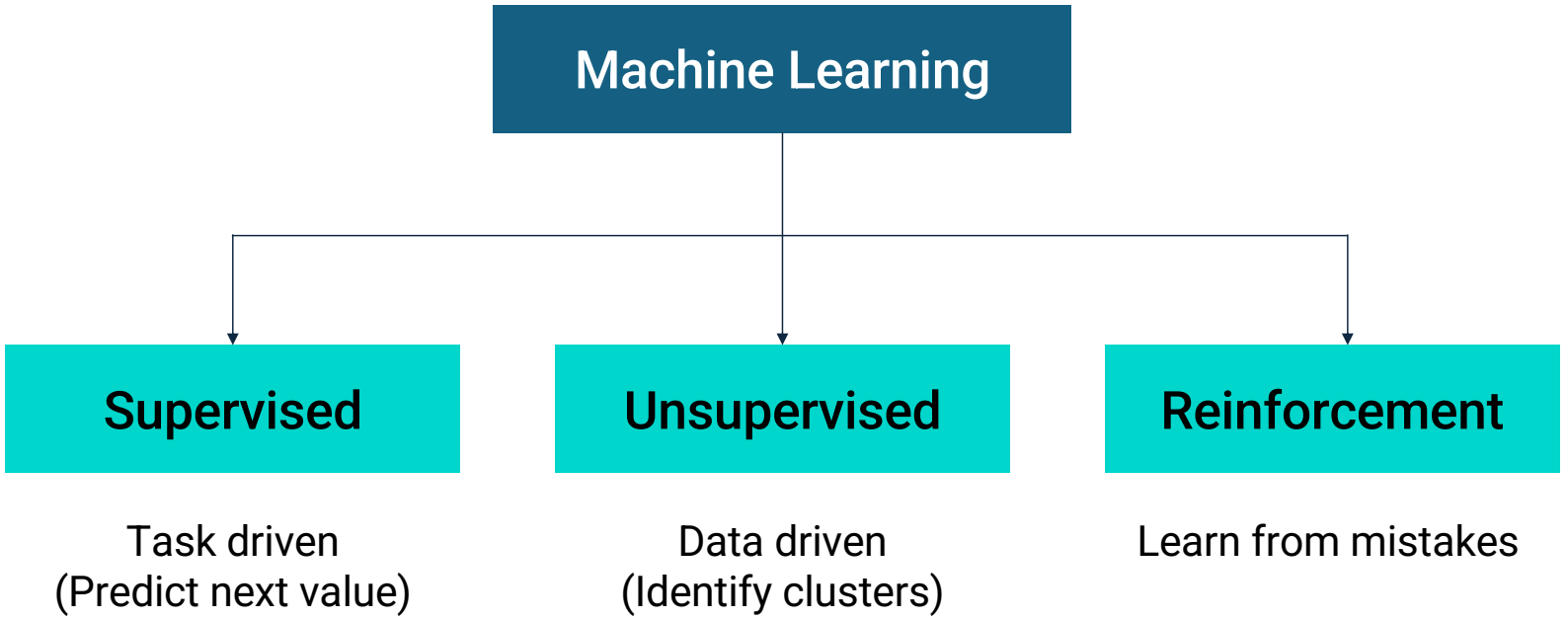
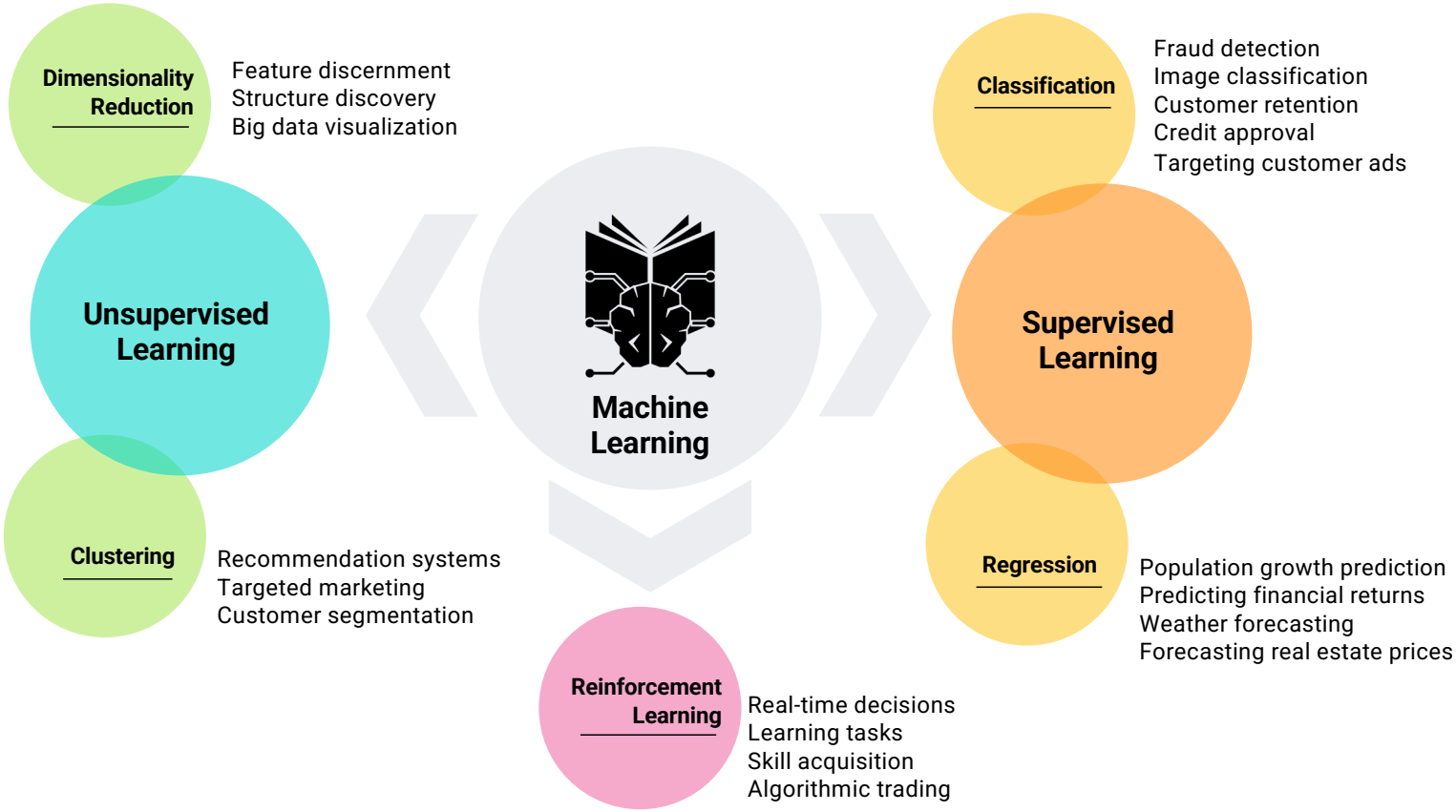
ML Overview for Data Engineers

Techcatalyst Data Engineering

Three Types of ML



Types of ML


Dimensionality Reduction
Feature discernment
Structure discovery
Big data visualization
Unsupervised Learning
Clustering
Recommendation systems
Targeted marketing
Customer segmentation
Machine Learning
Reinforcement Learning
Real-time decisions
Learning tasks
Skill acquisition
Algorithmic trading
Classification
Fraud detection
Image classification
Customer retention
Credit approval
Targeting customer ads
Supervised Learning
Regression
Population growth prediction
Predicting financial returns
Weather forecasting
Forecasting real estate prices

Types of ML

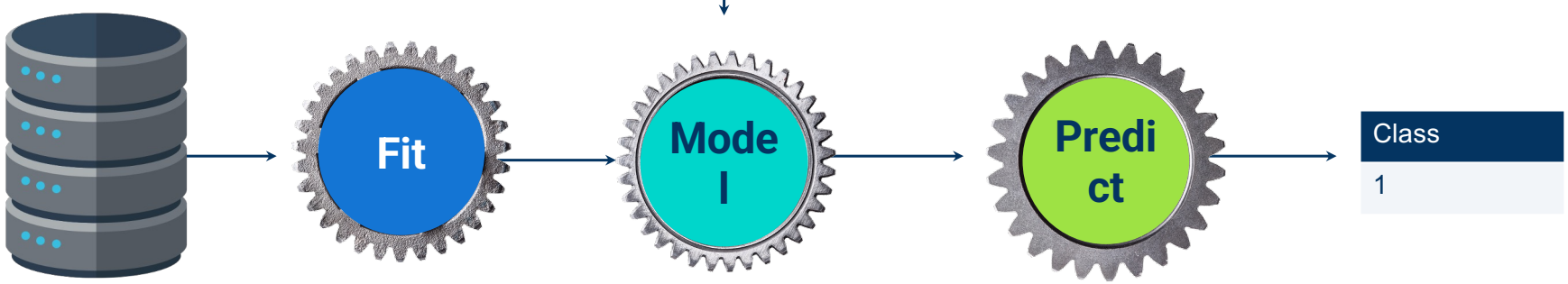
Most Python libraries for machine learning use a common interface to build and use machine learning models.

Training and Predicting

Regardless of the problem type, in machine learning, we follow a familiar paradigm: Model → Fit (Train) → Predict

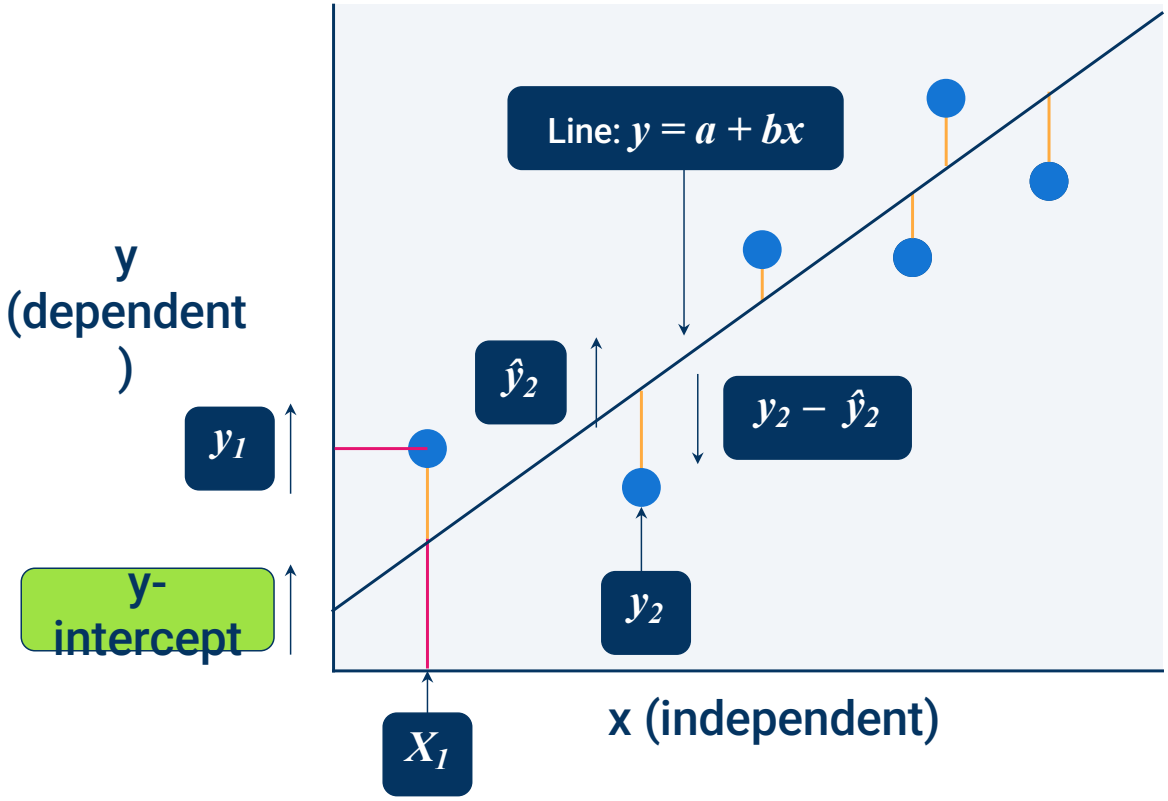
A	B	C	Class
11	16	22	1
10	8	4	2
…	…	…	…

A	B	C	Class
10	15	23	?



Class
1

Univariate Linear Regression Formula



Minimize:

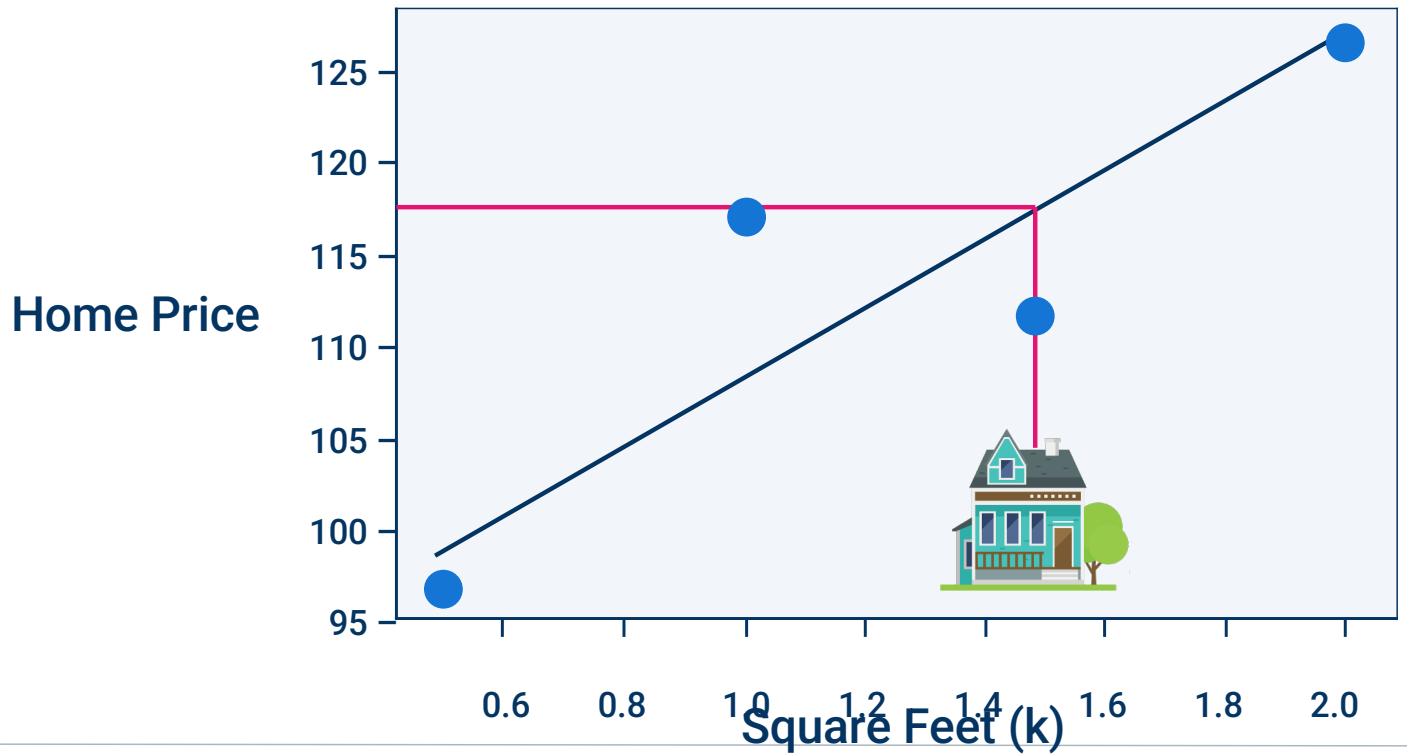
$$\sum_{i}^{n} (y_i - \hat{y}_i)2$$

Least squares method:

$$i = 1$$

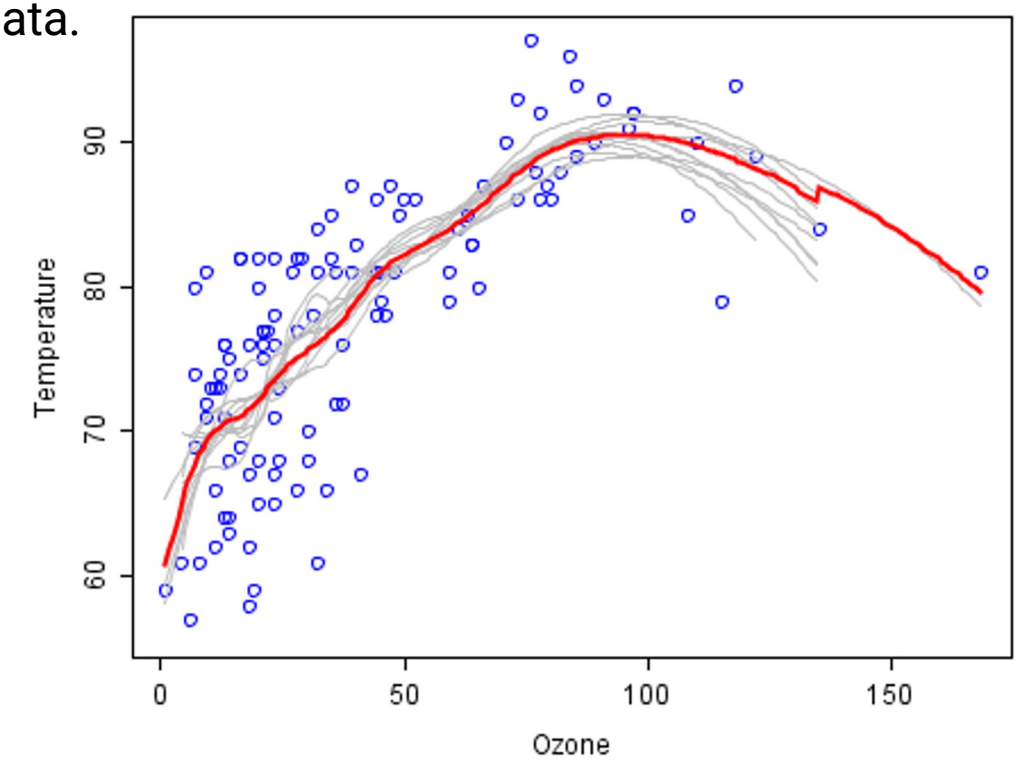
Univariate Linear Regression Formula

Example using linear regression to predict the home price:



Supervised Learning (Regression)

We'll be revisiting regression to predict the location of data points based on old data.



Quantifying Regression

Common Scoring Metrics:

R2 (R-Squared)

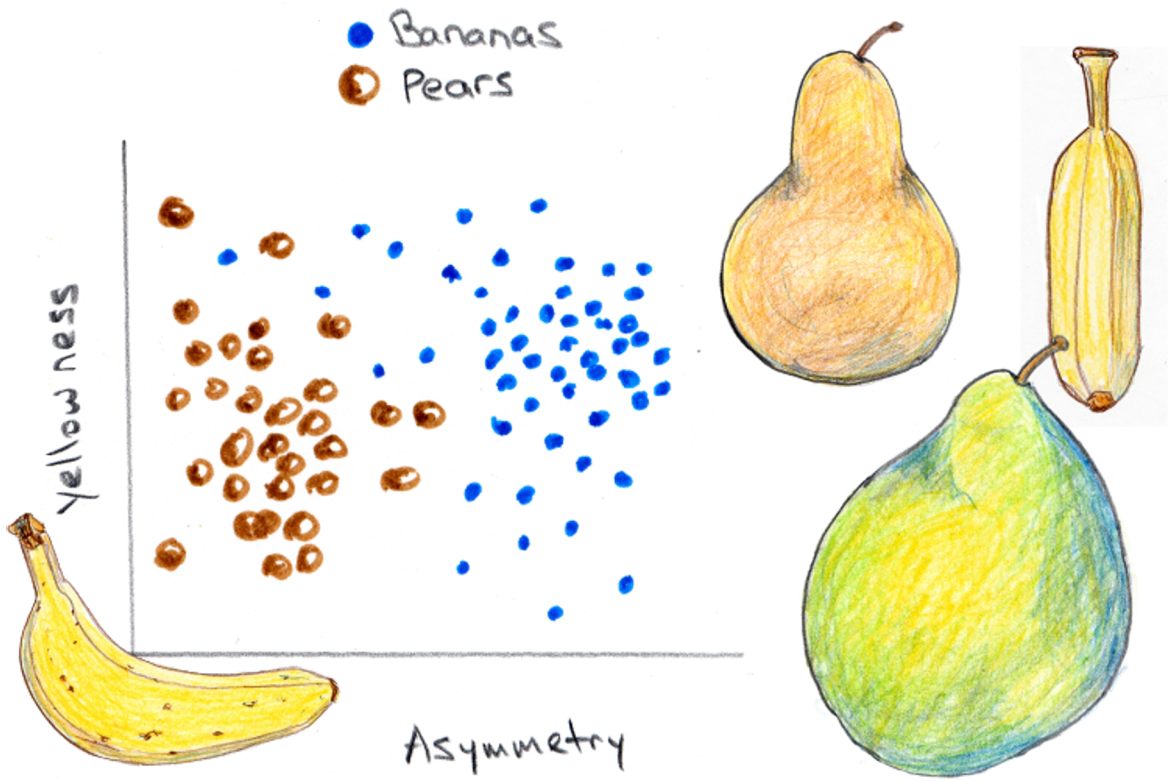
This is the baseline metric that many ML tools report on score. Higher R2 values signify that the model is “highly predictive.”

An R2 value of >0.90 means that our model roughly accounts for 90% of the variability of the data.

MSE (Mean Squared Error)

This measures the average of the squares of the errors or deviations.

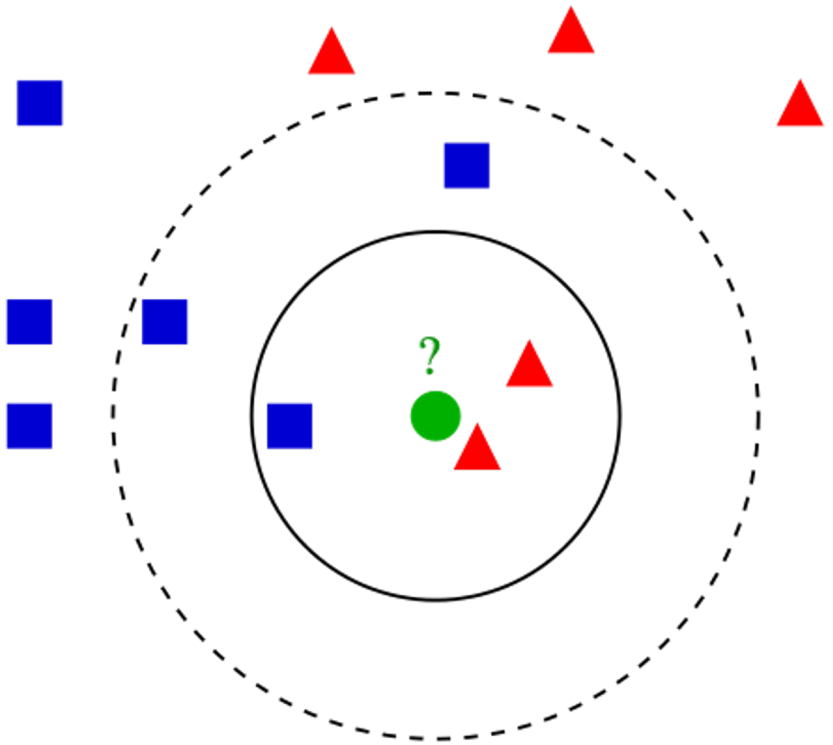
Supervised Learning (Classification)



KNN: Logistic Regression Friendly Neighbor

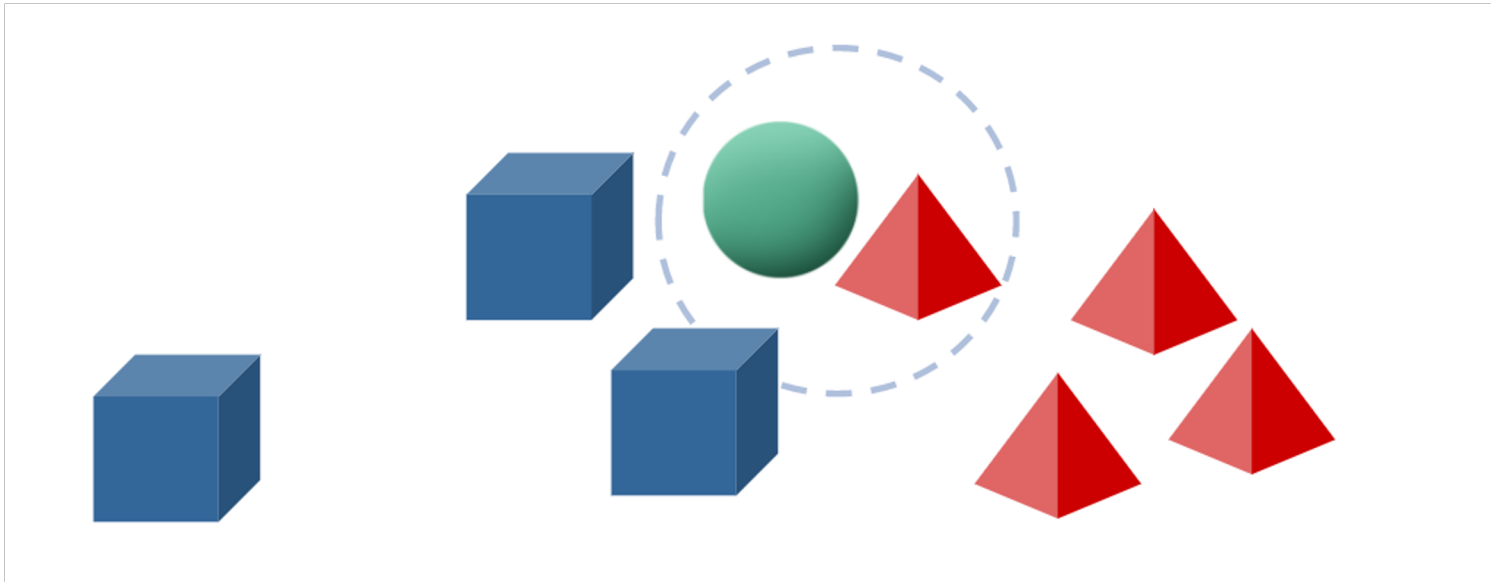
We set the k-value parameter to tell the algorithm to find the k-number of closest known data points.

Then, the algorithm determines what the majority of surrounding data points are classified to determine the class of the new data point.



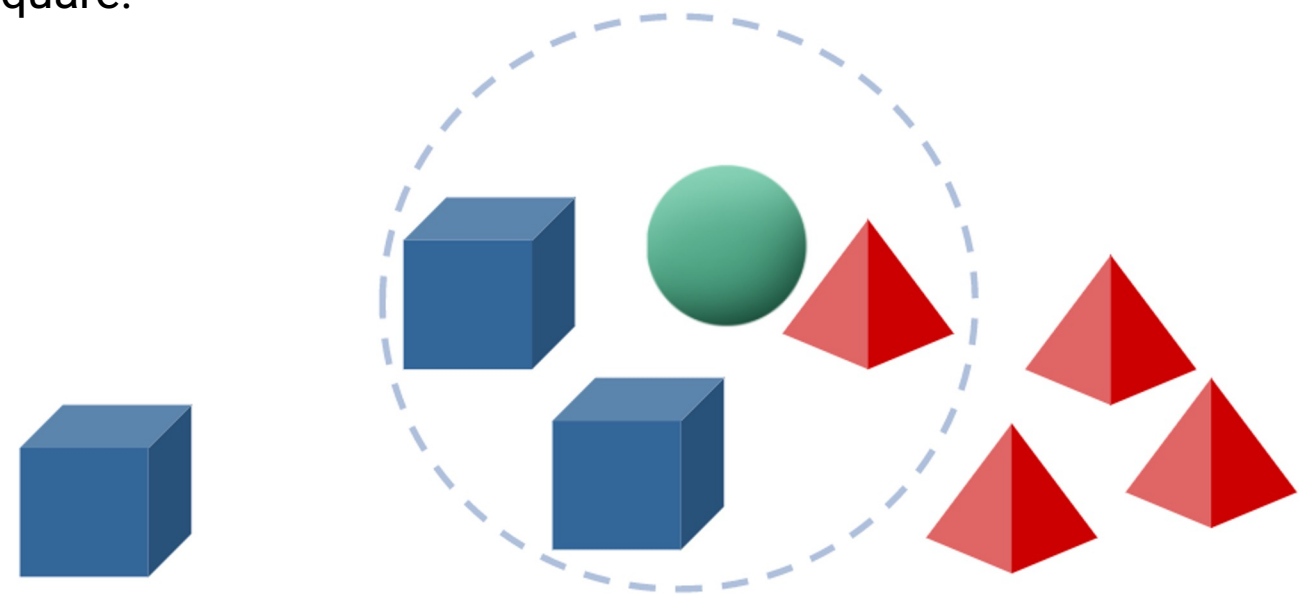
KNN: Friendly Neighbor

Consider the example we wanted to classify a new data point (green circle) from known square and triangle data points. If **k = 1**, then the algorithm classifies our new data point to whatever is the closest known single neighbor. In this case our green circle would be classified as a red triangle.



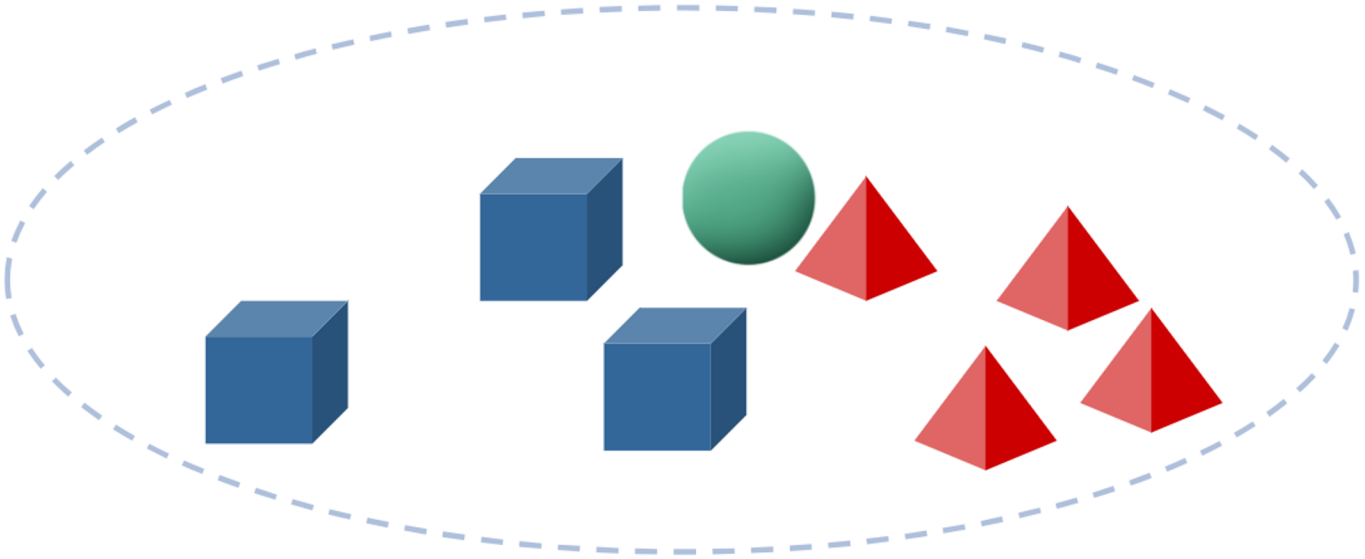
KNN: Friendly Neighbor

As the number of k-nearest neighbors increase, the distribution of nearest classifications can change. If in our example we used **k = 3**, there are two squares over the one triangle, so our model would classify the new data point as a blue square.



KNN: Friendly Neighbor

If we make our k-value too large, our k-nearest neighbor classification will classify all new data types as the most dominant classification. In this final part to our example, **k = 7** means that all known data points are considered and our green circle would be classified as a red triangle.



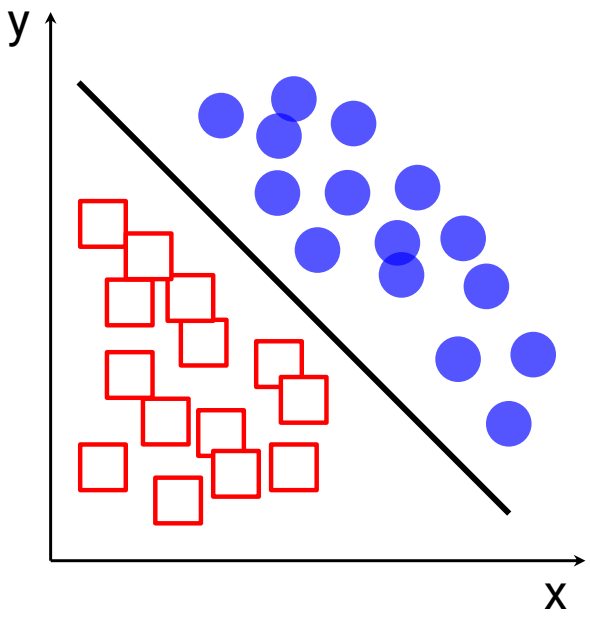
Confusion Matrix

A Confusion Matrix compares the predicted values from a model against the actual values. The entries of the confusion matrix are the number of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

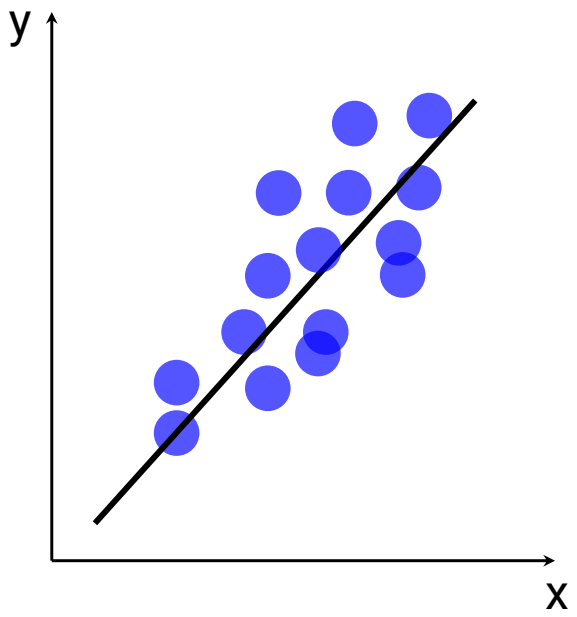
	Predicted True	**Predicted False**
Actually True	113 (True Positives)	12 (False Negatives)
Actually False	31 (False Positives)	36 (True Negatives)

Classification vs. Regression

Classification

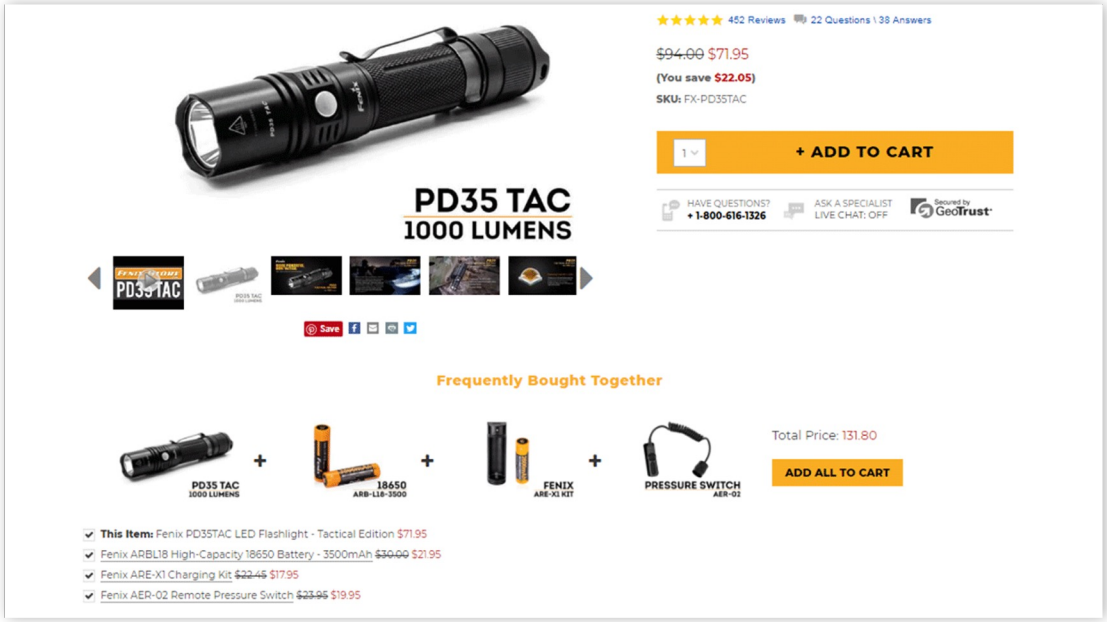


Regression



Introduction to Unsupervised Learning

For example, when you're reviewing a particular item for purchase on a website, unsupervised learning algorithms might be used to identify related items that are frequently bought together.



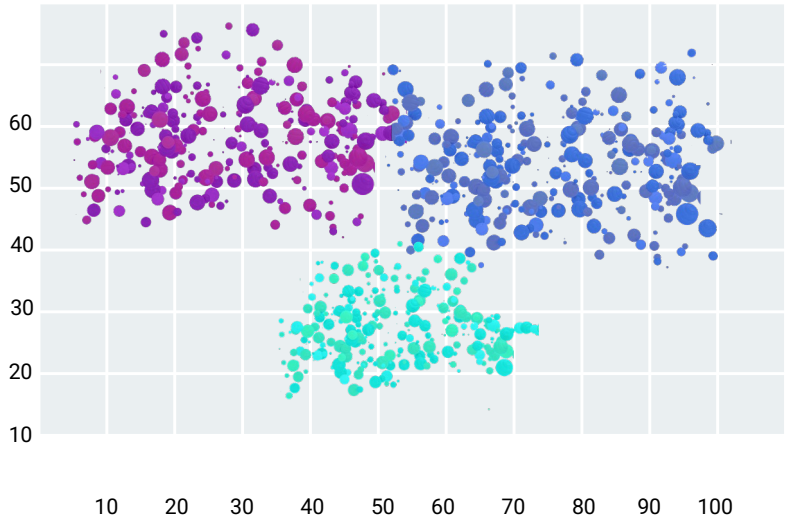
This power to recognize data patterns has broad applications in finance.

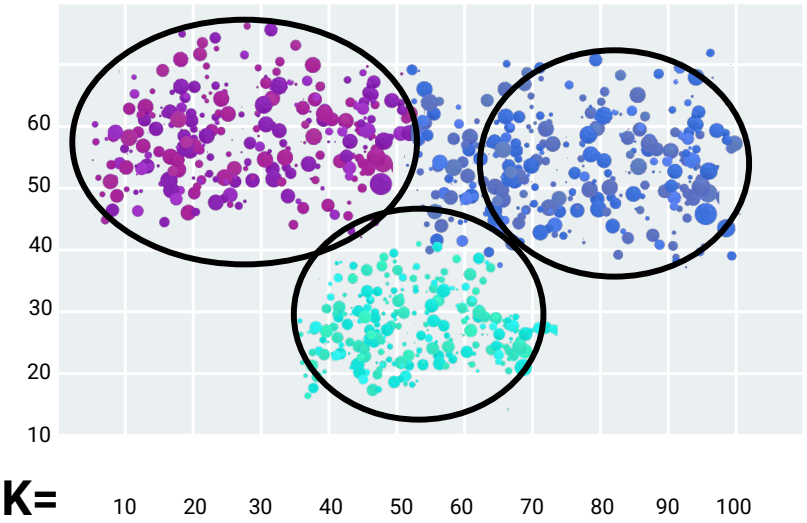
https://www.bigcommerce.com

Unsupervised learning can be used to **identify clusters**, or related groups, of clients to target with product offerings or marketing campaigns.

Clustering

In this clustering problem, we expect our algorithm to group data points based on their mutual similarities of features.

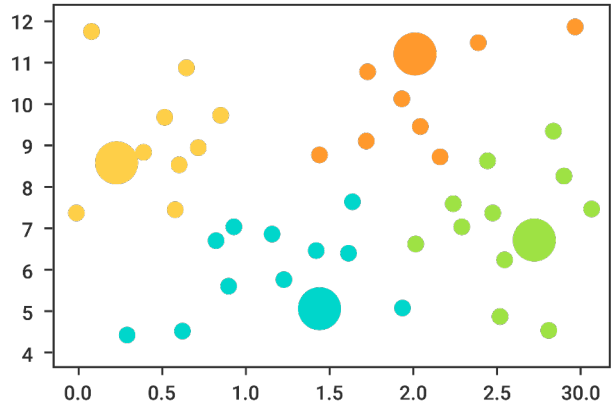




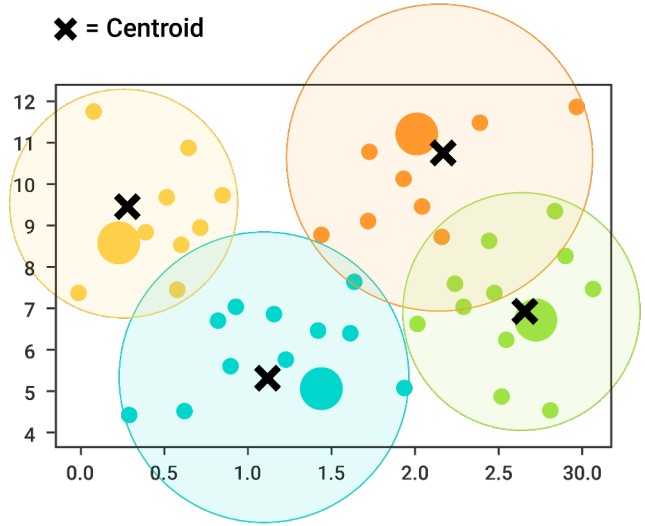
K= 3

The K-means Algorithm

K-means takes a predetermined amount of clusters and then assigns each data point to one of those clusters.



The algorithm assigns points to the closest cluster center.

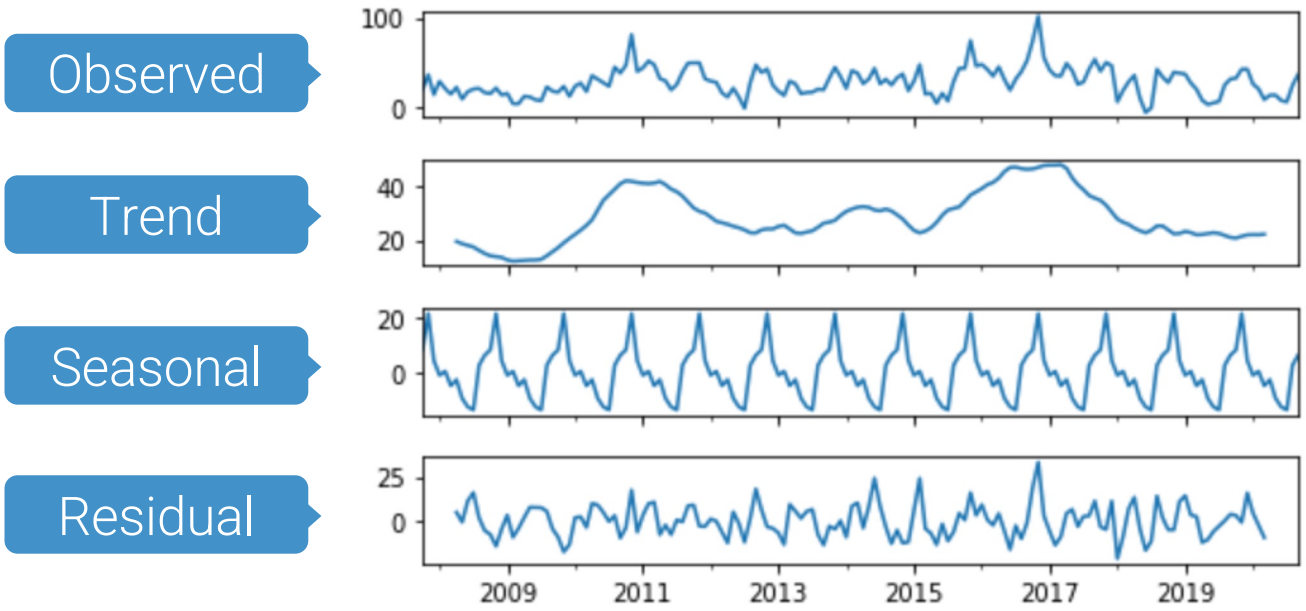


The algorithm readjusts the cluster's center by setting each center as the mean of all the data points contained within that cluster.


When analyzing time series, finding seasonal patterns is just one part of the job.

Identifying Patterned Relationships and Correlation

Another important task is identifying any relationships between time series patterns and determining if these relationships are predictable.



https://medium.datadriveninvestor.com

Data Preprocessing

Real life data almost always needs to be processed before it can be used in a machine learning algorithm

Preprocessing Data

Two major preprocessing steps are converting categorical data and scaling:

Converting Categorical Data

Categorical data is non-numeric data, like the day of the week or a person's education level, and needs to be converted to numeric data.

Scaling

Some machine learning algorithms are sensitive to large data values, so features need to be scaled to standardized ranges.

One-Hot Encoding and Label Encoding

Label Encoding

Label Encoding turns categorical variables into a series of integers, for example, “Sunday” becomes 0, “Monday” becomes 1, “Tuesday” becomes 2, and so on.

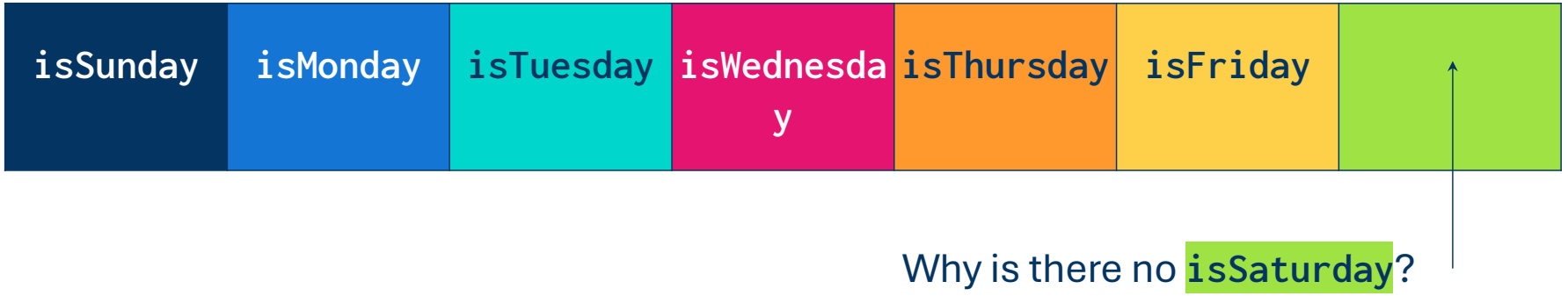
It can cause problems though, because the difference between Saturday and Sunday in our previous example is -6, but the difference for other consecutive days is +1.

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
0	1	2	3	4	5	6

One-Hot Encoding

One-Hot Encoding creates new “dummy” features for each category with 0 and 1 as Boolean values.

So the Weekday feature becomes 6 new features:



One-Hot Encoding

One-hot encoding involves taking a categorical variable, such as color, and creating three new variables of the colors. Each instance of the data now shows a **1** if it corresponds to that color, and **0** if it does not.

	A	B	C	D	E	F	G	H	I
1	Original data:			One-hot encoding Format					
2	id	Color		id	White	Red	Black	Purple	Gold
3	1	White		1	1	0	0	0	0
4	2	Red		2	0	1	0	0	0
5	3	Black		3	0	0	1	0	0
6	4	Purple		4	0	0	0	1	0
7	5	Gold		5	0	0	0	0	1

One-Hot Encoding with Pandas

The `get_dummies()` function.

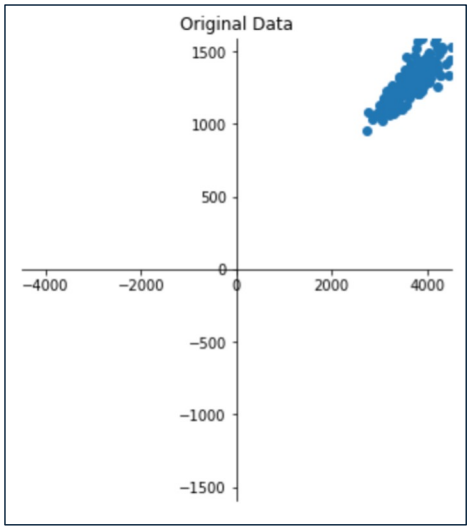
In Pandas, the `get_dummies()` function performs one-hot encoding. It can be applied to an entire DataFrame at once and returns a DataFrame of dummy-coded data.

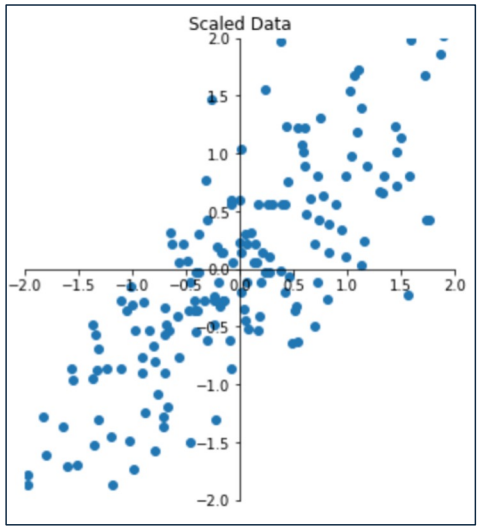
Setting the `drop_first` argument to `True` will automatically avoid the dummy trap.

Scaling/Normalization

We want all features to be shifted to similar numeric scales so that the magnitude of one feature doesn't bias the model during training.

`StandardScaler`	Scales data to have a mean of 0 and variance of 1. You should use `StandardScaler` when you do not have complete knowledge of your data.

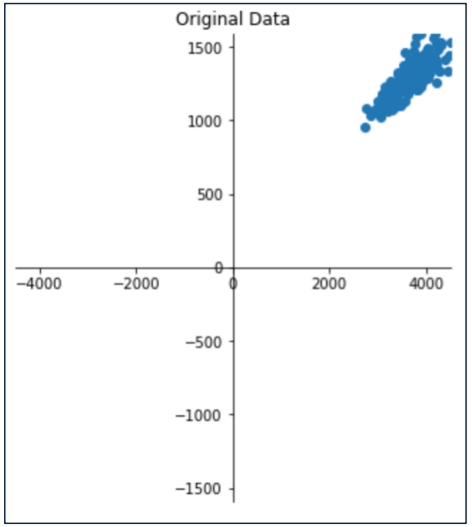


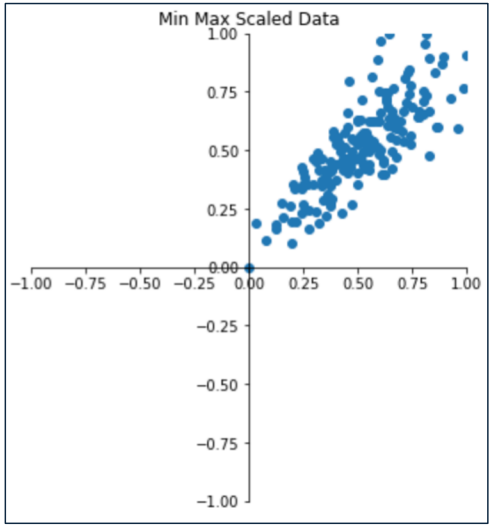


Scaling/Normalization

We want all features to be shifted to similar numeric scales so that the magnitude of one feature doesn't bias the model during training.

`MinMaxScaler`	`MinMaxScaler` is another scaler available in scikit-learn. It scales feature data to a minimum of 0 and a maximum of 1.





Normalizing Data

Remember, the K-means algorithm requires all the columns in a DataFrame to have numeric values.

- We should also ensure that the numeric values have the same scale.
- This is known as putting

Numeric Data Before Normalizing

eps	times_sales	total_assets	total_debt
2.61	63.73	222822.05	46244.82
0.12	17.55	234.42	0.00
7.96	44.14	239.78	15.24
-21.25	109.27	16872.89	0.00
62.48	387.85	156035.77	41128.51

The Same Data After Normalizing

eps	times_sales	total_assets	total_debt
-0.0575	-0.0797	-0.1134	-0.0864
-0.0570	0.0795	-0.1136	0.0864
-0.0594	-0.0796	-0.1136	-0.0864
-0.0567	-0.0770	0.1135	-0.0862
0.0484	0.2537	-0.0961	-0.0836

Scikit-learn's Preprocessing Paradigm

Preprocessors in scikit-learn follow the Fit -> Transform paradigm, similar to the Model -> Fit -> Predict paradigm for machine learning.

We fit our preprocessor (for example, `StandardScaler`) to training data. The preprocessor can be used to transform training data, testing data, or data to be predicted by a trained model.

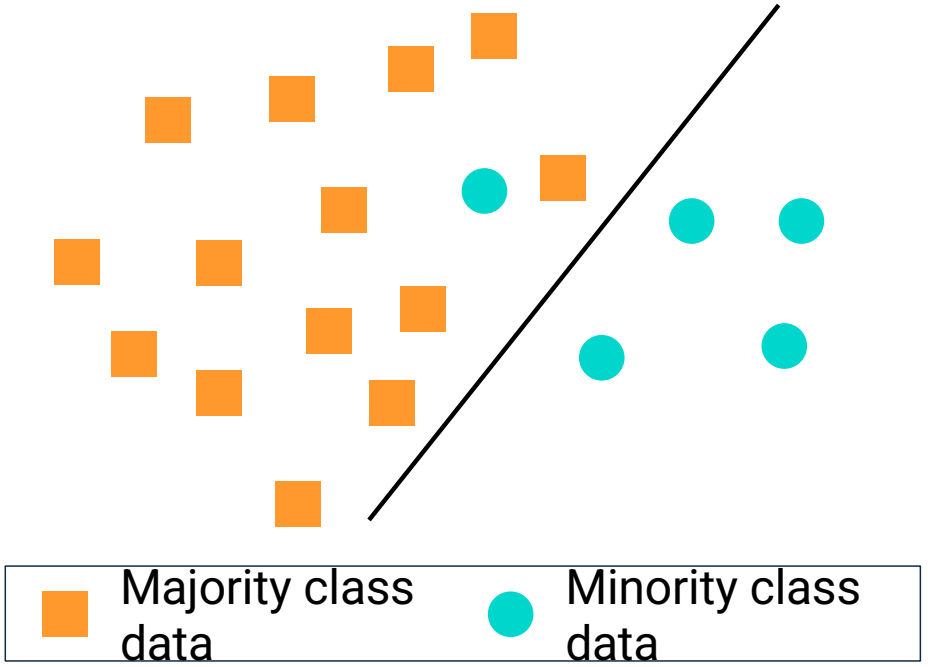
Imbalanced Data

Imbalanced data describes cases when one or more classes in the data are much more or much less frequent than the other class(es).

Imbalanced Data

Imbalanced data is problematic because it can cause your model to be biased toward the majority class.

- Basically, the model will be better at predicting the majority class as compared to the minority class because model fitting algorithms are designed to minimize the number of **total** incorrect classifications.
- If data is imbalanced, accuracy scores can be a misleading indicator of model quality.

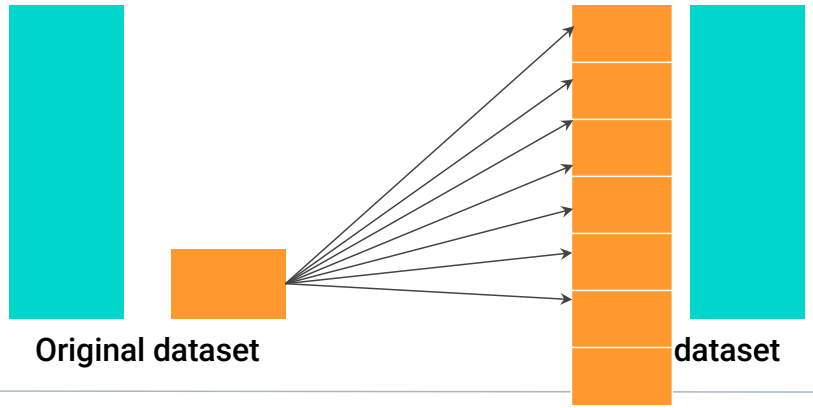


Imbalanced Data

The rest of the material will cover strategies for dealing with imbalanced classes. We will work mostly with two methods:

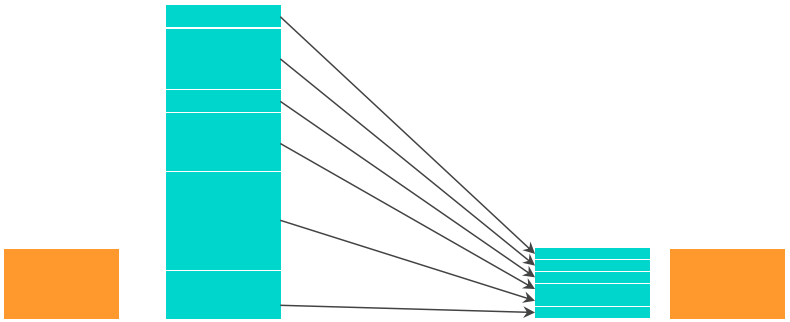
Oversampling

We sample the **minority class** with greater-than-random chance.



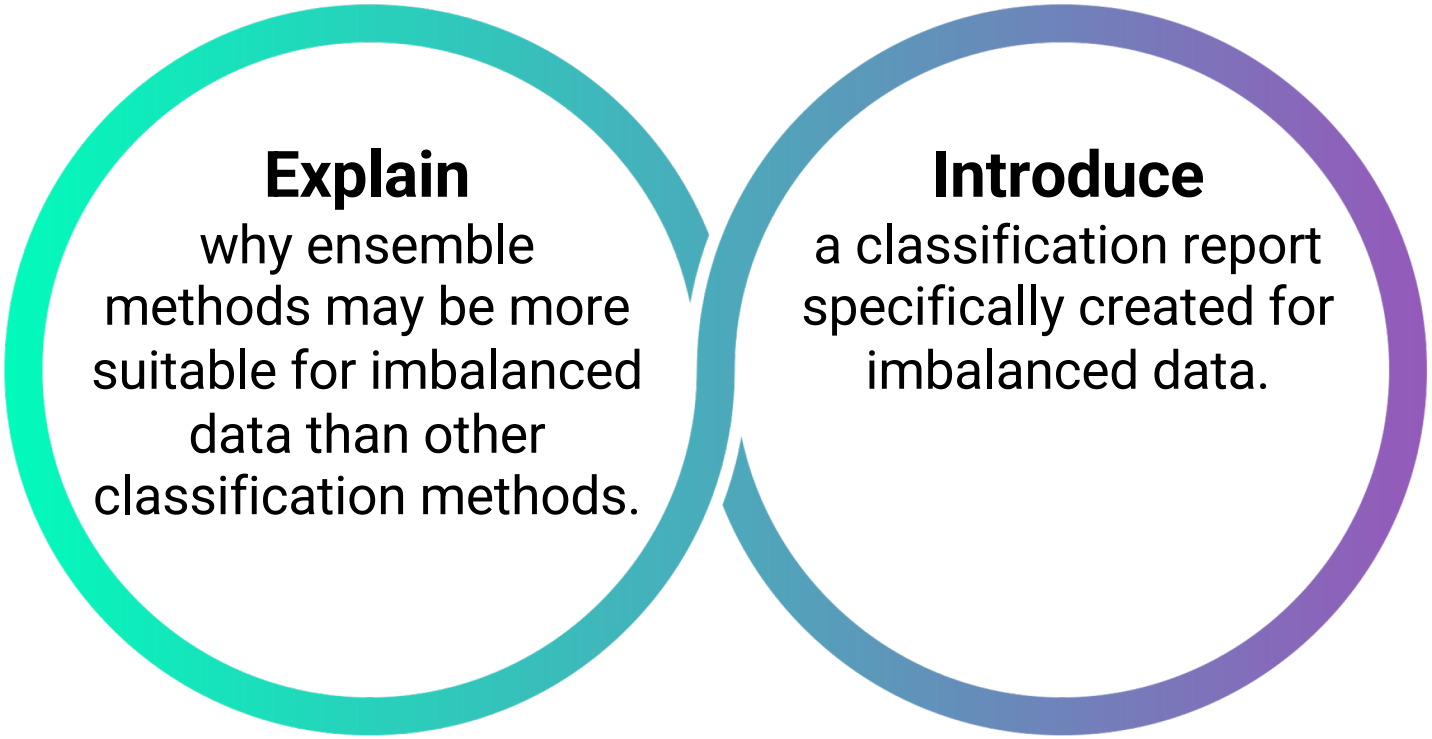
Undersampling

We sample the **majority class** with less-than-random chance.



Imbalanced Data

We will also:

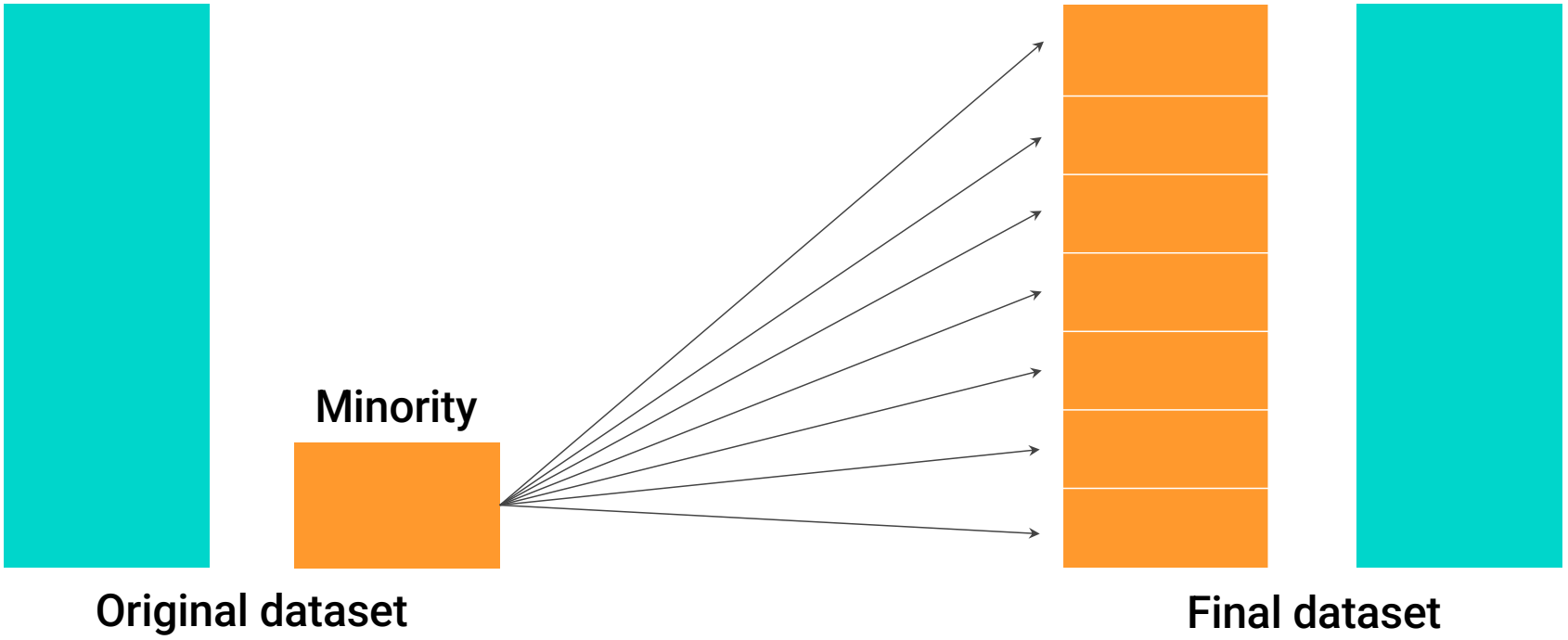


Random Sampling

Class imbalance refers to a situation in which the existing classes in a dataset aren’t equally represented.

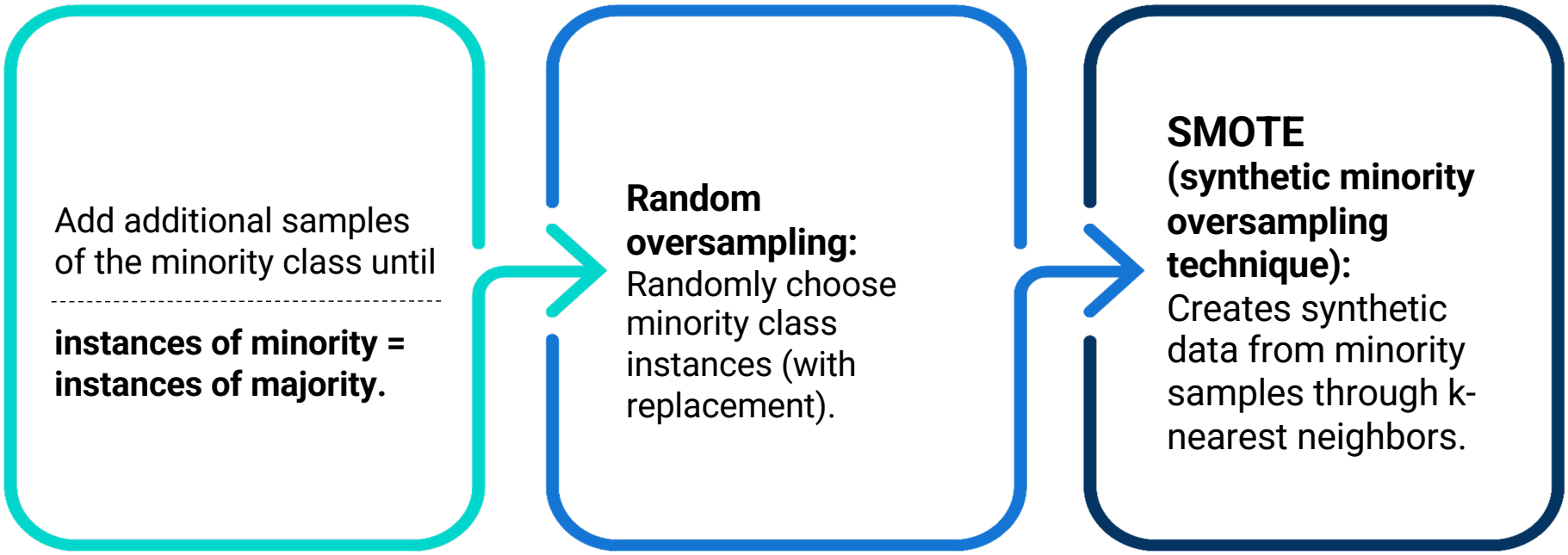
Oversampling

Creating more instances of a class label, usually for the smaller class.



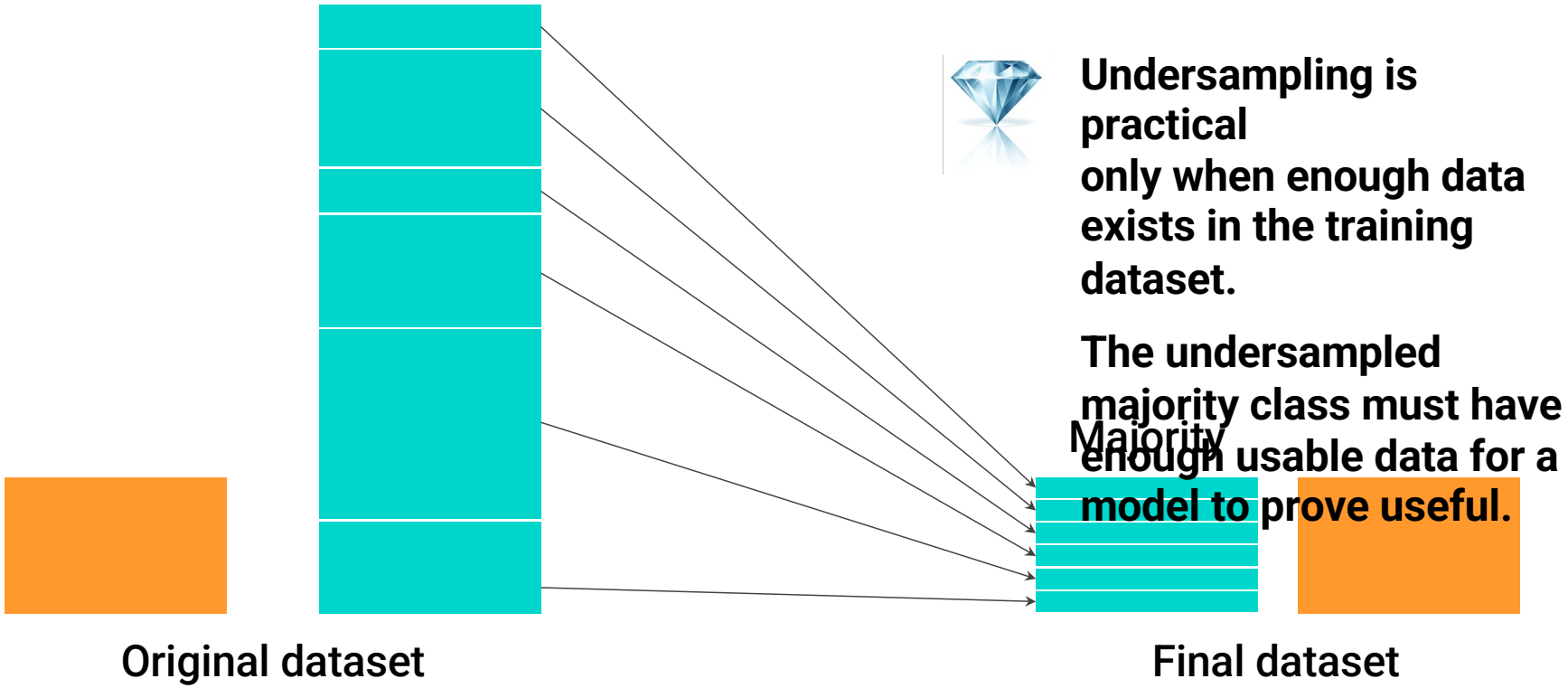
Oversampling

Potential strategies:



Undersampling

Creating fewer instances of a class label, usually for the larger class.

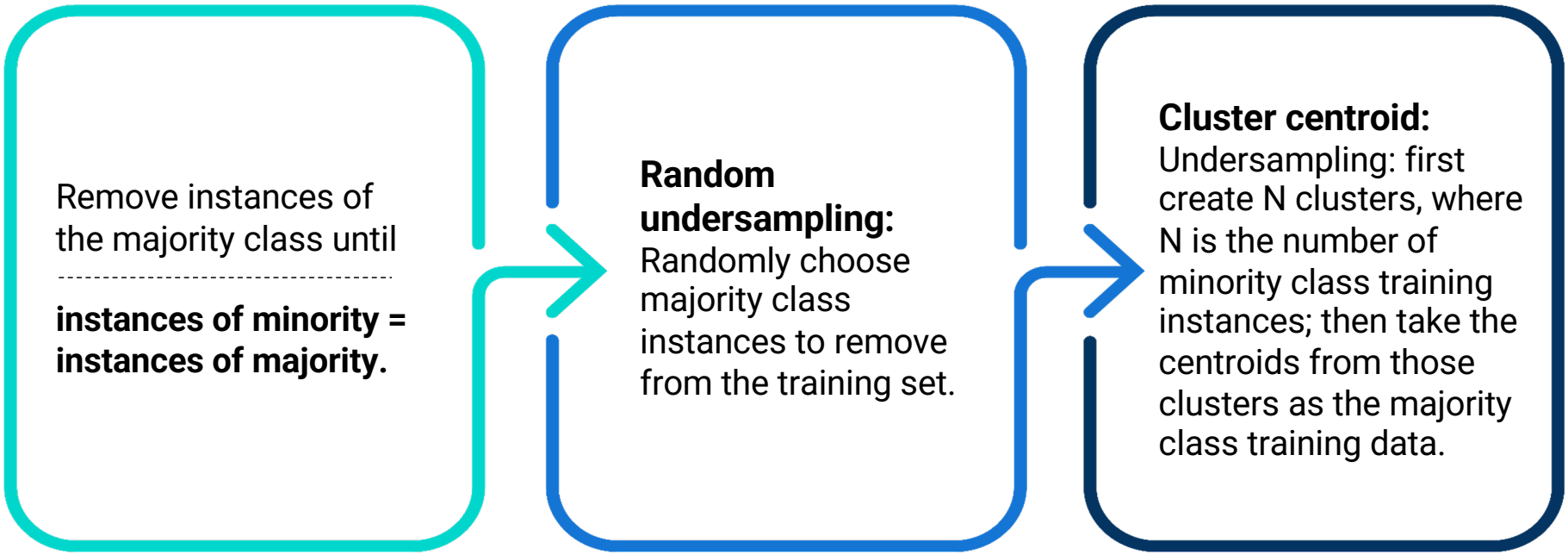


Undersampling is practical only when enough data exists in the training dataset.

The undersampled majority class must have enough usable data for a model to prove useful.

Undersampling

Potential strategies:



Random Sampling

Two methods that are commonly used to obtain new samples:

Random sampling

Our algorithm chooses **random instances** from the existing dataset.

We can use either oversampling or undersampling when sampling randomly, but we are using existing instances in our dataset and not creating new ones.

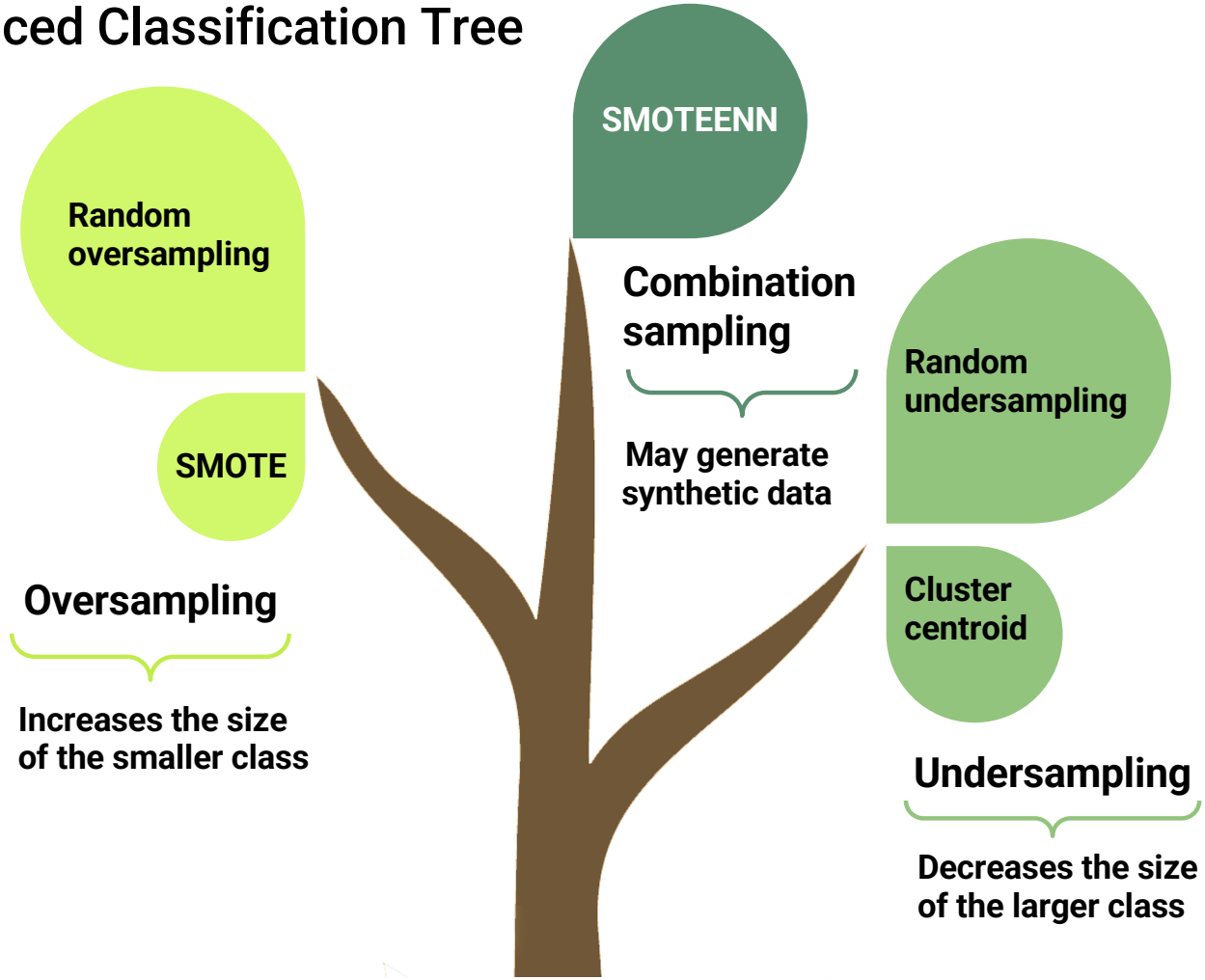
Synthetic sampling

Our algorithm generates **new instances** from observations about existing data.

In predicting loan defaults, we could use k-nearest neighbors to simulate the characteristics of a borrower who defaulted.

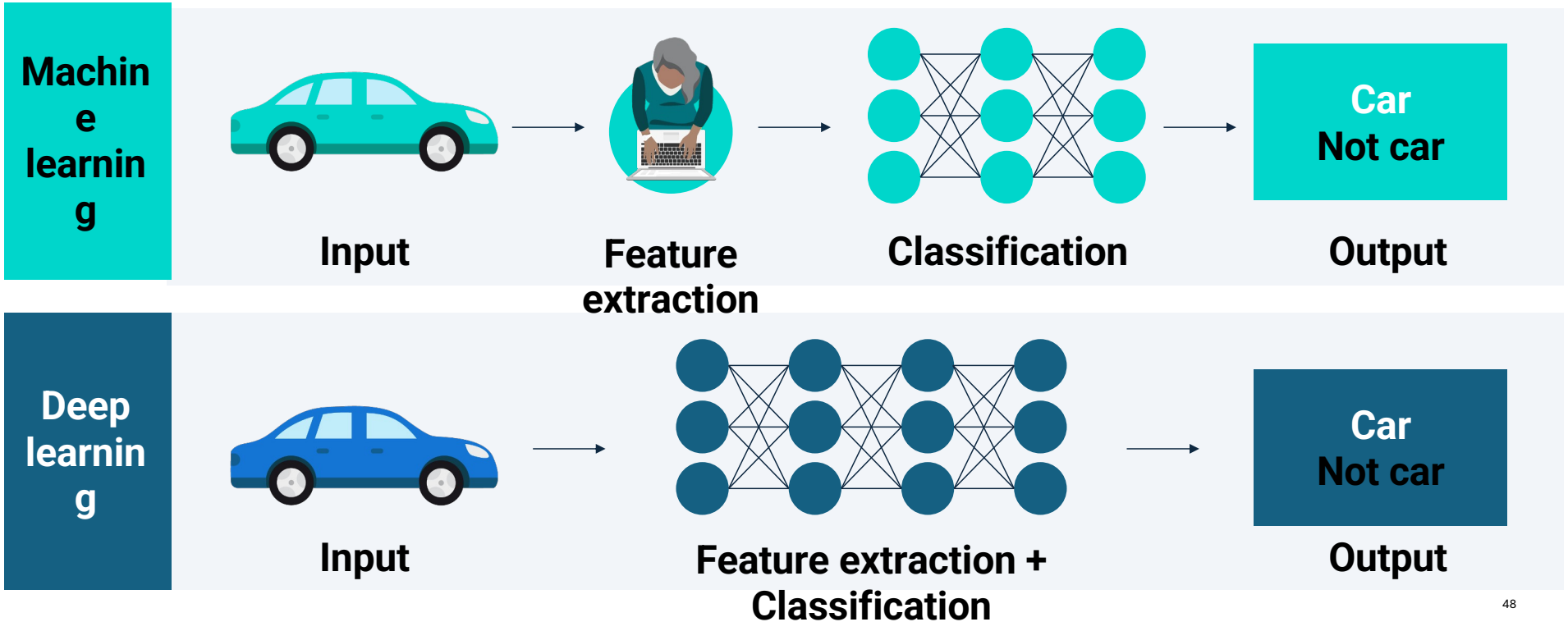
We would then add this simulated data to our original dataset.

The Imbalanced Classification Tree



Machine Learning vs. Deep Learning

Deep neural networks are much more effective than traditional machine-learning approaches at discovering nonlinear relationships among data.



Deep Neural Networks

Deep neural networks are often the best choice for complex or unstructured data like:

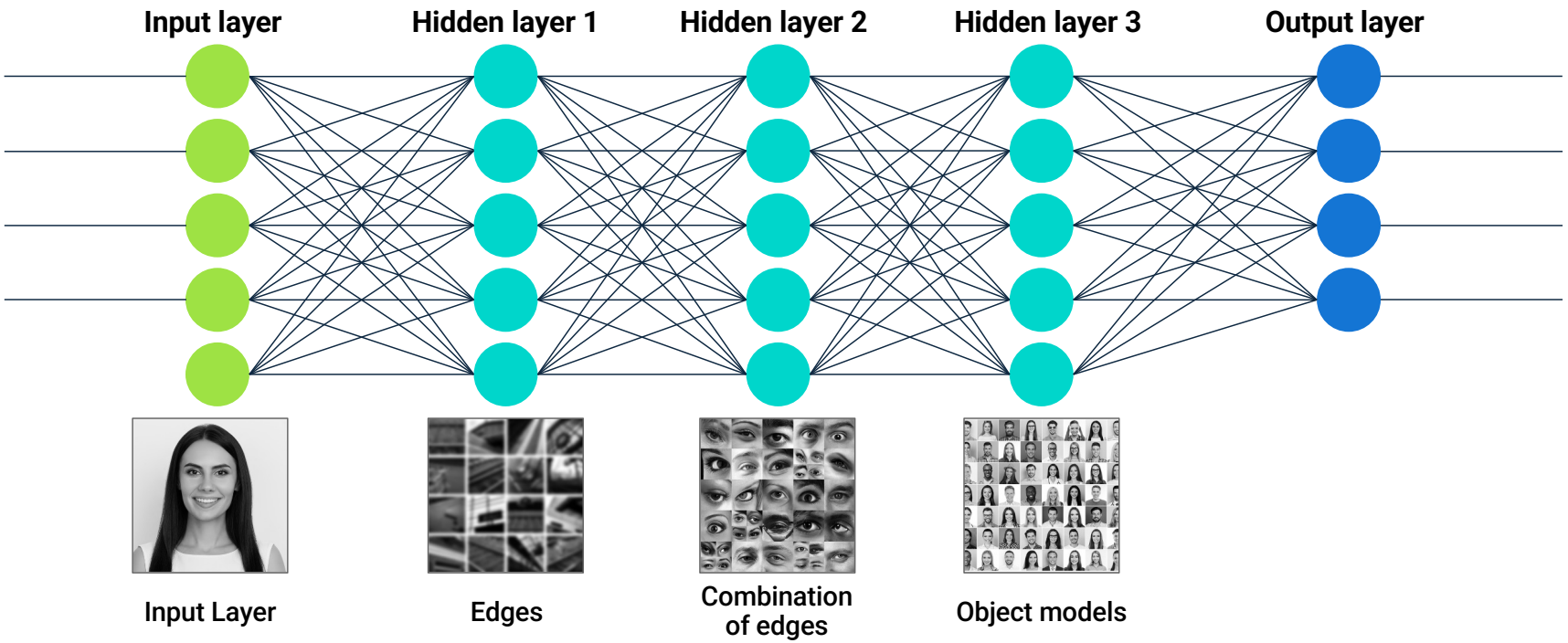
Image

Text

Voice

Deep Neural Networks

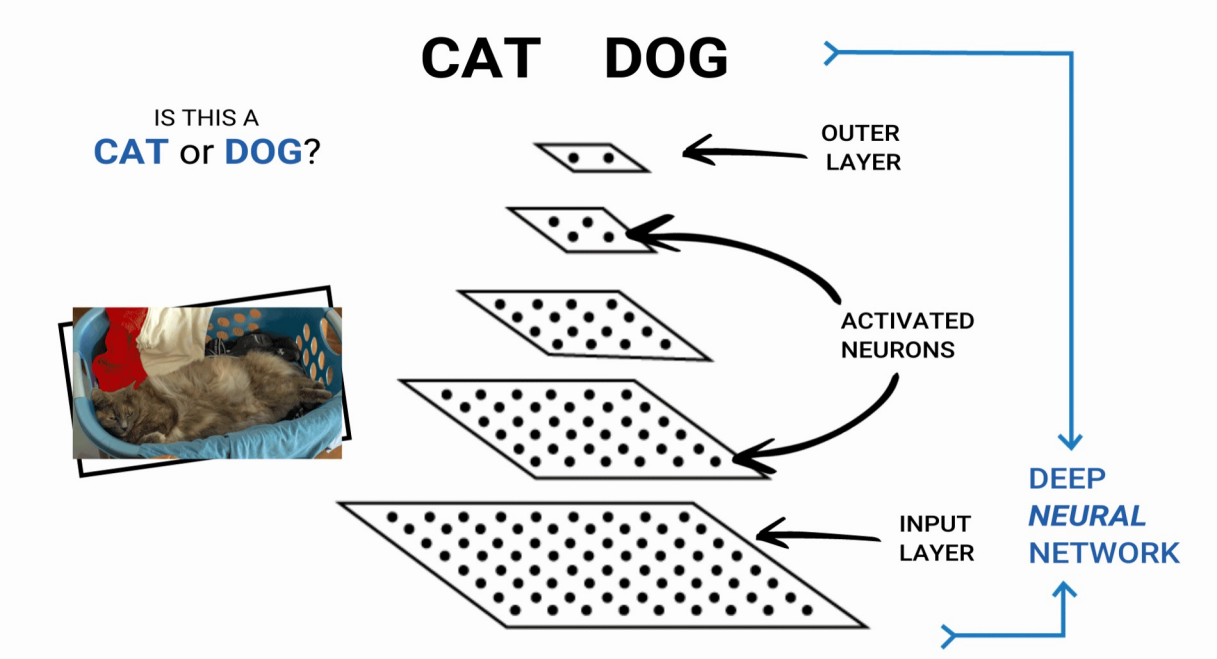
In image recognition, each layer can identify different image features in the process of defining or identifying the image.



Introduction to Deep Learning

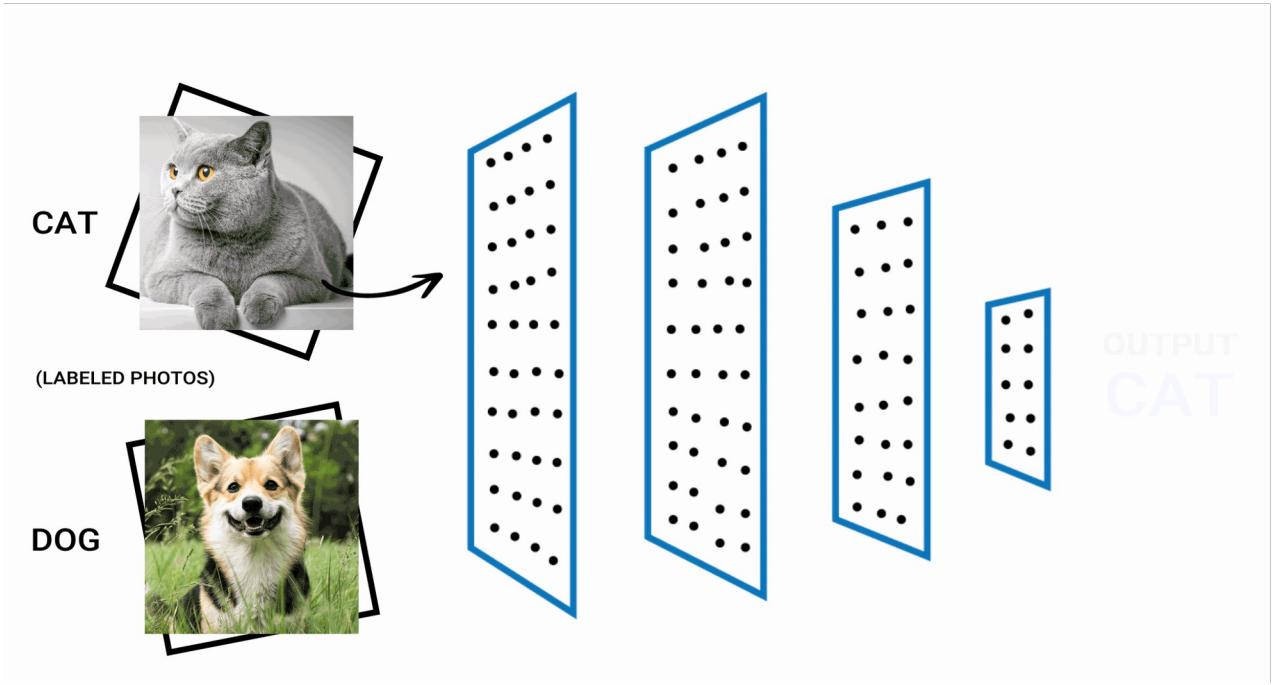
Introduction to Deep Learning

Neural networks calculate the weights of various input data and pass them to the next layer of neurons. This process continues until the data reaches the output layer, which makes the final decision on the predicted category or numerical value of an instance.



Introduction to Deep Learning

While definitions vary, we can consider neural networks with more than one hidden layer to be deep learning models. The decreasing cost and greater availability of computing power has increased our ability to create and use these models.



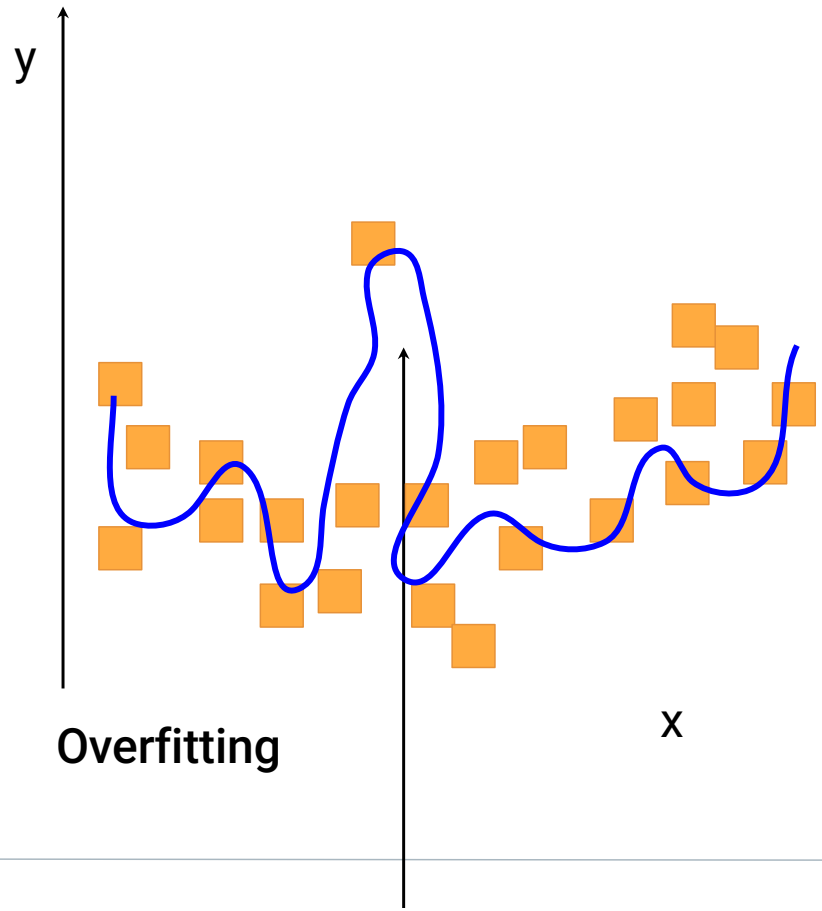

Overfitting results from
attempting to model the
training data too precisely.

Overfitting

Deep neural networks are beneficial because they can discover subtle, complex patterns in data that other models often can't find.

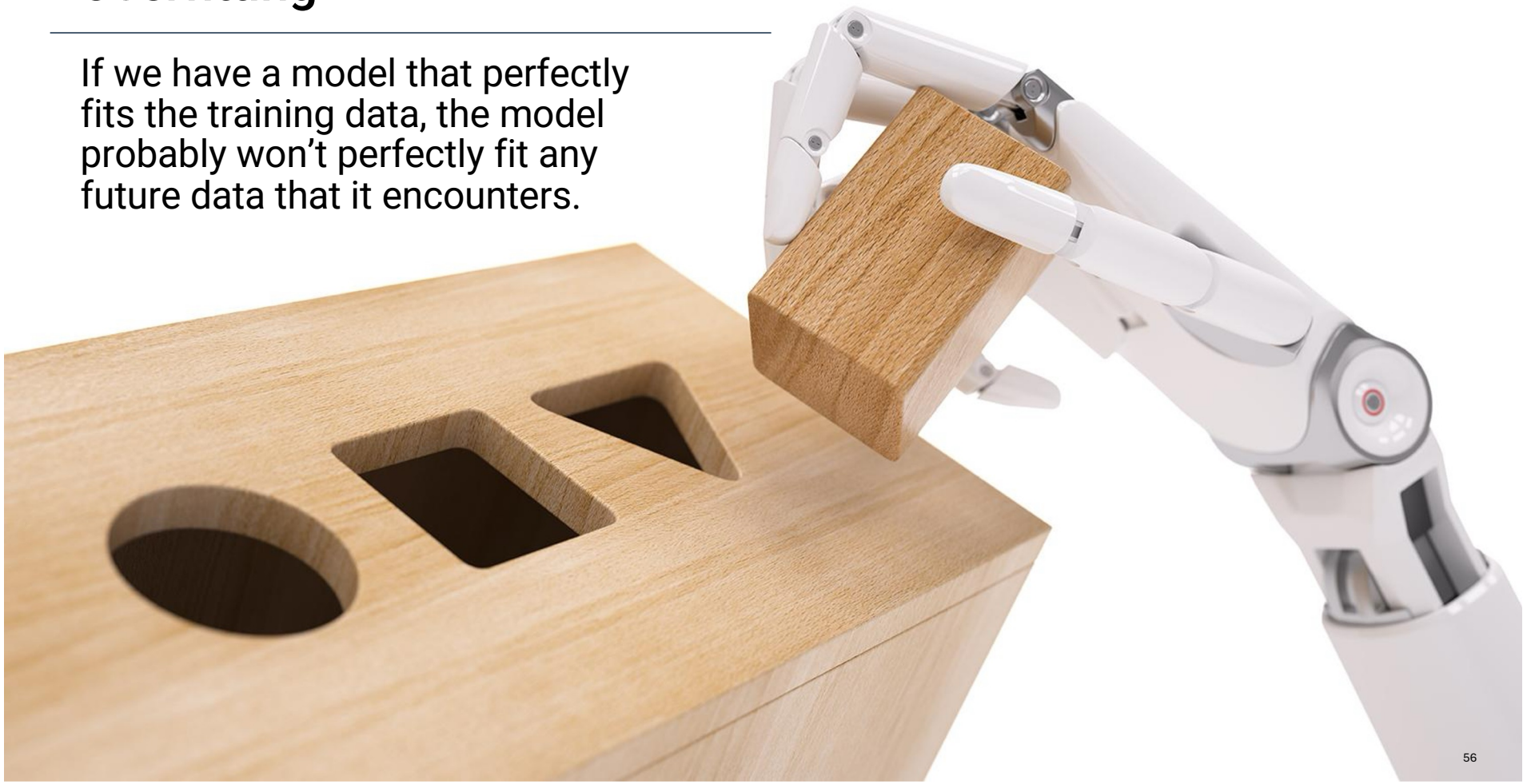
However, there's a limit to what we can learn about the data.

Random things that are unlikely to repeat— coincidences—can occur in the training data.



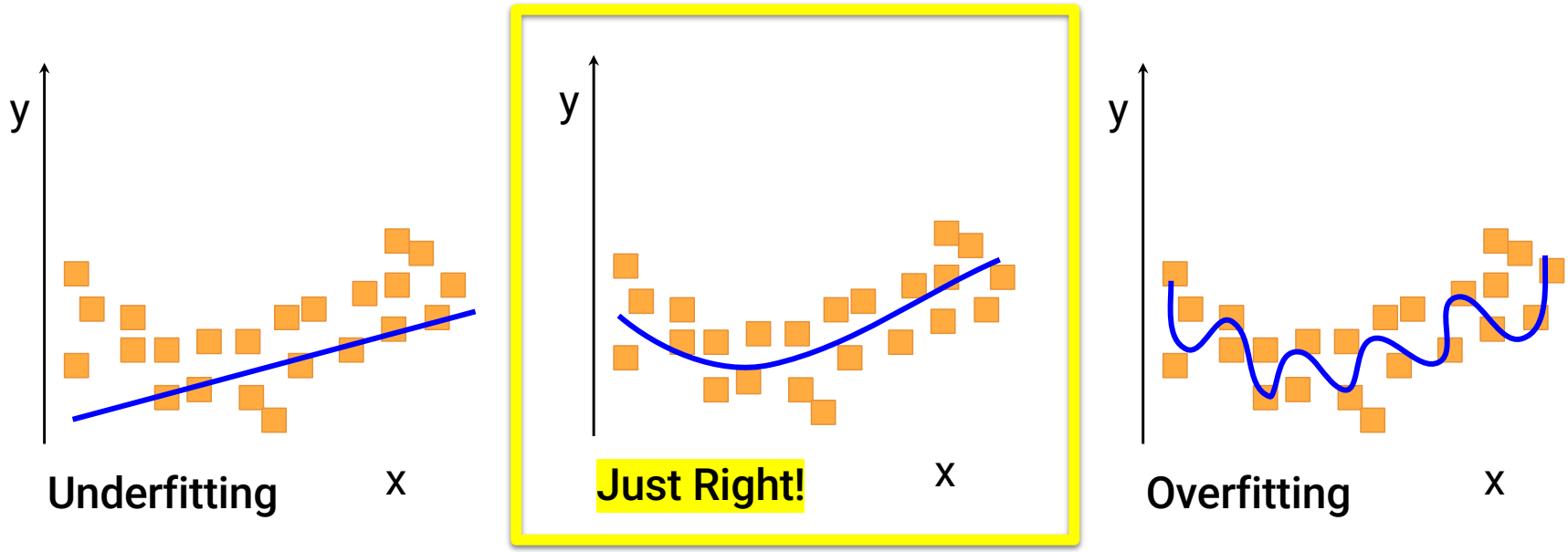
Overfitting

If we have a model that perfectly fits the training data, the model probably won't perfectly fit any future data that it encounters.



Overfitting

Sometimes, a complex model that is extremely well fit on the training data will perform worse on new data than a simpler model that used the same training data.



Techniques to Reduce Overfitting: Use of a Validation Set

A **validation set** is an additional dataset that we do not use for training the model. It's similar to the test dataset.

By monitoring the model's performance on the validation set as we continue to train the model, we can observe any increasing risk of overfitting.