University of Exeter

Metrics of Evaluation: What is a good model?

Week 07-BEM2031

Term2: 2023/24

Homework, assignment and final project: PROGRESS!

- After today, you will have everything you need to complete the assignment due March 15
- Video and materials uploaded for final project – start reading it now
- Code snippets – can be used for your final assignment. This document came from students from previous years – so I don't recommend all of it!

Month	Week	Monday	Tuesday	Wednesday	Thursday	Friday
January	**1**	15	16	17	18	19
	2	22	23	24	25	26
	3	29	30	31	1	2
February	**4**	5	6	7	8	9
	5	12	13	14	15	16
	6	Reading week				
	7	26	**27**	28	29	1
March	**8**	4	5	6	7	8
	9	11	12	13	14 **Assignment**	15
	10	18	19	20	21	22
	11	25	26	27	28 **Final Project**	29 (P/H)

Today:

- Supervised learning
  - Regression
  - Classification
- Metrics of evaluation (regression and classification)
  - Confusion matrix, ROC, AUC
- Overfitting/underfitting
- Cross-validation

Today:

[StatQuest: Confusion Matrix]

[StatQuest: Sensitivity and Specificity]

[StatQuest: ROC and AUC]

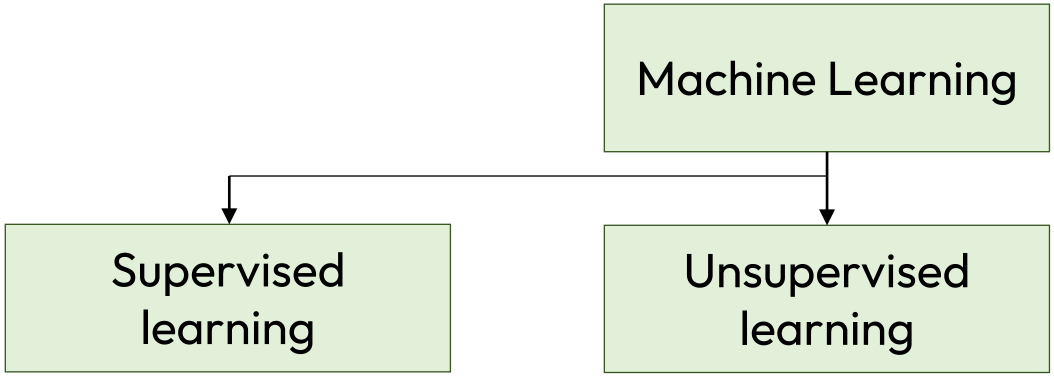
[StatQuest: Cross Validation]

[StatQuest: Bias and Variance Tradeoff]

Types of Analytics:

- Descriptive Analytics: WHAT happened (or is happening)?
- Diagnostic Analytics: WHY did it happen?
- Predictive Analytics: WHAT is likely to happen in the future?
- Prescriptive Analytics: WHAT can we do about it?

Supervised vs Unsupervised methods



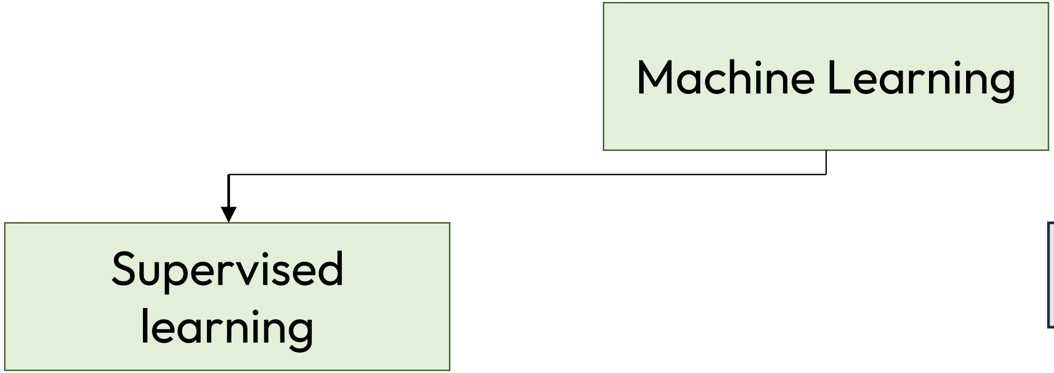
Supervised methods have a target, an objective.

“Can we find groups of customers who have particularly high likelihoods of cancelling their service soon after their contracts expire?”

Unsupervised methods have no specific target.

“Do our customers naturally fall into different groups

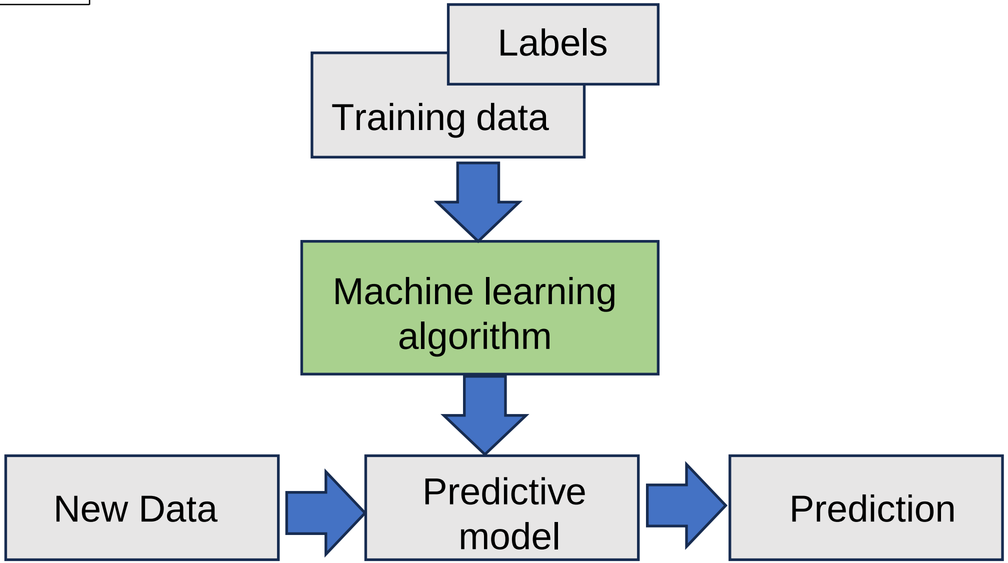
Supervised vs Unsupervised methods



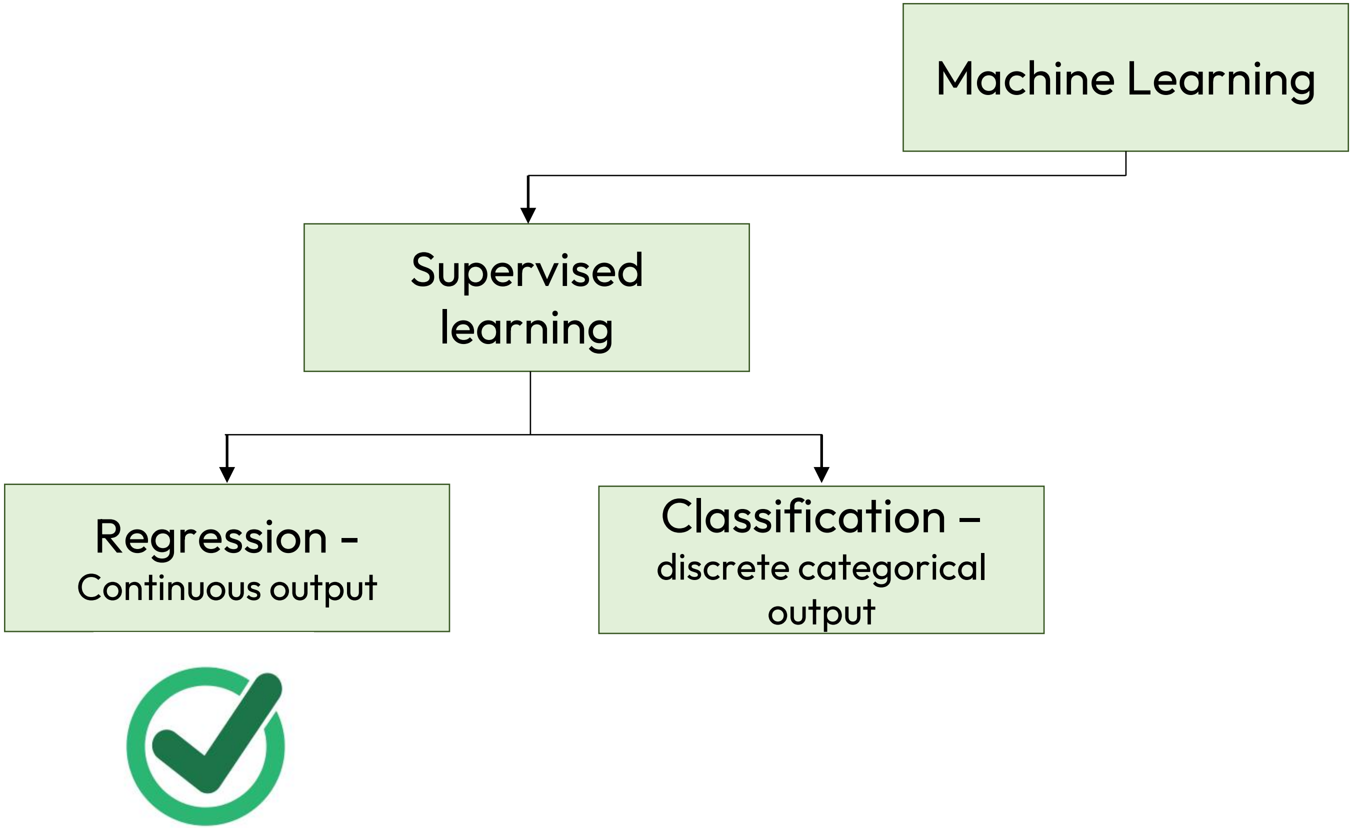
Label: Variable we are predicting (the dependent variable) (y)

Features: the attributes or variables which describe the data (independent variables) $\{x_1, x_2, \ldots, x_N\}$

Unseen (new) data: test the performance of the model *y=f(x)* on unlabelled data



Types of supervised learning



Example: House Price Prediction

Given a set of input features (which may influence the price of a house), the goal of the algorithm is to predict the price of a new house going to market

House Price Prediction

Square footage	Num of Rooms	Garden?	Parking Facility?	Num of floors	Price ($) in 1000's
..	3	Yes	..	2	460
1700	..	No	No	1	320
..	5	..	..	..	..
..	..	..	..	..	..
..	..	..	..	..	..
..	..	..	..	..	..

Labeled Exampl

Attributes/Features/Independent Variables ($\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_n$)

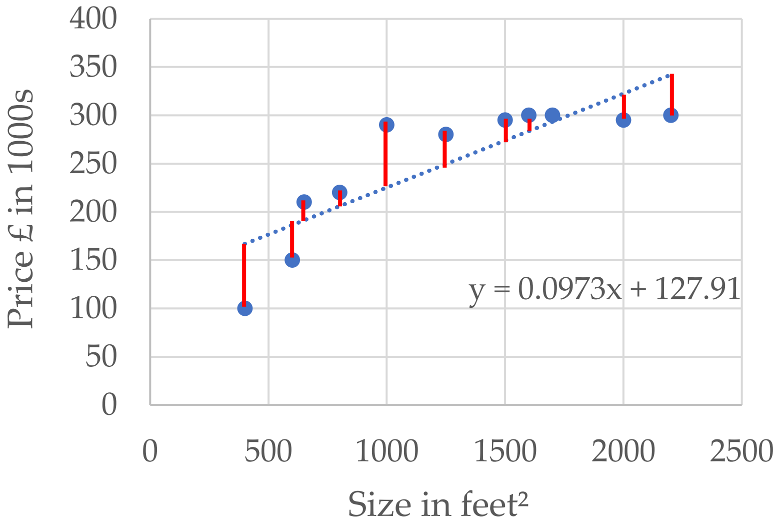
Target/Dependent Variable (y)

Simple Linear Regression

Consider the problem of **predicting house prices (y)**

- **Feature Selection** – input variables that can used to predict house prices
  – let's consider one input variable (size in sq.ft) → univariate/simple regression

- Simple linear regression finds a linear function (straight line) that predicts the target variable (*y*) as a function of the features or independent variables (*x*)
- $y = mx + c$ where m is the slope and c is the intercept



Features/independent variables (x)	Target/dependent variable (y)
Size in feet²	Price £ in 1000s
400	100
600	150
650	210
800	220
1000	290
1250	280
1500	295
1600	300
1700	300
2000	295
2200	300

[Supervised Learning in Machine Learning: Regression and Classification (DeepLearning.AI) | Coursera]

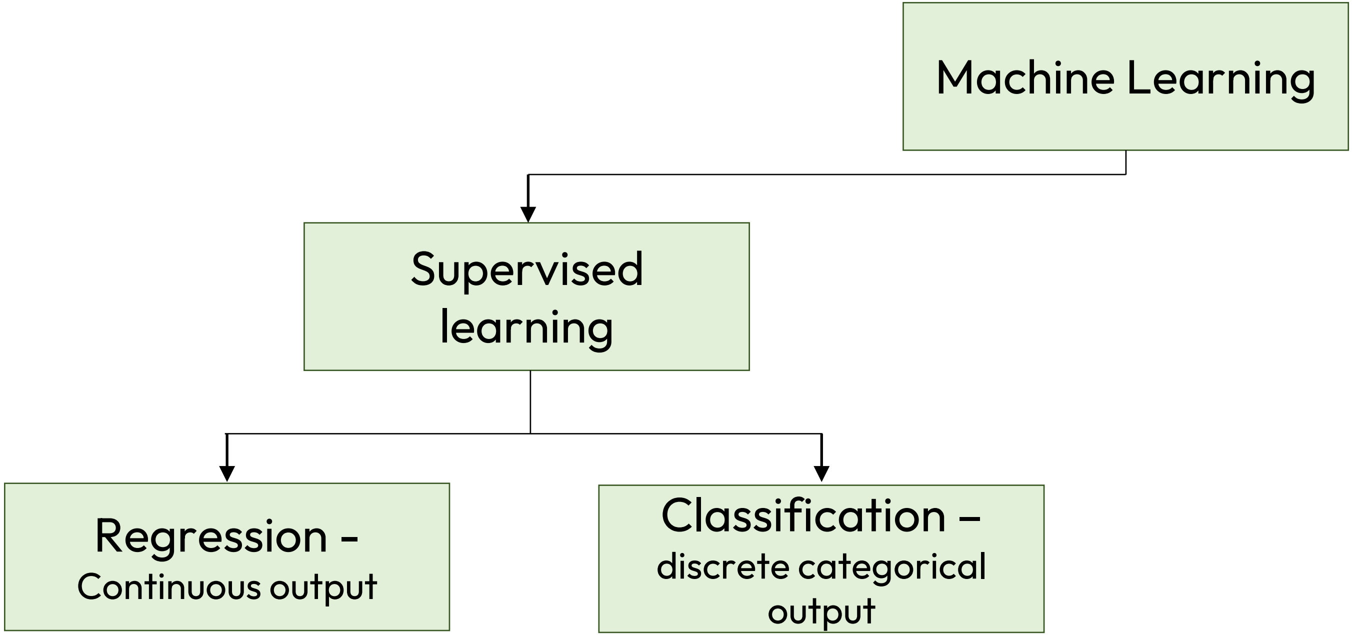
Metrics of Evaluation: Regression Models

- Mean Absolute Error (MAE)
- Mean Squared Error (MSE)
- Mean Absolute Percentage Error (MAPE)
- Root Mean Squared Error (RMSE)
- R-squared (R2) represents the proportion of variance of the target variable (y) that has been explained by the feature variables (x1,x2…). It indicates the goodness of fit of the model.
- Adjusted R-squared adjusts for the number of variables being considered. If you add more and more irrelevant predictor variables, the adjusted R2 will reduce. If you add more relevant variables, adjusted R2 will increase.

House Price Prediction: Metric Scores

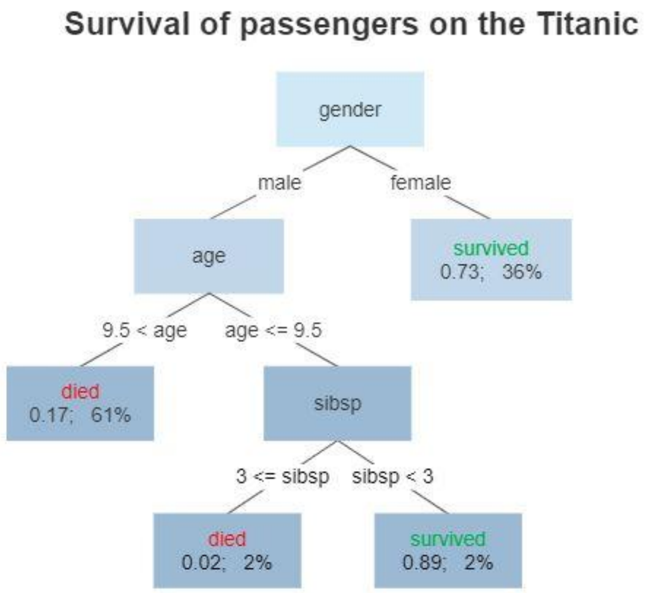
	Linear Regression	Decision Trees	SVM	Random Forest
MAE	3.57487	2.99606	5.59430	**2.27155**
MSE	21.89777	16.65047	80.95313	**8.85840**
RMSE	4.67950	4.08050	8.99740	**2.97631**
MAPE	0.17466	0.14898	0.23362	**0.11844**
R2	0.77894	0.83191	0.18277	**0.91057**
A-R2	0.75351	0.81258	0.08876	**0.90029**

Types of supervised learning



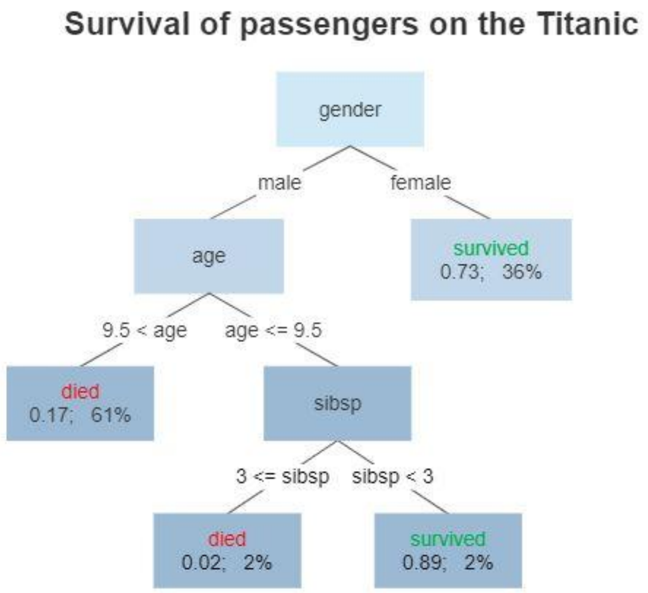
Popular Algorithms

- Logistic regression
- Naïve Bayes
- Decision Trees
- Support Vector Machines (SVM)
- Random Forest
- KNN (K Nearest Neighbour)



Metrics of Evaluation: Classification Models

- Confusion Matrix
- Accuracy and Error Rate
- Precision
- Sensitivity (recall)
- Specificity
- AUC – ROC curve



Metrics of Evaluation: Classification Models

Is it a ROSE or NOT A ROSE?

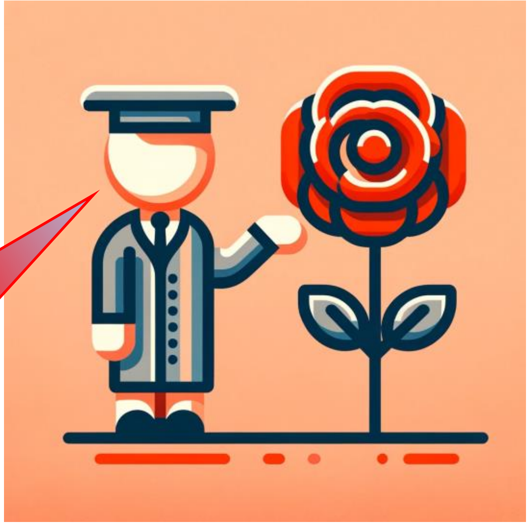
1 = Rose
0 = Not a Rose

Metrics of Evaluation: Type 1 and 2 error

1=Rose 0=Not a Rose

It's a rose!

It's not a rose!



Type I Error

Type II Error

Confusion Matrix

- A matrix that tells us where a model gets 'confused'
- It is a table (or matrix) that shows the mapping between the actual (true) classification of a data point, to the classification as predicted by the model.
- Ideally, all of the 'TRUE' entries would be along the diagonal of the matrix. This would mean that all of the predictions were correct.
- The 'off-diagonal' entries indicate some sort of error.

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

Confusion Matrix

- **True Positive (TP):** The model predicted it to be a rose, and it is a rose (predicted TRUE, and it is TRUE)
- **True Negative (TN):** The model predicted it to not be a rose, and it is not a rose (predicted FALSE, and it is FALSE)
- **False Positive (FP):** The model predicted it to be a rose, but it is not a rose (predicted TRUE, but it is FALSE) **Type I Error**
- **False Negative (FN):** The model predicted it to not be a rose, but it is a rose (predicted FALSE, but it is TRUE) **Type II Error**

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

```
> rg

Call:
 randomForest(formula = survived ~ pclass + sex + age + sibsp +      parch + fare, data = train_titanic)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 20%
Confusion matrix:
    0   1 class.error
0 405  36  0.08163265
1 112 187  0.37458194
> |
```

Predicted Class

Actual Class		Positive	Negative
	Positive	TP	FN
	Negative	FP	TN

```
> rg$confusion[,1:2] %>%
+   as.table %>%
+   confusionMatrix()
Confusion Matrix and Statistics

    0   1
0 405  36
1 112 187

               Accuracy : 0.8
                 95% CI : (0.7693, 0.8283)
    No Information Rate : 0.6986
    P-Value [Acc > NIR] : 2.852e-10

                  Kappa : 0.567

 Mcnemar's Test P-Value : 7.050e-10

            Sensitivity : 0.7834
            Specificity : 0.8386
         Pos Pred Value : 0.9184
         Neg Pred Value : 0.6254
             Prevalence : 0.6986
         Detection Rate : 0.5473
   Detection Prevalence : 0.5959
      Balanced Accuracy : 0.8110

       'Positive' Class : 0
```

Predicted Class

Actual Class	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Accuracy

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

$$Error\ Rate = 1 - accuracy = \frac{FP + FN}{TP + FP + TN + FN}$$

Accuracy: from all the classes (positive and negative), how many has the model predicted correctly?

Accuracy is limited when there is an imbalance in one class. A model where 95% of the dataset are instances of one class could achieve 95% accuracy simply by also predicting the majority class.

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

University of Exeter

Precision

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

Precision: from all the classes that the model predicted to be positive, how many are actually positive.

The proportion of positive results that were correctly classified.

Not the same as true positive rate (sensitivity)

Useful when FP as a more significant concern that FN. For example, if classifying a test result as ‘cancer’ or ‘not cancer’ – not affected by the number of TNs

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

Sensitivity

$$\textit{Sensitivity (recall)} = \frac{TP}{TP + FN}$$

Sensitivity: from all the positive classes, how many has the model predicted correctly.

True Positive Rate (TPR)

Sensitivity and specificity are inversely proportional, meaning that as sensitivity increases, the specificity decreases and vice versa.

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

Specificity

$$Specificity = \frac{TN}{TN + FP}$$

Sensitivity: how often a model can correctly predict the negative outcomes.

True Negative Rate (TNR)
False Positive Rate (FPR) = 1 - Specificity

Sensitivity and specificity are inversely proportional, meaning that as sensitivity increases, the specificity decreases and vice versa.

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

Why not just accuracy?

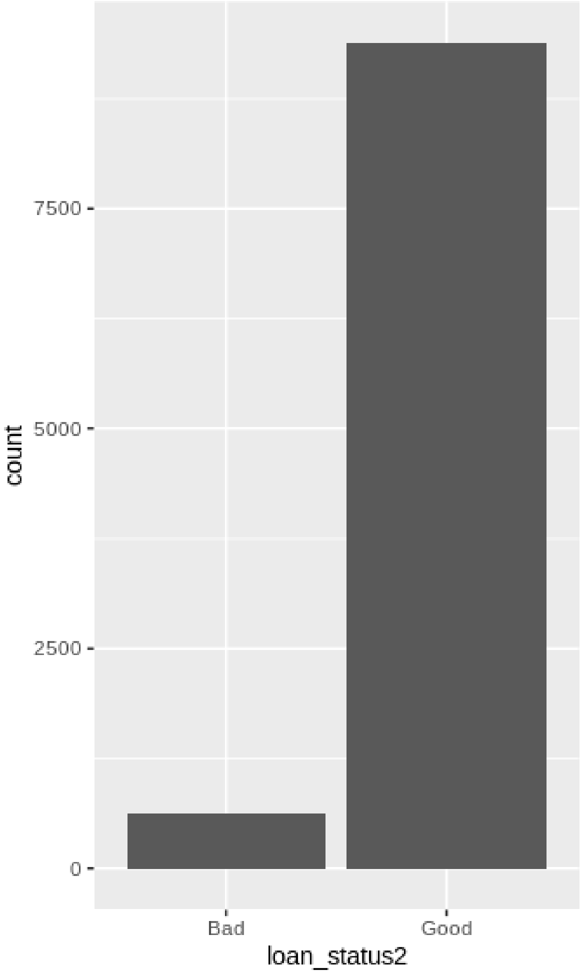
Good loan = 9382 *(predicted good)*
Bad loan = 618 *(predicted good)*

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

$$= \frac{9382+0}{9382+618+0+0}$$

$= 0.9382$ *(93.8%)*

		Predicted Class	
		Good	Bad
Actual Class	Good	9382	0
	Bad	618	0



Why not just accuracy?

Good loan (0) = 9382 (predicted not bad (0))
Bad loan (1) = 618 (predicted not bad) (0))

$$Specificity = \frac{TN}{TN + FP} = \frac{9382}{9382+0}$$

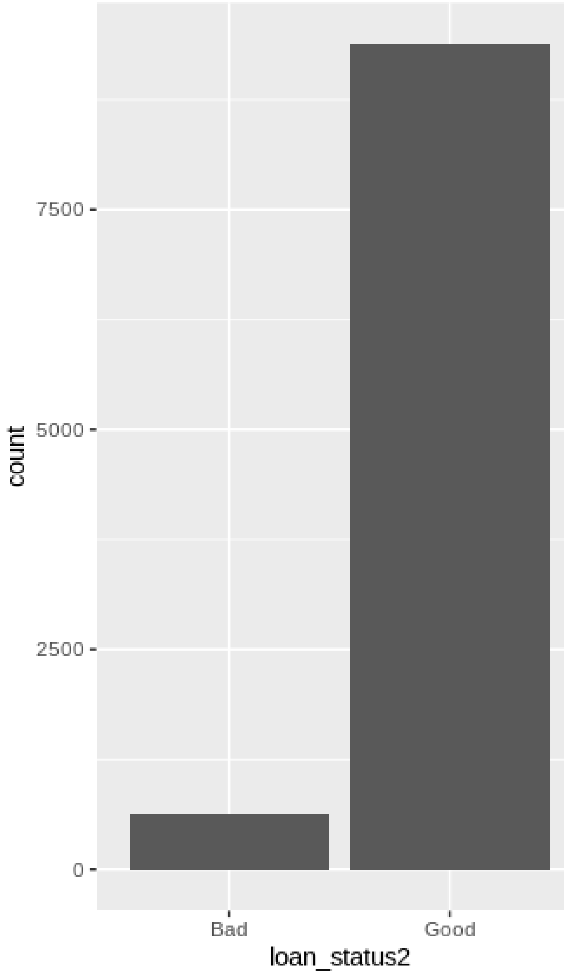
$= 1\ TNR$

$$Sensitivity = \frac{TP}{TP + FN} = \frac{0}{618+0}$$

$= 0\ TPR$

Predicted Class

Actual Class		Bad 1	Not Bad 0
	Bad 1	0 (TP)	618 (FN)
	Not Bad 0	0 (FP)	9382 (TN)



Why not just accuracy?

At 50% accuracy level!

$$Specificity = \frac{TN}{TN + FP} = \frac{17}{17+2}$$

$= 0.895$

$$Sensitivity = \frac{TP}{TP + FN} = \frac{5}{5+3}$$

$= 0.625$

Actual Class \ Predicted Class	Rose 1	Not Rose 0
Rose 1	5 (TP)	3 (FN)
Not Rose 0	2 (FP)	17 (TN)

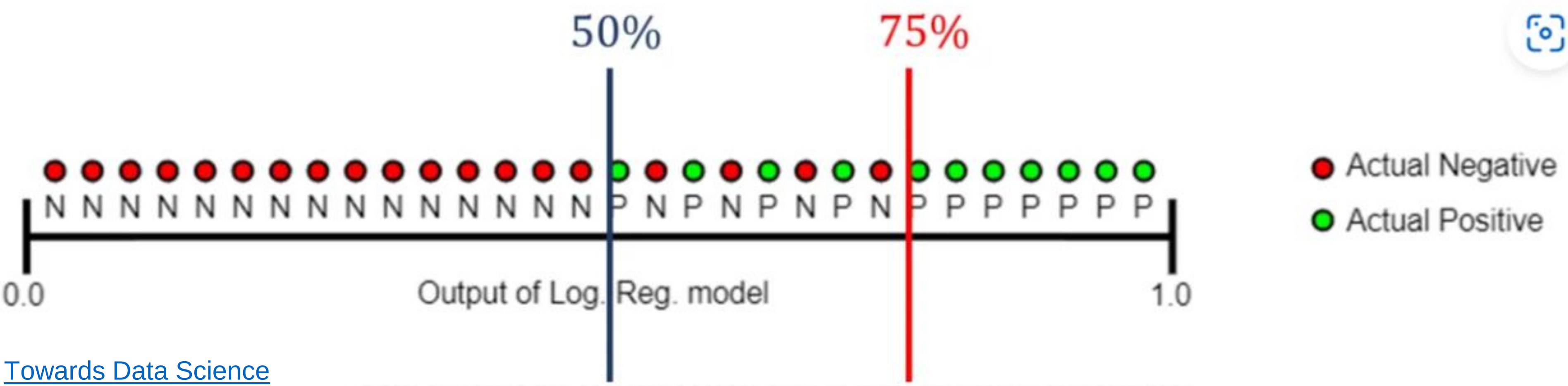
ROC curve

$$TPR = \text{Sensitivity (recall)} = \frac{TP}{TP + FN}$$

$$FPR = (1 - \text{Specificity}) = \frac{TN}{TN + FP}$$

ROC curves show the behaviour of the classifier for every threshold, e.g. :

- 50% threshold: FN=0, TP=11, FP=4, TN=15
- 75% threshold: FN=4, TP=7, FP=0, TN=19



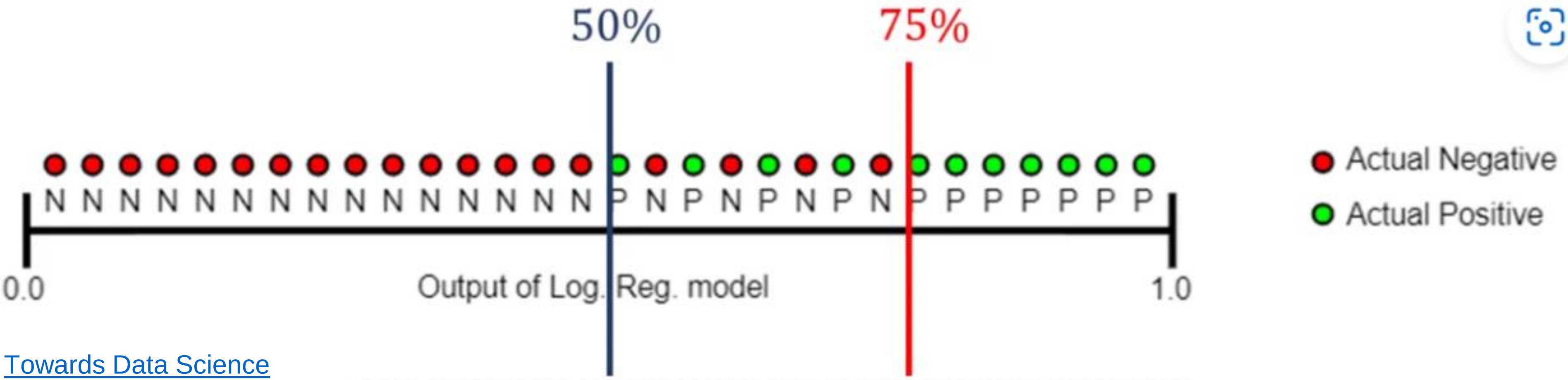
Towards Data Science

ROC curve

$$TPR = Sensitivity\ (recall) = \frac{TP}{TP + FN}$$

$$FPR = (1 - Specificity) = \frac{TN}{TN + FP}$$

	50%	75%
TPR	1.0	0.64
FPR	0.21	0.0

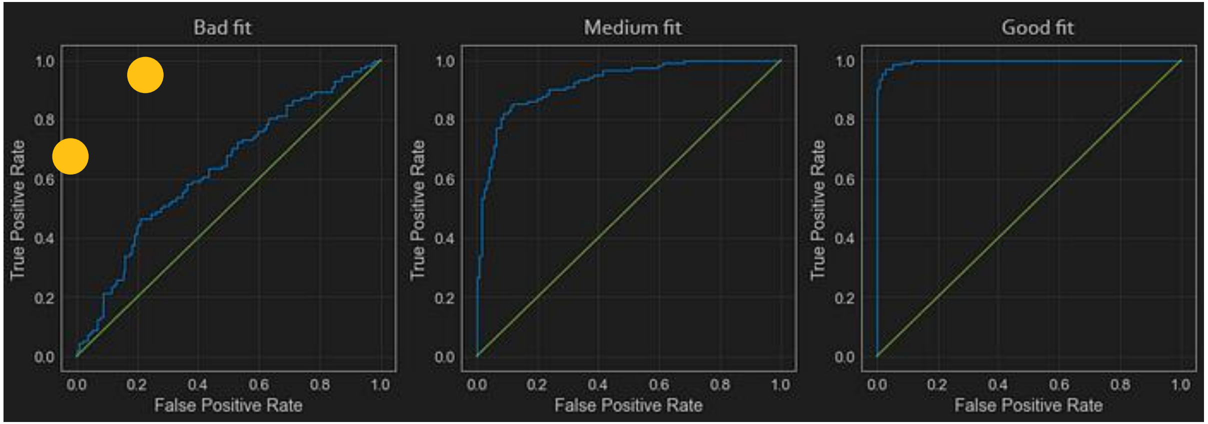


[Towards Data Science]

ROC curve

$$TPR = Sensitivity\ (recall) = \frac{TP}{TP + FN}$$

$$FPR = (1 - Specificity) = \frac{TN}{TN + FP}$$

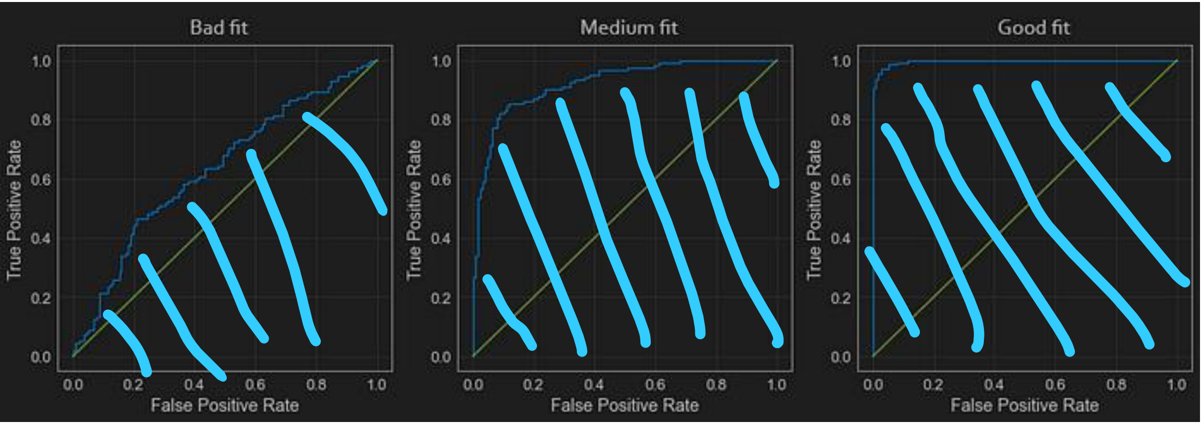


- ROC – Receiver Operating Characteristic Curve
- Plot between TPR (y-axis) and FPR (x-axis) at different classification thresholds
- Lowering the classification threshold means:
  - More data points classified as positive (more FPs and high TPs)
  - Increased sensitivity and reduced specificity
- Vice versa for increasing the classification threshold
- Best results (a better model) is high TPR, low FPR (top right corner)

[Towards Data Science]()

AUC (Area Under the Curve)

- AUC – used as a summary of the ROC curve
- Measures the 2D area under the ROC
- Represents the degree or measure of separability for the predicted classes (ie how good a model is at distinguishing between classes)



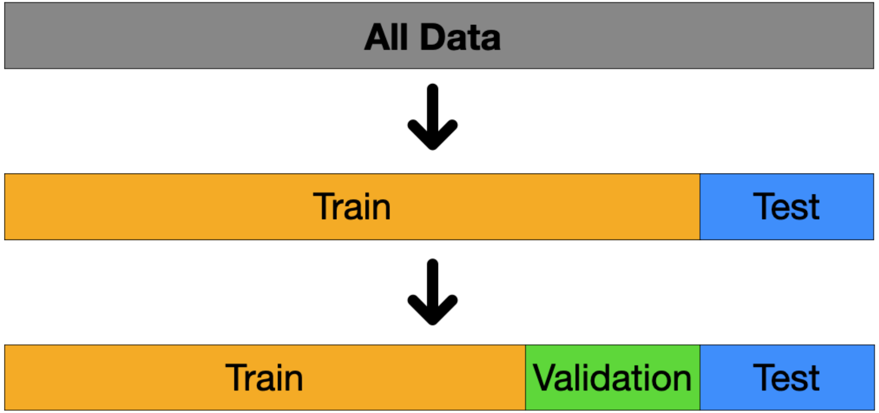
- A higher AUC is a better classification model
- AUC equals 0.5 – the model can't distinguish between positive and negative classes – it is totally random.
- A perfect prediction has an AUC = 1.0

Evaluating the model:

Partition your data set: the first step in developing a model is training and validation.

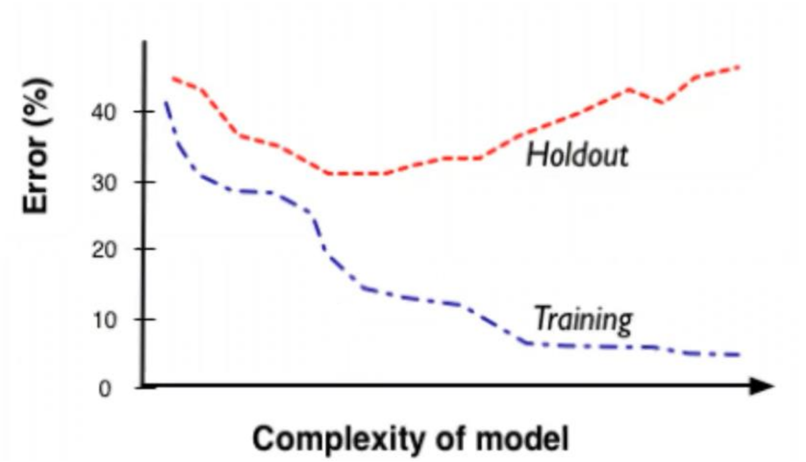
You must first partition your dataset, which involves choosing what percentage of your data to use for the training and testing sets.

- **Training set:** is the subsection of a dataset from which the model uncovers, or “learns,” relationships between the different variables (x1, x2, .., x3) and the response variable (y). Used to fit the model.
- **Validation set:** Often used to fine-tune model parameters before testing
- **Test set:** (also sometimes referred as holdout subset) provides an unbiased estimate of the model’s performance after it has been trained and validated. Used for comparing / selecting between models.



Evaluating the model:

- **Training set:** Used to fit the model.
- **Test set:** We know the value of the target variable, but it was not used to build the model.
- A **fitting graph** shows the accuracy of a model as function of complexity.



Bias / Variance Tradeoff

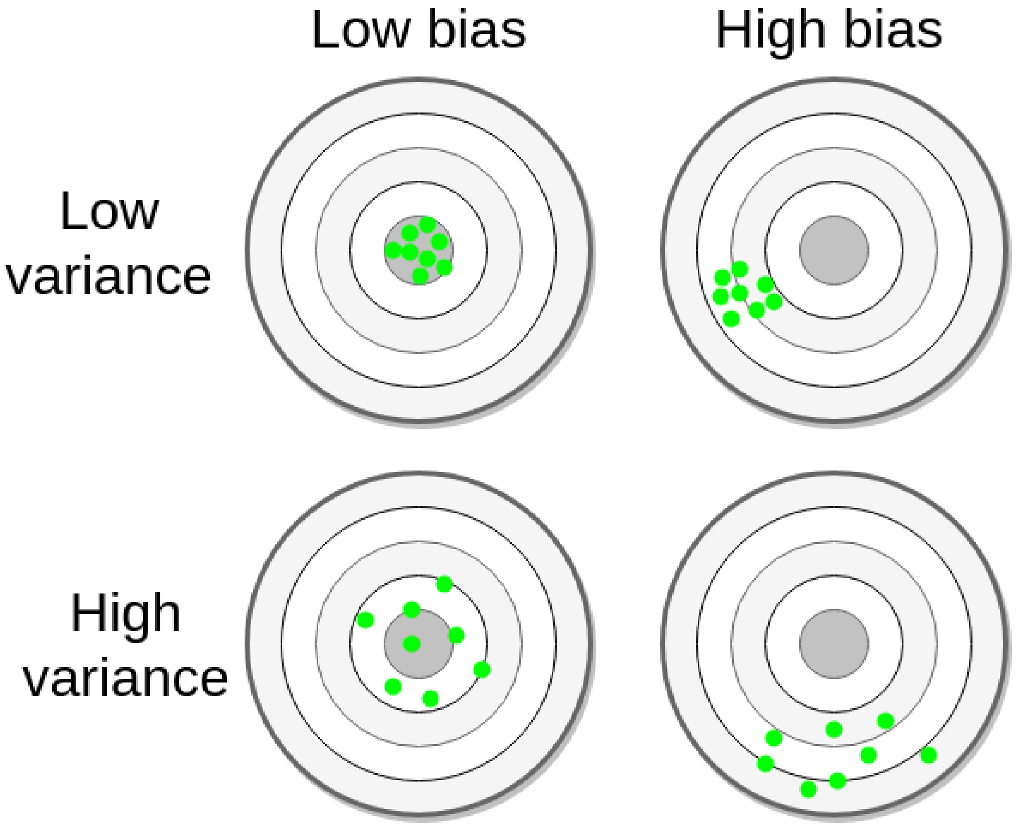
Bias refers to the error introduced by approximating a real-world problem, which may be complex, by a much simpler model.

The **variance** refers to the error introduced by sensitivity to fluctuations in the training set.

The bias-variance tradeoff: The goal is to achieve a balance between these two types of errors.

A highly complex model (low bias) that is tuned too much to the training data can capture the noise and lead to poor generalization (high variance).

Conversely, a model that is too simple (high bias) will not capture the complexity of the data and will perform poorly even on the training data.



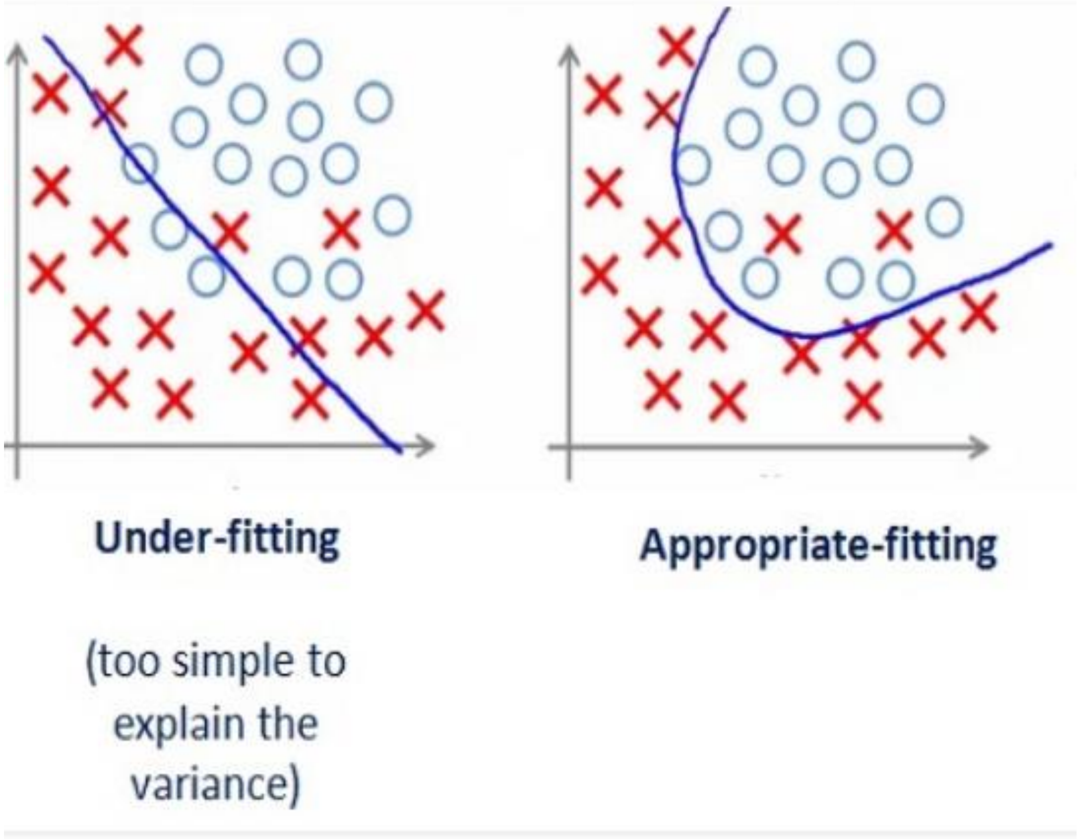
[Understanding the Bias-Variance Tradeoff (fortmann-roe.com)]

Problems faced: Underfitting

Underfitting Model is not complex enough to capture the underlying patterns in the data.

Leads to high bias.

If a model has high bias, simplifying assumptions mean that the model does not have the capacity to capture important regularities, and it tends to underperform.



Problems faced: Overfitting

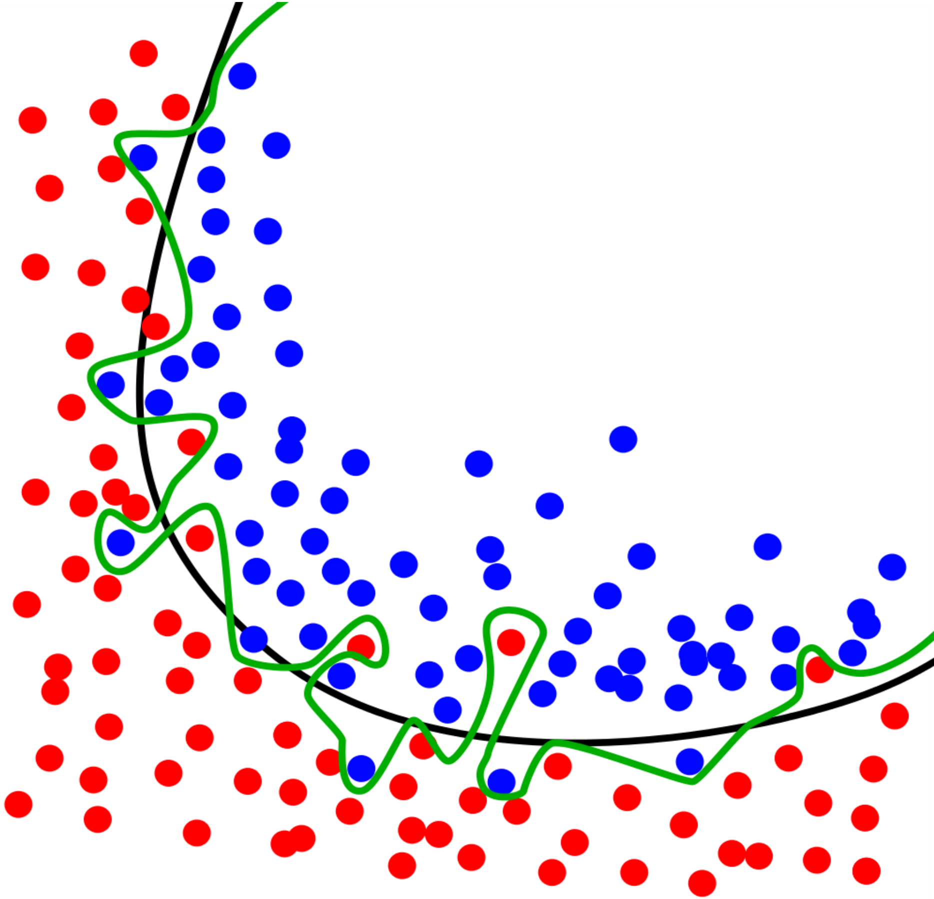
Overfitting Learning a function that perfectly explains the training data that the model learned from, but doesn't generalize well to unseen data (the test set).

The model overlearns from the training data to the point that it starts picking up idiosyncrasies that aren't representative of patterns in the real world.

Generalisation is the property of a model or modelling process to apply to data that were not used to build the model.

Lead to high variance.

Variance: How much your model's test error changes compared with training data.



Cross-validation

University of Exeter

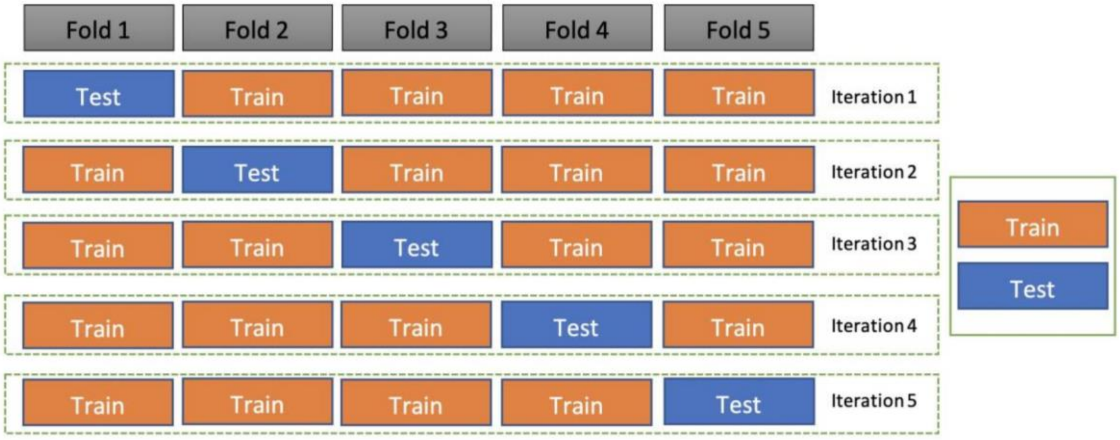
K-fold cross validation

Five-fold cross validation splits the original dataset into five equal sized pieces.

Each piece in turn is used as a test set while the other four are used to train the model.

The result is five different accuracy results, which can be used to compute the average accuracy and its variance.

Machine Learning Fundamentals: Cross Validation (youtube.com)



Cross-validation

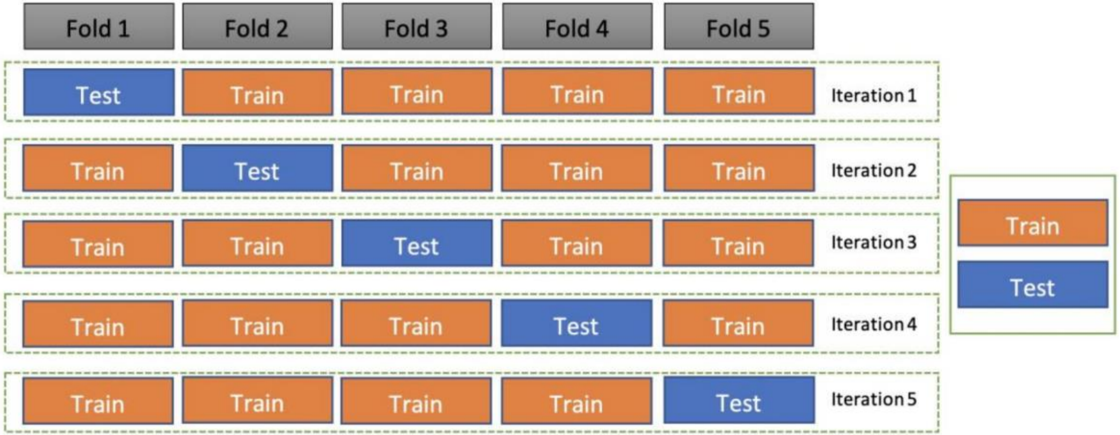
One way of dealing with overfitting.

A resampling procedure to evaluate a ML model.

k-fold CV: where k refers to number of groups a given data set should be split into.

Procedure:

1. Shuffle the dataset randomly
2. Split the dataset into k groups
3. For each unique group:
   a) Take the group as a hold out or test data set
   b) Take the remaining groups as a training data set
   c) Fit a model on the training set and evaluate on the test set
   d) Retain the evaluation score and discard the model.
4. Summarise the skill of the model using the sample of model evaluation scores



Next Week: Reading Week – For week 8:

- Textbook Ch. 10
- For reference, Text Mining for R
- Listen: Text Mining in R

Any questions?

?