Machine Learning Algorithms: I

semicolon

Learning Outcomes

- Participants should be able to understand the concept behind different machine learning algorithms and how to implement them

Introduction

- Machine learning is a set of methods that can automatically detect patterns in data.
  These uncovered patterns are then used to predict future data, or to perform other kinds of decision-making under uncertainty.
  The key premise is learning from data!!

Major Types of Machine Learning Models

- ***Supervised learning (SML)*****,** the learning algorithm is presented with labelled example inputs, where the labels indicate the desired output. SML itself is composed of classification, where the output is categorical, and regression, where the output is numerical

Examples:

- Decision Trees
- Linear Regression
- Support Vector Machines (SVM)
- Logistic Regression

Major Types of Machine Learning Models (Contd)

- ***Unsupervised learning (UML)***, no labels are provided, and the learning algorithm focuses solely on detecting structure in unlabeled input data

- Examples
- k-means clustering
- Association Rules

Major Types of Machine Learning Models (Contd)

- **Semi-supervised Learning**
- In the absence of labels most of the observations but present in few, semi-supervised algorithms can be suitable for such data.

- **Reinforcement Learning**
- Reinforced ML uses the technique called exploration/exploitation. The processes are simple - the action takes place, the consequences are observed, and the next action considers the results of the first action

Model Performance

- For regression, we will use the root mean squared error (RMSE), which is what linear regression (lm in R) seeks to minimise. For classification, we will use model prediction accuracy.
  Typically, we won't want to calculate any of these metrics using observations that were also used to calculate the model. This approach, called in-sample error leads to optimistic assessment of our model. Indeed, the model has already seen these data upon construction, and is considered optimized for these observations in particular; it is said to over-fit the data. We prefer to calculate an out-of-sample error, on new data, to gain a better idea of how to model performs on unseen data, and estimate how well the model generalizes.

Cross Validation

- Instead of doing a single training/testing split, we can systematise this process, produce multiple, different out-of-sample train/test splits, that will lead to a better estimate of the out-of-sample RMSE.
  We split the data into 3 random and complementary folds, so that each data point appears exactly once in each fold. This leads to a total test set size that is identical to the size of the full dataset but is composed of out-of-sample predictions

Cross Validation

- The procedure of creating folds and training the models is handled by the train function in caret

Confusion matrix

- A confusion matrix contains information about actual and predicted classifications done by a classification system. Performance of such systems is commonly evaluated using the data in the matrix. Confusion Matrix is often used to evaluate the performance of a model with binary responses or multiclass responses.
- The entries in the confusion matrix have the following meaning in the context of our study:

Confusion matrix

		Predicted	
Actual		Negative	Positive
	Negative	A	B
	Positive	C	D

- **A** is the number of correct predictions that an instance is negative,
- **B** is the number of incorrect predictions that an instance is positive,
- **C** is the number of incorrect of predictions that an instance negative, and
- **D** is the number of correct predictions that an instance is positive.

Confusion Matrix

- In confusion matrix, Correct results are true positives (TP) and true negatives (TN) are found along the diagonal. All other cells indicate false results, i.e false negatives (FN) and false positives (FP)

- Using the confusion matrix, there are certain metrics that should be taken into consideration when evaluating the models to select the best machine learning model.
- The most commonly used metric is the accuracy which is obtained by dividing the total number of correct classifications by the total number of observations

Model Evaluation Metrics

- $Accuracy = \frac{TP+TN}{TP+TN+FN+FP}$
- $Specificity = \frac{TN}{FP+TN}$
- $Sensitivity = \frac{TP}{TP+FN}$
- $F-Score = \frac{(1+\beta^2)TP}{(1+\beta^2)TP+\beta^2 FN+FP}$
- Where β is the weighting parameter. Both the precision and sensitivity are equally relevant and therefore the weight is set to β = 1

Featue Selection

- Feature selection can be regarded as variable selection or attribute selection. It is the selection of features or attributes in the data (such as columns in tabular data) that are very relevant to the predictive modelling problem we are working on. It is the process of selecting subset of relevant features for use in the model building.
- There different feature selection techniques, one of them is Recursive feature selection technique.

ML Models

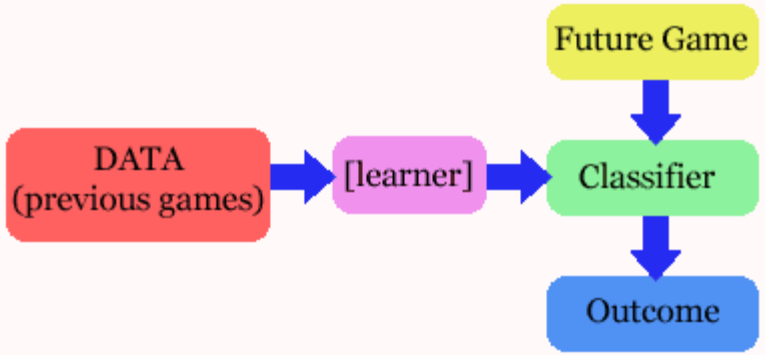
- In this course, we will be discussing the following ML Models

- Decision Trees
- Linear Regression
- Support Vector Machines (SVM)
- Logistic Regression

Both Linear and Logistic Regression have been discussed extensively in previous lessons. Review your previous lessons.

Decision Trees

- Given a set of training cases/objects and their attribute values, try to determine the target attribute value of new examples.
  - Classification
  - Prediction



Why decision tree?

- Decision trees are powerful and popular tools for classification and prediction
- The learning and classification steps of a decision tree are simple and fast
- Decision trees represent *rules*, which can be understood by humans

key requirements

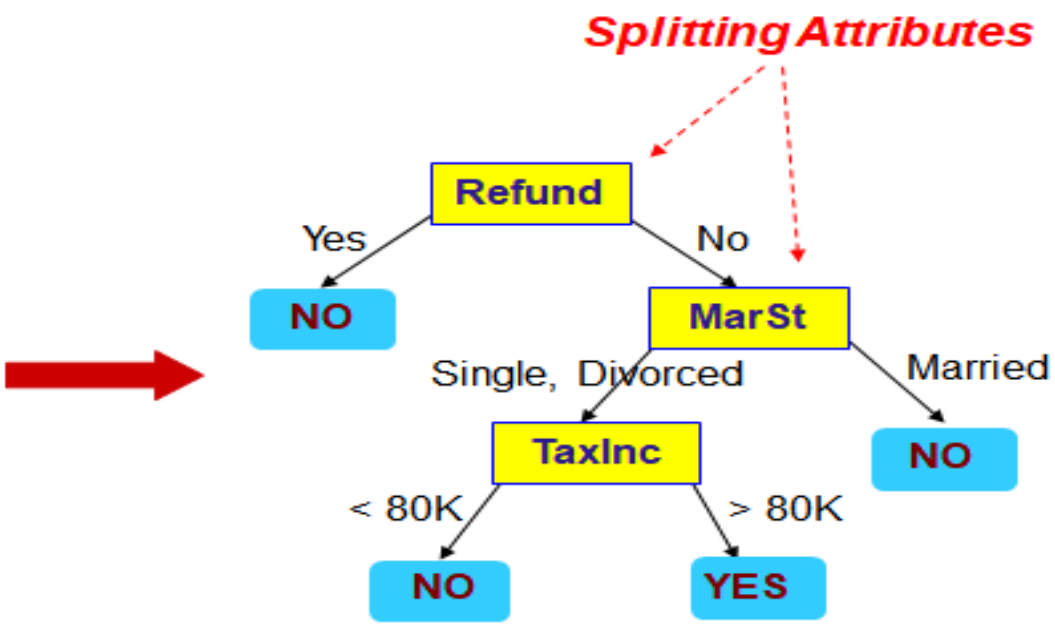
- **Attribute-value description:** object or case must be expressible in terms of a fixed collection of properties or attributes (e.g., hot, mild, cold).
- **Predefined classes (target values):** the target function has **discrete output values** (bollean or multiclass)
- **Sufficient data:** enough training cases should be provided to learn the model.

Example of a Decision Tree

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

categorical categorical continuous class

Training Data



Model: Decision Tree

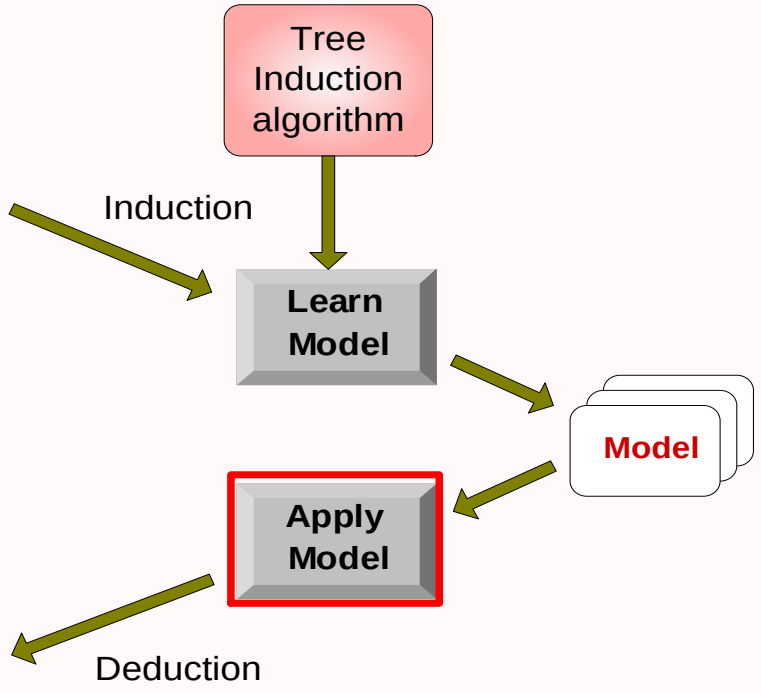
Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

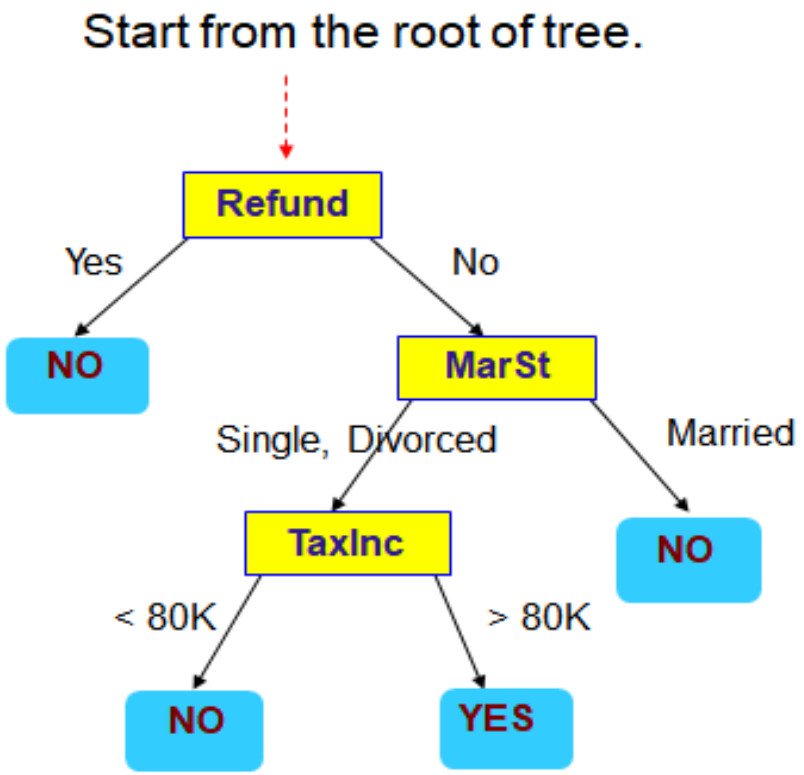
Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Apply Model to Test Data

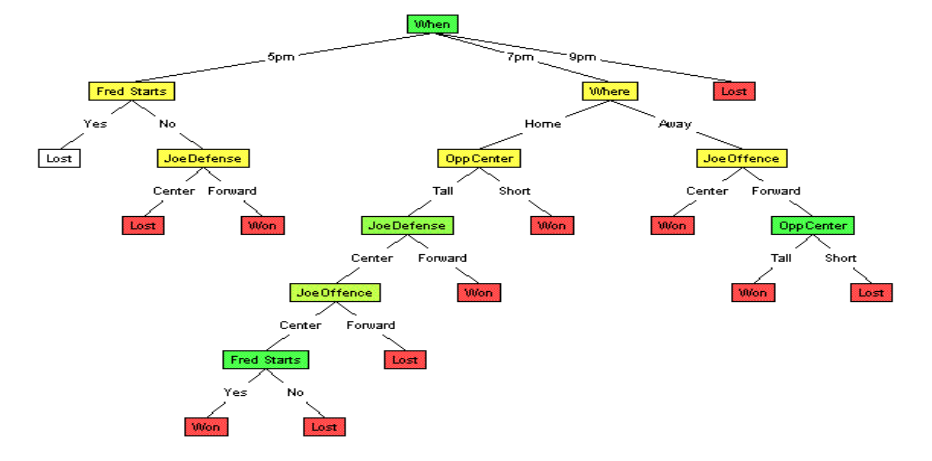


Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Random split

- The tree can grow huge
- These trees are hard to understand.
- Larger trees are typically less accurate than smaller trees.



semicolon

Principled Criterion

- Selection of an attribute to test at each node - choosing the most useful attribute for classifying examples.
- information gain
  - measures how well a given attribute separates the training examples according to their target classification
  - This measure is used to select among the candidate attributes at each step while growing the tree

Entropy

- A measure of homogeneity of the set of examples.
- Given a set S of positive and negative examples of some target concept (a 2-class problem), the entropy of set S relative to this binary classification is

$$E(S) = - p(P)\log_2 p(P) – p(N)\log_2 p(N)$$

- Suppose S has 25 examples, 15 positive and 10 negatives [15+, 10-]. Then the entropy of S relative to this classification is

$$E(S)=-(15/25)\log_2(15/25) - (10/25)\log_2 (10/25)$$

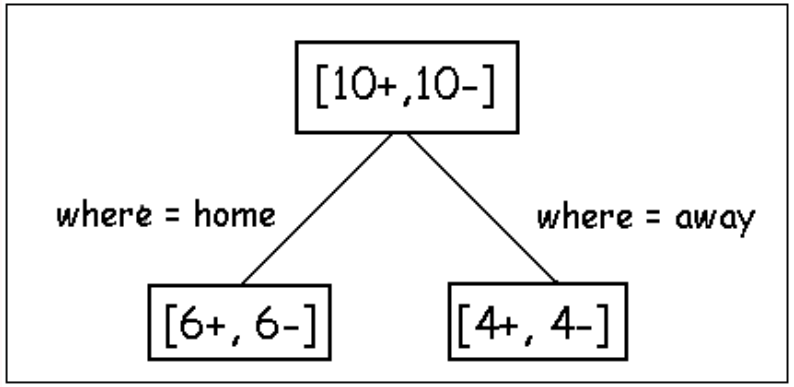
Identifying the Best Attributes

- Information gain measures the expected reduction in entropy, or uncertainty.

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

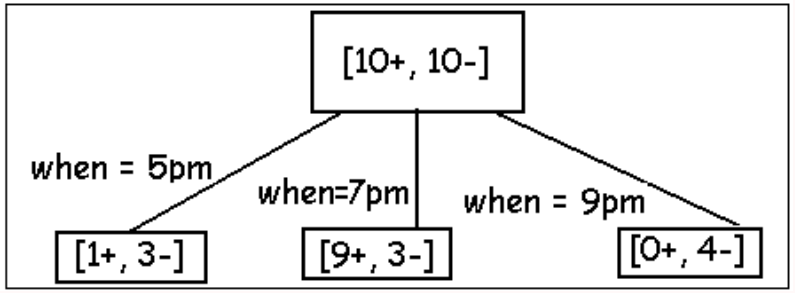
  - Values(A) is the set of all possible values for attribute A, and Sv the subset of S for which attribute A has value v Sv = {s in S | A(s) = v}.
  - the first term in the equation for *Gain* is just the entropy of the original collection *S*
  - the second term is the expected value of the entropy after S is partitioned using attribute A

Examples



- Before partitioning, the entropy is
  - *H(10/20, 10/20) = - 10/20 log(10/20) - 10/20 log(10/20) = 1*
- Using the ``where'' attribute, divide into 2 subsets
  - Entropy of the first set H(home) = - 6/12 log(6/12) - 6/12 log(6/12) = 1
  - Entropy of the second set H(away) = - 4/8 log(6/8) - 4/8 log(4/8) = 1
- Expected entropy after partitioning
  - 12/20 * H(home) + 8/20 * H(away) = 1

Examples



- Using the ``when'' attribute, divide into 3 subsets
  - Entropy of the first set H(5pm) = - 1/4 log(1/4) - 3/4 log(3/4);
  - Entropy of the second set H(7pm) = - 9/12 log(9/12) - 3/12 log(3/12);
  - Entropy of the second set H(9pm) = - 0/4 log(0/4) - 4/4 log(4/4) = 0
- Expected entropy after partitioning
  - 4/20 * H(1/4, 3/4) + 12/20 * H(9/12, 3/12) + 4/20 * H(0/4, 4/4) = 0.65
- Information gain 1-0.65 = 0.35

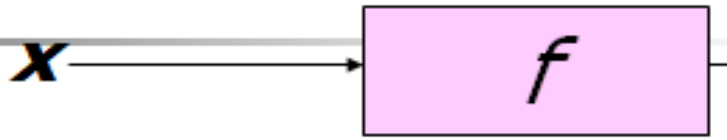
Decision

- Knowing the ``when'' attribute values provides larger information gain than ``where''.
- Therefore the ``when'' attribute should be chosen for testing prior to the ``where'' attribute.
- Similarly, we can compute the information gain for other attributes.
- At each node, choose the attribute with the largest information gain.
- **Stopping rule**
  - Every attribute has already been included along this path through the tree

Support Vectors Machine (SVM)

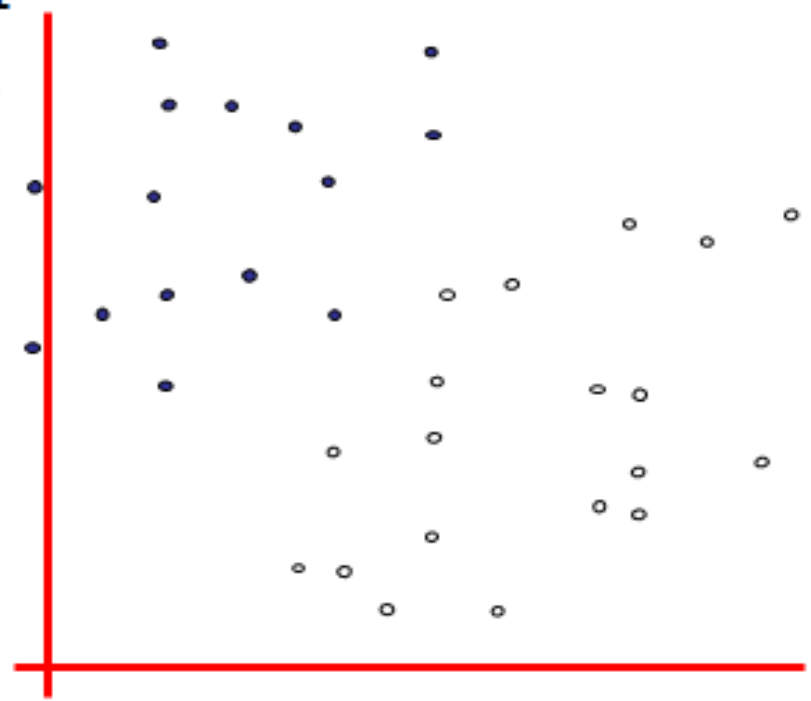
- **Support vector machines (SVM)** are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis.
- It is a **machine learning** approach.
- They analyze the large amount of data to identify patterns from them.
- SVMs are based on the idea of finding a hyperplane that best divides a dataset into two classes, as shown in the image below.

Linear Classifiers



$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\mathbf{x} - b)$

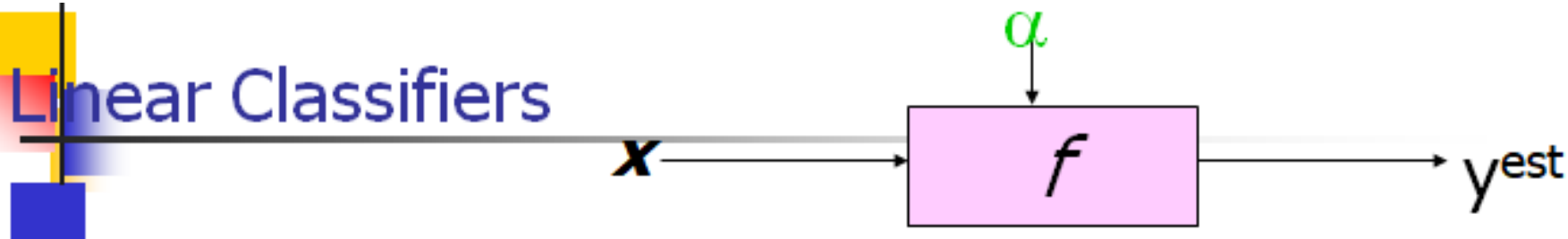
- denotes +1
- denotes -1

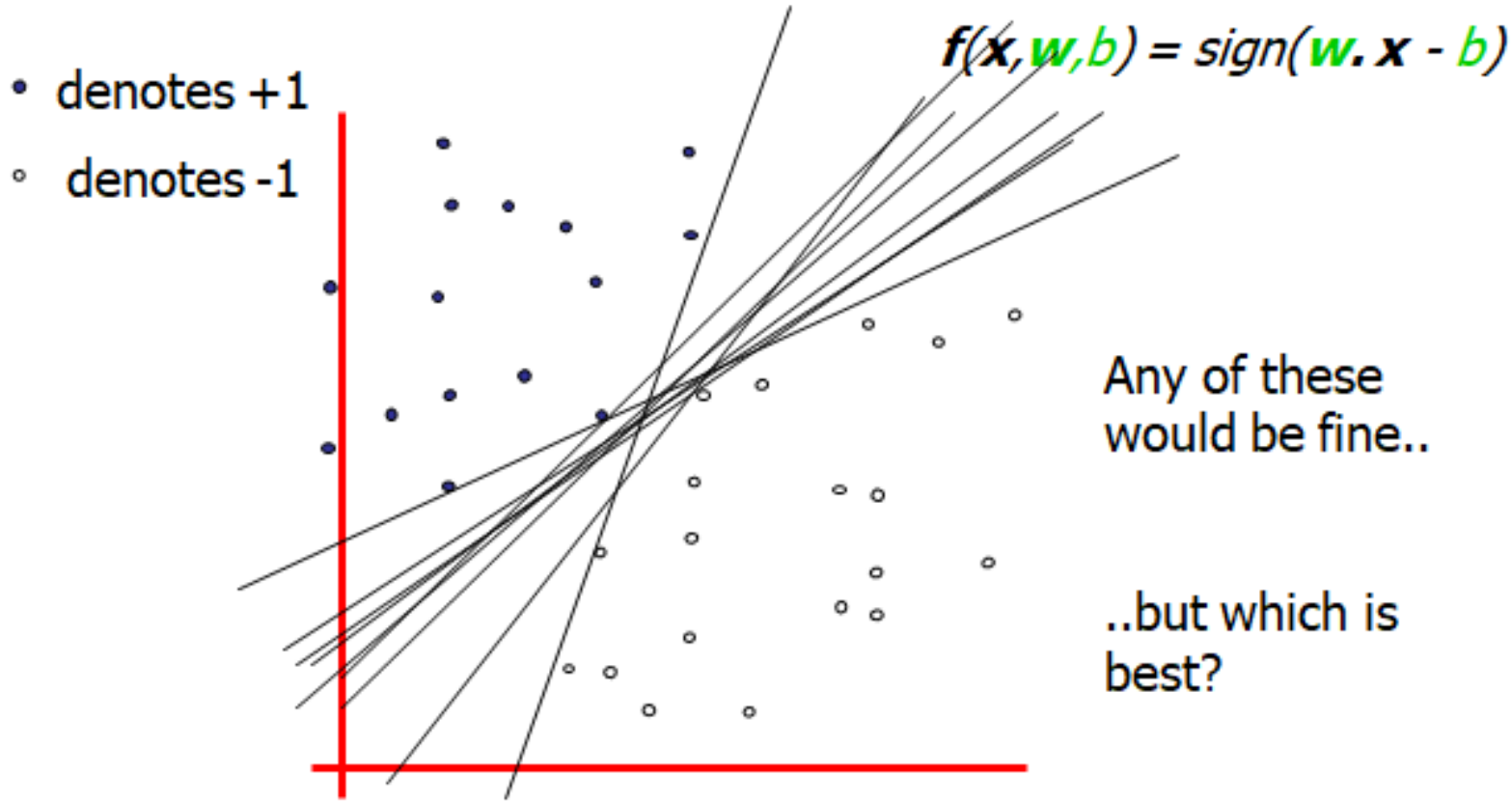


w: weight vector
x: data vector

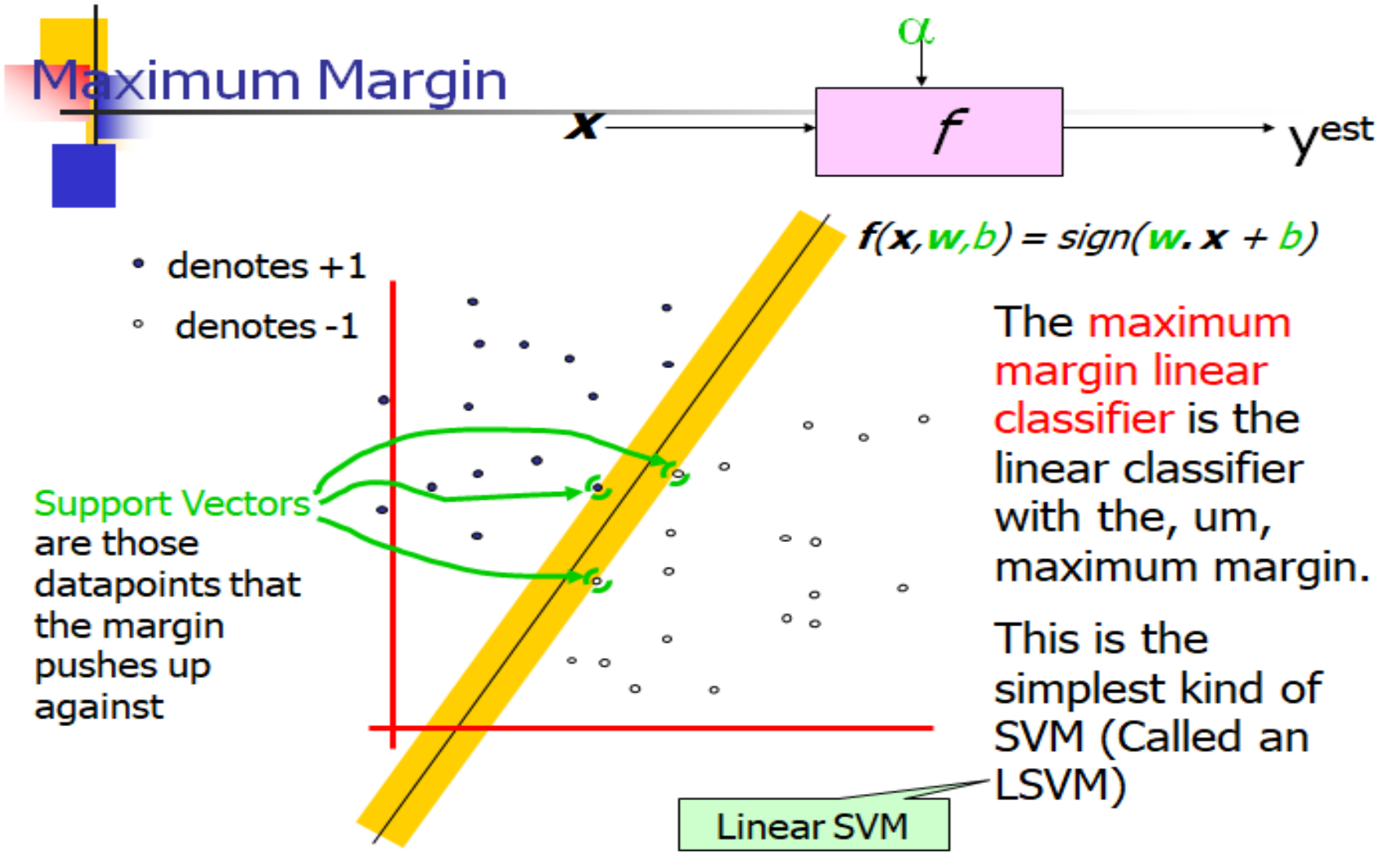
How would you classify this data?

Linear Classifiers





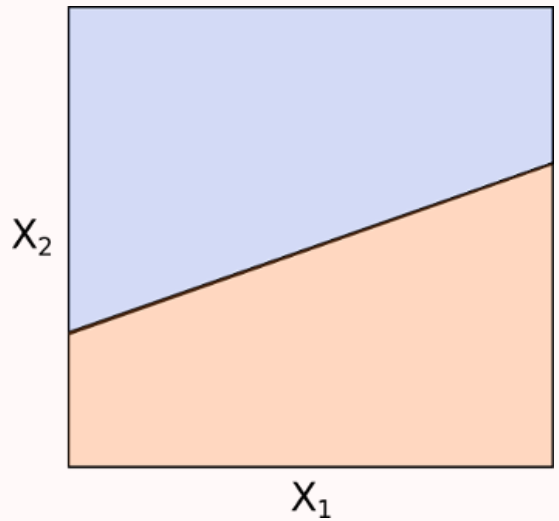
Maximum Margin



Support Vectors

- Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).
- Support vectors are the data points that lie closest to the decision surface (or hyperplane)
- They are the data points most difficult to classify
- They have direct bearing on the optimum location of the decision surface
- Support vectors are the data points nearest to the hyperplane, the points of a data set that, if removed, would alter the position of the dividing hyperplane. Because of this, they can be considered the critical elements of a data set.

- If an element $\vec{x} \in \Re^p$ satisfies this relation then it lives on the p−1-dimensional hyperplane. This hyperplane splits the p-dimensional feature space into two classification regions.



- Elements $\vec{x}$ above the plane satisfy:

$$\vec{b}.\vec{x} + b_0 > 0$$

While those below it satisfy:

$$\vec{b}.\vec{x} + b_0 < 0$$

- This can be formalized by considering the following function $f(\vec{x}^*)$, with a test observation $\vec{x}^*=(\vec{x}_1^*,\ldots,\vec{x}_p^*)$

$$f(\vec{x}^*) = \vec{b}.\vec{x}^* + b_0$$

- If $f(\vec{x}^*) > 0$, then $y^* = 1$, whereas $f(\vec{x}^*) < 0$, then $y^* = -1$.
- However, this tells us nothing about how we go about finding the b_j components of $\vec{b}$, as well as b_0, which are crucial in helping us determine the equation of the hyperplane separating the two regions.

Reference

http://www.svms.org/tutorials/Berwick2003.pdf

http://blog.aylien.com/support-vector-machines-for-dummies-a-simple/

https://www.svm-tutorial.com/2014/11/svm-understanding-math-part-1/

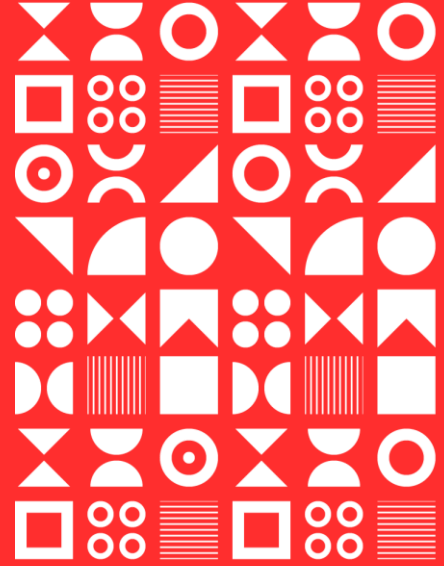
https://data-flair.training/blogs/svm-support-vector-machine-tutorial/

https://www.quantstart.com/articles/Support-Vector-Machines-A-Guide-for-Beginners

https://eight2late.wordpress.com/2017/02/07/a-gentle-introduction-to-support-vector-machines-using-r/

Kai Zhang., “Decision Tree Algorithm “

Questions ??



semicolon