Model Evaluation

Choosing the **best** model

Faculty of Mathematics and Computer Science, University of Bucharest
and
Sparktech Software

Academic Year 2018/2019, 1st Semester

Which model is better?

- Task: Predict a store's daily ice cream sales based on outside temperature.
- We only have recorded data from 10 days:

	Day	1	2	3	4	5	6	7	8	9	10
x	Temperature (in ℃)	12	33.9	30.1	14.1	4.6	7.3	3.1	29.2	21.1	27.2
y	Ice Cream Sales (in $)	182.2	445.1	394	167.7	53.7	66.8	114.8	344.2	179.5	267.7

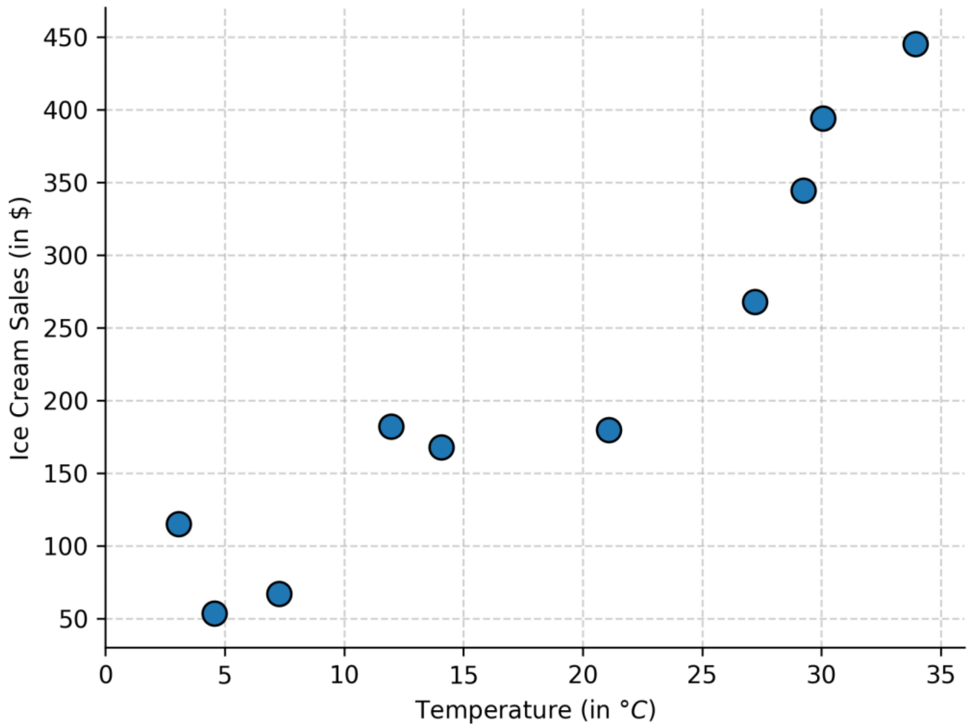
Which model is better?

- Task: Predict a store's daily ice cream sales based on outside temperature.
- We only have recorded data from 10 days:

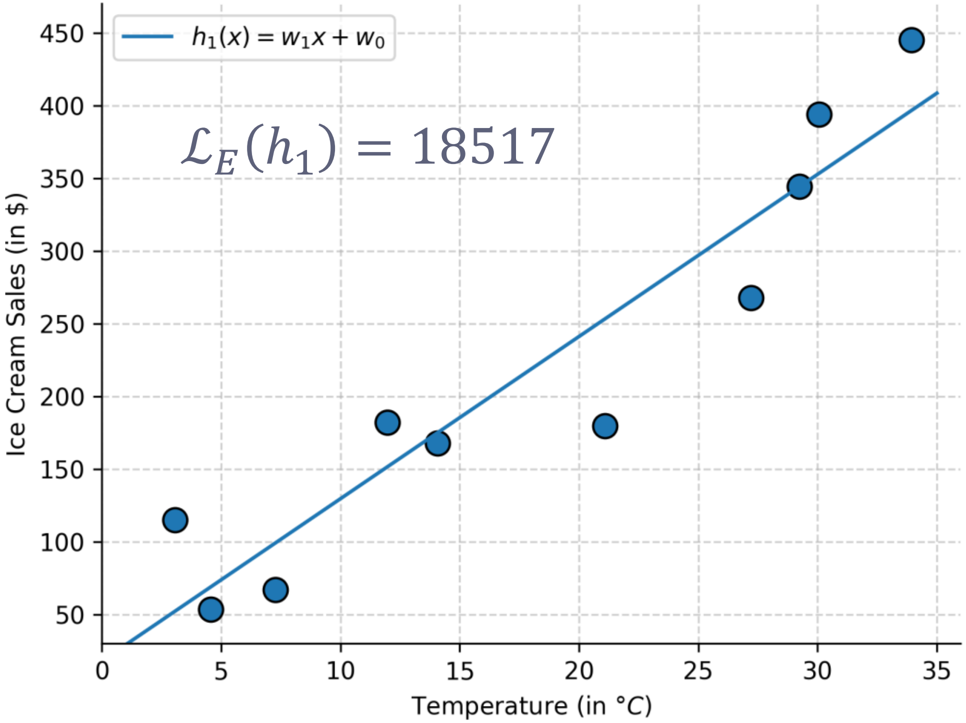
	Day	1	2	3	4	5	6	7	8	9	10
x	Temperature (in ℃)	12	33.9	30.1	14.1	4.6	7.3	3.1	29.2	21.1	27.2
y	Ice Cream Sales (in $)	182.2	445.1	394	167.7	53.7	66.8	114.8	344.2	179.5	267.7

- We must compare two models:
  - $h_1(x) = w_1 x + w_0$
  - $h_2(x) = w_5' x^5 + w_4' x^4 + \cdots + w_1' x + w_0$

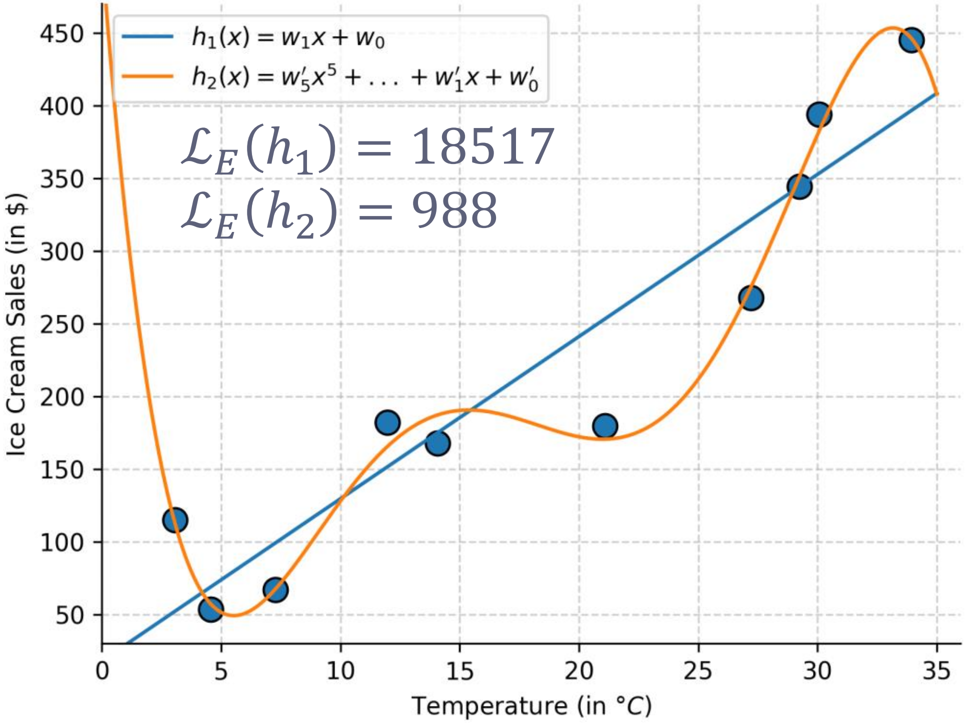
Which model is better?



Which model is better?

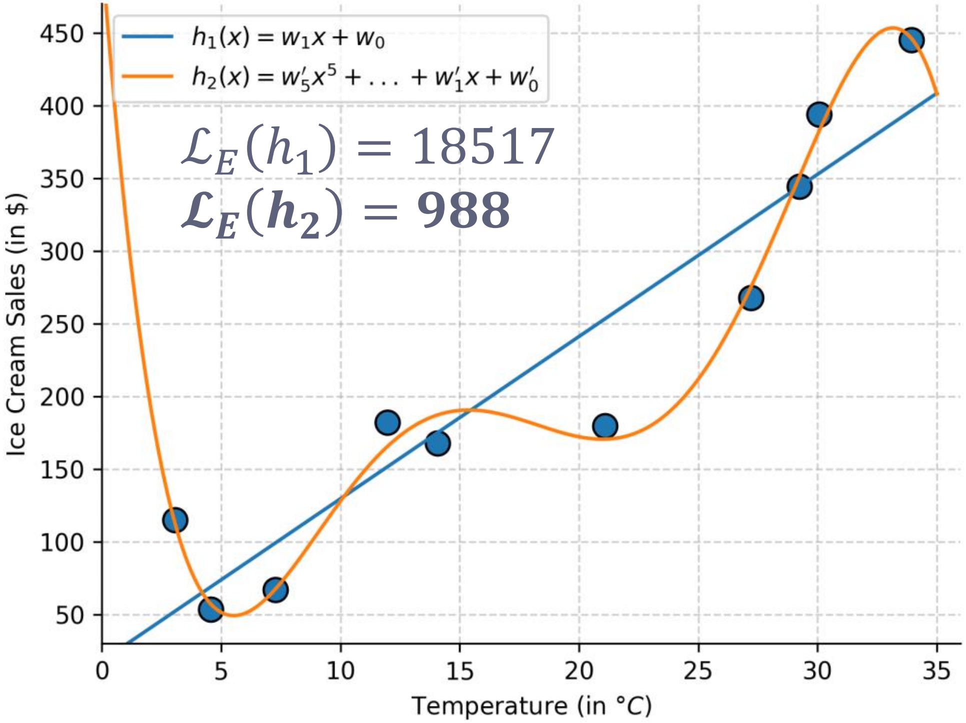


Which model is better?



Which model is better?

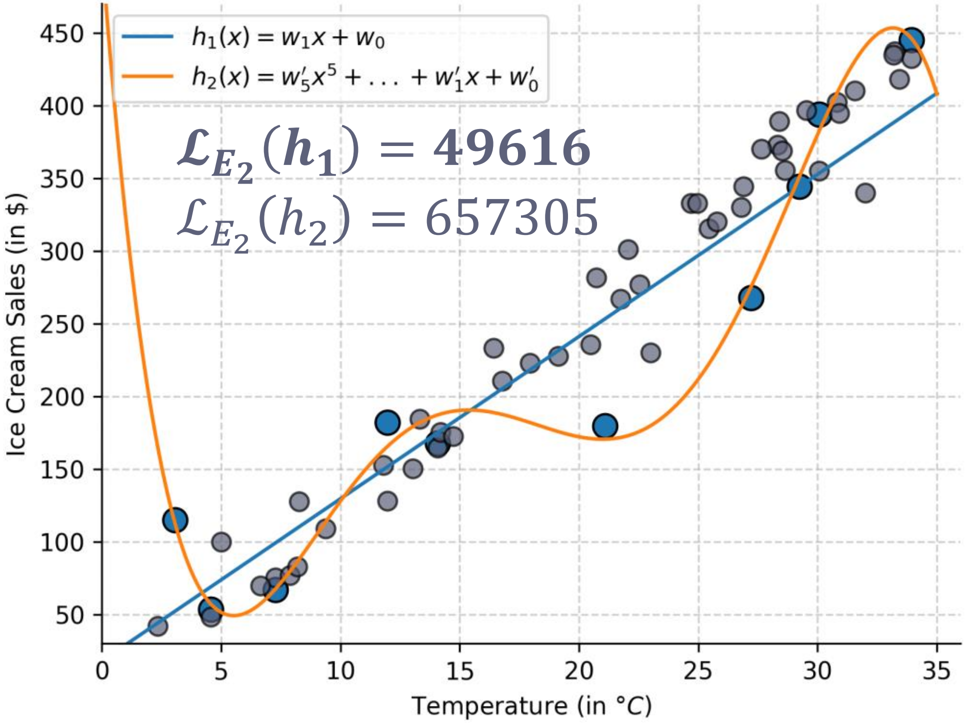
Which model is better?



Which model is better?

- h_2, according to training loss.

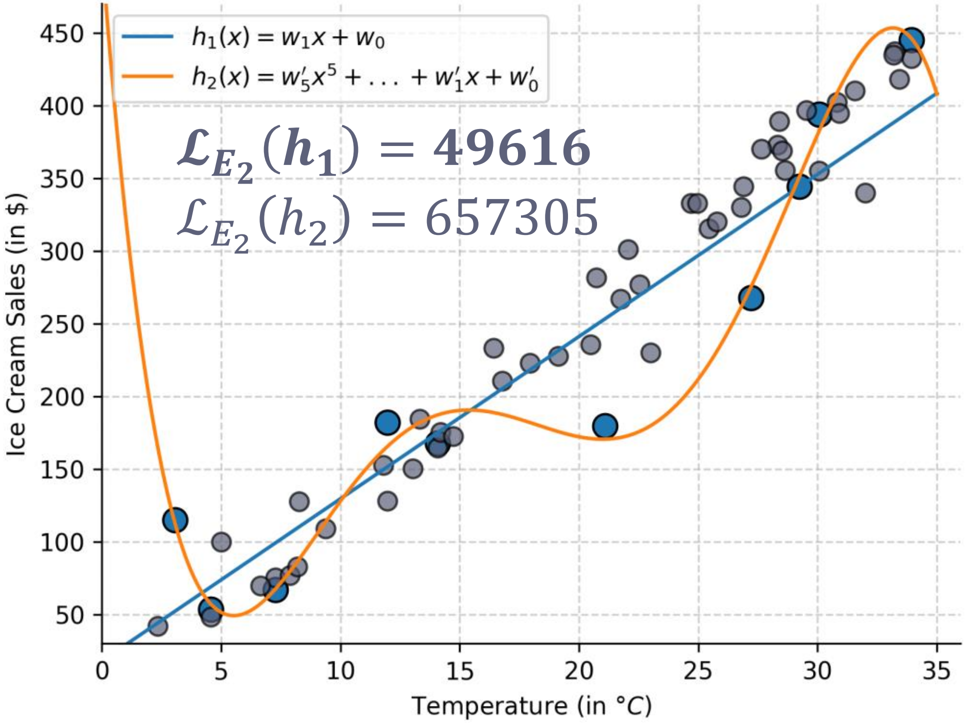
Which model is better?



Which model is better?

- h_2, according to training loss.
- But when we evaluate on other points, h_1 performs much better.

Which model is better?



Which model is better?

- h_2, according to training loss.
- But when we evaluate on other points, h_1 performs much better.

In practice, we don't always get to do this.

Data Splitting Strategies

Training – Test split

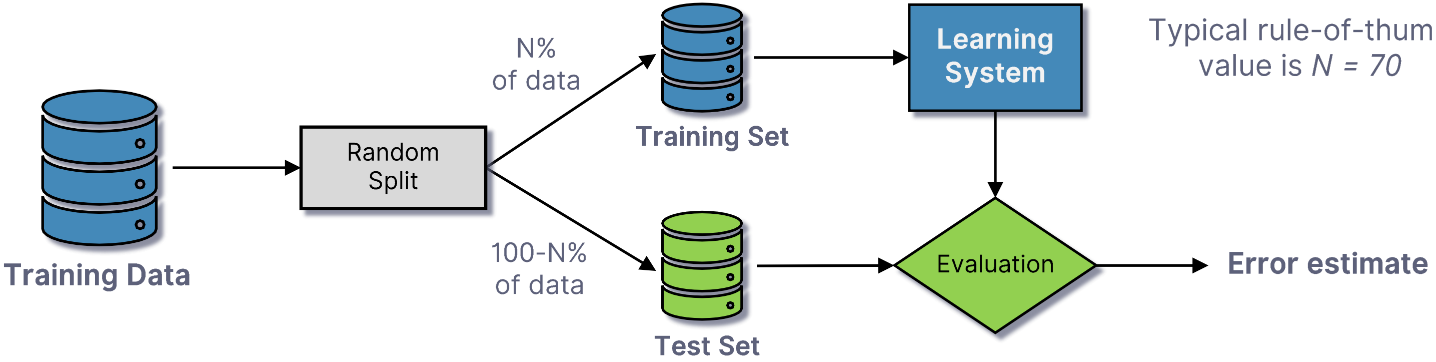
- In practice, we don’t get to see the *true distribution* and we don’t always get access to *more samples* to evaluate on.

Training – Test split

- In practice, we don’t get to see the *true distribution* and we don’t always get access to *more samples* to evaluate on.
- As we have seen, training error is a very optimistic estimate of true error.
    - It will almost always be lower than true error.
    - It will favor more complex models, which tend to overfit.

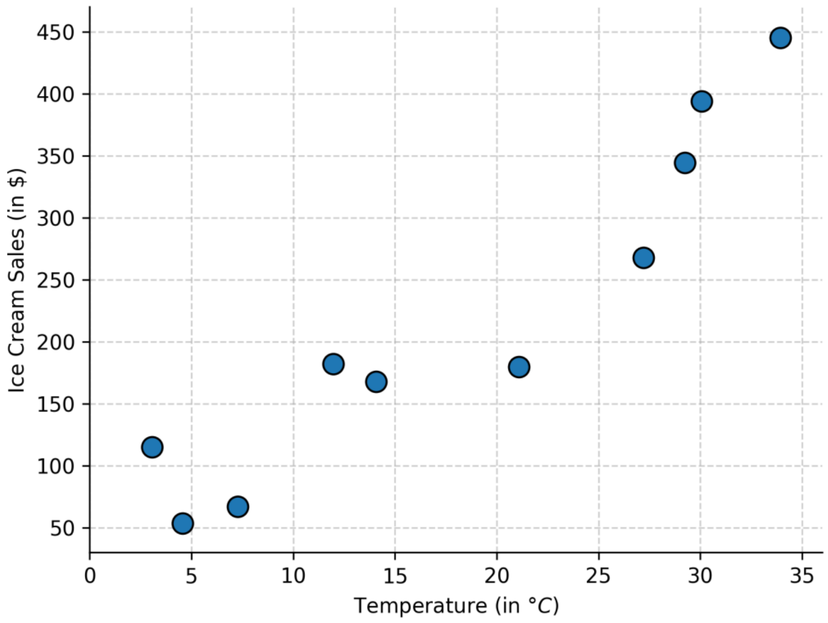
Training – Test split

- In practice, we don't get to see the *true distribution* and we don't always get access to *more samples* to evaluate on.
- As we have seen, training error is a very optimistic estimate of true error.
  - It will almost always be lower than true error.
  - It will favor more complex models, which tend to overfit.
- One possibility to improve the estimate is the **Hold-out method**:



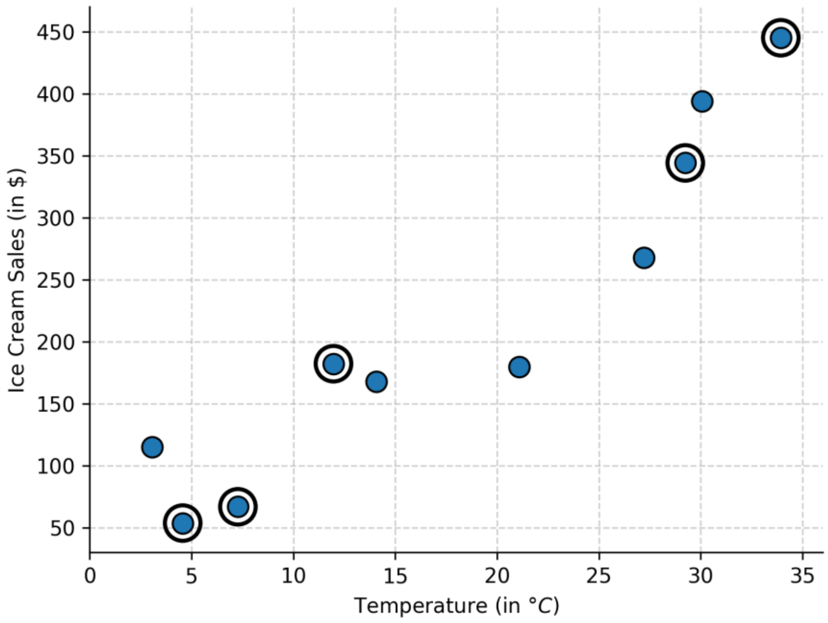
Typical rule-of-thumb value is *N = 70*

Training – Test split

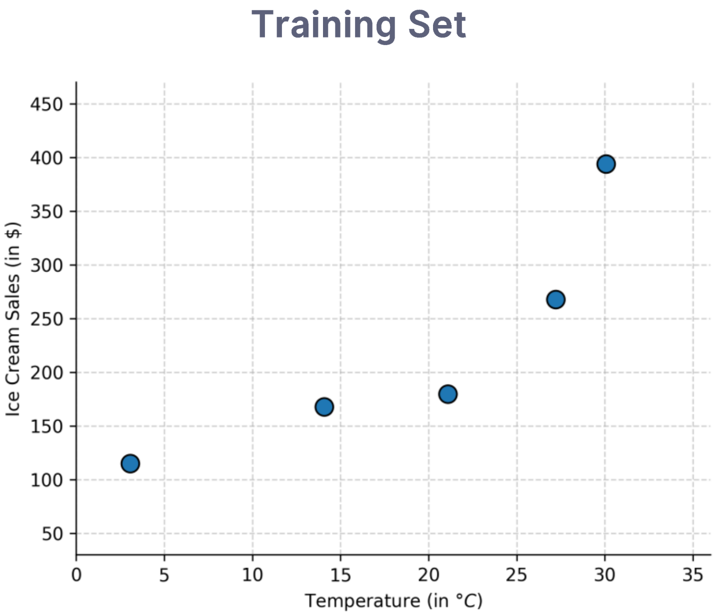


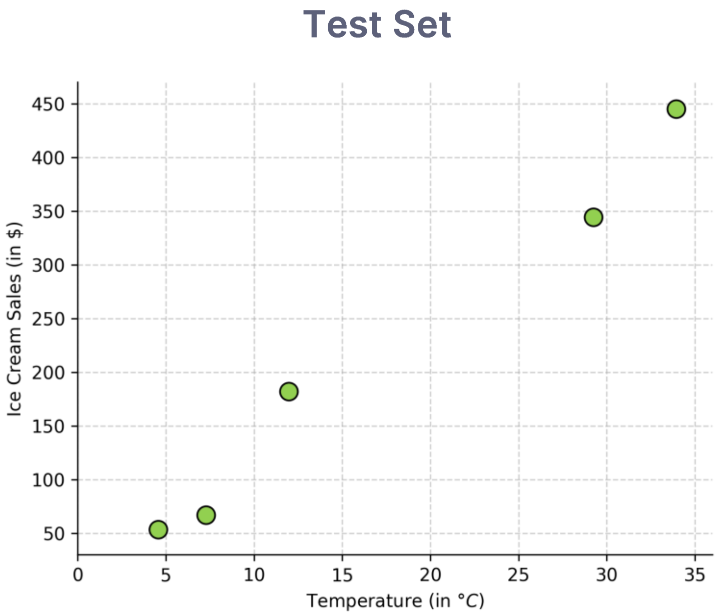
Training – Test split

Randomly select some points from the training data.



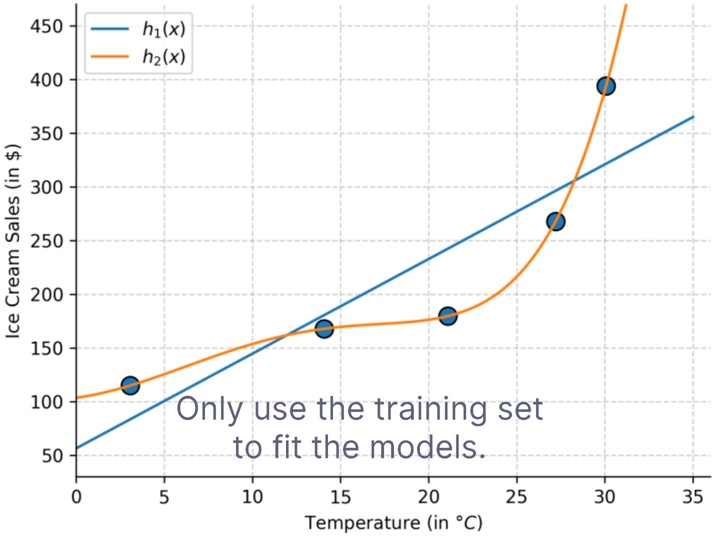
Training – Test split



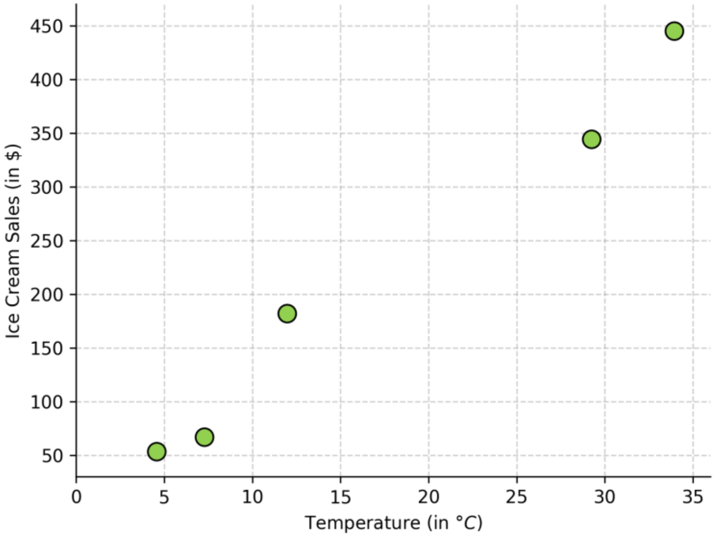


Training – Test split

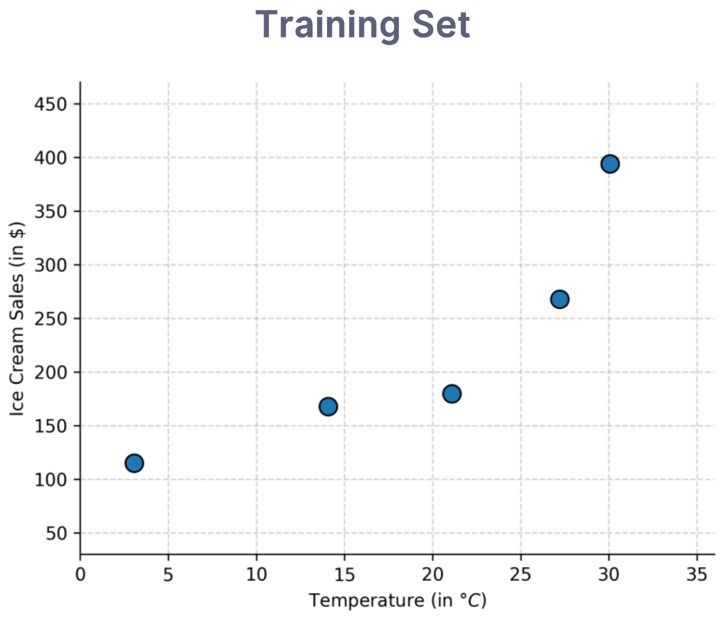
Training Set

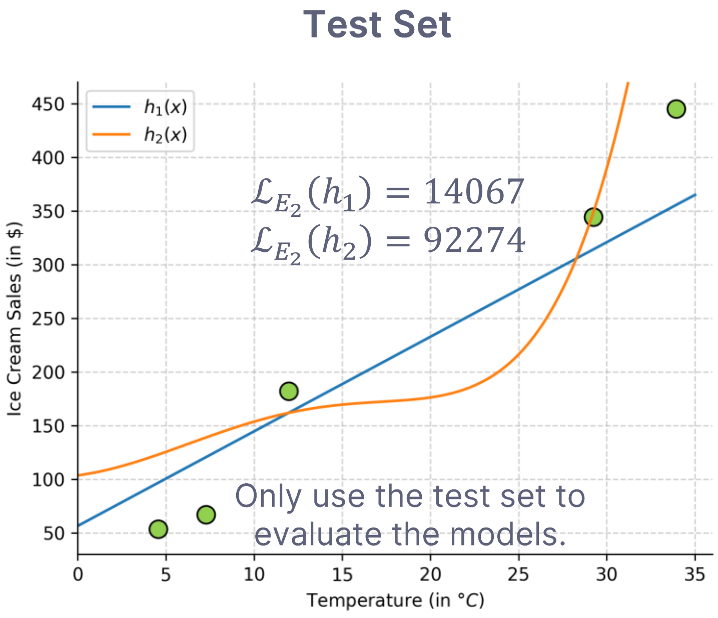


Test Set



Training – Test split

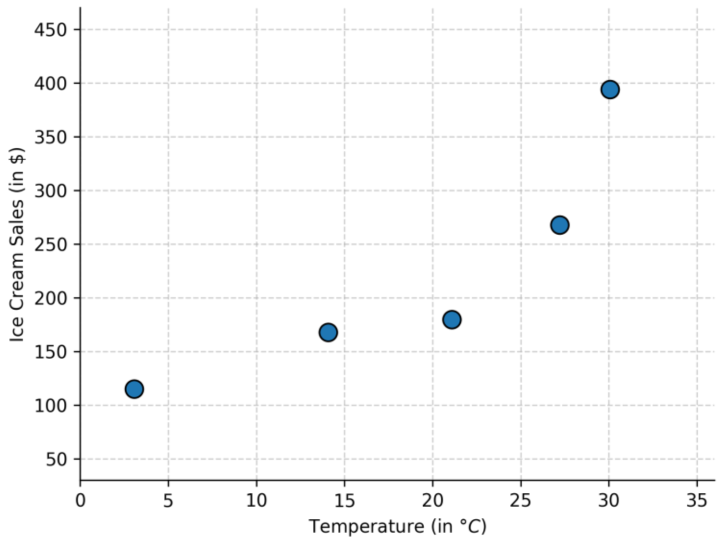




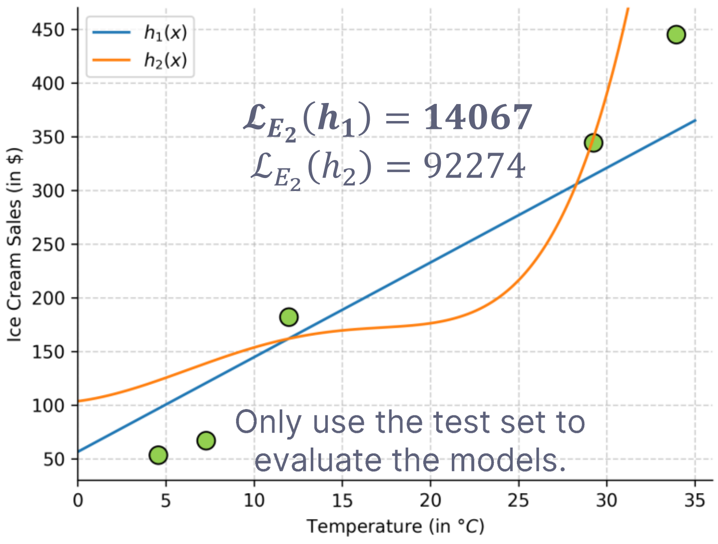
Training – Test split

We can conclude that h_1 is a better model for this task, without seeing any additional points.

Training Set



Test Set



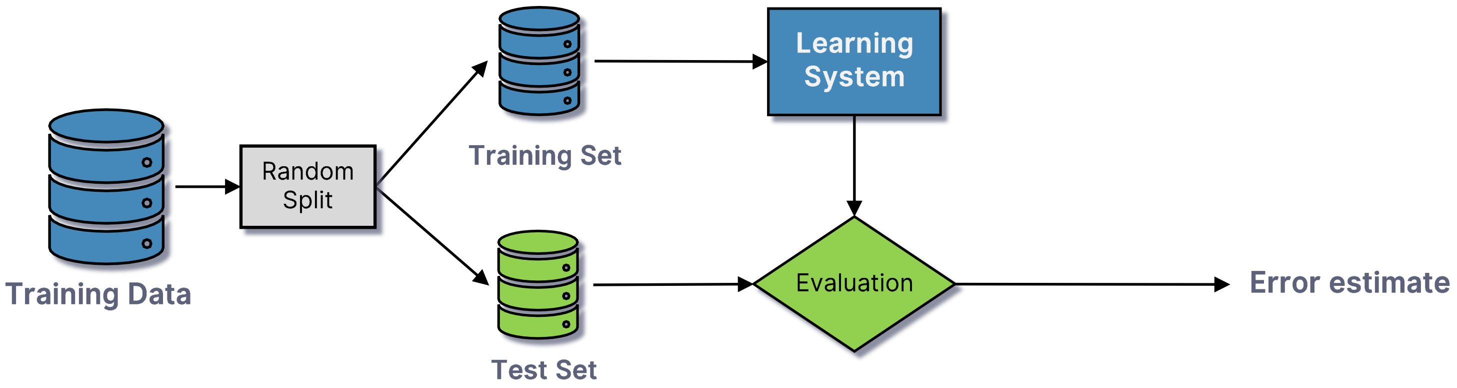
Problems with using a single train-test split

- Some algorithms have *hyperparameters*, which control the training process.
  - e.g. Parameter λ in Ridge Regression
- Repeatedly using the same train – test sets when trying different hyperparameters can "*wear out*" the test set.
  - Overfitting in hyperparameter space.

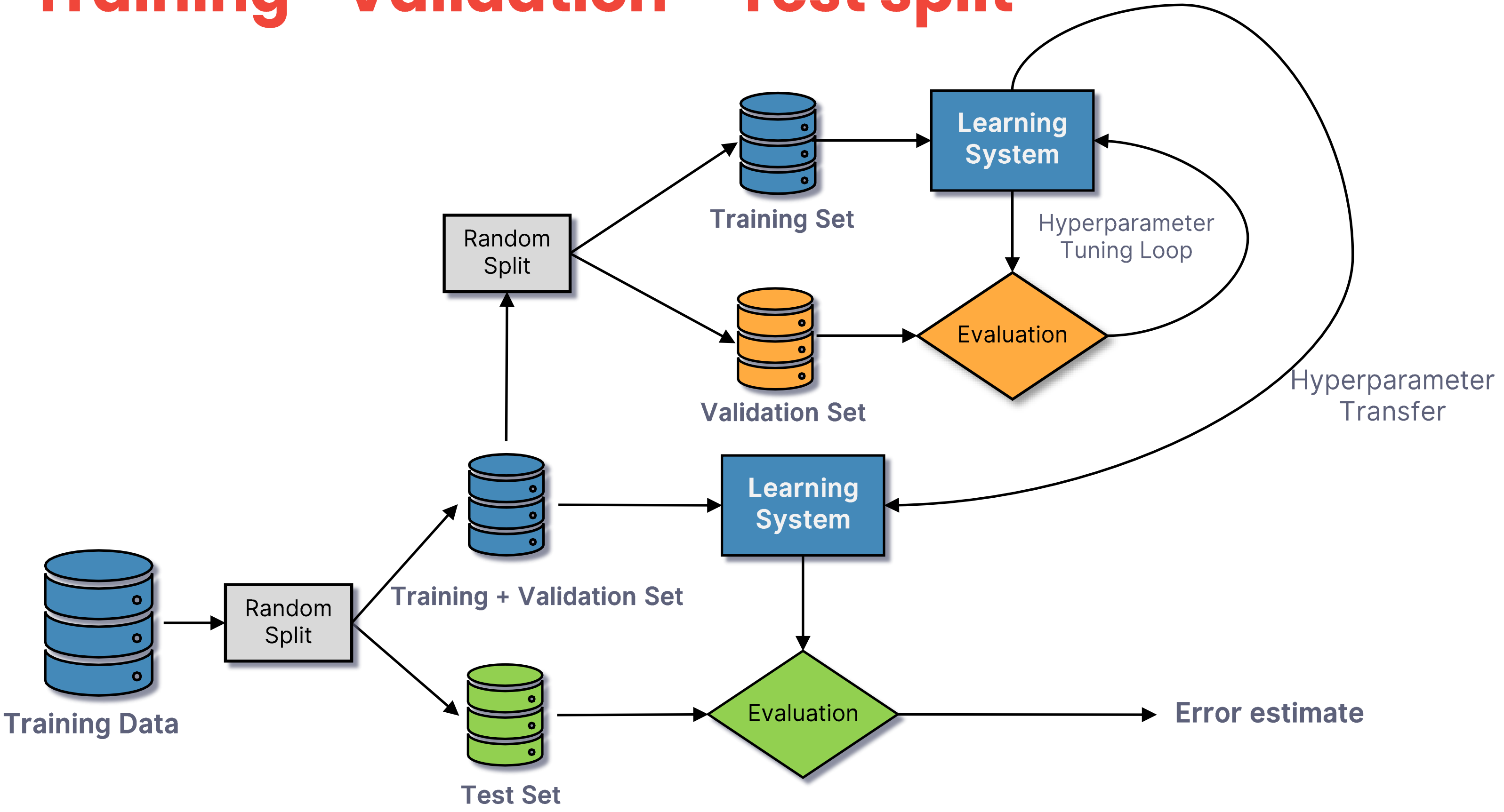
Problems with using a single train-test split

- Some algorithms have *hyperparameters*, which control the training process.
  - e.g. Parameter λ in Ridge Regression
- Repeatedly using the same train – test sets when trying different hyperparameters can "*wear out*" the test set.
  - Overfitting in hyperparameter space.
- We can improve the error estimate by using a separate **validation** set for *hyperparameter tuning*

Training– Validation – Test split



Training– Validation – Test split



K-fold Cross-Validation

- Sometimes we don’t have enough data to “*afford*” a validation set.
  - The remaining training set would simply be too small to fit a model on.

K-fold Cross-Validation

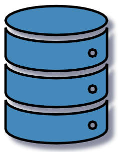
- Sometimes we don’t have enough data to “*afford*” a validation set.
  - The remaining training set would simply be too small to fit a model on.

- **K-fold Cross-Validation**:
  - Split the data into k equal parts (a.k.a. *folds*)
  - Repeat the train – test process k times, each time using one fold for testing and the rest for training.
  - Average out the errors.

K-fold Cross-Validation

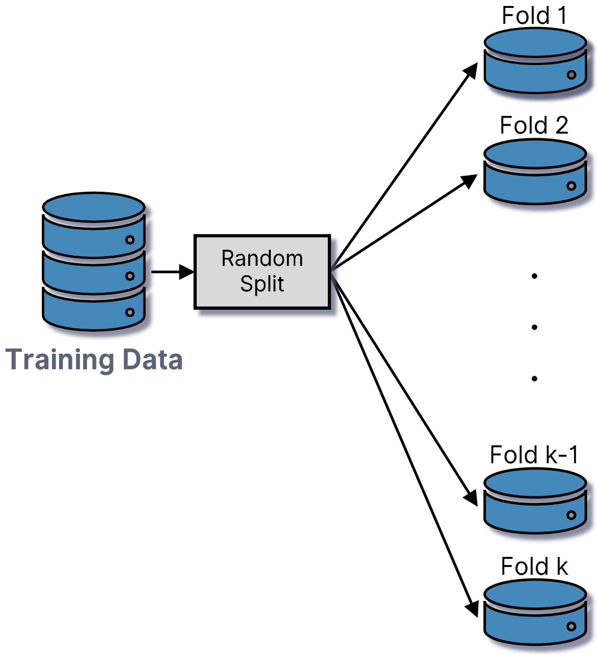
- Sometimes we don't have enough data to "*afford*" a validation set.
  - The remaining training set would simply be too small to fit a model on.
- **K-fold Cross-Validation**:
  - Split the data into k equal parts (a.k.a. *folds*)
  - Repeat the train – test process k times, each time using one fold for testing and the rest for training.
  - Average out the errors.
- Even if we have enough data for a validation set, we can obtain a better estimate of true error this way.
  - If we have enough computing power.

K-fold Cross-Validation

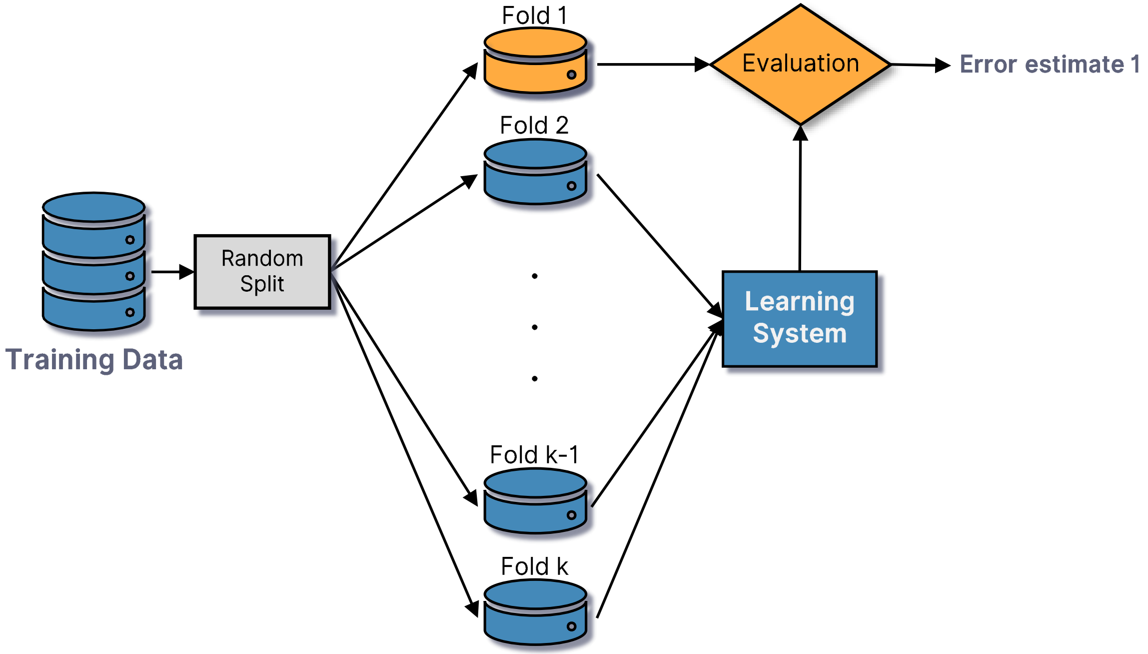


Training Data

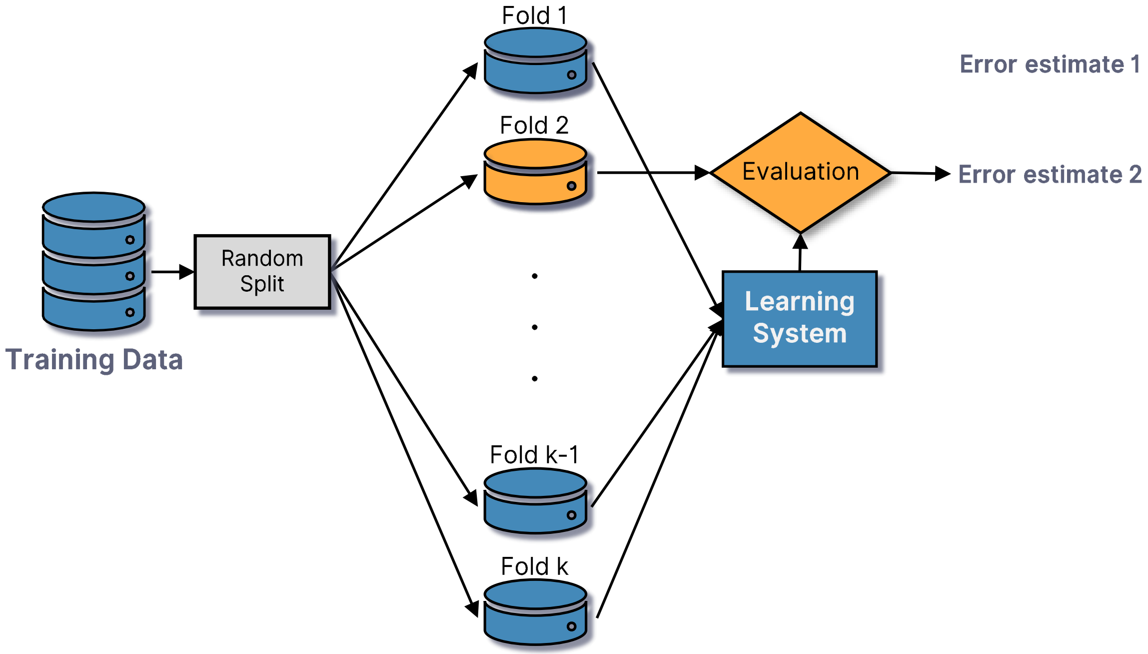
K-fold Cross-Validation



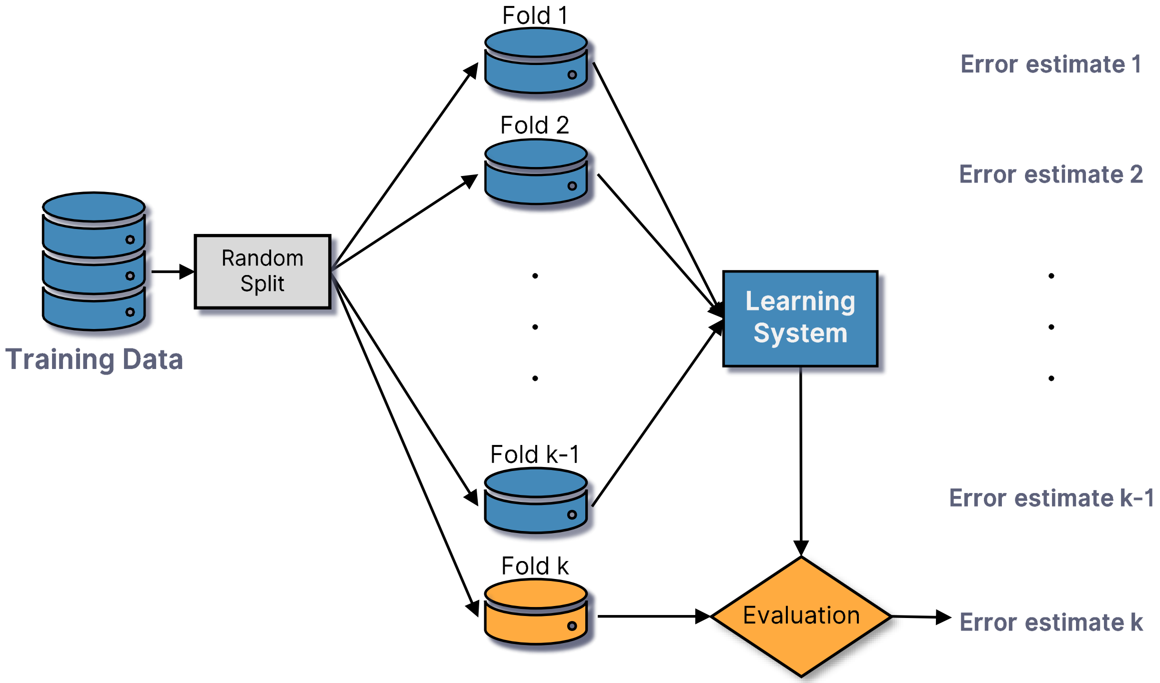
K-fold Cross-Validation



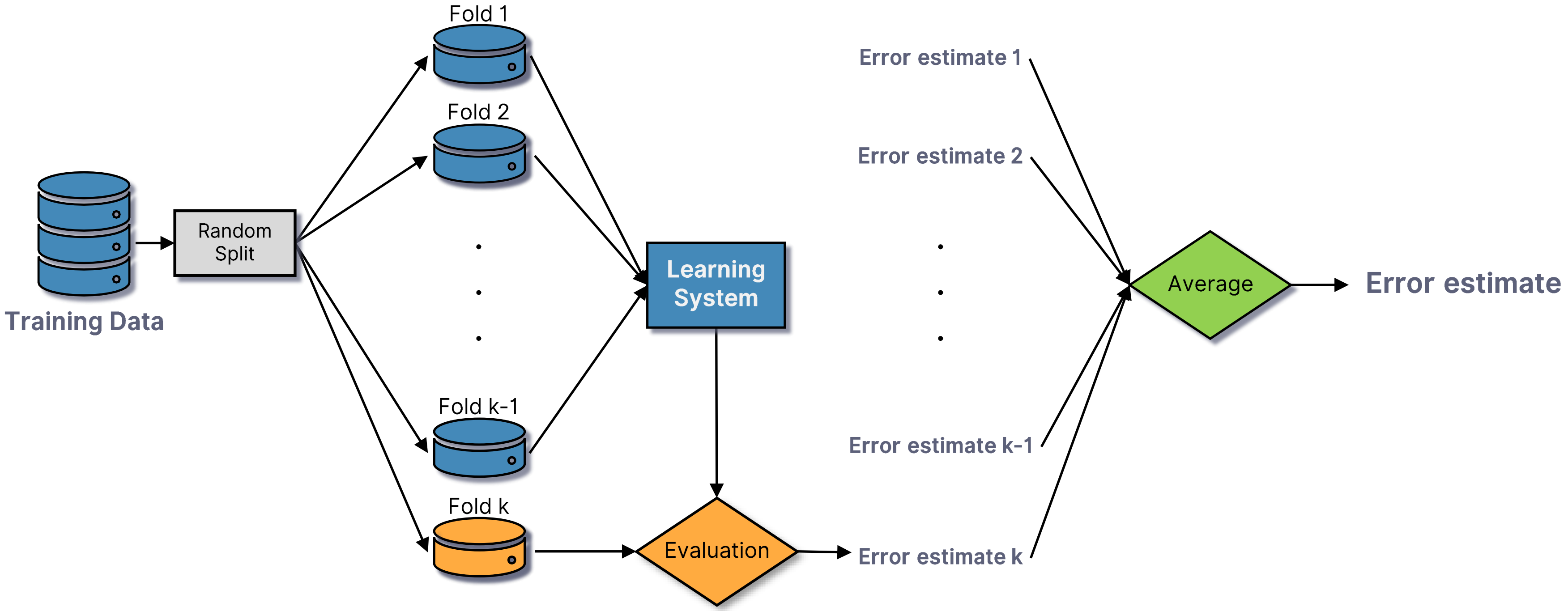
K-fold Cross-Validation



K-fold Cross-Validation

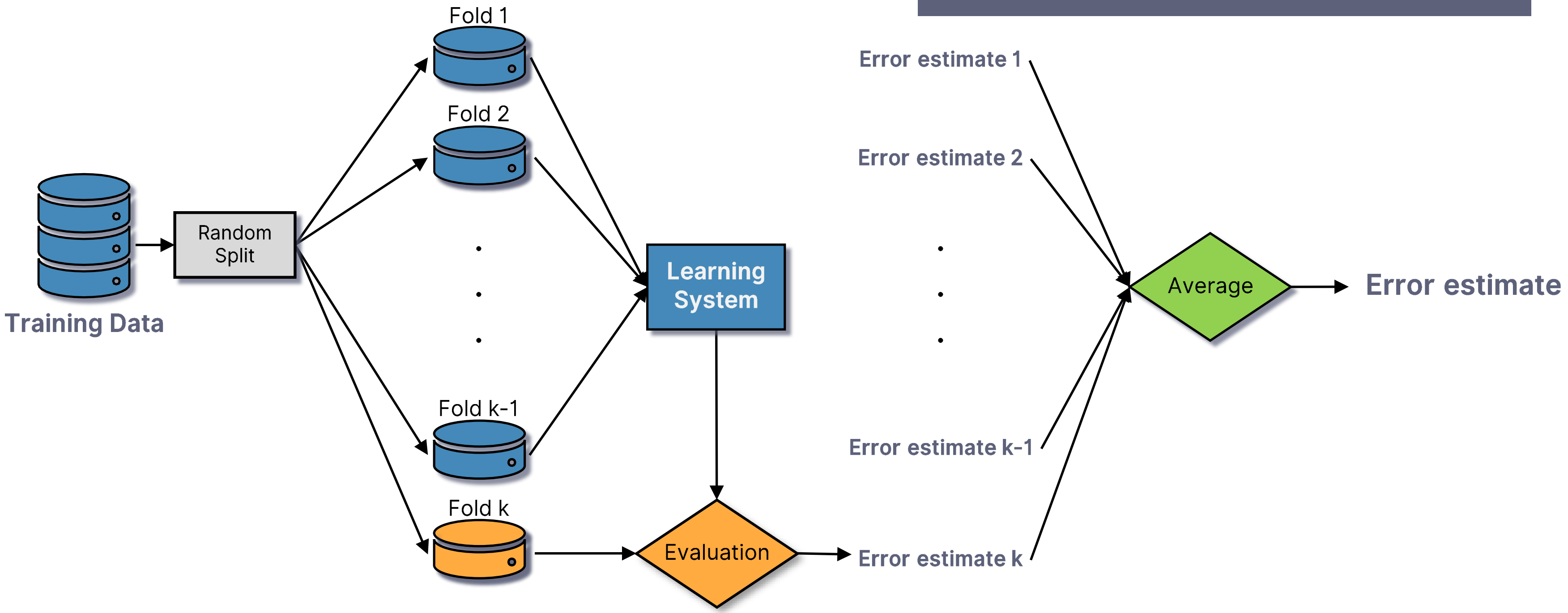


K-fold Cross-Validation



K-fold Cross-Validation

Extreme case:
Leave-One-Out Cross Validation
(*k* is equal to the number of data points).



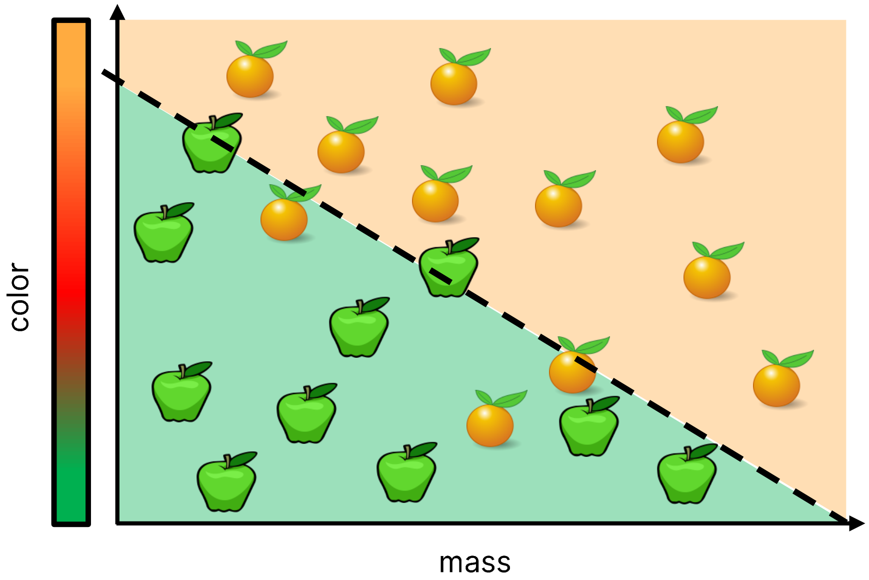
Summary

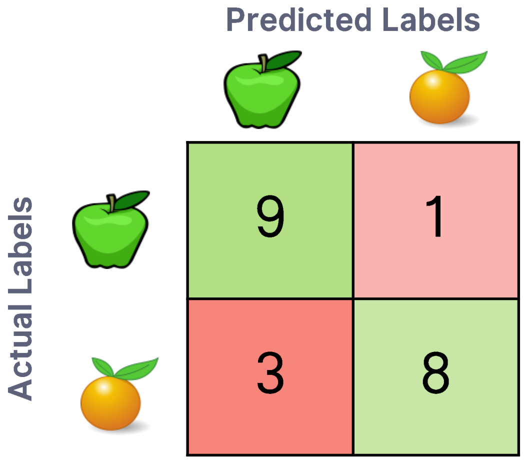
- Using training error for performance evaluation is usually a bad idea.
  - It is a very optimistic estimate of true error and it favors models which overfit.
- **Holding out data for testing** gives a much better estimate.
- However, if we use a single test set for *hyperparameter selection,* we might end up reducing the quality of the estimation.
  - Using a separate **validation set** for this is a much better idea.
- If we don’t have enough data or if we have enough computing power, **K-fold Cross-Validation** gives an even better estimate.
  - If we use each sample as a fold, we get **Leave-One-Out Cross Validation**

Performance Metrics for Classification

Confusion Matrix

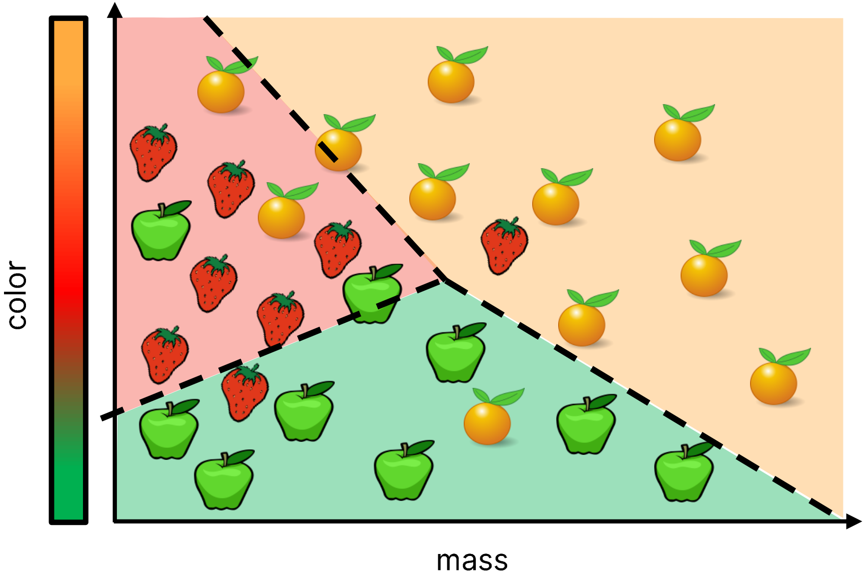
- A **confusion matrix** is a table which describes the performance of a classification model by displaying how often classes get confused.





Confusion Matrix

- A **confusion matrix** is a table which describes the performance of a classification model by displaying how often classes get confused.



Predicted Labels

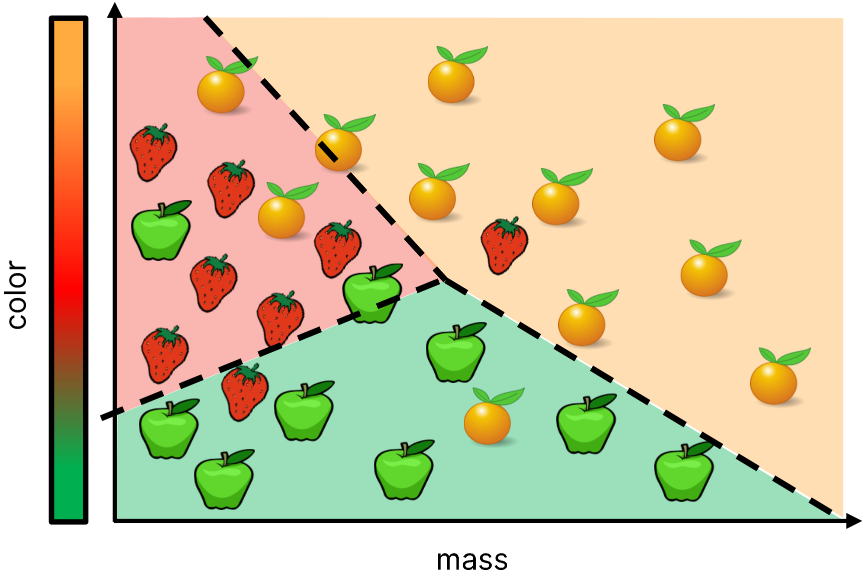
Actual Labels

	7	0	2
	1	8	2
	1	1	6

It can be used for any number of classes.

Accuracy

- **Accuracy** is the fraction (or percentage) of items for which the model predicted the correct class.



$$\text{Accuracy} = \frac{21}{28} = 0.75 = 75\%$$

Problems with Accuracy

- Imagine that we must train a classifier to detected if a patient has an extremely rare disease.
    - From 1000 patients, only 20 have the disease.
    - This is not just our dataset, this is the *true distribution*.

Problems with Accuracy

- Imagine that we must train a classifier to detected if a patient has an extremely rare disease.
  - From 1000 patients, only 20 have the disease.
  - This is not just our dataset, this is the *true distribution*.
- A classifier which predicts that nobody has the disease is 98% accurate.
  - Pretty good, right?
  - Not really... What is the problem?

Problems with Accuracy

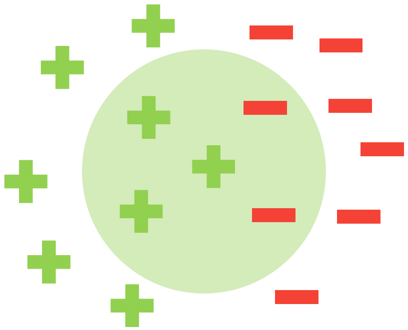
- Accuracy does not capture several aspects:
  - The dataset is very imbalanced.
    - One class is much more common than the other.
  - The 2 classes are conceptually different.
    - One is "*positive*", it represents what we are looking for.
    - One is "*negative*", it represent the default state.
  - There are 2 types of errors, which may differ in terms of severity.
    - *Type 1 error ("False Positive")*
    - *Type 2 error ("False Negative")*

Depending on the problem, one might more serious than the other.

Precision and Recall

- When there is a *positive* class or classes are imbalanced, precision and recall are more informative than accuracy.
    - **Precision** is the fraction of *selected* items which are *relevant*.
    - **Recall** is the fraction of *relevant* items which are *selected*.

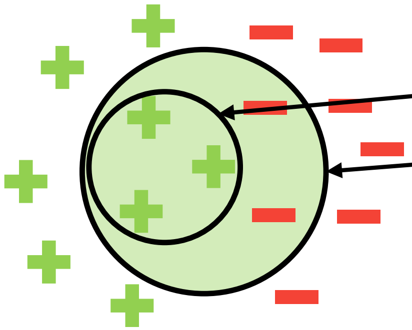
Relevant = *Positive*
Selected = *Predicted Positive*



Precision and Recall

- When there is a *positive* class or classes are imbalanced, precision and recall are more informative than accuracy.
  - **Precision** is the fraction of *selected* items which are *relevant.*
  - **Recall** is the fraction of *relevant* items which are *selected.*

Relevant = *Positive*
Selected = *Predicted Positive*

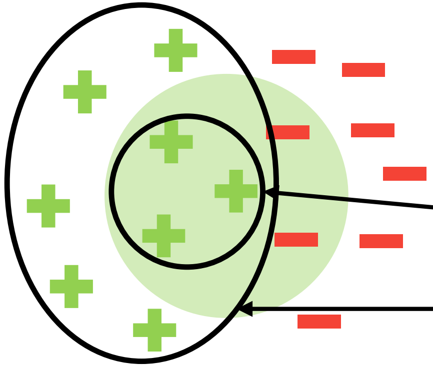


$$\text{Precision} = \frac{3}{5} = 0.6 = 60\%$$

Precision and Recall

- When there is a *positive* class or classes are imbalanced, precision and recall are more informative than accuracy.
  - **Precision** is the fraction of *selected* items which are *relevant*.
  - **Recall** is the fraction of *relevant* items which are *selected*.

Relevant = *Positive*
Selected = *Predicted Positive*



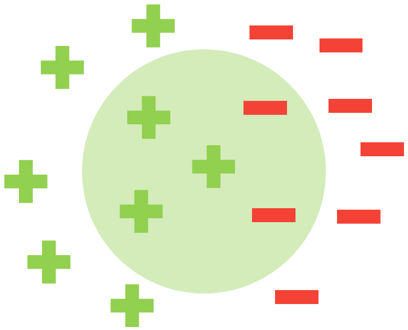
$$\text{Precision} = \frac{3}{5} = 0.6 = 60\%$$

$$\text{Recall} = \frac{\mathbf{3}}{\mathbf{8}} = 0.375 = 37.5\%$$

Precision and Recall

- When there is a *positive* class or classes are imbalanced, precision and recall are more informative than accuracy.
  - **Precision** is the fraction of *selected* items which are *relevant*.
  - **Recall** is the fraction of *relevant* items which are *selected*.

Relevant = *Positive*
Selected = *Predicted Positive*

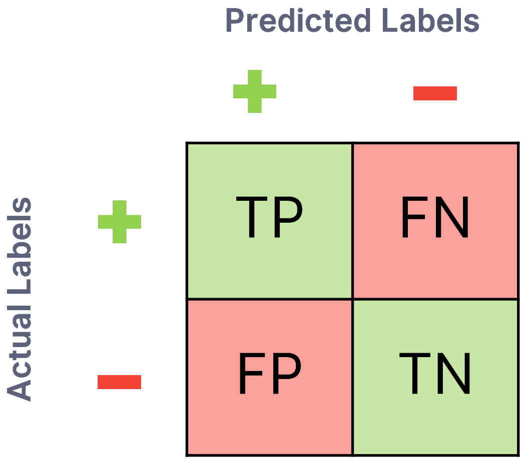


$$\text{Precision} = \frac{3}{5} = 0.6 = 60\%$$

$$\text{Recall} = \frac{3}{8} = 0.375 = 37.5\%$$

  - **F_1 score** is the harmonic mean of precision and recall.

Summary



Confusion Matrix

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F_1 = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2(\text{Precision} * \text{Recall})}{\text{Precision} + \text{Recall}}$$

Keywords

Training Set

Test Set

Error Estimate

Hyperparameter Tuning

Validation Set

Cross-Validation

K-fold

Leave-One-Out

Confusion Matrix

Accuracy

Precision

Recall

F_1 Score