AI200: APPLIED MACHINE LEARNING

MODEL EVALUATION FOR REGRESSION MODELS

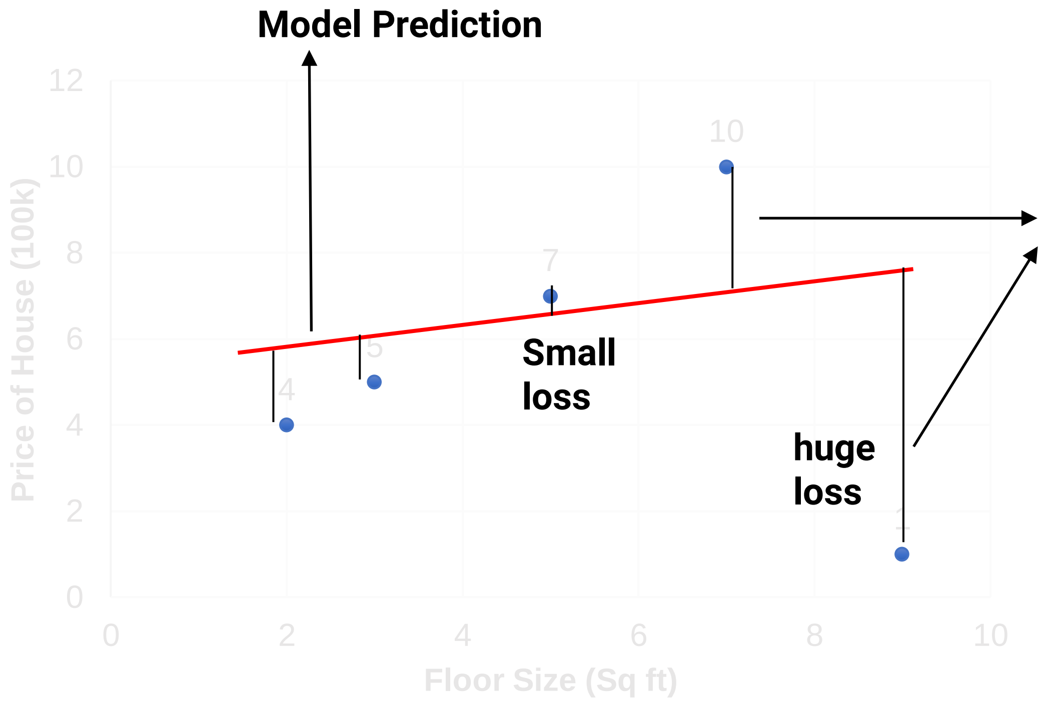
MODEL EVALUATION FOR REGRESSION MODELS

MODEL EVALUATION METRICS: RMSE

WHAT IS A RESIDUAL?

- When you generate a prediction for regression problems, it is unlikely you will get the exact same outcome. For instance, if a house is worth **$545,645.45** , it is not likely that a model will predict the same outcome to the exact decimals.

- **The difference between the predicted outcome and the actual outcome is termed as a residual**. You can also think of this as the magnitude of error in your prediction.



Error / loss / residual:
Penalty for a bad prediction. This is a number indicating how bad the model prediction was on a single example. If the prediction were perfect, the loss is zero; otherwise, the loss would deviate from zero.

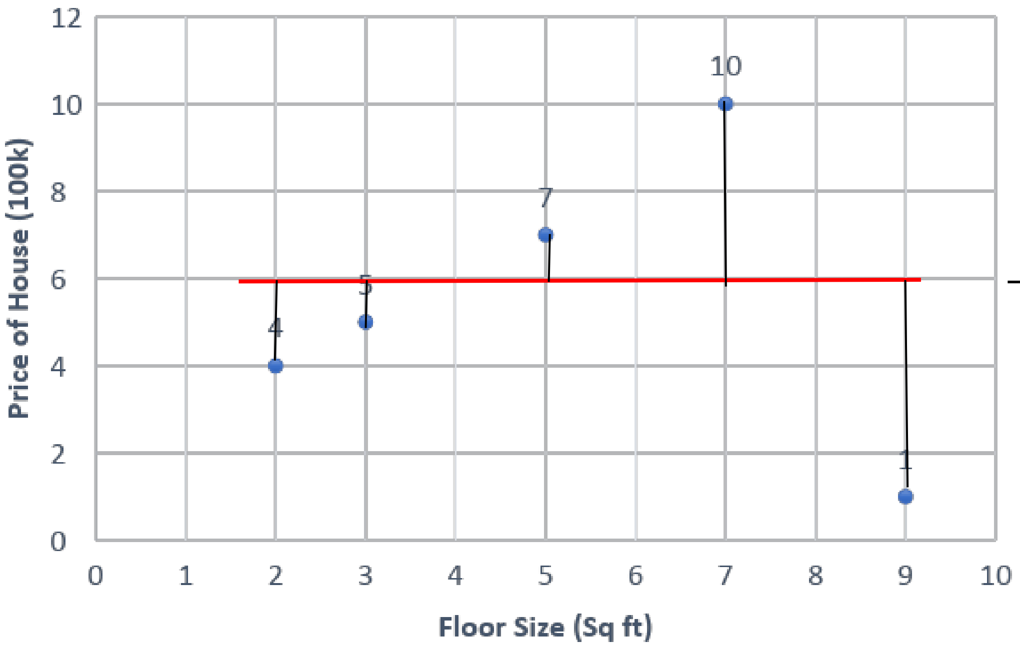
Graphically, it is the vertical distance / deviation between the prediction & the actual data point.

Note: The red line represents a regression model. We will cover details on this Linear Regression model shortly.

ROOT MEAN SQUARED ERROR (RMSE)

- So what if you **added up the residuals between all predicted outcomes and actual outcomes,** wouldn't this be a good indication of the collective error of the entire prediction model?

- However, you can't just simply add up all the errors values like this because:
  - Some errors are positive errors (more than zero) while others are negative
  - If you simply added them up together, the positive and negative errors would cancel each other out



If for example, if our model predicted a value of 600k for all houses

Actual	Prediction	Error
4	6	-2
5	6	-1
7	6	1
10	6	4
1	6	-5
Total		-3

As you can see, simply just adding the errors would just cancel each other out, and thus doesn't give an accurate representation of the total error of this model.

ROOT MEAN SQUARED ERROR (RMSE)

- Hence, we need a smarter way to calculate the error - **Root Mean Squared Error (RMSE)**
- The formula looks daunting, but the intuition is simple: this metric aggregates all the residuals.

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

Actual	**Prediction**	**Error**	**Squared of Error**
4	6	-2	$(-2)^2 = 4$
5	6	-1	$(-1)^2 = 1$
7	6	1	$(1)^2 = 1$
10	6	4	$(4)^2 = 16$
1	6	-5	$(-5)^2 = 25$
RMSE			$\sqrt{(4 + 1 + 1 + 16 + 25) / 5} = 3.066$

Squaring all values, means that you won't have negative terms. This also means the errors will not cancel out each other when added together

- If you don't understand the above, the most important thing is that you understand that:
  - The overall value indicates how wrong the model was overall.
  - The **smaller the RMSE, the better your model is.**

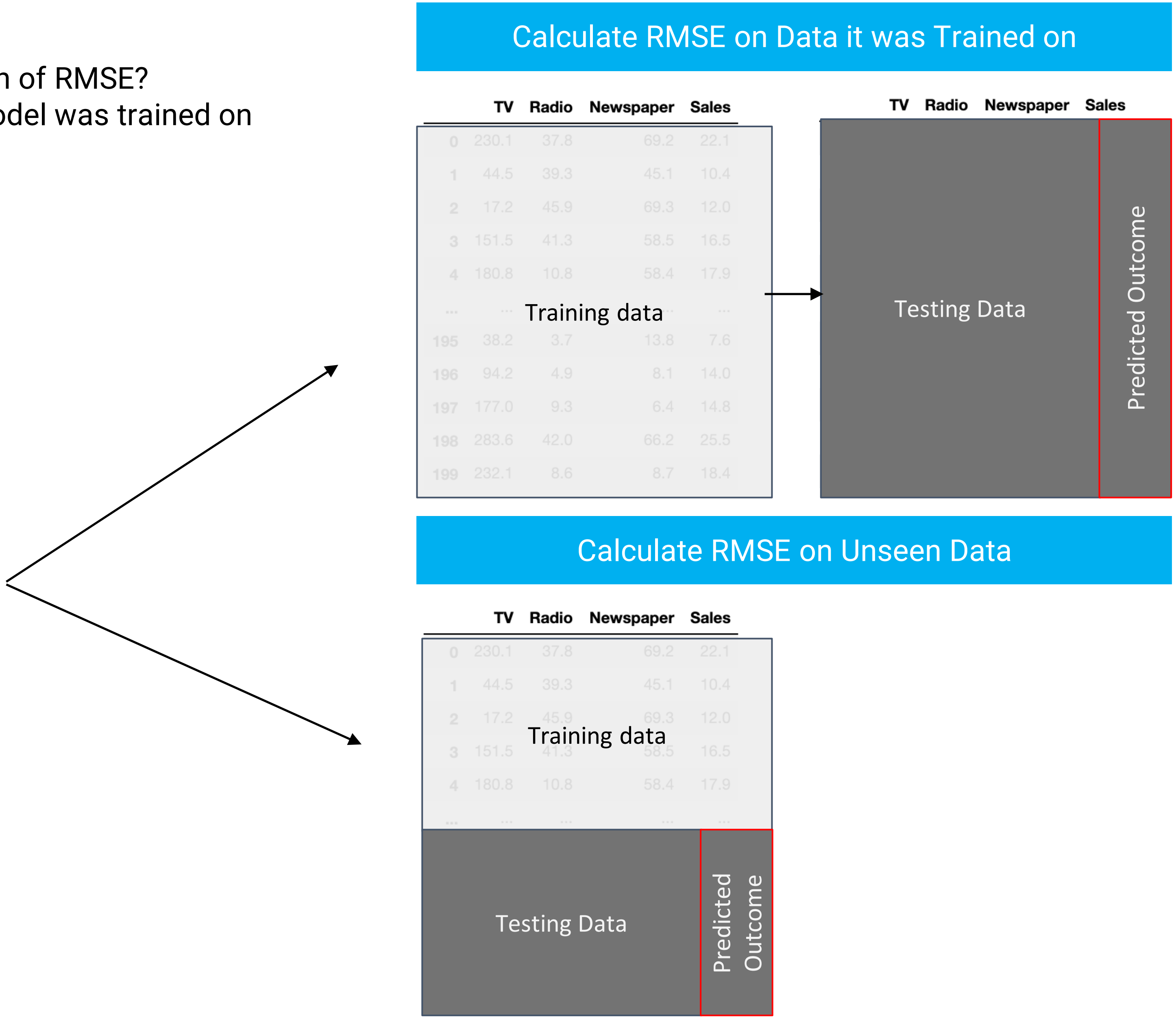
MODEL EVALUATION FOR REGRESSION MODELS

MODEL EVALUATION PROCEDURES

MODEL EVALUATION PROCEDURES

- Which do you think is a better estimation of RMSE?
  1. Calculate RMSE on the data the model was trained on
  2. Calculate RMSE on unseen data

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

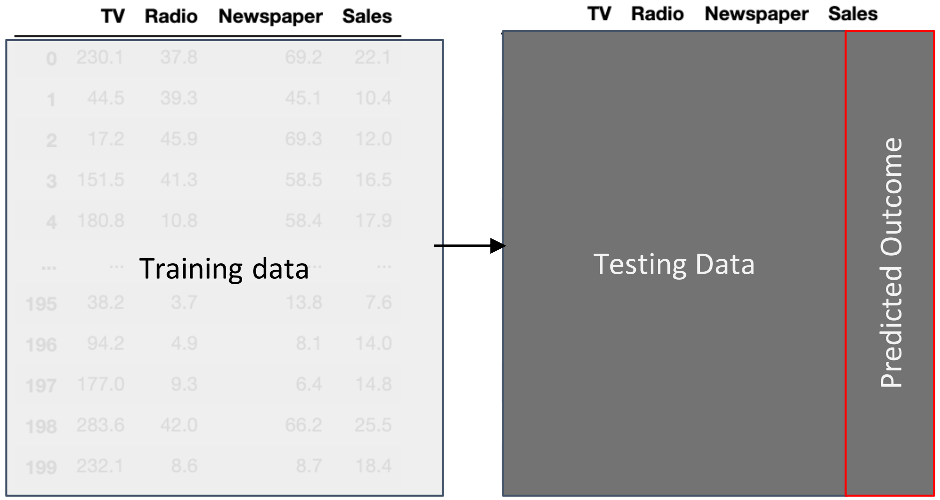


MODEL EVALUATION PROCEDURES

Calculate RMSE on Data it was Trained on

Analogy: To help you prepare for the final exam, you are ***given the answer sheet of the exact exam*** you're about to take. All you have to do is memorize as many of the answers as possible to do relatively well.

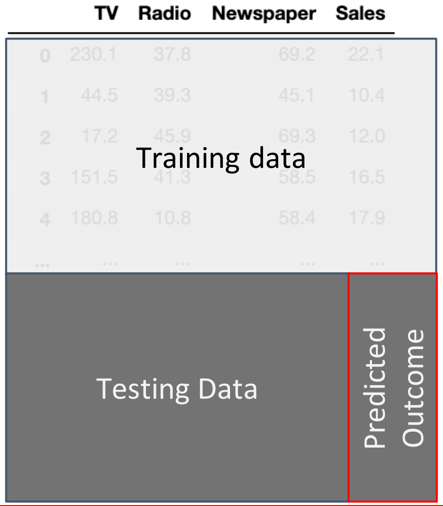
In this case, training error is <u>not a good estimate</u> of model accuracy beyond training data. This matters because the whole point of training models is to predict **unseen** data.



Calculate RMSE on Unseen Data

Analogy: You are given ***past-year exam papers*** with answers so you can look for trending topics, and understand the crux of each topic. Finally, when you take the unseen exam, you try to apply the concepts you learnt from past year papers.

This provides a fairer assessment of the model's performance in finding trends and generalising for other **unseen** data.

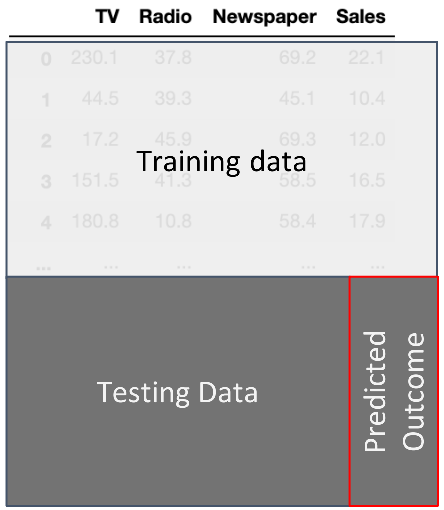


MODEL EVALUATION PROCEDURES

Unfortunately, we only get one set of data for model training, selection and testing.

Here are 2 ways of estimating test error with just one dataset:

- Train-Test Split
- Cross-Validation (more sophisticated, but preferred by industry)



This principle is adopted by various procedures. Here we share two commonly used procedures to gather model evaluation metrics:

- <u>Train-Test Split</u>
- <u>Cross-Validation</u>

MODEL EVALUATION PROCEDURE: TRAIN-TEST SPLIT

- Train-test split is a technique that estimates a model's test error by leaving out a subset of provided data during the training / fitting process and using it as a validation dataset. The specific steps are:

  - **Step 1**: Randomly divide available data into two parts (usual norm for splitting is 70-30):
    - **Training** set
    - **Validation** set
  - **Step 2**: Use the **training set** to fit the model
  - **Step 3**: Generate predictions for the **validation set** with the model
  - **Step 4**: Calculate the RMSE for the model based on the **validation set**

Age	Marital Status	Monthly Income	...	Job Satisfaction	...	Years at Company	Attrition
33	Single	4400	...	4	...	5	0
37	Married	3300	...	4	...	2	1
...							
27	Married	3200	...	3	...	1	0
...							
25	Single	3000	...	3	...	1	0

Instances/ Observations

Features / Attributes / Input Variables

Class label / Target Variable

MODEL EVALUATION PROCEDURE: TRAIN-TEST SPLIT

- Train-test split is a technique that estimates a model's test error by leaving out a subset of provided data during the training / fitting process and using it as a validation dataset. The specific steps are:
    - **Step 1**: Randomly divide available data into two parts (usual norm for splitting is 70-30):
        - **Training** set
        - **Validation** set
    - **Step 2**: Use the **training set** to fit the model
    - **Step 3**: Generate predictions for the **validation set** with the model
    - **Step 4**: Calculate the RMSE for the model based on the **validation set**

Age	Marital Status	Monthly Income	...	Job Satisfaction	...	Years at Company	Attrition
33	Single	4400	...	4	...	5	0
37	Married	3300	...	4	...	2	1
...							
27	Married	3200	...	3	...	1	0
...							
25	Single	3000	...	3	...	1	0

X_train

y_train

X_test

y_test

Instances/ Observations

Features / Attributes / Input Variables

Class label / Target Variable

MODEL EVALUATION PROCEDURE: CROSS VALIDATION

- Cross Validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called **k** which determines the number of groups to split the dataset into.
- As such, the procedure is often called k-fold **cross-validation.** These are the steps:

  - **Step 1**: Randomly split the dataset into K equal partitions or 'folds'
  - **Step 2**: Use **Fold 1 as validation set**, and the rest of the **other folds as training data**
  - **Step 3**: Fit the model on the **training set** and estimate the model's RSME using the **validation set**
  - **Step 4**: Repeat step 2-3 **using a different fold as validation set** at each iteration
  - **Step 5**: Take the average error as the estimate of test error

Age	Marital Status	Monthly Income	...	Job Satisfaction	...	Years at Company	Attrition
33	Single	4400	...	4	...	5	0
37	Married	3300	...	4	...	2	1
...							
27	Married	3200	...	3	...	1	0
...							
25	Single	3000	...	3	...	1	0

Instances/ Observations

Features / Attributes / Input Variables

Class label / Target Variable

MODEL EVALUATION PROCEDURE: CROSS VALIDATION

- Cross Validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called **k** which determines the number of groups to split the dataset into.
- As such, the procedure is often called k-fold **cross-validation.** These are the steps:
  - **Step 1**: Randomly split the dataset into K equal partitions or 'folds'
  - **Step 2**: Use **Fold 1 as validation set**, and the rest of the **other folds as training data**
  - **Step 3**: Fit the model on the **training set** and estimate the model's RSME using the **validation set**
  - **Step 4**: Repeat step 2-3 **using a different fold as validation set** at each iteration
  - **Step 5**: Take the average error as the estimate of test error

Age	Marital Status	Monthly Income	...	Job Satisfaction	...	Years at Company	Attrition
33	Single	4400	Fold-1	4	...	5	0 Fold-1
37	Married	3300	Fold-2	4	...	2	1 Fold-2
...							
27	Married	3200	...	3	...	1	0 ...
...							
25	Single	3000	Fold-k	3	...	1	0 Fold-k

If k = 10, that means each fold contains 10% of the data, where the entire dataset is split into 10 equal parts

MODEL EVALUATION PROCEDURE: CROSS VALIDATION

- Cross Validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called **k** which determines the number of groups to split the dataset into.
- As such, the procedure is often called k-fold **cross-validation.** These are the steps:
  - **Step 1**: Randomly split the dataset into K equal partitions or 'folds'
  - **Step 2**: Use **Fold 1 as validation set**, and the rest of the **other folds as training data**
  - **Step 3**: Fit the model on the **training set** and estimate the model's RSME using the **validation set**
  - **Step 4**: Repeat step 2-3 **using a different fold as validation set** at each iteration
  - **Step 5**: Take the average error as the estimate of test error

Age	Marital Status	Monthly Income	...	Job Satisfaction	...	Years at Company	Attrition
X_test							y_test
37	Married	3300	...	4	...	2	1
...							
27	Married	3200	X_train	[illegible]	...	1	y_train
...							
25	Single	3000	...	3	...	1	0

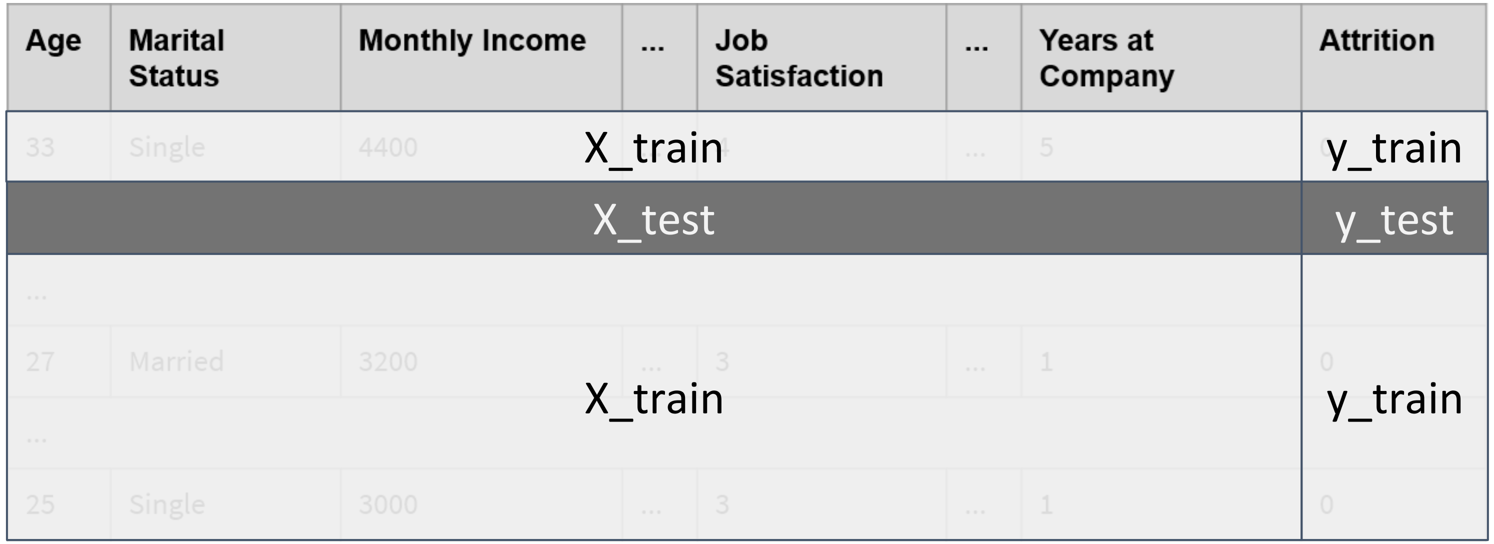
In the first iteration, we use Fold-2 to Fold-10 as train set, and **Fold-1 as validation set.**

We evaluate RMSE on Fold-1, which is ***unseen data*** by the model trained in that fold.

MODEL EVALUATION PROCEDURE: CROSS VALIDATION

- Cross Validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called **k** which determines the number of groups to split the dataset into.
- As such, the procedure is often called k-fold **cross-validation.** These are the steps:

  - **Step 1**: Randomly split the dataset into K equal partitions or 'folds'
  - **Step 2**: Use **Fold 1 as validation set**, and the rest of the **other folds as training data**
  - **Step 3**: Fit the model on the **training set** and estimate the model's RSME using the **validation set**
  - **Step 4**: Repeat step 2-3 **using a different fold as validation set** at each iteration
  - **Step 5**: Take the average error as the estimate of test error



We repeat this with the subsequent fold.

MODEL EVALUATION PROCEDURE: CROSS VALIDATION

- Cross Validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called **k** which determines the number of groups to split the dataset into.
- As such, the procedure is often called k-fold **cross-validation.** These are the steps:

    - **Step 1**: Randomly split the dataset into K equal partitions or 'folds'
    - **Step 2**: Use **Fold 1 as validation set**, and the rest of the **other folds as training data**
    - **Step 3**: Fit the model on the **training set** and estimate the model's RSME using the **validation set**
    - **Step 4**: Repeat step 2-3 **using a different fold as validation set** at each iteration
    - **Step 5**: Take the average error as the estimate of test error

Age	Marital Status	Monthly Income	...	Job Satisfaction	...	Years at Company	Attrition
33	Single	4400	...	4	...	5	0
37	Married	3300	...	4	...	2	1
...		X_train					y_train
27	Married	3200	...	3	...	1	0
...							
		X_test					y_test

And we keep doing this until each fold has served as the validation dataset.

MODEL EVALUATION PROCEDURE: CROSS VALIDATION

- Pros: Cross Validation / K-Fold Cross Validation provides a **more accurate estimate of test error**, since it is less sensitive to how the dataset was split into train & validation datasets. It also **uses the dataset more efficiently** than just a single train-test split

- Cons: However, the **tradeoff is the computational cost**. A k-fold cross validation takes **k times longer** than the train-split test approach.