GBDi
Government Big Data Institute
สถาบันส่งเสริมการวิเคราะห์และบริหารข้อมูลขนาดใหญ่ภาครัฐ (สวข.)

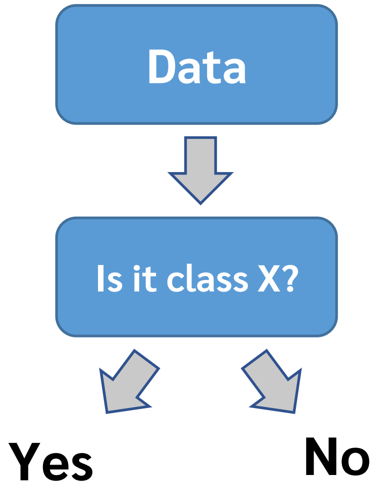
depa

Introduction to Supervised Learning: Basic Classification Models

Patipan Prasertsom

Classification

- In machine learning, classification problem is a problem of predicting the correct label (class) that a given data belongs to.
- There are many types of classification algorithms, such as
    - Regression-based classifiers: ex. logistic regressions
    - Tree-based classifiers: ex. decision tree, random forest, gradient tree boosting
    - Probability-based classifiers: ex. Bayes classifier
    - Similarity-based classifier: ex. k-Nearest Neighbor classifier
    - etc.
- Today, we will touch on some of the well-known classifiers



k-Nearest Neighbors classifier (kNN)

- **The idea**: similar data points are likely to be of the same type.
- We infer a class of a data point from its ***k* most similar** data
- How do we find the “most similar” (nearest) data points?
  - There are many way of measuring the distance between data point. One frequently discussed method is the Euclidean distance.
  - Euclidean distance – a distance between 2 points in cartesian coordinates

$$Euclidean\ Distance = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

- Basically asking, “**By looking at k data points that are most similar to me, which class am I most likely to be in?”**

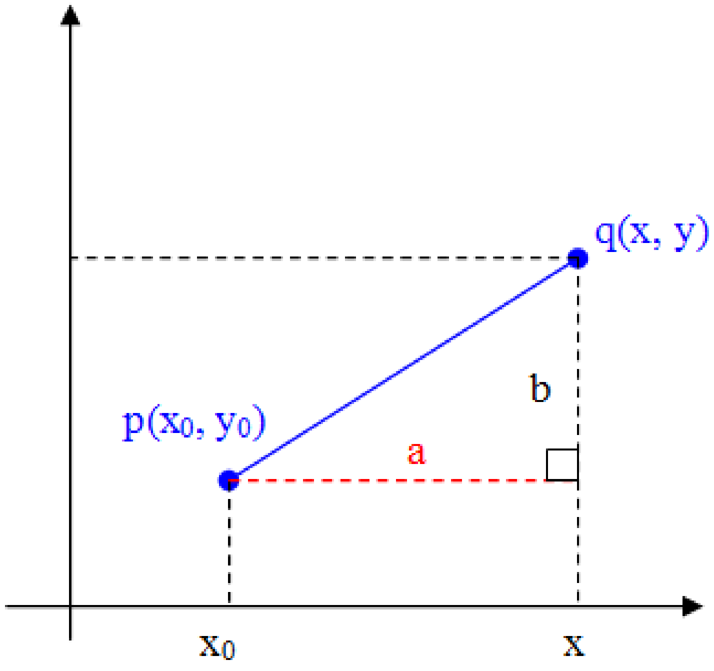


Figure from Illuyanka (https://commons.wikimedia.org/wiki/File:Dot_Product.svg)

k-Nearest Neighbors classifier (kNN)

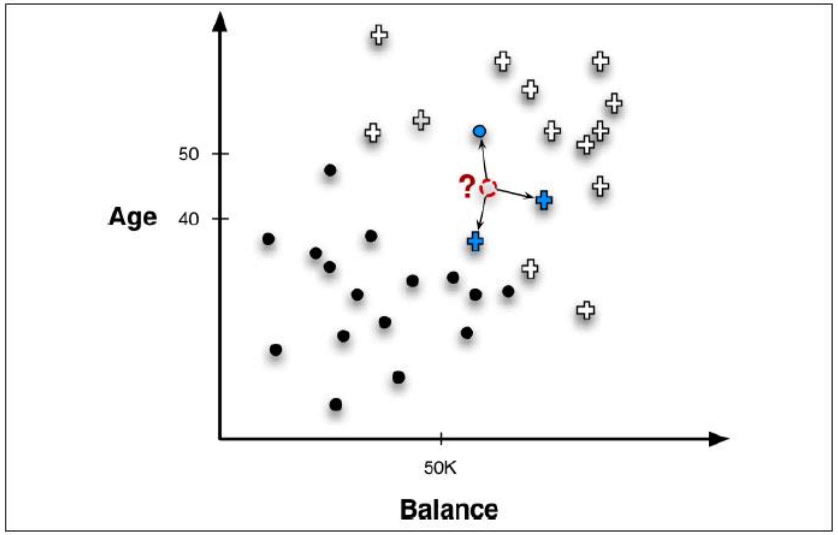
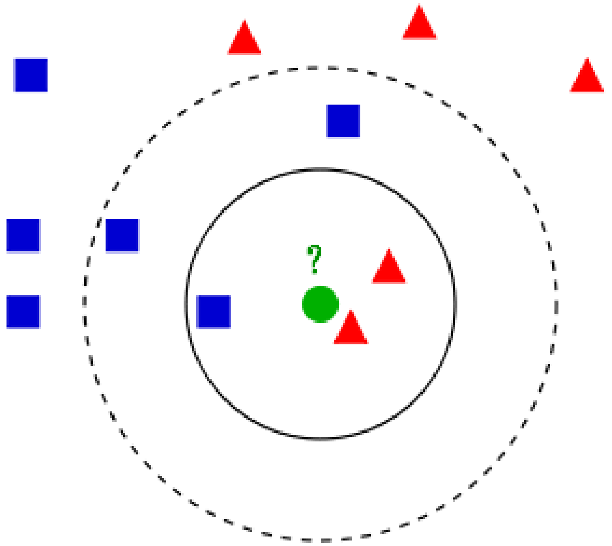


Figure 6-2. Nearest neighbor classification. The point to be classified, labeled with a question mark, would be classified + because the majority of its nearest (three) neighbors are +.



- Observes k closest data point and decide the class of data points by voting.
- Usually, k is chosen as an odd number

- The choice of ***k*** matters!
- Different number of k can result in different predictions
- Feature scale matters!

Figures from Data Science for Business by Foster Provost & Tom Fawcett and Antti Ajanki AnAj (https://commons.wikimedia.org/wiki/File:Dot_Product.svg)

k-Nearest Neighbors classifier (kNN)

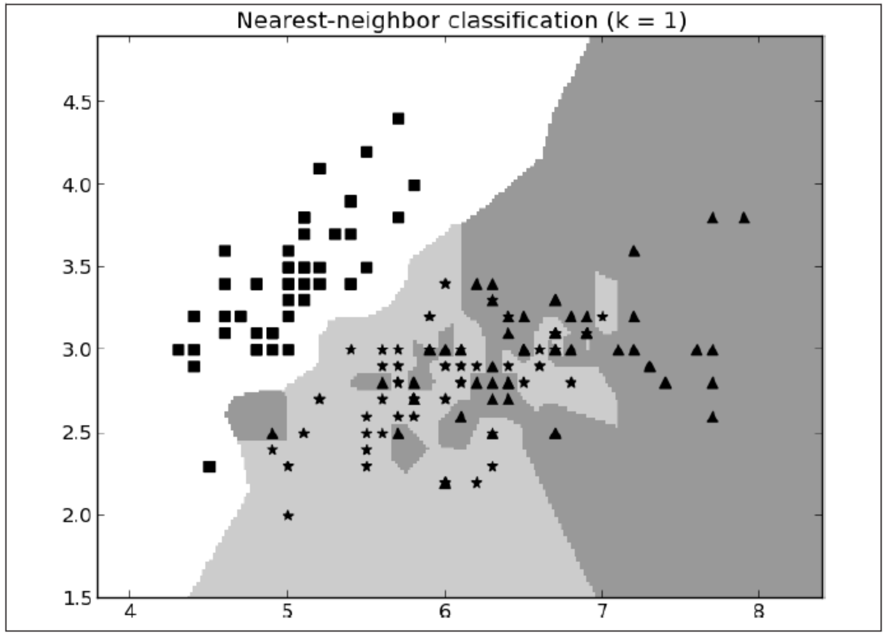


Figure 6-4. Classification boundaries created on a three-class problem created by 1-NN (single nearest neighbor).

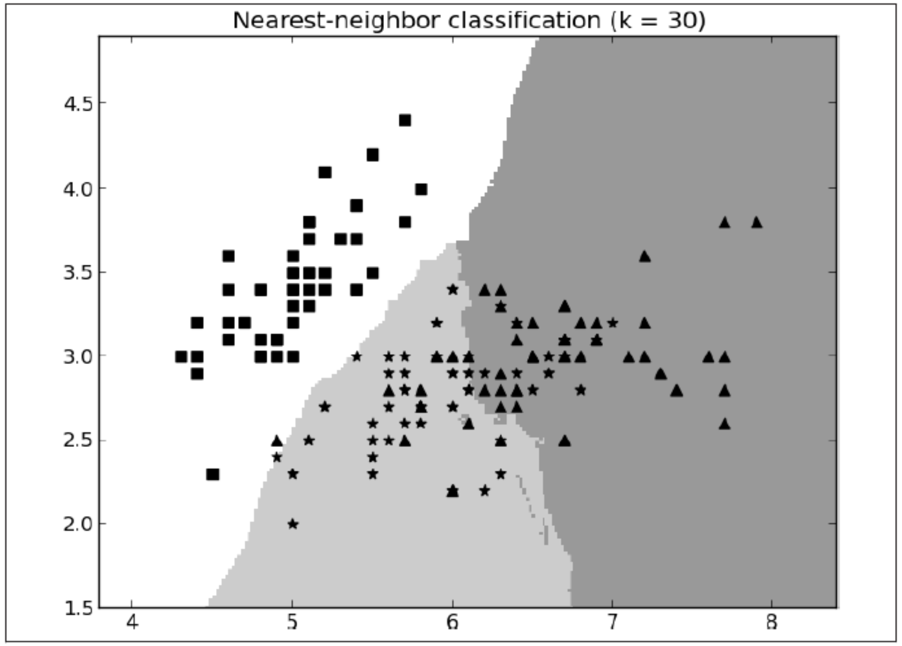


Figure 6-5. Classification boundaries created on a three-class problem created by 30-NN (averaging 30 nearest neighbors).

kNN models with a small *k*

- A finer granularity separation boundaries
- More susceptible to the presents of outlier.

kNN models with a large *k*

- More tolerant to noise
- Smooth but coarser separation boundaries

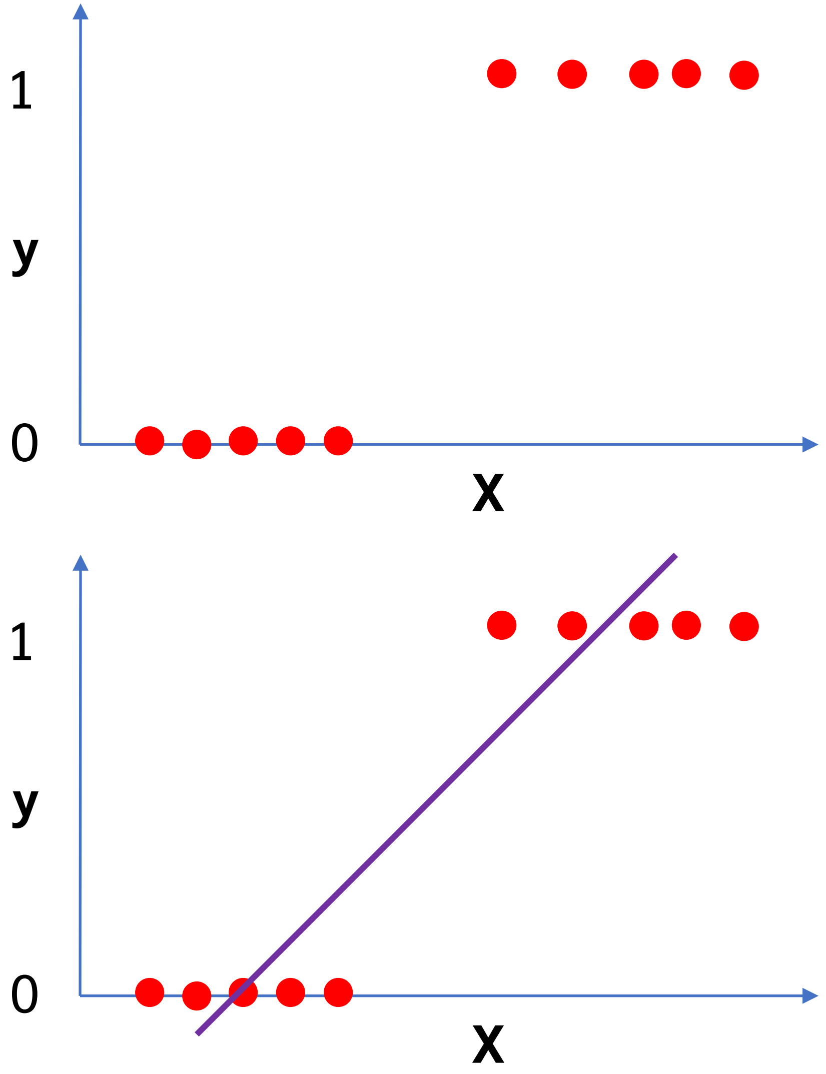
Figures from Data Science for Business by Foster Provost & Tom Fawcett

kNN Example

- Colab!

Logistic Regression

- Logistic regression is a binary (0/1) classification model, meaning it predict whether a data is of a certain class or not.
- **Motivation**: Let's say our data has classes (y=0 and y=1) as shown on the right. Can we use linear regression to predict them?
- While this looks like it can *somewhat* do the job, it is not appropriate
- We do not want the predicted output to be infinitely high/low. We simply want it to tell us whether the data is of our target type .
- Even better if it tells us how *confident* it is that the data is of our target type.

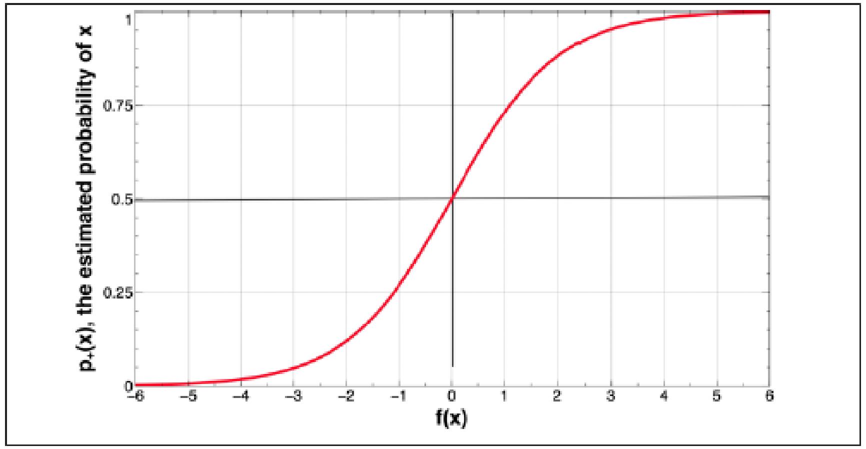


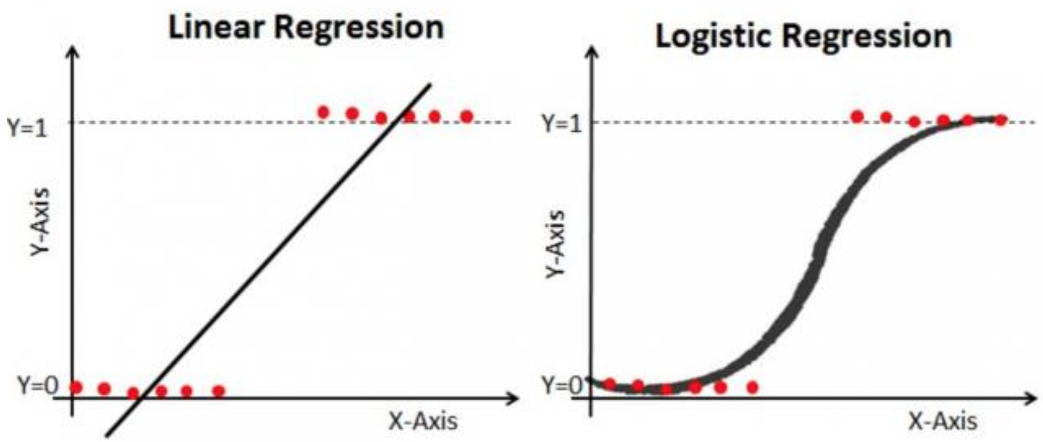
Logistic Regression

- A *sigmoid* function has a value ranging from 0 to 1 for its entire input range

$$y = \frac{1}{1 + e^{-x}}$$

- This makes it's a more appropriate choice to use compared to using purely linear equation





Figures from ODSC - Open Data Science (https://medium.com/@ODSC/logistic-regression-with-python-ede39f8573c7) and
Data Science for Business by Foster Provost & Tom Fawcett

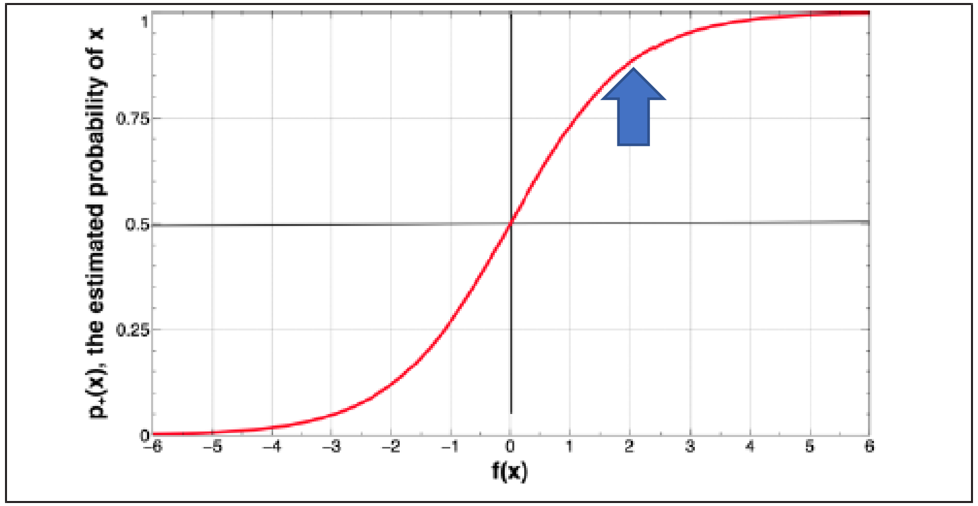
Logistic Regression

- Specifically, a logistic regression model uses a sigmoid function in conjunction with the linear equation.

$$y = \frac{1}{1 + e^{-f(x)}}$$

where

$$f(x) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_m x_m$$



- Fitting a logistic equation is akin to fitting a linear equation model which will then predict where we are on the input of the sigmoid function.
- This is similar to predicting how confidence we are of our prediction

Figure from ODSC - Open Data Science (https://medium.com/@ODSC/logistic-regression-with-python-ede39f8573c7)

Logistic Loss Function

- Since the goal for a logistic regression is to classify the data, its loss function needs to minimize the misclassification
    - Needs to have high value when predict incorrectly and low when predict correctly
- For example, we want a function that behaves like
    - $-\log(\hat{y}^{(i)})$ when $y^{(i)} = 1$
    - $-\log(1 - \hat{y}^{(i)})$ when $y^{(i)} = 0$

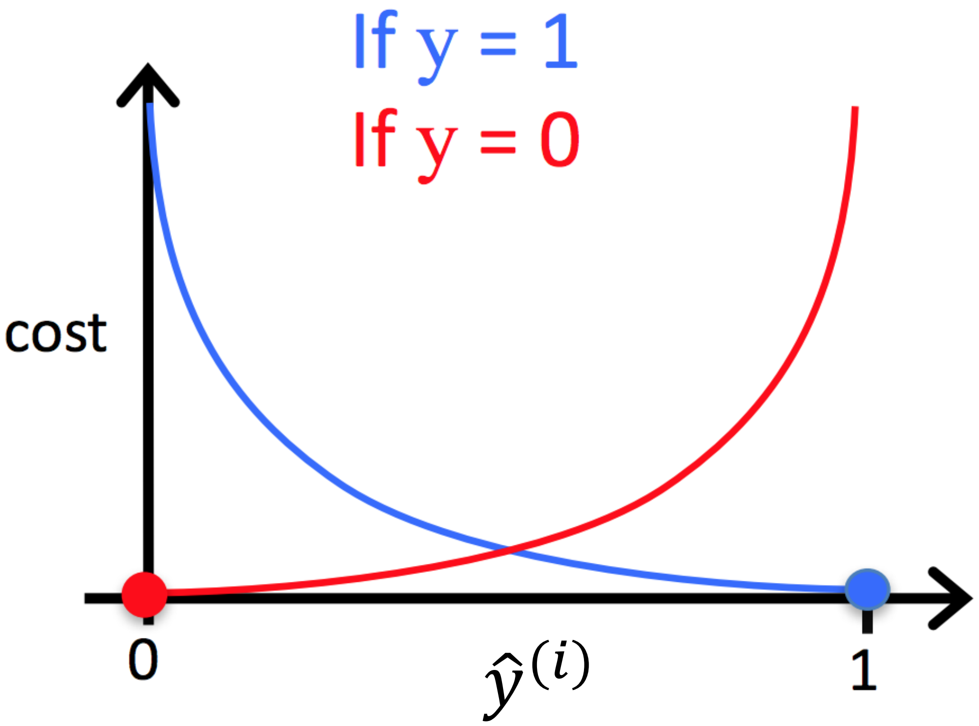


Figure from TwinPenguins (https://datascience.stackexchange.com/questions/40982/logistic-regression-cost-function)

Logistic Loss Function

- The logistic loss function is

$$Logistic\ Loss\ = -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\log\left(\hat{y}^{(i)}\right) + \left(1 - y^{(i)}\right)\log\left(1 - \hat{y}^{(i)}\right)\right)$$

where

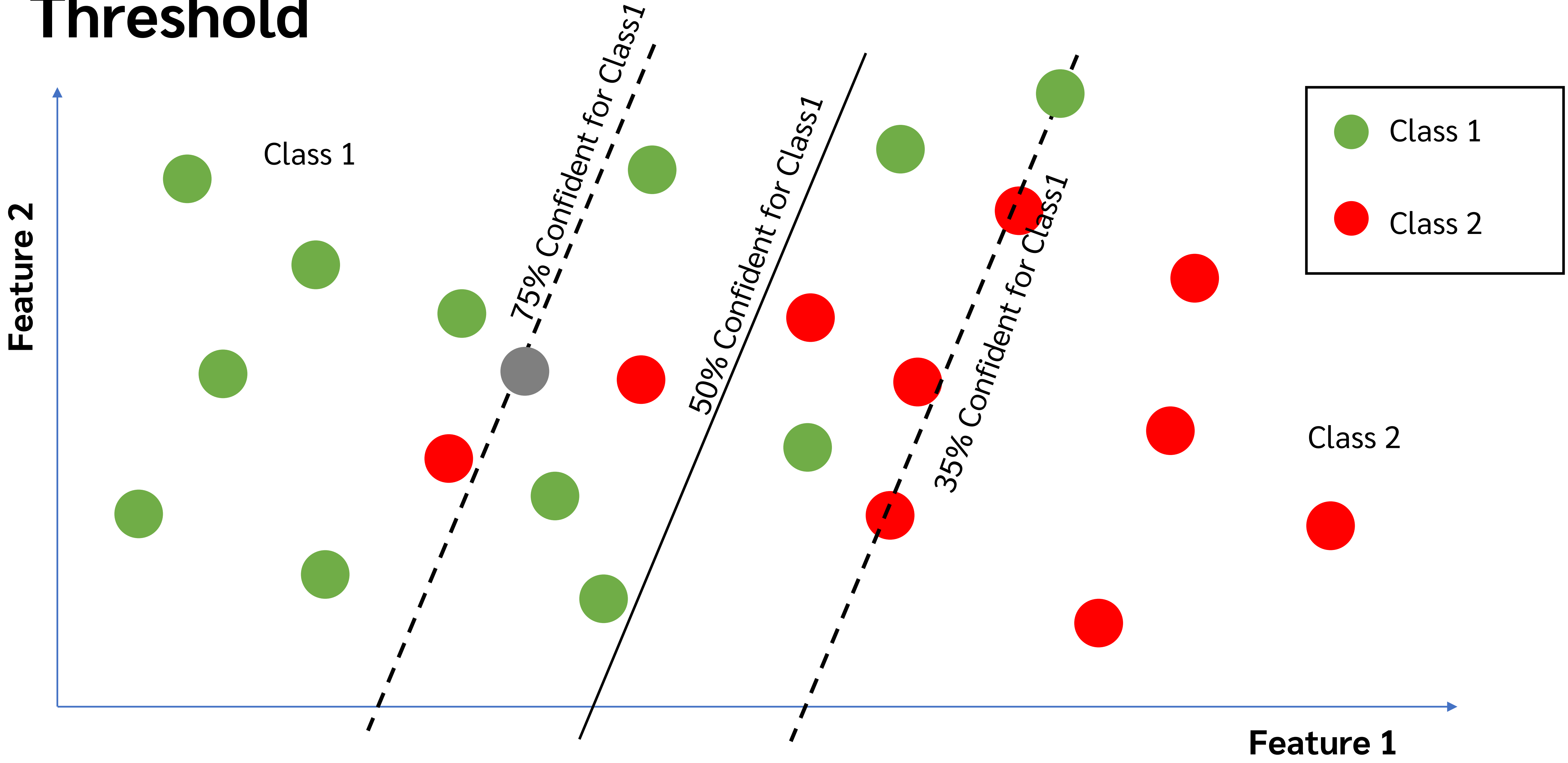
$$\hat{y}^{(i)} = \frac{1}{1 + e^{-f\left(x^{(i)}\right)}}$$

and

$$f\left(x^{(i)}\right) = \ w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)} + \cdots + w_m x_m^{(i)}$$

- Optimization technique like gradient descent can then be applied to find the optimal set of parameters
- Also note that Logistic regression has linear classification boundary

Logistic Regression: Separation Boundary and Threshold



Logistic Regression

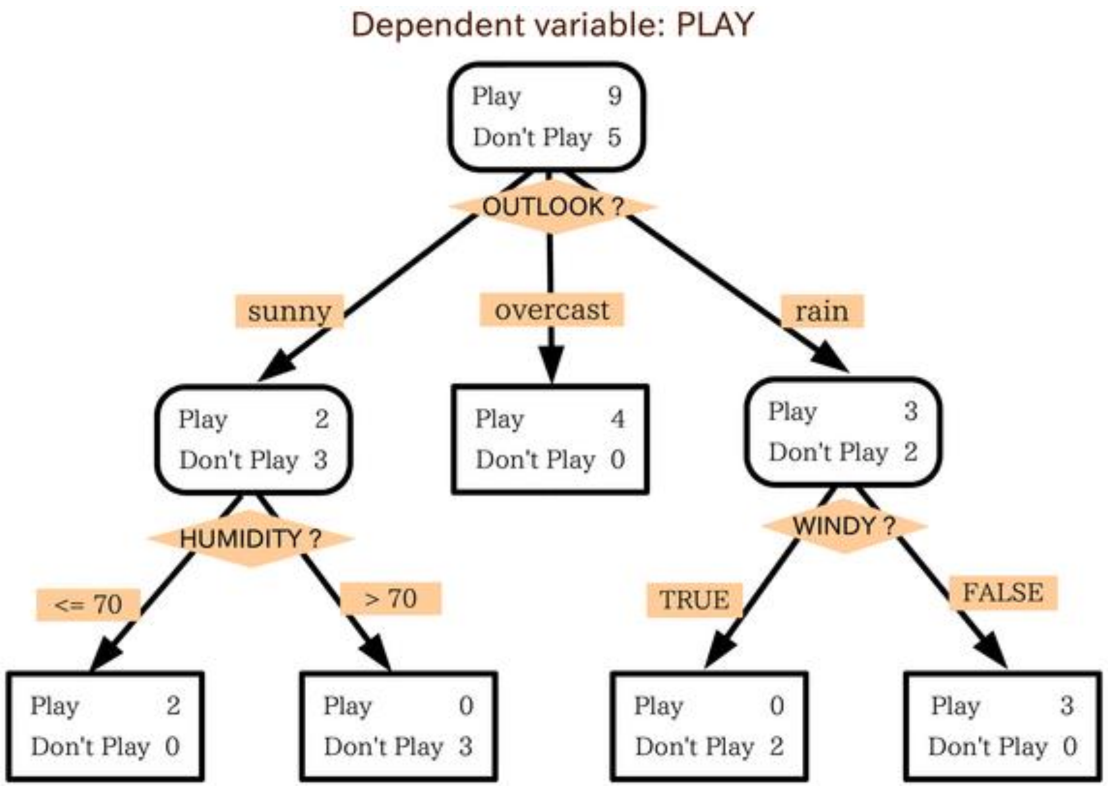
- One of the biggest advantage of logistic regression model is its explainability
- Similar to linear regression, the coefficients obtained for each feature in the final equation of the logistic regression are indicators of the direction (positive/negative) and strength of the relationships of each feature and the prediction target
    - Furthermore, this allows us to explain the expected consequence when changing the value of a feature.
- The coefficients obtained can also be used to perform feature selection by weeding out features with small coefficients.
- Also just like linear regression, regularization techniques can also be applied to improve the model's performance (in fact, regularization is applied by default in sklearn library)

Logistic Regression Example

- Colab!

Decision Tree

- A decision tree is a tree-like structure where the internal nodes represents series of “decisions” which eventually leads to an outcome (represented by the terminal nodes)
- Each node within a tree (apart from the final level) is a *segmentation criteria*.
- The leaf node (final level) represent the result of the classification. The label (classification decision) for data in each leaf node is made based on the content of the node’s data



Information Entropy

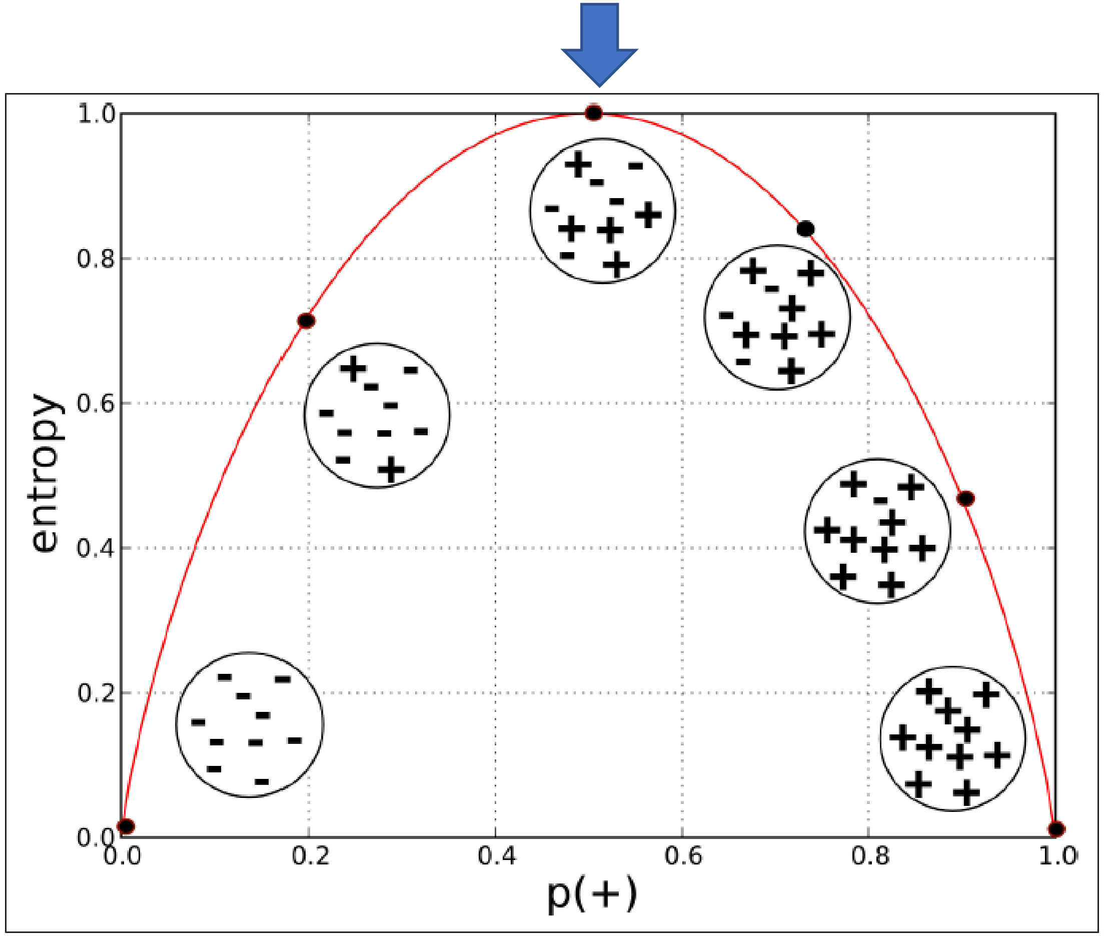


Figure 3-3. Entropy of a two-class set as a function of p(+).

- In thermodynamics, entropy represents a level of disorder within a system.
- Similarly, information entropy measure the average level of uncertainty for a random variable. It measure the *purity* (homogeneity) level of a group of data.
- Information entropy is calculated by the formula:

$$entropy = -p_1 \log(p_1) - p_2 \log(p_2) - \cdots$$

$$= \sum_{i=1}^{n} -p_i \log(p_i)$$

- For 2-classes data, the highest entropy occurs when the data contains samples of both classes *in an equal amount* as the expected value of a class is at the most uncertain state here.

Entropy

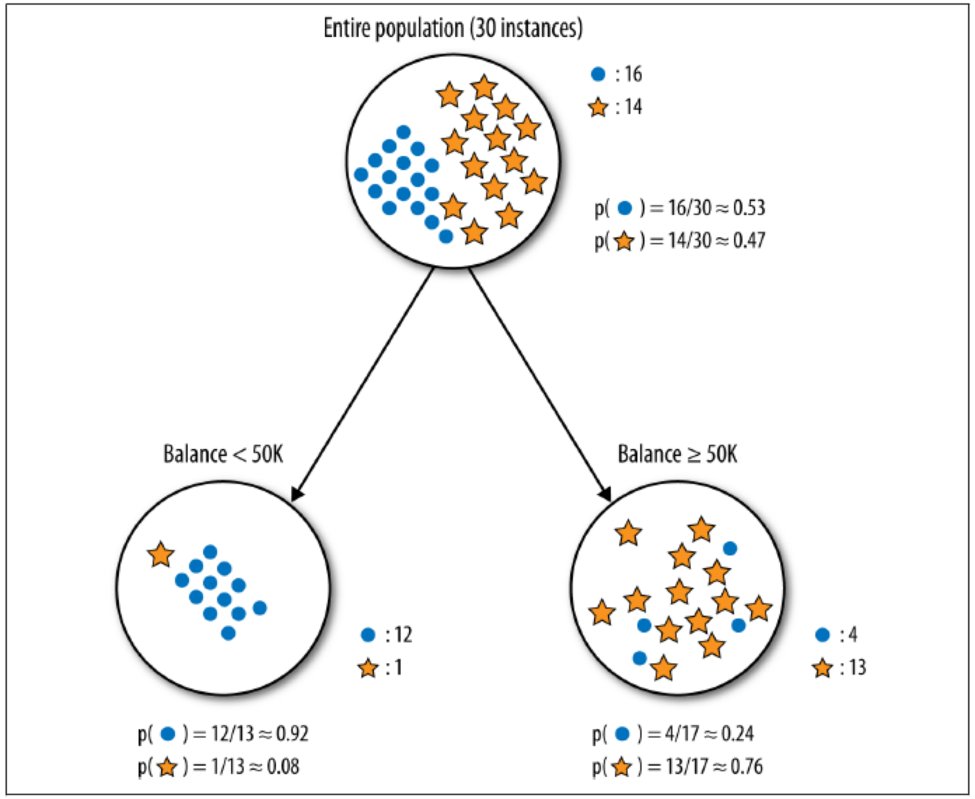


Figure 3-4. Splitting the "write-off" sample into two segments, based on splitting the Balance attribute (account balance) at 50K.

- The entropy of the *parent* (entire population) is

$$\begin{aligned} entropy(parent) &= -p(\bullet) \times \log_2 p(\bullet) - p(☆) \times \log_2 p(☆) \\ &\approx -0.53 \times -0.9 - 0.47 \times -1.1 \\ &\approx 0.99 \text{ (very impure)} \end{aligned}$$

- The entropy of the *left* child is:

$$\begin{aligned} entropy(Balance < 50K) &= -p(\bullet) \times \log_2 p(\bullet) - p(☆) \times \log_2 p(☆) \\ &\approx -0.92 \times (-0.12) - 0.08 \times (-3.7) \\ &\approx 0.39 \end{aligned}$$

- The entropy of the *right* child is:

$$\begin{aligned} entropy(Balance \geq 50K) &= -p(\bullet) \times \log_2 p(\bullet) - p(☆) \times \log_2 p(☆) \\ &\approx -0.24 \times (-2.1) - 0.76 \times (-0.39) \\ &\approx 0.79 \end{aligned}$$

Selecting the Best Split: Information Gain

- The most common splitting criterion is called an *information gain (IG)*
- We would like the split to be *informative*, meaning that it provide us with more information about our classification target. Therefore, we would like for the overall level of entropy to decrease after the segmentation
- We call the reduction in entropy level (the level of disorder) an *information gain (IG)*
  - i.e. Information Gain = total entropy before split - total entropy after split
- In other words ,

$$IG = entropy(parent) - p(c_1) \times entropy\ (c_1) - p(c_2) \times entropy(c_2) - \cdots$$

$$= entropy(parent) - \sum_i p(c_i) \times entropy\ (c_i)$$

- We pick the segmentation criteria to be the one that has the highest information gain.

Entropy

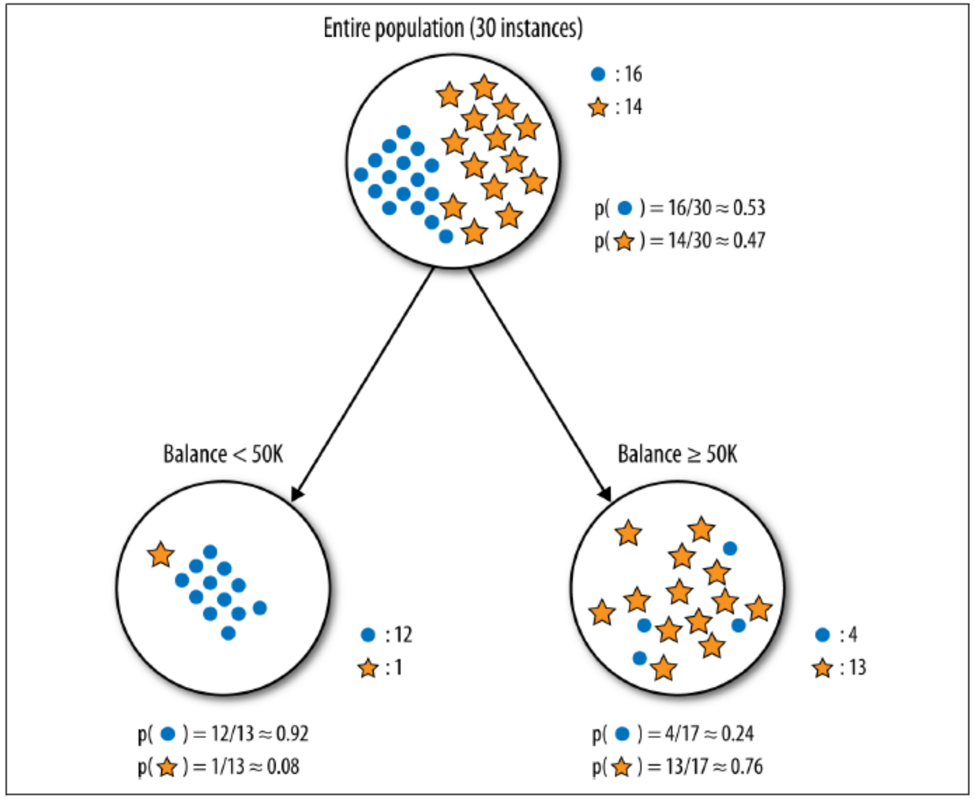


Figure 3-4. Splitting the "write-off" sample into two segments, based on splitting the Balance attribute (account balance) at 50K.

- The entropy of the *parent* is 0.99
- The entropy of the *left* child is 0.39
- The entropy of the *right* child is 0.79
- The Information Gain (IG) is:

$$IG = entropy(parent)$$
$$-[p(Balance < 50K) \times entropy(Balance < 50K)$$
$$+ p(Balance \geq 50K) \times entropy(Balance \geq 50K)]$$
$$\approx 0.99 - (0.43 \times 0.39 + 0.57 \times 0.79)$$
$$\approx 0.37$$

Entropy

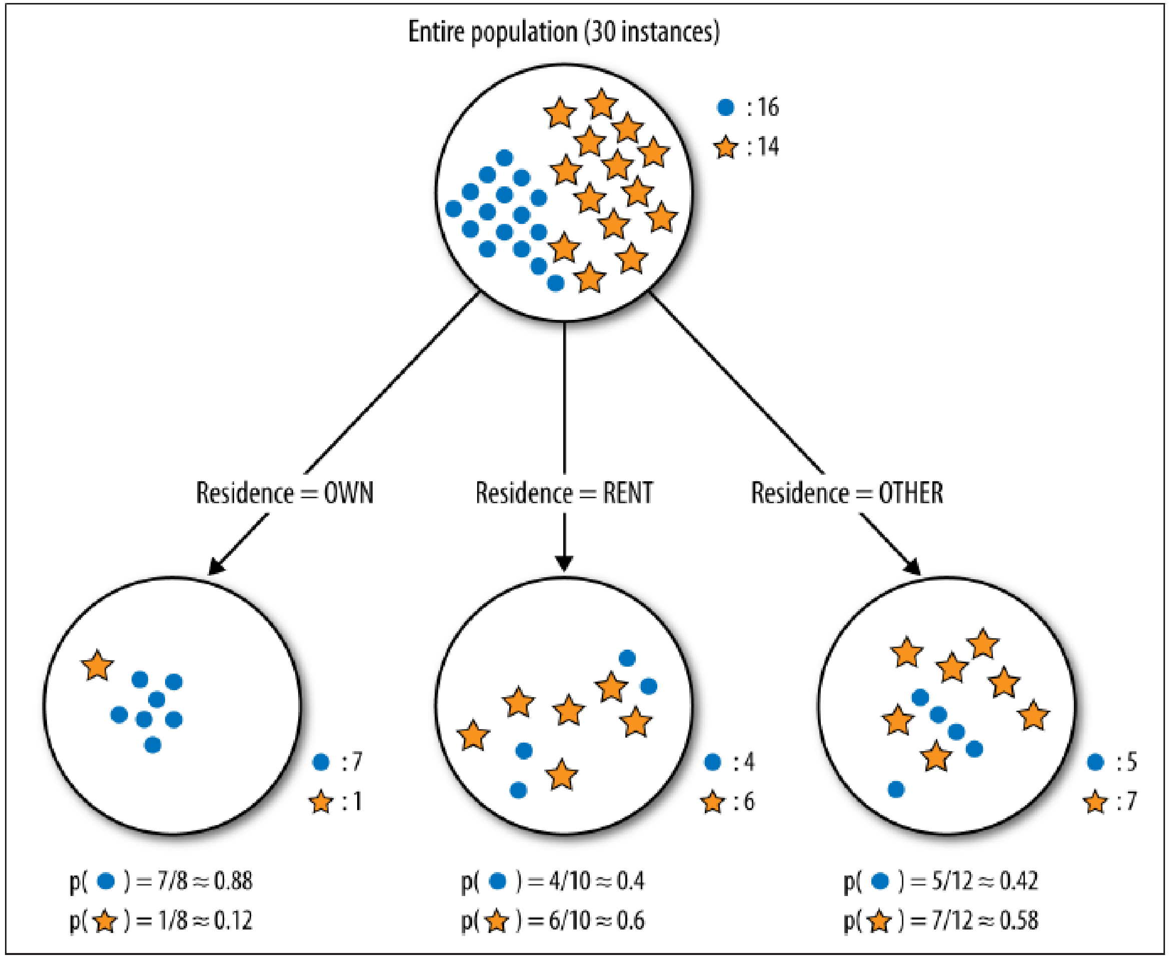


Figure 3-5. A classification tree split on the three-valued Residence attribute.

Figures and Contents from Data Science for Business by Foster Provost & Tom Fawcett

- $entropy(parent) \approx 0.99$
- $entropy(Residence = OWN) \approx 0.54$
- $entropy(Residence = RENT) \approx 0.97$
- $entropy(Residence = OTHER) \approx 0.98$
- $Information\ Gain\ (IG) \approx 0.13$

Decision Tree Example: Data and Attributes

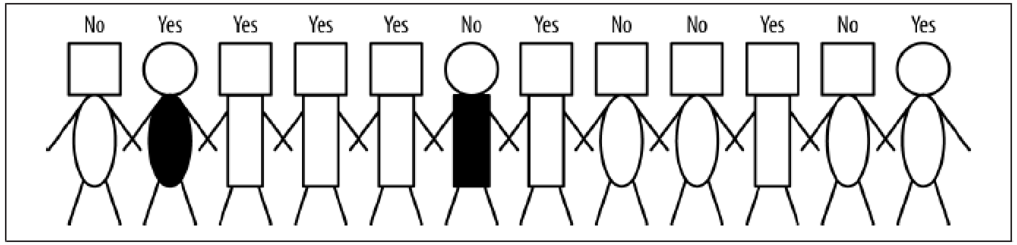


Figure 3-2. A set of people to be classified. The label over each head represents the value of the target variable (write-off or not). Colors and shapes represent different predictor attributes.

What are attributes that describes this data (people)?

Attributes:

- head-shape: square, circular
- body-shape: rectangular, oval
- body-color: gray, white

Target variable:

- write-off: Yes, No

What is the most informative attribute that can be used to segment this data (people)?

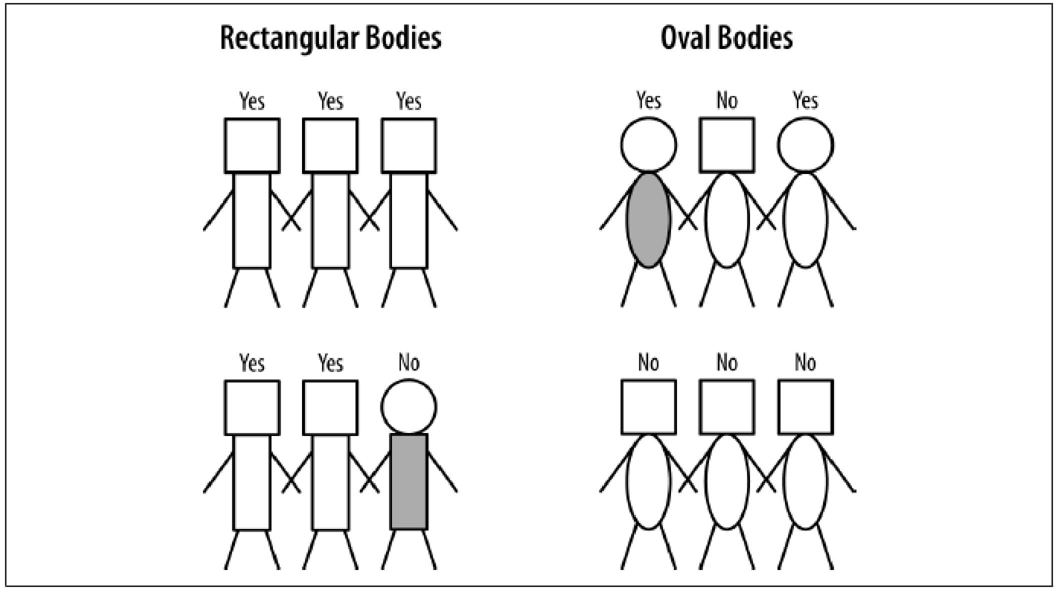


Figure 3-11. First partitioning: splitting on body shape (rectangular versus oval).

Decision Tree Example: Further Segmentation

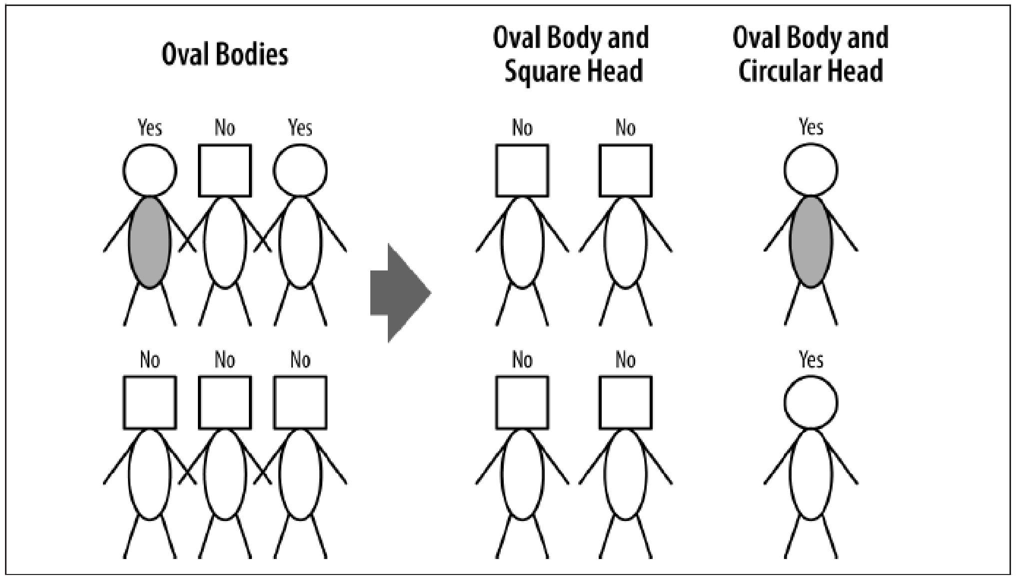


Figure 3-12. Second partitioning: the oval body people sub-grouped by head type.

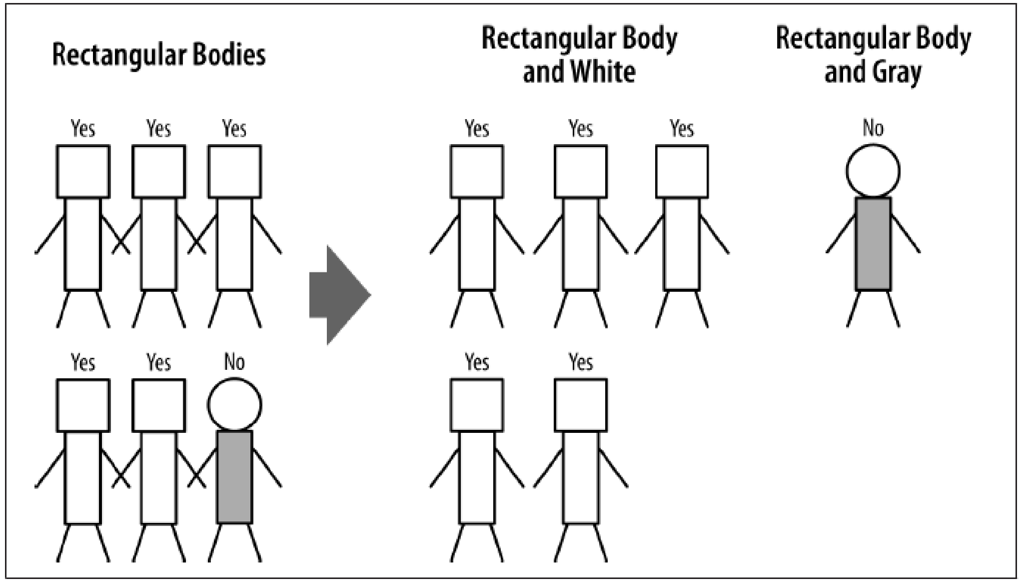


Figure 3-13. Third partitioning: the rectangular body people subgrouped by body color.

Figures and Contents from Data Science for Business by Foster Provost & Tom Fawcett

Decision Tree Example: Overall Decision Tree

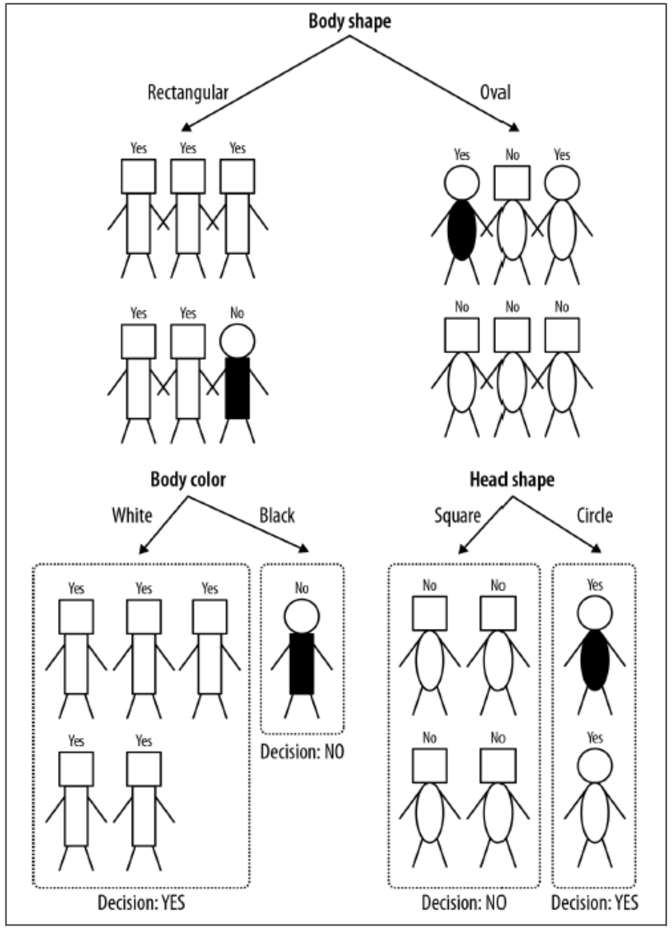
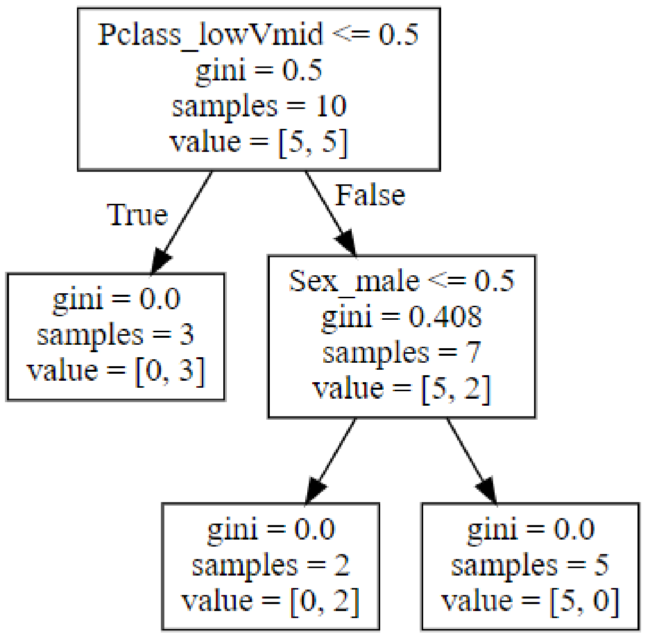


Figure 3-14. The classification tree resulting from the splits done in Figure 3-11 to Figure 3-13.

Note on Decision Tree



$$Gini\ Index = 1 - \sum_{i=1}^{C} (p_i)^2$$

- Decision Tree classification algorithm presented is a *greedy* algorithm.
  - This means it always choose the feature that best segmented the data at any given point.
  - This algorithm can result in sub-optimal solution.
- Another popular criterion used for deciding how to split data at each node of a decision tree is the *Gini impurity*
- Gini impurity is a measurement of the probability that an incorrectly labeled classification will be given to a randomly chosen element from a set if the said element is randomly labeled by the distribution of labels in that set. We want a variable split to have a low Gini index.

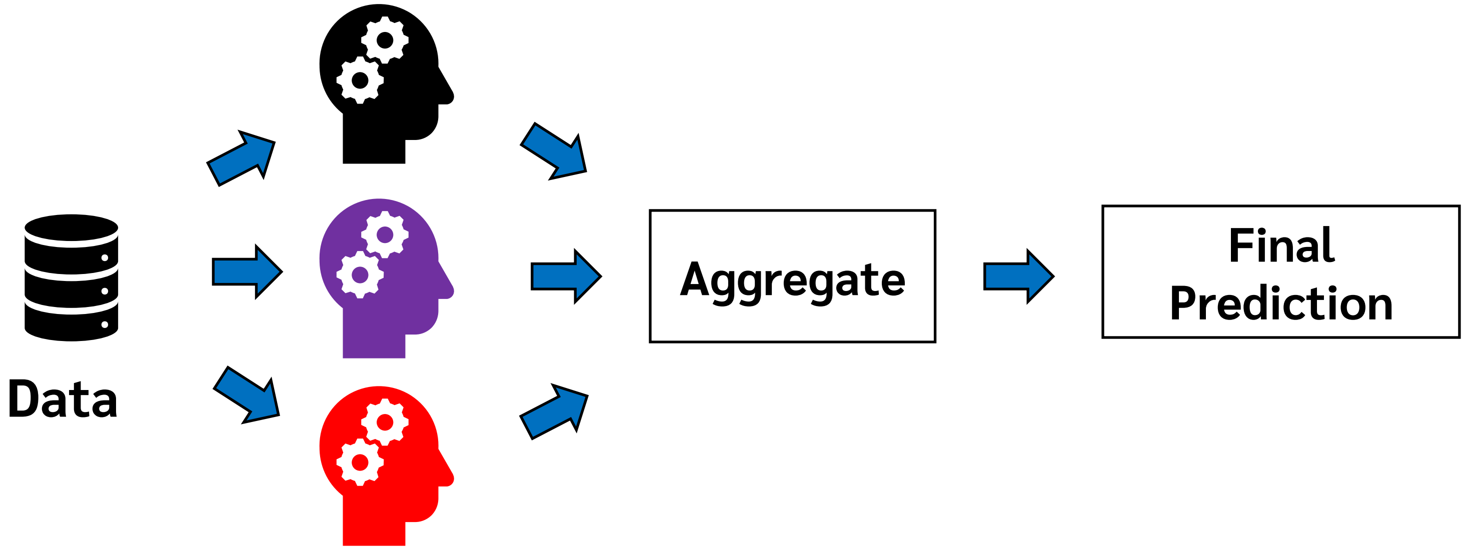
Figure from vivian (https://stackoverflow.com/questions/53699121/how-do-i-get-all-gini-indices-in-my-decision-tree)

Decision Tree Example

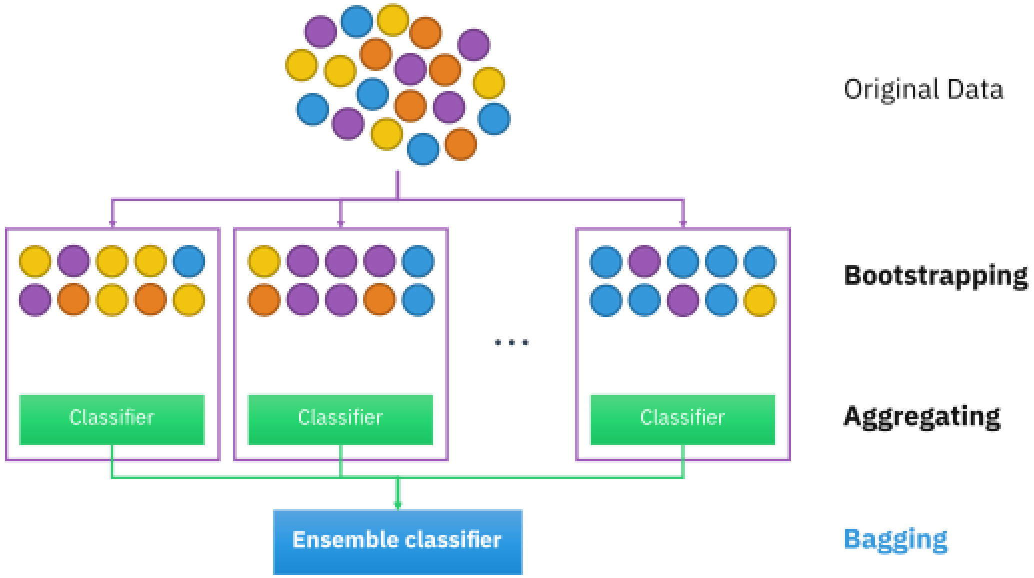
- Colab!

Ensemble Learning

- Ensemble learning is a process which utilizes multiple machine learning models to learn a particular task, then combines their results to make the final prediction.
- For example, an ensemble classifier may take result of its multiple classifiers then deciding the final prediction via voting or some other means of aggregations
- Random Forest and XGBoost models are examples of popular ensembles models



Bagging



- From the original data, repeatedly perform bootstrapping (taking random sampling with replacement) and create multiple sets of bootstrap samples
- Train multiple distinct classifiers on each set of bootstrap samples
- Uses all classifiers to predict a new input by voting / averaging
- Helps making the model more generalized

Figure from Sirakorn (https://commons.wikimedia.org/wiki/File:Ensemble_Bagging.svg)
Contents from KMUTT Big Data Experience Center’s lecture on Ensemble Learning

Random forest

- An ensemble model consists of multiple decision trees, each of which are trained by samples from bootstrap sampling with only a random subset features available.
  - With large number of trees, the model is less likely to overfit.
  - With only a random subset of features available for each tree, it is less likely for trees' results to be highly correlated.
- Each of decision trees are built and train in a greedy fashion using conditional entropy
- The final prediction is decided by average predictions of all trees.

Content from KMUTT Big Data Experience Center's lecture on Ensemble Learning and
Jason Brownlee (https://machinelearningmastery.com/bagging-and-random-forest-ensemble-algorithms-for-machine-learning/)

Random forest

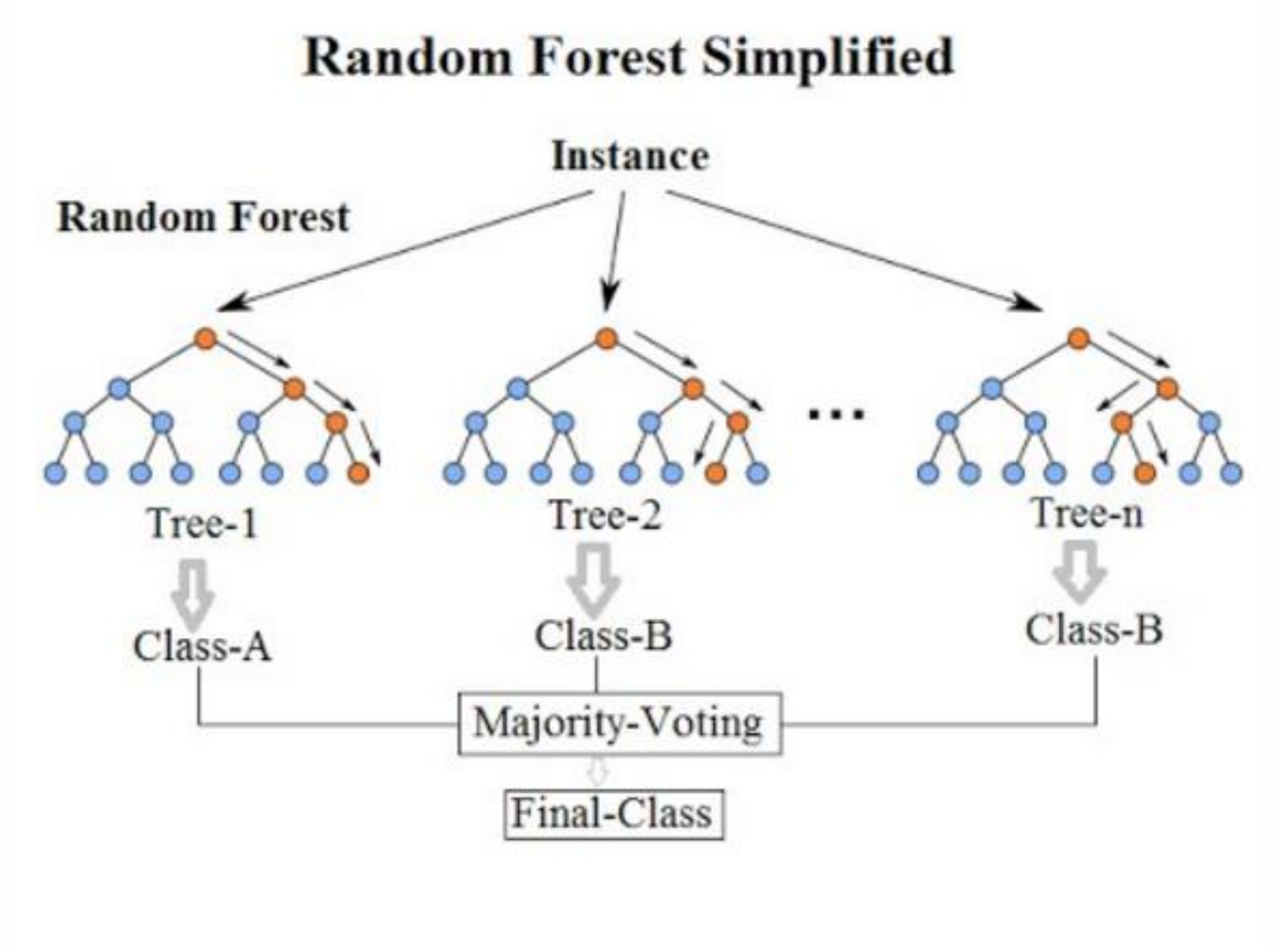


Figure from Venkata Jagannath (https://commons.wikimedia.org/wiki/File:Random_forest_diagram_complete.png)

Boosting

- Aims to improve the performance of the model by allowing it to fit the training data better
- Unlike bagging, it creates a sequence of classifiers and gives higher influence to a more accurate classifiers
- For each iteration, it makes examples currently misclassified more important (or less, in some cases) by giving larger weights in in the construction of the next classifiers.
- The final prediction is decided by combining the results of classifiers by weighted vote (weight given by classifier accuracy)

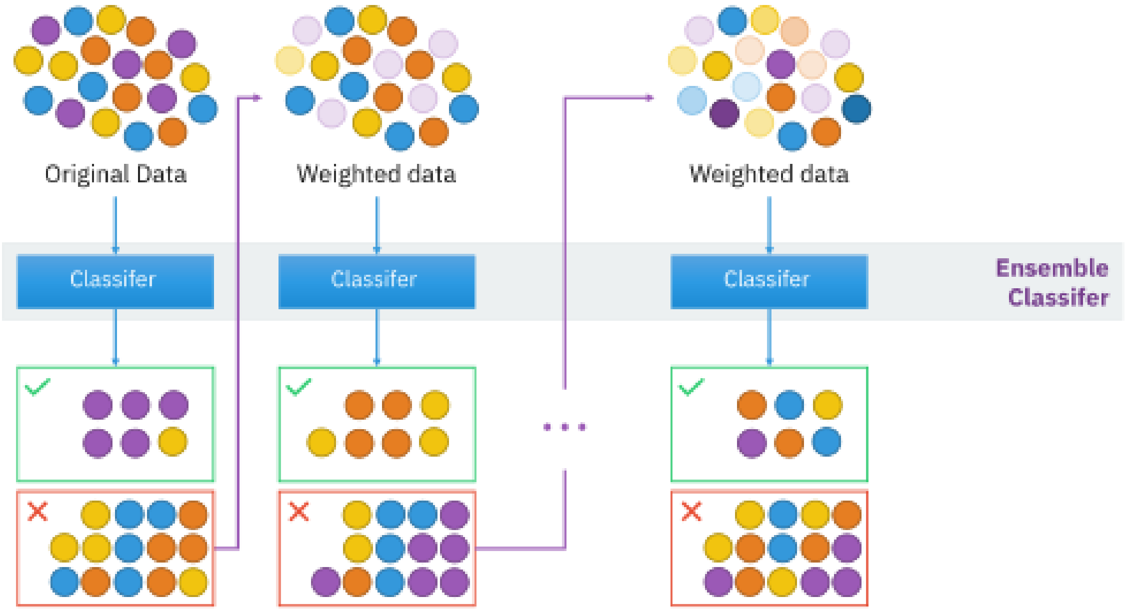


Figure from Sirakorn (https://commons.wikimedia.org/wiki/File:Ensemble_Boosting.svg)
Contents from KMUTT Big Data Experience Center's lecture on Ensemble Learning

Extreme Gradient Boosting (XGBoost)

- A scalable and portable and accurate implementation of gradient boosting machines
- One of the most popular models. Perform well across variety of applications.

Model Feature

- **Gradient Boosting** algorithm also called gradient boosting machine including the learning rate.
- **Stochastic Gradient Boosting** with sub-sampling at the row, column and column per split levels.
- **Regularized Gradient Boosting** with both L1 and L2 regularization.

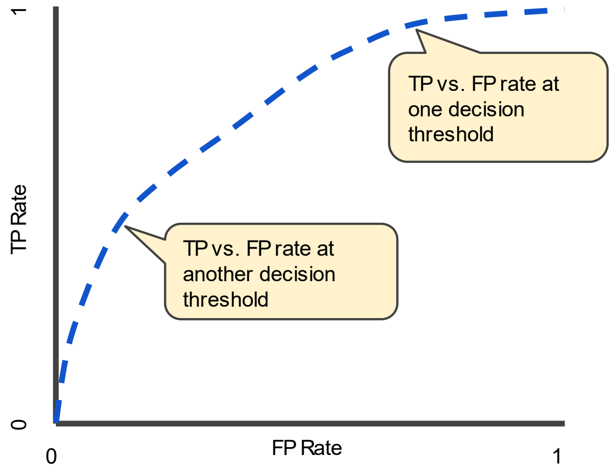
Model Selections: ROC curve and AUC

- The ROC (Receiver Operating Characteristic) curve is a plot that shows performance of a model at different classification threshold

- ROC Curves plots Recall vs. Sensitivity, essentially comparing the “hit rate” and the “false alarm rate”

$$True\ Positive\ Rate\ (Recall) = \frac{TP}{TP + FN}$$

$$False\ Positive\ Rate\ (Sensitivity) = \frac{FP}{FP + TN}$$

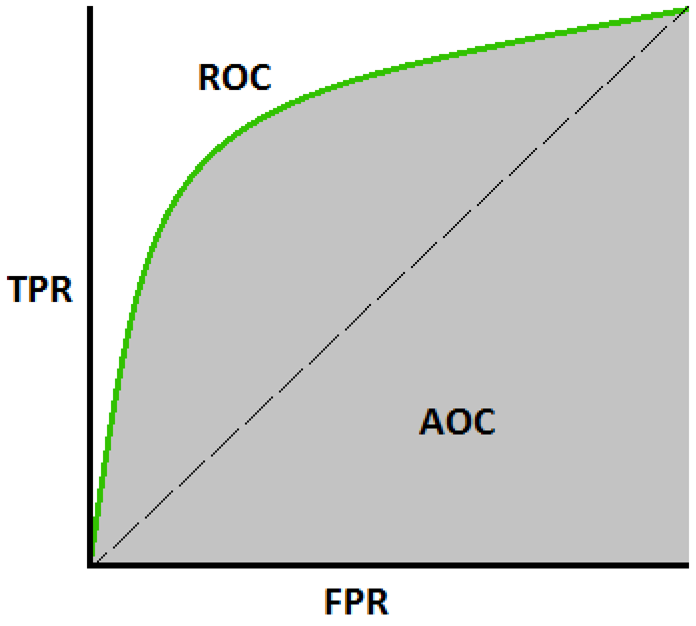


True Label

Predicted Label	Positive	Negative
Positive	True Positive	False Positive
Negative	False Negative	True Negative

Contents from Google’s Machine Learning Crash Course (https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc)

Model Selections: ROC curve and AUC



- AUC (Area Under the ROC Curve) provides a metric for considering the models performance across all classification threshold.

- *Side note: so far, we have been using a classification threshold of 0.5. However, it is also possible (and sometimes beneficial) for a classifier to vary.*

Contents from Google's Machine Learning Crash Course (https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc)
Figure from Sarang Narkhede (https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5)

Random Forest & XGBoost (& Model Selection) Example

- Colab!

Thank You