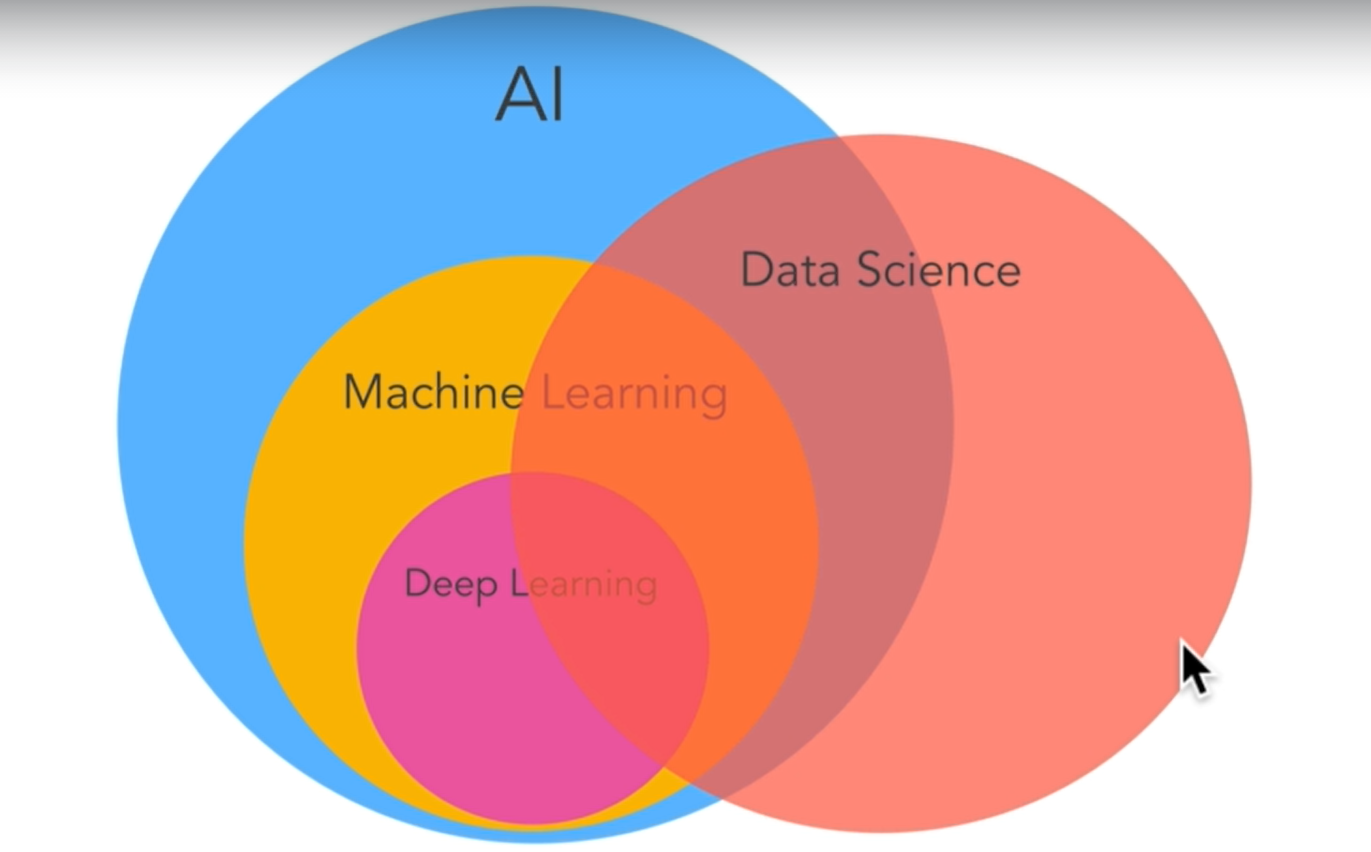
Machine Learning

# What is Machine Learning?

The goal is to make machine work like a human.

# What is AI/ML/Data Science



AI is also of two types:

1. Narrow AI – Currently present – Does a particular thing really well.
2. General AI – like humans, expert of all fields, not yet developed.

# What we are going to learn?



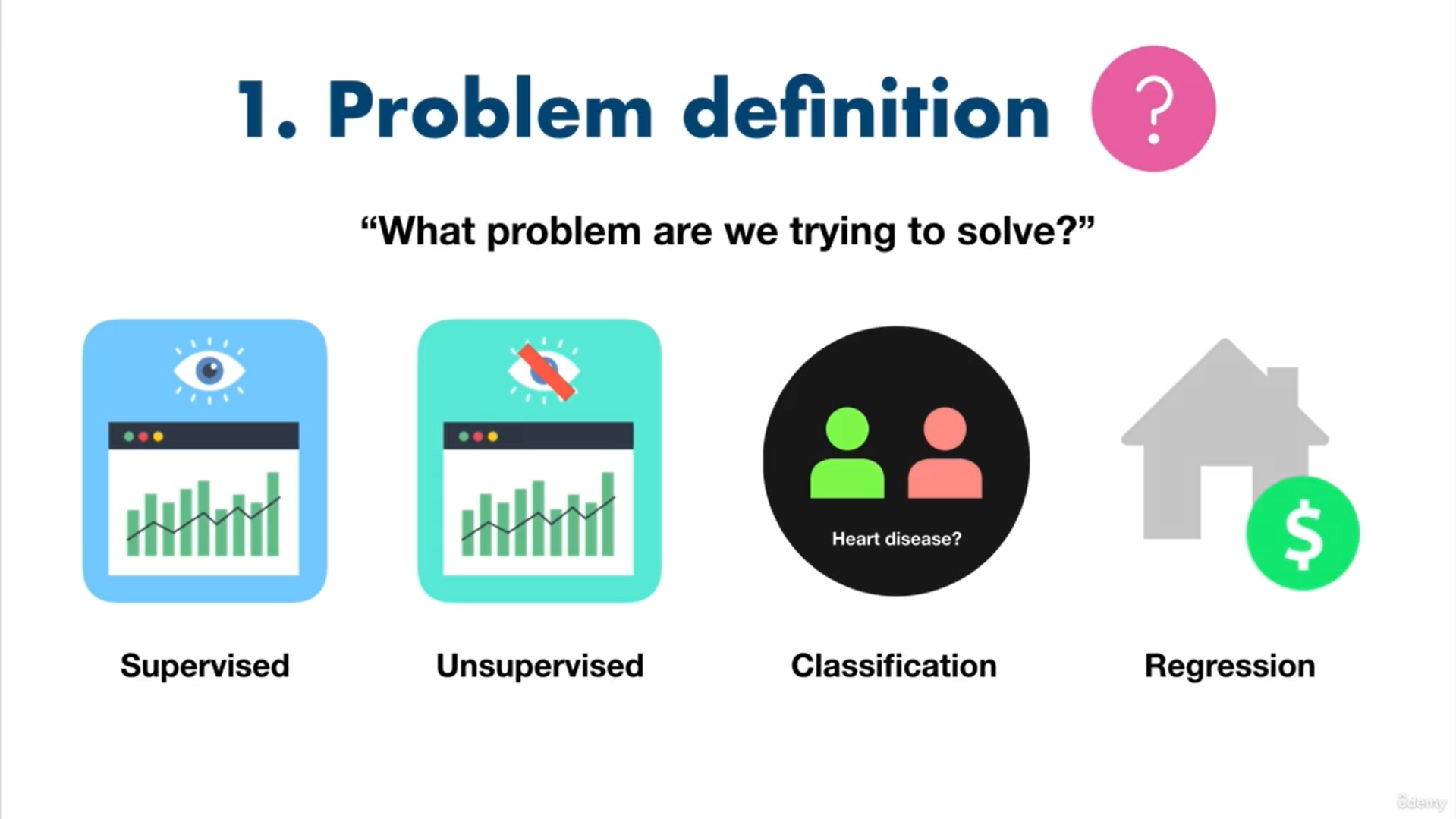
# Types of machine learning

1. Supervised ML: Here we have the dataset with features and labels, like in an excel sheet and we train a model based on that. Its like a teacher training a kid.
2. Unsupervised learning: Here we don’t have data with features and labels. The model will itself decide the clusters and associations.
3. Reinforcement learning: Its most commonly used in games, where the machine learns in real time. It is used for skill acquisition, etc.

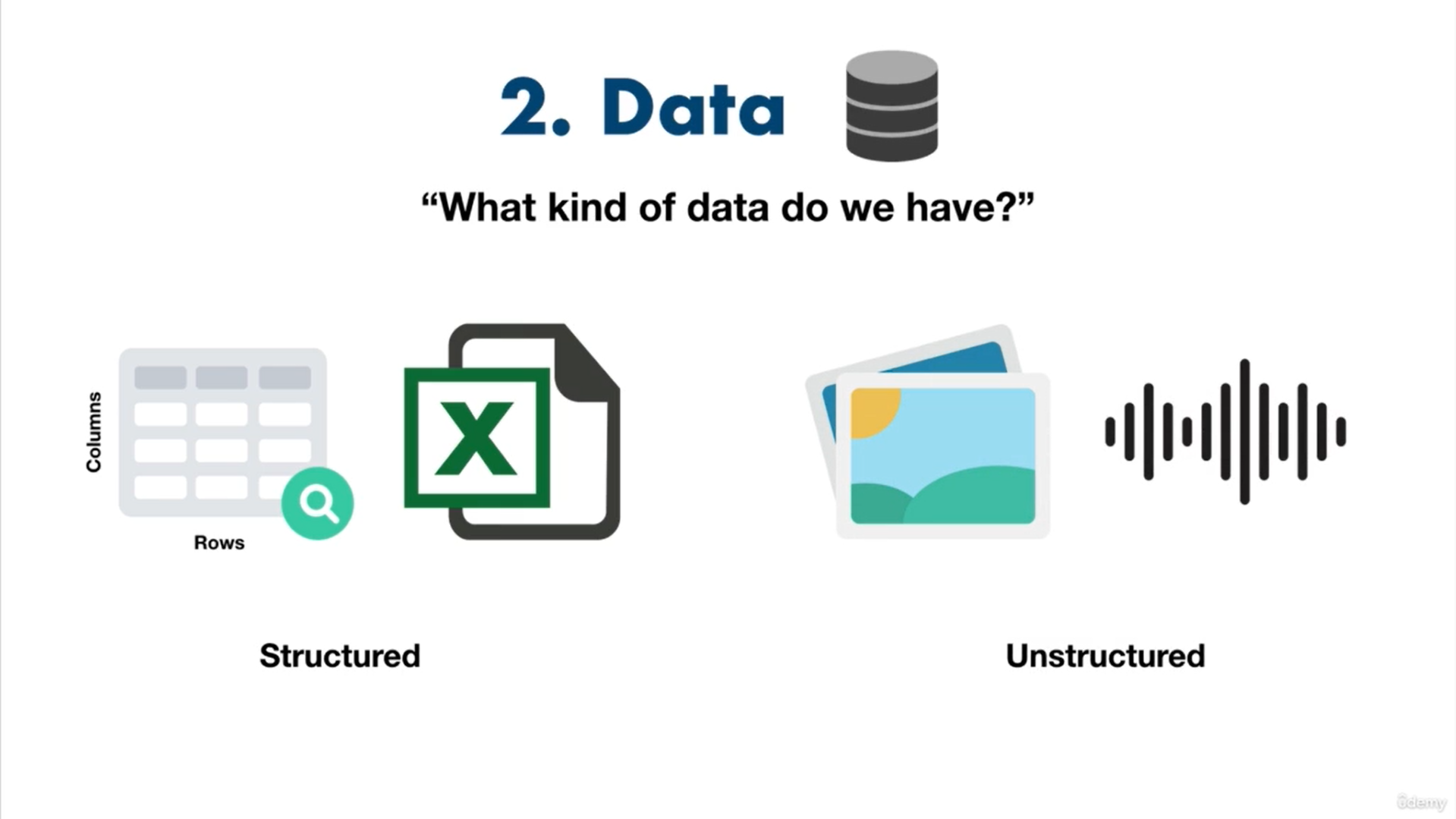
Machine learning is the art of figuring out instructions by giving an input and an ideal output. Traditional algorithms generate output given an input and set of instructions.

# Steps to start machine learning

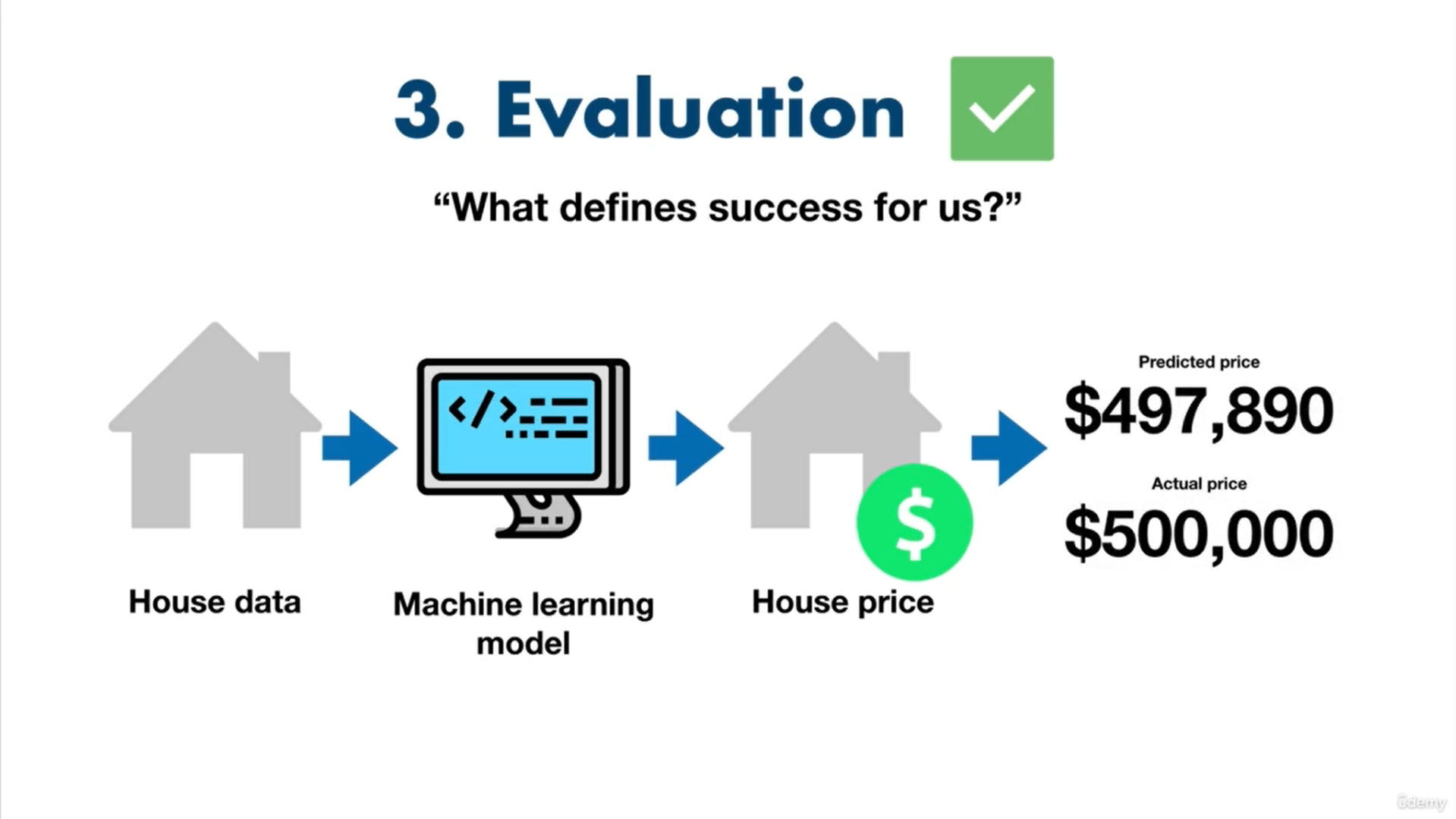
## Deciding on problem statement



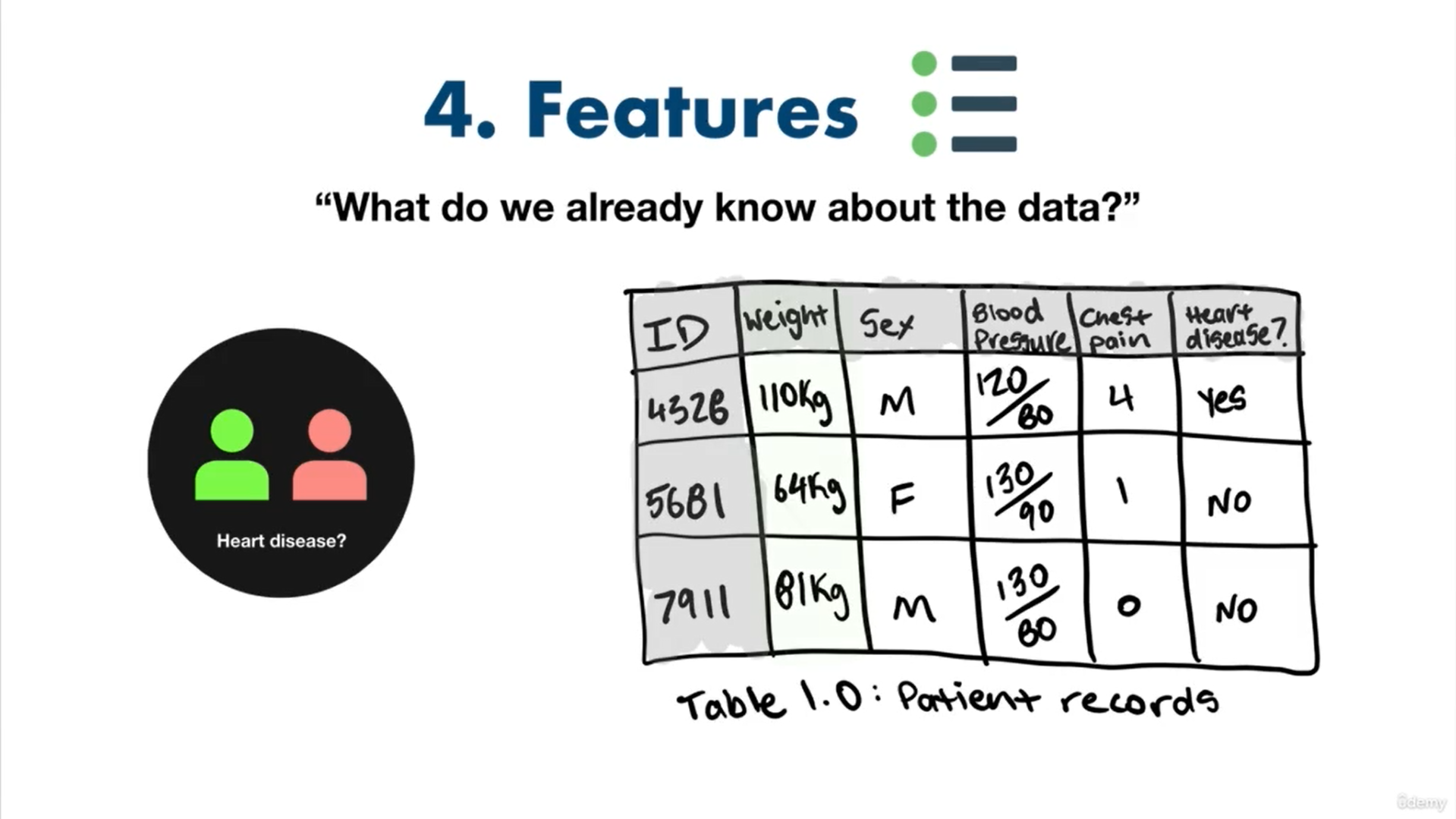
## Data



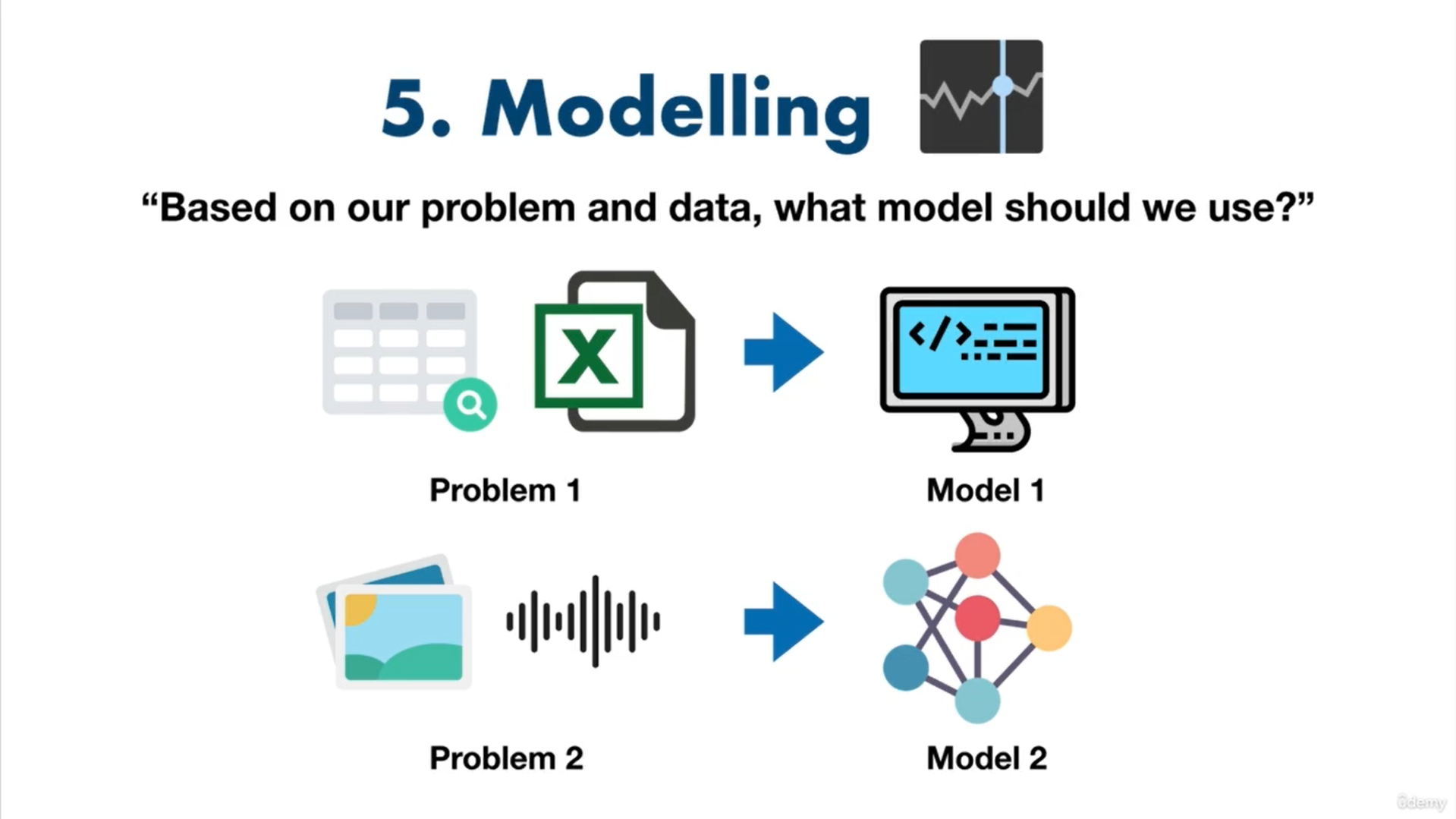
## Evaluation



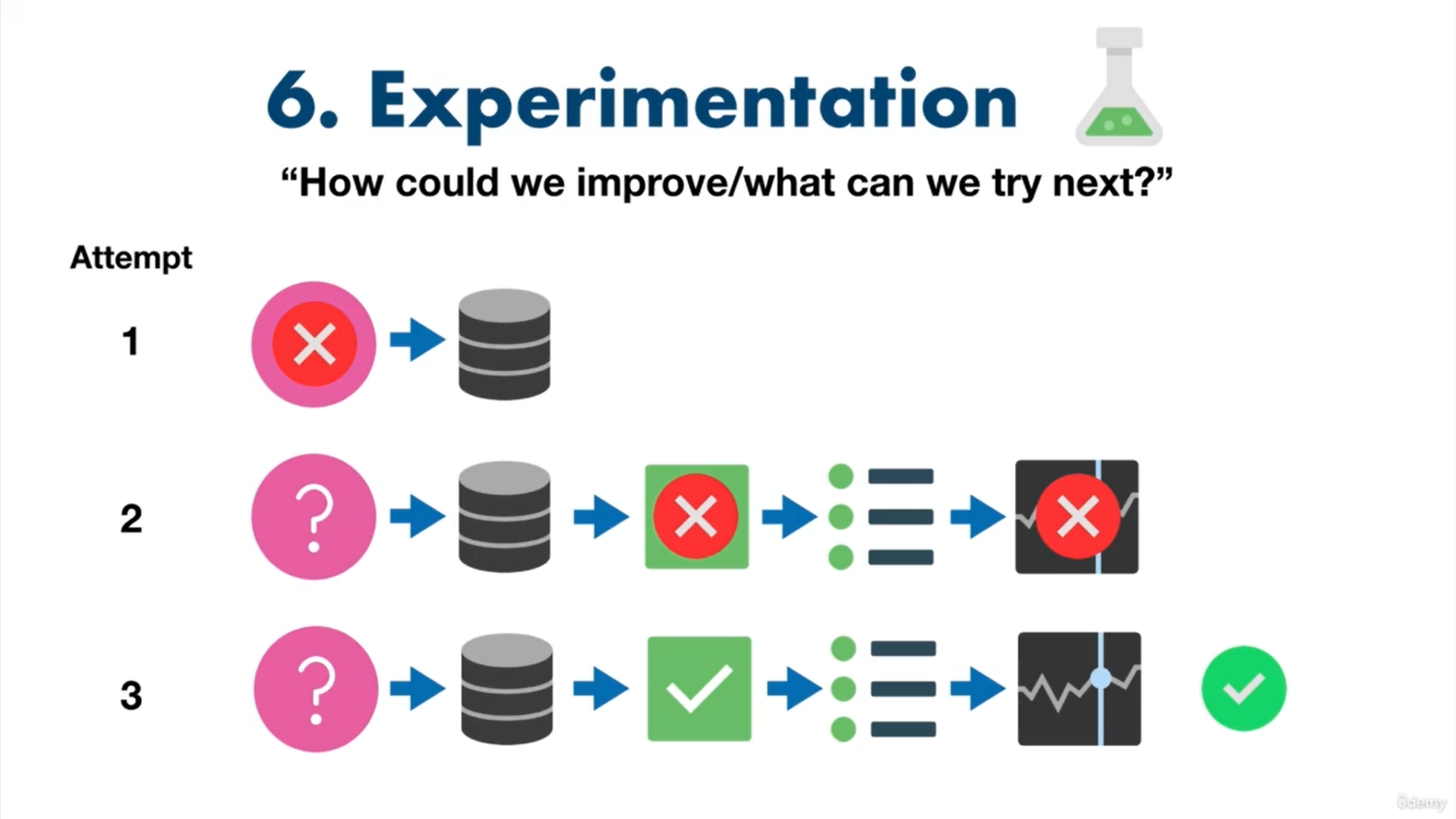
## Features



## Deciding Model



## Experimentation



# Main types of Machine Learning

1. Supervised
2. Un-supervised
3. Transfer learning
4. Reinforcement learning

## Supervised Learning

This is of 2 types

1. Classification
2. Regression

## Unsupervised Learning

This has data but no labels. The machine itself find the patterns. Example – recommendation systems

## Transfer Learning

Transferring a model to be used in another case without retraining it. Example – A car detection model is already trained on backgrounds such as roads, grass, buildings, etc. Now it can be used to detect dog images with minimal retraining, instead of training a completely new model.

## Reinforcement Learning

In this the computer is asked to do a task and is rewarded when it does it correctly and punished when not. Reward is a + value and punishment is a -ve value. Example, asking machine to play chess.

# Types of Data

The data is of 2 types

1. Structured – CSV, EXCEL, etc
2. Unstructured – Audio, video, emails, etc.

# Evaluation Metrics

|  |  |  |
| --- | --- | --- |
| Classficiation | Regression | Recommendation |
| Accuracy | MAE | Precision at *K* |
| Precision | MSE |  |
| Recall | RMSE |  |

# Features

The inputs to a machine learning model are called features. Features derived from the data are called **derived features** and this method is called **feature engineering.** Features can be categorical or numerical.

## Feature coverage

The process of ensuring that all features have similar values and are good enough to be taken into consideration is called feature coverage.

# Modelling

Modelling a machine learning model can be broken down into 3 parts

1. Choosing and training the model.
2. Tuning the model
3. Model comparison

## Splitting data

In this process, we break the given data into test, validate and training sets.

Training set is used to train the model, validate set is used to tune the model and test set is used to test the model on unseen data.

### Generalization

The ability of machine learning model to perform well on data that it has not seen before.

## Choosing the model

Different kinds of model work best in different kinds of data.

Use **decision trees, XGBoost, CatBoost, etc** for structured data.

Use **deep learning & transfer learning** for unstructured data.

Choose the lightest models first and then go for higher end models. Don’t go for slight accuracy increase if the model takes hours extra to do that.

## Tuning the model

A model is tuned on a validation split or on a test split. It means adjusting the parameters of the model, for example, tuning the number of hidden layers in a neural network, etc.

**A model’s first result are not the final results.**

## Model comparison

A model should have similar test and train accuracy. If its too high in test (**overfitting**) and too low in test (**underfitting**), then the model should be discarded. There should be a trade-off just enough to generalize well.

**Data leakage** into the training data from test data can lead to overfitting and **data mismatch** between test and training data (different features) can lead to underfitting.

### How to overcome underfitting

1. Try a more advanced model.
2. Increase model hyperparameters, like increasing the hidden layers in a neural network.
3. Reduce the amount of features.
4. Train for a longer period of time.

### How to overcome overfitting

1. Use lesser advanced model
2. Collect more data for split.

***Compare models on Accuracy, Training time and Prediction time.***

All experiments should be conducted on different portions of your data.

* **Training data set** — Use this set for model training, 70–80% of your data is the standard.
* **Validation/development data set** — Use this set for model hyperparameter tuning and experimentation evaluation, 10–15% of your data is the standard.
* **Test data set** — Use this set for model testing and comparison, 10–15% of your data is the standard.

These amounts can fluctuate slightly, depending on your problem and the data you have.

Poor performance on training data means the model hasn’t learned properly and is **underfitting**. Try a different model, improve the existing one through hyperparameter or collect more data.

Great performance on the training data but poor performance on test data means your model doesn’t generalize well. Your model may be **overfitting** the training data. Try using a simpler model or making sure you’re the test data is of the same style your model is training on.

Another form of **overfitting** can come in the form of better performance on test data than training data. This may mean your testing data is leaking into your training data (incorrect data splits) or you've spent too much time optimizing your model for the test set data. Ensure your training and test datasets are always kept separate and avoid optimizing a model’s performance on the test set (use the training and validation sets for model improvement).

Poor performance once deployed (in the real world) means there’s a difference in what you trained and tested your model on and what is happening. Ensure the data you're using during experimentation matches up with the data you're using in production.