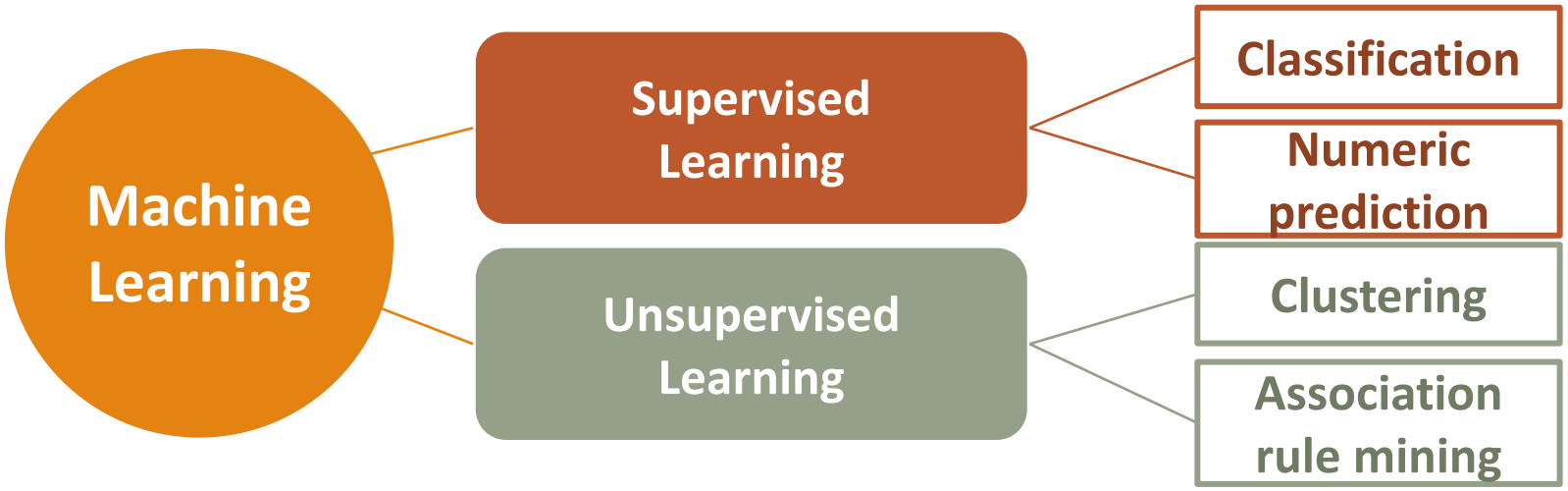
Exam Review

Data Mining Tasks



1. Identify the data mining task that would be the best choice for solving certain decision support problem
2. Understand the differences among different tasks

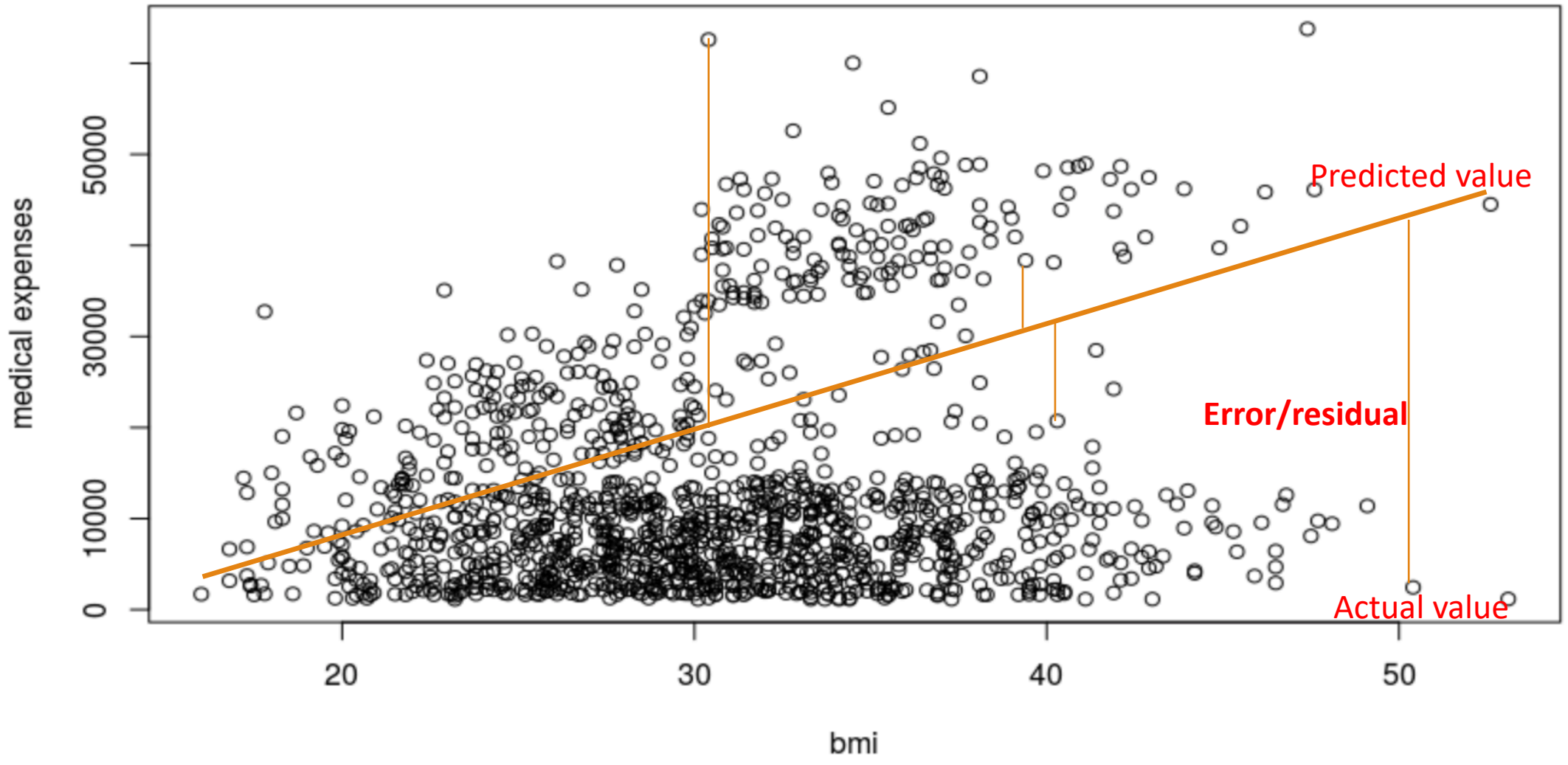
Data Mining Tasks

Method	Learning Task
Decision Trees	Classification
Naive Bayes	Classification
k-Nearest Neighbors	Classification
Linear Regression	Numeric Prediction
Regression Trees	Numeric Prediction
Model Trees	Numeric Prediction
Neural Networks	Classification/Numeric Prediction
Support Vector Machines	Classification/Numeric Prediction
K-means	Clustering
Apriori	Association Rule Mining

1. Identify the data mining algorithm that would be the best choice for solving certain decision support problem
2. Understand the mechanisms of methods and their differences

Multiple Linear Regression

Ordinary Least Squares (OLS) Estimation



Multiple Linear Regression

- The intercept is the predicted value of expenses when the independent variables are equal to zero.
  - Intercept is of little value alone because it is impossible to have values of zero for all features
- The beta coefficients indicate the estimated increase in expenses for an increase of one in each of the features, assuming all other values are held constant.

```
Call:
lm(formula = expenses ~ ., data = datTrain)

Residuals:
     Min       1Q   Median       3Q      Max
-11575.9  -2809.9   -832.8   1524.5  29716.8

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -12305.15    1147.21 -10.726  < 2e-16 ***
age               247.66      14.01  17.674  < 2e-16 ***
sexmale           228.55     392.78   0.582  0.56078
bmi               346.93      33.97  10.213  < 2e-16 ***
children          467.67     164.30   2.846  0.00452 **
smokeryes       23919.45     483.93  49.427  < 2e-16 ***
regionnorthwest    12.00     566.76   0.021  0.98311
regionsoutheast  -658.35     563.93  -1.167  0.24333
regionsouthwest  -553.26     560.17  -0.988  0.32358
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5978 on 929 degrees of freedom
Multiple R-squared:  0.7605,    Adjusted R-squared:  0.7585
F-statistic: 368.8 on 8 and 929 DF,  p-value: < 2.2e-16
```

Multiple Linear Regression

- lm() function automatically applied a technique known as **dummy coding** to each of the factor-type variables we included in the model.
- The results of the linear regression model make logical sense: old age, smoking, and obesity tend to be linked to additional health issues, while additional family member dependents may result in an increase in physician visits and preventive care such as vaccinations and yearly physical exams.

```
Call:
lm(formula = expenses ~ ., data = datTrain)

Residuals:
     Min       1Q   Median       3Q      Max
-11575.9  -2809.9   -832.8   1524.5  29716.8

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -12305.15    1147.21 -10.726  < 2e-16 ***
age                247.66      14.01  17.674  < 2e-16 ***
sexmale            228.55     392.78   0.582  0.56078
bmi                346.93      33.97  10.213  < 2e-16 ***
children           467.67     164.30   2.846  0.00452 **
smokeryes        23919.45     483.93  49.427  < 2e-16 ***
regionnorthwest     12.00     566.76   0.021  0.98311
regionsoutheast   -658.35     563.93  -1.167  0.24333
regionsouthwest   -553.26     560.17  -0.988  0.32358
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5978 on 929 degrees of freedom
Multiple R-squared:  0.7605,    Adjusted R-squared:  0.7585
F-statistic: 368.8 on 8 and 929 DF,  p-value: < 2.2e-16
```

Multiple Linear Regression

1. The **residuals** section provides summary statistics for the errors in our predictions, some of which are apparently quite substantial.
   - 50 percent of errors fall within the 1Q and 3Q values (the first and third quartile), so the majority of predictions were between $2,809.90 over the true value and $1,523.50 under the true value.

```
Call:
lm(formula = expenses ~ ., data = datTrain)

Residuals:
     Min       1Q   Median       3Q      Max
-11575.9  -2809.9   -832.8   1524.5  29716.8

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -12305.15    1147.21 -10.726  < 2e-16 ***
age                247.66      14.01  17.674  < 2e-16 ***
sexmale            228.55     392.78   0.582  0.56078
bmi                346.93      33.97  10.213  < 2e-16 ***
children           467.67     164.30   2.846  0.00452 **
smokeryes        23919.45     483.93  49.427  < 2e-16 ***
regionnorthwest     12.00     566.76   0.021  0.98311
regionsoutheast   -658.35     563.93  -1.167  0.24333
regionsouthwest   -553.26     560.17  -0.988  0.32358
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5978 on 929 degrees of freedom
Multiple R-squared:  0.7605,    Adjusted R-squared:  0.7585
F-statistic: 368.8 on 8 and 929 DF,  p-value: < 2.2e-16
```

1 2 3

Multiple Linear Regression

2. For each estimated regression coefficient, the **p-value**, denoted Pr(>|t|), provides an estimate of the probability that the true coefficient is zero given the value of the estimate.
   - Small p-values suggest that the true coefficient is very unlikely to be zero, which means that the feature is extremely unlikely to have no relationship with the dependent variable.
   - p-values less than the significance level are considered **statistically significant**.

```
Call:
lm(formula = expenses ~ ., data = datTrain)

Residuals:
     Min       1Q   Median       3Q      Max 
-11575.9  -2809.9   -832.8   1524.5  29716.8 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)    
(Intercept)     -12305.15    1147.21 -10.726  < 2e-16 ***
age                247.66      14.01  17.674  < 2e-16 ***
sexmale            228.55     392.78   0.582  0.56078    
bmi                346.93      33.97  10.213  < 2e-16 ***
children           467.67     164.30   2.846  0.00452 ** 
smokeryes        23919.45     483.93  49.427  < 2e-16 ***
regionnorthwest     12.00     566.76   0.021  0.98311    
regionsoutheast   -658.35     563.93  -1.167  0.24333    
regionsouthwest   -553.26     560.17  -0.988  0.32358    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5978 on 929 degrees of freedom
Multiple R-squared:  0.7605,    Adjusted R-squared:  0.7585 
F-statistic: 368.8 on 8 and 929 DF,  p-value: < 2.2e-16
```

1 2 3

Multiple Linear Regression

3. The **multiple R-squared value** provides a measure of how well our model as a whole explains the values of the dependent variable.
   - The model explains nearly 76 percent of the variation in the dependent variable.
   - High R-squared on training data indicates overfitting.

```
Call:
lm(formula = expenses ~ ., data = datTrain)

Residuals:
     Min       1Q   Median       3Q      Max
-11575.9  -2809.9   -832.8   1524.5  29716.8

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -12305.15    1147.21 -10.726  < 2e-16 ***
age               247.66      14.01  17.674  < 2e-16 ***
sexmale           228.55     392.78   0.582  0.56078
bmi               346.93      33.97  10.213  < 2e-16 ***
children          467.67     164.30   2.846  0.00452 **
smokeryes       23919.45     483.93  49.427  < 2e-16 ***
regionnorthwest    12.00     566.76   0.021  0.98311
regionsoutheast  -658.35     563.93  -1.167  0.24333
regionsouthwest  -553.26     560.17  -0.988  0.32358
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5978 on 929 degrees of freedom
Multiple R-squared:  0.7605,    Adjusted R-squared:  0.7585
F-statistic: 368.8 on 8 and 929 DF,  p-value: < 2.2e-16
```

1 2 3

Multiple Linear Regression Evaluation

- Make predictions on **testing data (30%)** and **training data (70%)**

age	sex	bmi	children	smoker	region	expenses	Predicted expenses
19	female	27.9	0	yes	southwest	16884.92	15445.54
18	male	33.8	1	no	southeast	1725.55	1455.21
28	male	33	3	no	southeast	4449.46	6799.98
33	male	22.7	0	no	northwest	21984.47	12351.13
32	male	28.9	0	no	northwest	3866.86	2434.12
31	female	25.7	0	no	southeast	3756.62	1235.12
46	female	33.4	1	no	southeast	8240.59	6394.12
37	female	27.7	3	no	northwest	7281.51	5450.74
37	male	29.8	2	no	northeast	6406.41	2509.23
60	female	25.8	0	no	northwest	28923.14	39455.26
25	male	26.2	0	no	northeast	2721.32	2311.65

70% training data

30% testing data

Improving Model Performance: Adding non-linear relationships

```
Call:
lm(formula = expenses ~ ., data = datTrain)

Residuals:
     Min       1Q   Median       3Q      Max
-11072.2  -2769.9   -946.4   1266.9  31341.4

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -5586.259   2002.048  -2.790 0.005374 **
age              -121.903     94.806  -1.286 0.198832
sexmale           -43.789    389.571  -0.112 0.910528
bmi               345.390     34.058  10.141  < 2e-16 ***
children          621.231    165.583   3.752 0.000186 ***
smokeryes       23847.010    481.129  49.565  < 2e-16 ***
regionnorthwest  -823.050    551.922  -1.491 0.136238
regionsoutheast -1725.564    563.951  -3.060 0.002279 **
regionsouthwest  -949.741    562.308  -1.689 0.091555 .
age2                4.813      1.182   4.071 5.07e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5921 on 928 degrees of freedom
Multiple R-squared:  0.7612,    Adjusted R-squared:  0.7589
F-statistic: 328.6 on 9 and 928 DF,  p-value: < 2.2e-16
```

Adding non-linear relationships

Increase age from 20 to 30

- (900 – 400) * 4.813 + (30-20)* (– 121.903)

Increase age from 40 to 50

- (2500 – 1600) * 4.813 + (50-40)* (– 121.903)

Improving Model Performance

```
Call:
lm(formula = expenses ~ . + bmi30 * smoker, data = datTrain)

Residuals:
     Min       1Q   Median       3Q      Max
-16954.0  -1659.8  -1195.4   -509.8  23935.7

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       -670.4243  1602.0246  -0.418  0.67569
age                -17.4262    70.3592  -0.248  0.80444
sexmale           -255.8048   288.6159  -0.886  0.37568
bmi                133.0085    40.6899   3.269  0.00112 **
children           656.4572   127.4982   5.149  3.2e-07 ***
smokeryes        13467.7242   521.2449  25.838  < 2e-16 ***
regionnorthwest     50.1895   415.8849   0.121  0.90397
regionsoutheast   -653.1946   414.0931  -1.577  0.11504
regionsouthwest  -1300.5816   411.9940  -3.157  0.00165 **
age2                3.4873     0.8752   3.984  7.3e-05 ***
bmi30           -1029.9937   501.4441  -2.054  0.04025 *
smokeryes:bmi30 19477.2235   710.6592  27.407  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4385 on 926 degrees of freedom
Multiple R-squared:  0.8716,    Adjusted R-squared:  0.8701
F-statistic: 571.3 on 11 and 926 DF,  p-value: < 2.2e-16
```

Adding non-linear relationships

For non-obese patients:

- Smoker increases the expenses by 13467.7242

For obese patients:

- Smoker increases the expenses by 13467.7242 + 19477.2235

Neural Networks

Network Topology

A neural network is put together by hooking together many of our simple "neurons," so that the output of a neuron can be the input of another.

- Leftmost layer is called **input layer**
- Rightmost layer is called **output layer**
- Middle layer of nodes is called **hidden layer**, since the value are not observed in the training set.
- The circles labeled "+1" are called **bias units**, and correspond to the intercept term.

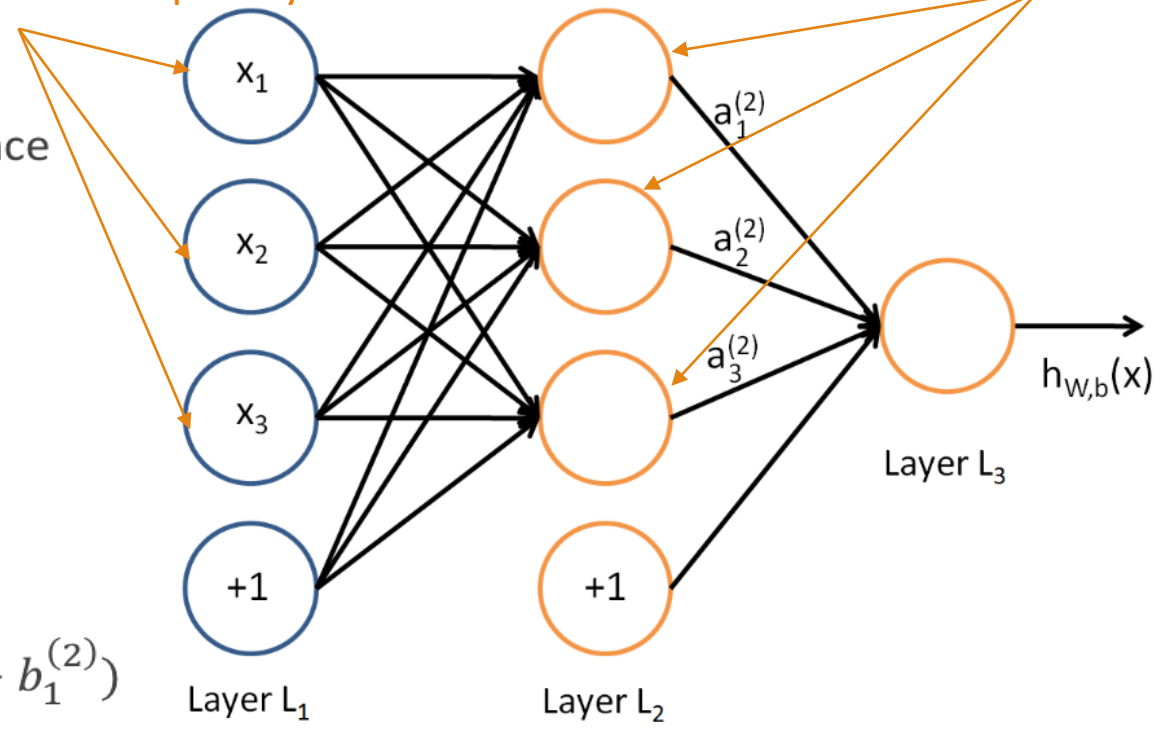
$$a_1^{(2)} = f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)})$$
$$a_2^{(2)} = f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)})$$
$$a_3^{(2)} = f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)})$$
$$h_{W,b}(x) = f(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)})$$

$f()$ is the **activation function**

Adding non-linear relationship

Three nodes on the input layer

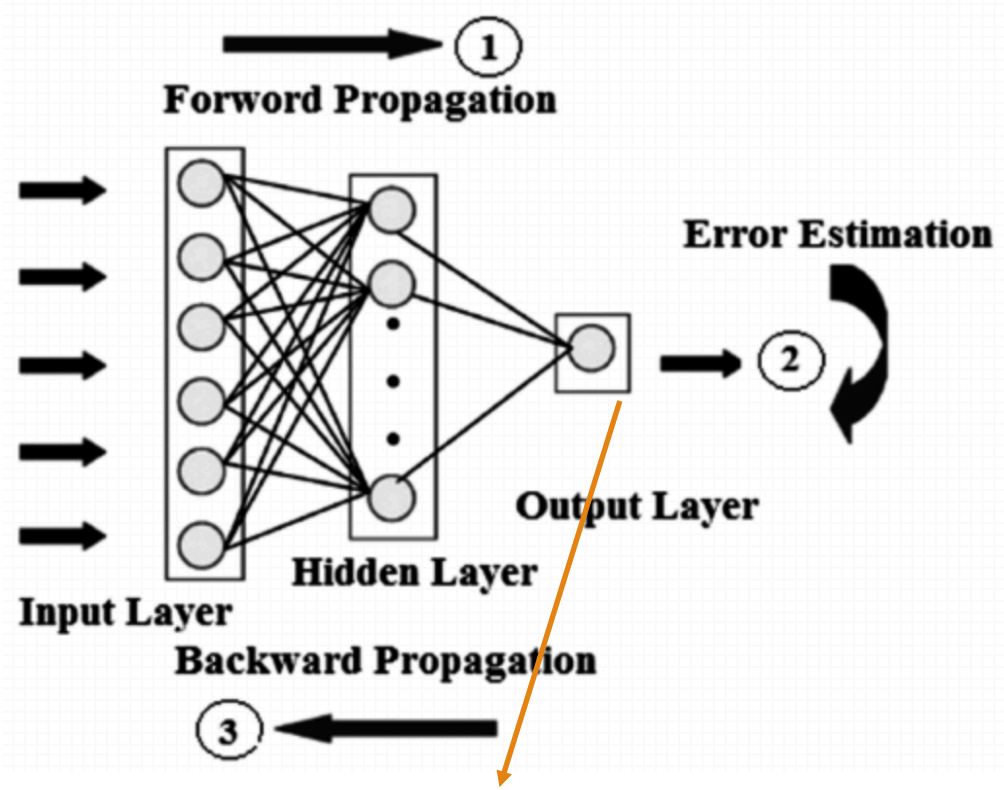
Hidden nodes/units



3 layer structure

Training Neural Networks with Backpropagation

- **Feedforward and backward propagation**
- **Input *x***: Set of input variables
- **Feedforward**: Information flows from the input layer to the output layer
- **Output error**: Compute the error with the cost function of the output value and real value
- **Backpropagate the error**: Calculate the error for each neuron from the output layer to the input layer
- **Update parameters**: Adjust the weight according to the backward propagation error



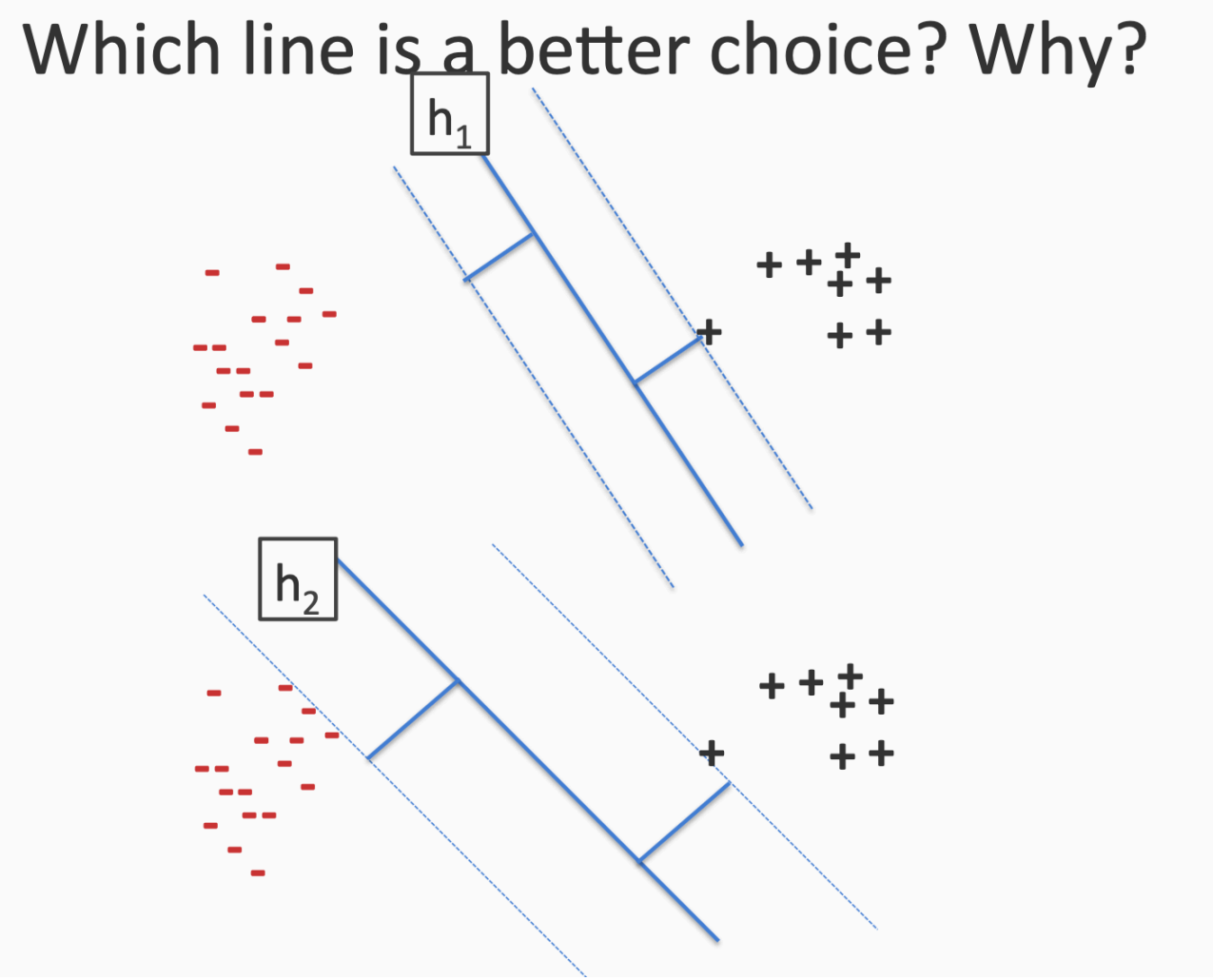
- For classification, ANN outputs class distribution
- For numeric prediction, ANN outputs predictive number

When Might ANN Be a Good Choice?

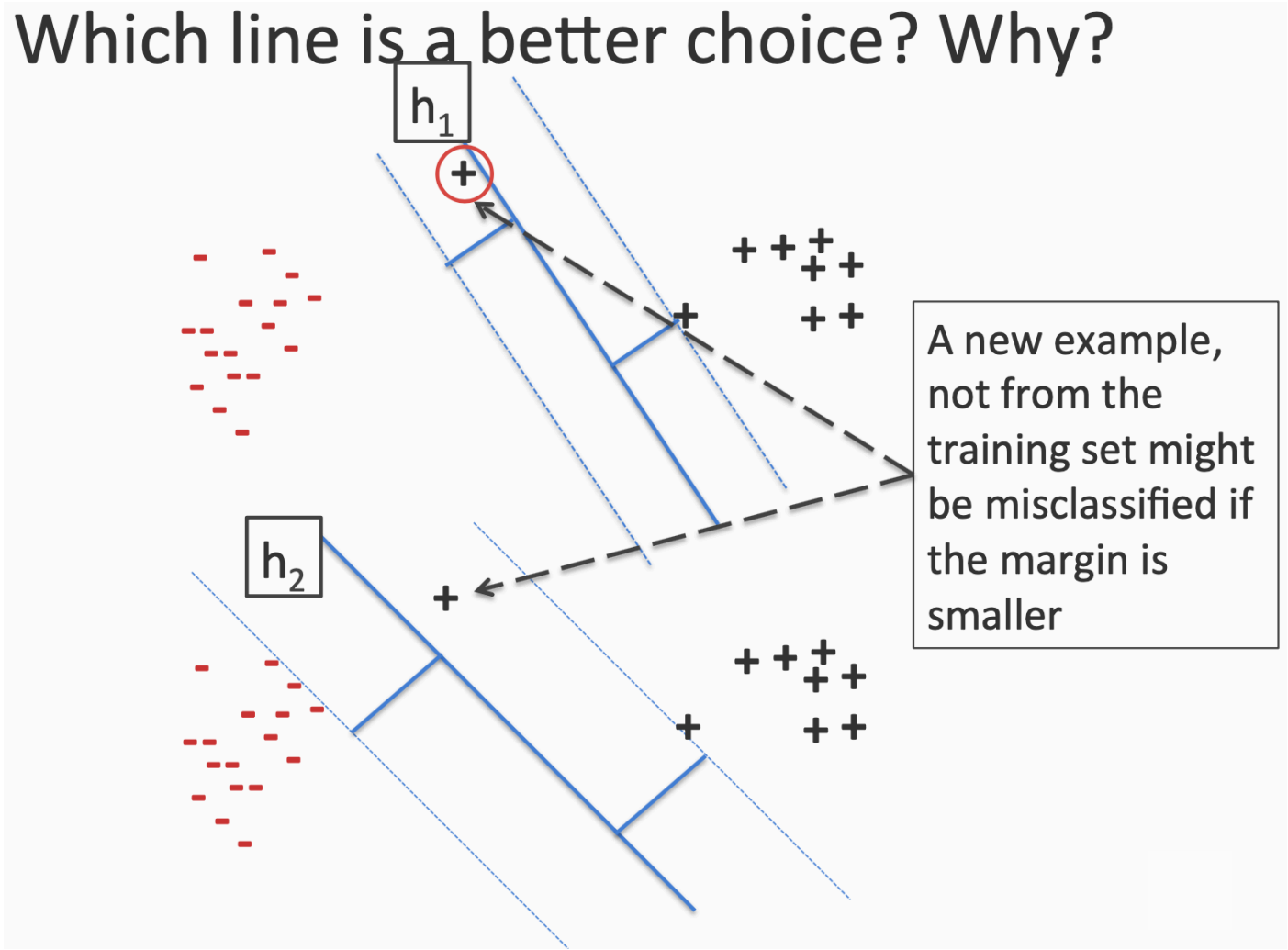
- Never hurt to try it with your application
- Studies have reported the following conditions under which ANN has outperformed other methods
  - Sufficient time and processing power to train and tune ANN models
  - Stake is high – high benefits of model accuracy – e.g., mission critical, high-prize competition
  - High-dimensional input, e.g., image, video, audial, textual input, big data
  - Possibly noisy data
  - Model interpretability is not necessary, e.g., for automated applications of dynamically learned models without human interventions
- ANN can be used for both classification and numeric prediction.

Support Vector Machines

Classification with Hyperplanes



Classification with Hyperplanes



Classification with Hyperplanes

- The case of nonlinearly separable data
  - A cost value (denoted as *C*) is applied to all points that "break into the margin"

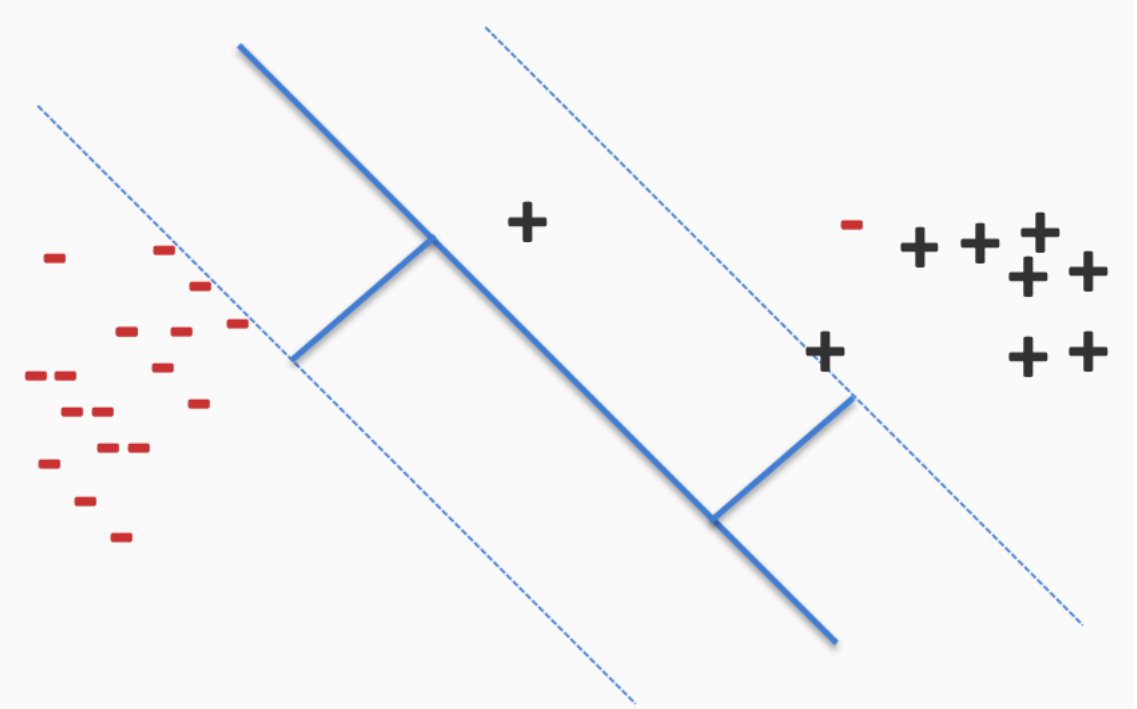
Maximize margin

Tradeoff between the two terms

$$\min_{\mathbf{w},\xi} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \xi_i$$

Minimize total slack (i.e allow as few examples as possible to violate the margin)

$$\text{s.t.} \quad \forall i, \quad y_i \mathbf{w}^T \mathbf{x}_i \geq 1 - \xi_i$$
$$\forall i, \quad \xi_i \geq 0.$$



Evaluation Metrics for Numeric Prediction

Evaluation Metrics for Numeric Prediction

- **Mean Absolute Error (MAE)**
  - The **mean absolute error (MAE)** is a quantity used to measure how close forecasts or predictions are to the eventual outcomes.
  - This is known as a scale-dependent accuracy measure and therefore cannot be used to make comparisons between series using different scales

$$\mathrm{MAE} = \frac{1}{\mathrm{n}}\sum_{t=1}^{n}|y_t - \widehat{y_t}|$$

- **Root Mean Squared Error (RMSE)**
  - RMSE gives a relatively high weight to large errors. This means the RMSE should be more useful when large errors are particularly undesirable.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{t=1}^{n}(y_t - \widehat{y_t})^2}{n}}$$

- **Mean absolute percentage error (MAPE)**
  - The **mean absolute percentage error** (**MAPE**) measures this accuracy as a percentage.
  - It cannot be used if there are zero values because there would be a division by zero.

$$\mathrm{MAPE} = \frac{1}{\mathrm{n}}\sum_{t=1}^{n}\left|\frac{y_t - \widehat{y_t}}{y_t}\right|$$

Not depend on the scale

- **Relative absolute error (RAE)**
  - The **relative absolute error** takes the total absolute error and normalizes it by dividing by the total absolute error of the mean estimator.

$$\mathrm{RAE} = \frac{\sum_{t=1}^{n}|y_t - \widehat{y_t}|}{\sum_{t=1}^{n}|y_t - \overline{y_t}|}$$

Evaluation Metrics for Numeric Prediction

- **Mean Absolute Error (MAE)**
  - Measures **scale-dependent accuracy** by **averaging the magnitudes of the forecast errors**.

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n} |y_t - \hat{y}_t|$$

Assume that you will lose each dollar your model's prediction misses due to an over-estimation or under-estimation.

- MAE tells you how much money you will lose.

```
> mmetric(datTrain2$expenses,prediction_on_train6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2355.32319 4571.88052   21.96882   25.65733
> mmetric(datTest2$expenses,prediction_on_test6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2766.02470 5064.75568   24.68685   31.13553
```

Evaluation Metrics for Numeric Prediction

- **Root Mean Squared Error (RMSE)**
  - The RMSE is easy for most people to interpret because of its similarity to the basic statistical concept of a standard deviation
  - **RMSE gives a relatively high weight to large errors**. This means the RMSE should be more useful when large errors are particularly undesirable.

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}{n}}$$

Assume that the penalty for an erroneous prediction increases with the difference between the actual and predicted values.

- RMSE is help you assess how much money you will lose

RMSE will be equal or greater than MAE

```
> mmetric(datTrain2$expenses,prediction_on_train6,metric=c("MAE","RMSE","MAPE","RAE"))
        MAE        RMSE        MAPE         RAE
2355.32319 4571.88052    21.96882    25.65733
> mmetric(datTest2$expenses,prediction_on_test6,metric=c("MAE","RMSE","MAPE","RAE"))
        MAE        RMSE        MAPE         RAE
2766.02470 5064.75568    24.68685    31.13553
```

Evaluation Metrics for Numeric Prediction

- **Mean absolute percentage error (MAPE)**
  - The **mean absolute percentage error (MAPE)** **measures predictive accuracy as a percentage**.
  - It cannot be used if there are zero values because there would be a division by zero.

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right|$$

MAPE can be used to compare model performances on different target variables

- Compare performances in predicting medical expenses in UT and CA

```
> mmetric(datTrain2$expenses,prediction_on_train6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2355.32319 4571.88052   21.96882   25.65733
> mmetric(datTest2$expenses,prediction_on_test6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2766.02470 5064.75568   24.68685   31.13553
```

Evaluation Metrics for Numeric Prediction

- **Relative absolute error (RAE)**
  - RAE is normalized MAE. RAE takes the total absolute error and normalizes it by dividing by the total absolute error of the mean estimator.

RAE indicates model effectiveness compared to a simple mean estimator

$$\text{RAE} = \frac{\sum_{t=1}^{n} |y_t - \hat{y}_t|}{\sum_{t=1}^{n} |y_t - \bar{y}_t|}$$

→ total absolute error of current predictive model

→ total absolute error of the mean estimator

```
> mmetric(datTrain2$expenses,prediction_on_train6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2355.32319 4571.88052   21.96882   25.65733
> mmetric(datTest2$expenses,prediction_on_test6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2766.02470 5064.75568   24.68685   31.13553
```

RAE and MAE are always consistent

Evaluation Metrics for Numeric Prediction

- SVM regression with different C values
  - C=1

```
> mmetric(datTrain2$expenses,prediction_on_train6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2355.32319 4571.88052   21.96882   25.65733
> mmetric(datTest2$expenses,prediction_on_test6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2766.02470 5064.75568   24.68685   31.13553
```

  - C = 5

```
> mmetric(datTrain2$expenses,prediction_on_train7,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2107.13391 4271.72790   20.41174   22.95372
> mmetric(datTest2$expenses,prediction_on_test7,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2714.76124 4982.99992   24.21993   30.55849
```

  - C = 10

```
> mmetric(datTrain2$expenses,prediction_on_train8,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2002.93488 4099.88096   20.29481   21.81864
> mmetric(datTest2$expenses,prediction_on_test8,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2784.50102 5055.22912   25.74966   31.34351
```

Which model has better performances?

Evaluation Metrics for Numeric Prediction

- SVM regression with C=5

```
> mmetric(datTrain2$expenses,prediction_on_train7,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2107.13391 4271.72790   20.41174   22.95372
> mmetric(datTest2$expenses,prediction_on_test7,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2714.76124 4982.99992   24.21993   30.55849
```

- ANN with 2 hidden layers (16, 8)

```
> mmetric(datTrain2$expenses,prediction_on_train5,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2345.88167 4585.39501   19.71702   25.55448
> mmetric(datTest2$expenses,prediction_on_test5,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2667.92899 5012.17672   23.24695   30.03132
```

Which model has better performances?

Evaluation Metrics for Numeric Prediction

- SVM regression with C=5

```
> mmetric(datTrain2$expenses,prediction_on_train7,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2107.13391 4271.72790   20.41174   22.95372
> mmetric(datTest2$expenses,prediction_on_test7,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2714.76124 4982.99992   24.21993   30.55849
```

- ANN with 2 hidden layers (8, 4)

```
> mmetric(datTrain2$expenses,prediction_on_train6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2489.76175 4507.40811   25.21162   27.12181
> mmetric(datTest2$expenses,prediction_on_test6,metric=c("MAE","RMSE","MAPE","RAE"))
       MAE       RMSE       MAPE        RAE
2795.43547 4930.50234   27.64945   31.46659
```

Real value: $y_1 = 1, y_2 = 10$

Prediction1: $\widehat{y_1} = 2, \widehat{y_2} = 8$

- $e_1 = 1,\ e_2 = 2$
- MAE = 1.5, RMSE = 1.58, MAPE = 60

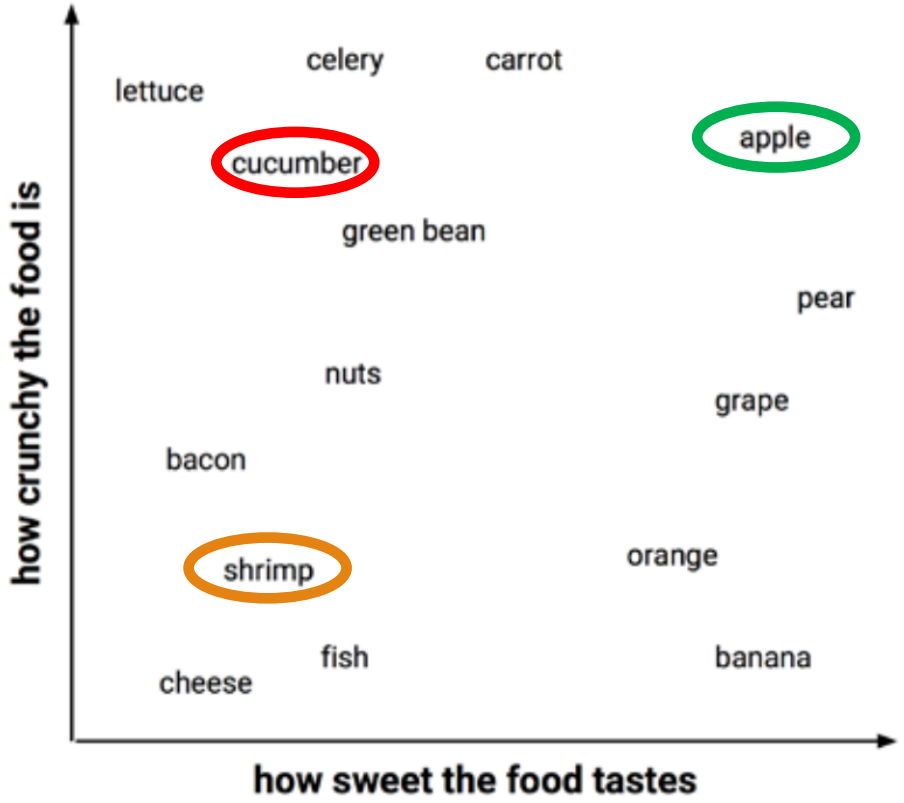
Prediction2: $\widehat{y_1} = 3, \widehat{y_2} = 9$

- $e_1 = 2,\ e_2 = 1$
- MAE = 1.5 , RMSE = 1.58
- MAPE = 105

Clustering

The k-means Clustering Algorithm

- Step 1: choosing *k* points in the feature space to serve as the cluster centers.



k = *3*: 3 data points are randomly selected

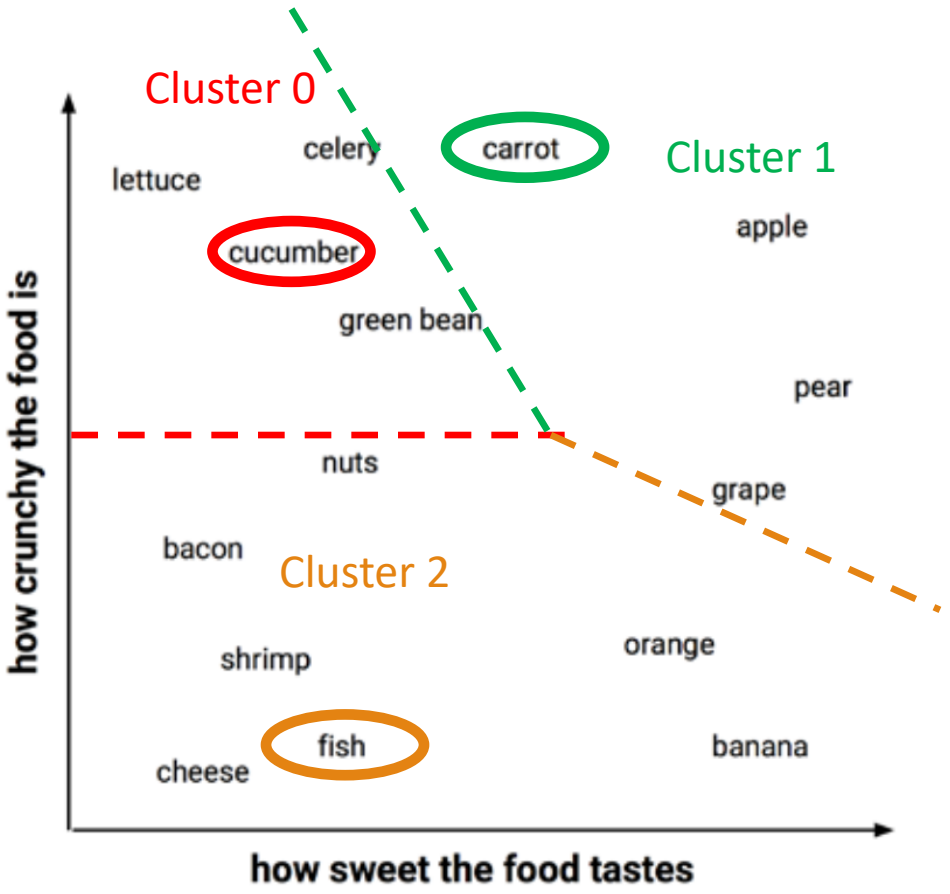
The k-means Clustering Algorithm

- Step 2: Assigning the remaining examples to the cluster center that is nearest according to the distance.
  - Euclidean distance between example *x* and example *y*

$$\text{dist}(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

  - Using this distance function, we find the distance between each example and each cluster center. The example is then assigned to the nearest cluster center.

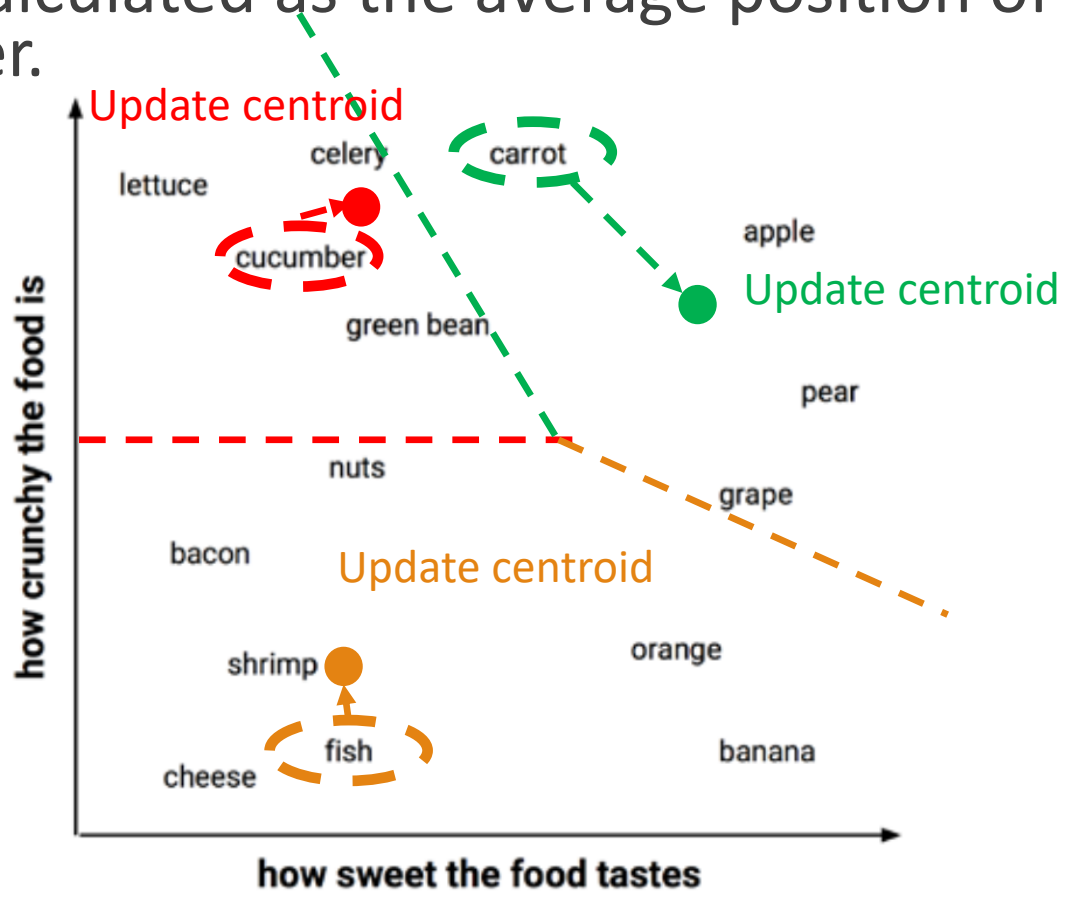
The k-means Clustering Algorithm



Keep in mind that as we are using distance calculations, all the features need to be numeric, and the values should be normalized to a standard range ahead of time.

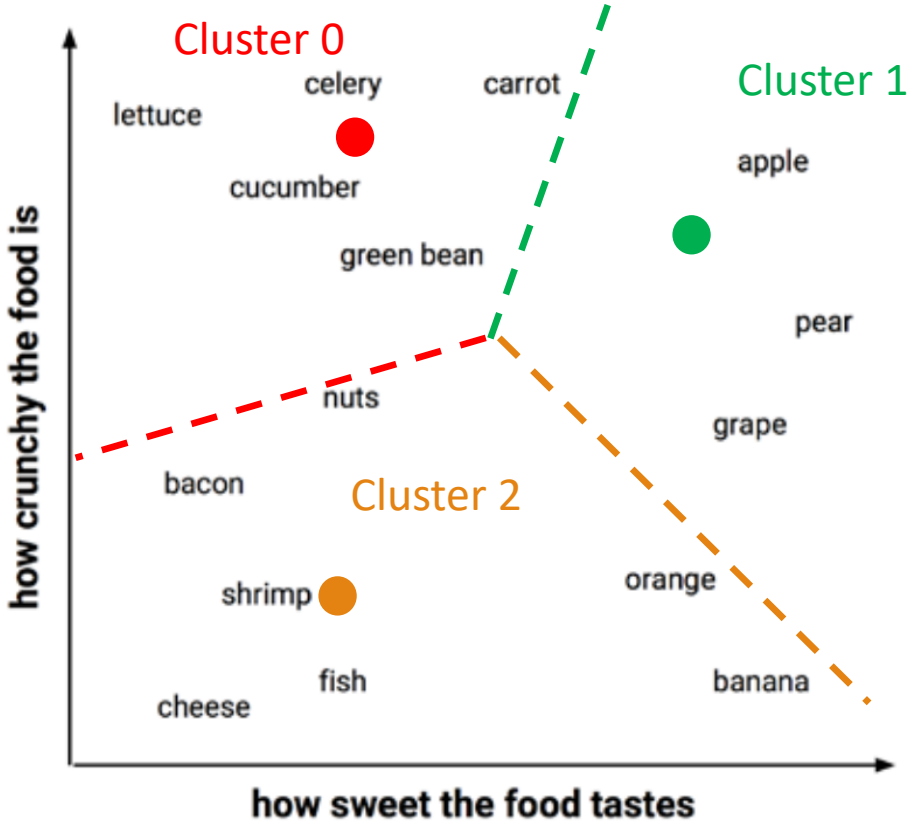
The k-means Clustering Algorithm

- Step 3: shifting the initial centers to a new location, known as the **centroid**, which is calculated as the average position of the points currently assigned to that cluster.



The k-means Clustering Algorithm

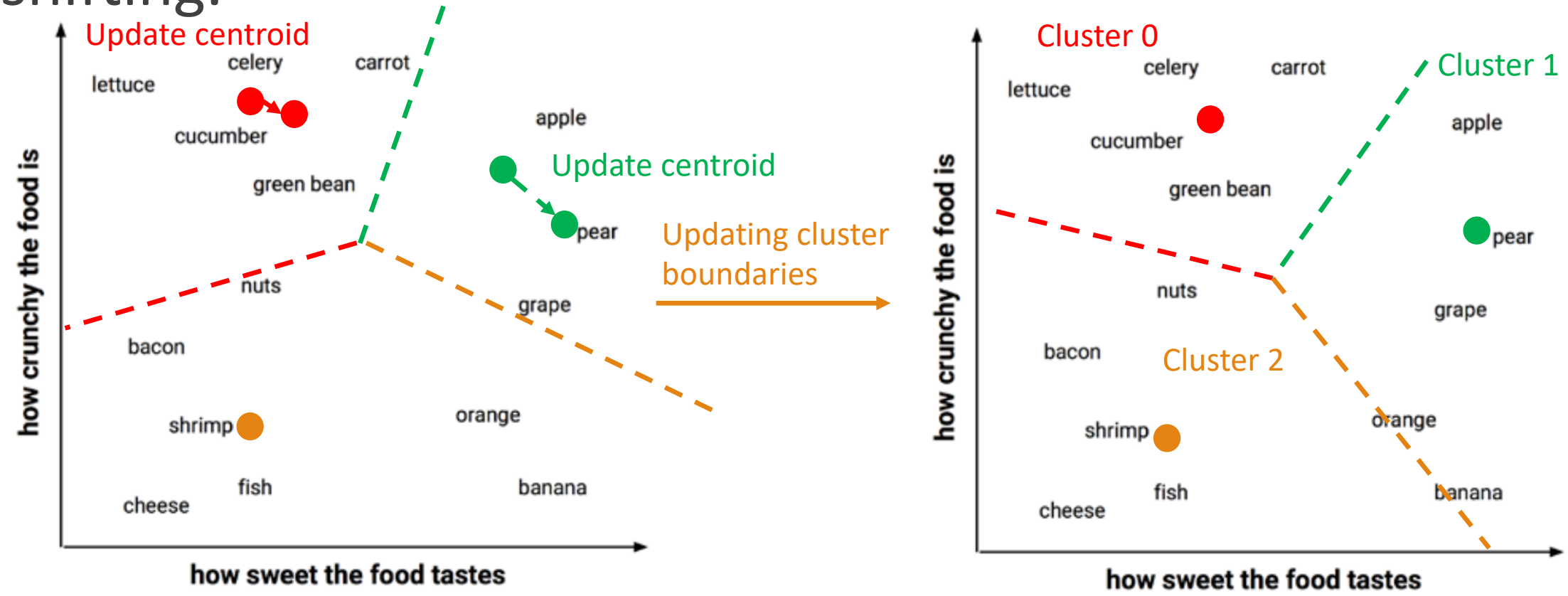
- Step 4: Updating the cluster boundaries, and reassigning points into new clusters.



Update clusters

The k-means Clustering Algorithm

- Step 5: Repeat step 3 and step 4 until the centroids stop shifting.



The k-means Clustering Algorithm

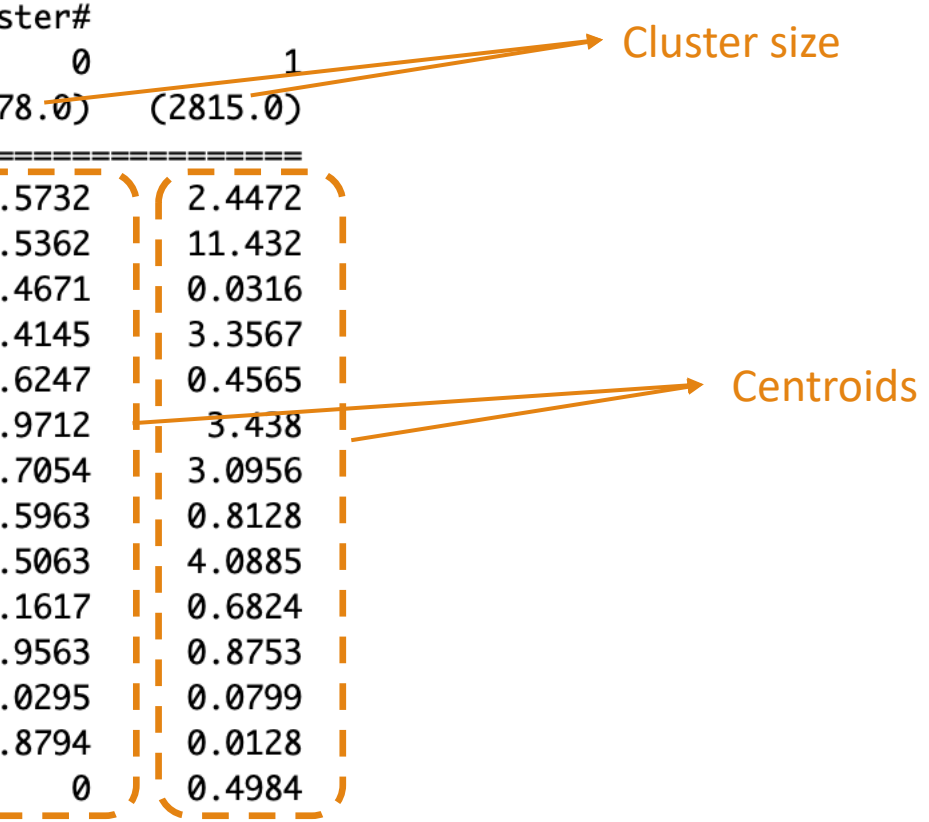
- Step 1: choosing *k* points in the feature space to serve as the cluster centers.
- Step 2: Assigning the remaining examples to the cluster center that is nearest according to the distance.
- Step 3: shifting the initial centers to a new location, known as the **centroid**, which is calculated as the average position of the examples currently assigned to that cluster.
- Step 4: Updating the cluster boundaries, and reassigning examples into new clusters.
- Step 5: Repeat step 3 and step 4 until the centroids stop shifting.

Clustering on BART Riders

- k=2

Final cluster centroids:

Attribute	Full Data	Cluster# 0	1
	(5493.0)	(2678.0)	(2815.0)
Age	3.4837	4.5732	2.4472
DistToWork	11.4828	11.5362	11.432
DualInc	0.2439	0.4671	0.0316
Education	3.8724	4.4145	3.3567
Gender	0.5385	0.6247	0.4565
Income	5.1606	6.9712	3.438
NbrInHouseHold	2.9053	2.7054	3.0956
NbrInHouseholdUnder18	0.7073	0.5963	0.8128
YrsInArea	4.2922	4.5063	4.0885
Rider	0.4285	0.1617	0.6824
Language_English	0.9148	0.9563	0.8753
Language_Spanish	0.0553	0.0295	0.0799
OwnRent_own	0.4353	0.8794	0.0128
OwnRent_Parent	0.2554	0	0.4984



Clustering on BART Riders

- k=3

Final cluster centroids:

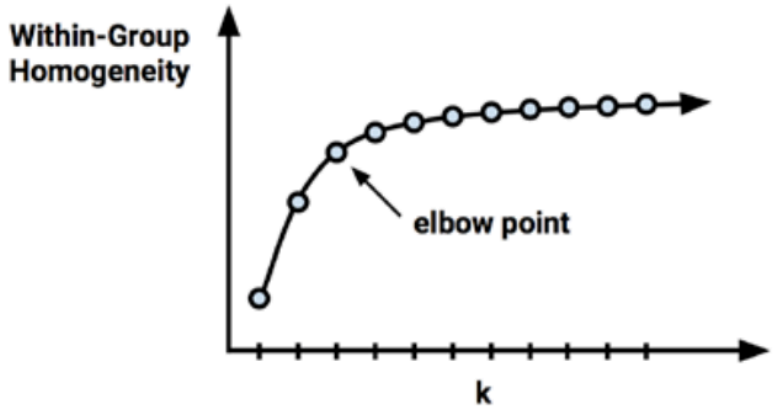
Attribute	Full Data (5493.0)	Cluster# 0 (2368.0)	1 (1210.0)	2 (1915.0)
Age	3.4837	4.6917	3.2603	2.1311
DistToWork	11.4828	11.5051	11.5231	11.4298
DualInc	0.2439	0.4472	0.1694	0.0397
Education	3.8724	4.4054	4.1347	3.0475
Gender	0.5385	0.5904	0.4893	0.5055
Income	5.1606	6.9949	5.6868	2.5598
NbrInHouseHold	2.9053	2.7758	2.2653	3.47
NbrInHouseholdUnder18	0.7073	0.6326	0.3702	1.0125
YrsInArea	4.2922	4.5743	3.9256	4.1749
Rider	0.4285	0.1858	0.0149	0.9901
Language_English	0.9148	0.951	0.9612	0.8407
Language_Spanish	0.0553	0.0329	0.0256	0.1018
OwnRent_own	0.4353	0.9996	0	0.0125
OwnRent_Parent	0.2554	0	0.1157	0.6595

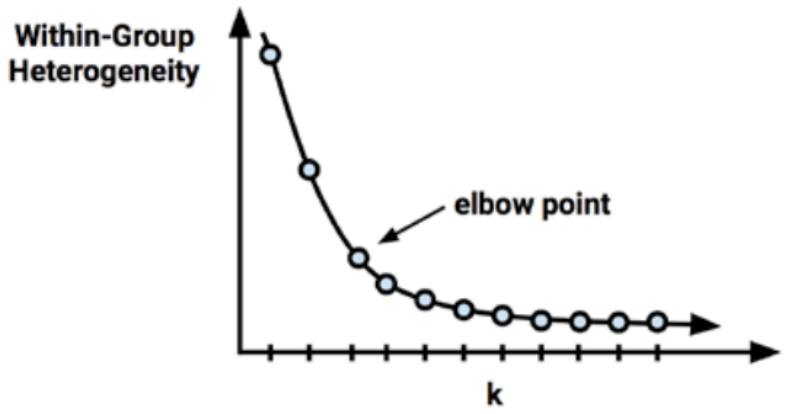
Based on the cluster size and centroids, which k (k=2 or k=3) you will use, and why?

If we segment residents into 3 clusters, what marketing plans you can use to target each cluster?

Elbow Method

- Choosing the appropriate number of clusters: **elbow method**
  - You could continue to see improvements until each example is in its own cluster, the goal is not to maximize homogeneity or minimize heterogeneity, but rather to find *k* so that there are diminishing returns beyond that point. This value of *k* is known as the **elbow point** because it looks like an elbow .





Classification with Clustering

- Classification with Clustering
  - Step 1: clustering
  - Step 2: build a classification model for each cluster

Classification with Clustering

- Testing performance with cross validation

```
> df <- BartRider
> target <- 10
> nFolds <- 10
> seedVal <- 1
> prediction_method <- C5.0
> metrics_list <- c("ACC","PRECISION","TPR","F1")
> cv_function_test(df, target, nFolds, seedVal, prediction_method, metrics_list)
```

	ACC	PRECISION1	PRECISION2	TPR1	TPR2	F11	F12
Fold01	89.44	88.79	90.41	93.31	84.26	90.99	87.22
Fold02	87.45	90.97	83.27	86.62	88.56	88.74	85.83
Fold03	90.53	93.38	87.04	89.81	91.49	91.56	89.21
Fold04	87.27	86.09	89.15	92.68	80.08	89.26	84.38
Fold05	90.16	92.48	87.24	90.13	90.21	91.29	88.70
Fold06	88.52	90.88	85.54	88.85	88.09	89.86	86.79
Fold07	88.34	90.32	85.77	89.17	87.23	89.74	86.50
Fold08	89.80	89.78	89.82	92.65	86.02	91.19	87.88
Fold09	88.16	88.07	88.29	91.72	83.40	89.86	85.78
Fold10	92.36	95.33	88.80	91.08	94.07	93.16	91.36
Mean	89.20	90.61	87.53	90.60	87.34	90.57	87.36
Sd	1.57	2.68	2.21	2.08	4.11	1.31	2.01

No clustering: use the whole dataset to perform 10-fold cross validation

Classification with Clustering

- Perform clustering without target variable

Final cluster centroids:

Attribute	Full Data (5493.0)	Cluster# 0 (2389.0)	1 (1404.0)	2 (1700.0)
Age	3.4837	4.6735	1.7521	3.2418
DistToWork	11.4828	11.5036	11.4309	11.4965
DualInc	0.2439	0.4429	0.0064	0.1606
Education	3.8724	4.3951	2.76	4.0565
Gender	0.5385	0.5873	0.5726	0.4418
Income	5.1606	6.9481	2.9915	4.44
NbrInHouseHold	2.9053	2.7848	3.9972	2.1729
NbrInHouseholdUnder18	0.7073	0.6342	1.2778	0.3388
YrsInArea	4.2922	4.5701	4.4295	3.7882
Language_English	0.9148	0.9473	0.8504	0.9224
Language_Spanish	0.0553	0.036	0.0933	0.0512
OwnRent_own	0.4353	1	0	0.0012
OwnRent_Parent	0.2554	0	0.9993	0

Classification with Clustering

- Create three clusters based on the clustering results

```
BartRider$class_ids = BartRider_clustering3$class_ids
BartRider1 = BartRider[which(BartRider$class_ids==0),]
BartRider2 = BartRider[which(BartRider$class_ids==1),]
BartRider3 = BartRider[which(BartRider$class_ids==2),]
```

Classification with Clustering

```
'data.frame':   2389 obs. of  15 variables:
 $ Age   'data.frame':   1404 obs. of  15 variables:
 $ DistTo $ Age  'data.frame':   1700 obs. of  15 variables:
 $ DualIn $ Dist $ Age                    : int  7 3 2 3 2 4 6 3 7 3 ...
 $ Educat $ Dual $ DistToWork             : int  14 9 10 12 13 11 13 10 10 9 ...
 $ Gender $ Educ $ DualInc                : num  0 0 0 0 0 0 1 0 0 1 ...
 $ Income $ Gend $ Education              : int  3 3 4 4 3 3 1 5 4 4 ...
 $ NbrInH $ Inco $ Gender                 : num  1 0 0 1 0 1 1 1 1 1 ...
 $ NbrInH $ NbrI $ Income                 : int  3 1 3 4 3 3 2 4 1 6 ...
 $ YrsInA $ NbrI $ NbrInHouseHold         : int  1 1 4 4 1 1 4 2 1 2 ...
 $ Rider  $ YrsI $ NbrInHouseholdUnder18: int  0 0 2 1 0 0 1 0 0 0 ...
 $ Langua $ Ride $ YrsInArea              : int  5 5 2 2 5 4 3 3 5 1 ...          2 ...
 $ Langua $ Lang $ Rider                  : Factor w/ 2 levels "0","1": 2 2 2 2 2
 $ OwnRen $ Lang $ Language_English       : num  1 1 1 1 1 1 0 1 1 1 ...
 $ OwnRen $ OwnR $ Language_Spanish       : num  0 0 0 0 0 0 1 0 0 0 ...
 $ class_ $ OwnR $ OwnRent_own            : num  0 0 0 0 0 0 0 0 0 0 ...
          $ clas $ OwnRent_Parent         : num  0 0 0 0 0 0 0 0 0 0 ...
                 $ class_ids              : int  2 2 2 2 2 2 2 2 2 2 ...
```

Classification with Clustering

- Performance on cluster 0

	ACC	PRECISION1	PRECISION2	TPR1	TPR2	F11	F12
Fold01	92.47	94.87	81.82	95.85	78.26	95.36	80.00
Fold02	93.31	92.75	96.88	99.48	67.39	96.00	79.49
Fold03	94.56	95.45	90.24	97.93	80.43	96.68	85.06
Fold04	91.21	93.43	80.49	95.85	71.74	94.63	75.86
Fold05	92.47	93.53	86.84	97.41	71.74	95.43	78.57
Fold06	92.02	95.77	77.55	94.27	82.61	95.01	80.00
Fold07	91.67	92.61	86.49	97.41	68.09	94.95	76.19
Fold08	94.54	96.86	85.11	96.35	86.96	96.61	86.02
Fold09	92.05	93.50	84.62	96.89	71.74	95.17	77.65
Fold10	91.21	93.88	79.07	95.34	73.91	94.60	76.40
Mean	92.55	94.27	84.91	96.68	75.29	95.44	79.52
Sd	1.22	1.40	5.73	1.48	6.49	0.75	3.53

Use the cluster 0
to perform 10-fold cross validation

Classification with Clustering

- Performance on cluster 1

	ACC	PRECISION1	PRECISION2	TPR1	TPR2	F11	F12
Fold01	90.00	75.00	90.44	18.75	99.19	30.00	94.62
Fold02	89.29	60.00	90.37	18.75	98.39	28.57	94.21
Fold03	89.29	66.67	89.78	12.50	99.19	21.05	94.25
Fold04	89.36	55.56	91.67	31.25	96.80	40.00	94.16
Fold05	92.20	100.00	91.91	31.25	100.00	47.62	95.79
Fold06	90.78	66.67	92.42	37.50	97.60	48.00	94.94
Fold07	92.20	100.00	91.91	31.25	100.00	47.62	95.79
Fold08	90.00	66.67	91.04	25.00	98.39	36.36	94.57
Fold09	92.86	100.00	92.54	37.50	100.00	54.55	96.12
Fold10	92.86	100.00	92.54	37.50	100.00	54.55	96.12
Mean	90.88	79.06	91.46	28.12	98.96	40.83	95.06
Sd	1.50	18.70	1.00	8.96	1.14	11.54	0.81

Use the cluster 1
to perform 10-fold cross validation

Classification with Clustering

- Performance on cluster 2

	ACC	PRECISION1	PRECISION2	TPR1	TPR2	F11	F12
Fold01	81.76	82.46	80.36	89.52	69.23	85.84	74.38
Fold02	87.57	88.89	85.25	91.43	81.25	90.14	83.20
Fold03	83.53	86.67	78.46	86.67	78.46	86.67	78.46
Fold04	85.29	90.82	77.78	84.76	86.15	87.68	81.75
Fold05	84.12	86.11	80.65	88.57	76.92	87.32	78.74
Fold06	80.00	84.47	73.13	82.86	75.38	83.65	74.24
Fold07	82.94	88.00	75.71	83.81	81.54	85.85	78.52
Fold08	81.76	83.64	78.33	87.62	72.31	85.58	75.20
Fold09	77.78	84.00	69.01	79.25	75.38	81.55	72.06
Fold10	82.94	82.20	84.62	92.38	67.69	87.00	75.21
Mean	82.77	85.72	78.33	86.69	76.43	86.13	77.18
Sd	2.72	2.87	4.92	4.07	5.73	2.32	3.55

Use the cluster 2
to perform 10-fold cross validation

Association Rule Mining

The Apriori Algorithm for Association Rule Learning

- Rules like *{get well card}* → *{flowers}* are known as **strong rules (interesting rules)**, because they have both high support and confidence.
  - Support value: 0.6
  - Confidence value: 1.0

Apriori Algorithm

- Consider a supermarket scenario
  - Sell five items: Onion, Burger, Potato, Milk, Beer.
  - The database consists of six transactions where 1 represents the presence of the item and 0 the absence.

Transaction ID	Onion	Potato	Burger	Milk	Beer
t_1	1	1	1	0	0
t_2	0	1	1	1	0
t_3	0	0	0	1	1
t_4	1	1	0	1	0
t_5	1	1	1	0	1
t_6	1	1	1	1	0

The Apriori Algorithm makes the following assumptions.

- All subsets of a frequent itemset should be frequent.
- The itemset containing infrequent items should be infrequent.
- Set a threshold support level. In our case, we shall fix it at 50%.

Apriori Algorithm

- **Step 1.** Create a frequency table of all the items that occur in all the transactions.
- **Step 2.** Here, support threshold is 50%, hence only those items are frequent which occur in equal or more than three transactions and such items are Onion(O), Potato(P), Burger(B), and Milk(M).
- **Step 3.** make all the possible pairs of the significant items (the order doesn't matter)

1-item itemsets

Item	Frequency (No. of transactions)
Onion(O)	4
Potato(P)	5
Burger(B)	4
Milk(M)	4
Beer(Be)	2

Generate frequent items

1-item itemsets that meet the minimum support threshold

Item	Frequency (No. of transactions)
Onion(O)	4
Potato(P)	5
Burger(B)	4
Milk(M)	4

2-item itemsets

Itemset	Frequency (No. of transactions)
OP	4
OB	3
OM	2
PB	4
PM	3
BM	2

- **Step 4.** Only those itemsets are significant which cross the support threshold (OP, OB, PB, and PM)
- **Step 5.** Now we would like to look for 3-item itemsets that are purchased together. We will use the item sets found in step 4 and create 3-item itemsets.
- **Step 6.** Applying the threshold rule again, we find that OPB is the only frequent 3-item itemset.

2-item itemsets

Itemset	Frequency (No. of transactions)
OP	4
OB	3
OM	2
PB	4
PM	3
BM	2

2-item itemsets that meet the minimum support threshold

Itemset	Frequency (No. of transactions)
OP	4
OB	3
OM	2
PB	4
PM	3
BM	2

3-item itemsets

Itemset	Frequency (No. of transactions)
OPB	4

How about OPM, OBM, and PBM?

- OM and BM does not meet a desired support threshold

The Apriori Algorithm for Association Rule Learning

- Frequent itemsets
  - O, P, B, M, OP, OB, PB, PM, and OPB
- From frequent itemsets to association rules
  - O -> P, P -> O
  - O -> B, B -> O
  - P -> B, B -> P
  - P -> M, M -> P
  - {O,P} -> B, {B,P} -> O, {O,B} -> P, B -> {O,P}, O -> {B,P}, P -> {O,B}
- Exam the confidence value to generate **strong rules**

The Apriori Algorithm for Association Rule Learning

- Building a set of **strong rules** with the Apriori principle
  - Apriori algorithm creates rules in two phase:
    1. Identifying all the itemsets that meet a minimum support threshold.
    2. Creating rules from these itemsets using those meeting a minimum confidence threshold.

Identifying Frequently Purchased Groceries with Association Rules

```
> summary(groceries)
transactions as itemMatrix in sparse format with
 9835 rows (elements/itemsets/transactions) and
 169 columns (items) and a density of 0.02609146

most frequent items:
      whole milk other vegetables       rolls/buns             soda           yogurt          (Other)
            2513             1903             1809             1715             1372            34055

element (itemset/transaction) length distribution:
sizes
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24   26   27   28
2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55   46   29   14   14    9   11    4    6    1    1    1    1
  29   32
   3    1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.000   3.000   4.409   6.000  32.000
```

The output 9835 rows refers to the number of transactions

The output 169 columns refers to the 169 different items that might appear in someone's grocery basket.

Items that were most commonly found in the transactional data

A total of 2,159 transactions contained only a single item, while one transaction had 32 items

Identifying Frequently Purchased Groceries with Association Rules

- The first rule can be read in plain language as, "if a customer buys butter, he/she will also buy whole milk."

```
> # Display top 5 rules
> inspect(sort(groceries_rules, by = "confidence")[1:5])
    lhs                     rhs                support    confidence lift     count
[1] {butter}             => {whole milk}       0.02755465 0.4972477  1.946053 271
[2] {curd}               => {whole milk}       0.02613116 0.4904580  1.919481 257
[3] {domestic eggs}      => {whole milk}       0.02999492 0.4727564  1.850203 295
[4] {onions}             => {other vegetables} 0.01423488 0.4590164  2.372268 140
[5] {whipped/sour cream} => {whole milk}       0.03223183 0.4496454  1.759754 317
```

Identifying Frequently Purchased Groceries with Association Rules

- Support for association rule X->Y: $support(X,Y) = \frac{count(X,Y)}{N}$

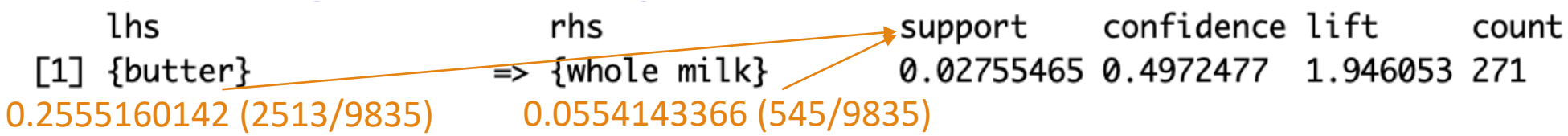
- Confidence is defined as the support of the itemset containing both *X* and *Y* divided by the support of the itemset containing only *X*:

$$\text{confidence}(X \rightarrow Y) = \frac{\text{support}(X,Y)}{\text{support}(X)}$$

- The lift of a rule measures: how much more likely one item or itemset is purchased relative to its typical rate of purchase, given that you know another item or itemset has been purchased.
  - Unlike confidence where the item order matters, lift(X → Y) is the same as lift(Y → X).

$$\text{lift}(X \rightarrow Y) = \frac{\text{confidence}(X \rightarrow Y)}{\text{support}(Y)} = \frac{support(X,Y)}{support(X) * support(Y)}$$

Identifying Frequently Purchased Groceries with Association Rules



- Support: 0.02755465 (271/9835)
  - The probability of buying butter and whole milk together is 0.02755465
  - 2.755% transactions include both butter and whole milk. This rule covers 2.755% transactions.
- Confidence: 0.4972477 (271/545)
  - When customers bought butter, they will also buy whole milk about 49.72477% of the time.
  - Among transactions including butter, 49.72477% of them also include whole milk.
- Lift: 1.946053 (0.02755465 /(0.0554143366*0.2555160142))
  - Butter and whole milk have positive effect (because lift value greater than 1) on each other.
  - Purchasing butter increases the probability of buying whole milk by 1.946 times.
  - Purchasing whole milk increases the probability of buying butter by 1.946 times.

Identifying Frequently Purchased Groceries with Association Rules

In spite of the fact that the confidence and lift are high, does {butter} → {whole milk} seem like a very useful rule?

Identifying Frequently Purchased Groceries with Association Rules

- A common approach is to take the association rules and divide them into the following three categories:
  - Actionable
  - Trivial
  - Inexplicable
- Obviously, the goal of a market basket analysis is to find actionable rules that provide a clear and useful insight.
  - Some rules are clear, others are useful
  - it is less common to find a combination of both of these factors.

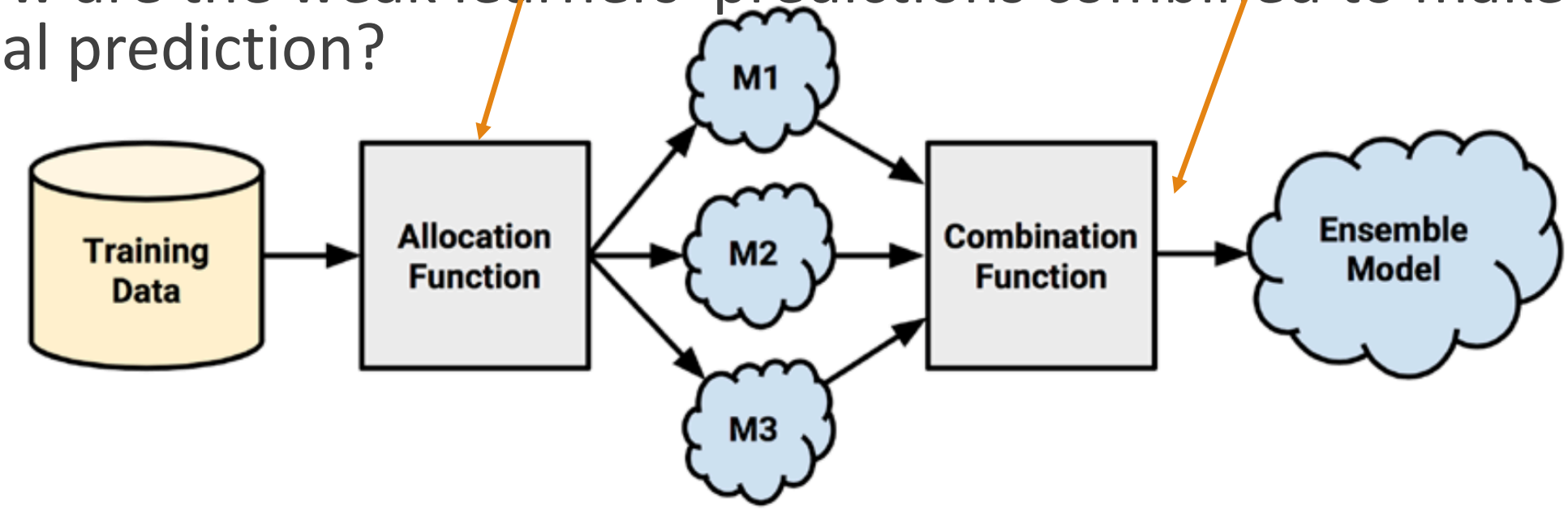
Identifying Frequently Purchased Groceries with Association Rules

- **Trivial rules** include any rules that are so obvious that they are not worth mentioning
  - They are clear, but not useful.
  - {diapers} → {formula}
- Rules are **inexplicable** if the connection between the items is so unclear that figuring out how to use the information is impossible or nearly impossible.
  - Simply be a random pattern in the data

Improving Model Performance

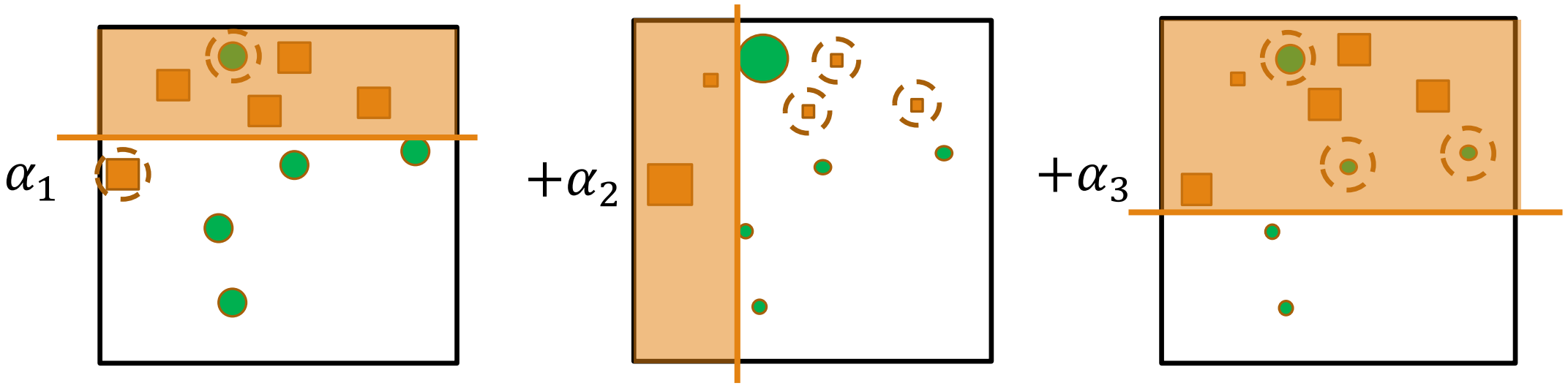
Understanding Ensembles

- **Ensemble** method is based on the idea that by combining multiple weaker learners, a stronger learner is created.
  - How are the weak learning models chosen and/or constructed?
  - How are the weak learners' predictions combined to make a single final prediction?



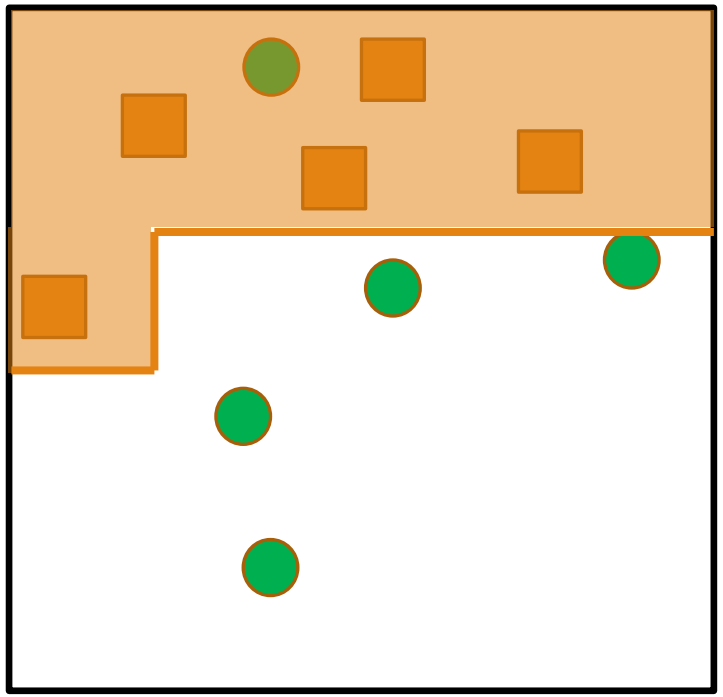
Ensemble Method: Boosting

- **AdaBoost** or **adaptive boosting**



Ensemble Method: Boosting

- **AdaBoost** or **adaptive boosting**



Final Exam

- 13 multiple choice, True/False, multiple answers
- Describe differences between two methods
- Multiple linear regression
- Model performance (numeric prediction) analysis
- Clustering
- Association rule mining
- R code