



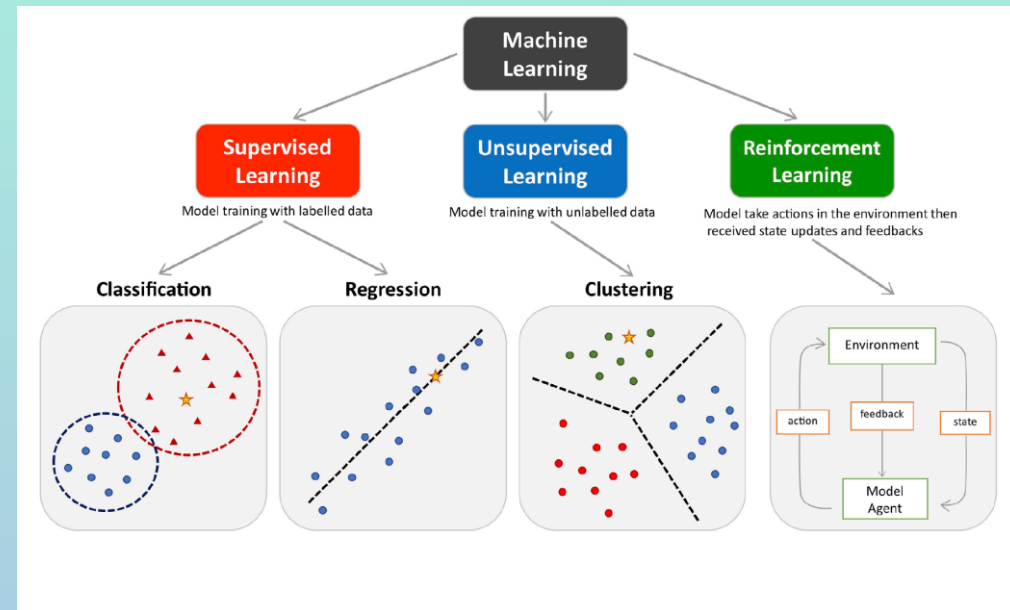
Artificial Intelligence (ACCA7026)

Supervised and Unsupervised Learning

Dr. Seena Joseph

Machine Learning algorithm

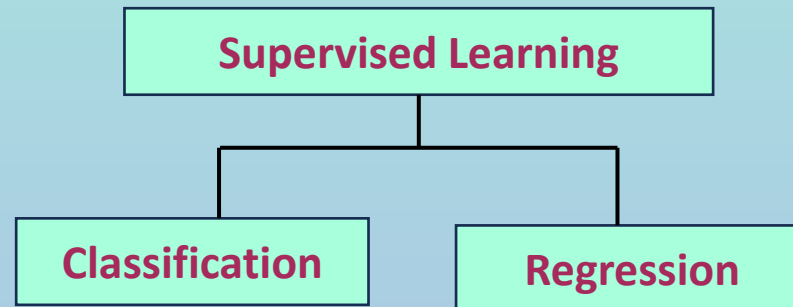
- Supervised learning
- Unsupervised learning
- Reinforcement learning



Supervised Learning

Supervised learning algorithm is trained on a labelled dataset

- Labelled data-> Model training-> Prediction (**Classification and Regression**)
- Classification – prediction of output of discrete values [predicting Classes]
- Regression – prediction of output of continuous values [**Predicting Values**]



Supervised Learning

Classification and Regression

Regression



price: 124 032\$

Classification



category: cookie

Supervised Learning

Classification

Training set



weight: 62
width: 14
is_cookie: 1



weight: 45
width: 12
is_cookie: 1



weight: 12
width: 6
is_cookie: 0



weight: 59
width: 12.5
is_cookie: 1



weight: 16
width: 5
is_cookie: 0

} features
} target

New instance

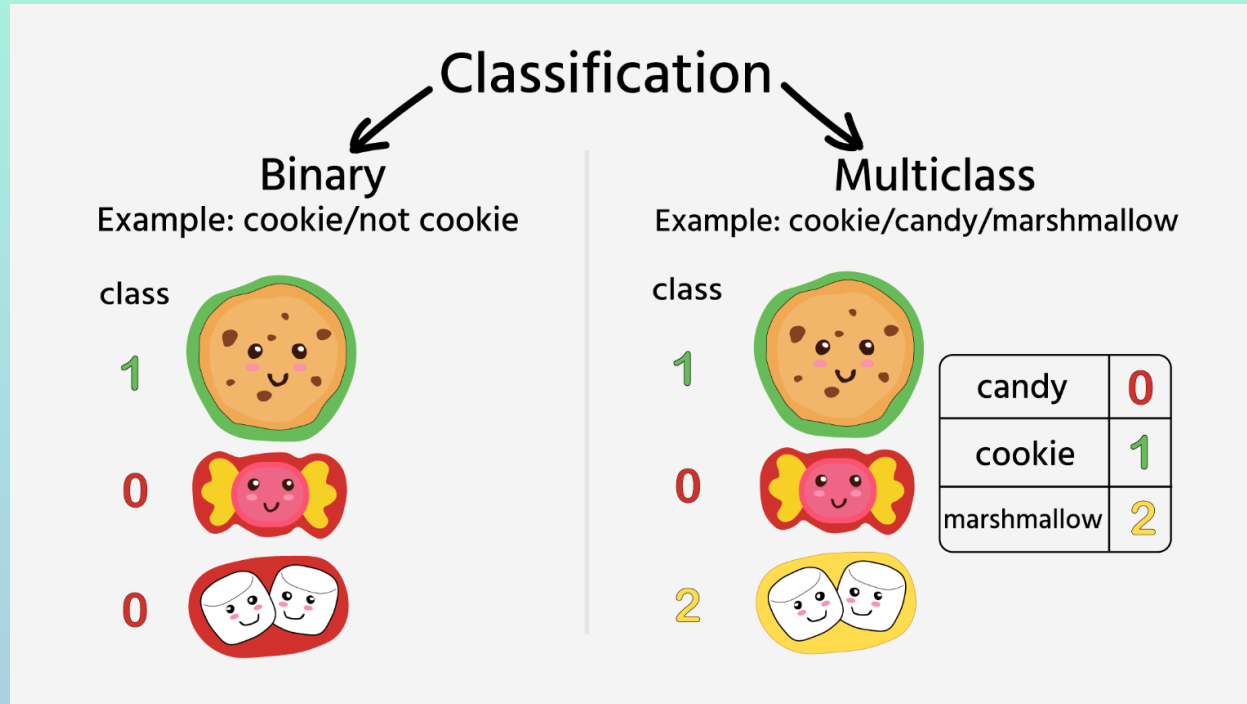


weight: 26
width: 9
is_cookie: ???

need to predict

Supervised Learning

Classification – Binary or Multiclass

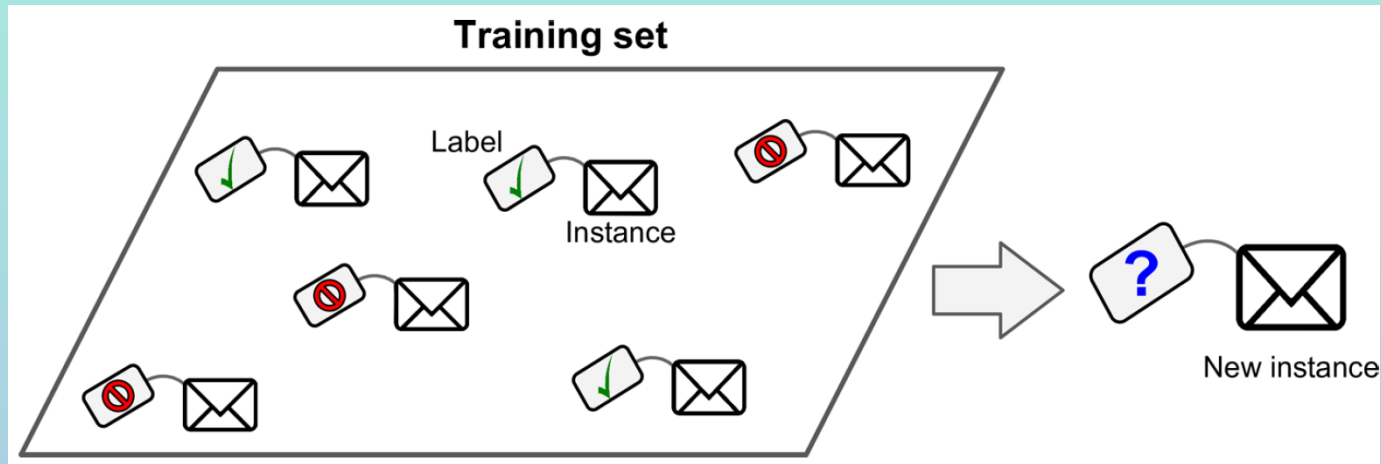


Supervised Learning

The training set contains the desired solution.

Spam filter is a good example for supervised classification.

It is trained with many example emails with their classes (Spam or Ham) to learn how to classify new emails.



Is this binary or multiclass ?

Supervised Learning

Example for supervised learning algorithms

- Linear Regression (Regression)
- Logistic Regression (Classification)
- Support Vector Machine (Classification and Regression)
- K-Nearest Neighbor (Classification)
- Decision Tree (Classification and Regression)
- Random Forest (Classification and Regression)
- Naïve Bayes (Classification)
- Neural Networks (Classification and Regression)

Unsupervised Learning

- The purpose is to either transform the data or discover pattern without the need of a set of explicit labeled data.
- The training data is unlabeled

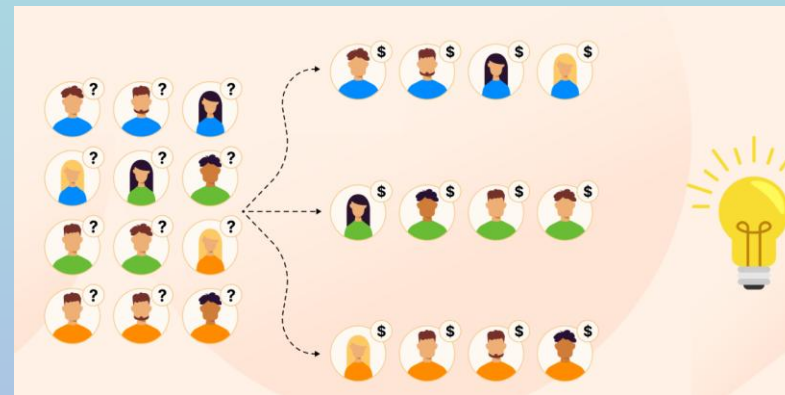
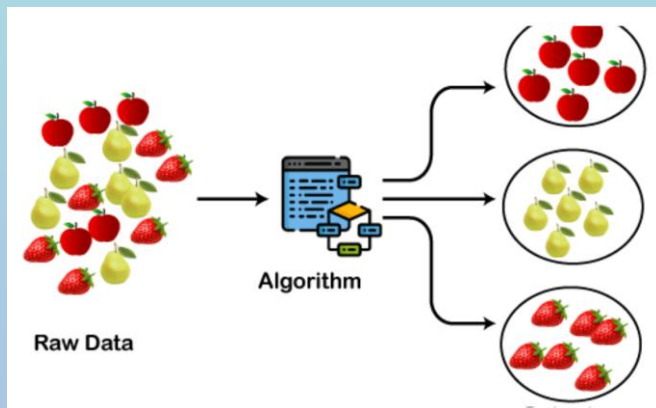
Common Unsupervised Learning algorithm

- Clustering
- Dimensionality Reduction
- Association rule learning

Unsupervised Learning

Clustering : Group similar instance into clusters.

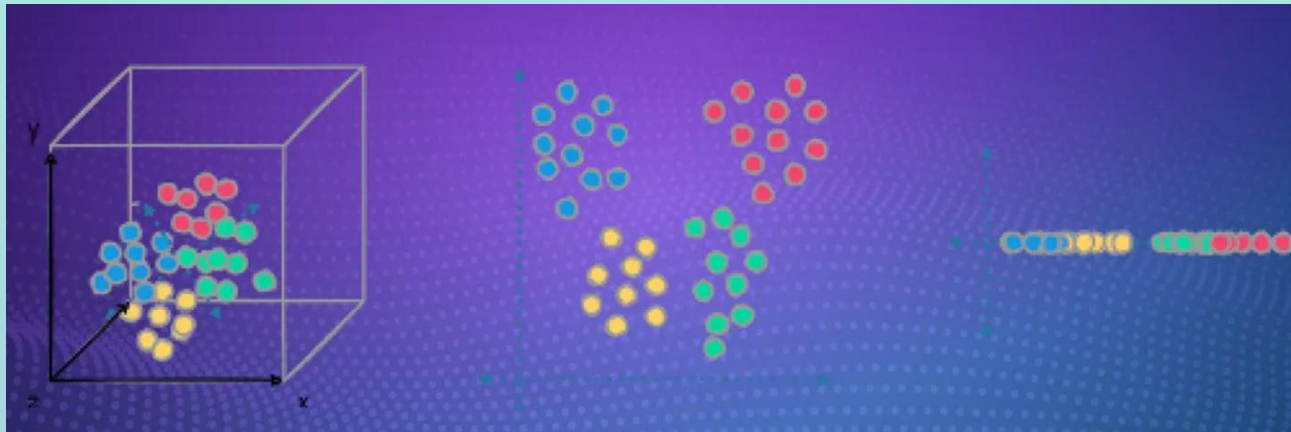
- Examples: K Means, Fuzzy C Means, DBSCAN, MeanShift, Spectral Clustering etc.
- Applications
 - Customer segmentation – cluster customers based on the activity on the website or their purchases
 - For data analysis – easy to analyse the data in each cluster separately



Unsupervised Learning

Dimensionality reduction : Reducing the number of dimensions. Reduce the number of features/dimensions while keeping the essential information.

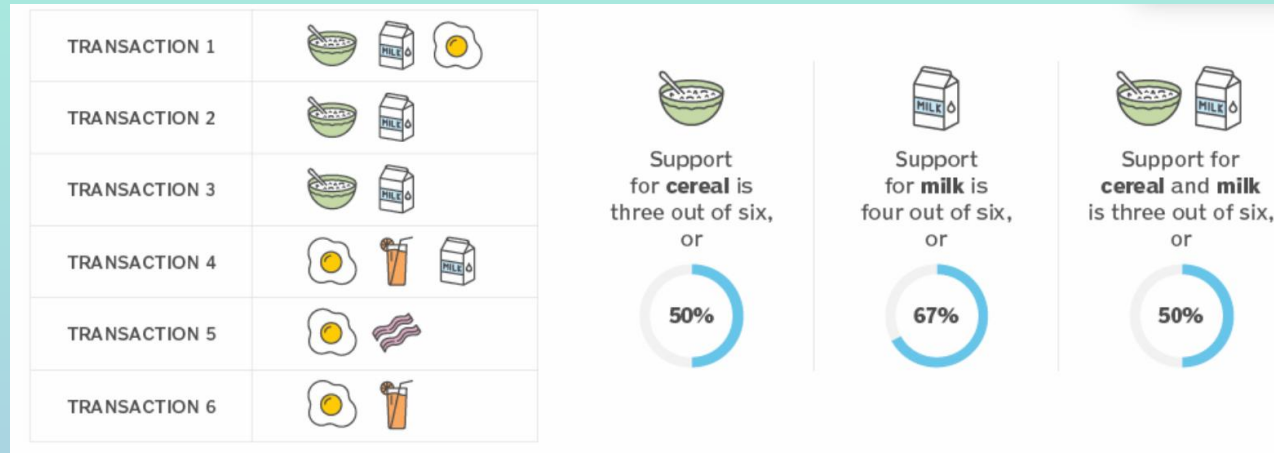
- Example: Principal Component Analysis (PCA)



Unsupervised Learning

- Association rule learning : Discover interesting relationship variables in a dataset.

Example : Apriori, Eclat



ML Model - Performance Measures

Confusion Matrix

A binary classification problem [YES/NO]

Is this transaction Fraudulent ?

		<u>Predicted</u> Fraud	
		Y	N
<u>Is Actual</u> Fraud	Y	+ / + TP	- / + FN
	N	+ / - FP	- / - TN

ML Model - Performance Measures

Confusion Matrix

		<u>Predicted</u> Fraud	
		Y	N
<u>Is Actual Fraud</u>	Y	+ / + TP	- / + FN
	N	+ / - FP	- / - TN

Case 1 [True Positive]: A transaction was fraud, and the classifier correctly labelled it as fraud.

Case 2 [False Positive]: A transaction was not fraud and the classifier incorrectly labelled it as fraud. [Type I Error]

Case 3 [False Negative]: A transaction was fraud, and the classifier incorrectly labelled it as not fraud. [Type II Error]

Case 4 [True Negative]: A transaction was not fraud and the classifier correctly labelled it as not fraud.

ML Model - Performance Measures

Precision

		Predicted Fraud	
		Y	N
Is Actual Fraud	Y	2 TP	4 FN
	N	8 FP	86 TN

Precision – What proportion of the positive identification was actually correct ?

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

$$\text{Example, Precision} = \frac{2}{2+8} = \frac{2}{10} = 0.2$$

What percentage of the time an instance labelled as positive (predicted as positive) was actually positive ?

FP should be low for a high Precision.

ML Model - Performance Measures

Recall

		<u>Predicted</u> Fraud	
		Y	N
<u>Is Actual</u> Fraud	Y	2 TP	4 FN
	N	8 FP	86 TN

Recall – what proportion of the actual positives were identified correctly ?

$$\text{Recall} = \frac{TP}{TP+FN}$$

$$\text{Example, Recall} = \frac{2}{2+4} = \frac{2}{6} = 0.3$$

How **sensitive** the classifier is the indicators of possible fraud?

FN should be low for a High Recall.

ML Model - Performance Measures

F1 Score – the harmonic mean of Precision and Recall

Low Recall (Miss Some Fraud)
High Precision (Few False Positive)

High Recall (Catch Most Fraud)
Low Precision (Many False Positive)



Conservative
Flag fewer transaction

Aggressive
Flag more transaction

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

ML Model - Performance Measures

Accuracy – it is the ratio of the correct predictions made by a classification model out of the total predictions. It provide an overall idea of how often the model is correct.

The correct prediction is sum of TP and TN.

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

$$= \frac{2+86}{2+8+86+4} = .88 \quad (88\%)$$

	Y	N
Y	2 TP	4 FN
N	8 FP	86 TN

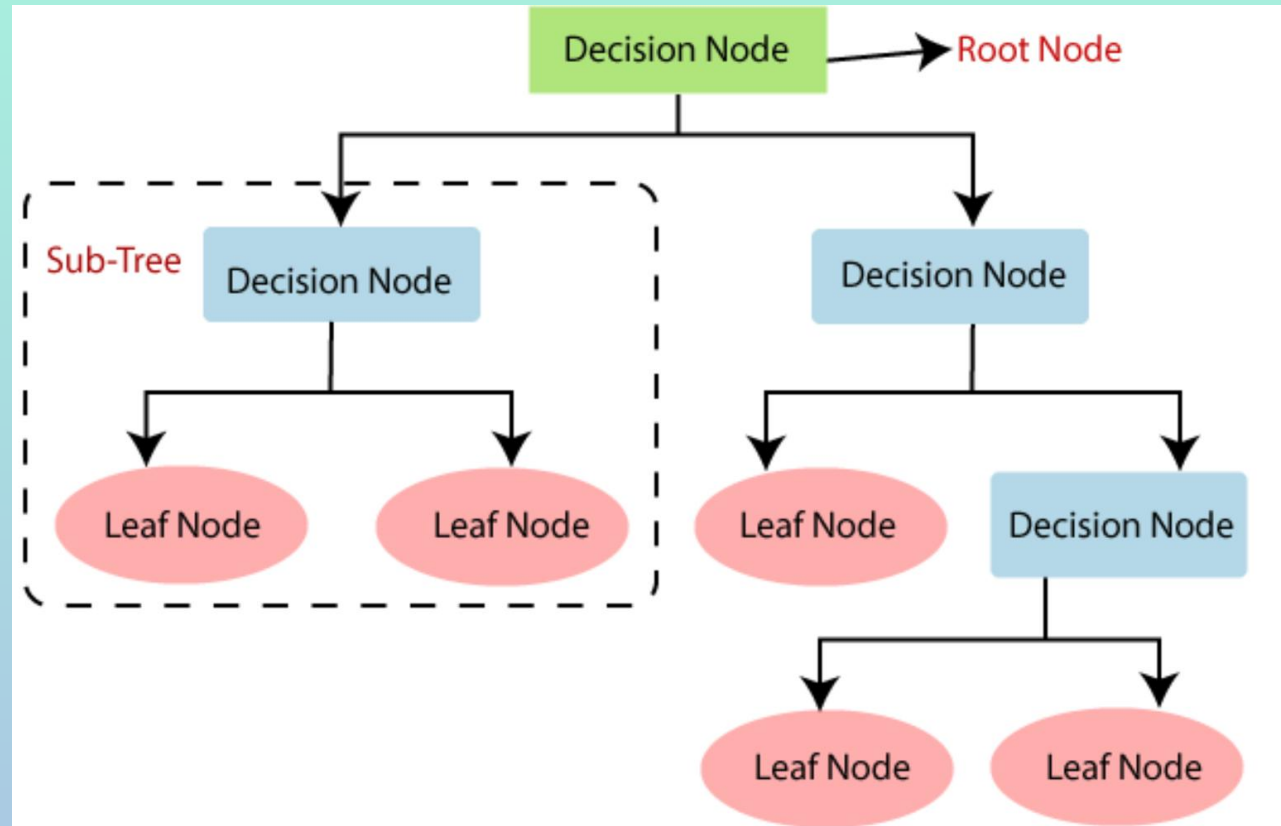
ML Model - Performance Measures

Activity

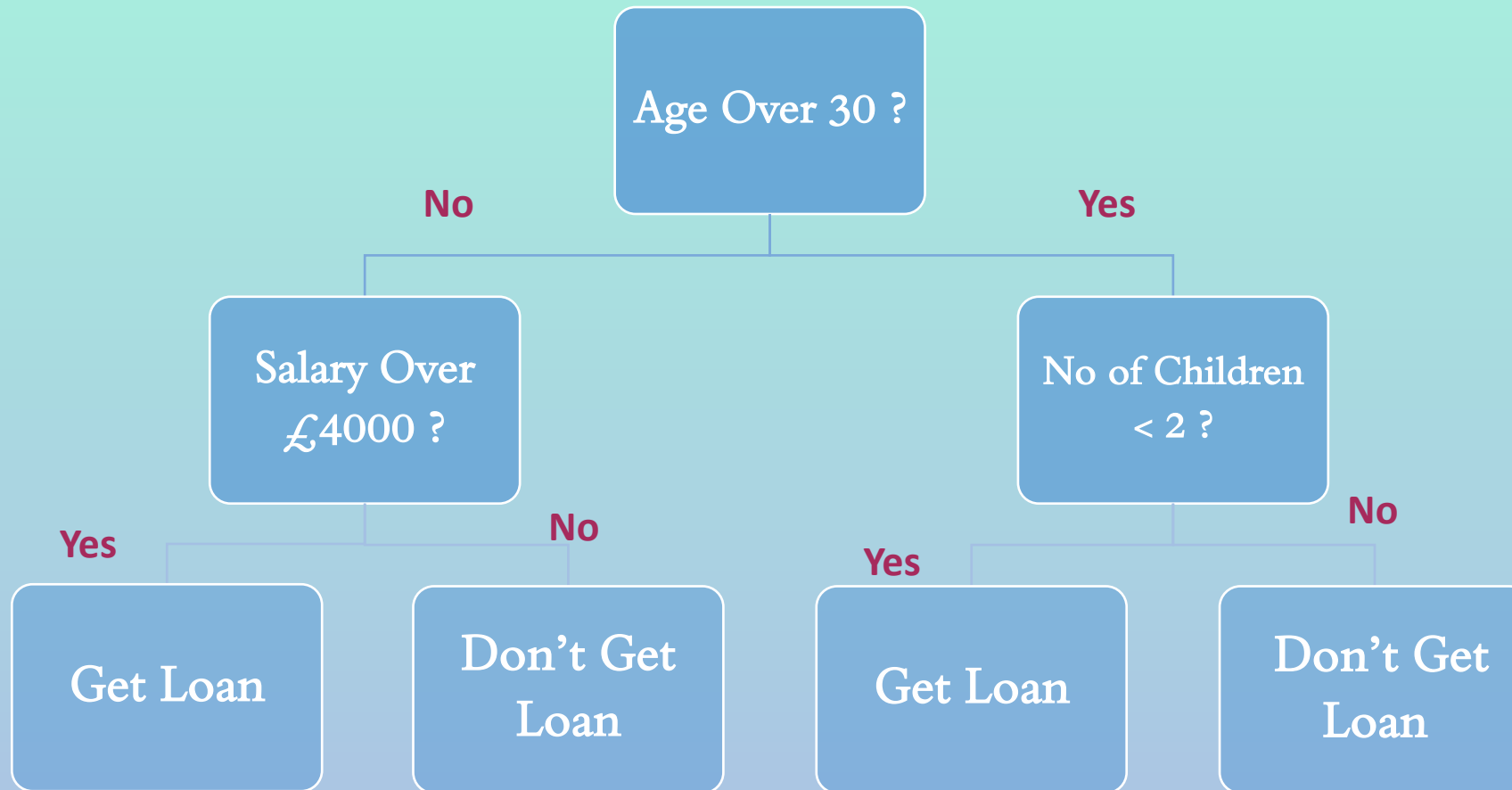
1. "Imagine you're developing a test to detect a rare but dangerous disease. Would it be more important for this test to have high precision or high recall? Why?"
2. For an email spam filter, would you prefer high precision or high recall, and why?"
3. "Imagine you are developing an automated hiring system that shortlists candidates. Would it be more beneficial to prioritize high precision or high recall?"
4. "For a credit card fraud detection system, would high recall or high precision be more important, and why?"

Supervised Learning - Decision Tree

- General structure of a Decision Tree



Supervised Learning - Decision Tree



Supervised Learning - Decision Tree

- Components

- Root node: the topmost node of the tree where the decision-making process starts based
- Internal node : Represent the decision based on a feature or attribute
- Leaf nodes : Represent the final output or prediction node
- Parent and child node : base node is the parent node and the remaining nodes are child node.
- Branches : A subtree formed by splitting the tree ; outcome of a decision
- Pruning – process of removing unwanted branches from the tree
- Splitting – process of dividing the decision node/ root node into sub-nodes based on the given condition

Supervised Learning - Decision Tree

Attribute/Features – characteristics or property of dataset that is used to make decisions at each node of the tree.

Each attribute corresponds to a question that helps to split the data into subsets.

Type of features

1. Numerical features

Continuous or discrete values (age, salary, temperature)

Numerical features are used to split the tree by comparing against a threshold.. Eg is **Age > 50**

2. Categorical feature

These are discrete, non-numerical values that represent categories or classes. Example is the weather sunny, rainy or cloudy ? Split of categorical features are by checking the membership in a set of values (is the weather sunny ?)

Supervised Learning - Decision Tree

Role of feature in splitting

A feature is used to split the data-based on a specific condition

Example: is **Age** > 50 ? or Is **Salary** high ?

The purpose of feature selection is split the data into homogenous subsets, reducing impurity.

Feature selection – The decision tree algorithm evaluate the multiple features at each node and selects the one with the highest information gain or less impurity.

Supervised Learning - Decision Tree

Attribute Selection Measures (ASM)

How to select the best attribute for the root node and sub nodes.

ASM involves the collection of optimum attributes concerning the source node as well as for the sub-nodes.

The goal of **ASM** is to find the features that split the data into subsets that are homogenous/pure as possible.

Best practices for ASM are:

1. Information gain
2. Gini index

Supervised Learning - Decision Tree

Information gain- Calculates the amount of information that is provided by a feature regarding the class.

Information Gain = Entropy (S) – [(Weighted Average)*Entropy (Every Feature)]

Entropy – is the measure of impurity or disorder or uncertainty;

Entropy values : range from 0 to 1; less the entropy value better the outcome;

Entropy controls how a decision tree decides to split the data.

Supervised Learning - Decision Tree

How to calculate entropy ?

$$\text{Entropy}(s) = -P(\text{Yes})\log_2 P(\text{Yes}) - P(\text{No})\log_2 P(\text{No})$$

where $P(\text{Yes}) \rightarrow$ probability/Chance of getting yes

$P(\text{No}) \rightarrow$ probability/Chance of getting No

Supervised Learning - Decision Tree

Gini Index – Measures the impurity or purity used during the creation of a decision tree algorithm.

$$Gini(D) = 1 - \sum_{i=1}^k p_i^2$$

Where p_i is the probability of choosing a sample of class i .
K is the number of classes

A lower Gini Index represents the better split

Supervised Learning - Decision Tree

Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).

Step-3: Divide the S into subsets that contains possible values for the best attributes.

Step-4: Generate the decision tree node, which contains the best attribute.

Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Supervised Learning - Decision Tree

Example

Dataset: Weather and Decision to Play

Weather	Windy	Play?
Sunny	No	No
Sunny	Yes	No
Overcast	No	Yes
Rainy	No	Yes
Rainy	Yes	No
Rainy	No	Yes
Overcast	Yes	Yes
Sunny	No	Yes

- Feature 1 : Weather (Sunny, Overcast, Rainy)
- Feature 2: Windy (Yes, No)
- Target: Play? (Yes, No)

Supervised Learning - Decision Tree

Example for calculating information gain

Number of instances for Yes or No for the Target variable, Play.

Yes = 5

No = 3

Step 1:

Entropy for the whole dataset, S

$$\text{Entropy}(S) = -P(\text{Yes})\log_2 P(\text{Yes}) - P(\text{No})\log_2 P(\text{No})$$

$$= -5/8\log_2(5/8) - (3/8)\log_2(3/8)$$

$$= -(0.625 \log_2 0.625 + 0.375 \log_2 0.375)$$

$$= -(0.625 \times -0.678 + 0.375 \times -1.415) = 0.954$$

Supervised Learning - Decision Tree

Step 2:

Calculate the entropy for Weather

Sunny; Overcast; Rainy

Sunny = 3 instances with 2 No and 1 Yes (entropy =0.918)

Overcast = 2 instances ; 2 Yes (entropy =0)

Rainy = 3 instances with 2 No and 1 Yes ; (entropy =0.918)

Weighted entropy : $\frac{3}{8} \times 0.918 + \frac{2}{8} \times 0 + \frac{3}{8} \times 0.918 = 0.688$

Step 3: entropy (s) – entropy (weather)

= 0.954 - 0.688 = **0.266**

Supervised Learning - Decision Tree

Example Gini Index

Number of instances for Yes or No for the Target variable, Play.

Yes = 5

No = 3

Step 1:

$$\begin{aligned} \text{Gini}(S) &= 1 - \left(\left(\frac{5^2}{8} \right) + \left(\frac{3^2}{8} \right) \right) \\ &= 1 - (0.3906 + 0.1406) = 1 - 0.5312 = 0.4688 \end{aligned}$$

Supervised Learning - Decision Tree

Example Gini Index

Step 2 : Calculate Gini Index for Splitting on "Weather"

Sunny : 3 instances, 2 "No", 1 "Yes" (Gini = 0.444).

Overcast: 2 instances, 2 "Yes" (Gini = 0).

Rainy: 3 instances, 2 "Yes", 1 "No" (Gini = 0.444).

Calculate the weighted Gini Index

$$= 3/8 \times 0.444 + 2/8 \times 0 + 3/8 \times 0.444 = 0.333$$

Supervised Learning - Decision Tree

Example interpretation

Information Gain for splitting on "Weather" is **0.266**. This shows that splitting on this feature reduces uncertainty (entropy), but the reduction is moderate.

Gini Index after the split is **0.333**, indicating that the groups formed by this split are somewhat pure, though not entirely.

Supervised Learning - Decision Tree

Example interpretation

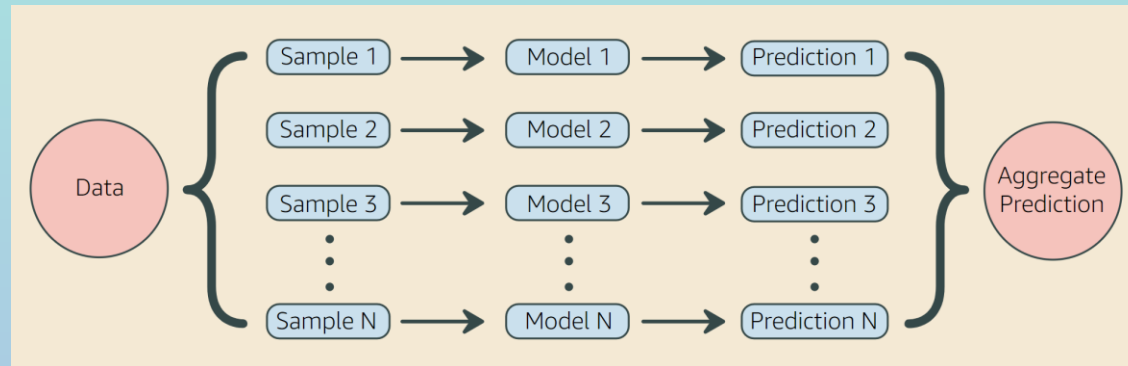
Information Gain for splitting on "Weather" is **0.266**. This shows that splitting on this feature reduces uncertainty (entropy), but the reduction is moderate.

Gini Index after the split is **0.333**, indicating that the groups formed by this split are somewhat pure, though not entirely.

Supervised Learning - Random Forest

Ensemble Learning

Create a stronger model by aggregating the prediction of multiple weak models. This is the basis of the Random forest model.



Ensemble Learning

Types of Ensemble Learning

- Bagging
- Boosting
- Stacking

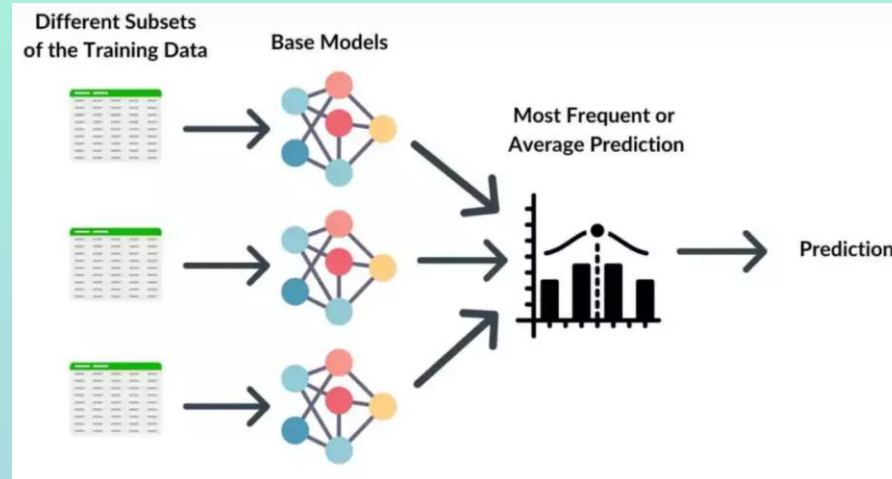
Bagging – Multiple version of the same model is trained on a different subset of the training data (Created using bootstrapping and samples are randomly selected).

Each subset will be unique and may contain some repeated instance.

Each model makes prediction and aggregate the prediction for the final model.

Ensemble Learning

Bagging



- Regression – Average Prediction
- Classification – Majority vote (most frequent prediction)

Random forest is an example of Bagging Ensemble

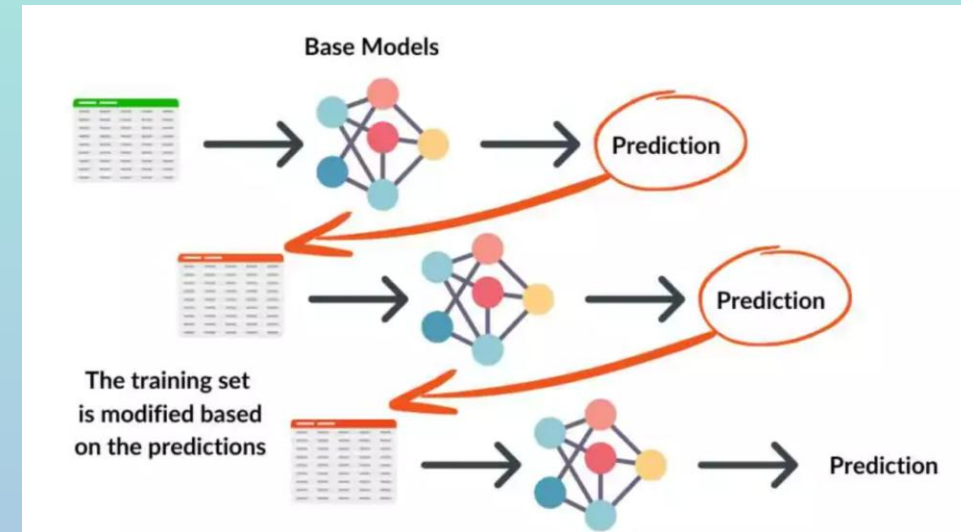
Ensemble Learning

Boosting

Boosting builds multiple models sequentially, where each model tries to correct the errors of the previous one.

Train predictors sequentially and each trying to correct his predecessor.

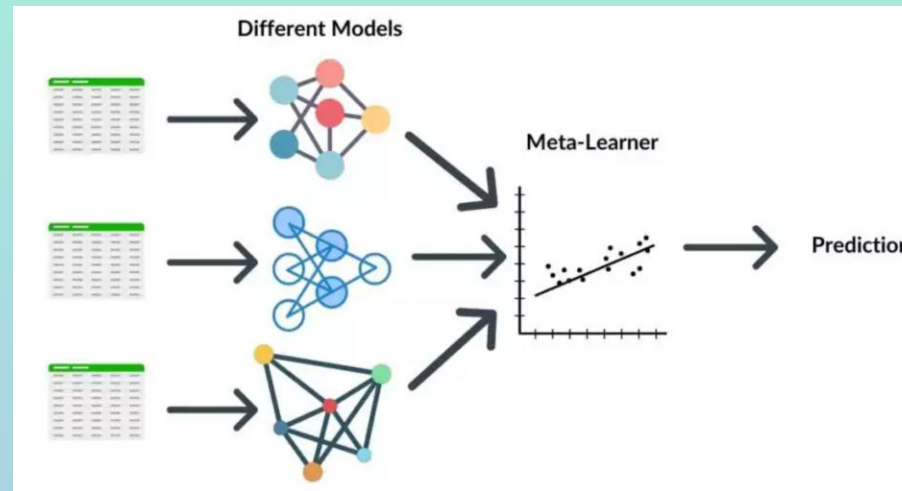
Example: AdaBoost, XGBoost



Ensemble Learning

Stacking

Combines the prediction of multiple heterogenous models using a meta-learner to produce the final output.



- A stacking ensemble might combine predictions from a decision tree, logistic regression, and a support vector machine, with a final model (like linear regression) that uses these predictions as input to make the ultimate prediction.

Supervised Learning - Random Forest

Random forest is an example of **bagging ensemble** where each model is a decision tree.

Train multiple Decision Trees

Trees are trained independently in parallel

For classification – Majority Voting

For Regression – Average

Majority vote aggregation and weighted majority vote aggregation

[Random Forest](#)

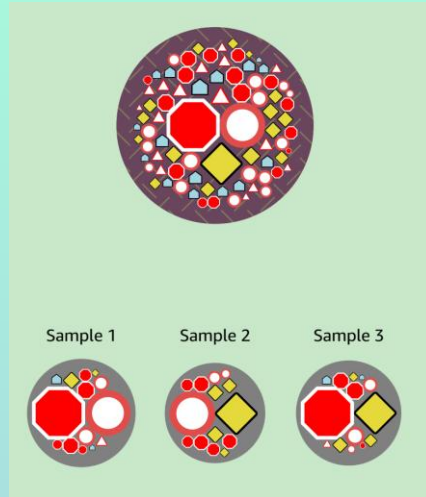
Supervised Learning - Random Forest

How Random Forest works

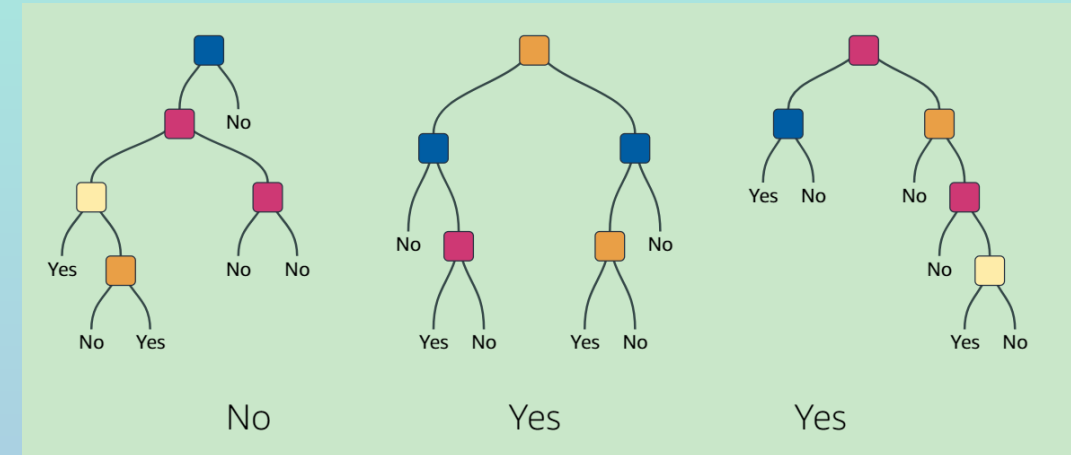
- Data sampling (Bagging)
 - Multiple decision trees, by selecting random subsets of the original dataset.
 - Sampling with replacement – Data points are shared among samples.
 - Each tree is trained on a different dataset
- Random Feature selection
 - Selects a random subset of features rather than considering all features
- Prediction Aggregation
 - In classification – each tree votes for a class and the Random forest chooses the class with the most votes.
 - In regression tasks, the prediction from each tree are averaged to get the final prediction.

Supervised Learning - Random Forest

Bootstrapping



Random Feature selection



Random forest – Simulation

[Random Forest](#)

Hands-on Project

Python Demo

Python code snippet – Refer Lecture notes 2

Adaptive Boosting (Adaboost)

Boosting is an ensemble learning technique that combines many weak learners (usually shallow decision trees) to form a single strong learner.

AdaBoost focuses on misclassified samples.

- It increases their importance (weights) so that the next weak learner pays more attention to them.
- Each weak model (often a “stump” a one-level decision tree) contributes to the final model based on how accurate it is.

Adaptive Boosting (Adaboost)

Start with equal weights:

Every training sample is given the same weight initially.

Train a weak learner:

Fit a simple classifier (like a decision stump).

Compute errors:

Find which samples are misclassified.

Update weights:

Increase weights for misclassified samples (make them more important).

Decrease weights for correctly classified ones.

Compute model weight:

The model's vote depends on its accuracy — better models get higher weights.

Repeat:

Train the next model using updated weights.

Combine all models into a weighted sum (final prediction).

AdaBoost says: “Let’s focus on what I got wrong last time.”

It reweights the data instead of changing the algorithm itself.

Adaptive Boosting (Adaboost)

Pros and Cons ??

Gradient Boosting

- Gradient Boosting takes a more mathematical approach. Instead of reweighting data like AdaBoost, it fits new models to the residual errors (the difference between actual and predicted values).
- Each new model learns to predict the remaining mistakes of the combined model so far.

Gradient Boosting

Start with an initial prediction:

For regression, this might be the mean of the target variable.

Compute residuals:

$\text{Residual} = (\text{Actual value} - \text{Predicted value}).$

These residuals represent the errors we still need to fix.

Train a weak learner (tree) on residuals:

The tree learns how to correct those errors.

Update predictions:

Add the new tree's predictions (scaled by a learning rate) to the previous model's predictions.

Repeat for many trees:

Each new tree improves the model by focusing on what previous ones missed.

It's called "gradient" boosting because it uses the gradient (slope) of the loss function such as Mean Squared Error or Log Loss to figure out how to improve the model step by step, similar to how gradient descent works in neural networks.

Gradient Boosting

Pros and Cons ??

XGBoost (short for Extreme Gradient Boosting) is an advanced and optimized version of the Gradient Boosting algorithm.

XGBoost = Gradient Boosting + Speed + Regularization + Efficiency.

Extra reading

[Bagging, Boosting & Stacking Made Simple \[3 How To Tutorials\]](#)

[1.11. Ensembles: Gradient boosting, random forests, bagging, voting, stacking — scikit-learn 1.5.2 documentation](#)

[Random Forest vs Decision Tree | Which Is Right for You?](#)

[What is A Confusion Matrix in Machine Learning? The Model Evaluation Tool Explained | DataCamp](#)

Reference

Géron, Aurélien. *Hands-on Machine Learning with Scikit-Learn, Keras and TensorFlow : Concepts, Tools, and Techniques to Build Intelligent Systems*. Third edition. Sebastapol, CA: O'Reilly Media, Inc., 2023. Print.

[\[2\] Decision Tree Algorithm in Machine Learning – Javatpoint](#)