Performance Evaluation

Peerapon Vateekul, Ph.D.

peerapon.v@4amconsult.com

Outlines

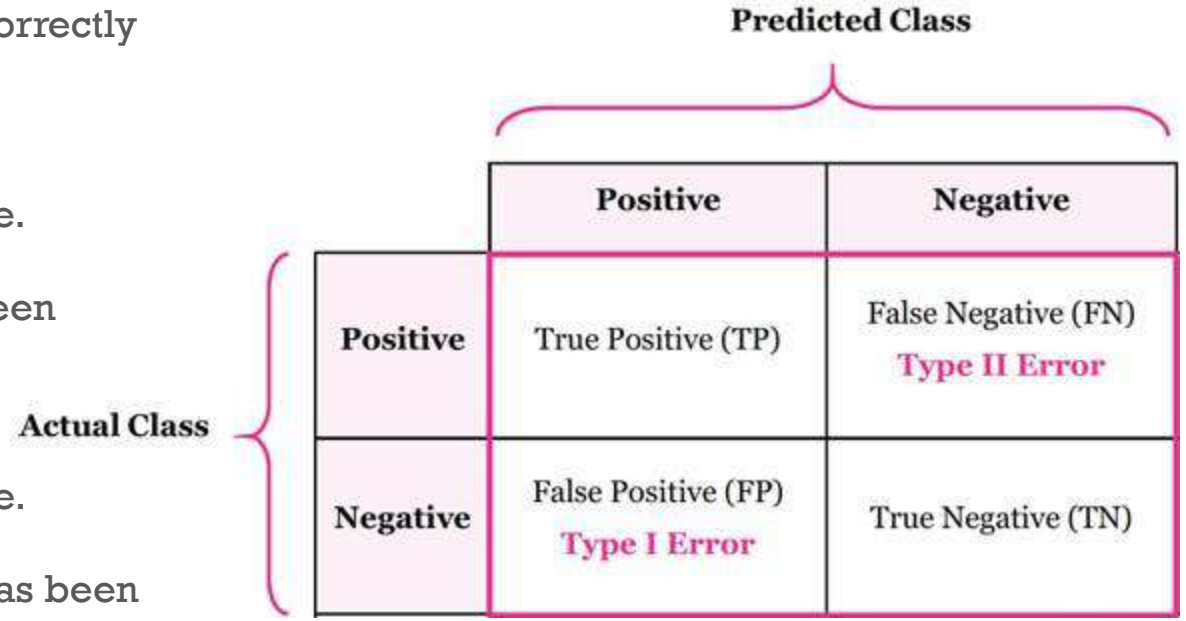
- Classification
  - Confusion matrix
    - TP, FP, TN, FN
    - Accuracy, Precision, Recall, F1
  - Micro, Macro
  - ROC (graph)
- Regression
  - MSE
  - RMSE
  - R2

Classification

- Classification
  - Confusion matrix
    - TP, FP, TN, FN
    - Accuracy, Precision, Recall, F1
  - Micro, Macro
  - ROC (graph)

Confusion matrix

- **True Positive** (TP)
  - Number of **positive** class **correctly** identified as positive
  - Example: Given class is spam and the classifier has been correctly predicted it as spam.
- **False Negative** (FN)
  - Number of **positive** class **incorrectly** identified as negative.
  - Example: Given class is spam however, the classifier has been incorrectly predicted it as non-spam.
- **False positive** (FP)
  - Number of **negative** class **incorrectly** identified as positive.
  - Example: Given class is non-spam however, the classifier has been incorrectly predicted it as spam.
- **True Negative** (TN)
  - Number of **negative** class **correctly** identified as negative.
  - Example: Given class is spam and the classifier has been correctly predicted it as negative.



Reference : https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

Performance Evaluation

- Accuracy
- Precision
- Recall
- F1-score

		Predicted Class: Positive	Predicted Class: Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

		Predicted Class: Bad	Predicted Class: Good
Actual Class	Bad	TP=45	FN=20
	Good	FP=5	TN=30

Reference : https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

Confusion matrix command in sklearn

```
>>> from sklearn.metrics import confusion_matrix
>>> y_true = [2, 0, 2, 2, 0, 1]
>>> y_pred = [0, 0, 2, 2, 0, 2]
>>> confusion_matrix(y_true, y_pred)
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

```
>>> y_true = ["cat", "ant", "cat", "cat", "ant", "bird"]
>>> y_pred = ["ant", "ant", "cat", "cat", "ant", "cat"]
>>> confusion_matrix(y_true, y_pred, labels=["ant", "bird", "cat"])
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

Reference : https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

Performance Evaluation

- **Accuracy**
  - Most intuitive performance measure
  - The proportion of the total number of predictions that are correct
- *Accuracy = (45+30)/(45+20+5+30) = 75%*
- The 75% of examples are correctly classified by the classifier

		Predicted Class: Bad	Predicted Class: Good
Actual Class	Bad	TP=45	FN=20
	Good	FP=5	TN=30

Reference : https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

Performance Evaluation

- ***Recall or* Sensitivity**
  - *True Positive Rate*
  - It is measure of **positive** examples labeled as **positive** by classifier
- *Sensitivity = 45/(45+20) = 69.23%*
- The *69.23%* bad defaults are correctly classified

		Predicted Class: Bad	Predicted Class: Good
Actual Class	Bad	TP=45	FN=20
	Good	FP=5	TN=30

Reference : https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

Performance Evaluation

- **Specificity**
  - *True Negative Rate*.
  - It is measure of **negative** examples labeled as **negative** by classifier
- *specificity = 30/(30+5) = 85.71%*
- The 85.71% good defaults are accurately classified

		Predicted Class: Bad	Predicted Class: Good
Actual Class	Bad	TP=45	FN=20
	Good	FP=5	TN=30

Reference : https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

Performance Evaluation

- **Precision**
  - It is ratio of total number of correctly classified **positive** examples and the total number of predicted **positive** examples
  - It shows correctness achieved in **positive** prediction.
- *Precision = 45/(45+5)= 90%*
- The 90% of examples are classified as bad defaults are actually bad defaults

		Predicted Class: Bad	Predicted Class: Good
Actual Class	Bad	TP=45	FN=20
	Good	FP=5	TN=30

Reference : https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

Performance Evaluation

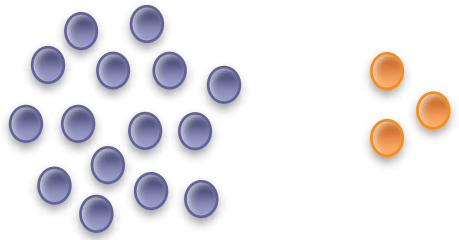
- **F1 score**
  - It is a weighted average of the recall (sensitivity) and precision
  - F1 score might be good choice when you seek to balance between Precision and Recall
- It helps to compute recall and precision in one equation so that the problem to distinguish the models with low recall and high precision or vice versa could be solved.
  - *Precision = 45/(45+5)= 90%*
  - *Recall = 45/(45+20) = 69.23%*
  - *F1-Score = 2*(90*69.23)/(90+69.23) = 78.26%*

$$\text{F1 Score} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

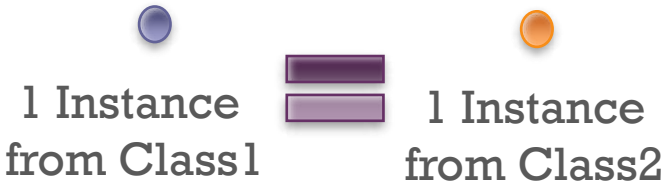
Performance Evaluation

- Which one is the best
  - General-> F1 score
  - General + All class is equally important -> Accuracy
  - General + Some class is more important than other-> F1 score
  - Domain
    - Health care -> Recall

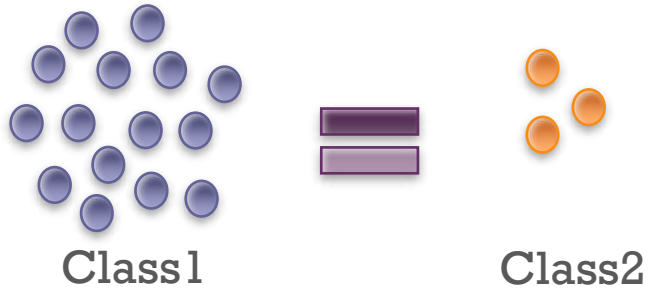
Macro and Micro



- Micro
  - Treat all **instances** is equally important



- Macro
  - Treat all **classes** is equally important



Macro and Micro

- Micro
  - Treat all instances is equally important
  - Sum up TP, FP and FN and then compute Precision, Recall, F1-score

```
from sklearn.metrics import classification_report
```

```
print(classification_report(test_predict, test_df[target_col]))
```

	precision	recall	f1-score	support
0	0.18	0.58	0.28	955
1	0.99	0.94	0.97	44045
accuracy			0.94	45000
macro avg	0.59	0.76	0.62	45000
weighted avg	0.97	0.94	0.95	45000

$$\text{F1 Score} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

From Example
- Class 0 has 955 instances
- Class 1 has 44045 instances
- Thus, Micro avg F1 ~ F1 of class 1

Reference : https://medium.com/@ramit.singh.pahwa/micro-macro-precision-recall-and-f-score-44439de1a044

Macro and Micro

- Macro
  - Treat all classes is equally important
  - Compute Precision, Recall, F1-score each class then compute average of them

```
from sklearn.metrics import classification_report
```

```
print(classification_report(test_predict, test_df[target_col]))
```

	precision	recall	f1-score	support
0	0.18	0.58	0.28	955
1	0.99	0.94	0.97	44045
accuracy			0.94	45000
macro avg	0.59	0.76	0.62	45000
weighted avg	0.97	0.94	0.95	45000

$$Macro - Precision = \frac{Pecision1 + Precision2}{2}$$

$$Macro - Recall = \frac{Recall1 + Recall2}{2}$$

$$Macro - F - Score = 2.\frac{Macro - Precision\,.\,Macro - Recall}{Macro - Precision + Macro - Recall}$$

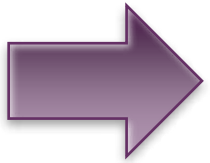
ROC

- AUC - ROC curve
  - is a performance measurement for classification problem at various thresholds settings
  - ROC is a probability curve
  - AUC represents degree or measure of separability
  - It tells how much model is capable of distinguishing between classes.
  - Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s.
  - By analogy, Higher the AUC, better the model is at distinguishing between patients with disease and no disease
- The ROC curve is plotted with TPR against the FPR where TPR is on y-axis and FPR is on the x-axis.

Reference : https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

ROC

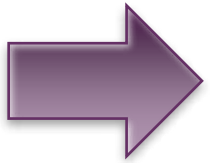
ID	Actual	Prob
x1	1	0.90
x2	1	0.87
x3	0	0.71
x4	1	0.65
x5	0	0.55
x6	1	0.42
x7	1	0.21
x8	0	0.11
x9	0	0.05
x10	0	0.02



Criteria	TP	TP Rate	FN	FN Rate
1.0	0	0.00	0	0.00
0.9	1	0.20	0	0.00
0.8	2	0.40	0	0.00
0.7	2	0.40	1	0.20
0.6	3	0.60	1	0.20
0.5	3	0.60	2	0.40
0.4	4	0.80	2	0.40
0.3	4	0.80	2	0.40
0.2	5	1.00	2	0.40
0.1	5	1.00	3	0.60
0.0	5	1.00	5	1.00

ROC

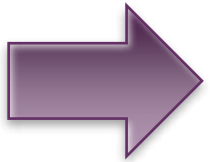
ID	Actual	Prob
x1	1	0.90
x2	1	0.87
x3	0	0.71
x4	1	0.65
x5	0	0.55
x6	1	0.42
x7	1	0.21
x8	0	0.11
x9	0	0.05
x10	0	0.02



Criteria	TP	TP Rate	FN	FN Rate
1.0	0	0.00	0	0.00
0.9	1	0.20	0	0.00
0.8	2	0.40	0	0.00
0.7	2	0.40	1	0.20
0.6	3	0.60	1	0.20
0.5	3	0.60	2	0.40
0.4	4	0.80	2	0.40
0.3	4	0.80	2	0.40
0.2	5	1.00	2	0.40
0.1	5	1.00	3	0.60
0.0	5	1.00	5	1.00

ROC

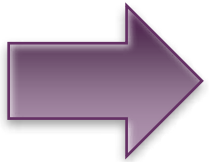
ID	Actual	Prob
x1	1	0.90
x2	1	0.87
x3	0	0.71
x4	1	0.65
x5	0	0.55
x6	1	0.42
x7	1	0.21
x8	0	0.11
x9	0	0.05
x10	0	0.02



Criteria	TP	TP Rate	FN	FN Rate
1.0	0	0.00	0	0.00
0.9	1	0.20	0	0.00
0.8	2	0.40	0	0.00
0.7	2	0.40	1	0.20
0.6	3	0.60	1	0.20
0.5	3	0.60	2	0.40
0.4	4	0.80	2	0.40
0.3	4	0.80	2	0.40
0.2	5	1.00	2	0.40
0.1	5	1.00	3	0.60
0.0	5	1.00	5	1.00

ROC

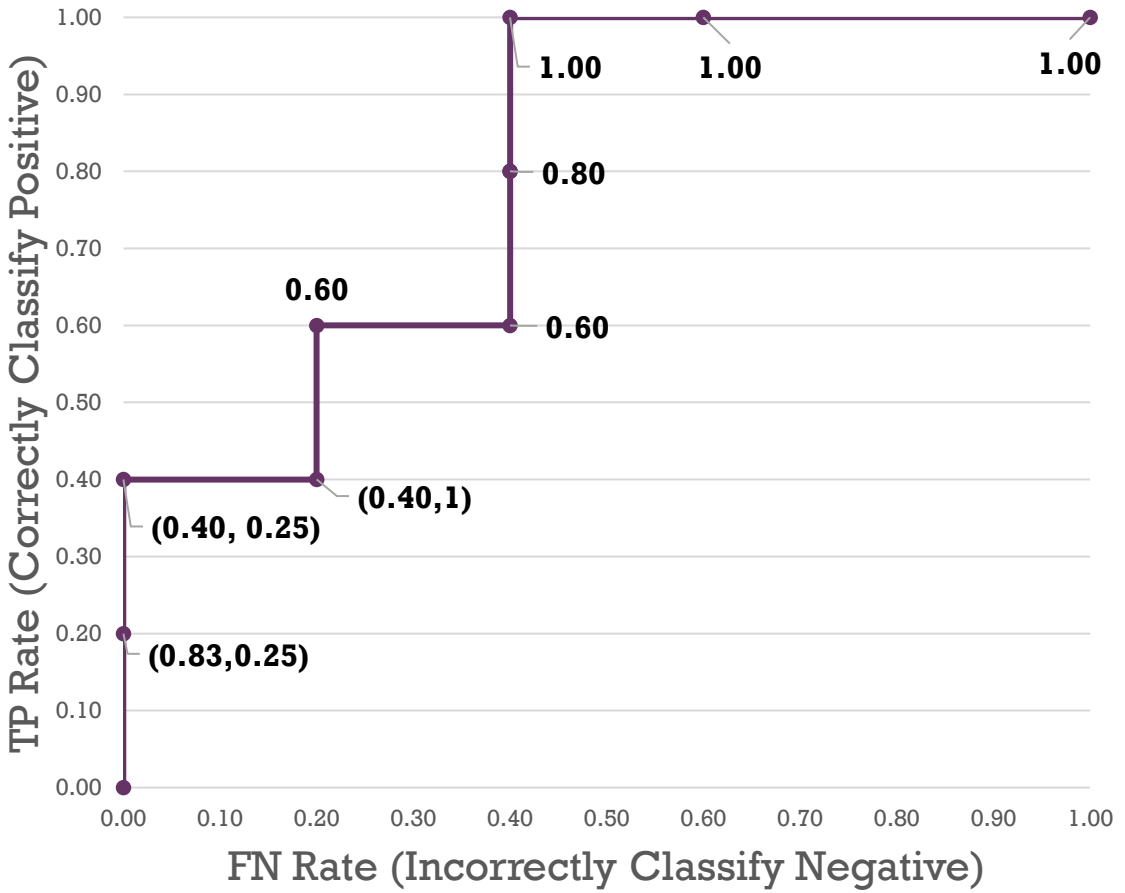
ID	Actual	Prob
x1	1	0.90
x2	1	0.87
x3	0	0.71
x4	1	0.65
x5	0	0.55
x6	1	0.42
x7	1	0.21
x8	0	0.11
x9	0	0.05
x10	0	0.02



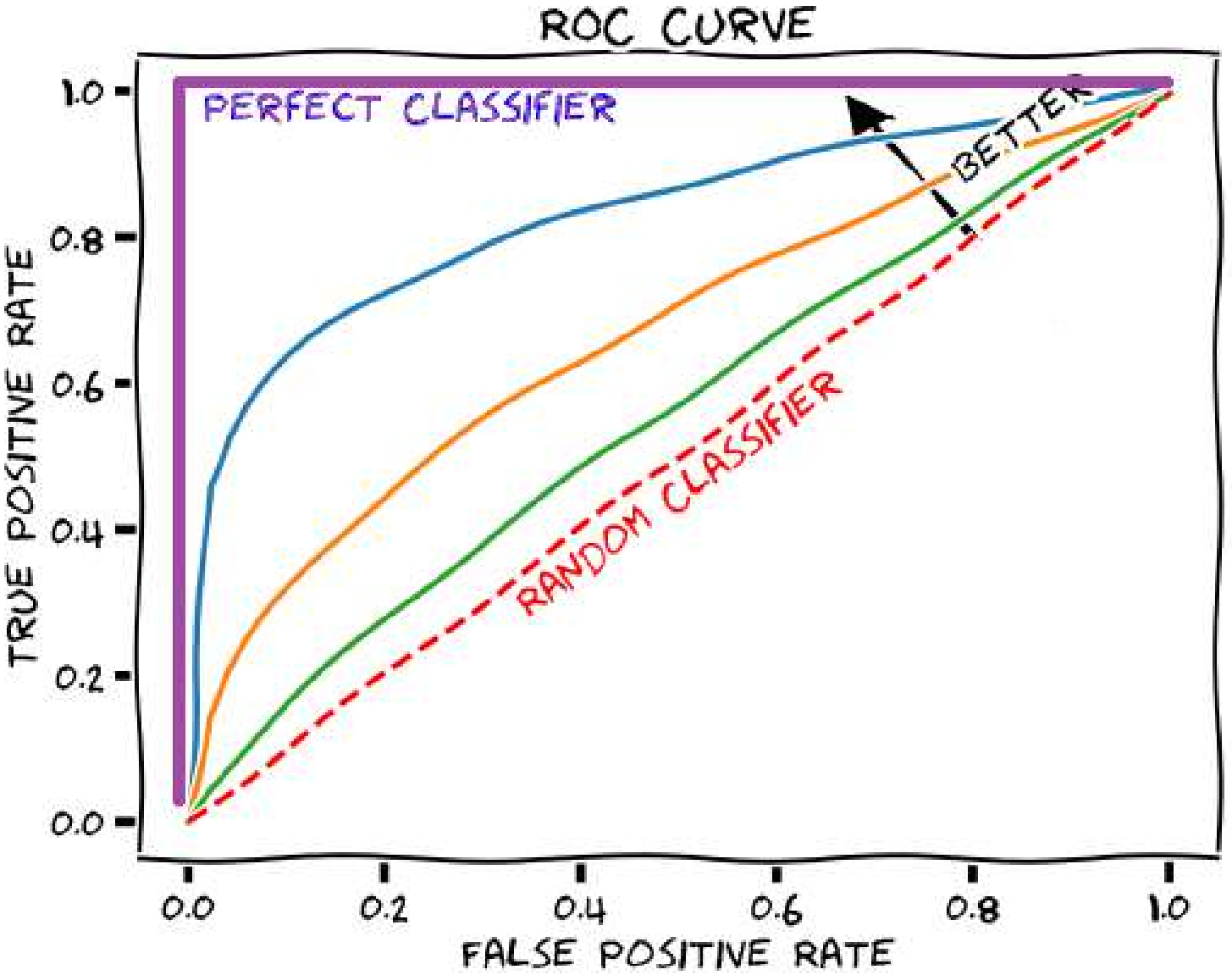
Criteria	TP	TP Rate	FN	FN Rate
1.0	0	0.00	0	0.00
0.9	1	0.20	0	0.00
0.8	2	0.40	0	0.00
0.7	2	0.40	1	0.20
0.6	3	0.60	1	0.20
0.5	3	0.60	2	0.40
0.4	4	0.80	2	0.40
0.3	4	0.80	2	0.40
0.2	5	1.00	2	0.40
0.1	5	1.00	3	0.60
0.0	5	1.00	5	1.00

ROC

Criteria	TP	TP Rate	FN	FN Rate
1.0	0	0.00	0	0.00
0.9	1	0.20	0	0.00
0.8	2	0.40	0	0.00
0.7	2	0.40	1	0.20
0.6	3	0.60	1	0.20
0.5	3	0.60	2	0.40
0.4	4	0.80	2	0.40
0.3	4	0.80	2	0.40
0.2	5	1.00	2	0.40
0.1	5	1.00	3	0.60
0.0	5	1.00	5	1.00



ROC



Reference : https://glassboxmedicine.com/2019/02/23/measuring-performance-auc-auroc/

Regression

- Regression
  - MSE
  - RMSE
  - R2

MSE

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)}^{2}$$

The square of the difference between actual and predicted

- Mean Square Error (MSE)
  - Average of the square of the errors

ID	Actual	Predict	Error	Sqrt Error	Sum Sqrt Error	MSE
x1	1	0.90	0.10	0.01	1.93	0.19
x2	1	0.87	0.13	0.02		
x3	0	0.71	-0.71	0.50		
x4	1	0.65	0.35	0.12		
x5	0	0.55	-0.55	0.30		
x6	1	0.42	0.58	0.34		
x7	1	0.21	0.79	0.62		
x8	0	0.11	-0.11	0.01		
x9	0	0.05	-0.05	0.00		
x10	0	0.02	-0.02	0.00		

Reference : https://www.dataquest.io/blog/understanding-regression-error-metrics/

RMSE

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

- Root Mean Square Error (RMSE)
  - Square root of the mean square error

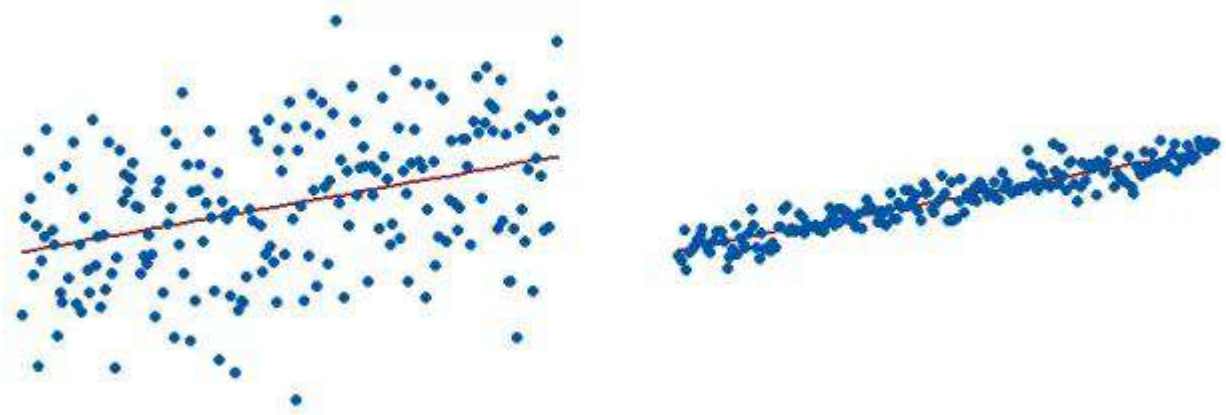
ID	Actual	Predict	Error	Error^2	Sum Error^2	MSE	RMSE
x1	1	0.90	0.10	0.01	1.93	0.19	0.44
x2	1	0.87	0.13	0.02			
x3	0	0.71	-0.71	0.50			
x4	1	0.65	0.35	0.12			
x5	0	0.55	-0.55	0.30			
x6	1	0.42	0.58	0.34			
x7	1	0.21	0.79	0.62			
x8	0	0.11	-0.11	0.01			
x9	0	0.05	-0.05	0.00			
x10	0	0.02	-0.02	0.00			

Reference : https://towardsdatascience.com/what-does-rmse-really-mean-806b65f2e48e

R^2

$$R^2 = 1 - \frac{\text{Unexplained Variation}}{\text{Total Variation}}$$

- R Square
  - Coefficient of determination
  - Evaluates the scatter of the data points around the fitted regression line
  - higher R-squared values represent smaller differences between the observed data and the fitted values
  - Always between 0 and 100
  - Usually, the larger the R^2. the better the regression model fits your observations.



  - The R-squared for the regression model on the left is 15%, and for the model on the right it is 85%.

Reference : https://statisticsbyjim.com/regression/interpret-r-squared-regression/

+

Any Questions?