Concepts of Model Assessment

Predictive Modeling & Statistical Learning

Gaston Sanchez

Model Performance

Model Performance

How do we define what a "good" model is?

- A model that fits the data well?
  (e.g. minimize resubstitution error)
- A model with optimal parameters?
  (e.g. most likely coefficients)
- A model that adequately predicts new (unseen) observations?
  (e.g. minimize generalization error)

In the Predictive Modeling arena ...

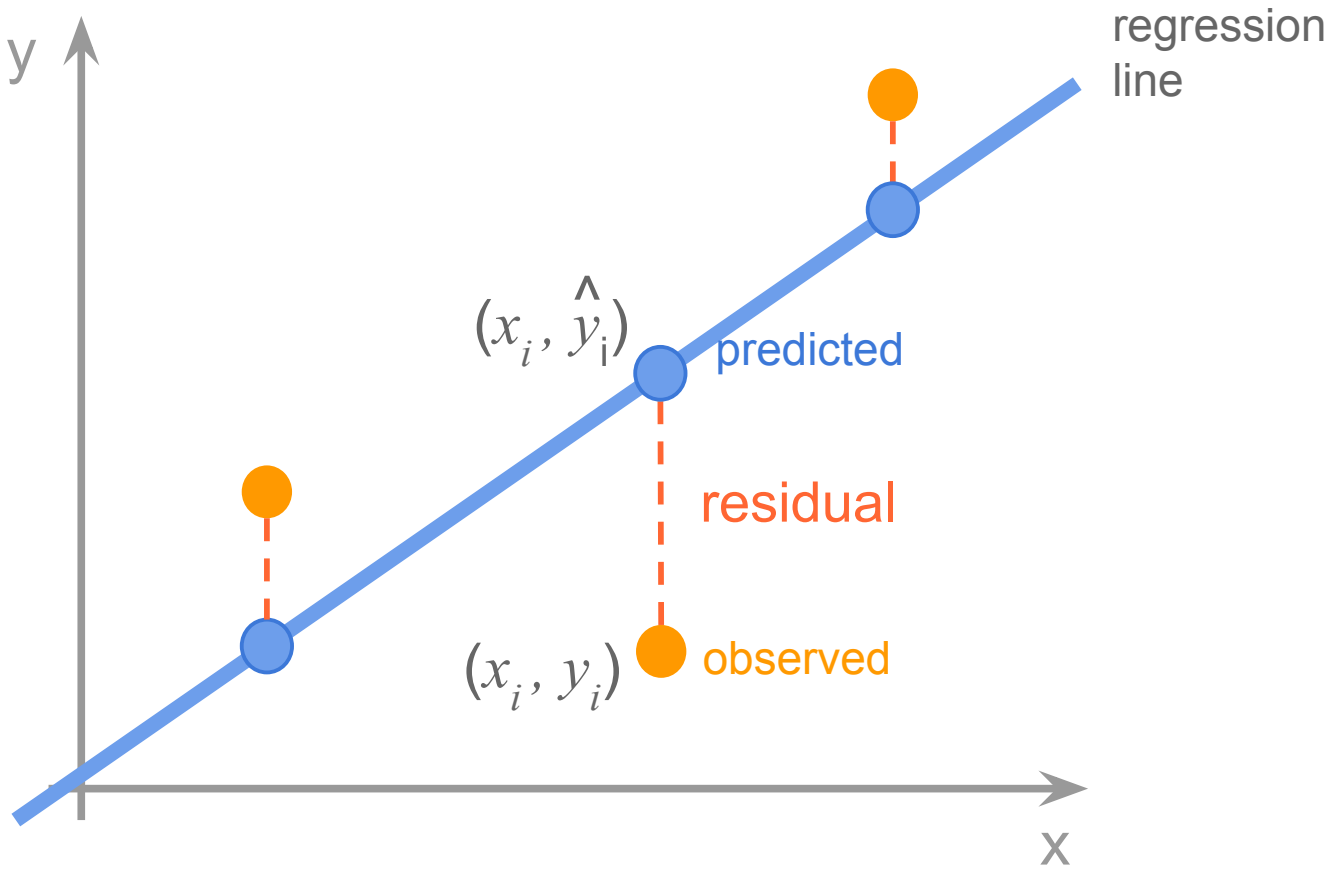
- A good model is one which gives accurate predictions.
- But what type of predictions?
- What do we mean by "predictions"?
- Predictions of observed data?
  observed: data used to fit the model
- Predictions of unobserved data?
  unobserved: data not used to fit the model
- How do we measure prediction accuracy?

About Measures of Accuracy

Assessing predictions

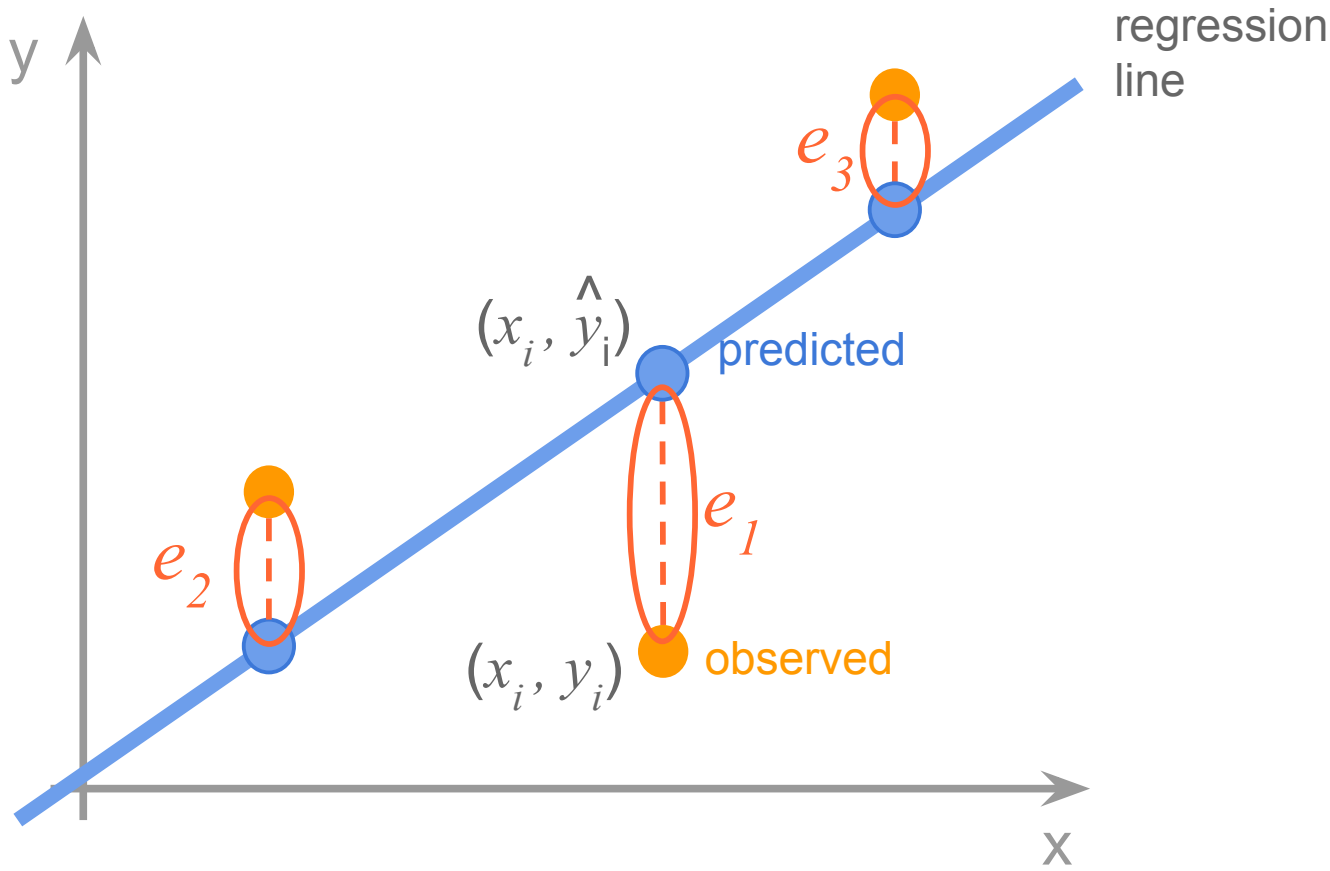
Observed y_i -vs- Predicted $\hat{y}_i$

Looking at the amount of residuals



Measuring the overall size of the differences

Looking at the amount of residuals



How big are these differences?

Starting Point: Residuals

- The main idea consists of comparing observed inputs y_i against predicted inputs $\hat{y}_i$.
- We need to measure the discrepancy between observed inputs and predicted inputs.
- So how do we quantify the difference between y_i and $\hat{y}_i$ for all $i = 1, \ldots, n$?
- The starting point involves considering residuals $e_i = y_i - \hat{y}_i$

Starting Point: Residuals

To measure the size of all residuals, $e_i = y_i - \hat{y}_i$, we commonly use a quadractic "loss" or squared norm function.

The summary considered up to this point is the RSS:

$$\text{RSS} = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Is RSS a good measure of model performance?

kind of ...

Residual Sum of Squares

One issue with the RSS is that it is a *total* value that depends on the number of residuals:

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Preferably we would like to have a summary that is a "representative" measure of the **typical** error.

Mean Squared Error (MSE)

A more "representative" measure of the typical error can be achieved by averaging RSS, getting what is called the Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{n}\text{RSS} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Think of MSE as the "typical" squared error.

Mean Squared Error (MSE)

- The MSE says how far typical points are above or below the regression line.
- Think of MSE as the typical size of prediction errors.
- In a general sense, MSE is a measure of scatter for residuals.
- MSE is a measure of how accurate our predictions are, on average.

Root Mean Squared Error (RMSE)

A side effect of using squared residuals is that of having to deal with squared units.

Some authors and practitioners prefer to take the square root of MSE in order to recover the original units. This produces the Root Mean Squared Error (RMSE):

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Mean Absolute Error (MAE)

MSE (and RMSE) are not the only possible options.

We can also consider the **absolute value** of residuals $|y_i - \hat{y}_i|$, and the corresponding Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Absolute value is more robust than quadratic value because squaring may exacerbate atypical values (e.g. very large residuals).

Course Assumption

For this course, unless stated otherwise, we are going to use **MSE** *as the standard measure of model performance.*

Keep in mind that some software and functions return RMSE.

The Concept of Learning

What type of predictions

Now that we have described the common available measures of accuracy (e.g. MSE, RMSE, MAE), let's go back to the discussion about types of "predictions"

- What do we mean by "predictions"?
- Predictions of observed data?
  observed: data used to fit the model
- Predictions of unobserved data?
  unobserved: data not used to fit the model

What type of predictions

To build a model $\hat{f}(X)$, we necessarily must use a set of n observations (x_i, y_i), $i = 1, \ldots, n$.

Here x_i represents a vector p predictor variables $X_1, X_2, \ldots, X_p$.

Having fitted $\hat{f}$, we begin assessing its quality by comparing observed response values y_i against the fitted values $\hat{y}_i$

In practice, though, we will typically be calculating $\hat{y}_{new} = \hat{f}(x_{new})$ for new observations x_{new} of which we don't know the value of the response: $y_{new} =?$

Main Challenge

If we are going to end up using $\hat{f}()$ to get $\hat{y}_{new} = \hat{f}(x_{new})$ without really knowing what the value of y_{new} is, then an essential question to be considered is:

How do we measure the accuracy of $\hat{f}()$?

This seems to be a question impossible to answer.

Main Challenge

Let's suppose that we have a set of n observations (x_i, y_i) that are used to fit a model $\hat{f}()$.

Suppose also that we had a set of m observations (x_0, y_0) that are NOT used to fit $\hat{f}()$, but we DO know the y_0 values.

We could pretend that we don't know y_0, calculate $\hat{y}_0 = \hat{f}(x_0)$, and then compare y_0 against $\hat{y}_0$. This could gives us an idea of how well/bad our model *generalizes* when faced with predicting "new" data.

Two types of predictions

On one hand:

We can measure how well our model $\hat{f}(x_i) = \hat{y}_i$ fits values y_i that we used to build the model:

$$\text{MSE of used data} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

This gives us the so-called **resubstitution** or apparent error.

Two types of predictions

On the other hand:

We can also measure how well our model $\hat{f}(x_0) = \hat{y}_0$ fits values y_0 that were NOT used to build the model:

$$\text{MSE of unused data} = \frac{1}{m}\sum_{0=1}^{m}(y_0 - \hat{y}_0)^2$$

This gives us the so-called **generalization** error.

In the Predictive Modeling arena ...

Prediction of seen observations

- Prediction $\hat{y}_i$ for y_i that was used to build the model
- *resubstitution* error: $e_i = y_i - \hat{y}_i$
- "already seen" MSE (less honest measure of accuracy)

Prediction of unseen observations

- Prediction $\hat{y}_0$ for y_0 that was NOT used to build the model
- *generalization* error: $e_0 = y_0 - \hat{y}_0$
- "unseen" MSE (more honest measure of accuracy)

Idea of Learning

If what we want is a model $f()$ that gives accurate predictions (i.e. predictions of unseen data), then how should we build $f()$?

By "how" I'm referring to the overall mechanics/strategy of the model building process.

This is where the concept of **learning** comes in.

Idea of Learning

- We are interested in getting a model with good generalization power.
- We want our model to *learn* as much as possible.
- It sounds reasonable to use as much data as possible so the obtained model is able to generalize adequately.

Learning Issues

- In theory, we could use all the available observations to fit a model.
- However, evaluating the model by using the data employed to fit the model produces a resubstitution or apparent measure of error.
- We are not really evaluating the model with new/unseen data.
- In other words, we are not really evaluating the generalization ability of the model.

Learning Issues

Building a model by using as many observations as possible looks like a good strategy.

And this strategy should work (in theory) as long as the used data is *representative* of the phenomenon under study.

The problem is: most of the time we don't know for sure if the available data is really representative.

Learning Issues

We run the risk that we use data that is not representative.

Perhaps the built model fits the used data extremely well, but it may lack generalization ability.

Moreover, by using all the available data we don't really have an honest measurement of the model accuracy (we only know the resubstitution MSE).

Learning Dilemmas

On one hand ...

- You want to use as much data as possible to train a model.
- You want to feed your model with as many examples as possible.

Learning Dilemmas

On one hand ...

- You want to use as much data as possible to train a model.
- You want to feed your model with as many examples as possible.

On the other hand ...

- You also want to know how will your model behave when new input data is available?
- How will your model generalize (when predicting unseen data)?

Dilemma

We face a BIG dilemma

- On one hand, you want to use as much data as possible to train a model.
- On the other hand, you want to have new/unseen data to evaluate the performance of your model and see how well it generalizes.

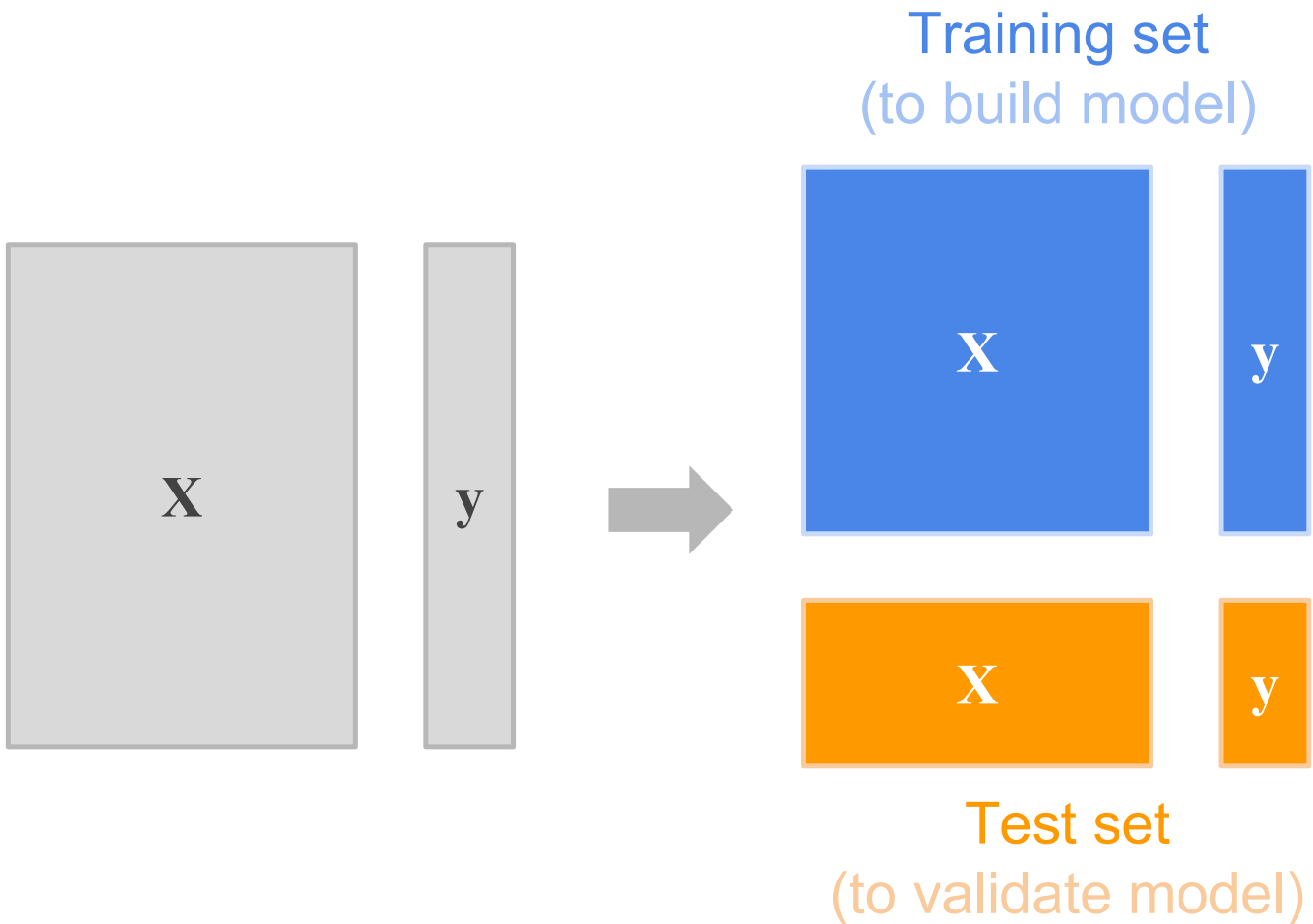
Training and Test Sets

The solution to this conundrum is to dispose of two data sets:

- **Training Dataset**: used to train the model
- **Test Dataset**: used to test the model and evaluate predicting performance

We assume that both the training and the test data sets are representative of the studied phenomenon. And that the test dataset is NOT used in the training stage.

Training and Test Sets

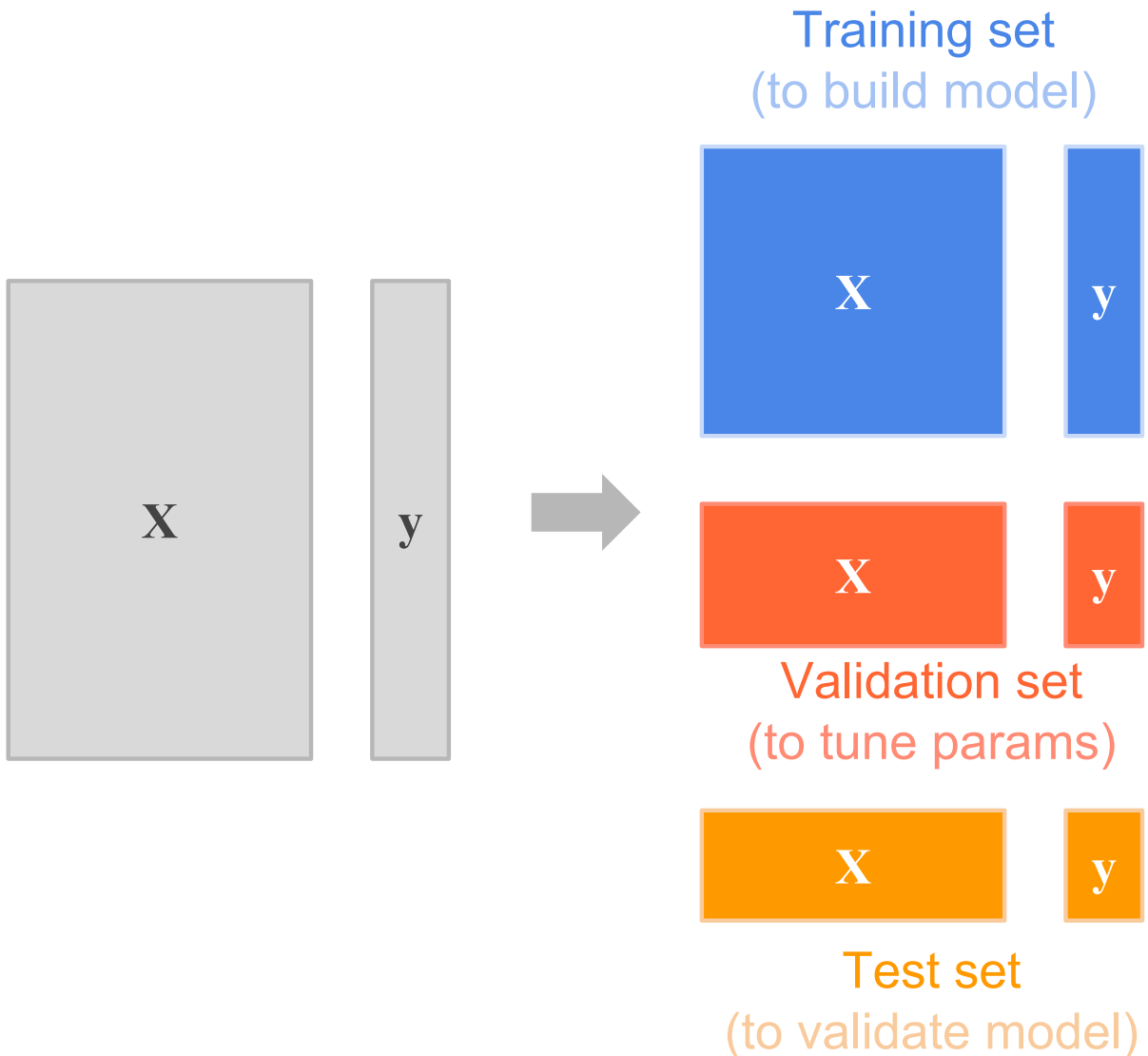


Training and Test Sets

Some authors go further and propose a three dataset scheme:

- Training set to fit the model
- Validation set to tune parameters
- Test set to assess predicting performance and select the best model

Training-Validation-Test Sets



Training and Test Sets

How do you decide what samples go into the training set, and what samples go into the test set?

Training and Test Sets

Selecting an adequate strategy to form the training and test sets obviously mainly depends on the amount of data.

With a vast amount of data, one could split the data set in halfs, one half as the training set, the other half as the test set.

The problem is: we don't always have vast amounts of data.

Either because the size of the data is small ("few observations"), or because the quality of the data is not good, and the best samples form a small subset.

Test-MSE

In order to have an "honest" measure of performance, we calculate **MSE on test data** (the so-called *test-MSE*).

Evaluating the predicting power on unseen data will give us a better idea of the model performance than when we just use the train-MSE.

Idea: Various Test Sets

Depending on how you form your train and test sests, you may end up with sets that are not truly representative of the studied phenomenon.

So instead of using just one test set, some authors propose to use several training-test sets.

Of course, with this suggestion we go back again to the issue of the size of the data. The solution to this limitation comes from using re-sampling methods.

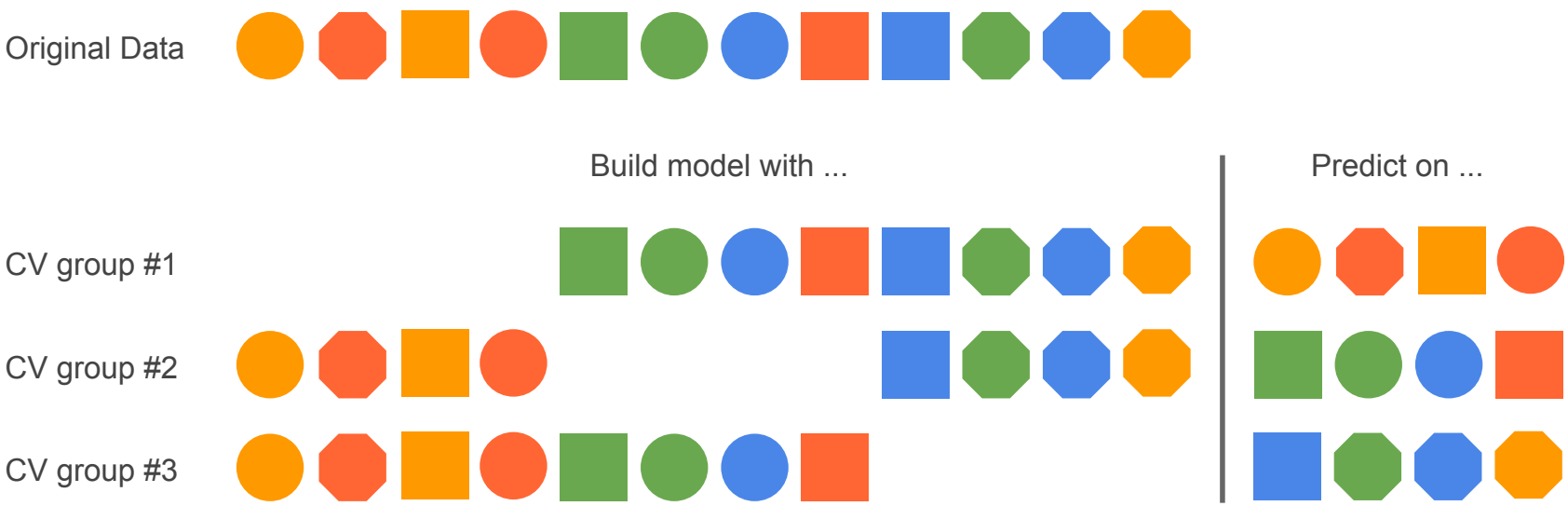
Cross-Validation

Cross-Validation (CV)

In Cross-Validation the main idea is to hold-out part of the data.
CV uses a systematic way of sampling the data.

- k-fold CV
- Leave-One-Out CV (loocv)
- 10-fold CV

Cross-validation three-fold



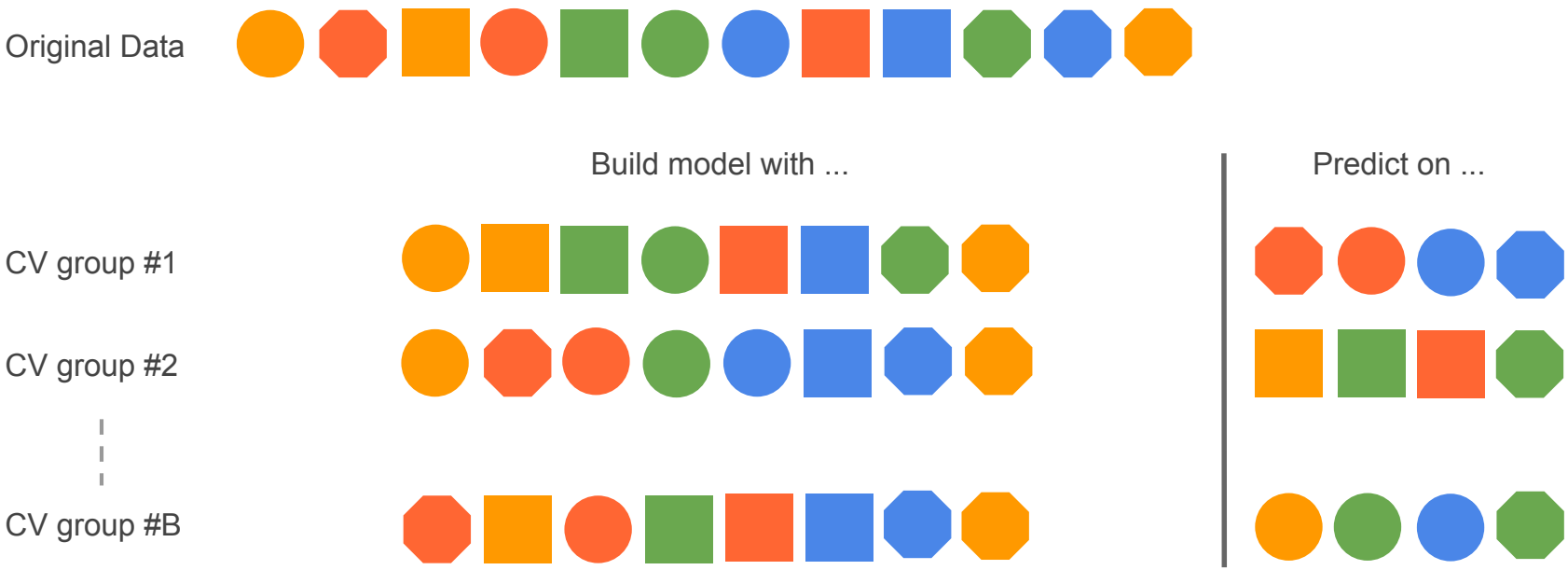
k-fold Cross-Validation

- The samples are randomly partitioned into k sets of roughly equal size.
- A model is fit using all the samples except the first subset.
- The hold-out samples are predicted by this model and used to estimate performance measures.
- The first subset is returned to the training set, and the procedure repeates with the second subset held out.
- The k resampled estimates of performance are summarized (usually with the mean and standard error).
- The choice of k is usually 5 or 10, but there is no formal rule.

Leave-One-Out Cross-Validation (loocv)

- A special version is the leave-one-out cross-validation.
- This is a special case for $k = 1$.
- Only one sample is held out at a time.
- The overall performance is calculated from the k individual held out predictions.

B Repeated cross-validation



B Repeated cross-validation

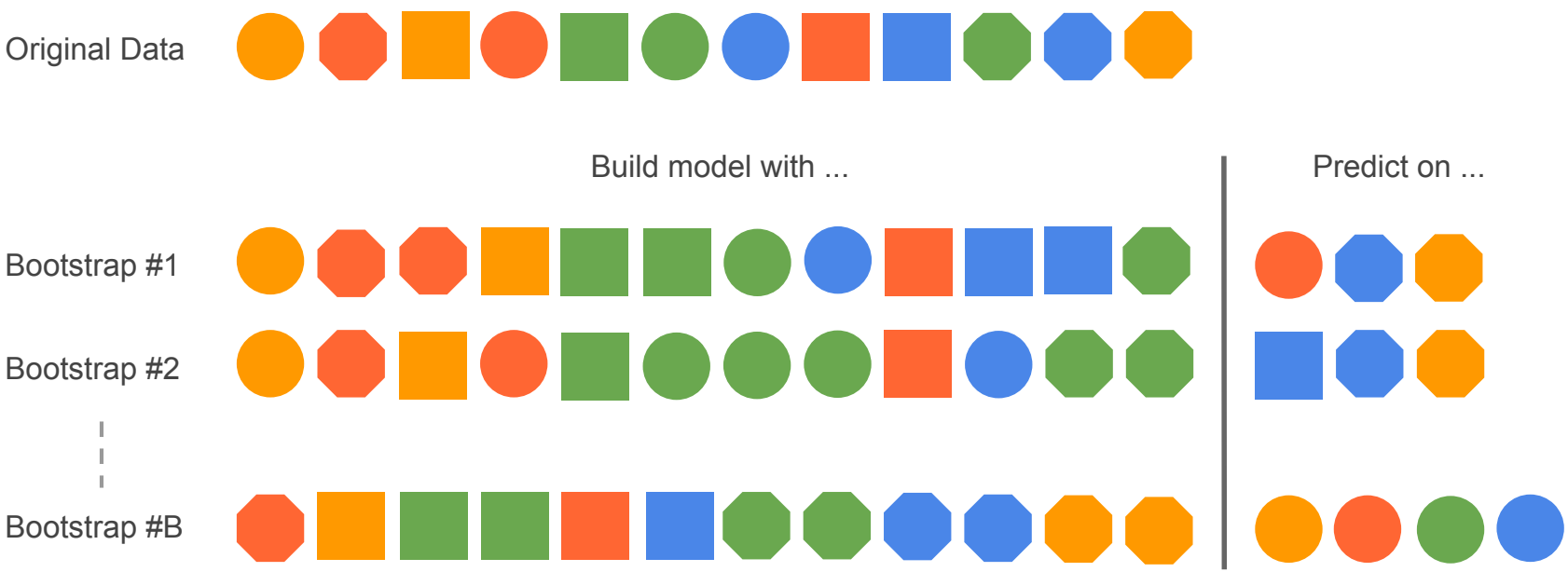
- Repeated training/set splits is also known as "leave-group-out" cross-validation.
- You simply create multiple splits of the data intro training and test sets.
- A good rule of thumb is about 75-80% training, 25-20% test.
- Increasing the number of subsets has the effect of decreasing the uncertainty of the performance estimates.
- It is suggested to choose a larger number of repetitions (say 50-200).

The Bootstrap

Bootstrap Resampling

- A bootstrap sample is a random sample of the data taken *with replacement*.
- The bootstrap sample is the same size as the original data set
- As a result, some samples will be represented multiple times in the bootstrap sample while other will not be selected at all.
- The samples not selected are usually referred to as the out-of-bag samples.

Bootstrap Resampling



For a given iteration, a model is built on the selected samples and is used to predict the out-of-bag samples.

Bootstrap Resampling

- On average, 63.2% of the data points in the bootstrap sample are represented at least once.
- Bootstrap resampling has bias similar to k-fold cross-validation when $k = 2$.
- If the training set size is small, this bias may be problematic, but will decrease as the training set sample size becomes larger.

References

- **Statistical Regression and Classification** by Norman Matloff (2017). CRC Press.
- **Statistical Learning from a Regression Perspective** by Richard Berk (2008). *Chapter 1: Regression Framework*. Springer.
- **Data Mining: Practical Machine Learning Tools and Techniques** by Ian Witten and Eibe Frank (2005). Elsevier.
- **Modern Multivariate Statistical Techniques** by Julian Izenman (2008). *Chapter 5: Prediction Accuracy and Model Assessment*. Springer.