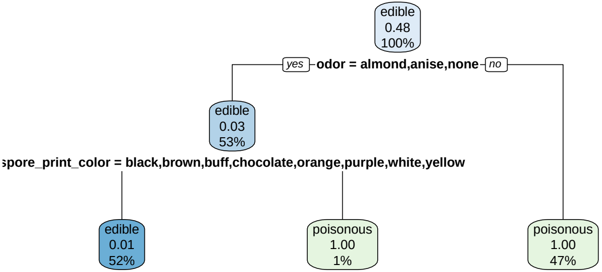


Implementing classfication trees in R

By default, `rpart` uses Gini index for splitting

- But we can instead split using Entropy (Information) by adding a `parms` input:

```
library(rpart)
library(rpart.plot)
mushroom_tree<-rpart(edibility ~ ., data = mushrooms_train, parms = list(split = "information"))
rpart.plot(mushroom_tree)
```



- In this case, we produced the same tree with both metrics, but this doesn't always happen

Model Accuracy

- How well did we do on test data?

Model Accuracy

- How well did we do on test data?

```
library(yardstick)
mushroom_preds <- predict(mushroom_tree, mushrooms_test, type = "class")
mushroom_probs <- predict(mushroom_tree, mushrooms_test, type = "prob")[,"edible"]

results <- data.frame(obs = mushrooms_test$edibility,preds = mushroom_preds,
                      probs = mushroom_probs)

accuracy(results, truth = obs, estimate = preds)
```

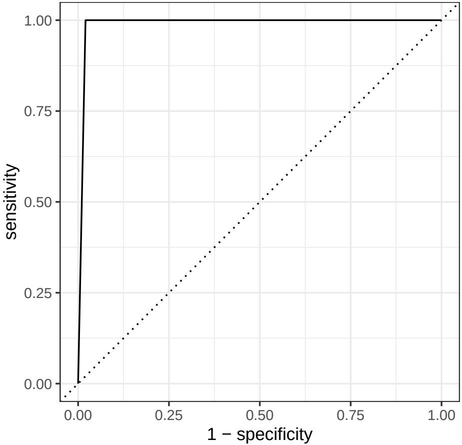
```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.990
```

- Looks like we have fantastic accuracy!

ROC Curve

Look at that ROC curve!

```
roc_curve(results, truth = obs, probs) %>%
autoplot()
```



Confusion Matrix

- Just one more thing to check:

Confusion Matrix

- Just one more thing to check:

```
conf_mat(results, truth = obs, estimate = preds)
```

Confusion Matrix

- Just one more thing to check:

```
conf_mat(results, truth = obs, estimate = preds)
```

```
##            Truth
## Prediction  edible poisonous
##   edible       842        16
##   poisonous      0       768
```

Confusion Matrix

- Just one more thing to check:

```
conf_mat(results, truth = obs, estimate = preds)
```

```
##            Truth
## Prediction  edible poisonous
##   edible       842        16
##   poisonous      0       768
```

Improvements

How can we reduce the **type II error** of our classifier? (rate of poison mushrooms identified as `edible`)

Improvements

How can we reduce the **type II error** of our classifier? (rate of poison mushrooms identified as `edible`)

- *Option 1:* Everything is poisonous!

Improvements

How can we reduce the **type II error** of our classifier? (rate of poison mushrooms identified as `edible`)

- *Option 1:* Everything is poisonous!
  - Downside: No tasty mushrooms :(
- *Option 2:* change classification threshhold

Improvements

How can we reduce the **type II error** of our classifier? (rate of poison mushrooms identified as `edible`)

- *Option 1:* Everything is poisonous!
  - Downside: No tasty mushrooms :(
- *Option 2:* change classification threshhold
  - I.e. classify as poisonous when $P(\text{poisonous}) > 0.01\%$
  - Might drastically reduce accuracy and sensitivity

Improvements

How can we reduce the **type II error** of our classifier? (rate of poison mushrooms identified as `edible`)

- *Option 1:* Everything is poisonous!
  - Downside: No tasty mushrooms :(
- *Option 2:* change classification threshhold
  - I.e. classify as poisonous when $P(\text{poisonous}) > 0.01\%$
  - Might drastically reduce accuracy and sensitivity
- *Option 3:* Incorporate relative loss into Gini index.

Improvements

How can we reduce the **type II error** of our classifier? (rate of poison mushrooms identified as `edible`)

- *Option 1:* Everything is poisonous!
  - Downside: No tasty mushrooms :(
- *Option 2:* change classification threshhold
  - I.e. classify as poisonous when $P(\text{poisonous}) > 0.01\%$
  - Might drastically reduce accuracy and sensitivity
- *Option 3:* Incorporate relative loss into Gini index.

$$G = \sum_i \sum_j L(i,j) p_i p_j$$

Improvements

How can we reduce the **type II error** of our classifier? (rate of poison mushrooms identified as `edible`)

- *Option 1:* Everything is poisonous!
  - Downside: No tasty mushrooms :(
- *Option 2:* change classification threshhold
  - I.e. classify as poisonous when $P(\text{poisonous}) > 0.01\%$
  - Might drastically reduce accuracy and sensitivity
- *Option 3:* Incorporate relative loss into Gini index.

$$G = \sum_i \sum_j L(i,j) p_i p_j$$

- Here, $L(i,j)$ is the loss occurred when predicting level j when the truth is level i.

Additional Parameters

- To incorporate loss, create a penalty matrix and add to the `parms` argument in `rpart`:

```
penalty_matrix <- matrix(c(0,1,20,0), byrow = T, nrow = 2)
penalty_matrix
```

```
##      [,1] [,2]
## [1,]    0    1
## [2,]   20    0
```

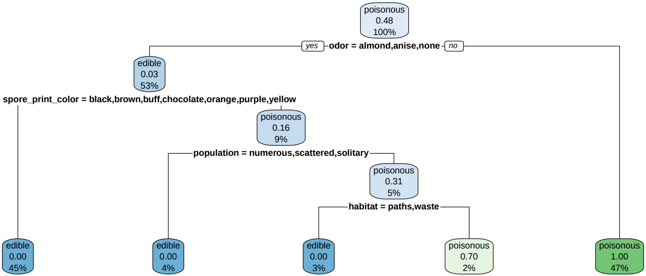
Additional Parameters

- To incorporate loss, create a penalty matrix and add to the parms argument in rpart:

```
penalty_matrix <- matrix(c(0,1,20,0), byrow = T, nrow = 2)
penalty_matrix
```

```
##      [,1] [,2]
## [1,]    0    1
## [2,]   20    0
```

```
mushroom_no_poison <- rpart(edibility ~., data = mushrooms_train,
                            parms = list(loss = penalty_matrix))
rpart.plot(mushroom_no_poison)
```



New Results

- Now how did we do?

New Results

- Now how did we do?

```
results %>% group_by(model) %>% accuracy( truth = obs, estimate = preds)
```

```
## # A tibble: 2 x 4
##   model        .metric  .estimator .estimate
##   <chr>        <chr>    <chr>          <dbl>
## 1 with loss    accuracy binary         0.994
## 2 without loss accuracy binary         0.990
```

New Results

- Now how did we do?

```
results %>% group_by(model) %>% accuracy( truth = obs, estimate = preds)
```

```
## # A tibble: 2 x 4
##   model        .metric  .estimator .estimate
##   <chr>        <chr>    <chr>          <dbl>
## 1 with loss    accuracy binary         0.994
## 2 without loss accuracy binary         0.990
```

```
results %>% filter(model == "with loss") %>% conf_mat(truth = obs, estimate = preds)
```

```
##            Truth
## Prediction  edible poisonous
##   edible       833         0
##   poisonous      9       784
```

New Results

- Now how did we do?

```
results %>% group_by(model) %>% accuracy( truth = obs, estimate = preds)
```

```
## # A tibble: 2 x 4
##   model         .metric  .estimator .estimate
##   <chr>         <chr>    <chr>          <dbl>
## 1 with loss     accuracy binary         0.994
## 2 without loss accuracy binary         0.990
```

```
results %>% filter(model == "with loss") %>% conf_mat(truth = obs, estimate = preds)
```

```
##             Truth
## Prediction  edible poisonous
##   edible       833         0
##   poisonous      9       784
```

- But can we now improve that Type I error?

New Results

- Now how did we do?

```
results %>% group_by(model) %>% accuracy( truth = obs, estimate = preds)
```

```
## # A tibble: 2 x 4
##   model         .metric  .estimator .estimate
##   <chr>         <chr>    <chr>          <dbl>
## 1 with loss     accuracy binary         0.994
## 2 without loss accuracy binary         0.990
```

```
results %>% filter(model == "with loss") %>% conf_mat(truth = obs, estimate = preds)
```

```
##            Truth
## Prediction  edible poisonous
##   edible       833         0
##   poisonous      9       784
```

- But can we now improve that Type I error?
  - To reclaim some of those “poisonous” mushrooms, we’ll need to build a deeper tree.

Deeper Trees

- We can control tree depth by setting the minimum `cp` parameter in `rpart.control`

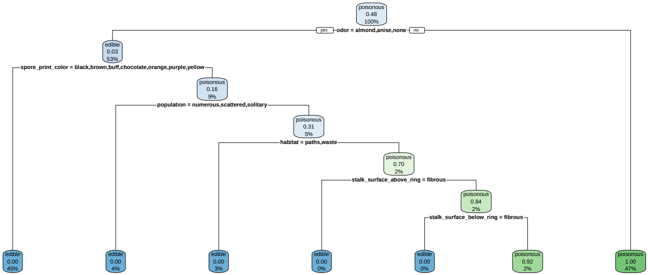
Deeper Trees

- We can control tree depth by setting the minimum `cp` parameter in `rpart.control`
  - Any split that does not decrease overall lack of fit by a factor of `cp` is not attemped.
  - Setting low values of `cp` lead to deeper trees

Deeper Trees

- We can control tree depth by setting the minimum cp parameter in `rpart.control`
  - Any split that does not decrease overall lack of fit by a factor of cp is not attemped.
  - Setting low values of cp lead to deeper trees

```
mushroom_deep <- rpart(edibility ~., data = mushrooms_train,
                       parms = list(loss = penalty_matrix),
                       control = rpart.control(cp = .00001))
rpart.plot(mushroom_deep)
```

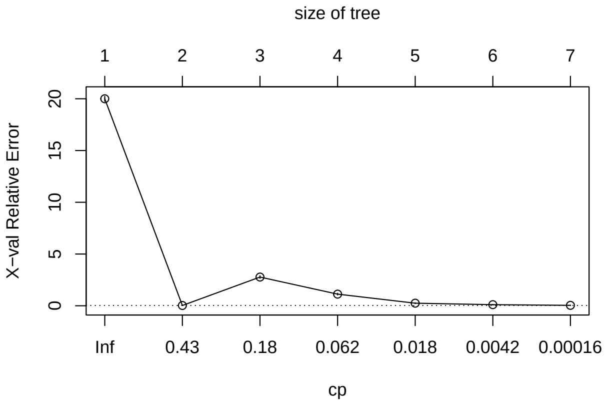


Pruning

- Let's look at cross-validated relative error

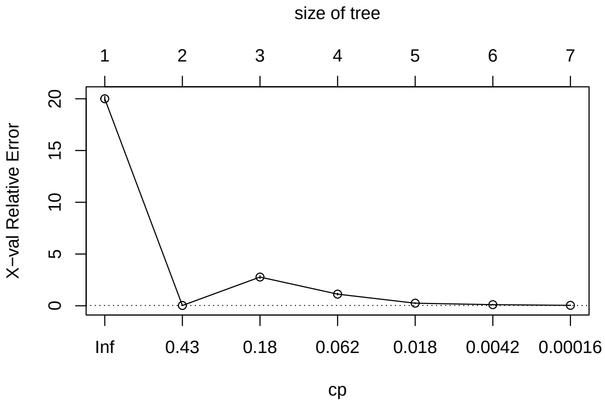
Pruning

- Let's look at cross-validated relative error



Pruning

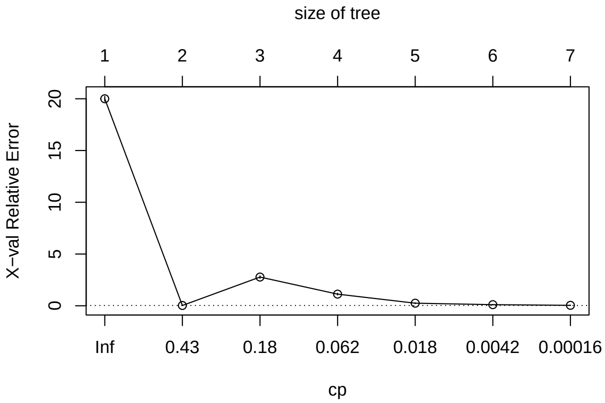
- Let's look at cross-validated relative error



- It's possible we are now overfitting. It may be best to reduce to tree with 6 leaves.

Pruning

- Let's look at cross-validated relative error



- It's possible we are now overfitting. It may be best to reduce to tree with 6 leaves.

```
mushroom_prune <- prune(mushroom_deep, cp = 0.0042)
```

Final Results

- How do our deep and pruned models do?

Final Results

- How do our deep and pruned models do?

```
results %>% group_by(model) %>% accuracy( truth = obs, estimate = preds)
```

```
## # A tibble: 4 x 4
##   model         .metric  .estimator .estimate
##   <chr>         <chr>    <chr>          <dbl>
## 1 deep          accuracy binary         0.998
## 2 pruned        accuracy binary         0.996
## 3 with loss     accuracy binary         0.994
## 4 without loss accuracy binary         0.990
```

```
results %>% filter(model == "deep") %>% conf_mat(truth = obs, estimate = preds)
```

```
##             Truth
## Prediction  edible poisonous
##   edible       838         0
##   poisonous      4       784
```

```
results %>% filter(model == "pruned") %>% conf_mat(truth = obs, estimate = preds)
```

```
##             Truth
## Prediction  edible poisonous
##   edible       835         0
##   poisonous      7       784
```