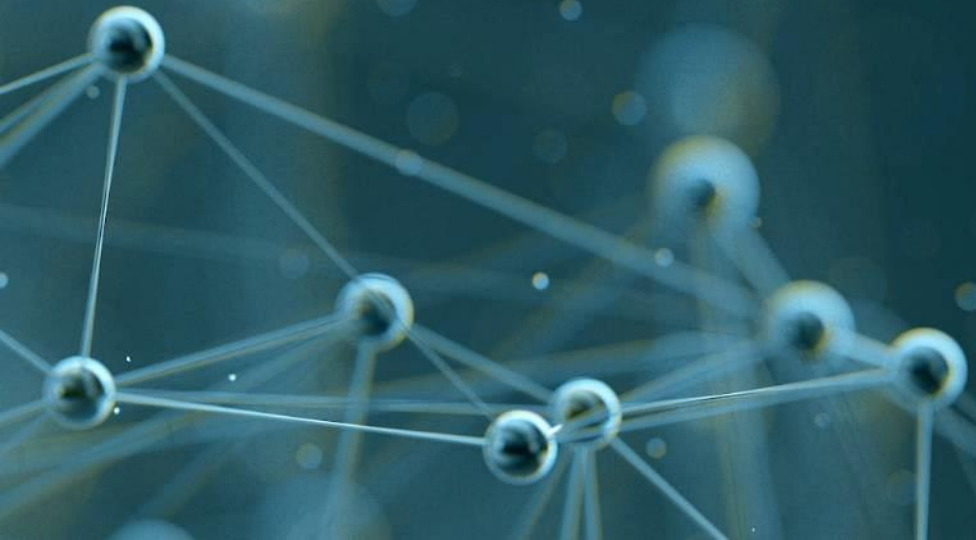
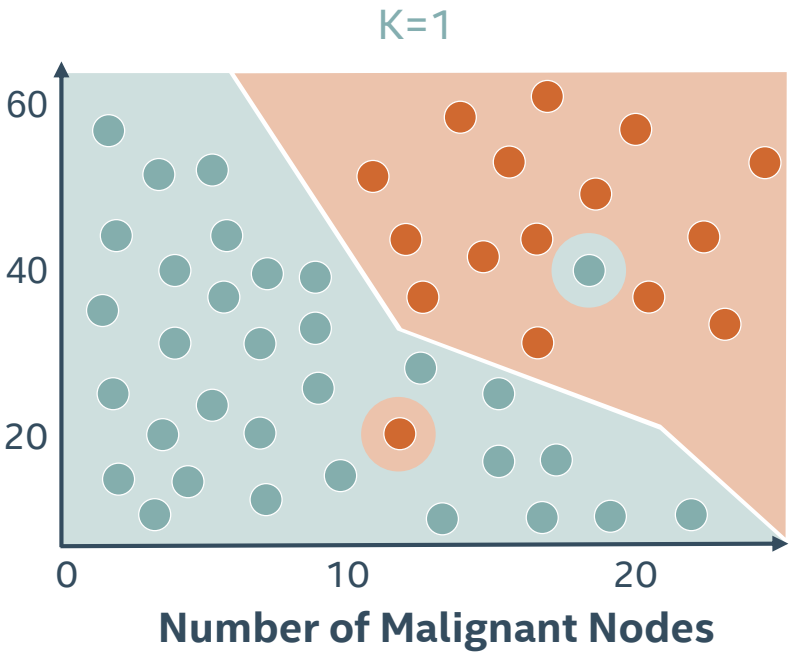
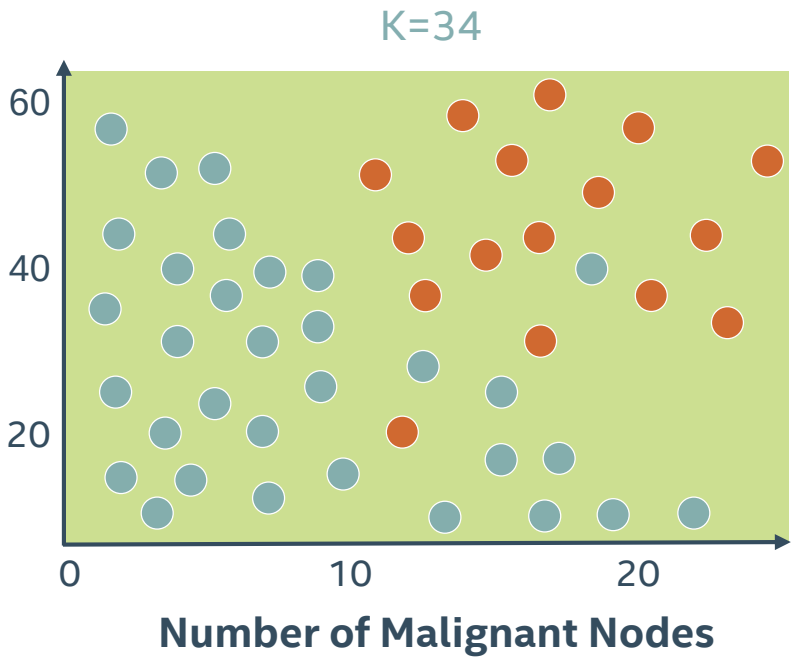
MODEL GENERALIZATION



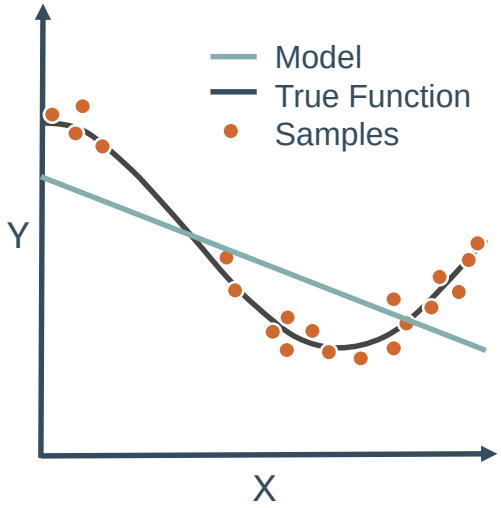
K VALUE AFFECTS DECISION BOUNDARY



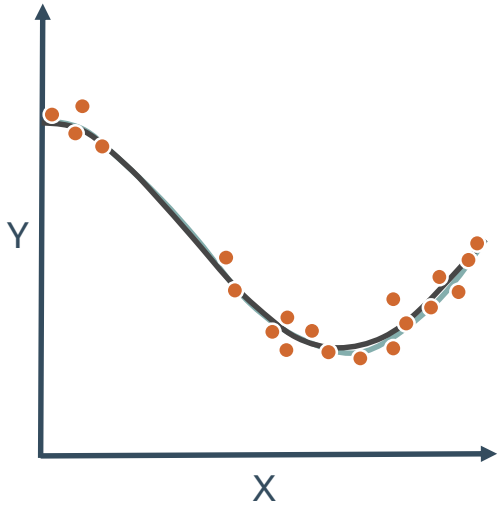


CHOOSING BETWEEN DIFFERENT COMPLEXITIES

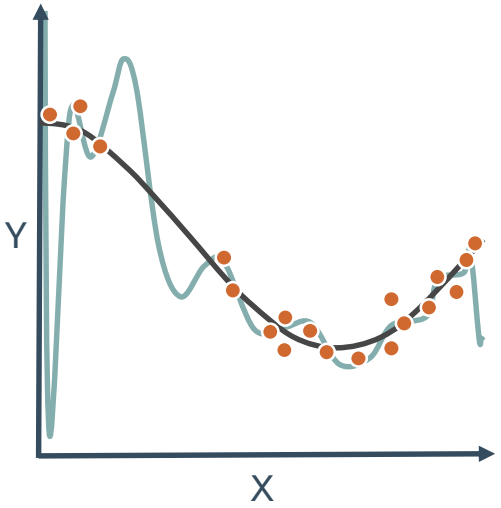
Polynomial Degree = 1



Polynomial Degree = 4

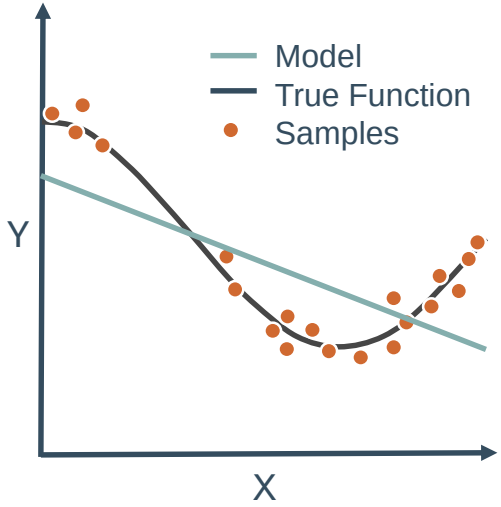


Polynomial Degree = 15



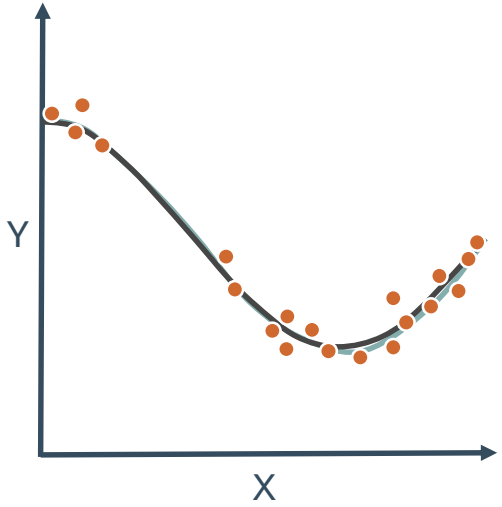
HOW WELL DOES THE MODEL GENERALIZE?

Polynomial Degree = 1



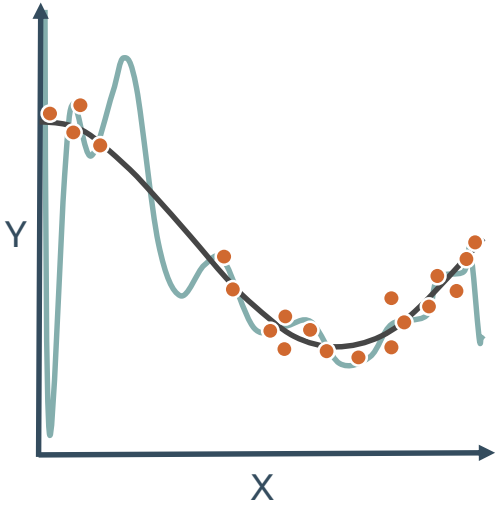
Poor at Training
Poor at Predicting

Polynomial Degree = 4



Just Right

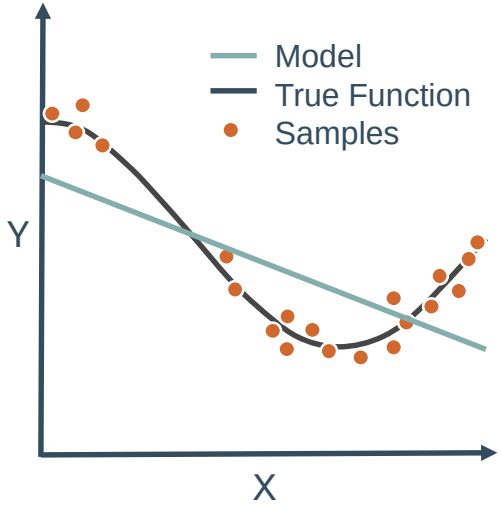
Polynomial Degree = 15



Good at Training
Poor at Predicting

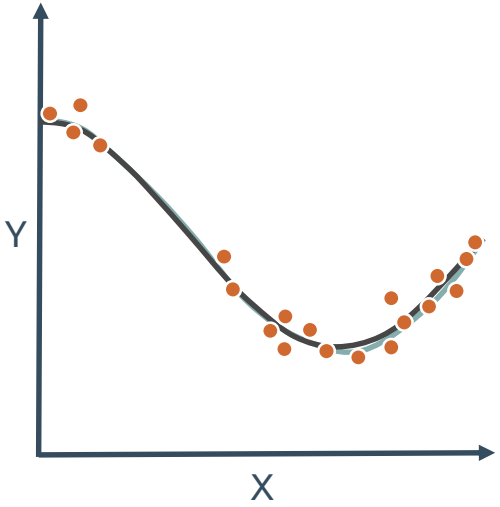
UNDERFITTING VS OVERFITTING

Polynomial Degree = 1



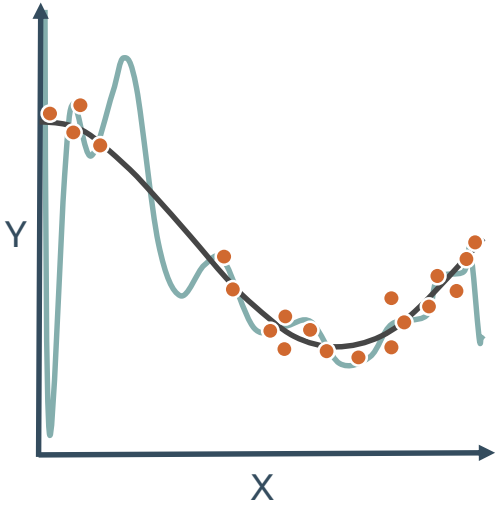
Underfitting

Polynomial Degree = 4



Just Right

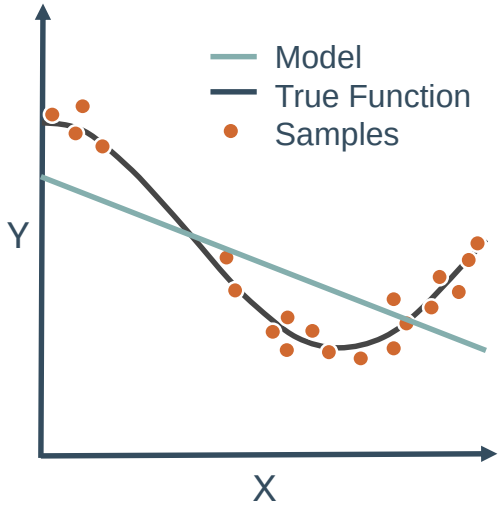
Polynomial Degree = 15



Overfitting

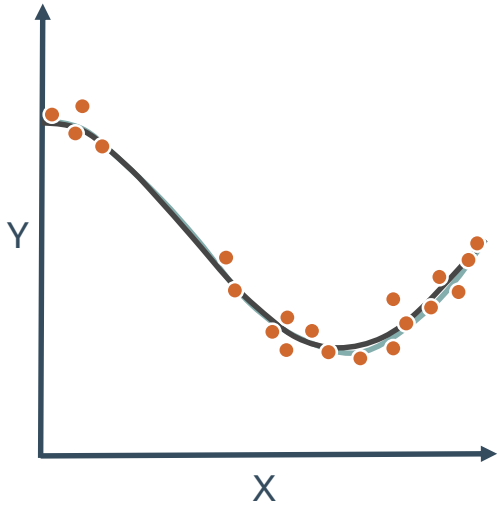
BIAS–VARIANCE TRADEOFF

Polynomial Degree = 1



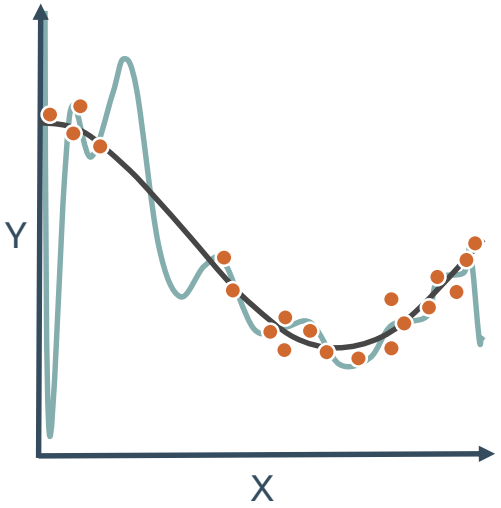
High Bias
Low Variance

Polynomial Degree = 4



Just Right

Polynomial Degree = 15



Low Bias
High Variance

TRAINING AND TEST SPLITS

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

TRAINING AND TEST SPLITS

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

TRAINING DATA

TEST DATA

USING TRAINING AND TEST DATA

TRAINING DATA

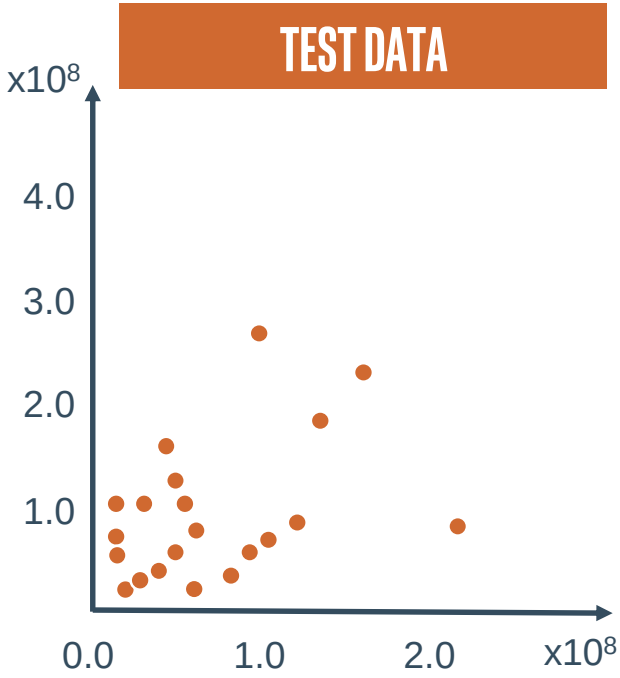
fit the model

TEST DATA

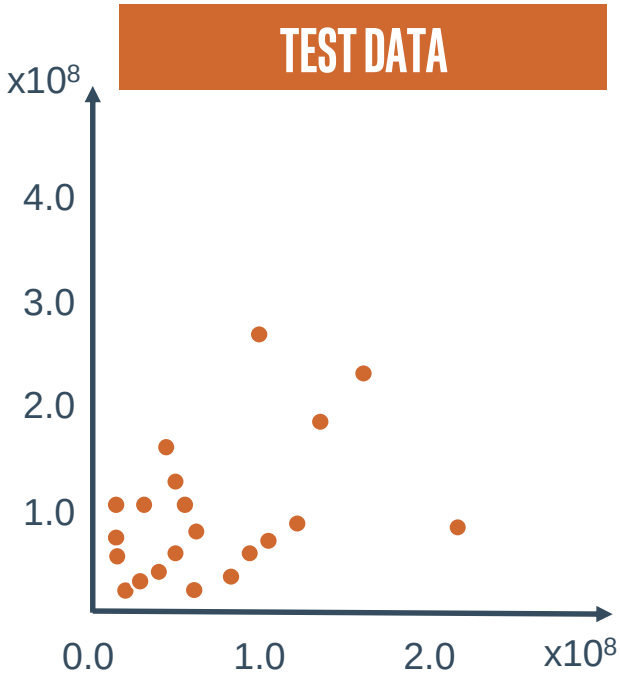
measure performance

- predict label with model
- compare with actual value
- measure error

USING TRAINING AND TEST DATA

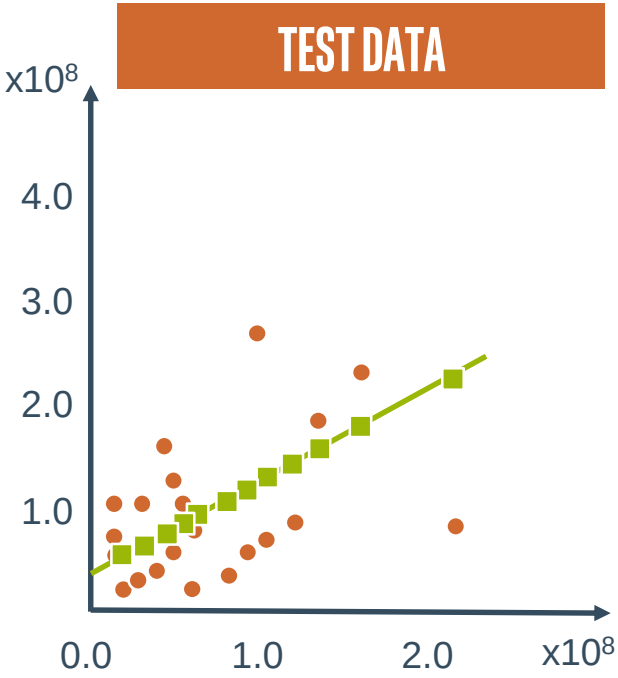


USING TRAINING AND TEST DATA



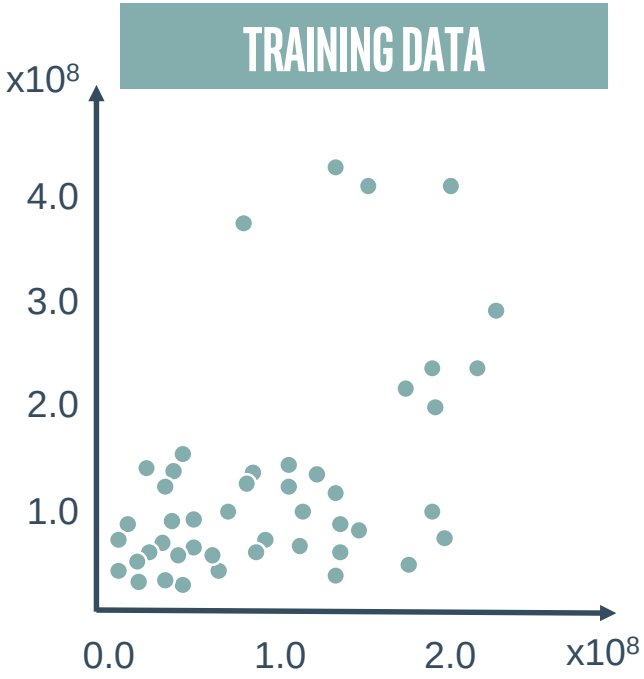
Fit the model

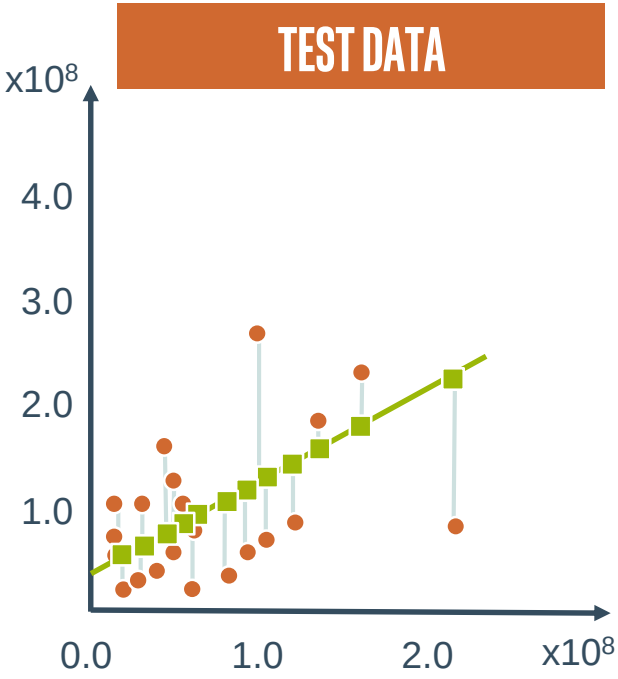
USING TRAINING AND TEST DATA



Make predictions

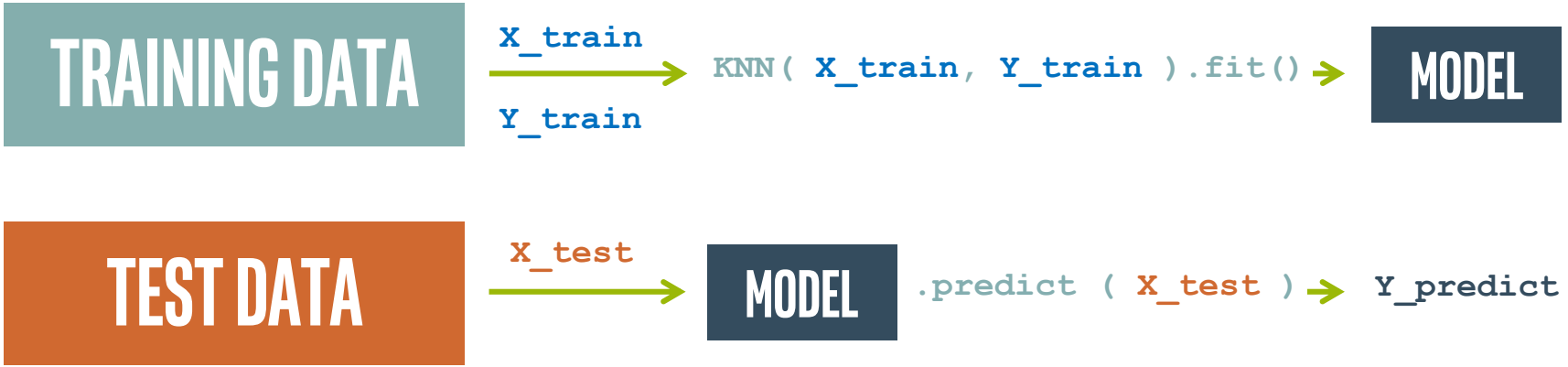
USING TRAINING AND TEST DATA



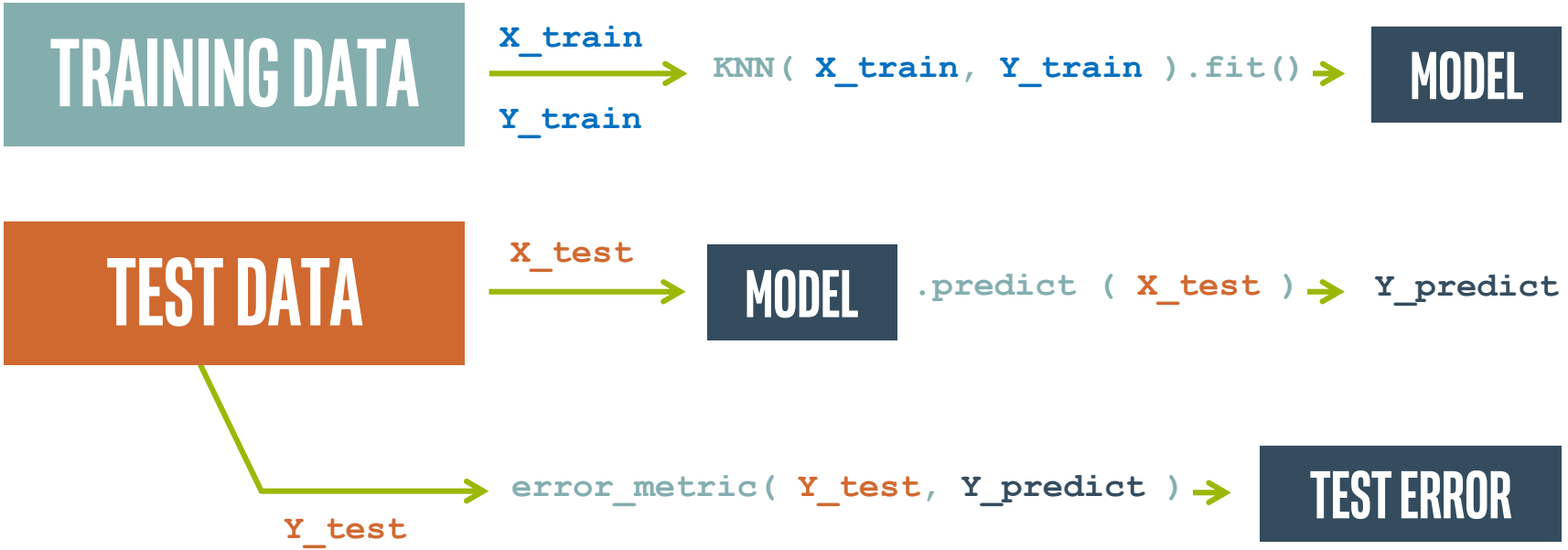


Measure error

FITTING TRAINING AND TEST DATA



FITTING TRAINING AND TEST DATA



TRAIN AND TEST SPLITTING: THE SYNTAX

Import the train and test split function

```
from sklearn.model_selection import train_test_split
```

TRAIN AND TEST SPLITTING: THE SYNTAX

Import the train and test split function

```
from sklearn.model_selection import train_test_split
```

Split the data and put 30% into the test set

```
train, test = train_test_split(data, test_size=0.3)
```

TRAIN AND TEST SPLITTING: THE SYNTAX

Import the train and test split function

```
from sklearn.model_selection import train_test_split
```

Split the data and put 30% into the test set

```
train, test = train_test_split(data, test_size=0.3)
```

Other method for splitting data:

```
from sklearn.model_selection import ShuffleSplit
```

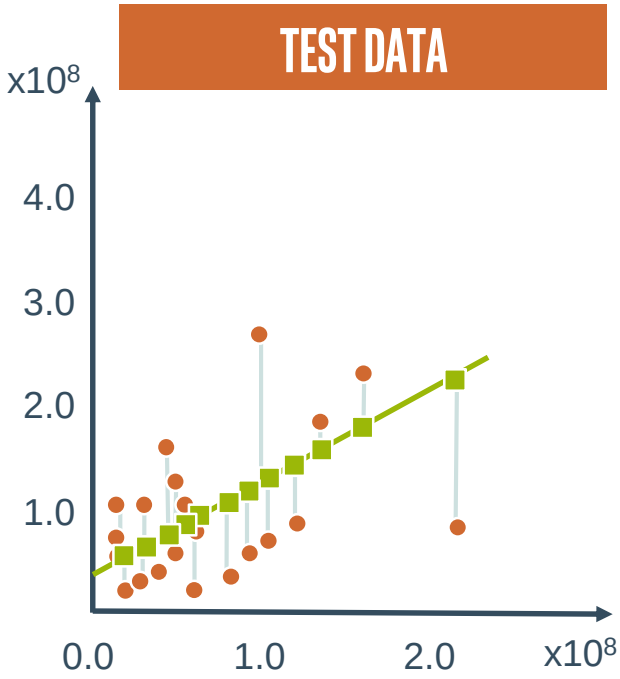
BEYOND A SINGLE TEST SET: CROSS VALIDATION

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

TRAINING DATA

VALIDATION DATA

BEYOND A SINGLE TEST SET: CROSS VALIDATION



Best model for <u>this</u> test set

BEYOND A SINGLE TEST SET: CROSS VALIDATION

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

TRAINING DATA 1

VALIDATION DATA 1

BEYOND A SINGLE TEST SET: CROSS VALIDATION

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

TRAINING DATA 2

VALIDATION DATA 2

BEYOND A SINGLE TEST SET: CROSS VALIDATION

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

VALIDATION DATA 3

TRAINING DATA 3

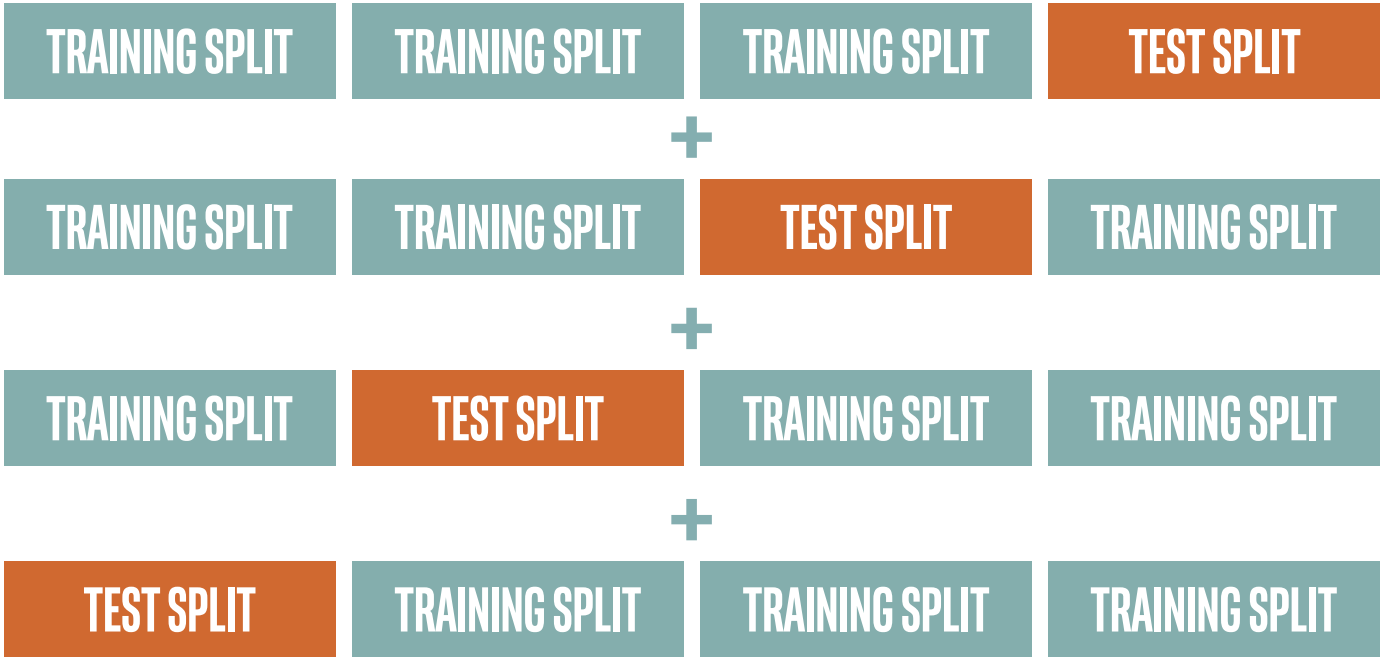
BEYOND A SINGLE TEST SET: CROSS VALIDATION

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

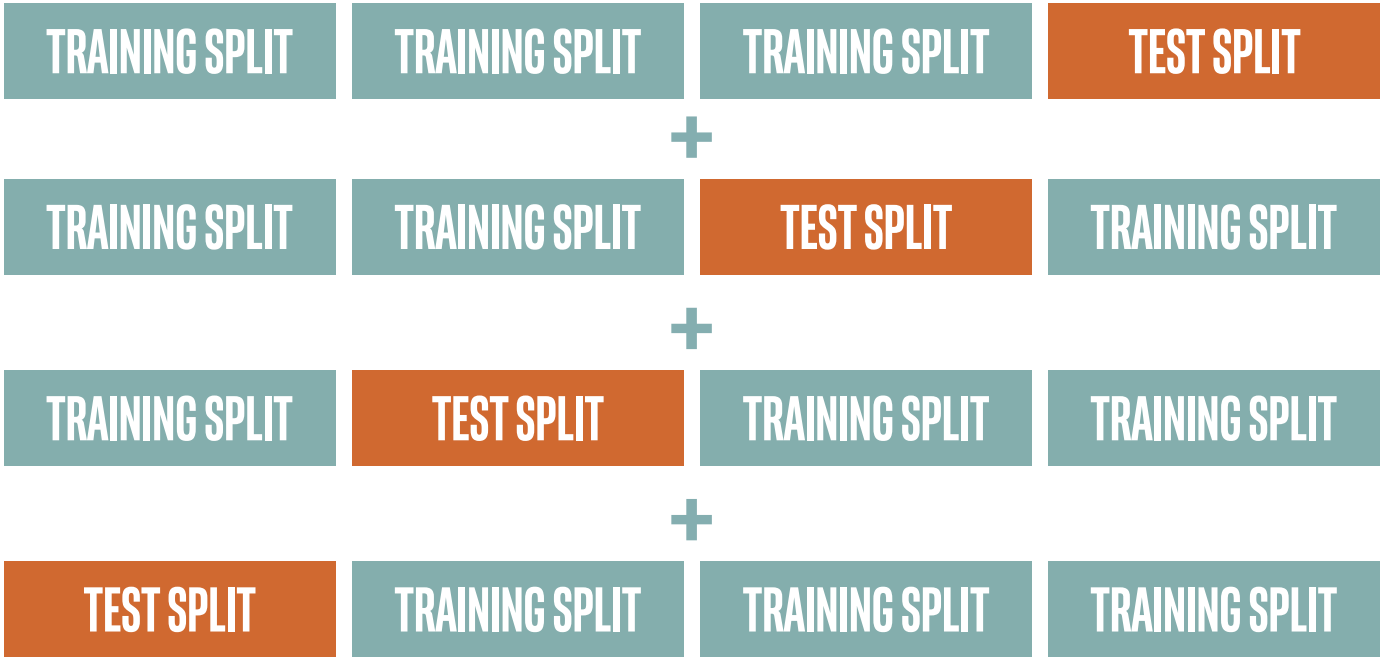
VALIDATION DATA 4

TRAINING DATA 4

BEYOND A SINGLE TEST SET: CROSS VALIDATION

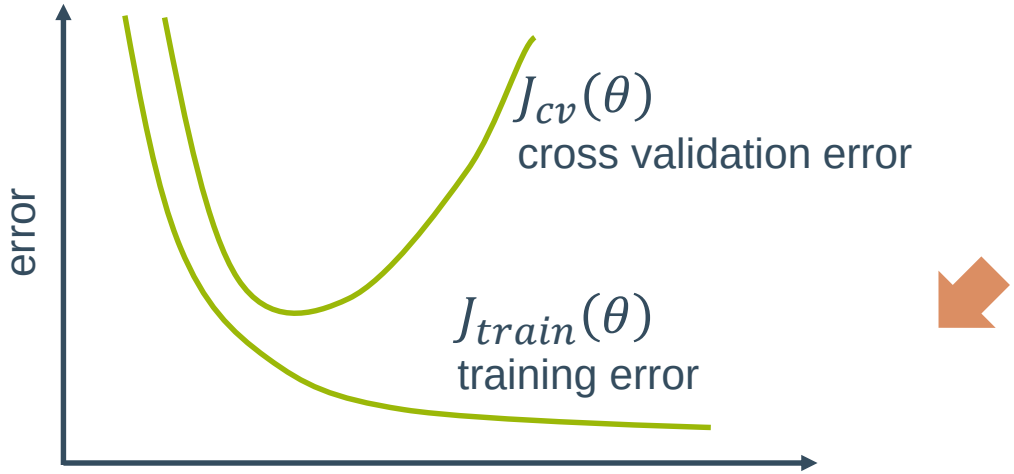


BEYOND A SINGLE TEST SET: CROSS VALIDATION

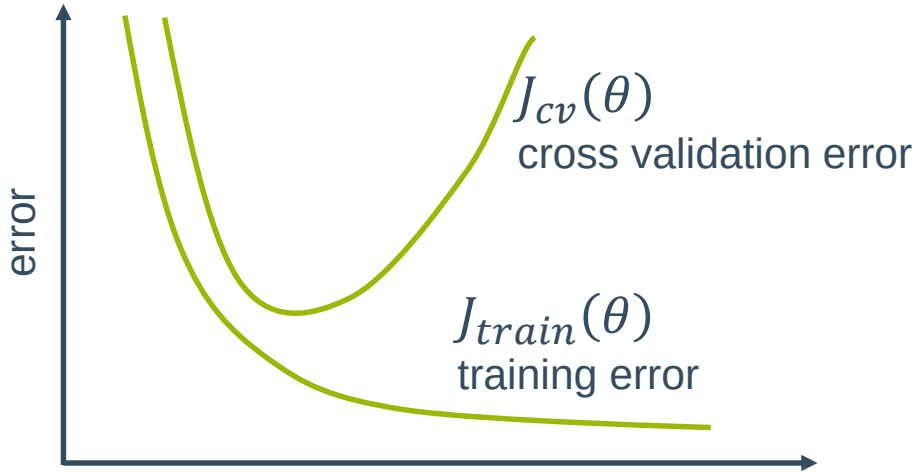


Average cross validation results

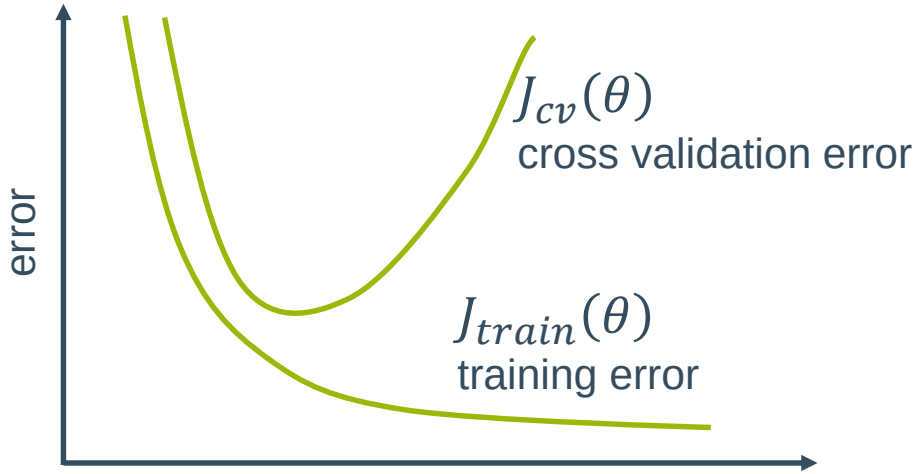
MODEL COMPLEXITY VS ERROR



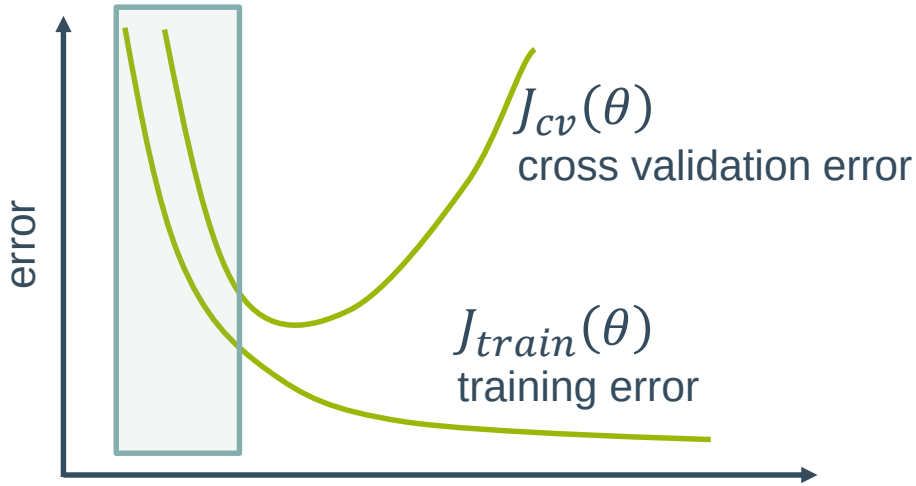
MODEL COMPLEXITY VS ERROR



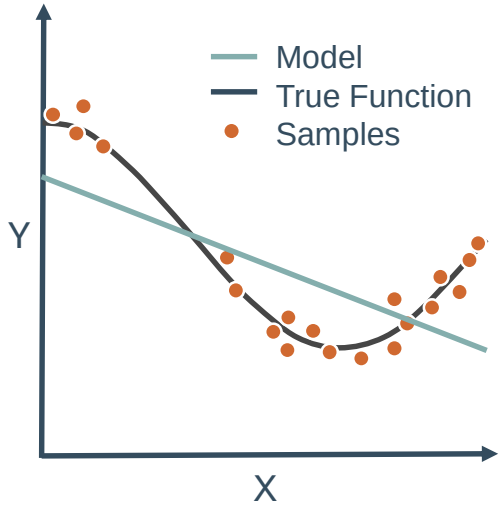
MODEL COMPLEXITY VS ERROR



MODEL COMPLEXITY VS ERROR

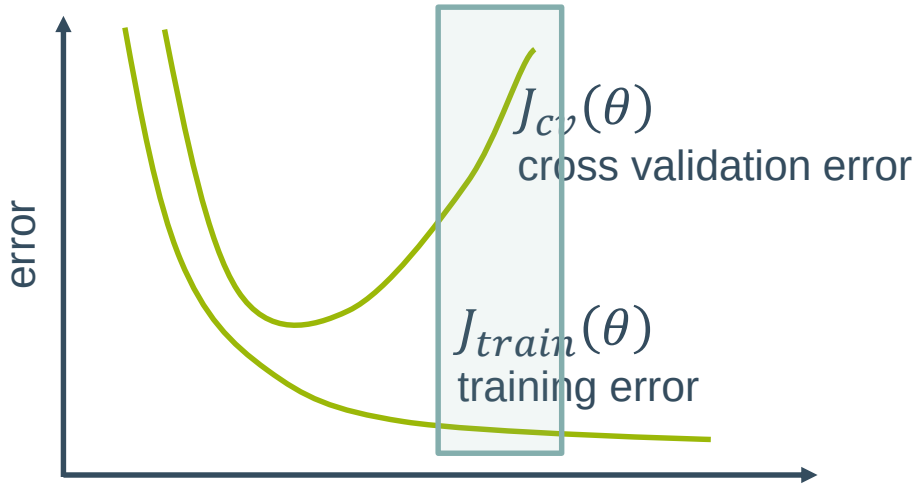


Polynomial Degree = 1

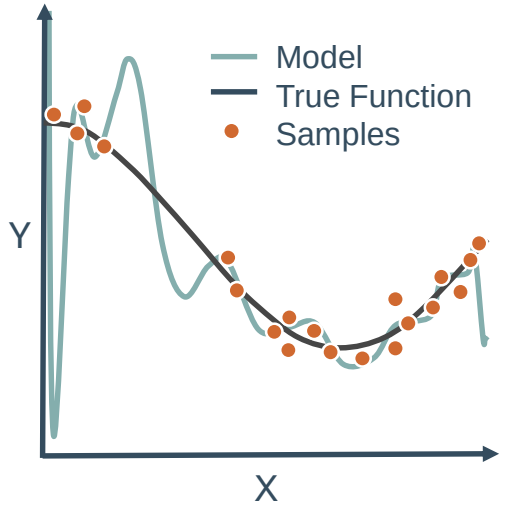


Underfitting: training and cross validation error are high

MODEL COMPLEXITY VS ERROR

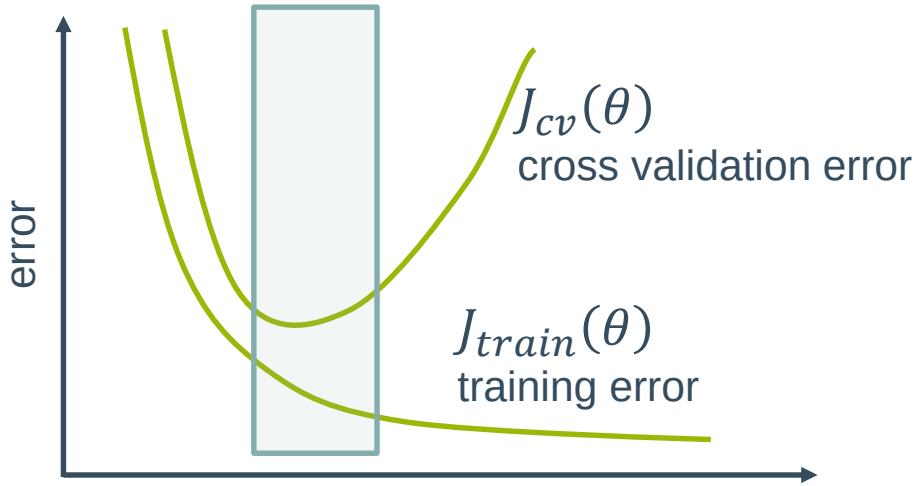


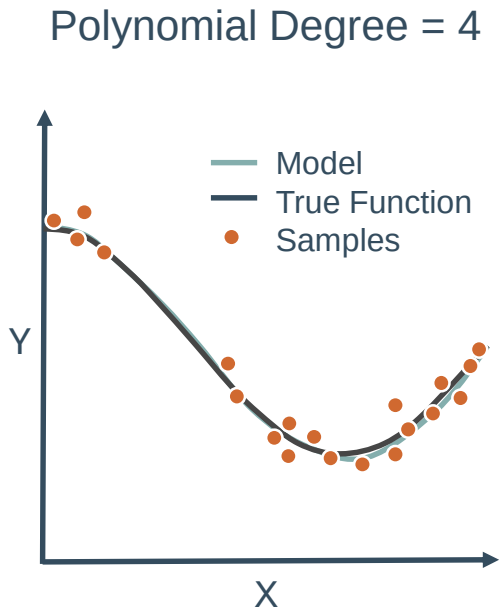
Polynomial Degree = 15



Overfitting: training error is low, cross validation is high

MODEL COMPLEXITY VS ERROR





Just right: training and cross validation errors are low

CROSS VALIDATION: THE SYNTAX

Import the train and test split function

```
from sklearn.model_selection import cross_val_score
```

CROSS VALIDATION: THE SYNTAX

Import the train and test split function

```
from sklearn.model_selection import cross_val_score
```

Perform cross-validation with a given model

```
cross_val = cross_val_score(KNN, X_data, y_data, cv=4,
                            scoring='neg_mean_squared_error')
```

CROSS VALIDATION: THE SYNTAX

Import the train and test split function

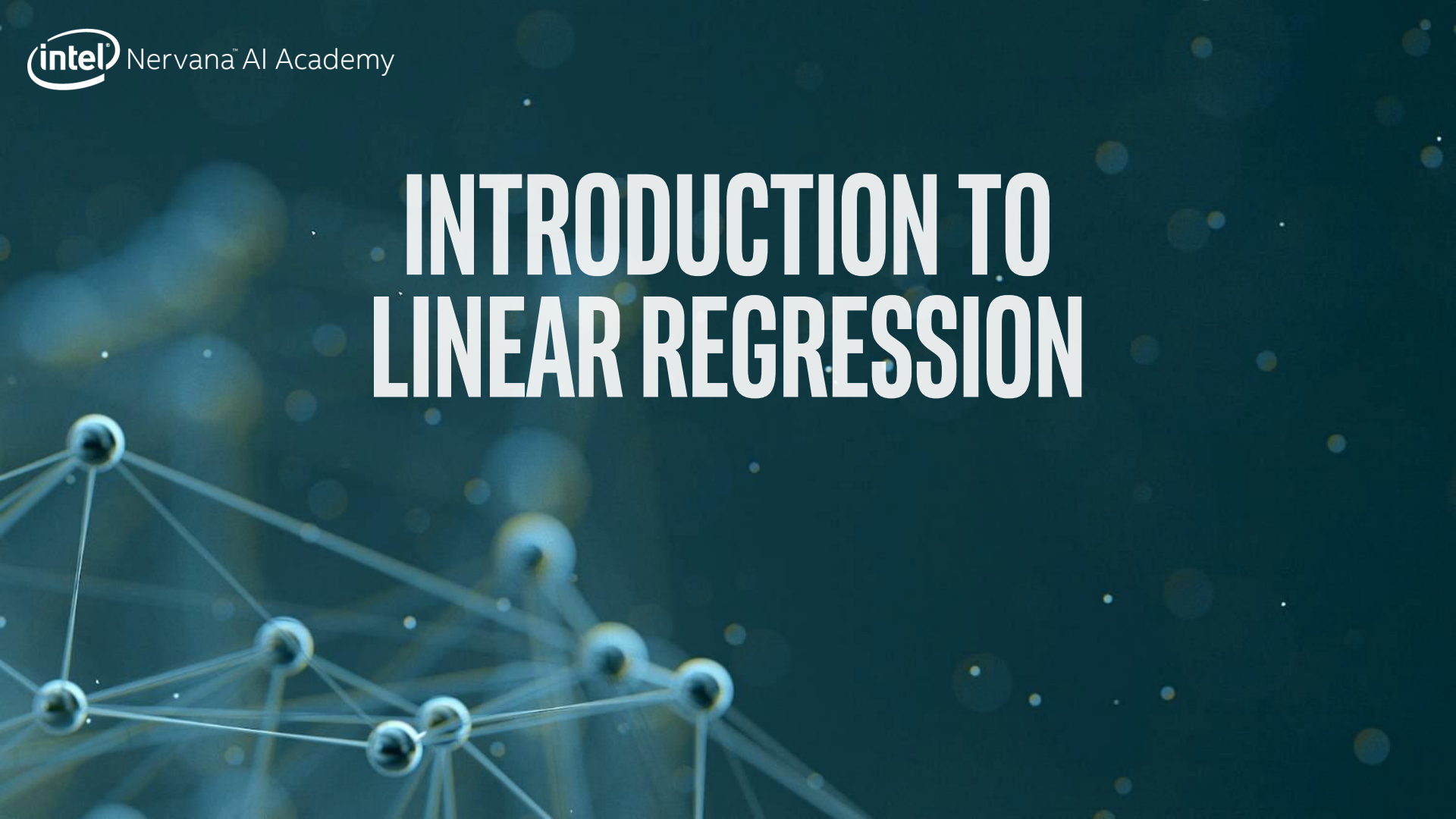
```
from sklearn.model_selection import cross_val_score
```

Perform cross-validation with a given model

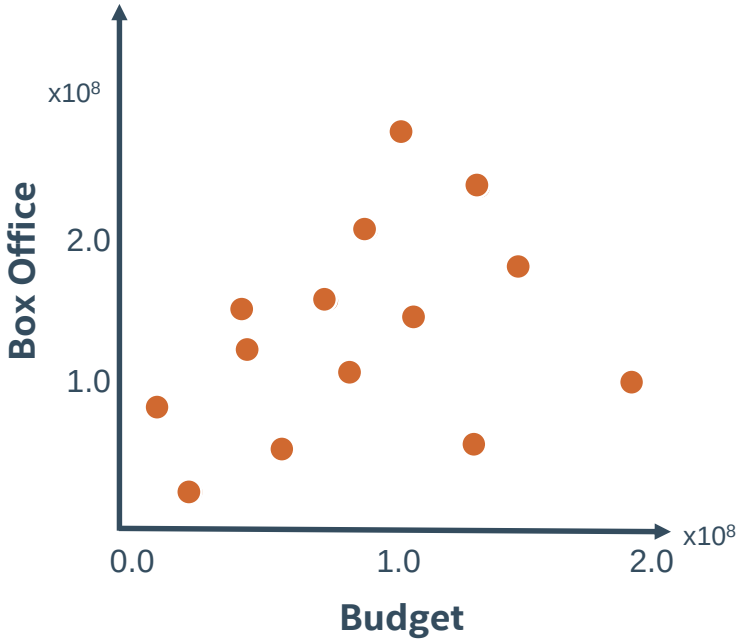
```
cross_val = cross_val_score(KNN, X_data, y_data, cv=4,
                            scoring='neg_mean_squared_error')
```

Other methods for cross validation:

```
from sklearn.model_selection import KFold, StratifiedKFold
```

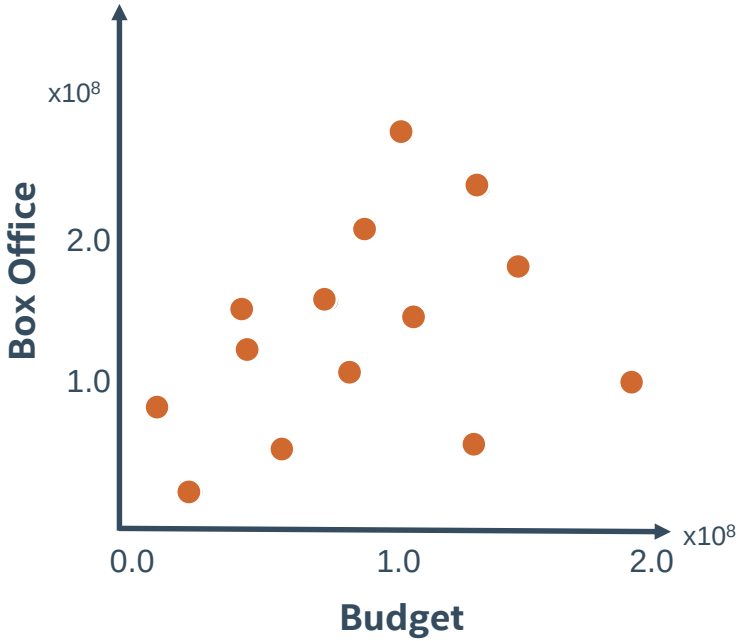

intel Nervana AI Academy
INTRODUCTION TO
LINEAR REGRESSION

INTRODUCTION TO LINEAR REGRESSION



$$y_\beta(x) = \beta_0 + \beta_1 x$$

INTRODUCTION TO LINEAR REGRESSION



$$y_\beta(x) = \beta_0 + \beta_1 x$$

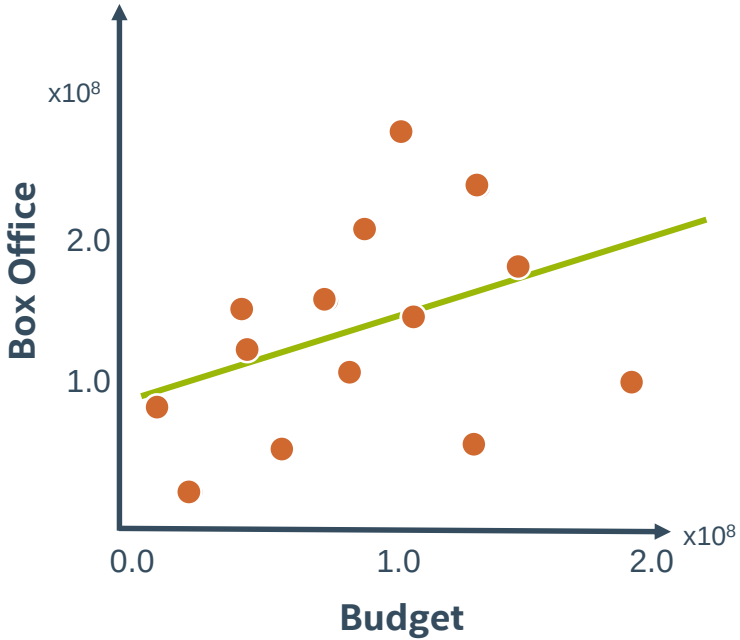
box office revenue

coefficient 0

coefficient 1

movie budget

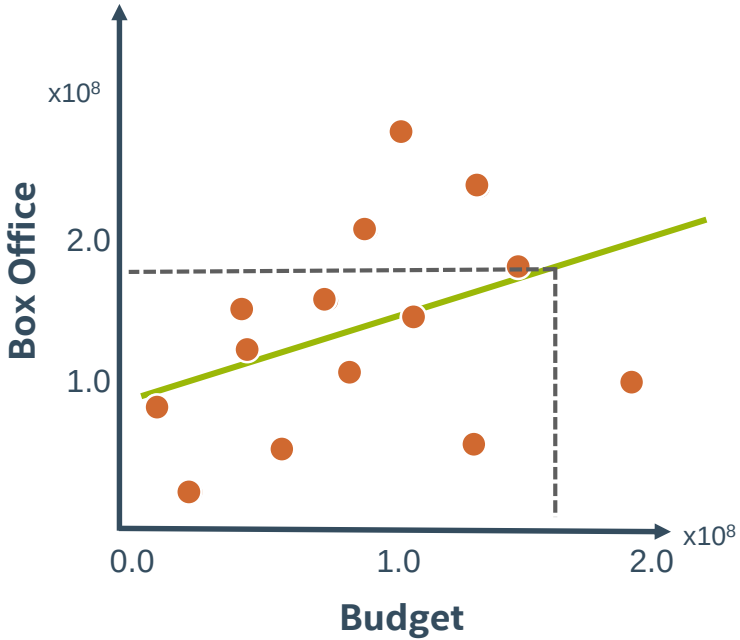
INTRODUCTION TO LINEAR REGRESSION



$$y_\beta(x) = \beta_0 + \beta_1 x$$

β_0= 80 million, β_1= 0.6

PREDICTING FROM LINEAR REGRESSION

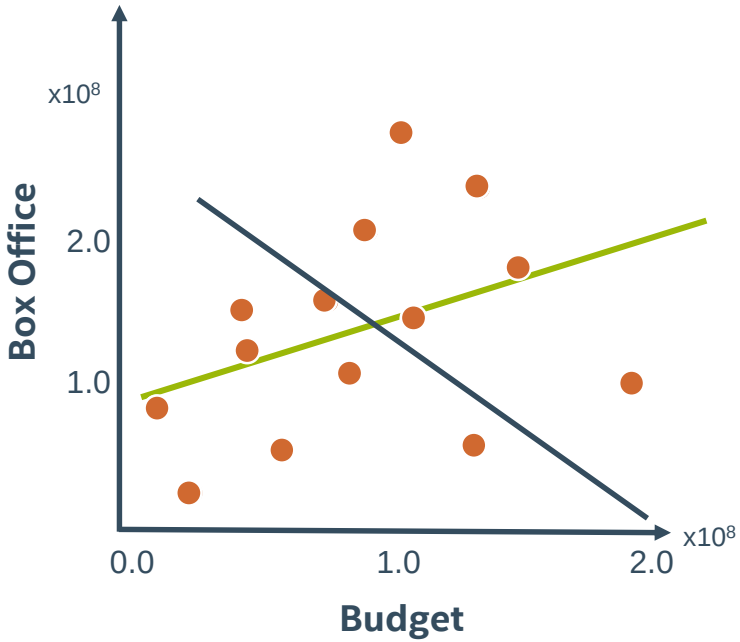


$$y_\beta(x) = \beta_0 + \beta_1 x$$

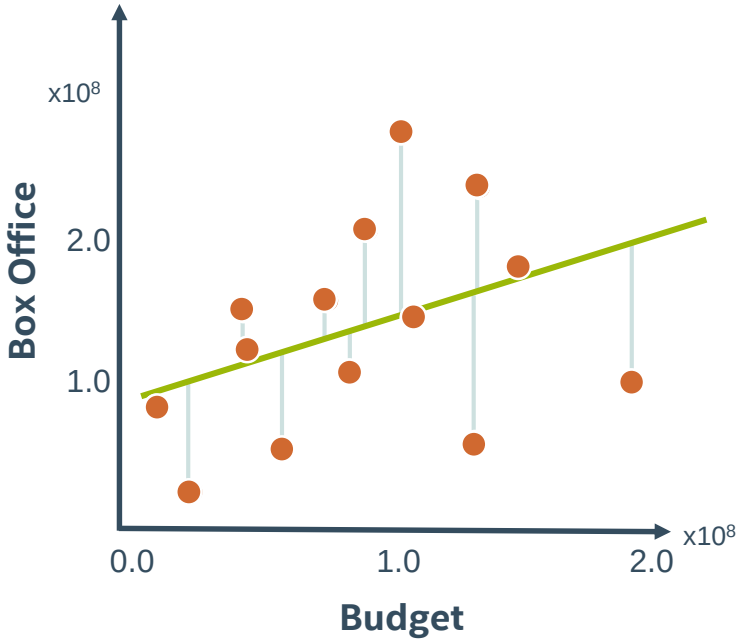
β_0= 80 million, β_1= 0.6

Predict 175 Million Gross for 160 Million Budget

WHICH MODEL FITS THE BEST?



CALCULATING THE RESIDUALS

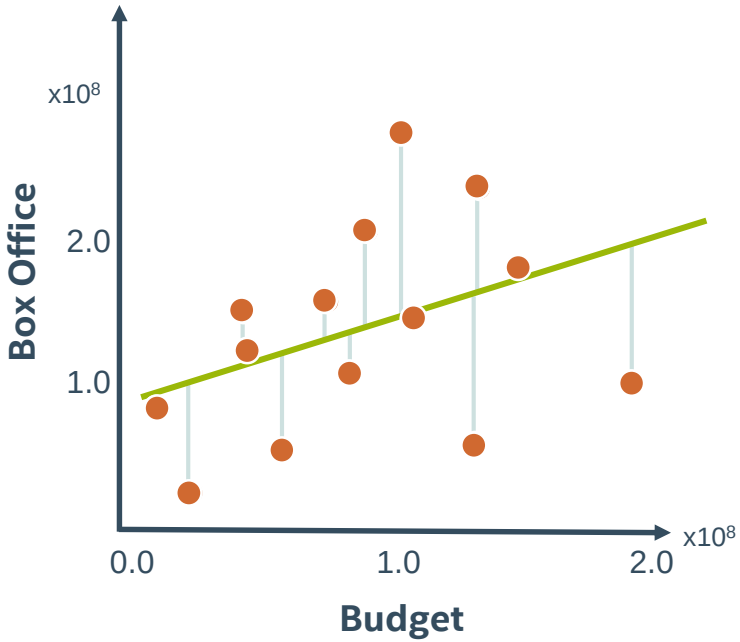


$$y_\beta\left(x_{obs}^{(i)}\right) - y_{obs}^{(i)}$$

Predicted value

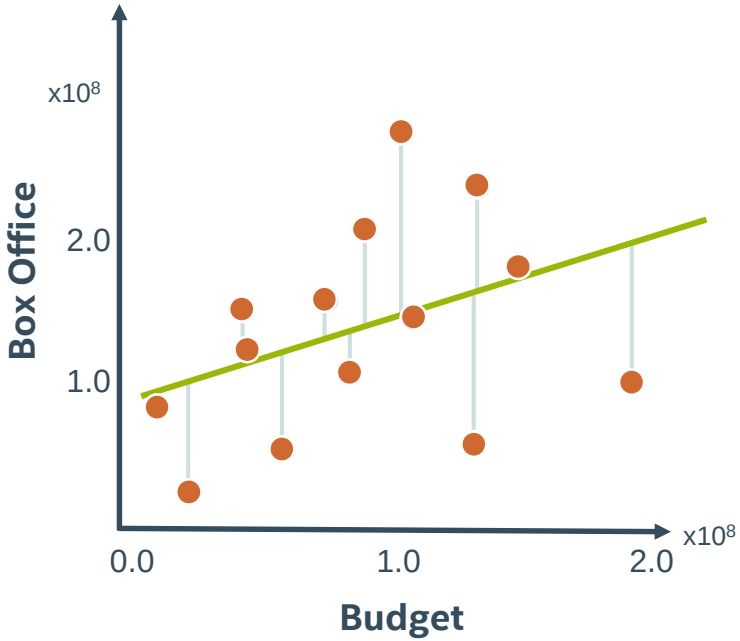
Observed value

CALCULATING THE RESIDUALS



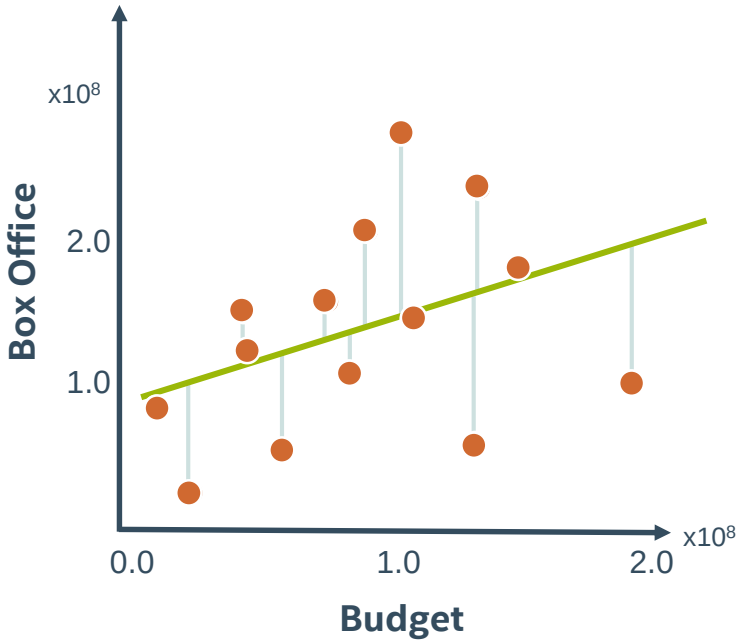
$$\left(\beta_0 + \beta_1 x_{obs}^{(i)}\right) - y_{obs}^{(i)}$$

MEAN SQUARED ERROR



$$\frac{1}{m}\sum_{i=1}^{m}\left(\left(\beta_0 + \beta_1 x_{obs}^{(i)}\right) - y_{obs}^{(i)}\right)^2$$

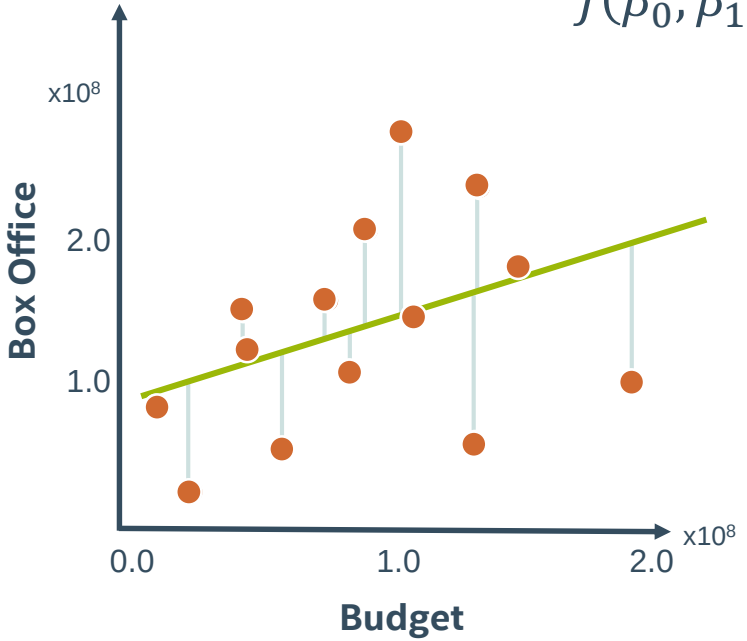
MINIMUM MEAN SQUARED ERROR



$$\min_{\beta_0,\beta_1} \frac{1}{m} \sum_{i=1}^{m} \left(\left(\beta_0 + \beta_1 x_{obs}^{(i)} \right) - y_{obs}^{(i)} \right)^2$$

COST FUNCTION

$$J(\beta_0, \beta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(\left(\beta_0 + \beta_1 x_{obs}^{(i)} \right) - y_{obs}^{(i)} \right)^2$$



MODELLING BEST PRACTICE

- Use cost function to fit model
- Develop multiple models
- Compare results and choose best one

OTHER MODEL METRICS

Sum of Squared Error (SSE):

$$\sum_{i=1}^{m} \left(y_\beta(x^{(i)}) - y_{obs}^{(i)} \right)^2$$

OTHER MEASURES OF ERROR

Sum of Squared Error (SSE):

$$\sum_{i=1}^{m} \left(y_\beta(x^{(i)}) - y_{obs}^{(i)} \right)^2$$

Total Sum of Squares (TSS):

$$\sum_{i=1}^{m} \left(\overline{y_{obs}} - y_{obs}^{(i)} \right)^2$$

OTHER MEASURES OF ERROR

Sum of Squared Error (SSE):

$$\sum_{i=1}^{m} \left(y_\beta(x^{(i)}) - y_{obs}^{(i)} \right)^2$$

Total Sum of Squares (TSS):

$$\sum_{i=1}^{m} \left(\overline{y_{obs}} - y_{obs}^{(i)} \right)^2$$

Correlation Coefficient (R2):

$$1 - \frac{SSE}{TSS}$$

COMPARING LINEAR REGRESSION AND KNN

LINEAR REGRESSION

- **Fitting involves minimizing cost function** (slow)

K NEAREST NEIGHBORS

- **Fitting involves storing training data** (fast)

COMPARING LINEAR REGRESSION AND KNN

LINEAR REGRESSION

- **Fitting involves minimizing cost function** (slow)
- **Model has few parameters** (memory efficient)

K NEAREST NEIGHBORS

- **Fitting involves storing training data** (fast)
- **Model has many parameters** (memory intensive)

COMPARING LINEAR REGRESSION AND KNN

LINEAR REGRESSION

- **Fitting involves minimizing cost function** (slow)
- **Model has few parameters** (memory efficient)
- **Prediction involves calculation** (fast)

K NEAREST NEIGHBORS

- **Fitting involves storing training data** (fast)
- **Model has many parameters** (memory intensive)
- **Prediction involves finding closest neighbors** (slow)

LINEAR REGRESSION: THE SYNTAX

Import the class containing the regression method

```
from sklearn.linear_model import LinearRegression
```

LINEAR REGRESSION: THE SYNTAX

Import the class containing the regression method

```
from sklearn.linear_model import LinearRegression
```

Create an instance of the class

```
LR = LinearRegression()
```

LINEAR REGRESSION: THE SYNTAX

Import the class containing the regression method

```
from sklearn.linear_model import LinearRegression
```

Create an instance of the class

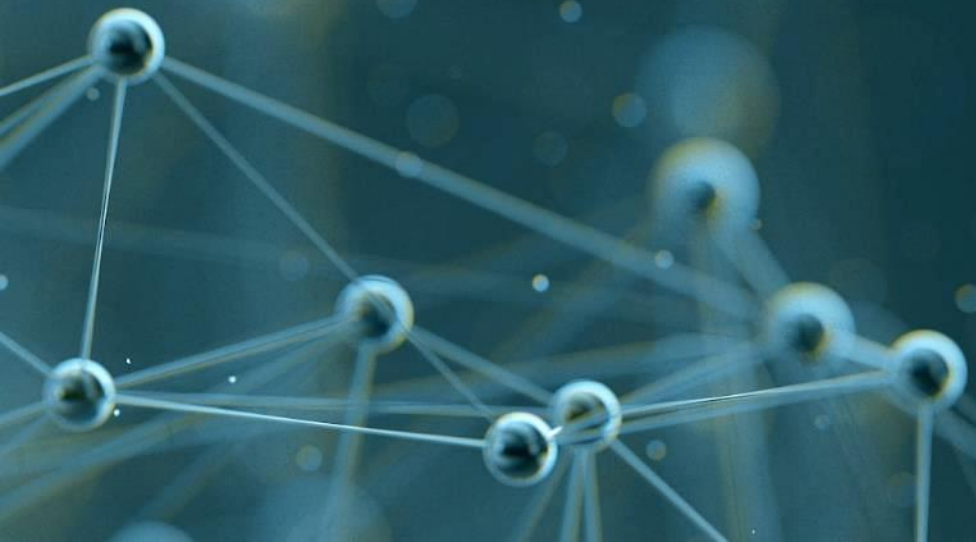
```
LR = LinearRegression()
```

Fit the instance on the data and then predict the expected value

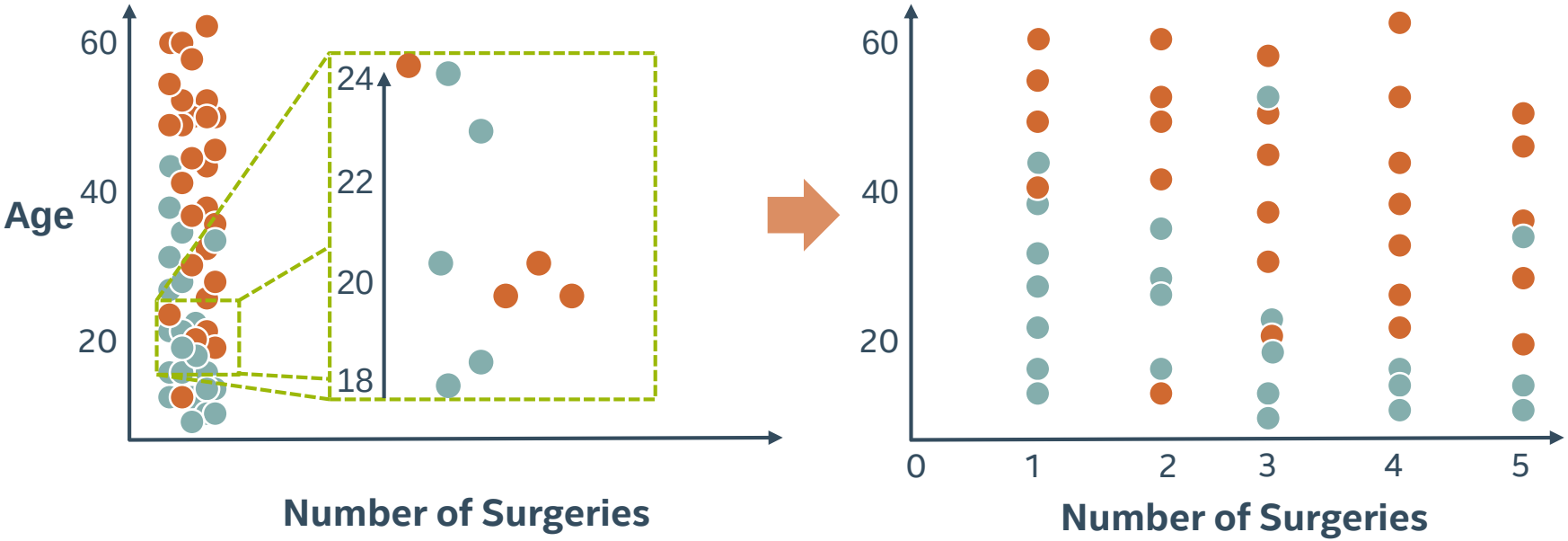
```
LR = LR.fit(X_train, y_train)

y_predict = LR.predict(X_test)
```

intel Nervana™ AI Academy

ADVANCED LINEAR REGRESSION

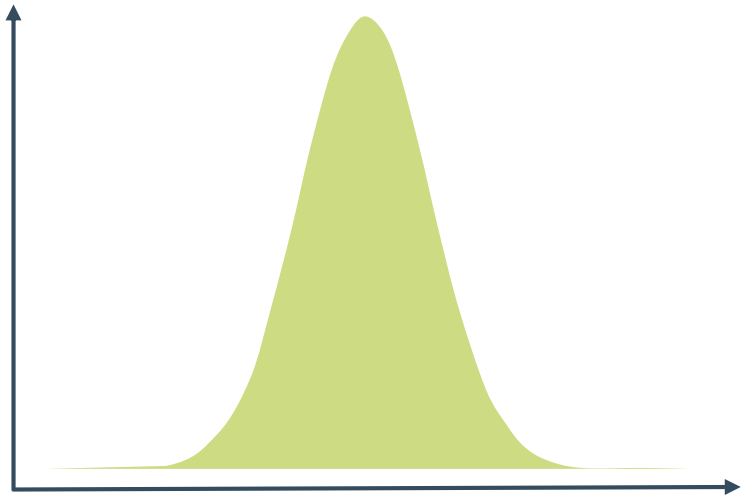


SCALING IS A TYPE OF FEATURE TRANSFORMATION



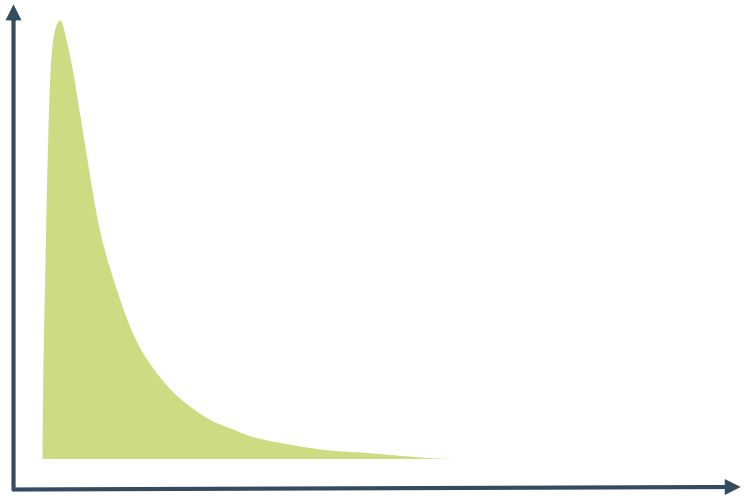
TRANSFORMATION OF DATA DISTRIBUTIONS

- Predictions from linear regression models assume residuals are normally distributed

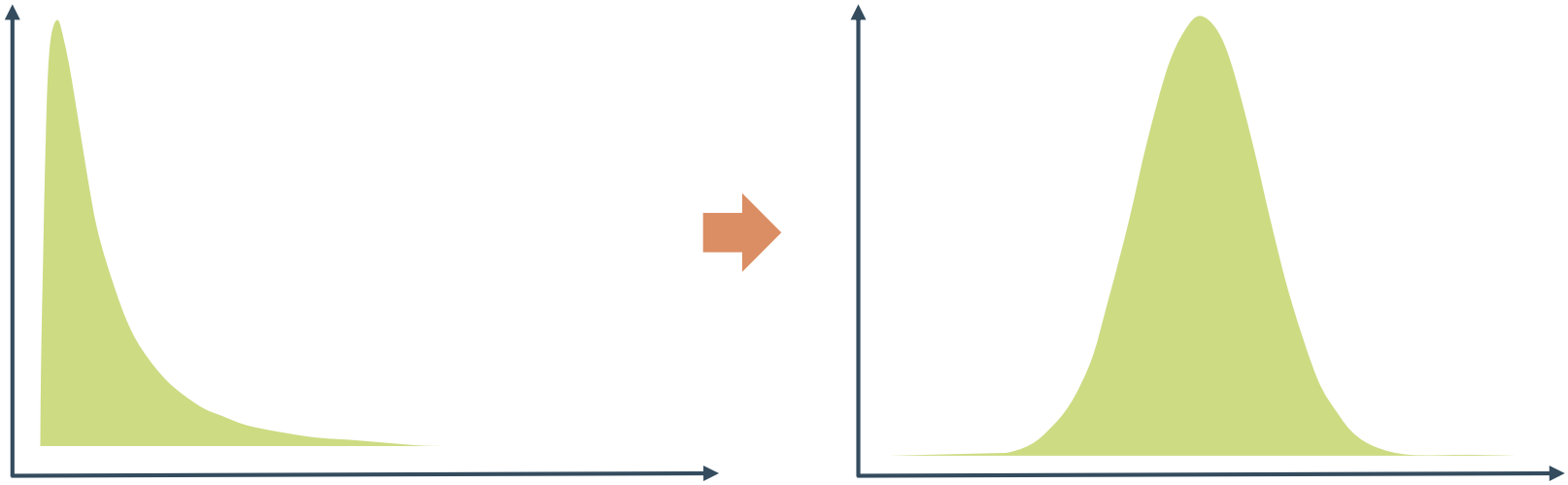


TRANSFORMATION OF DATA DISTRIBUTIONS

- Predictions from linear regression models assume residuals are normally distributed
- Features and predicted data are often skewed



TRANSFORMATION OF DATA DISTRIBUTIONS

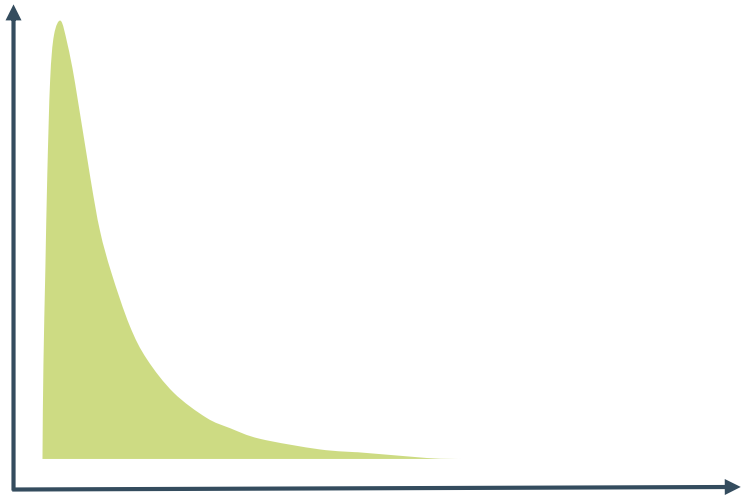


from numpy import log, log1p

from scipy.stats import boxcox

TRANSFORMATION OF DATA DISTRIBUTIONS

- Predictions from linear regression models assume residuals are normally distributed
- Features and predicted data are often skewed
- Data transformations can solve this issue



FEATURE TYPES

FEATURE TYPE	TRANSFORMATION

- **Continuous:** numerical values

FEATURE TYPES

FEATURE TYPE	TRANSFORMATION
▪ **Continuous:** numerical values	▪ Standard Scaling, Min-Max Scaling

FEATURE TYPES

FEATURE TYPE	TRANSFORMATION
Continuous: numerical values	Standard Scaling, Min-Max Scaling
Nominal: categorical, unordered features (True or False)	One-hot encoding (0, 1)

from sklearn.preprocessing import LabelEncoder, LabelBinarizer, OneHotEncoder

FEATURE TYPES

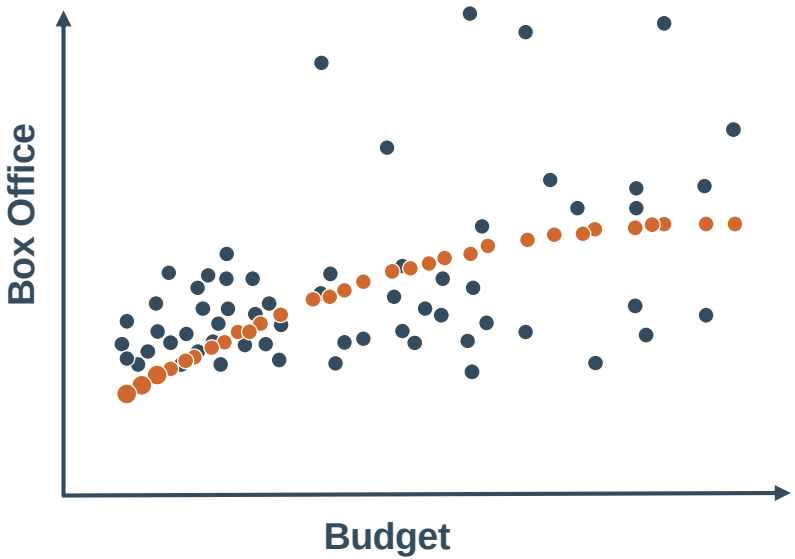
FEATURE TYPE	TRANSFORMATION
Continuous: numerical values	Standard Scaling, Min-Max Scaling
Nominal: categorical, unordered features (True or False)	One-hot encoding (0, 1)
Ordinal: categorical, ordered features (movie ratings)	Ordinal encoding (0, 1, 2, 3)

```
from sklearn.feature_extraction import DictVectorizer

from pandas import get_dummies
```

ADDITION OF POLYNOMIAL FEATURES

- Capture higher order features of data by adding polynomial features

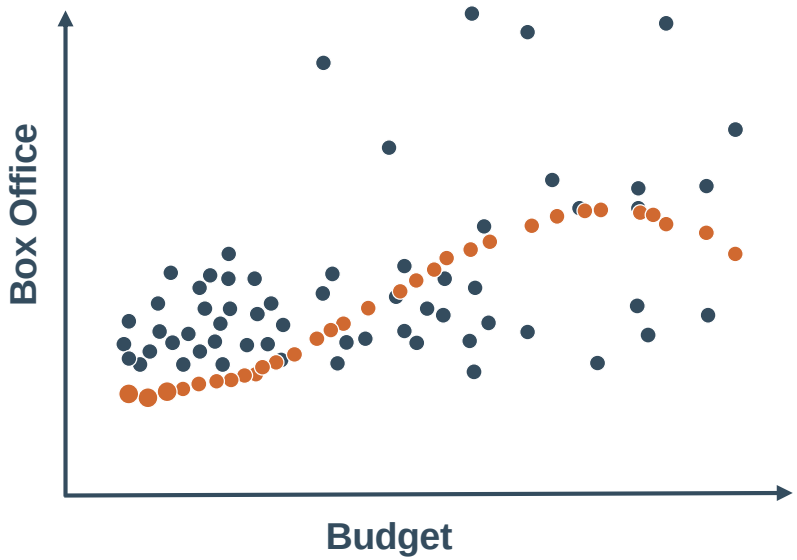
$$y_\beta(x) = \beta_0 + \beta_1 x + \beta_2 x^2$$



ADDITION OF POLYNOMIAL FEATURES

- Capture higher order features of data by adding polynomial features
- "Linear regression" means linear combinations of features

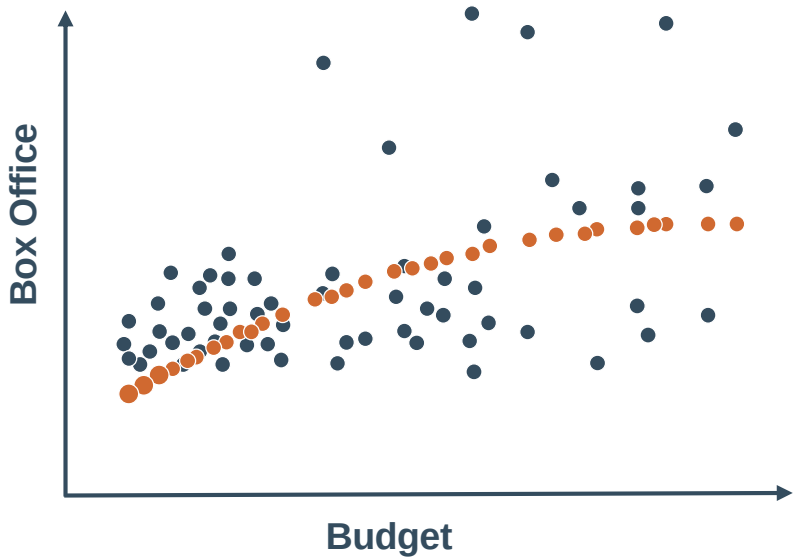
$$y_\beta(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$



ADDITION OF POLYNOMIAL FEATURES

- Capture higher order features of data by adding polynomial features
- "Linear regression" means linear combinations of features

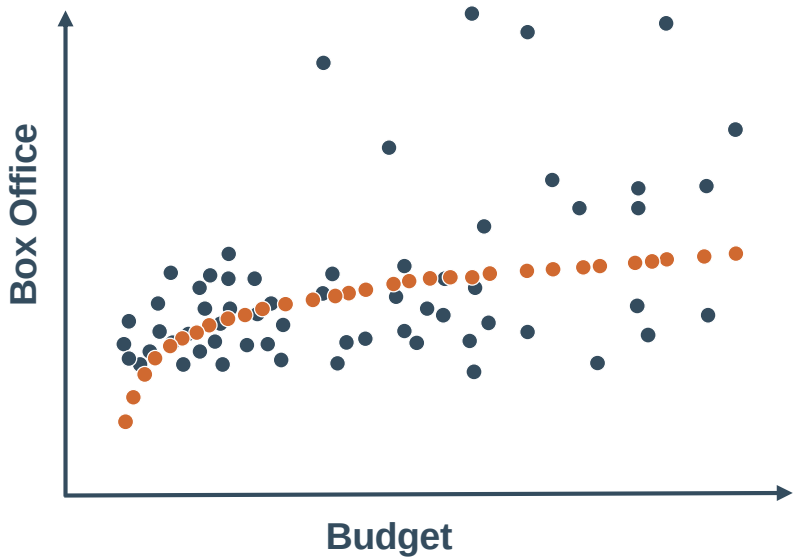
$$y_\beta(x) = \beta_0 + \beta_1 x + \beta_2 x^2$$



ADDITION OF POLYNOMIAL FEATURES

- Capture higher order features of data by adding polynomial features
- "Linear regression" means linear combinations of features

$$y_\beta(x) = \beta_0 + \beta_1 \log(x)$$



ADDITION OF POLYNOMIAL FEATURES

- Can also include variable interactions

$$y_\beta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$$

ADDITION OF POLYNOMIAL FEATURES

- Can also include variable interactions

$$y_\beta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$$

- How is the correct functional form chosen?

Check relationship of each variable or with outcome

POLYNOMIAL FEATURES: THE SYNTAX

Import the class containing the transformation method

```
from sklearn.preprocessing import PolynomialFeatures
```

POLYNOMIAL FEATURES: THE SYNTAX

Import the class containing the transformation method

```
from sklearn.preprocessing import PolynomialFeatures
```

Create an instance of the class

```
polyFeat = PolynomialFeatures(degree=2)
```

POLYNOMIAL FEATURES: THE SYNTAX

Import the class containing the transformation method

```
from sklearn.preprocessing import PolynomialFeatures
```

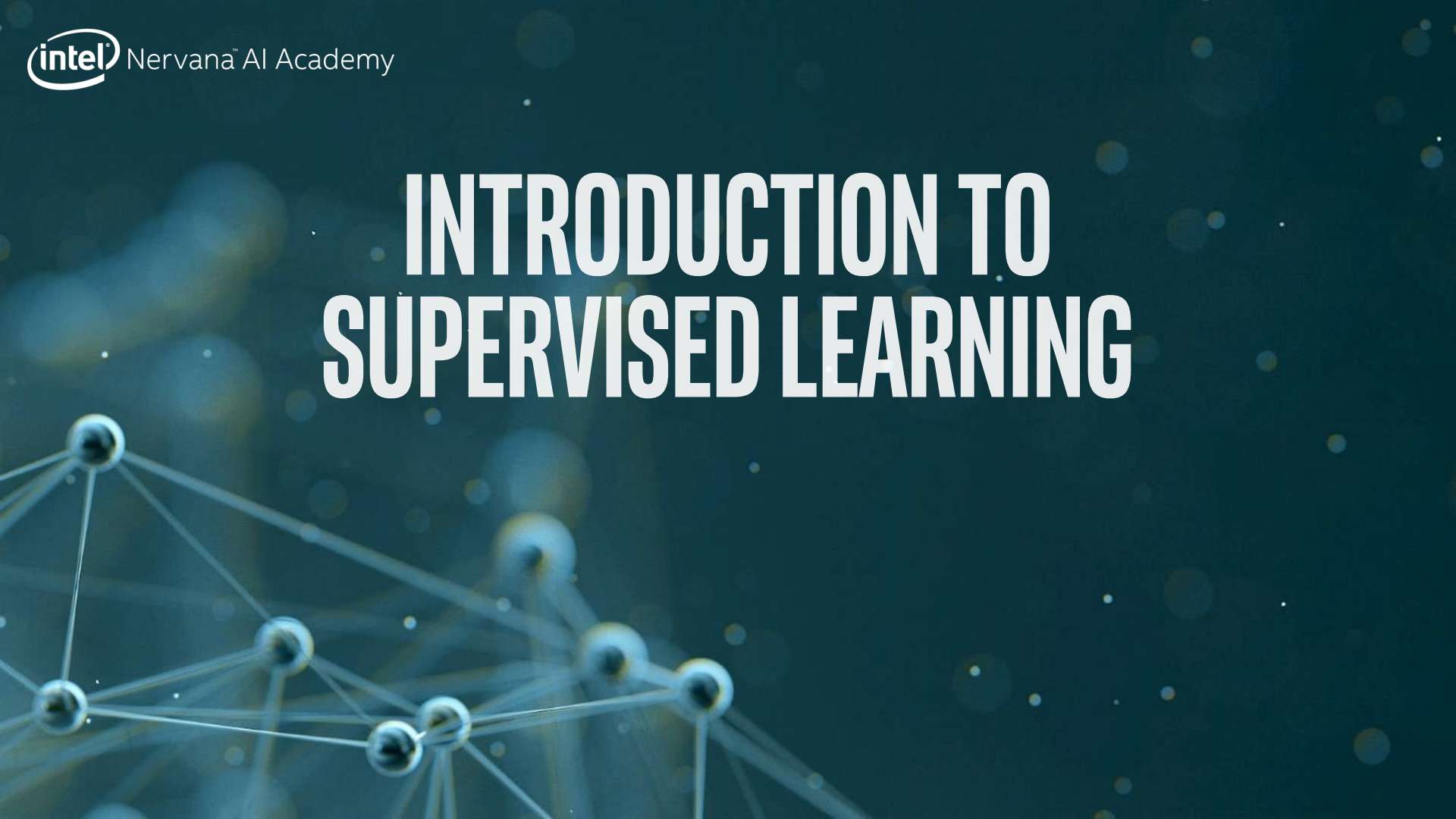
Create an instance of the class

```
polyFeat = PolynomialFeatures(degree=2)
```

Create the polynomial features and then transform the data

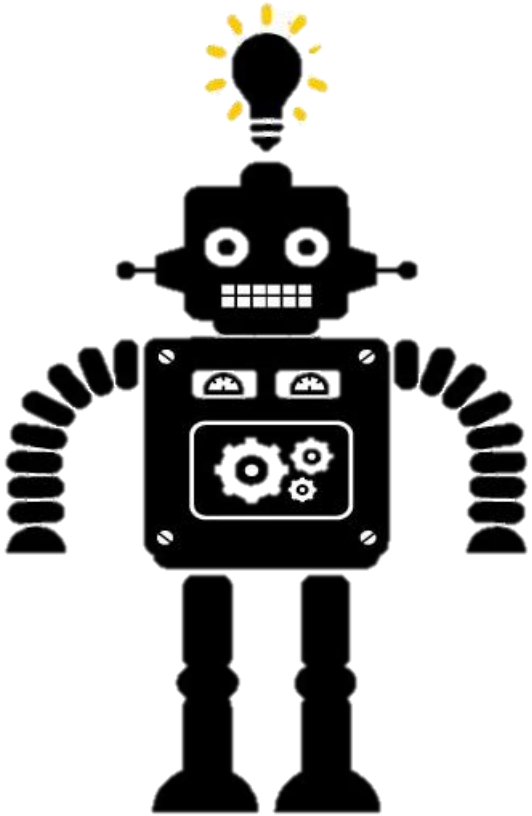
```
polyFeat = polyFeat.fit(X_data, y_data)

x_poly = polyFeat.transform(X_data)
```

intel
Nervana™ AI Academy


intel Nervana AI Academy
INTRODUCTION TO
SUPERVISED LEARNING

WHAT IS MACHINE LEARNING?

Machine learning allows computers to learn and infer from data.



MACHINE LEARNING IN OUR DAILY LIVES

SPAM FILTERING

MACHINE LEARNING IN OUR DAILY LIVES

SPAM FILTERING

WEB SEARCH

MACHINE LEARNING IN OUR DAILY LIVES

SPAM FILTERING

WEB SEARCH

POSTAL MAIL ROUTING

MACHINE LEARNING IN OUR DAILY LIVES

SPAM FILTERING

WEB SEARCH

POSTAL MAIL ROUTING

FRAUD DETECTION

MOVIE RECOMMENDATIONS

VEHICLE DRIVER ASSISTANCE

WEB ADVERTISEMENTS

SOCIAL NETWORKS

SPEECH RECOGNITION

TYPES OF MACHINE LEARNING

SUPERVISED

Data points have known outcome

TYPES OF MACHINE LEARNING

SUPERVISED — Data points have known outcome

UNSUPERVISED — Data points have unknown outcome

TYPES OF MACHINE LEARNING

SUPERVISED — Data points have known outcome

UNSUPERVISED — Data points have unknown outcome

TYPES OF SUPERVISED LEARNING

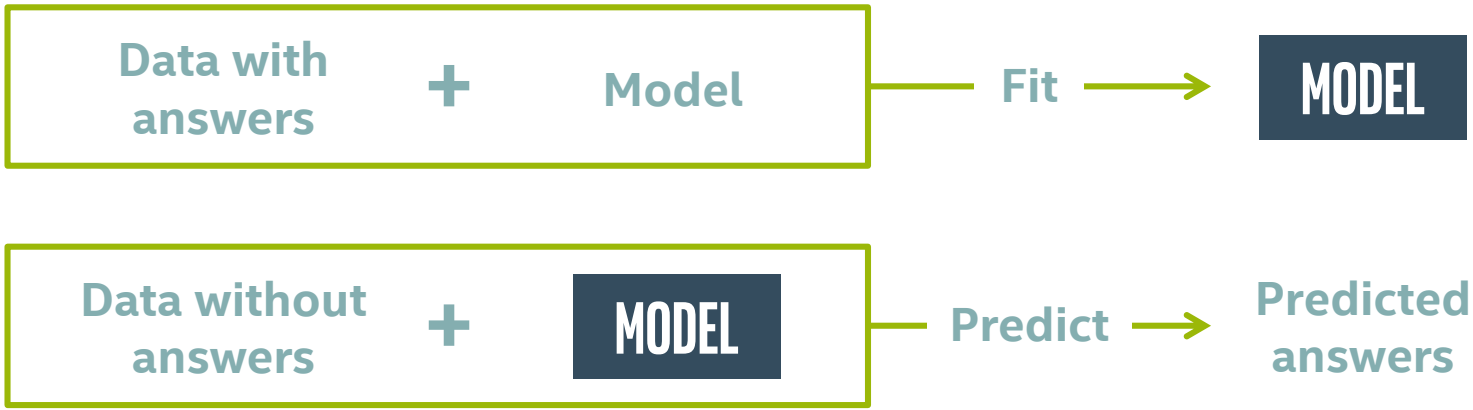
REGRESSION

Outcome is continuous (numerical)

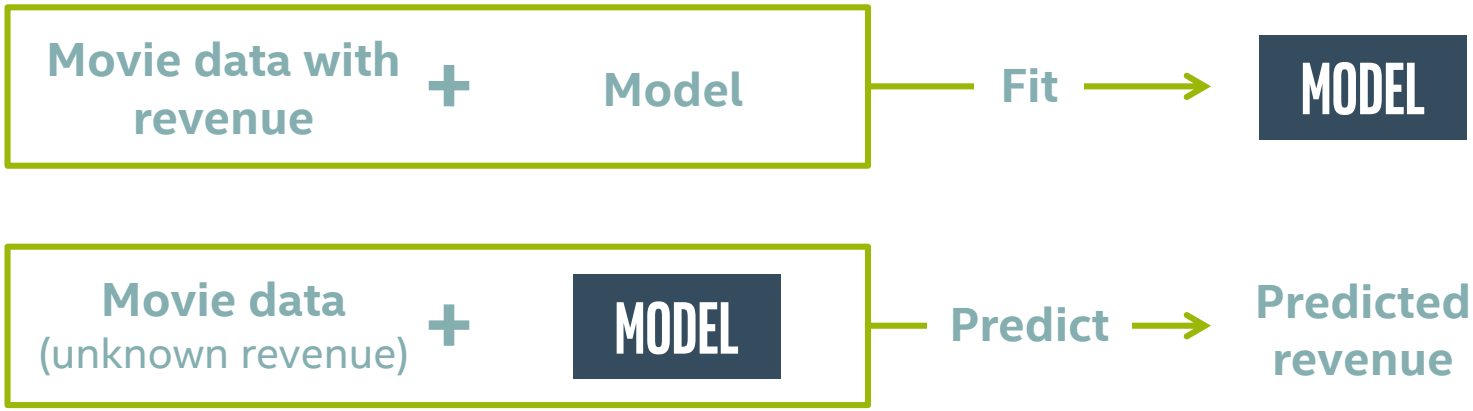
TYPES OF SUPERVISED LEARNING

REGRESSION	Outcome is continuous (numerical)
CLASSIFICATION	Outcome is a category

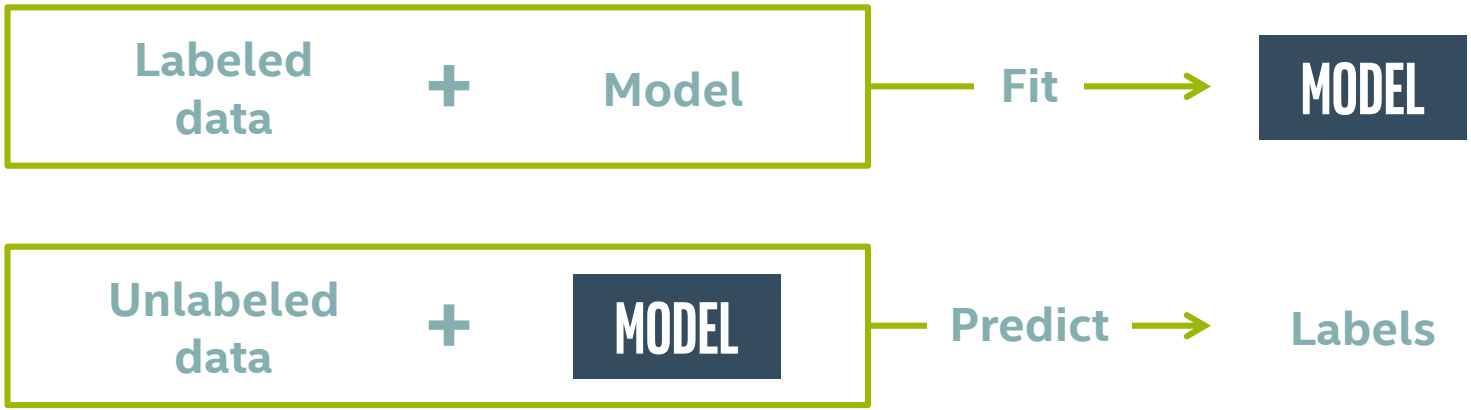
SUPERVISED LEARNING OVERVIEW



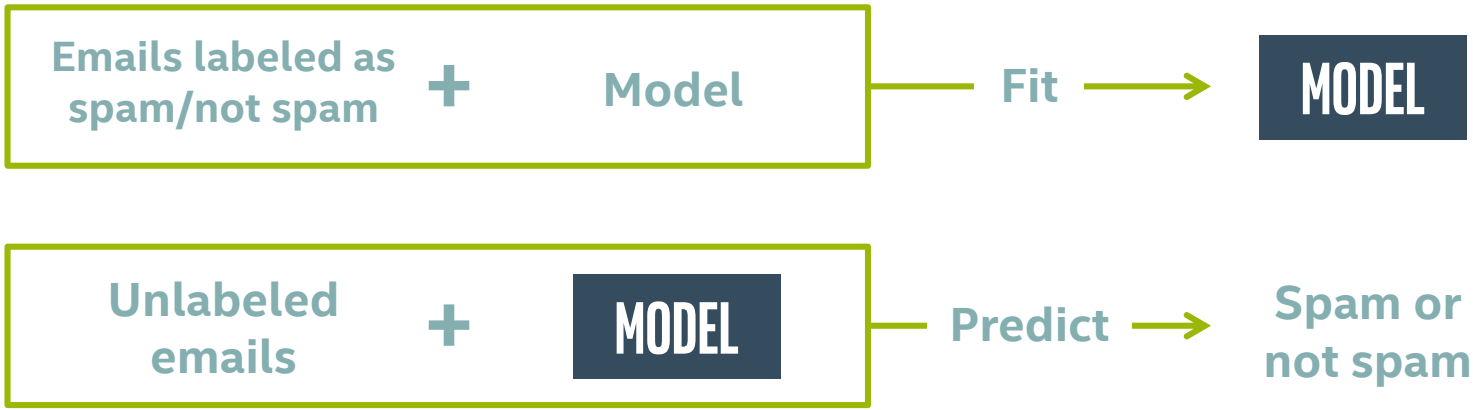
REGRESSION: NUMERICAL ANSWERS



CLASSIFICATION: CATEGORICAL ANSWERS



CLASSIFICATION: CATEGORICAL ANSWERS



MACHINE LEARNING VOCABULARY

- **Target: predicted category or value of the data**
  (column to predict)

MACHINE LEARNING VOCABULARY

Sepal length	Sepal width	Petal length	Petal width	Species
6.7	3.0	5.2	2.3	Virginica
6.4	2.8	5.6	2.1	Virginica
4.6	3.4	1.4	0.3	Setosa
6.9	3.1	4.9	1.5	Versicolor
4.4	2.9	1.4	0.2	Setosa
4.8	3.0	1.4	0.1	Setosa
5.9	3.0	5.1	1.8	Virginica
5.4	3.9	1.3	0.4	Setosa
4.9	3.0	1.4	0.2	Setosa
5.4	3.4	1.7	0.2	Setosa

MACHINE LEARNING VOCABULARY

Sepal length	Sepal width	Petal length	Petal width	Species
6.7	3.0	5.2	2.3	Virginica
6.4	2.8	5.6	2.1	Virginica
4.6	3.4	1.4	0.3	Setosa
6.9	3.1	4.9	1.5	Versicolor
4.4	2.9	1.4	0.2	Setosa
4.8	3.0	1.4	0.1	Setosa
5.9	3.0	5.1	1.8	Virginica
5.4	3.9	1.3	0.4	Setosa
4.9	3.0	1.4	0.2	Setosa
5.4	3.4	1.7	0.2	Setosa

MACHINE LEARNING VOCABULARY

- **Target: predicted category or value of the data**
  (column to predict)
- **Features: properties of the data used for prediction**
  (non-target columns)

MACHINE LEARNING VOCABULARY

Features

Sepal length	Sepal width	Petal length	Petal width	Species
6.7	3.0	5.2	2.3	Virginica
6.4	2.8	5.6	2.1	Virginica
4.6	3.4	1.4	0.3	Setosa
6.9	3.1	4.9	1.5	Versicolor
4.4	2.9	1.4	0.2	Setosa
4.8	3.0	1.4	0.1	Setosa
5.9	3.0	5.1	1.8	Virginica
5.4	3.9	1.3	0.4	Setosa
4.9	3.0	1.4	0.2	Setosa
5.4	3.4	1.7	0.2	Setosa

MACHINE LEARNING VOCABULARY

- **Target: predicted category or value of the data**
  (column to predict)
- **Features: properties of the data used for prediction**
  (non-target columns)
- **Example: a single data point within the data**
  (one row)

MACHINE LEARNING VOCABULARY

Examples →

Sepal length	Sepal width	Petal length	Petal width	Species
6.7	3.0	5.2	2.3	Virginica
6.4	2.8	5.6	2.1	Virginica
4.6	3.4	1.4	0.3	Setosa
6.9	3.1	4.9	1.5	Versicolor
4.4	2.9	1.4	0.2	Setosa
4.8	3.0	1.4	0.1	Setosa
5.9	3.0	5.1	1.8	Virginica
5.4	3.9	1.3	0.4	Setosa
4.9	3.0	1.4	0.2	Setosa
5.4	3.4	1.7	0.2	Setosa

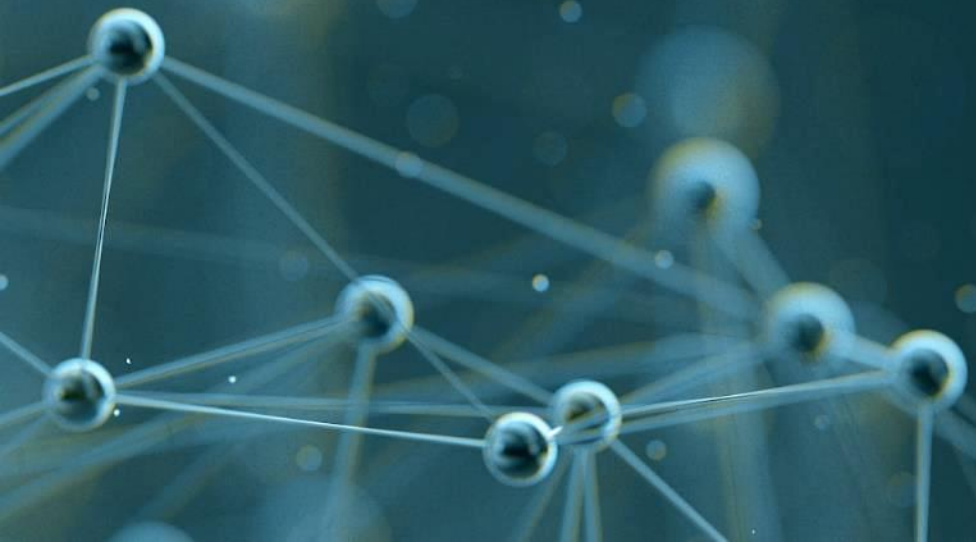
MACHINE LEARNING VOCABULARY

- **Target: predicted category or value of the data**
  (column to predict)
- **Features: properties of the data used for prediction**
  (non-target columns)
- **Example: a single data point within the data**
  (one row)
- **Label: the target value for a single data point**

MACHINE LEARNING VOCABULARY

Sepal length	Sepal width	Petal length	Petal width	Species
6.7	3.0	5.2	2.3	Virginica
6.4	2.8	5.6	2.1	Virginica
4.6	3.4	1.4	0.3	Setosa
6.9	3.1	4.9	1.5	Versicolor
4.4	2.9	1.4	0.2	Setosa
4.8	3.0	1.4	0.1	Setosa
5.9	3.0	5.1	1.8	Virginica
5.4	3.9	1.3	0.4	Setosa
4.9	3.0	1.4	0.2	Setosa
5.4	3.4	1.7	0.2	Setosa

K - NEAREST NEIGHBORS



WHAT IS CLASSIFICATION?

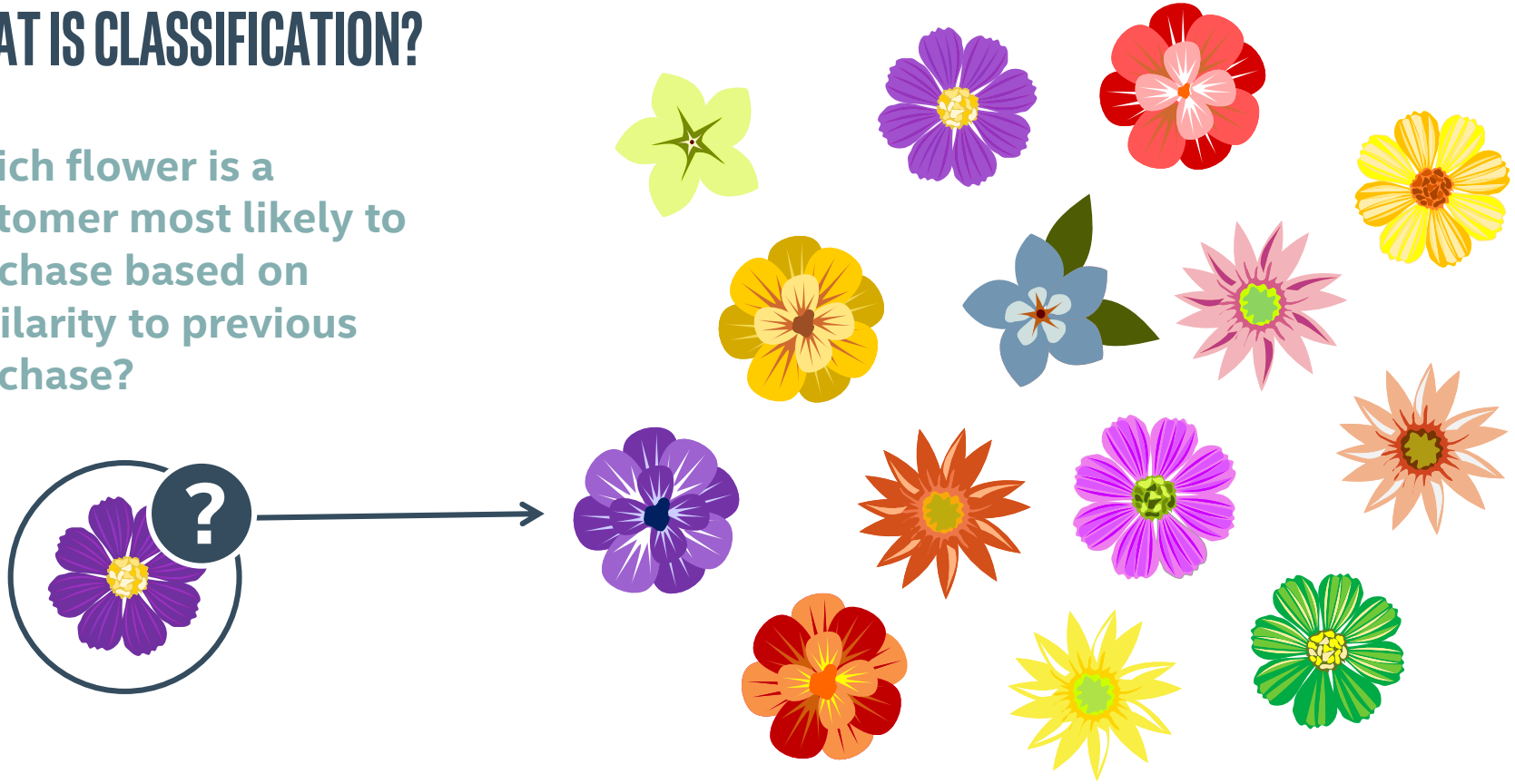
A flower shop wants to guess a customer's purchase from similarity to most recent purchase.

WHAT IS CLASSIFICATION?

Which flower is a customer most likely to purchase based on similarity to previous purchase?

WHAT IS CLASSIFICATION?

Which flower is a customer most likely to purchase based on similarity to previous purchase?

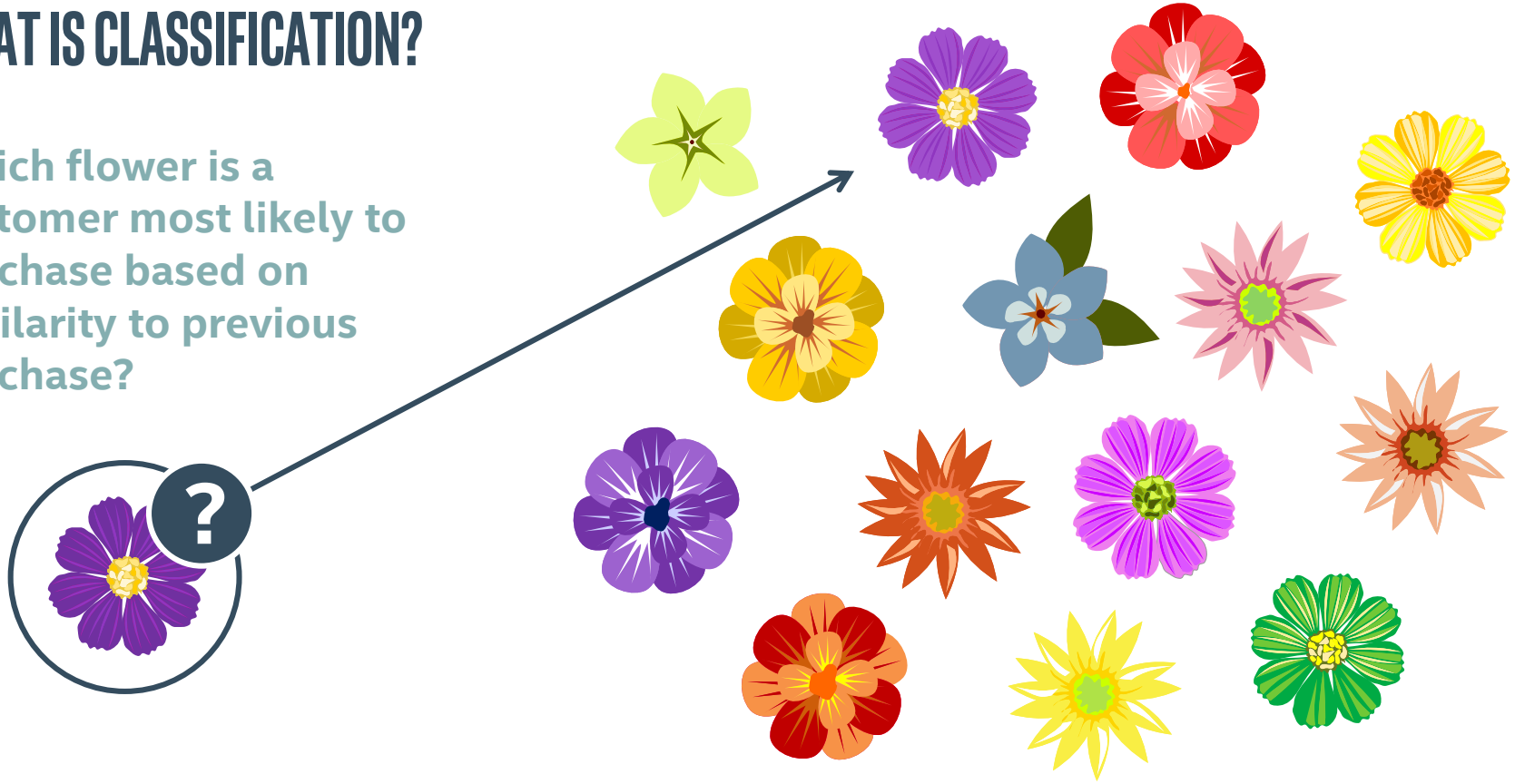


WHAT IS CLASSIFICATION?

Which flower is a customer most likely to purchase based on similarity to previous purchase?

WHAT IS CLASSIFICATION?

Which flower is a customer most likely to purchase based on similarity to previous purchase?



WHAT IS NEEDED FOR CLASSIFICATION?

- **Model data with:**
  - Features that can be quantitated

WHAT IS NEEDED FOR CLASSIFICATION?

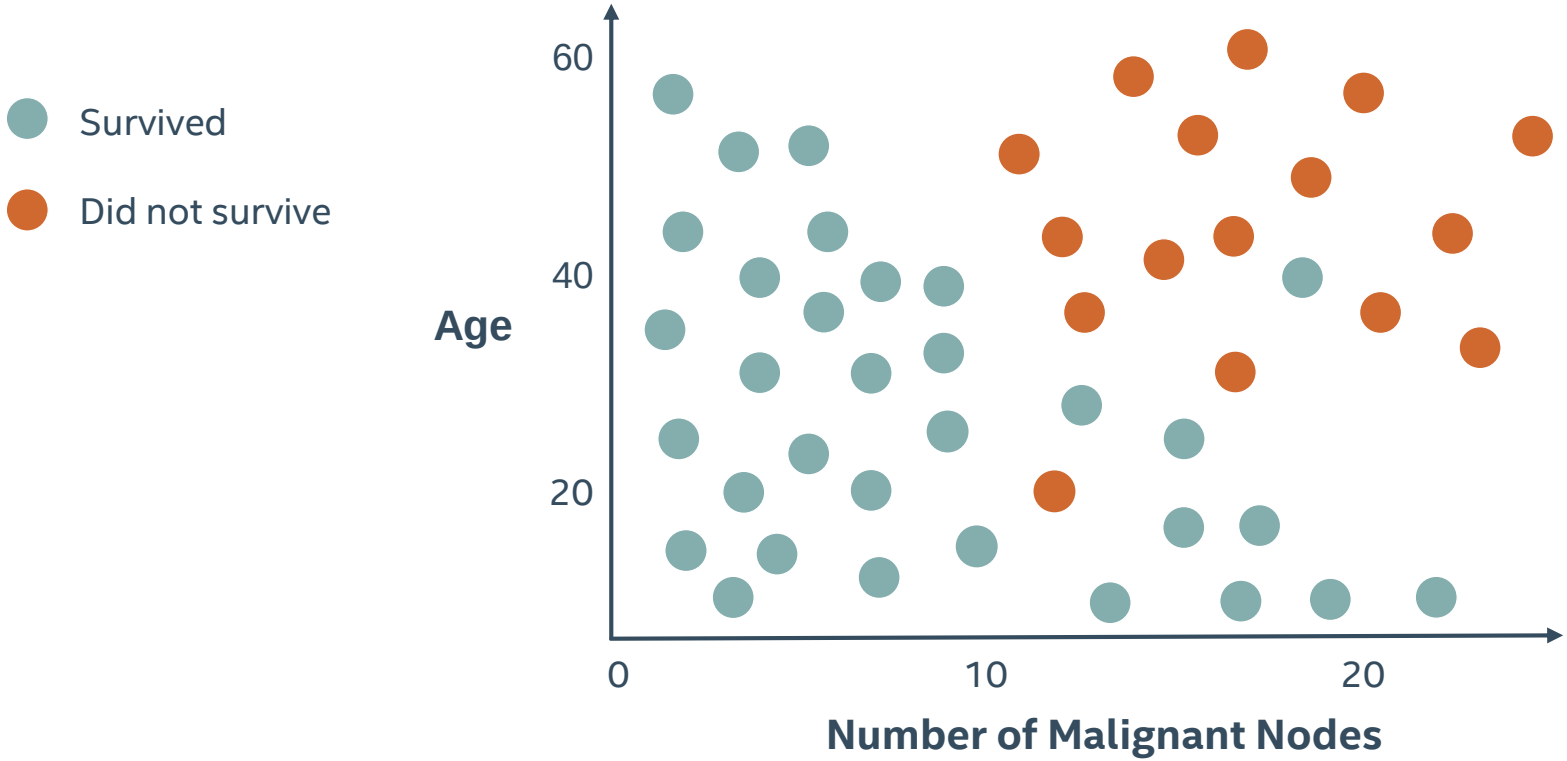
- **Model data with:**
  - Features that can be quantitated
  - Labels that are known

WHAT IS NEEDED FOR CLASSIFICATION?

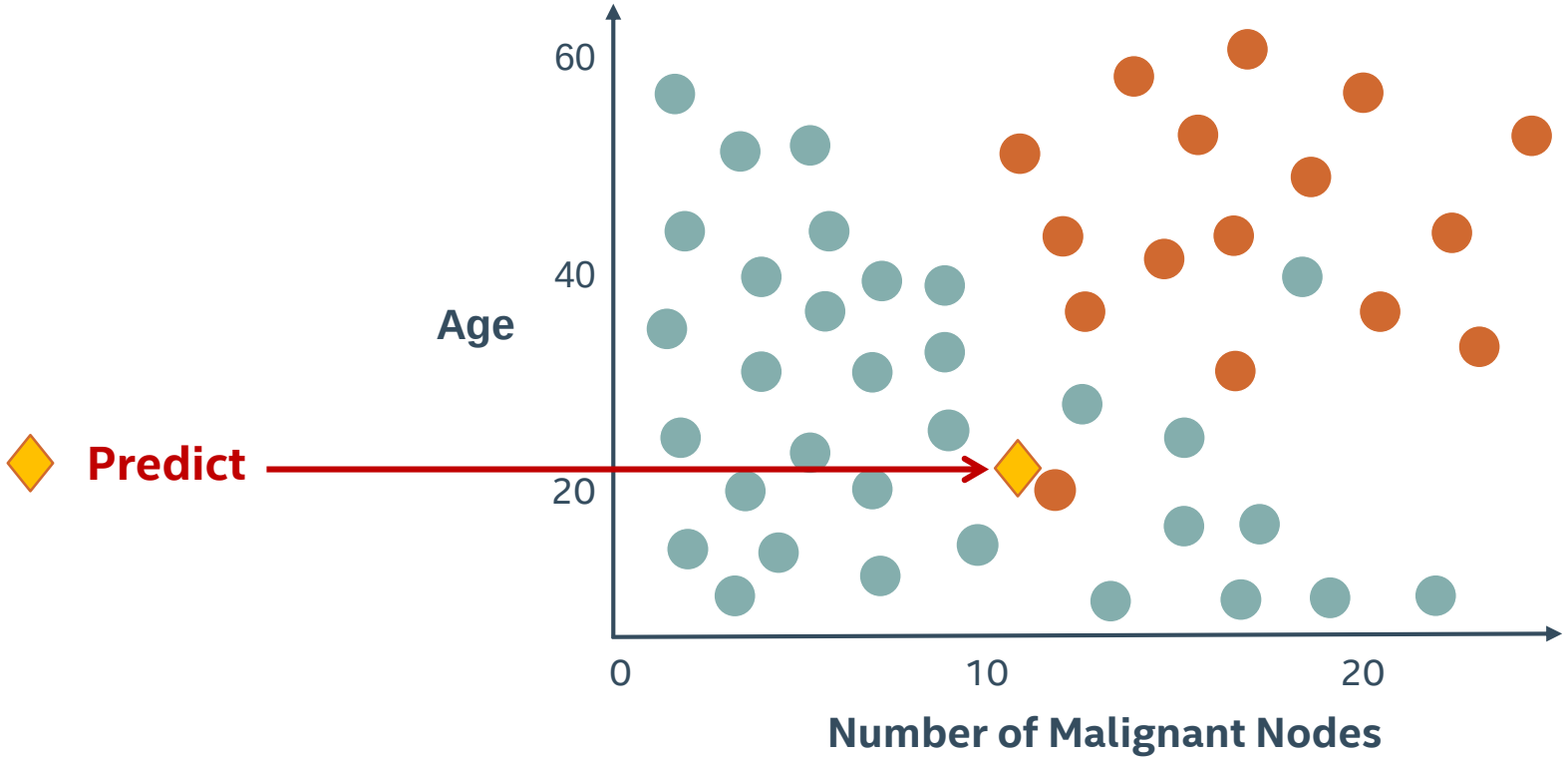
- **Model data with:**
  - Features that can be quantitated
  - Labels that are known
- **Method to measure similarity**

K NEAREST NEIGHBORS CLASSIFICATION

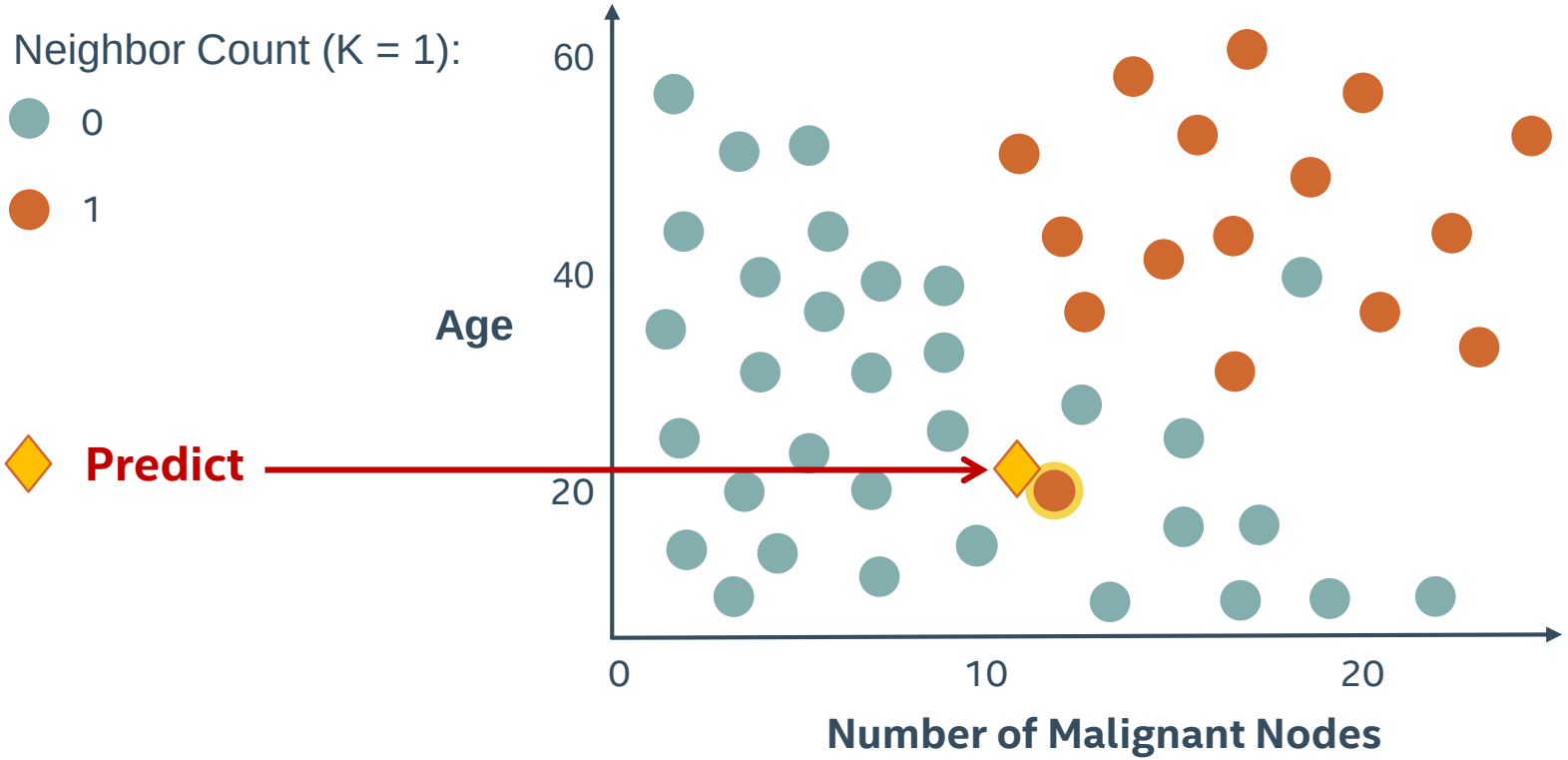
K NEAREST NEIGHBORS CLASSIFICATION



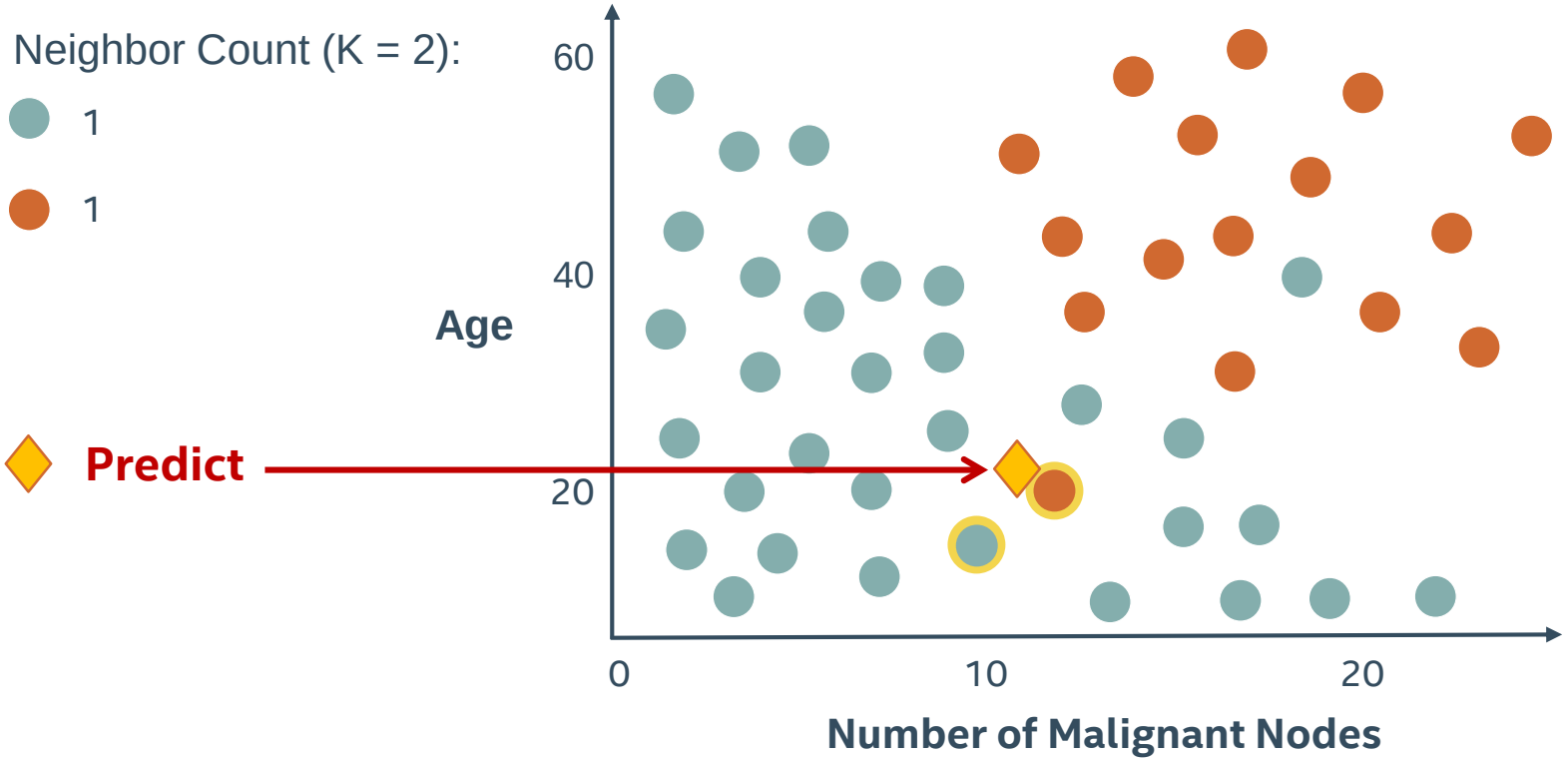
K NEAREST NEIGHBORS CLASSIFICATION



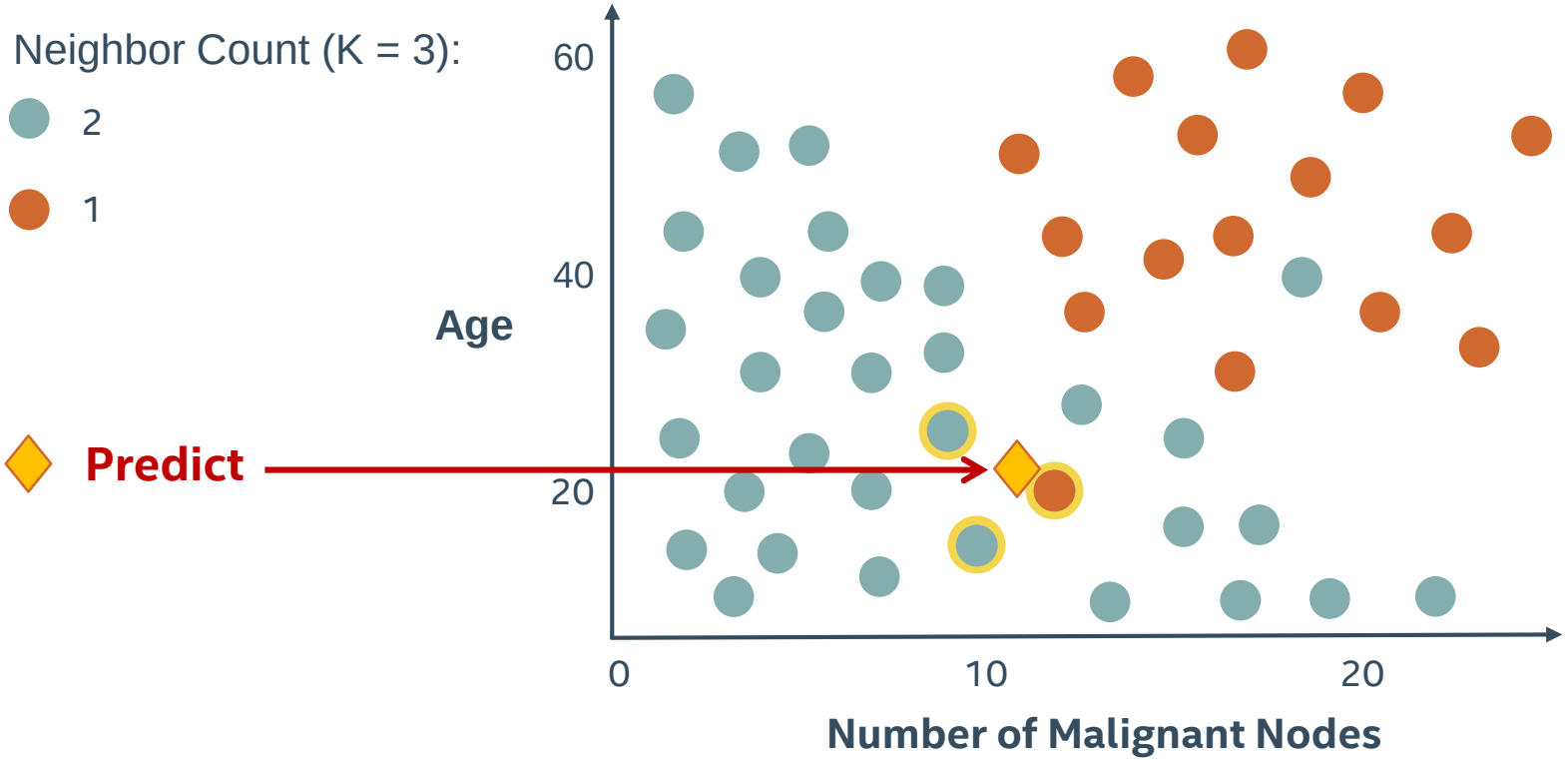
K NEAREST NEIGHBORS CLASSIFICATION



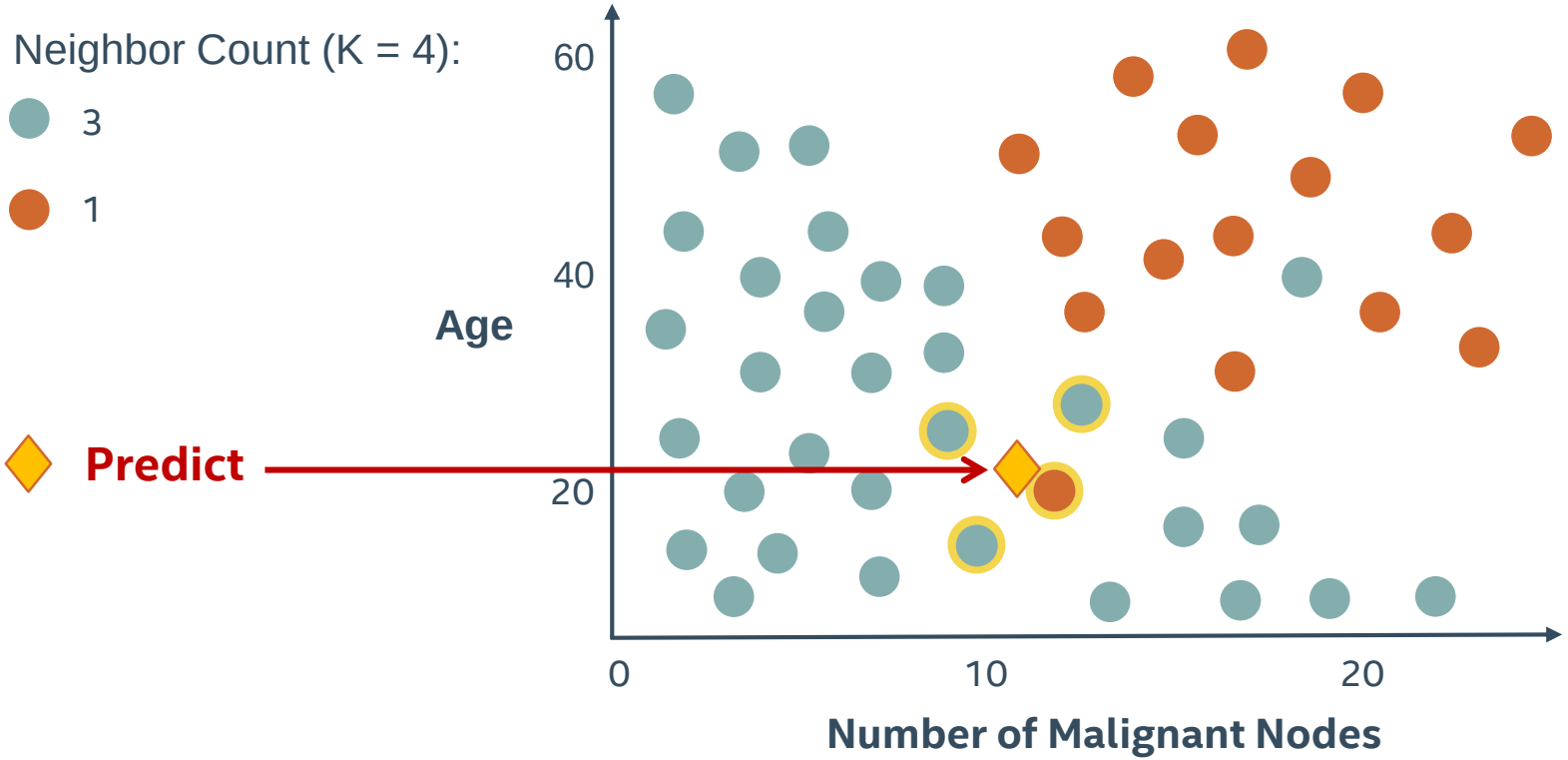
K NEAREST NEIGHBORS CLASSIFICATION



K NEAREST NEIGHBORS CLASSIFICATION



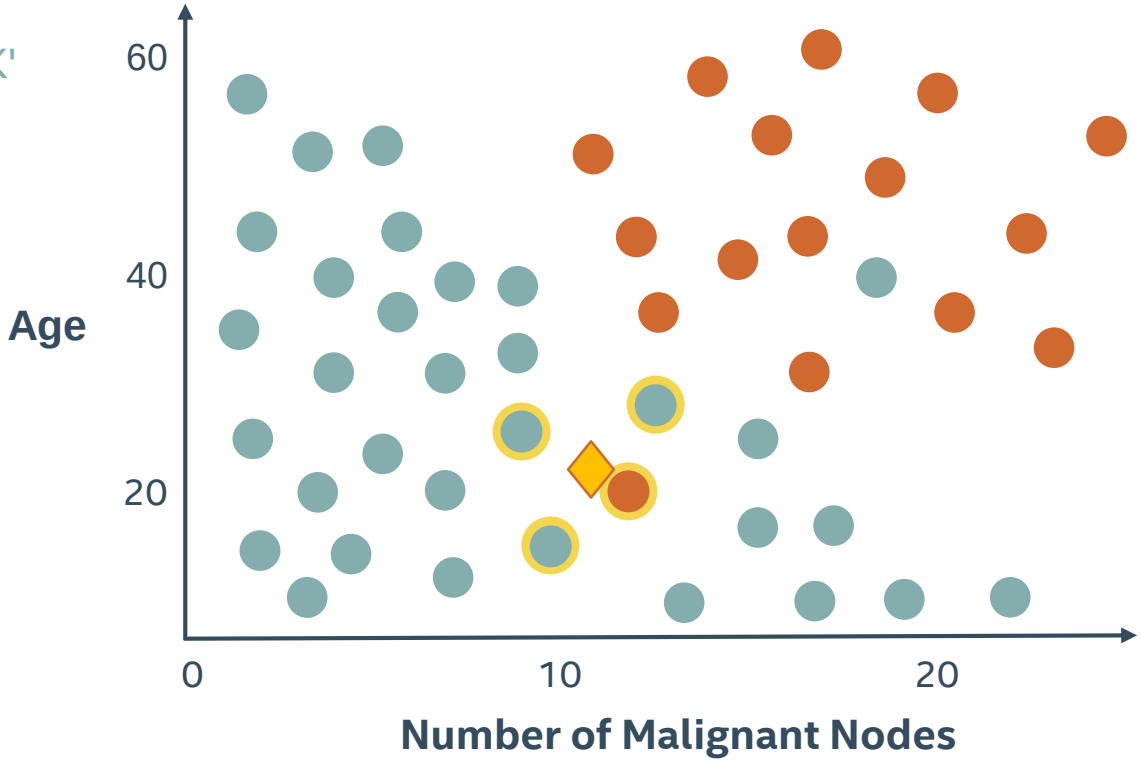
K NEAREST NEIGHBORS CLASSIFICATION



WHAT IS NEEDED TO SELECT A KNN MODEL?

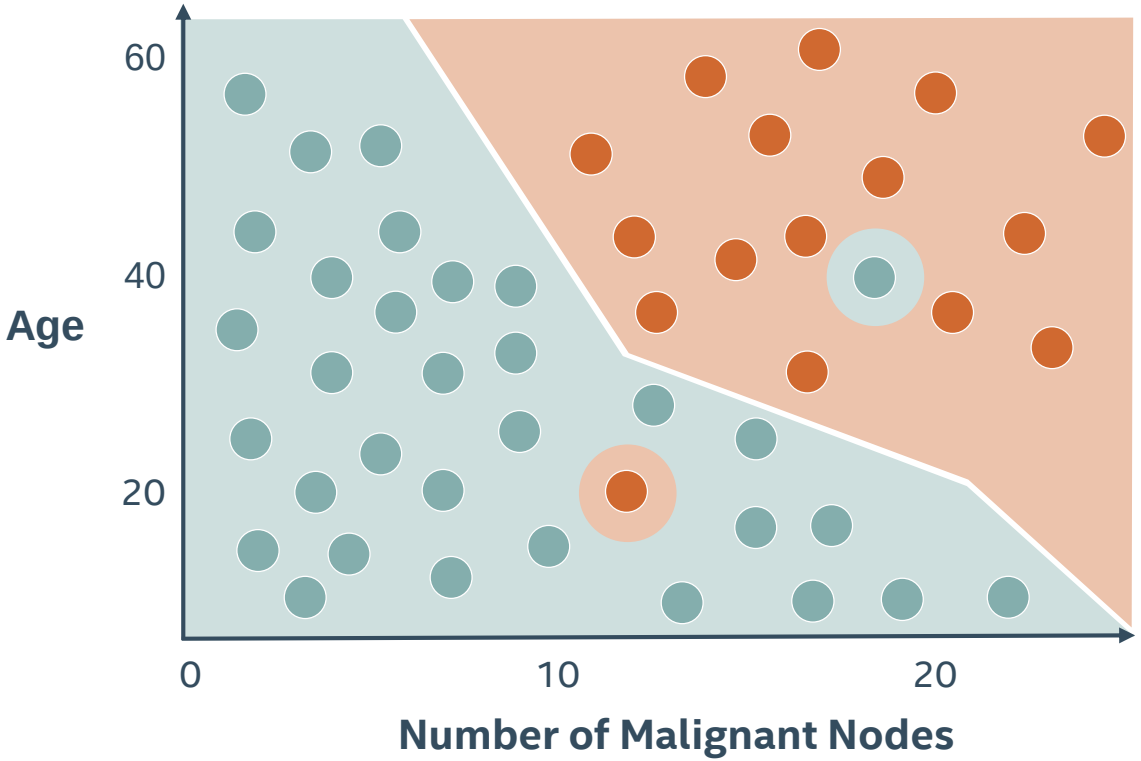
WHAT IS NEEDED TO SELECT A KNN MODEL?

- Correct value for 'K'
- How to measure closeness of neighbors?



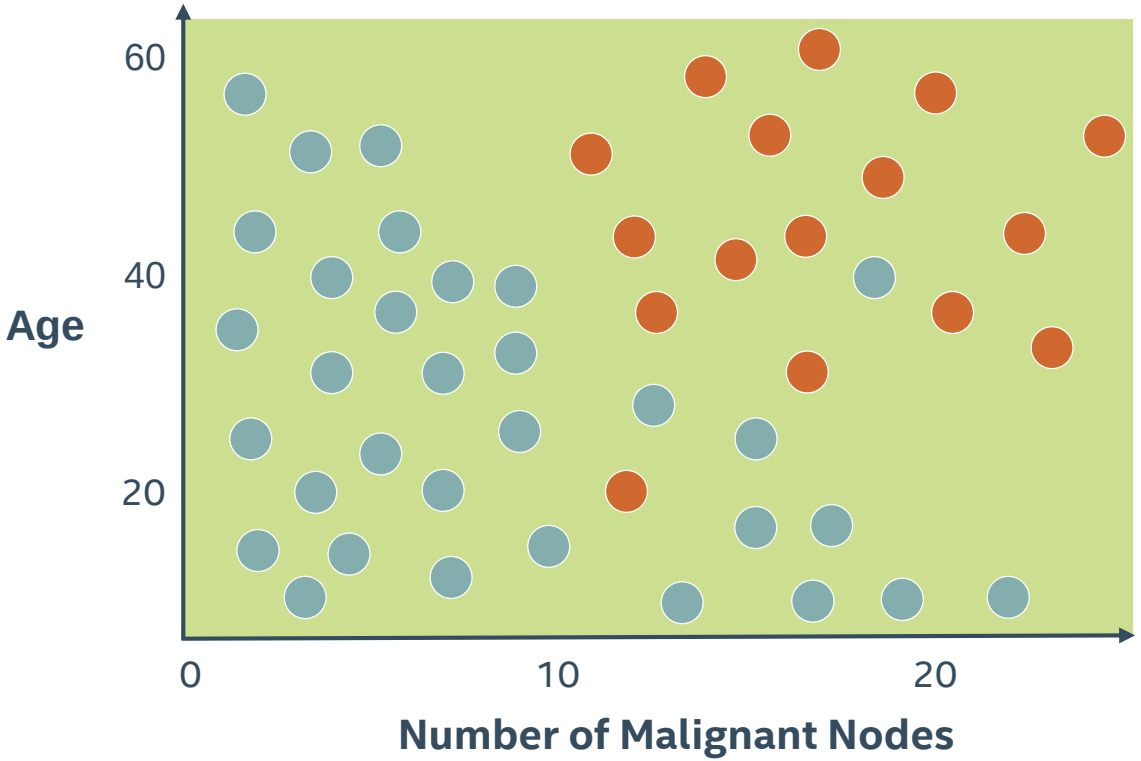
K NEAREST NEIGHBORS DECISION BOUNDARY

K=1

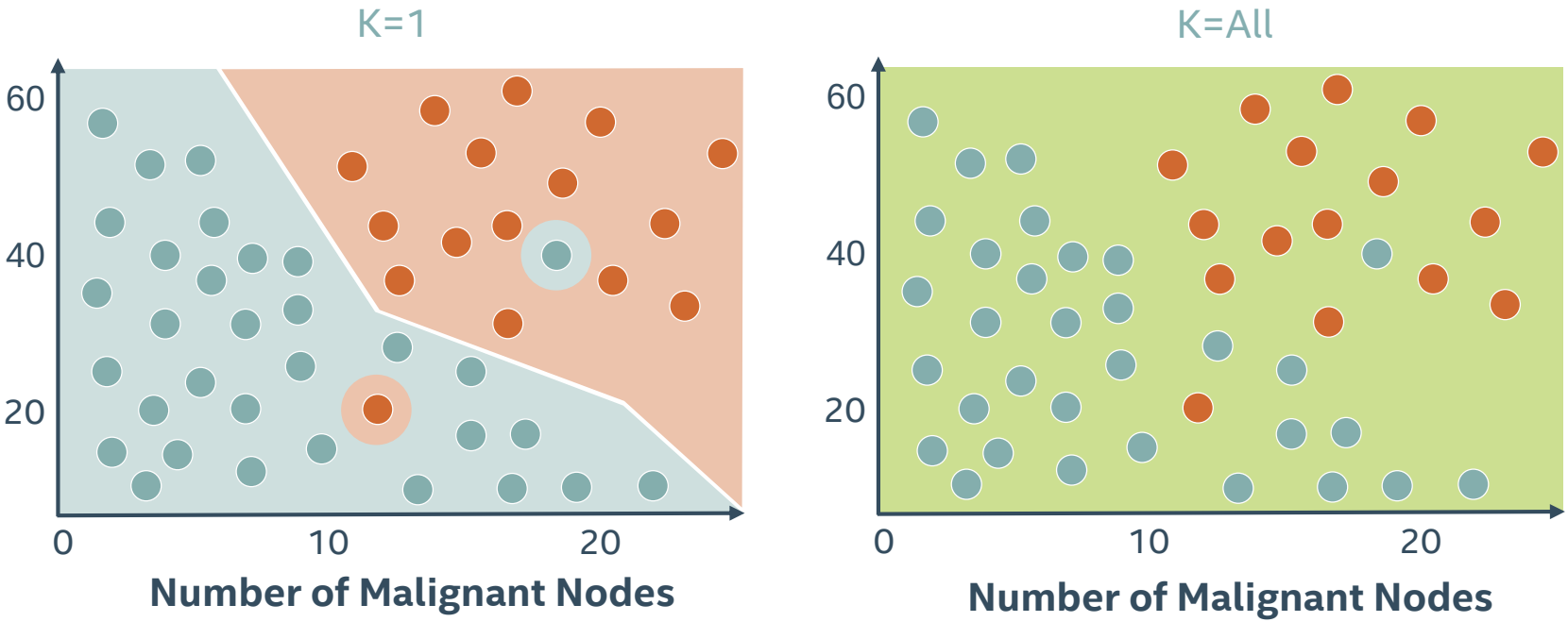


K NEAREST NEIGHBORS DECISION BOUNDARY

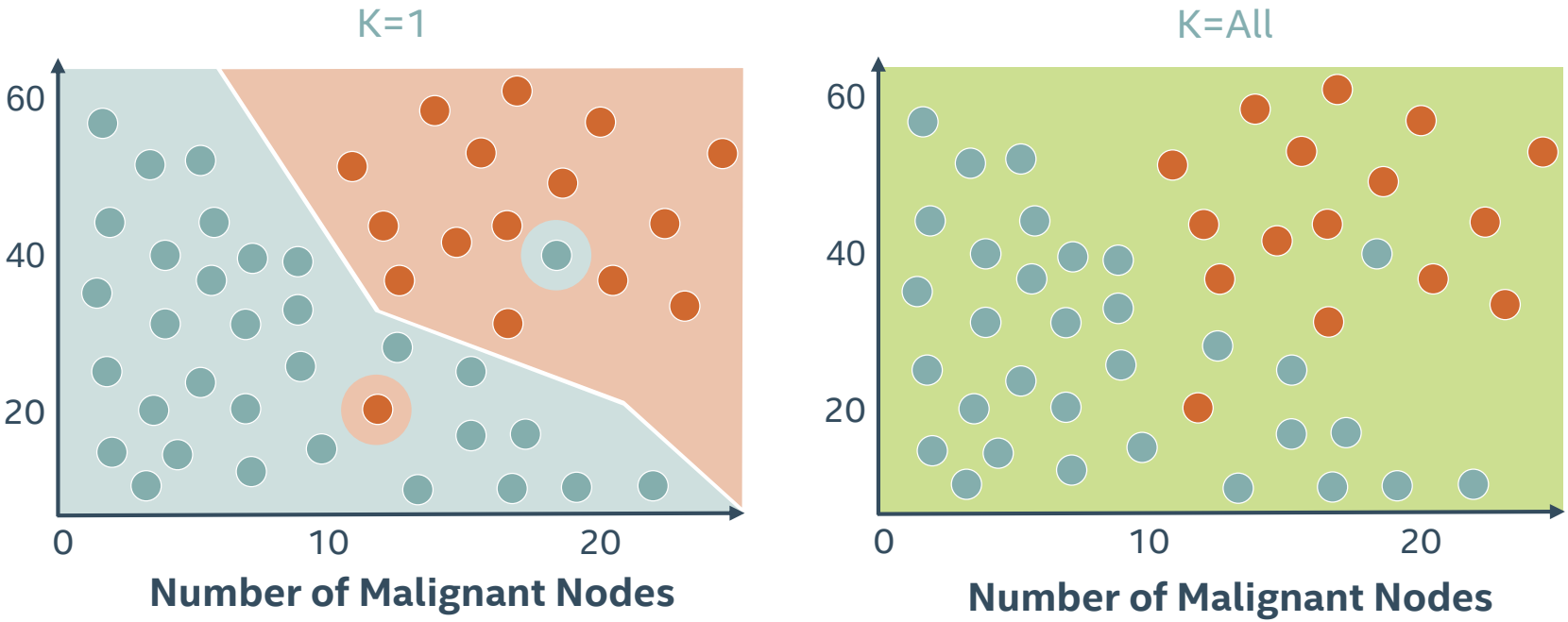
K = All



VALUE OF 'K' AFFECTS DECISION BOUNDARY

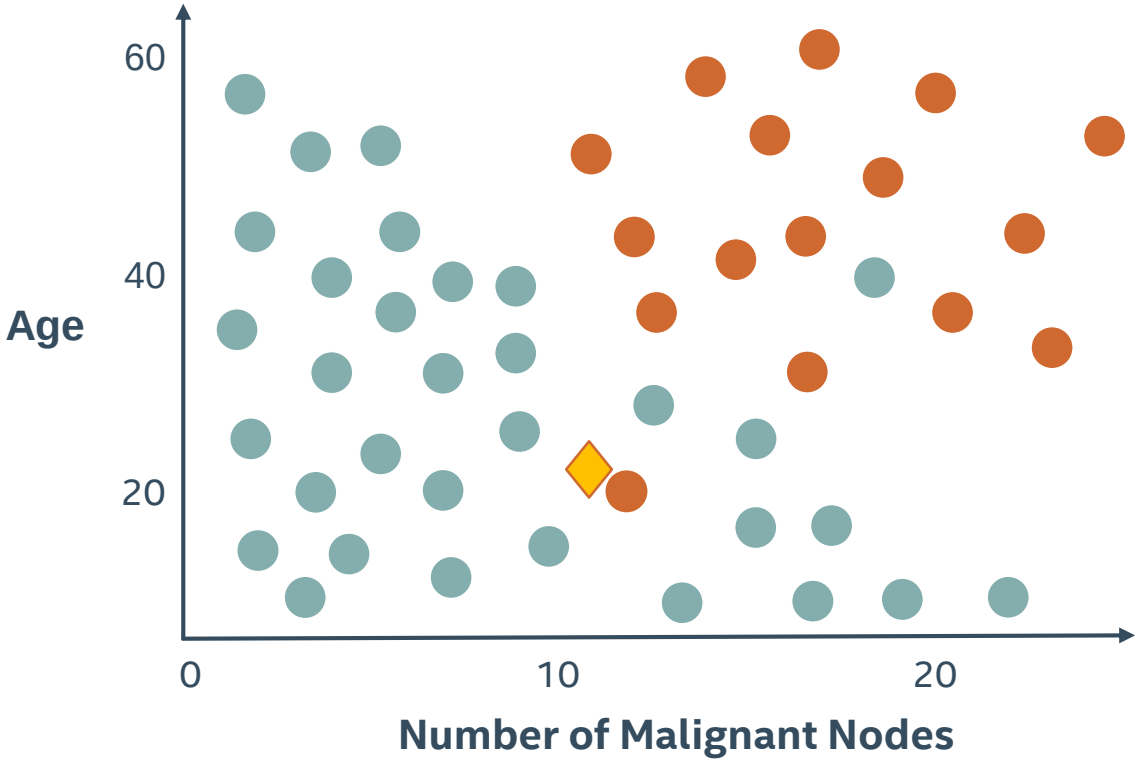


VALUE OF 'K' AFFECTS DECISION BOUNDARY

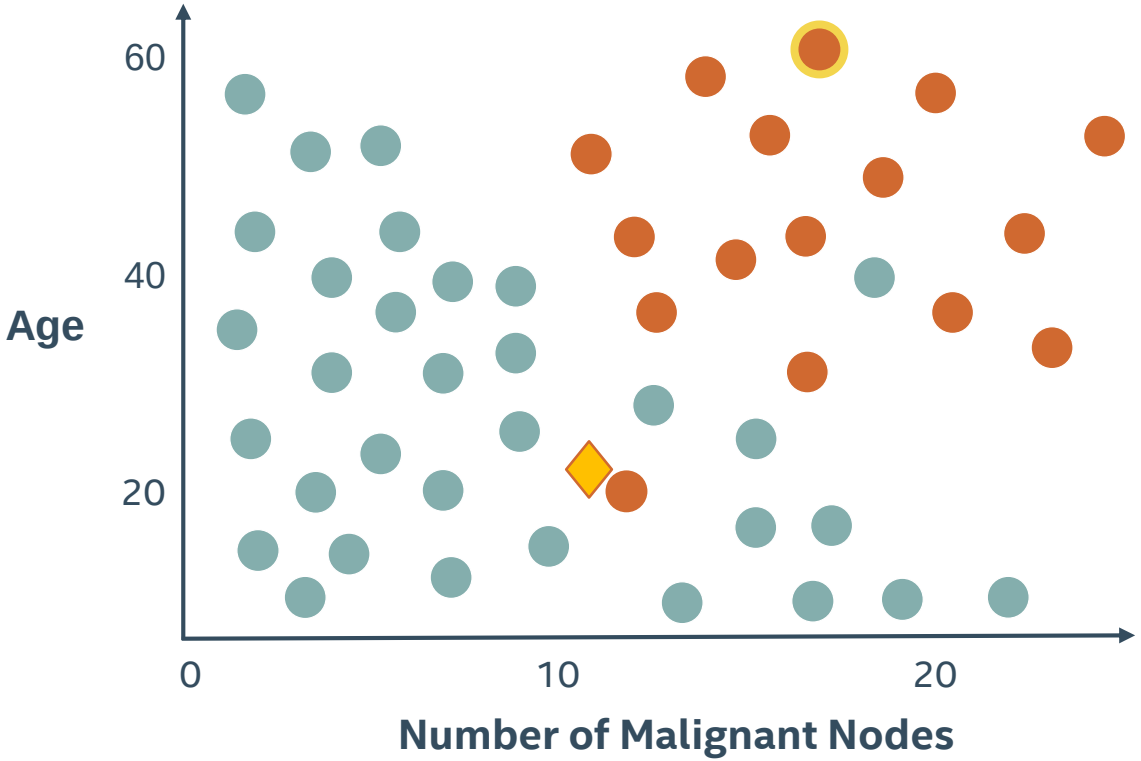


Methods for determining 'K' will be discussed in next lesson

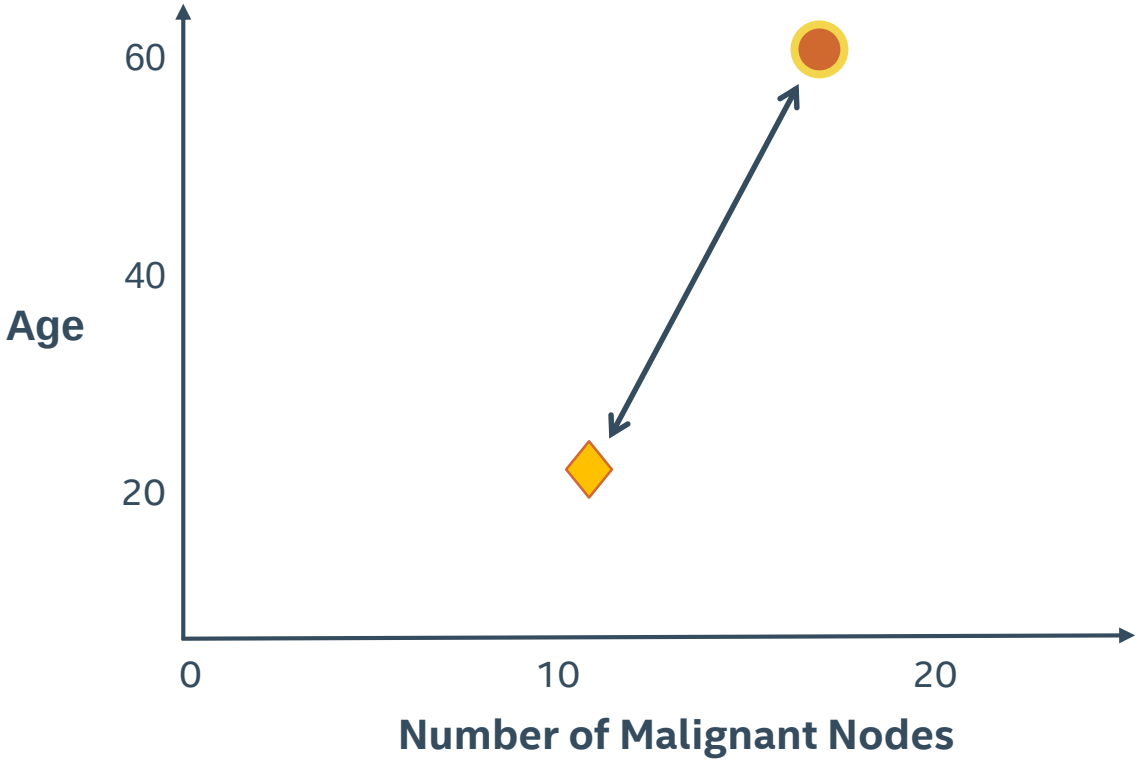
MEASUREMENT OF DISTANCE IN KNN



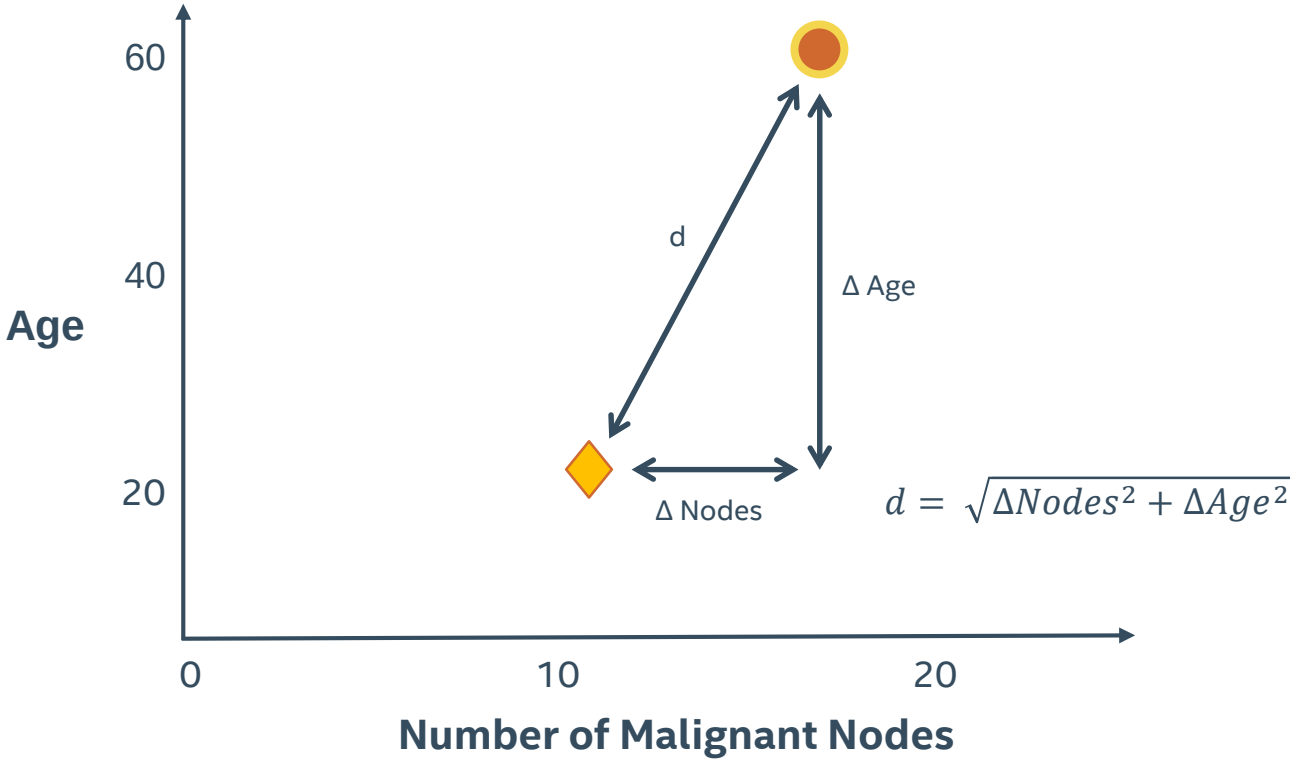
MEASUREMENT OF DISTANCE IN KNN



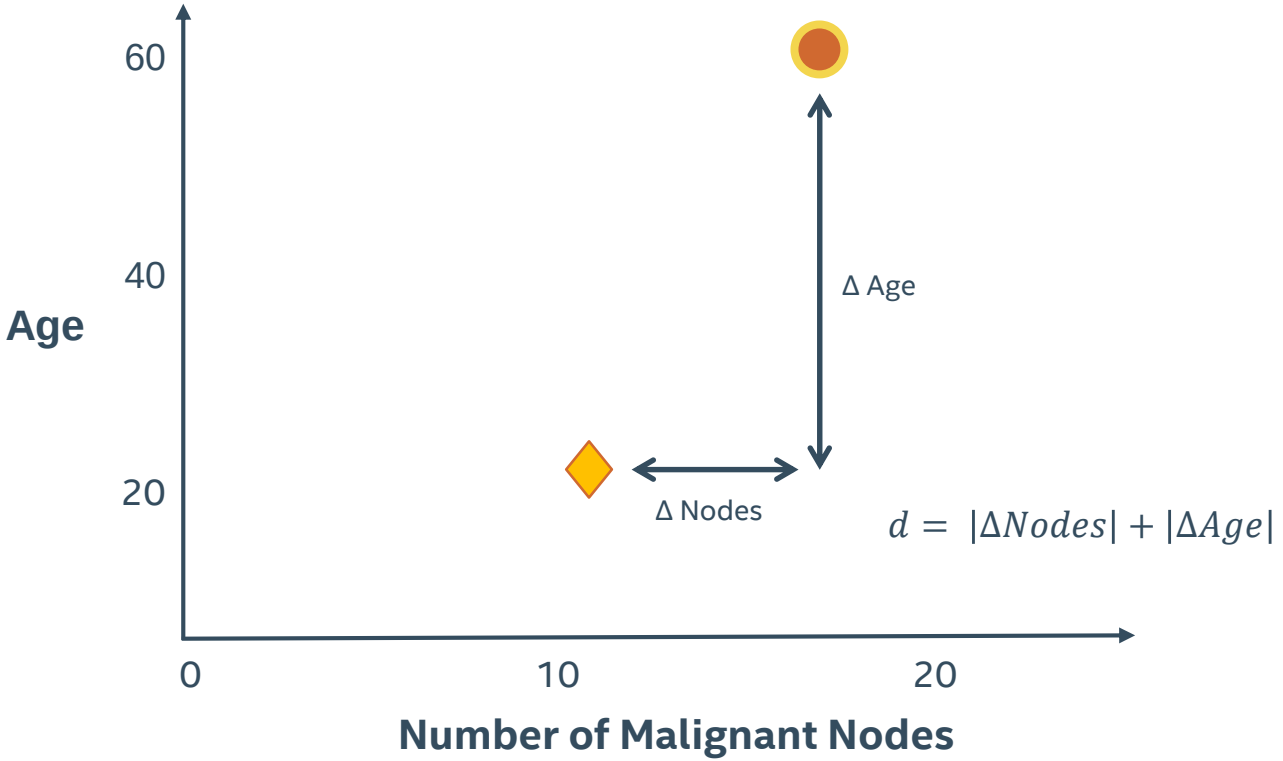
EUCLIDEAN DISTANCE



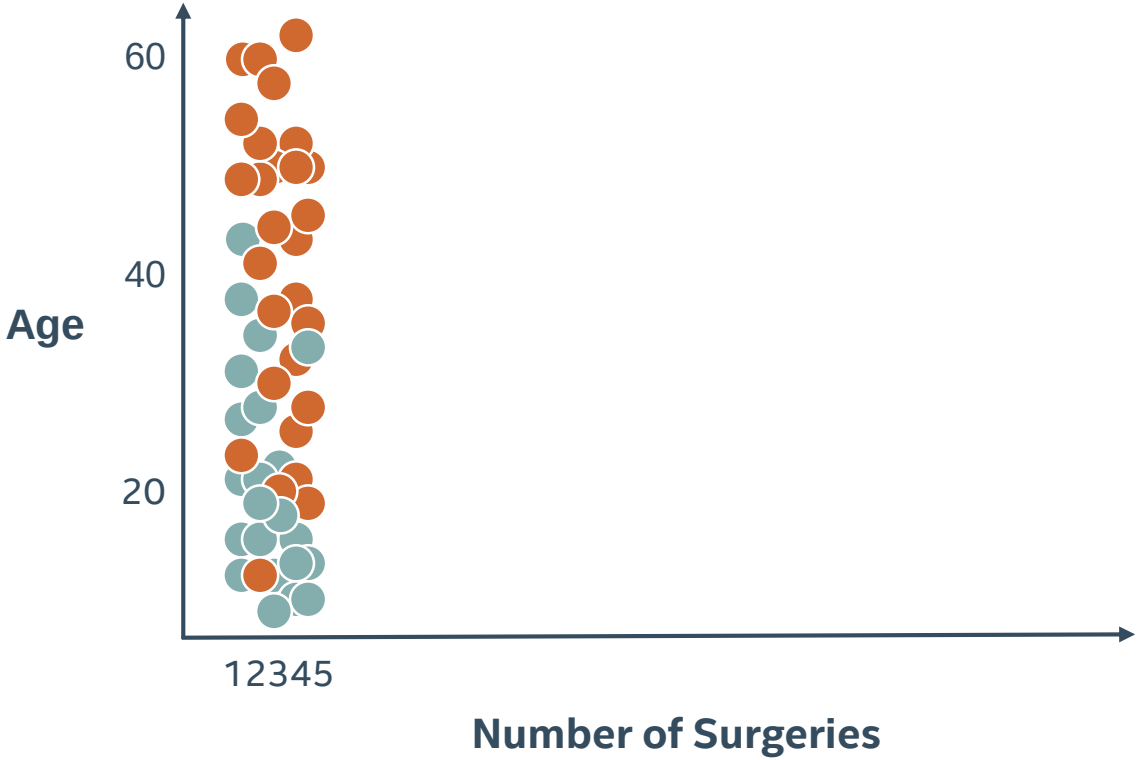
EUCLIDEAN DISTANCE (L2 DISTANCE)



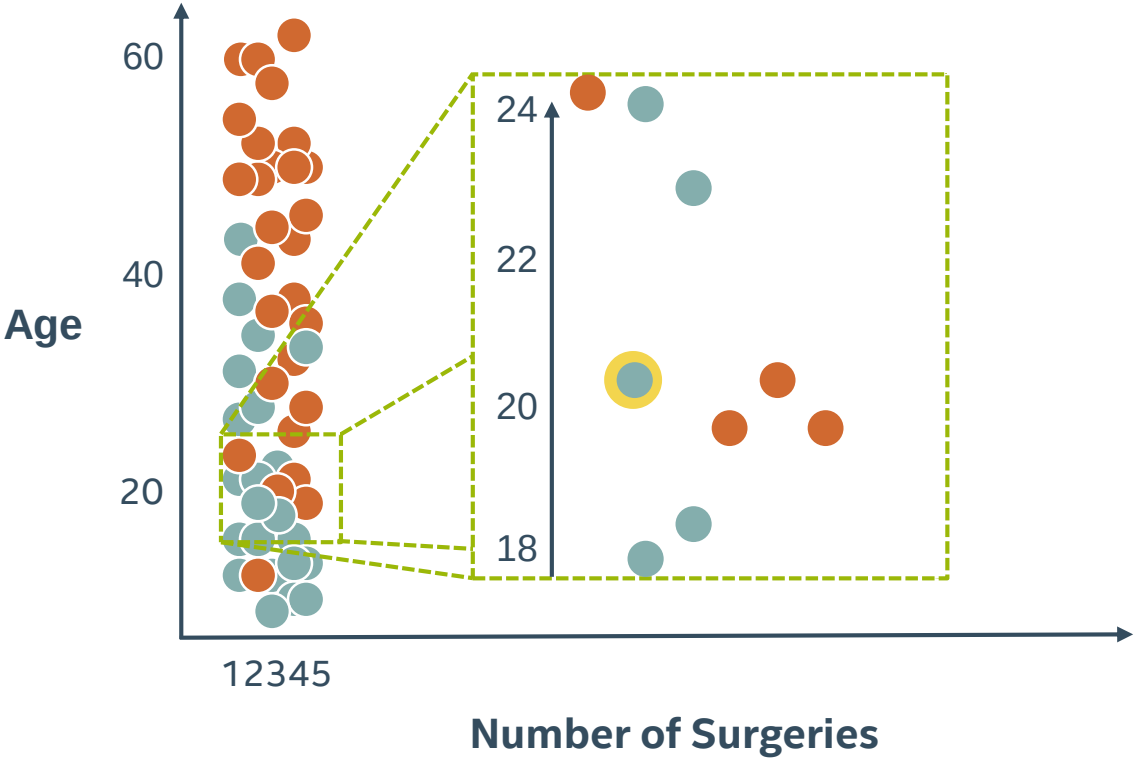
MANHATTAN DISTANCE (L1 OR CITY BLOCK DISTANCE)



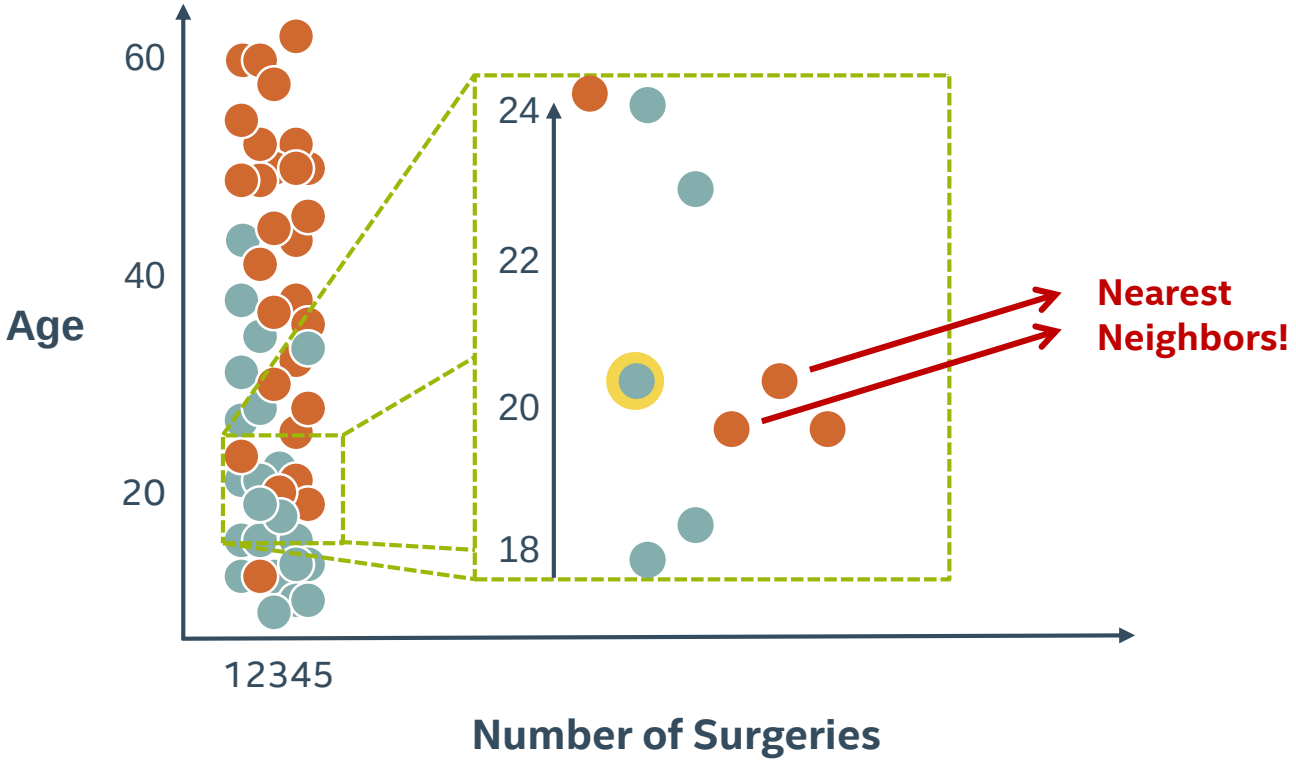
SCALE IS IMPORTANT FOR DISTANCE MEASUREMENT



SCALE IS IMPORTANT FOR DISTANCE MEASUREMENT

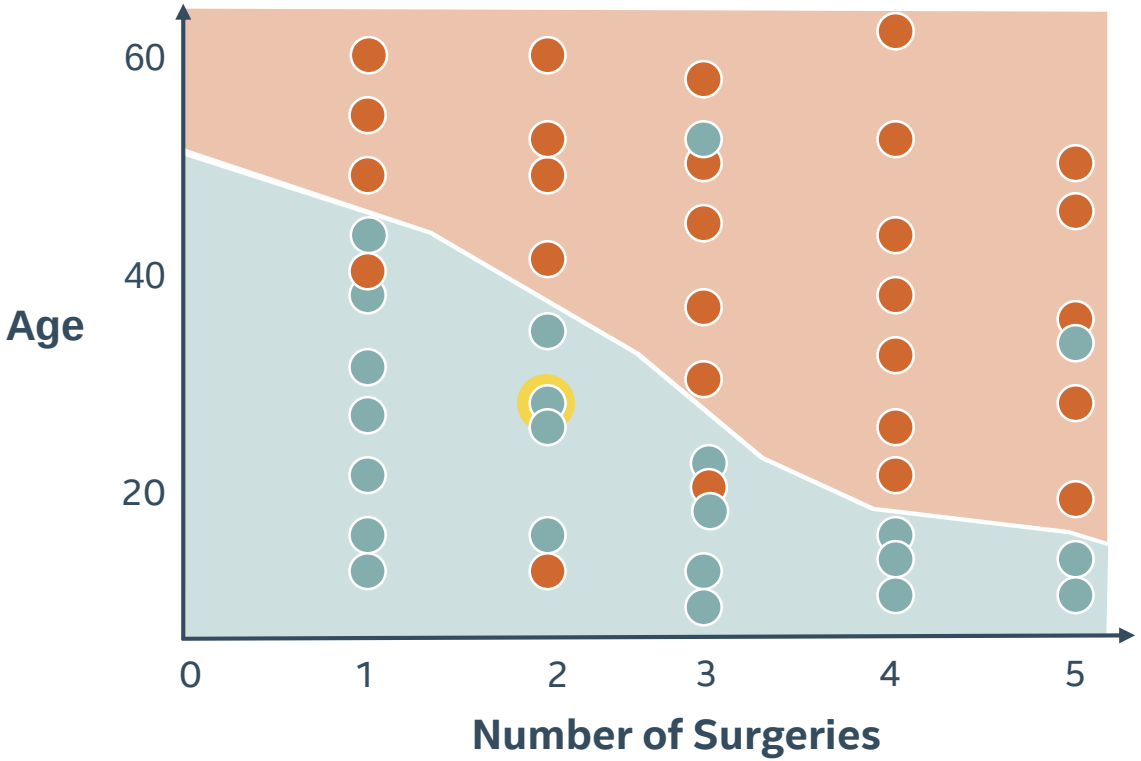


SCALE IS IMPORTANT FOR DISTANCE MEASUREMENT



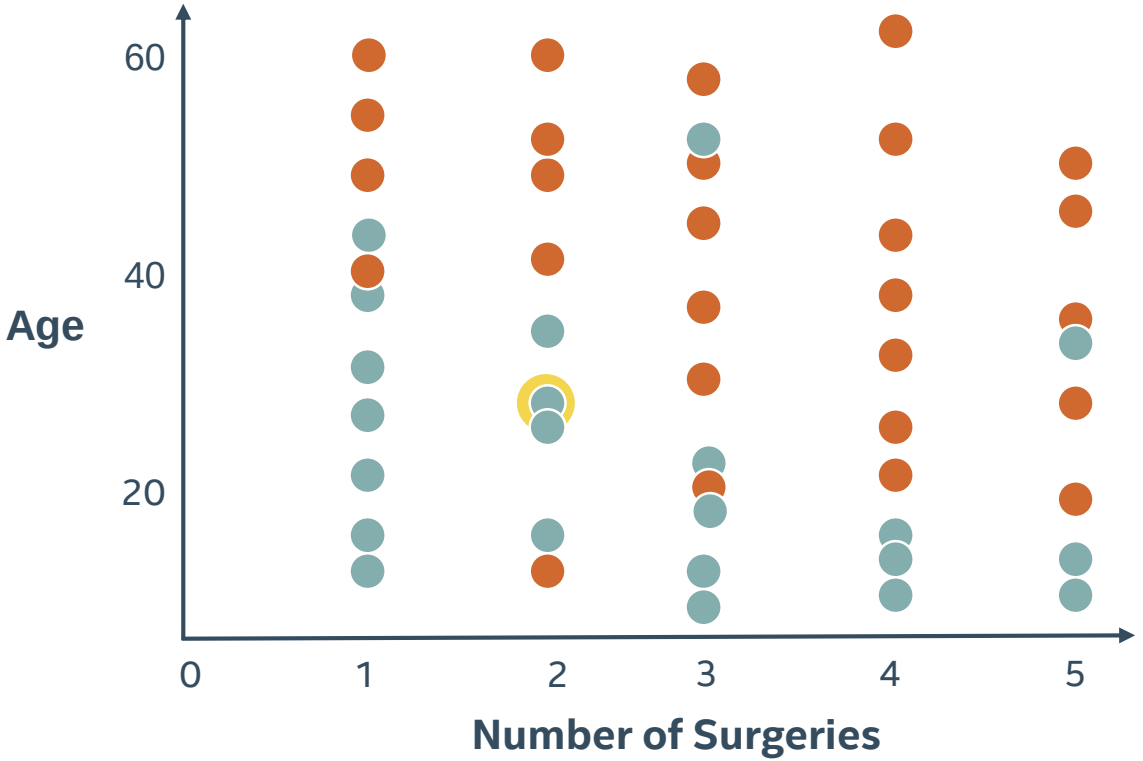
SCALE IS IMPORTANT FOR DISTANCE MEASUREMENT

"Feature Scaling"



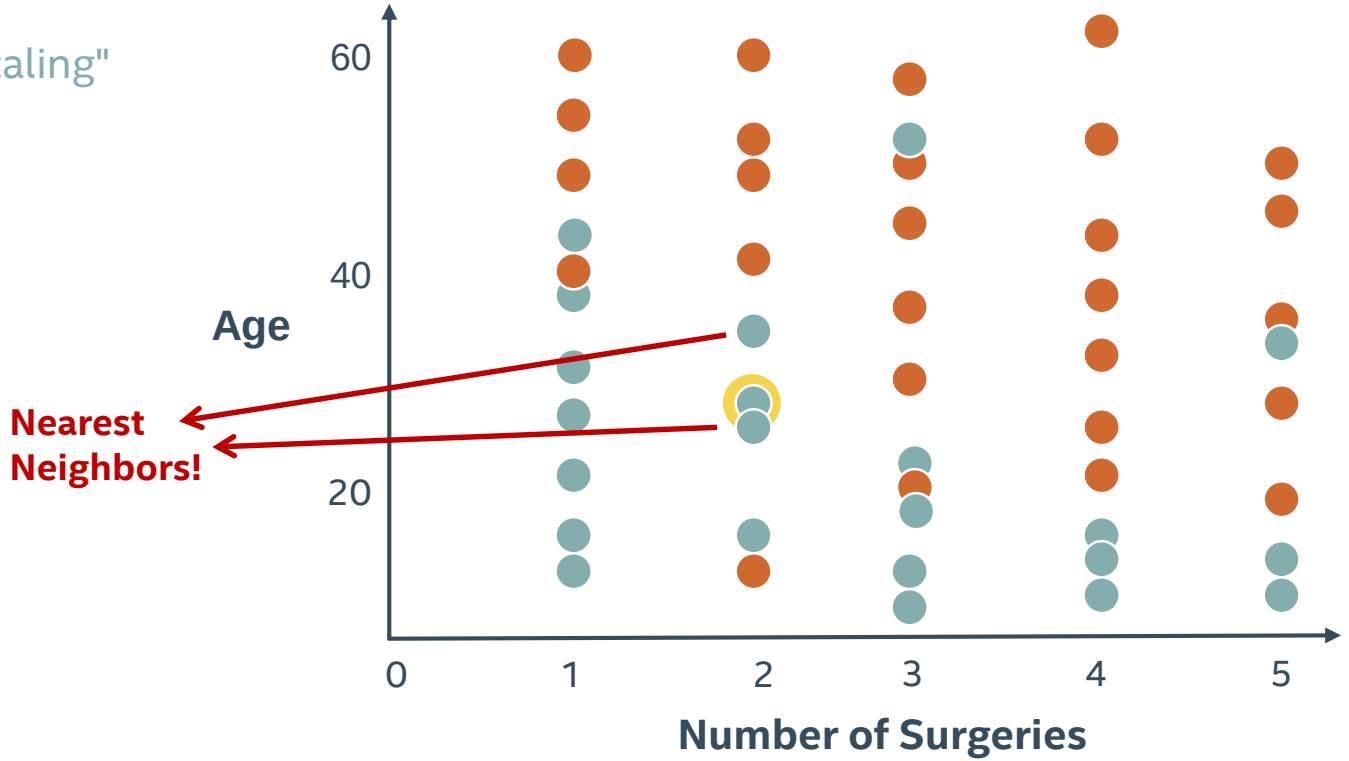
SCALE IS IMPORTANT FOR DISTANCE MEASUREMENT

"Feature Scaling"



SCALE IS IMPORTANT FOR DISTANCE MEASUREMENT

"Feature Scaling"



COMPARISON OF FEATURE SCALING METHODS

- **Standard Scaler:** Mean center data and scale to unit **variance**
- **Minimum-Maximum Scaler:** Scale data to fixed range (usually 0–1)
- **Maximum Absolute Value Scaler:** Scale maximum absolute value

FEATURE SCALING: THE SYNTAX

Import the class containing the scaling method

```
from sklearn.preprocessing import StandardScaler
```

FEATURE SCALING: THE SYNTAX

Import the class containing the scaling method

```
from sklearn.preprocessing import StandardScaler
```

Create an instance of the class

```
StdSc = StandardScaler()
```

FEATURE SCALING: THE SYNTAX

Import the class containing the scaling method

```
from sklearn.preprocessing import StandardScaler
```

Create an instance of the class

```
StdSc = StandardScaler()
```

Fit the scaling parameters and then transform the data

```
StdSc = StdSc.fit(X_data)
```

```
X_scaled = KNN.transform(X_data)
```

FEATURE SCALING: THE SYNTAX

Import the class containing the scaling method

```
from sklearn.preprocessing import StandardScaler
```

Create an instance of the class

```
StdSc = StandardScaler()
```

Fit the scaling parameters and then transform the data

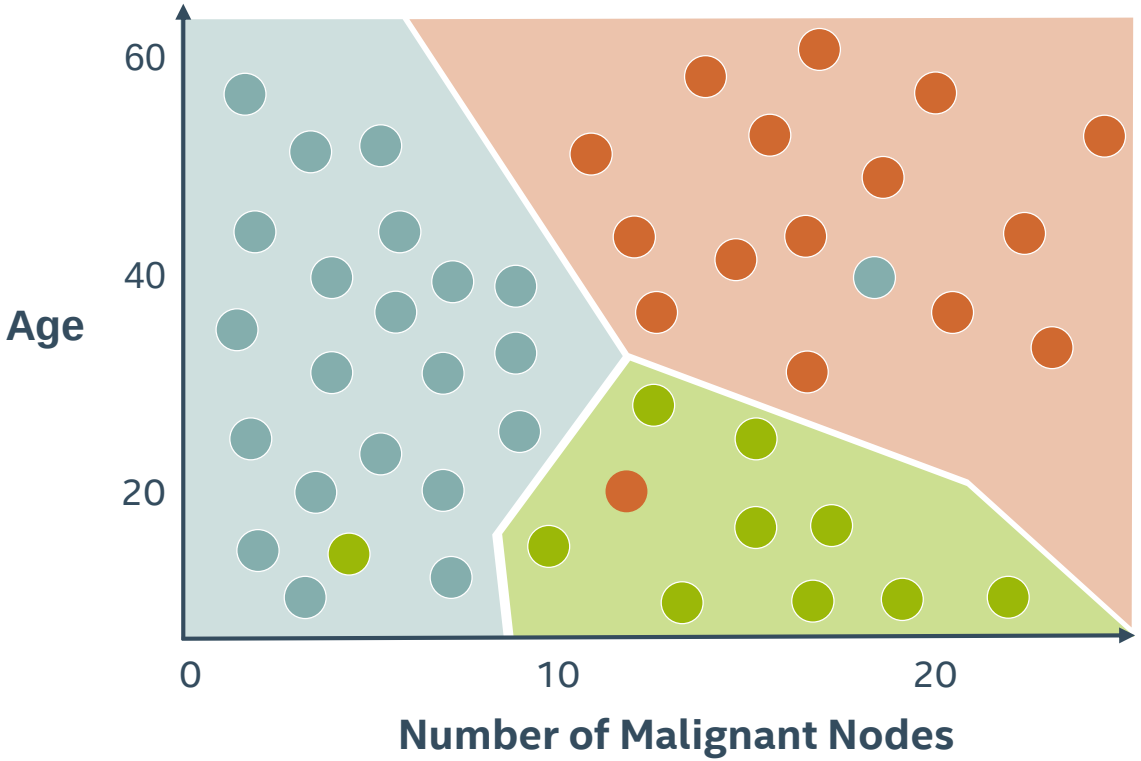
```
StdSc = StdSc.fit(X_data)

X_scaled = KNN.transform(X_data)
```

Other scaling methods exist: MinMaxScaler, MaxAbsScaler.

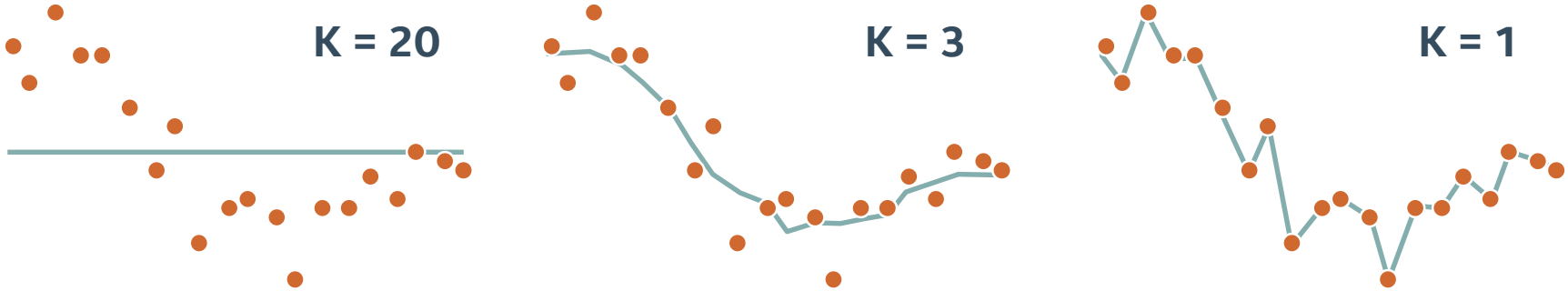
MULTICLASS KNN DECISION BOUNDARY

K=5

- Full remission
- Partial remission
- Did not survive



REGRESSION WITH KNN



CHARACTERISTICS OF A KNN MODEL

- Fast to create model because it simply stores data
- Slow to predict because many distance calculations
- Can require lots of memory if data set is large

K NEAREST NEIGHBORS: THE SYNTAX

Import the class containing the classification method

```
from sklearn.neighbors import KNeighborsClassifier
```

K NEAREST NEIGHBORS: THE SYNTAX

Import the class containing the classification method

```
from sklearn.neighbors import KNeighborsClassifier
```

Create an instance of the class

```
KNN = KNeighborsClassifier(n_neighbors=3)
```

K NEAREST NEIGHBORS: THE SYNTAX

Import the class containing the classification method

```
from sklearn.neighbors import KNeighborsClassifier
```

Create an instance of the class

```
KNN = KNeighborsClassifier(n_neighbors=3)
```

Fit the instance on the data and then predict the expected value

```
KNN = KNN.fit(X_data, y_data)

y_predict = KNN.predict(X_data)
```

K NEAREST NEIGHBORS: THE SYNTAX

Import the class containing the classification method

```
from sklearn.neighbors import KNeighborsClassifier
```

Create an instance of the class

```
KNN = KNeighborsClassifier(n_neighbors=3)
```

Fit the instance on the data and then predict the expected value

```
KNN = KNN.fit(X_data, y_data)

y_predict = KNN.predict(X_data)
```

The fit and predict/transform syntax will show up throughout the course.

K NEAREST NEIGHBORS: THE SYNTAX

Import the class containing the classification method

```
from sklearn.neighbors import KNeighborsClassifier
```

Create an instance of the class

```
KNN = KNeighborsClassifier(n_neighbors=3)
```

Fit the instance on the data and then predict the expected value

```
KNN = KNN.fit(X_data, y_data)

y_predict = KNN.predict(X_data)
```

Regression can be done with KNeighborsRegressor.

intel
Nervana™ AI Academy