IMPERIAL

Machine Learning for Neuroscience

ML4NS

Ensemble Models and Kernel-based Methods

Payam Barnaghi
Department of Brain Sciences &
School of Convergence Science in Human and Artificial Intelligence
Imperial College London

January 2025

1

1

Feature Vectors

IMPERIAL

– So far, we have assumed that each item we wish to classify, cluster, or process can be represented as a fixed-size feature vector.
– However, for certain types of items/concepts, it is not clear how to best represent them as fixed-sized feature vectors.
– For example, how do we represent a text document or protein sequence, which can be of variable length? or a molecular structure, which has complex 3d geometry? or an evolutionary tree, which has variable size and shape?

2

2

Kernel functions

IMPERIAL

- One approach to the latter problem is to assume that we have some way of measuring the similarity between items that don't require pre-processing them into a feature vector format.
- For example, we can apply a function to transform the original data into a higher dimensional space in which the data items can then be compared/processed.
- This process can be done via kernel functions.

3

3

Linear separability

IMPERIAL

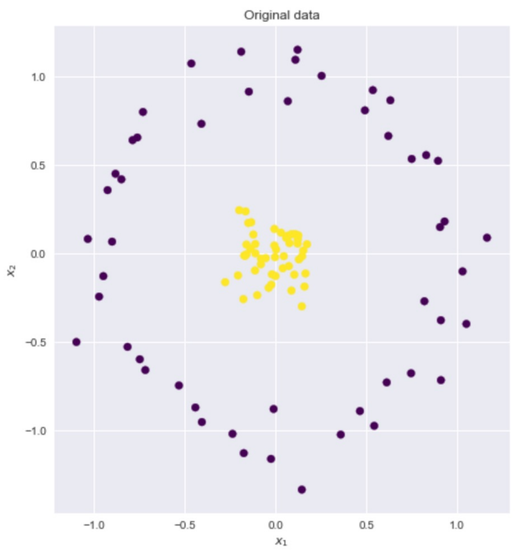
- We have also often assumed that the data items are linearly separable, but what if the data was not linearly separable?
- In this case, we can again apply a kernel that maps each data instance (from the original non-linear observation space) into a higher-dimensional space in which, in this new space, the data instances become separable.

4

4

Example

IMPERIAL



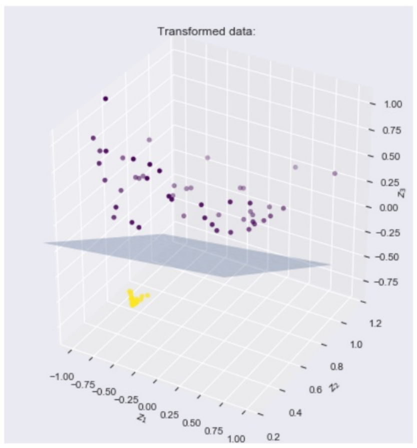
Source: https://xavierbourretsicotte.github.io/Kernel_feature_map.html

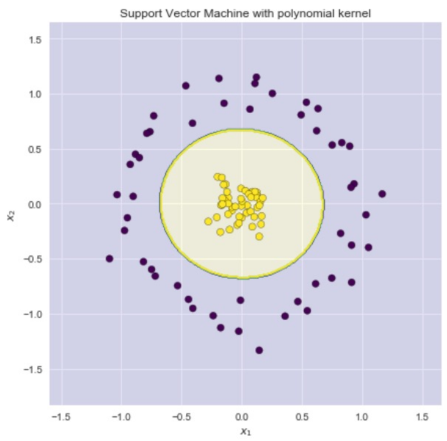
5

5

Example – applying polynomial kernel

IMPERIAL





Source: https://xavierbourretsicotte.github.io/Kernel_feature_map.html

6

6

What is a polynomial kernel?

IMPERIAL

- If our data is a set of (x_1, x_2)
- Transform it to (z_1,z_2,z_3) in which:

  $z_1 = x_1$

  $z_2 = x_2$

  $z_3 = x_1^2 + x_2^2$

  $(z1,z2,z3) = (x_1, x_2, x_1^2 + x_2^2)$

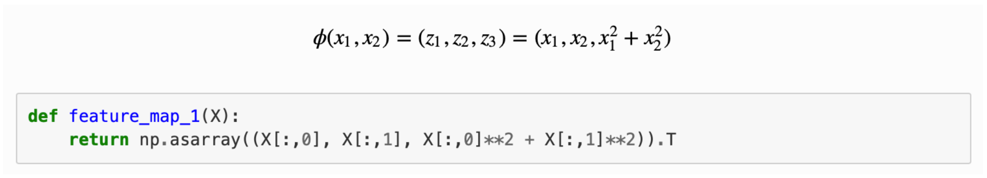
- This is in fact sum of polynomials

7

7

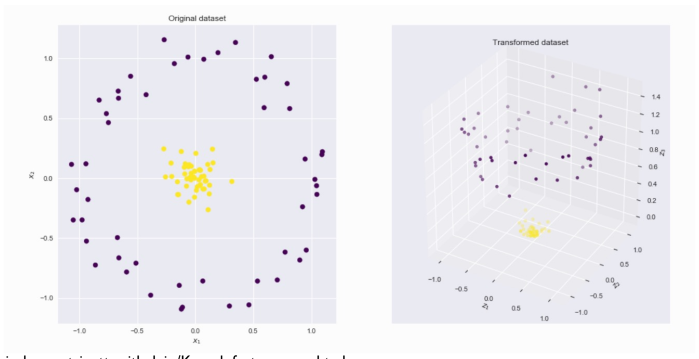
A polynomial kernel

IMPERIAL



$$\phi(x_1, x_2) = (z_1, z_2, z_3) = (x_1, x_2, x_1^2 + x_2^2)$$

```python
def feature_map_1(X):
    return np.asarray((X[:,0], X[:,1], X[:,0]**2 + X[:,1]**2)).T
```



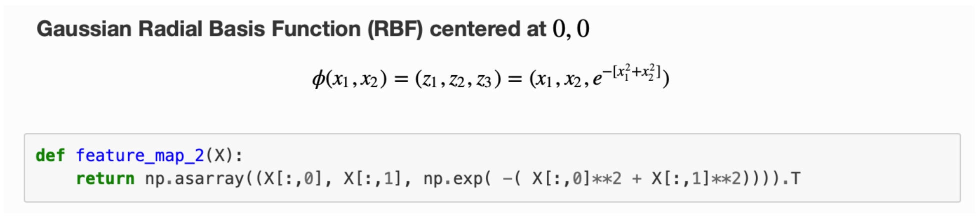
Source: https://xavierbourretsicotte.github.io/Kernel_feature_map.html

8

8

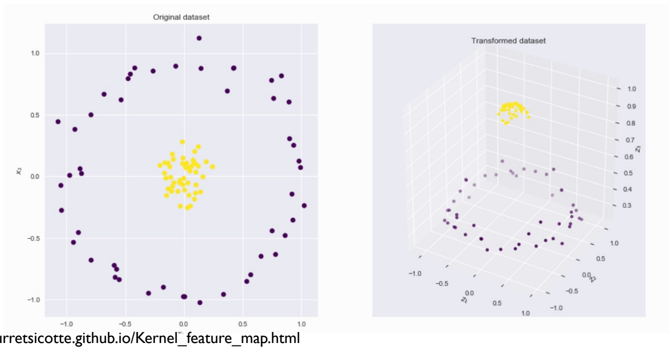
Gaussian Radial Basis Function (RBF)

IMPERIAL


Gaussian Radial Basis Function (RBF) centered at $0, 0$

$$\phi(x_1, x_2) = (z_1, z_2, z_3) = (x_1, x_2, e^{-[x_1^2 + x_2^2]})$$

```python
def feature_map_2(X):
    return np.asarray((X[:,0], X[:,1], np.exp( -( X[:,0]**2 + X[:,1]**2)))).T
```



Source: https://xavierbourretsicotte.github.io/Kernel_feature_map.html

RBF Kernel in more generalised form*

IMPERIAL

The **squared exponential kernel** (SE kernel) or **Gaussian kernel** is defined by

$$\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^T \mathbf{\Sigma}^{-1}(\mathbf{x} - \mathbf{x}')\right)$$

If $\mathbf{\Sigma}$ is diagonal, this can be written as

$$\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}\sum_{j=1}^{D}\frac{1}{\sigma_j^2}(x_j - x_j')^2\right)$$

Source: Kevin Murphy, Machine Learning: A probabilistic Approach, MIT Press.

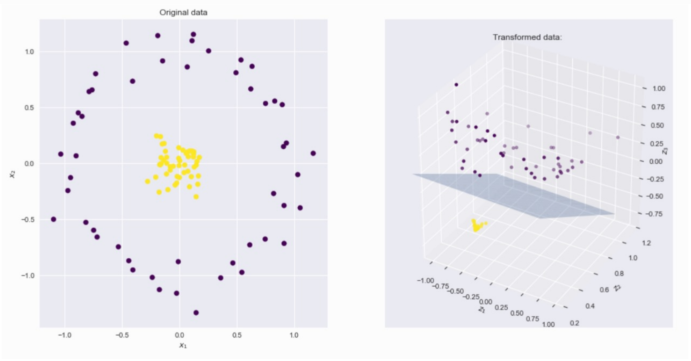
Another polynomial function

IMPERIAL

$$x_1, x_2 :\rightarrow z_1, z_2, z_3$$

$$z_1 = \sqrt{2}x_1x_2 \quad z_2 = x_1^2 \quad z_3 = x_2^2$$

```
def feature_map_3(X):
    return np.asarray(( np.sqrt(2) *X[:,0] * X[:,1], X[:,0]**2, X[:,1]**2)).T
```



Source: https://xavierbourretsicotte.github.io/Kernel_feature_map.html

11

Support Vector Machines

IMPERIAL

- Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outlier detection.
- The advantages of support vector machines are:
  - Effective in high-dimensional spaces.
  - It could still be effective in cases where the number of dimensions is greater than the number of samples.
  - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
  - Versatile: Different kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to define custom kernels.

Source: Scikit learn

12

SVM - disadvantages

IMPERIAL

- The disadvantages of support vector machines include:
  - If the number of features exceeds the number of samples, the method will likely give poor performances.
  - SVMs do not directly provide probability estimates.

13

13

SVM Classifier

IMPERIAL

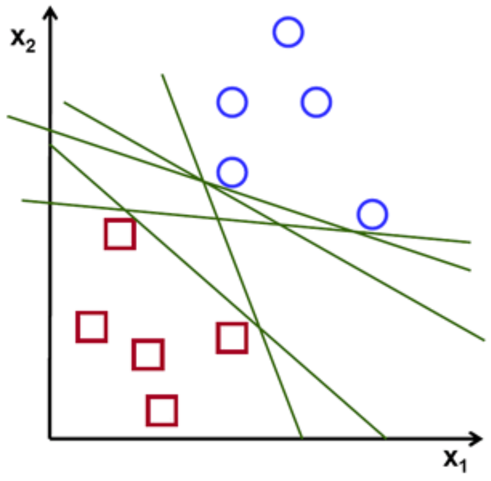
- A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane.
- In other words, given labelled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorises new examples.
- e.g., For a linearly separable set of 2D-points which belong to one of two classes, find a separating straight line.

14

14

SVM Example

IMPERIAL



Source: https://vovkos.github.io/doxyrest-showcase/opencv/sphinx_rtd_theme/page_tutorial_introduction_to_svm.html

Linear separation

IMPERIAL

- In this example, we deal with lines and points in the Cartesian plane instead of hyperplanes and vectors in a high-dimensional space.
- This is a simplification of the problem.
- It is important to understand that this is done only because our intuition is better built from easily imagined examples.
- However, the same concepts apply to tasks where the examples to classify lie in a space whose dimension is higher than two.

Source: https://vovkos.github.io/doxyrest-showcase/opencv/sphinx_rtd_theme/page_tutorial_introduction_to_svm.html

SVM – linear separation

IMPERIAL

- We can intuitively define a criterion to estimate the worth of the lines:
  - A line is bad if it passes too close to the points because it will be noise-sensitive and will not generalise correctly. Therefore, our goal should be to find the line passing as far as possible from all points.
  - Then, the operation of the SVM algorithm is based on finding the hyperplane that gives the largest minimum distance to the training examples.
  - This distance receives the important name of margin within SVM's theory. Therefore, the optimal separating hyperplane maximises the margin of the training data.

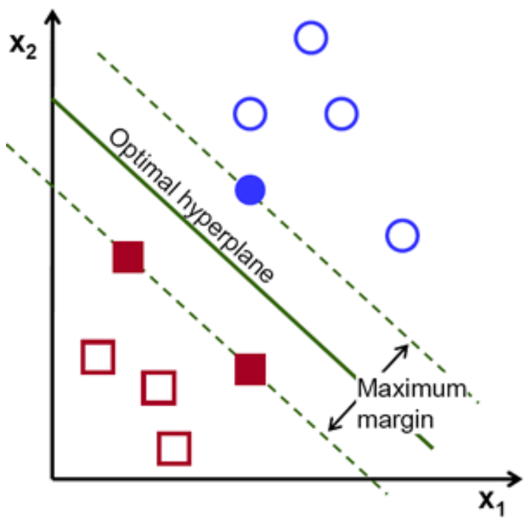
Source: https://vovkos.github.io/doxyrest-showcase/opencv/sphinx_rtd_theme/page_tutorial_introduction_to_svm.html

17

17

Example: SVM for a linearly separable set of 2d-points

IMPERIAL



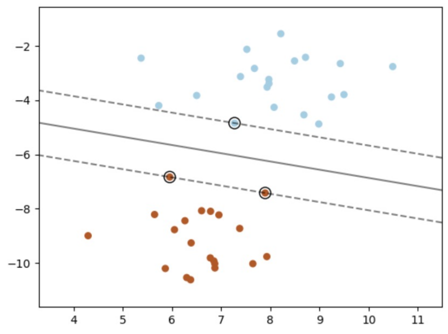
Source: https://vovkos.github.io/doxyrest-showcase/opencv/sphinx_rtd_theme/page_tutorial_introduction_to_svm.html

18

18

IMPERIAL

Mathematical formulation*



$$\sum_{i \in SV} y_i \alpha_i K(x_i, x) + b$$

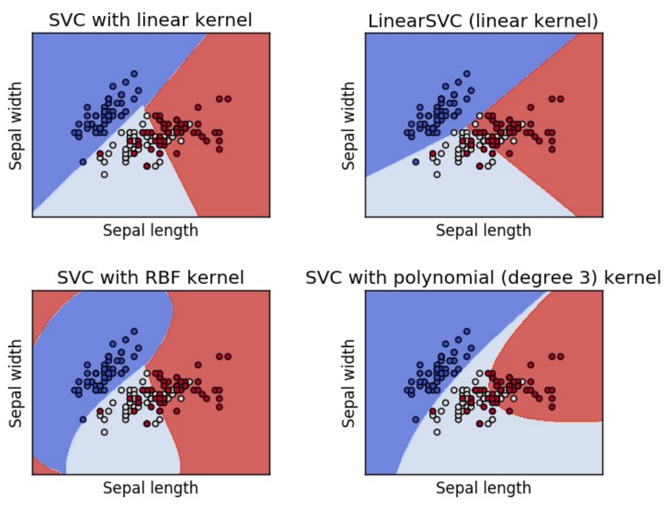
Source: https://scikit-learn.org/stable/modules/svm.html

19

19

IMPERIAL

SVM with different kernels



Source: https://vovkos.github.io/doxyrest-showcase/opencv/sphinx_rtd_theme/page_tutorial_introduction_to_svm.html

20

20

Decision Trees

IMPERIAL

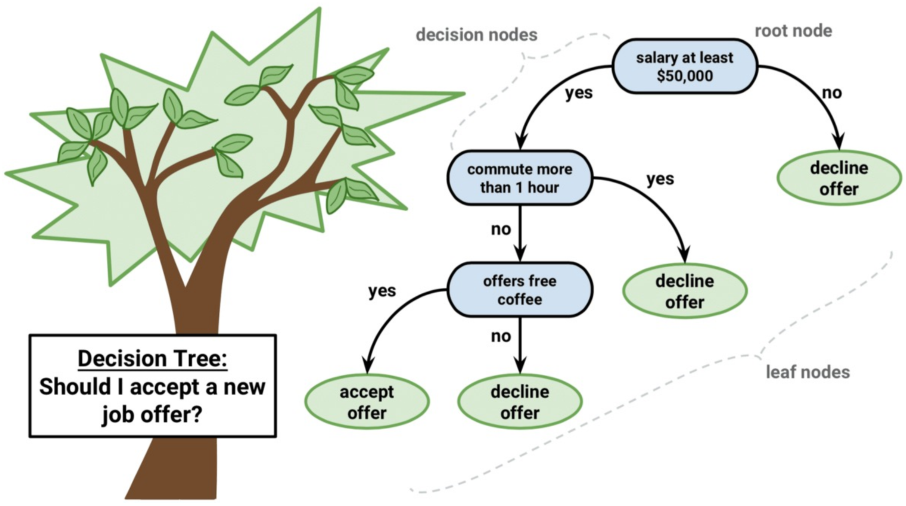
- A decision tree model is an interpretable model in which the final output is based on a series of comparisons of the values of predictors against threshold values.
- DTs are made up of nodes, branches, and leaves.
- Each node represents a feature, each branch a choice, and each leaf an outcome.
- DTs take a top-down approach to data, attempting to group and classify similar observations and searching for the optimal criteria to partition the dissimilar observations until they reach a particular level of similarity.

21

21

Decision Trees - Example

IMPERIAL



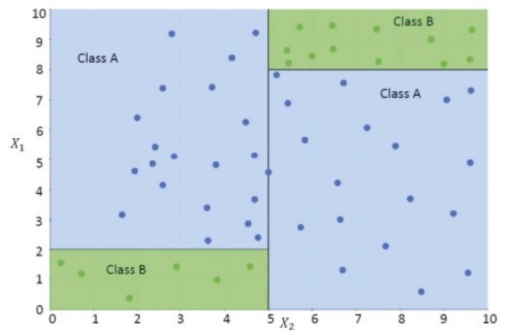
Source: https://regenerativetoday.com/simple-explanation-on-how-decision-tree-algorithm-makes-decisions/

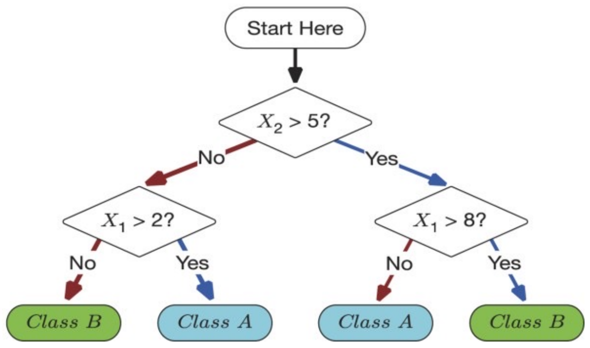
22

22

Decision Making Using DTs

IMPERIAL

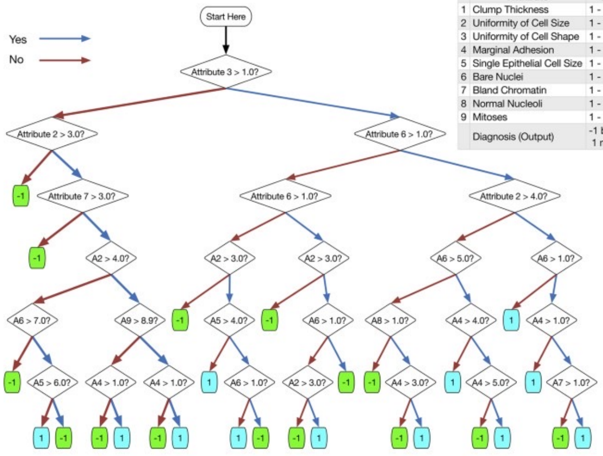




Source: Valdes, G., Luna, J., Eaton, E. *et al.* MediBoost: a Patient Stratification Tool for Interpretable Decision Making in the Era of Precision Medicine. *Sci Rep* 6, 37854 (2016). https://doi.org/10.1038/srep37854

DTs in Clinical Applications

IMPERIAL



	Attribute	Value
1	Clump Thickness	1 - 10
2	Uniformity of Cell Size	1 - 10
3	Uniformity of Cell Shape	1 - 10
4	Marginal Adhesion	1 - 10
5	Single Epithelial Cell Size	1 - 10
6	Bare Nuclei	1 - 10
7	Bland Chromatin	1 - 10
8	Normal Nucleoli	1 - 10
9	Mitoses	1 - 10
	Diagnosis (Output)	-1 benign 1 malign

Example Interpretable Rules Induced by MediBoost:

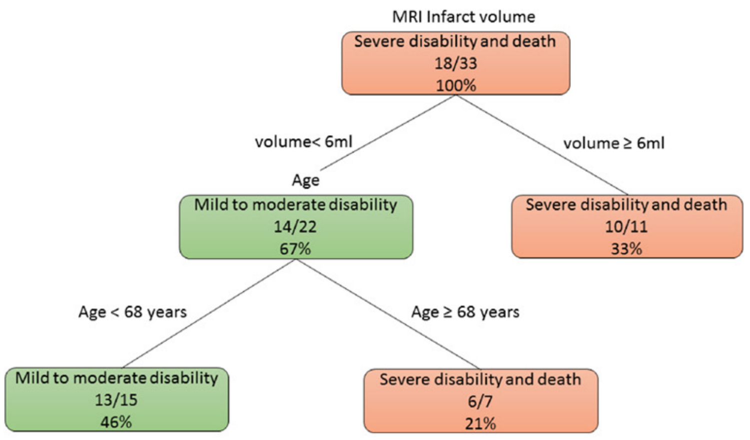
A3 Uniformity of Cell Shape ≤ 1.0 $\wedge$ A2 Uniformity of Cell Size > 3.0 $\wedge$ A7 Bland Chromatin ≤ 3.0 $\Rightarrow$ predict benign

A3 Uniformity of Cell Shape > 1.0 $\wedge$ A6 Bare Nuclei ≤ 1.0 $\wedge$ A2 Uniformity of Cell Size ≤ 3.0 $\Rightarrow$ predict benign

Source: Valdes, G., Luna, J., Eaton, E. *et al.* MediBoost: a Patient Stratification Tool for Interpretable Decision Making in the Era of Precision Medicine. *Sci Rep* 6, 37854 (2016). https://doi.org/10.1038/srep37854

A decision tree model in TBI

IMPERIAL


MRI Infarct volume
Severe disability and death
18/33
100%
volume< 6ml
volume ≥ 6ml
Age
Mild to moderate disability
14/22
67%
Severe disability and death
10/11
33%
Age < 68 years
Age ≥ 68 years
Mild to moderate disability
13/15
46%
Severe disability and death
6/7
21%

AUC=0.94 (95% CI 0.82 -1.00)

Source: Phan, T. G., Chen, J., Singhal, S., Ma, H., Clissold, B. B., Ly, J., & Beare, R. (2018). Exploratory Use of Decision Tree Analysis in Classification of Outcome in Hypoxic–Ischemic Brain Injury. *Frontiers in Neurology*, 9. https://doi.org/10.3389/fneur.2018.00126

25

25

Learning decision trees

IMPERIAL

- Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression.
- The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.
- A tree can be seen as a piecewise constant approximation.

Source: https://scikit-learn.org/stable/modules/tree.html

26

26

Learning decision trees

IMPERIAL

- There are different algorithms that can be used to learn decision tree structure from the data.
- In principle, these algorithms start with a root node in the tree structure and explore the features in the data and create rules that can separate the data one step at a time.
- One of the basic algorithms is ID3.

27

27

The ID3 algorithms

IMPERIAL

- Start with the dataset and set the root node.
- Start iterating through the dataset, and at each step, go through the features of the data and calculate the Entropy($\mathcal{H}$) and Information gain(IG) of the features.
- Select the feature that has the lowest entropy or highest information gain.
- Split the tree by the selected feature to create a new branch in the tree structure.
- Continue the algorithm by considering only the features that have never been selected before.

28

28

Entropy and Information Gain*

IMPERIAL

Consider a dataset with $S = \{9\text{ positive}, 5\text{ negative}\}$.

Step 1: Compute Entropy of S:

$$H(S) = -\left(\frac{9}{14}\log_2\frac{9}{14} + \frac{5}{14}\log_2\frac{5}{14}\right)$$

$$H(S) \approx 0.940$$

Step 2: Split on a Feature A:

Suppose splitting S based on A results in two subsets:

- $S_1 = \{6\text{ positive}, 2\text{ negative}\}$,
- $S_2 = \{3\text{ positive}, 3\text{ negative}\}$.

Calculate the weighted entropy after the split:

$$H(S|A) = \frac{8}{14}H(S_1) + \frac{6}{14}H(S_2)$$

$$H(S|A) = \frac{8}{14}(-\frac{6}{8}\log_2\frac{6}{8} - \frac{2}{8}\log_2\frac{2}{8}) + \frac{6}{14}(-\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6})$$

$$H(S|A) \approx 0.788$$

Step 3: Compute Information Gain:

$$IG(S,A) = H(S) - H(S|A)$$

$$IG(S,A) = 0.940 - 0.788 = 0.152$$

The feature A has an Information Gain of 0.152.

29

29

Decision Trees - Limitations

IMPERIAL

- Prone to overfit
- Poor generalisation performances
- High variance
  - a slight change in the data can result in a completely different set of splits, which can make interpretation difficult
- They can be inherently unstable
  - the effect of an error in the top splits propagates down to all the splits below due to their hierarchical nature
- Produce biased trees for an unbalanced dataset

30

30

How to overcome these limitations?

IMPERIAL

- Ensemble:
  - Single DT is fast but does not perform well
  - Learn from multiple trees

- We need to be careful not to learn the same tree over and over again.

- Bagging:
  - Bootstrap aggregating, a method that results in low variance
  - Construct N trees and learn a classifier for each bootstrap sample and average them
  - Can improve the accuracy of unstable models that tend to overfit
- Feature Randomness:
  - Feature bagging generates a random subset of features, which ensures low correlation among models

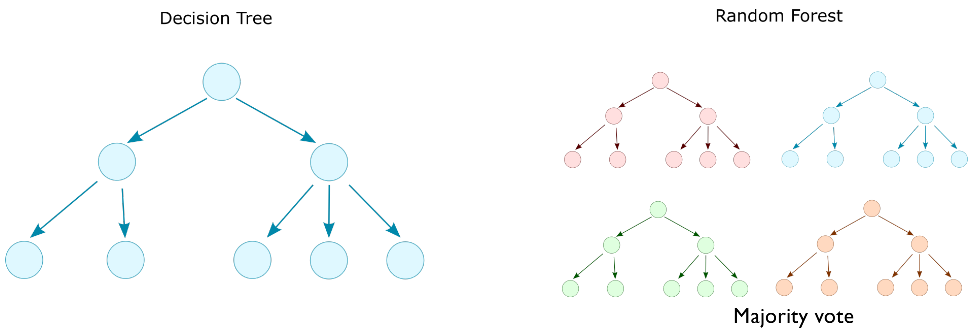
31

31

Random Forest

IMPERIAL

- Random Forest was introduced in 2001 to overcome the limitations of decision trees [1].
- Ensemble model consisting of multiple decision trees.



[1] Breiman, Leo. "Random forests." Machine learning 45.1 (2001): 5-32.

32

32

Ensemble models

IMPERIAL

- Ensemble methods combine the predictions of several base estimators built with a given learning algorithm to improve generalisability and robustness over a single estimator.

- Two families of ensemble methods are usually distinguished:
  - Averaging methods
  - Boosting methods

Source: https://scikit-learn.org/stable/modules/ensemble.html

33

33

Averaging methods

IMPERIAL

- In averaging methods, the driving principle is to build several estimators independently and then average their predictions.

- The combined estimator is usually better than any single base estimator because its variance is reduced.

- Examples: Bagging methods

Source: https://scikit-learn.org/stable/modules/ensemble.html

34

34

Bagging

IMPERIAL

- There are different Bagging methods but they mostly differ from each other by the way they draw random subsets of the training set:
  - When random subsets of the dataset are drawn as random subsets of the samples, then this algorithm is known as Pasting.
  - When samples are drawn with replacement, then the method is known as Bagging.
  - When random subsets of the dataset are drawn as random subsets of the features, then the method is known as Random Subspaces.
  - Finally, when base estimators are built on subsets of both samples and features, then the method is known as Random Patches.
- In scikit-learn, bagging methods are offered as a unified BaggingClassifier meta-estimator (resp. BaggingRegressor), taking as input a user-specified estimator along with parameters specifying the strategy to draw random subsets.

Source: https://scikit-learn.org/stable/modules/ensemble.html

35

35

Boosting methods

IMPERIAL

- By contrast, in **boosting methods**, base estimators are built sequentially, and one tries to reduce the bias of the combined estimator.
- The motivation is to combine several weak models to produce a powerful ensemble.
- Examples: XGBoost, Gradient Tree Boosting

Source: https://scikit-learn.org/stable/modules/ensemble.html

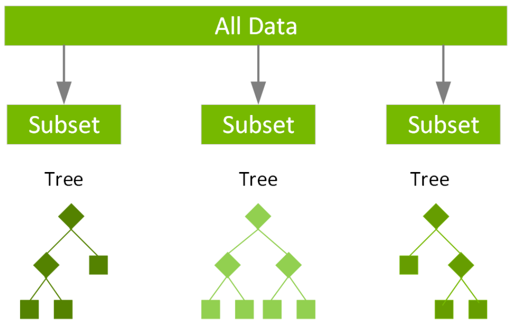
36

36

Gradient Boosting Trees

IMPERIAL

- A Gradient Boosting Decision Tree (GBDT) is a decision tree ensemble learning algorithm similar to the random forest for classification and regression.
- Ensemble learning algorithms combine multiple machine learning algorithms to obtain a better model.
- Both random forest and GBDT build a model consisting of multiple decision trees. The difference is in how the trees are built and combined.



Source: NVIDIA, https://www.nvidia.com/en-us/glossary/data-science/xgboost/

37

37

Gradient Boosting

IMPERIAL

- The term "gradient boosting" comes from the idea of "boosting" or improving a single weak model by combining it with several other weak models to generate a collectively strong model.
- Gradient boosting sets targeted outcomes for the next model to minimise errors.
- Targeted outcomes for each case are based on the gradient of the error (hence the name gradient boosting) with respect to the prediction.

Source: NVIDIA, https://www.nvidia.com/en-us/glossary/data-science/xgboost/

38

38

XGBoost

IMPERIAL

- GBDTs iteratively train an ensemble of shallow decision trees, with each iteration using the error residuals of the previous model to fit the next model.
- The final prediction is a weighted sum of all of the tree predictions. Random forest “bagging” minimises the variance and overfitting, while GBDT “boosting” minimises the bias and underfitting.
- With XGBoost, trees are built in parallel instead of sequentially like GBDT.
- XGBoost follows a level-wise strategy, scanning across gradient values and using these partial sums to evaluate the quality of splits at every possible split in the training set.

Source: NVIDIA, https://www.nvidia.com/en-us/glossary/data-science/xgboost/

39

39

Random Forests

IMPERIAL

- In random forests, each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set.
- When splitting each node during the construction of a tree, the best split is found either from all input features or a random subset of size `max_features`.
- The purpose of these two sources of randomness is to decrease the variance of the forest estimator.

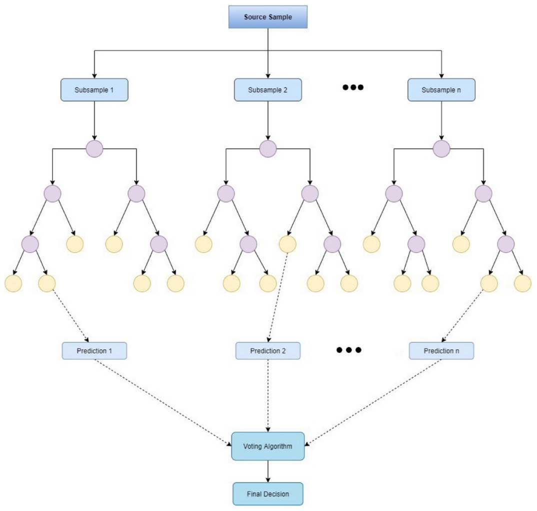
Source: https://scikit-learn.org/stable/modules/ensemble.html

40

40

Structure of Random Forests

IMPERIAL



Source: Malebary, S.J., Khan, Y.D. Evaluating machine learning methodologies for identification of cancer driver genes. *Sci Rep* **11**, 12281 (2021). https://doi.org/10.1038/s41598-021-91656-8

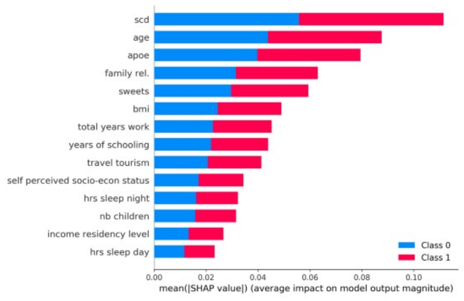
41

41

Feature important in RF: Example

IMPERIAL

– Selecting the most important self-assessed features for predicting conversion to mild cognitive impairment with random forest and permutation-based methods:
  – Gómez-Ramírez, J., et al.(2020), https://doi.org/10.1038/s41598-020-77296-4



(b) Stacked bar plot with the mean absolute value of the SHAP values in the horizontal axis and the features in the vertical axis. Blue represents Class 0 or Healthy and in red is depicted Class 1 or MCI converters.

42

42

Random Forests

IMPERIAL

- Individual decision trees typically exhibit high variance and tend to overfit.
- The injected randomness in forests yields decision trees with somewhat decoupled prediction errors.
- By taking an average of those predictions, some errors can cancel out. Random forests achieve a reduced variance by combining diverse trees, sometimes at the cost of a slight increase in bias.

Source: https://scikit-learn.org/stable/modules/ensemble.html

43

43

Practical note on RFs

IMPERIAL

- In practice, the variance reduction is often significant, yielding an overall better model.
- In contrast to the original publication, the scikit-learn implementation combines classifiers by averaging their probabilistic prediction instead of letting each classifier vote for a single class.

Source: https://scikit-learn.org/stable/modules/ensemble.html

44

44

SHAP

IMPERIAL

- SHAP (SHapley Additive exPlanations) is a game theoretic approach to explaining the output of any machine learning model.
- It connects optimal credit allocation with local explanations using the classic Shapley values from game theory and their related extensions.

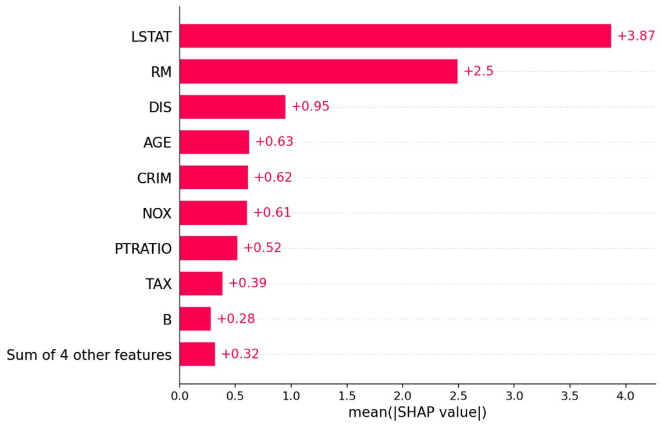
- See: https://github.com/slundberg/shap

45

45

SHAP example 1

IMPERIAL

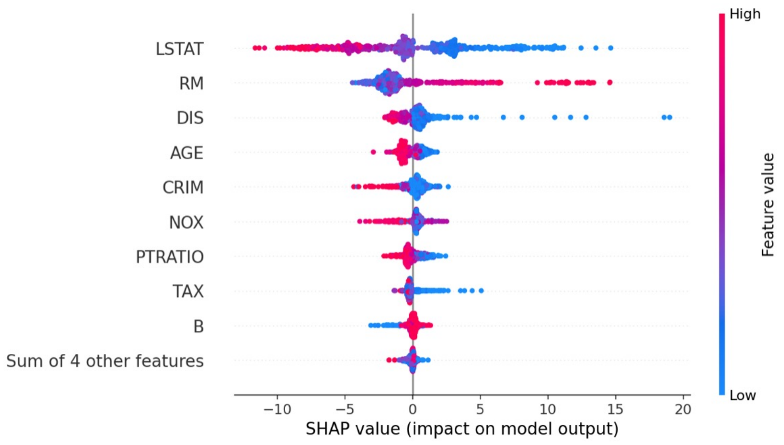


46

46

SHAP example 2

IMPERIAL

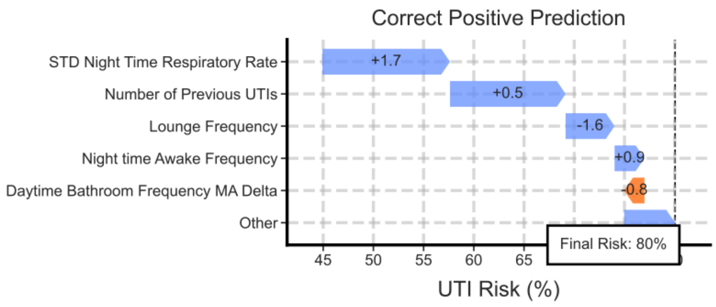


47

47

SHAP example 3

IMPERIAL



Capstick, A., Palermo, F., Zakka, K. *et al.* Digital remote monitoring for screening and early detection of urinary tract infections. *npj Digit. Med.* **7**, 11 (2024). https://doi.org/10.1038/s41746-023-00995-5

48

48

IMPERIAL

Review questions

49

Q1 – RF vs DT

IMPERIAL

- If we have a dataset with multiple features, the samples of which vary, and the different features contribute to the overall class prediction, which of DT or RF would be a good choice?

- menti code will be provided.

50

50

Q2

IMPERIAL

- In a random forest model, when samples are drawn with replacement, what is the method known?

51

51

Q3

IMPERIAL

- In SVM models, there is a hyperparameter known as the Soft Margin of SVM (shown as **C** in the model implementation). The **C** hyperparameter introduces a penalty for each misclassified sample.
- If you set **C** to a large value, it will imply a small margin in support vectors. What issue do you think the small margins in support vectors could then cause?

52

52

If you have any questions

IMPERIAL

- Please feel free to arrange a meeting or email (p.barnaghi@imperial.ac.uk).
- My office: 928, Sir Michael Uren Research Hub, White City Campus.

53

53