IST407/707 Applied Machine Learning

KNN, SVM, Random Forest, Ensemble learning

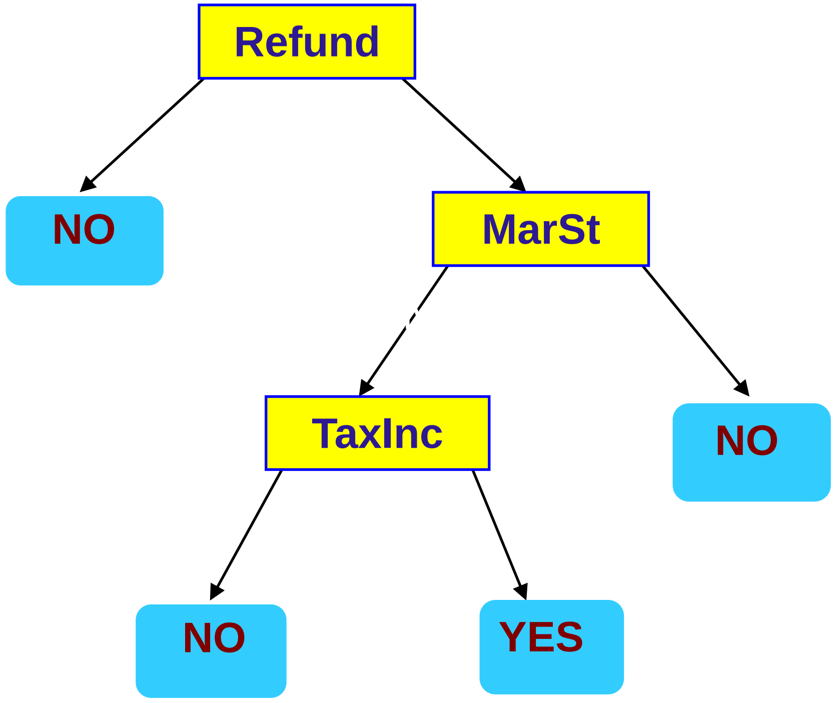
Syracuse University
School of Information Studies

K NEAREST NEIGHBORS

Syracuse University
School of Information Studies

Machine Learning algorithms

Algorithms like decision tree and naïve Bayes will construct a learning model from training examples, and then apply the model for prediction on new test examples.



naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1
P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/3
P(Marital Status=Divorced|Yes)=1/3
P(Marital Status=Married|Yes) = 0

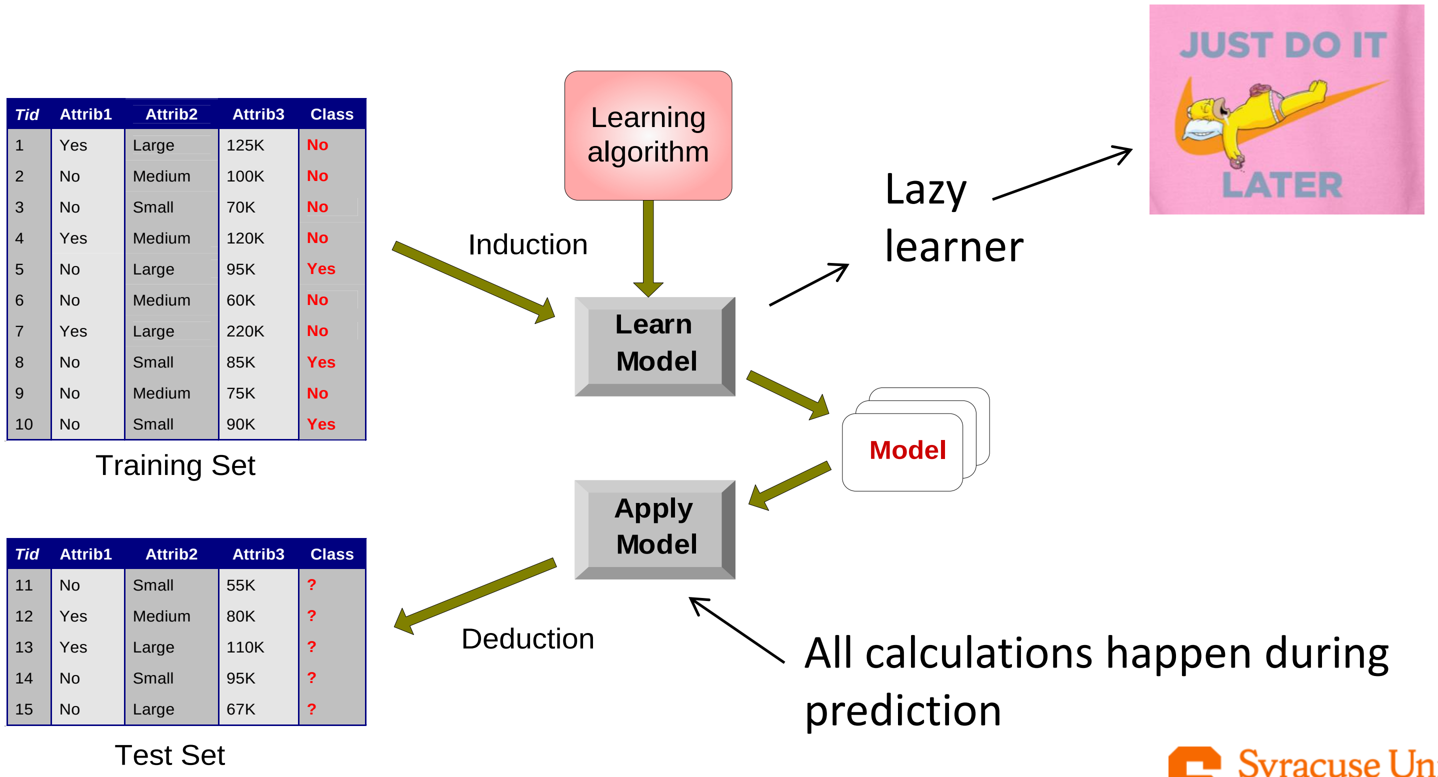
For taxable income:

If class=No:	sample mean=110
	sample variance=2975
If class=Yes:	sample mean=90
	sample variance=25

ersity
tion Studies

Instance-based learning

In contrast, instance-based learning methods simply store the training examples without doing any calculations during training process, and classification/prediction is delayed until new examples are given.



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

K-Nearest Neighbor (k-NN)

Training process:

- Read in all training examples

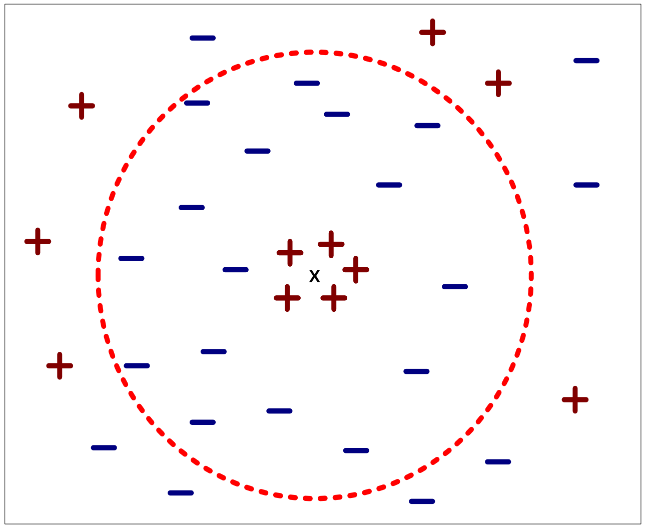
Classification process:

- Given a test example, compare the similarity between the test example and all training examples, choose the majority-voted category label in the k nearest training examples

Nearest Neighbor Classification…

Choosing the value of k:

- If k is too small, sensitive to noise points
- If k is too large, neighborhood may include points from other classes



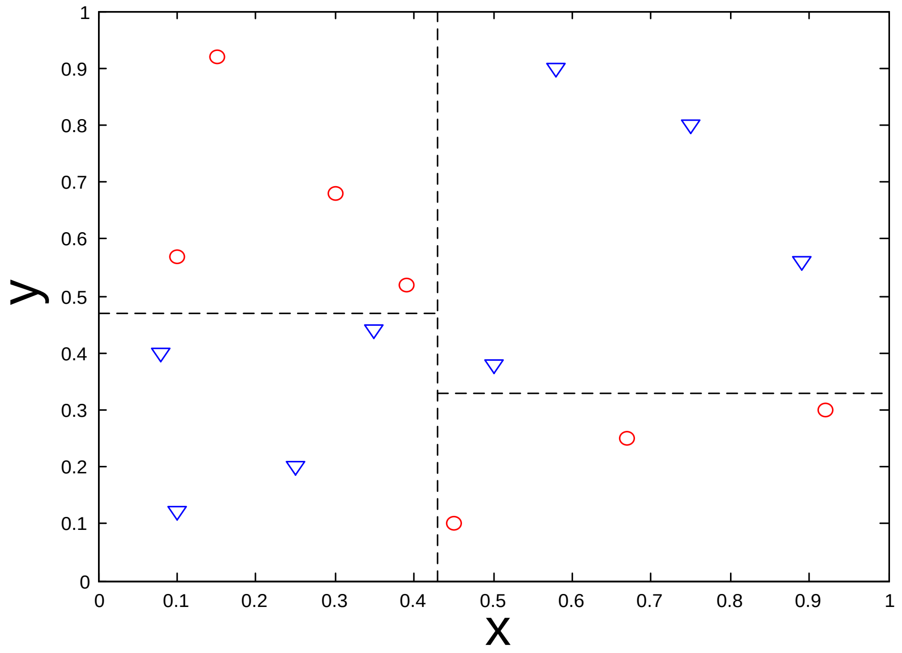
Advantages of kNN

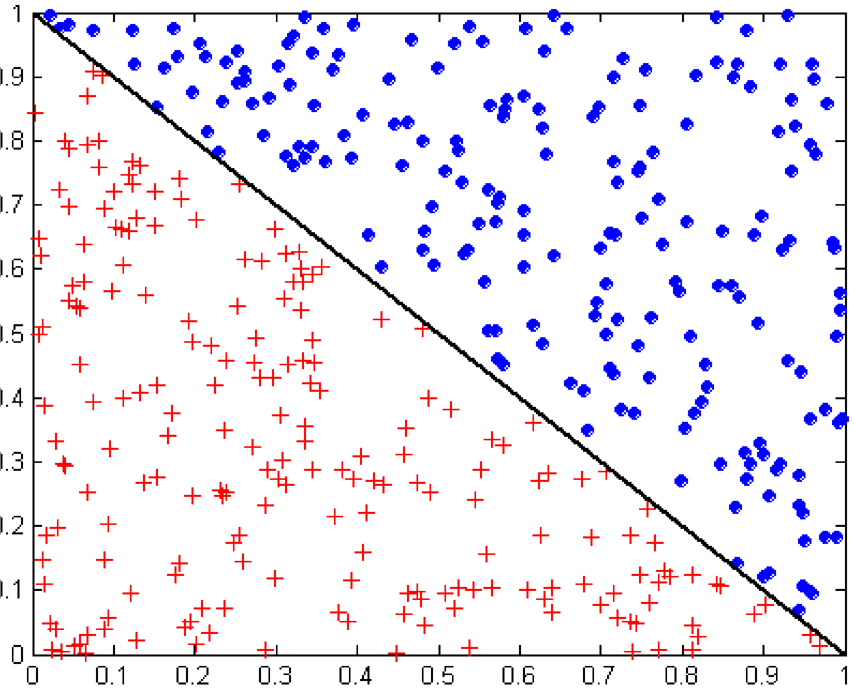
No assumptions made

- Remember the independence assumption in naïve Bayes

Works well when the decision function to be learned is very complex

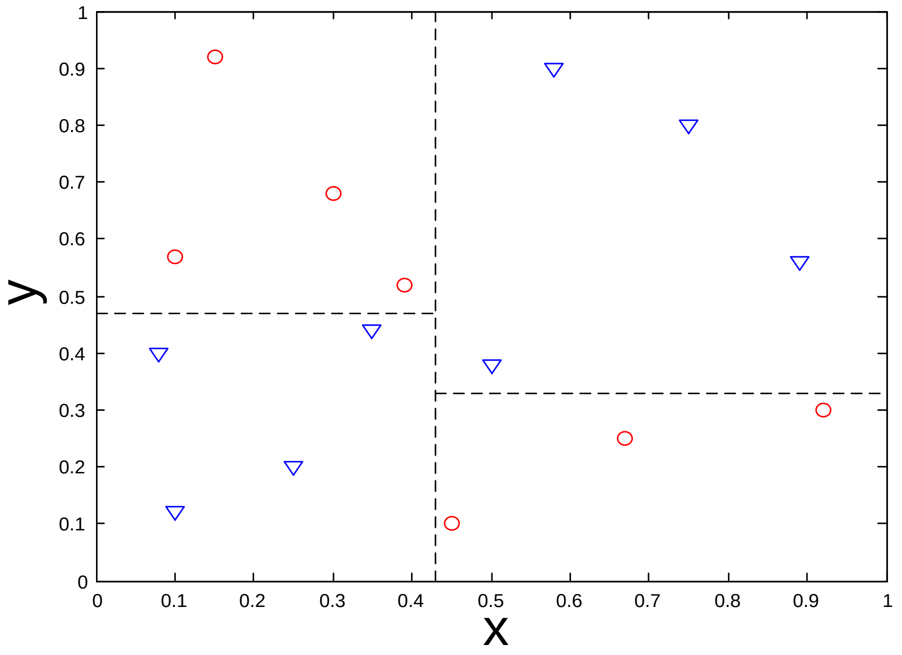
The shape of decision boundary matters

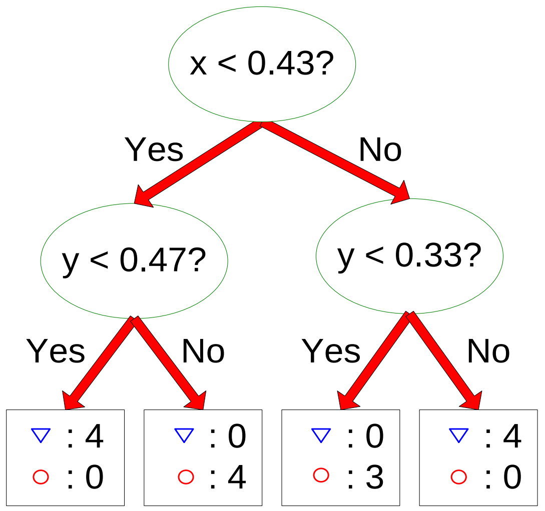




Syracuse University
School of Information Studies

Decision boundary of decision tree models





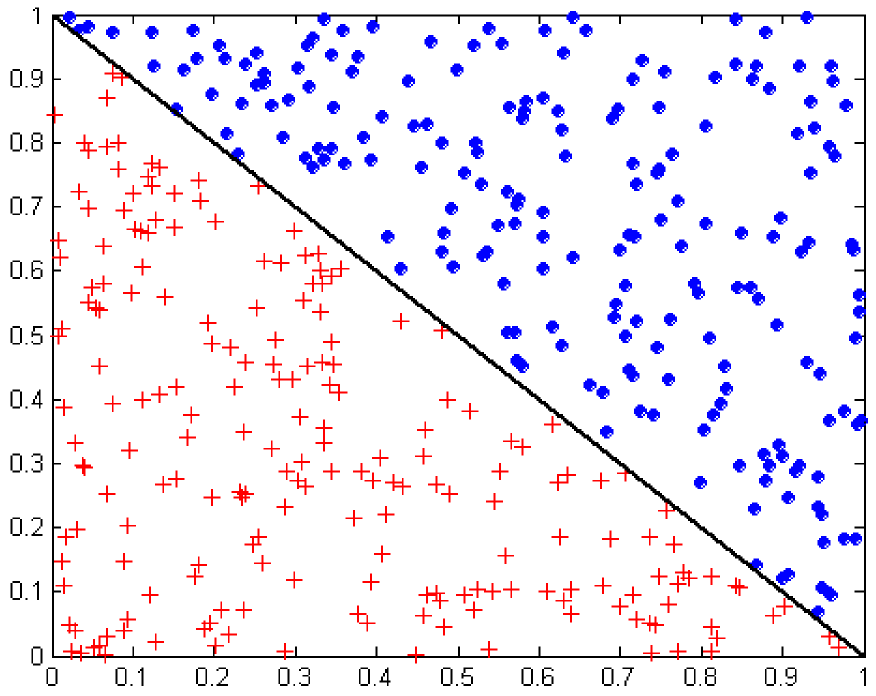
Syracuse University
School of Information Studies

Decision boundary of linear models

Linear: naïve Bayes, SVM

How many parameters to determine a line in 2D space?

- Y=ax+b
- Weight
- intercept

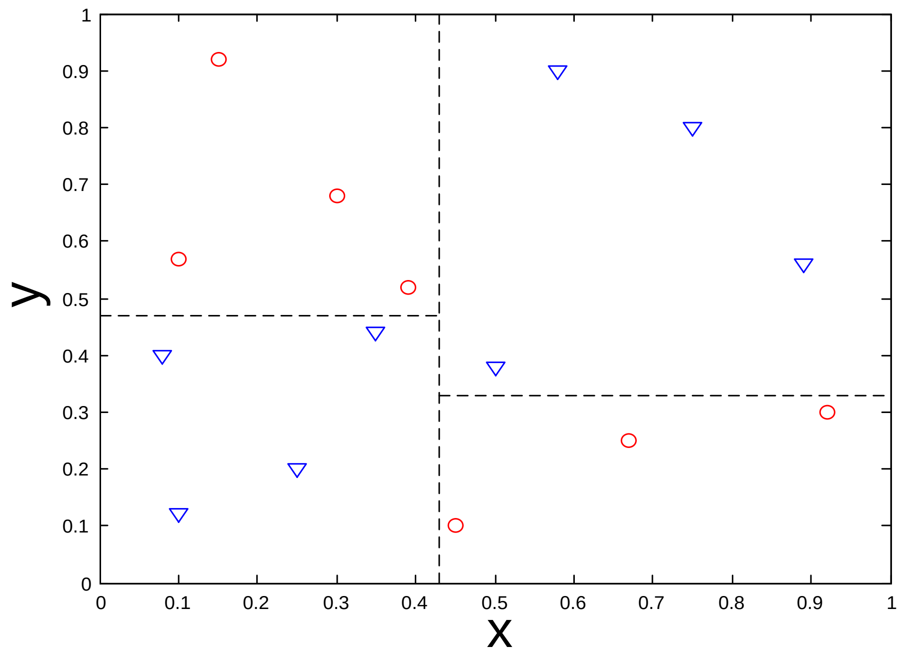


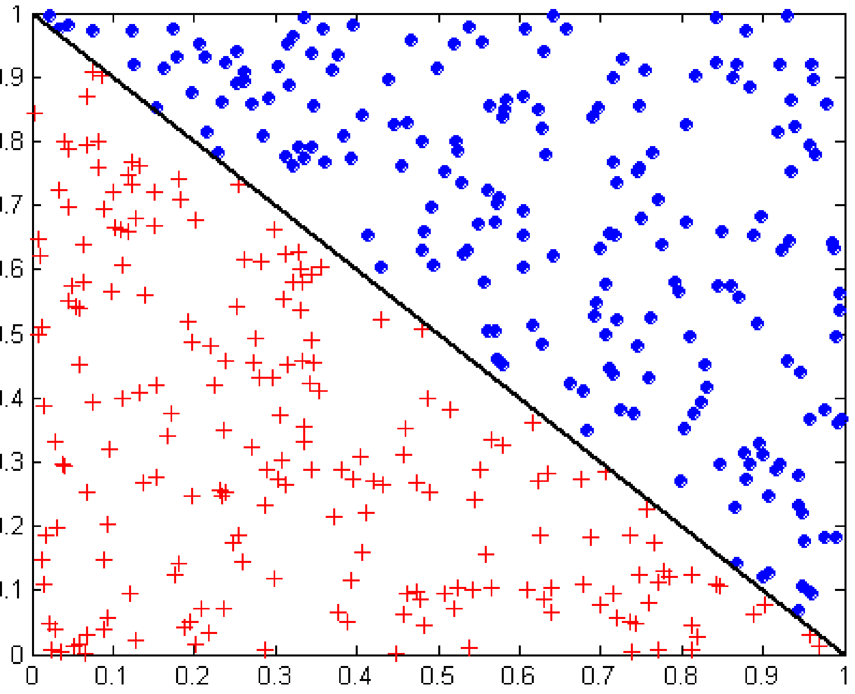
Why is NB a linear classifier?

The decision function can be re-written to a linear function

- Original decision function
  - Prob(Ci)*Prob(T1|Ci)*Prob(T2|Ci) * ... *Prob(Tm|Ci))
- Apply log transformation
  - log(Prob(Ci)) + log(Prob(T1|Ci)) + log(Prob(T2|Ci)) + ... + log(Prob(Tm|Ci))

The shape of decision boundary matters

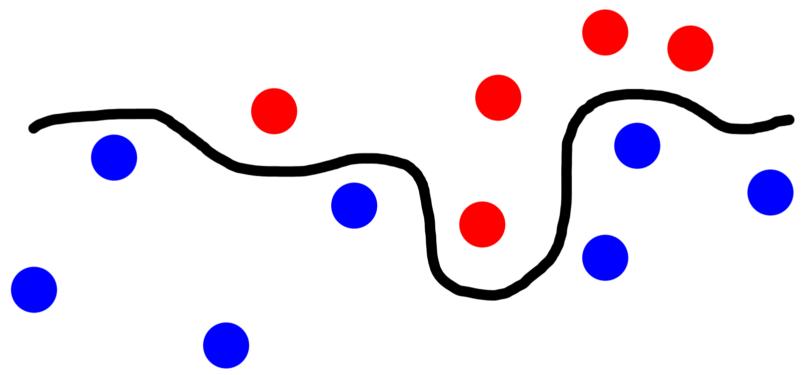




Syracuse University
School of Information Studies

Advantage of kNN

The decision boundary has no pre-defined shape



Syracuse University
School of Information Studies

Disadvantages of kNN (1)

Sensitive to noisy training data

- All attributes participate in classification
- If only a few relevant attributes are relevant to prediction, the participation of those irrelevant attributes would harm the prediction performance.

Disadvantage of kNN (2)

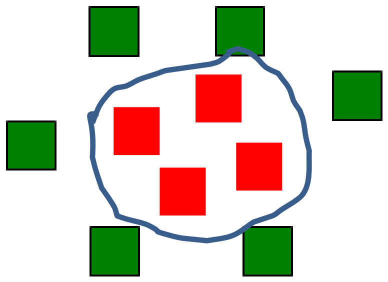
High computational cost.

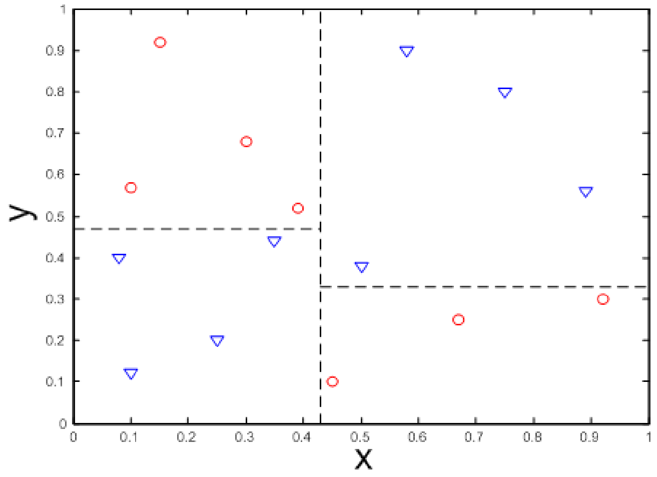
- Pre-computed models can be quickly applied to test data
- Since there is no training step, nearly all computation takes place in the prediction step.

SUPPORT VECTOR MACHINES

The shape of decision boundary

Some data are not linearly separable

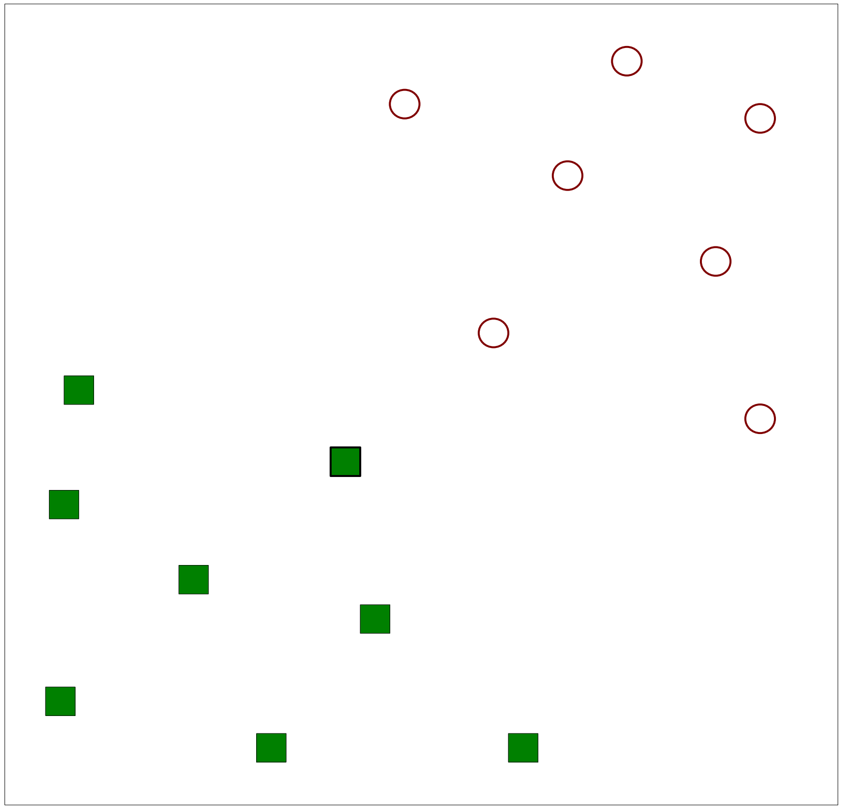




Support Vector Machines (SVMs): an algorithm that can solve both linearly separable and inseparable problems

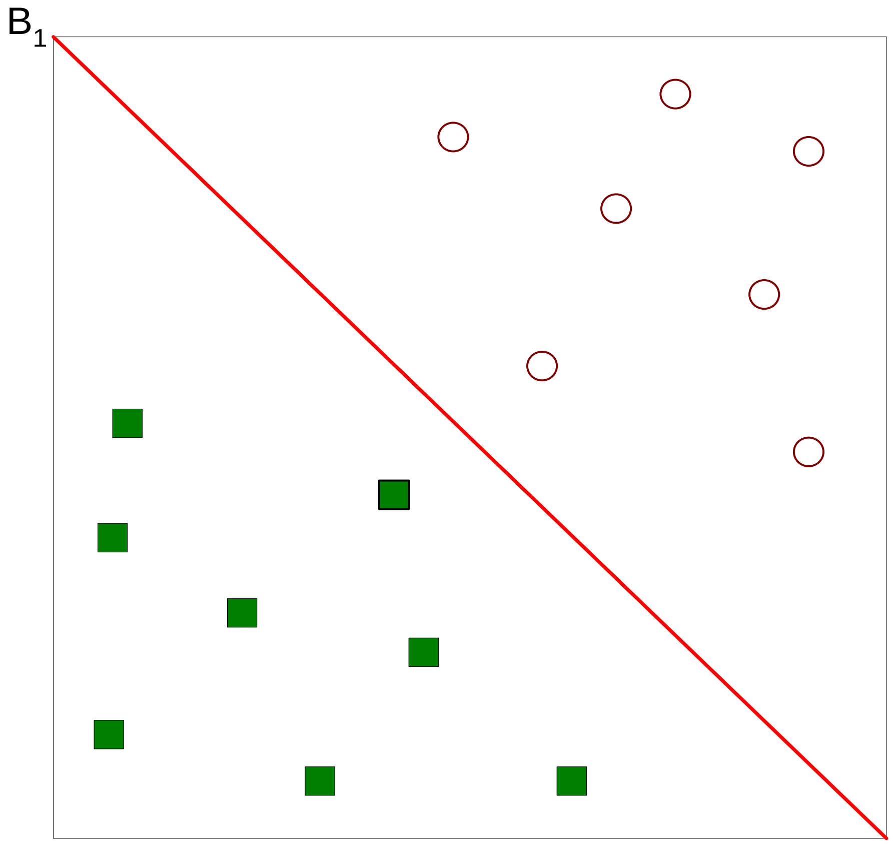
Linear decision boundaries

Find a linear hyperplane (decision boundary) that can separate the data



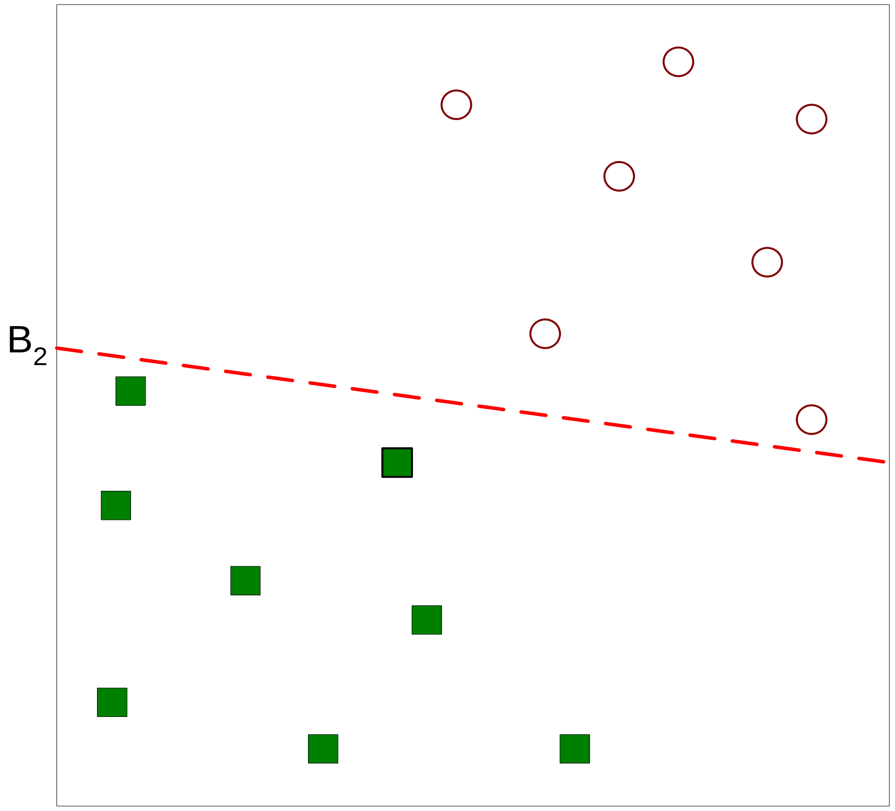
Linear decision boundaries

One Possible Solution



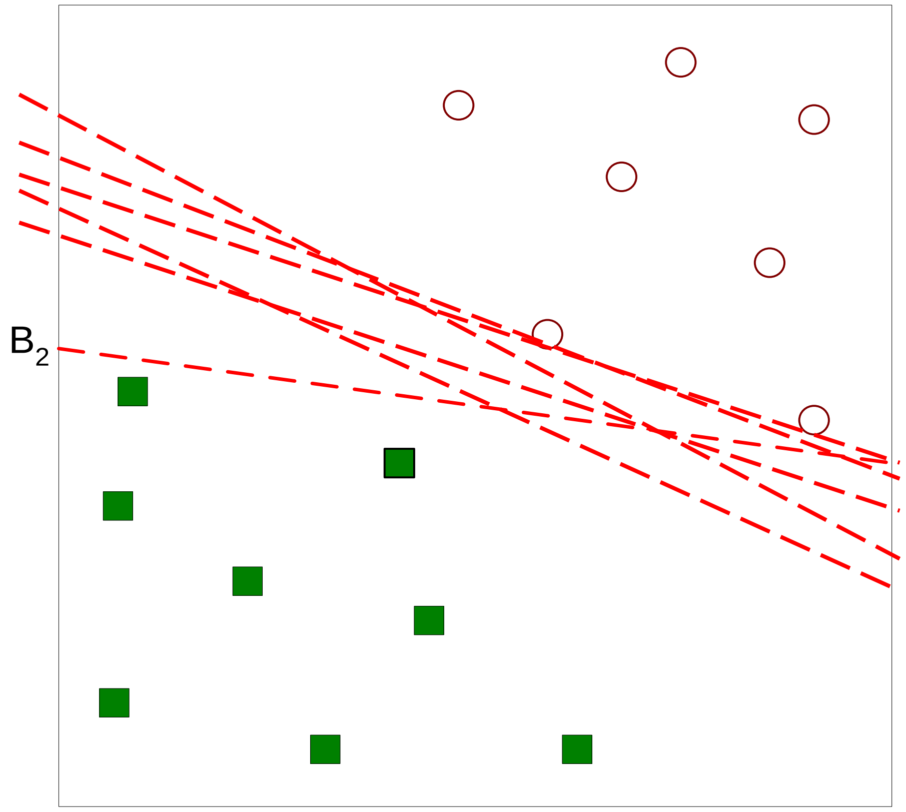
Linear decision boundaries

Another possible solution



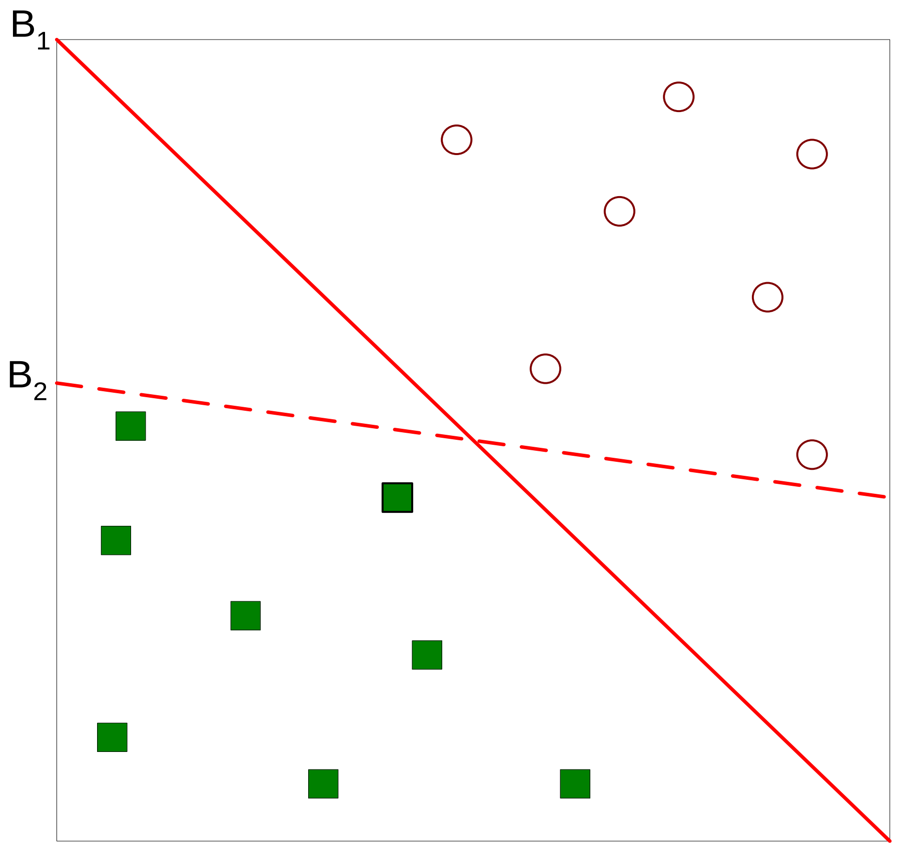
Linear decision boundaries

Numerous possible solutions



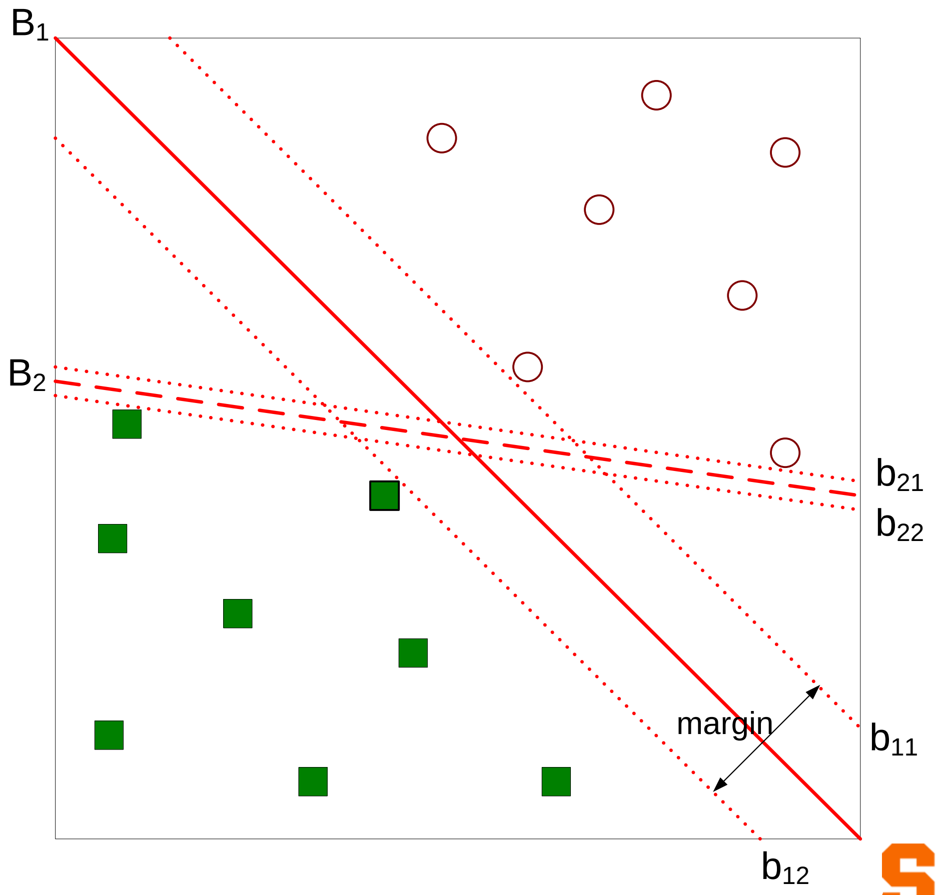
Linear decision boundaries

Which one is better? B1 or B2?
How do you define better? (e.g. least square fitting)

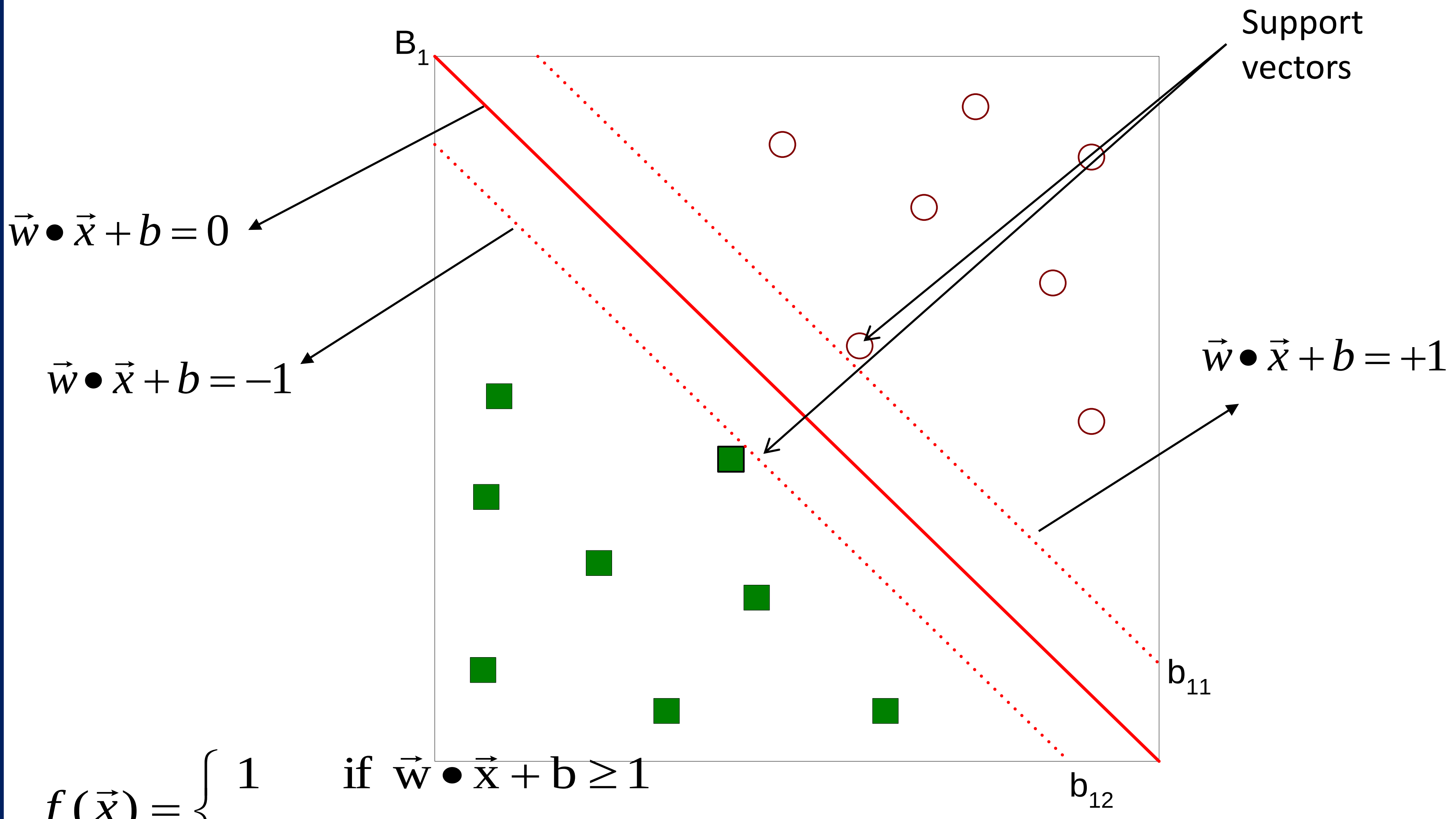


Linear decision boundaries

Find a hyperplane that maximizes the margin => B1 is better than B2



Linear decision boundaries



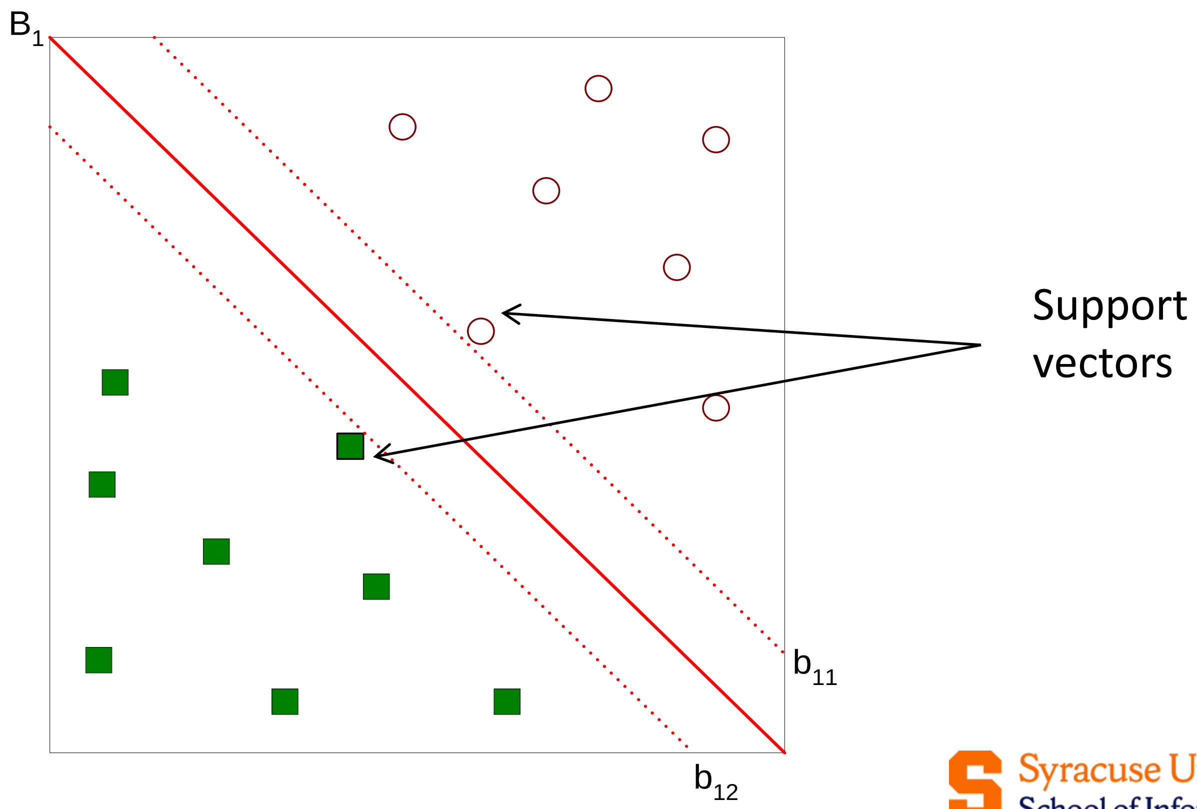
B_1
Support vectors
$\vec{w}\bullet\vec{x}+b=0$
$\vec{w}\bullet\vec{x}+b=-1$
$\vec{w}\bullet\vec{x}+b=+1$
b_{11}
b_{12}

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w}\bullet\vec{x}+b \geq 1 \\ -1 & \text{if } \vec{w}\bullet\vec{x}+b \leq -1 \end{cases}$$

Syracuse University
School of Information Studies

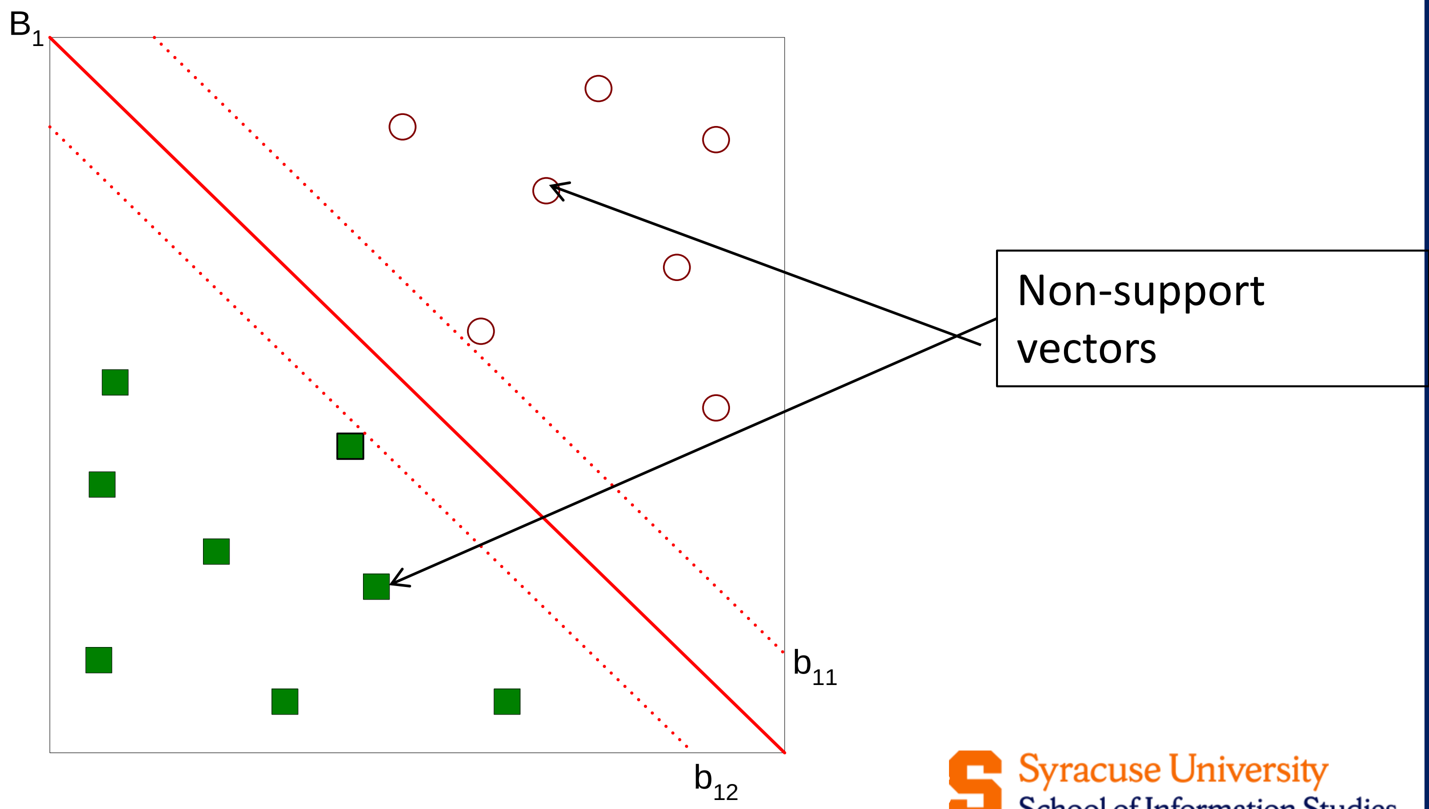
Support vectors

Support vectors are the training examples ("vectors") that are located on the margins



Non-support vectors

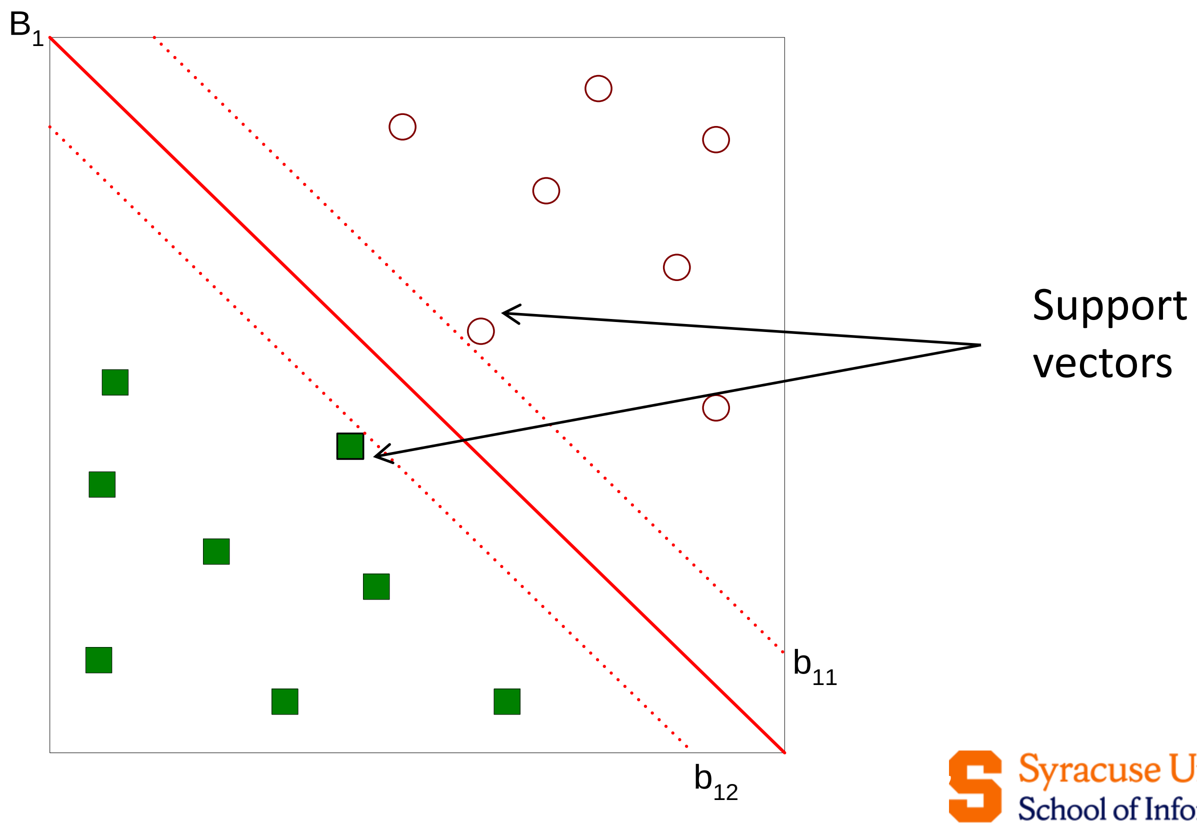
Training examples which are not support vectors do not participate in prediction



Syracuse University
School of Information Studies

Model complexity

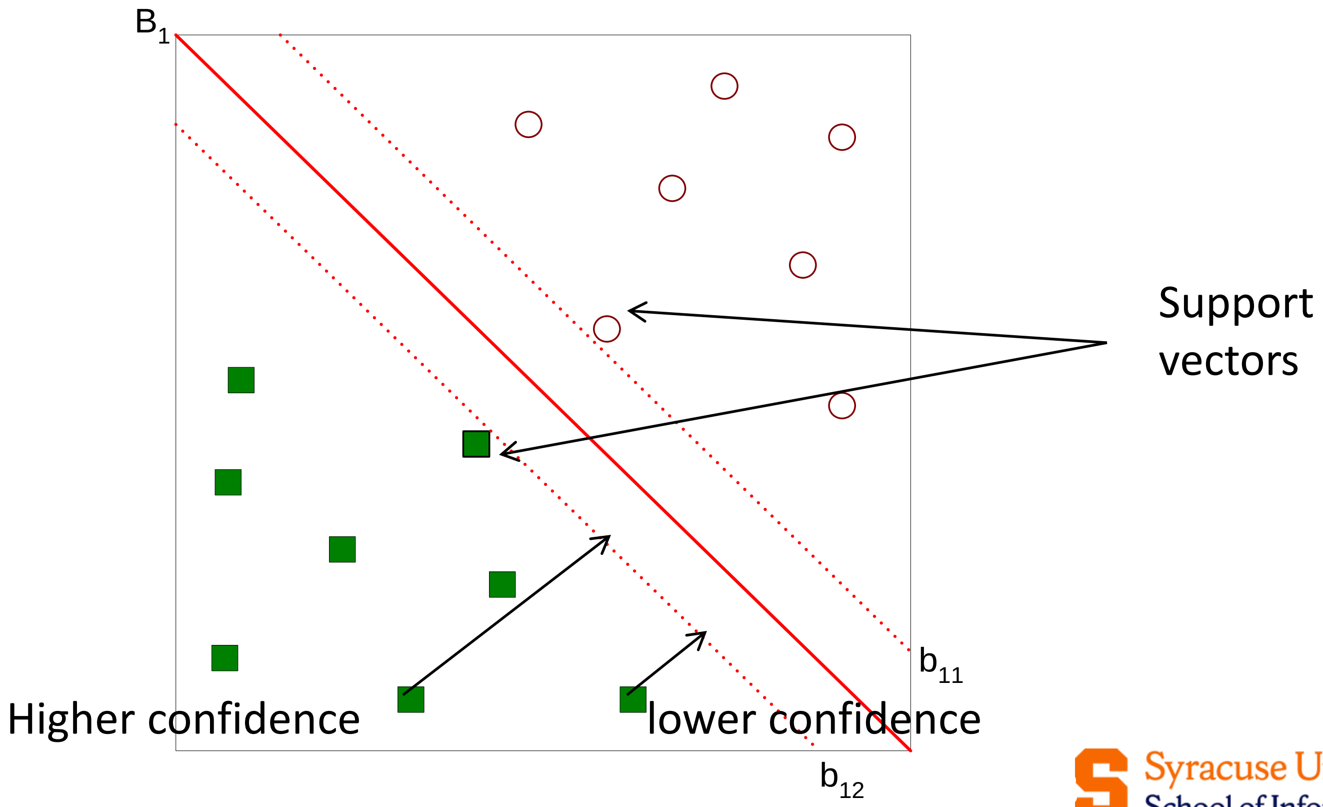
The number of support vectors is an indicator of the complexity of the trained SVM model



Syracuse University
School of Information Studies

Prediction confidence

The distance between the example and the decision boundary is an indicator of prediction confidence: the farther the better



Prediction confidence

SVMs Prediction result can be sorted by confidence, and thus is suitable for semi-supervised learning and active learning

Variant SVMs algorithm can be used for regression

Soft-margin SVMs

No perfect linear boundary can be found between the two classes due to outliers

Introduce a slack variable ξ to pay a cost for each misclassified example

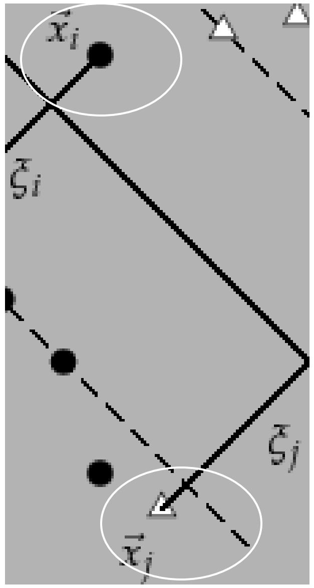


Figure 15.5 from http://nlp.stanford.edu/IR-book/html/htmledition/soft-margin-classification-1.html

Regularization in C-SVC

Tune the regularization parameter C (cost for misclassification)

Default value C=1

When C is large (high cost), the algorithm tries to build model with fewest training errors, resulting in narrow margin and high chance of overfitting

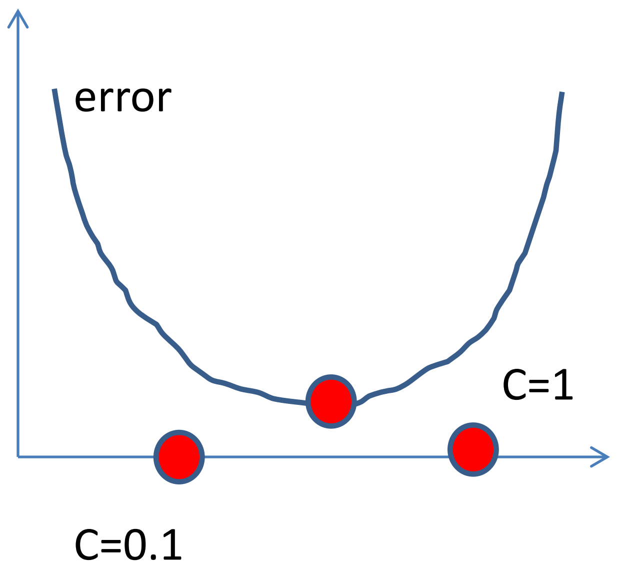
When C is small (low cost), wider margin, more robust

However, C cannot be too small, or else it does not respect the data at all.

Regularization

Use manual tuning or gradient descent search to find the best C

- E.g. set C's search range from 0.1 to 1.0 and increase with step size 0.05.



A visualization from Coursera

https://class.coursera.org/ml-003/lecture/72

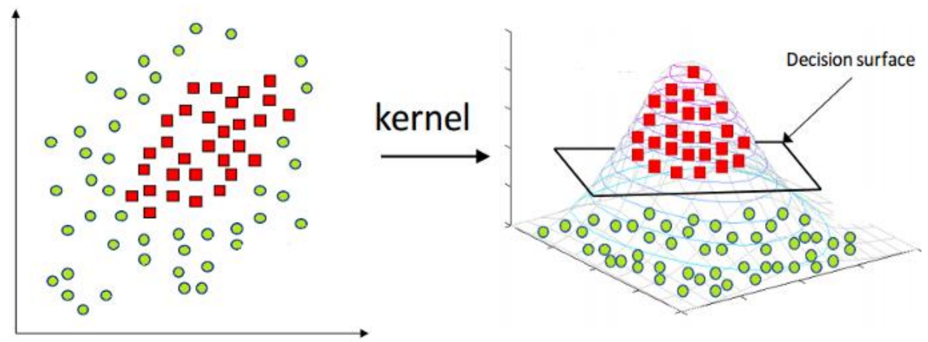
07:13-9:00

The Kernel trick in SVM

Some data are not linearly separable

But they are linearly separable in higher-dimensional space

Map the data to higher dimensional space, so that inseparable problems become separable



Kernel functions

SVM algorithm maximizes the margin between the two separating hyperplanes by finding the maximum of the function

$$W(\alpha)=\sum_{i=1}^{l}\alpha_i-\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j \boxed{K(x_i,x_j)}$$

Subject to the constraints

$$\sum_{i=1}^{l}\alpha_i y_i=0,\ \alpha_i\geq 0,\ i=1,2,\ldots,l$$

SVM—Kernel functions

- Linear kernel: $K(\mathbf{X}_i, \mathbf{X}_j) = \mathbf{X}_i \cdot \mathbf{X}_j$ (cosine similarity)
- Higher rank kernels: instead of computing on the transformed data tuples, it is mathematically equivalent to instead applying a kernel function $K(\mathbf{X}_i, \mathbf{X}_j)$ to the original data,
  i.e., $K(\mathbf{X}_i, \mathbf{X}_j) = \Phi(\mathbf{X}_i)\ \Phi(\mathbf{X}_j)$
- Typical Kernel Functions

$$\text{Polynomial kernel of degree } h: \quad K(X_i, X_j) = (X_i \cdot X_j + 1)^h$$

$$\text{Gaussian radial basis function kernel}: \quad K(X_i, X_j) = e^{-\|X_i - X_j\|^2/2\sigma^2}$$

$$\text{Sigmoid kernel}: \quad K(X_i, X_j) = \tanh(\kappa X_i \cdot X_j - \delta)$$

Syracuse University
School of Information Studies

Extend binary classification to multi-class

Given n classes, e.g.

- Sentiment = {positive, negative, neutral, no opinion}

One-vs-one (pairwise) strategy:

- Create n(n-1)/2 classifiers: pos|neg, pos|neu, pos|np, neg|neu, neg|np, neu|np
- Pick the most confident prediction

One-vs-all strategy:

- Create n classifiers: positive or not, negative or not, neutral or not, np or not
- Pick the most confident prediction

SVMs Strength

High tolerance to noisy data

Flexibility in data representation: well-suited for continuous- or discrete-valued inputs and outputs

Probabilistic prediction result

Scalability: successful on extremely large problems

Successful on a wide array of real-world data

SVMs weakness

Require a number of parameters for each kernel type

Interpretability

- Easy interpretation for linear kernel
- Difficult to interpret the model generated by nonlinear kernels

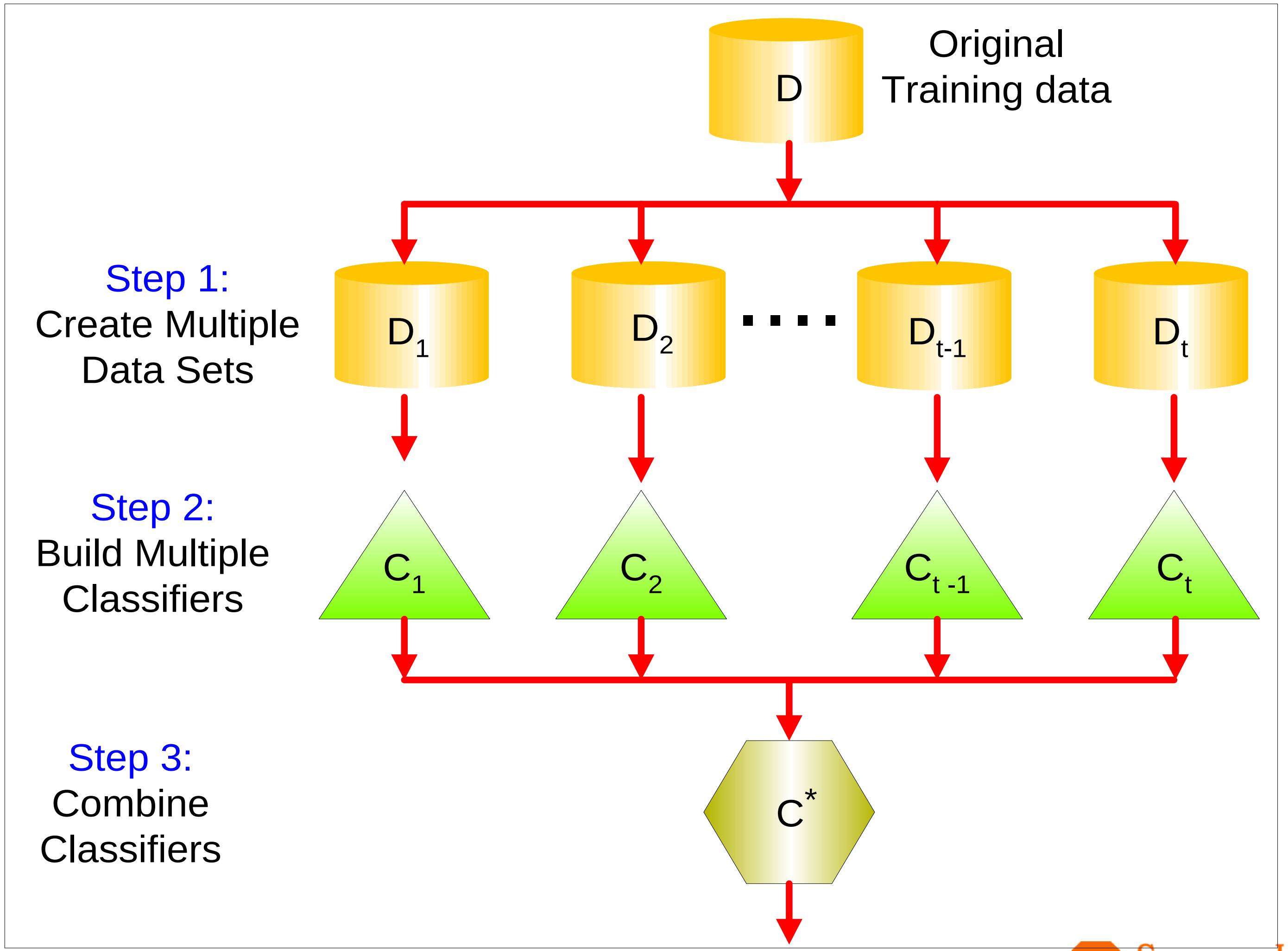
ENSEMBLE METHODS AND RANDOM FORESTS

Ensemble Methods

Construct a set of classifiers from the training data

Predict class label of previously unseen records by aggregating predictions made by multiple classifiers

General Idea


D
Original Training data
Step 1: Create Multiple Data Sets
D_1
D_2
....
D_{t-1}
D_t
Step 2: Build Multiple Classifiers
C_1
C_2
C_{t-1}
C_t
Step 3: Combine Classifiers
C^*

Why does ensemble work?

Suppose there are 25 base classifiers

- Each classifier has error rate, ε = 0.35 (weak learner)
- Assume classifiers are independent
- Use majority voting to combine results, so ensemble makes a wrong prediction only if over half of the base classifiers are wrong
- Probability that the ensemble classifier makes a wrong prediction:

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.06$$

- Error rate is reduced from .35 to .06
- In practice, the base classifiers may not be totally independent for a reduction in error rate to occur

Bagging

Bagging is used when the goal is to reduce the variance of a classifier.

Here the objective is to create several subsets of data from training sample chosen randomly with replacement. Each collection of subset data is used to train their classification model.

As a result, we get an ensemble of different models. Average of all the predictions from different trees are used which is more robust than a single classification model.

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

https://analyticsindiamag.com/primer-ensemble-learning-bagging-boosting/

Boosting

Boosting is used to create a collection of predictors. In this technique, learners are learned sequentially with early learners fitting simple models to the data and then analyzing data for errors.

IDEA: adaptively change distribution of training data by focusing more on previously misclassified records

- Initially, all N records are assigned equal weights
- Unlike bagging, weights may change at the end of boosting round

Boosting

Records that are wrongly classified will have their weights increased

Records that are classified correctly will have their weights decreased

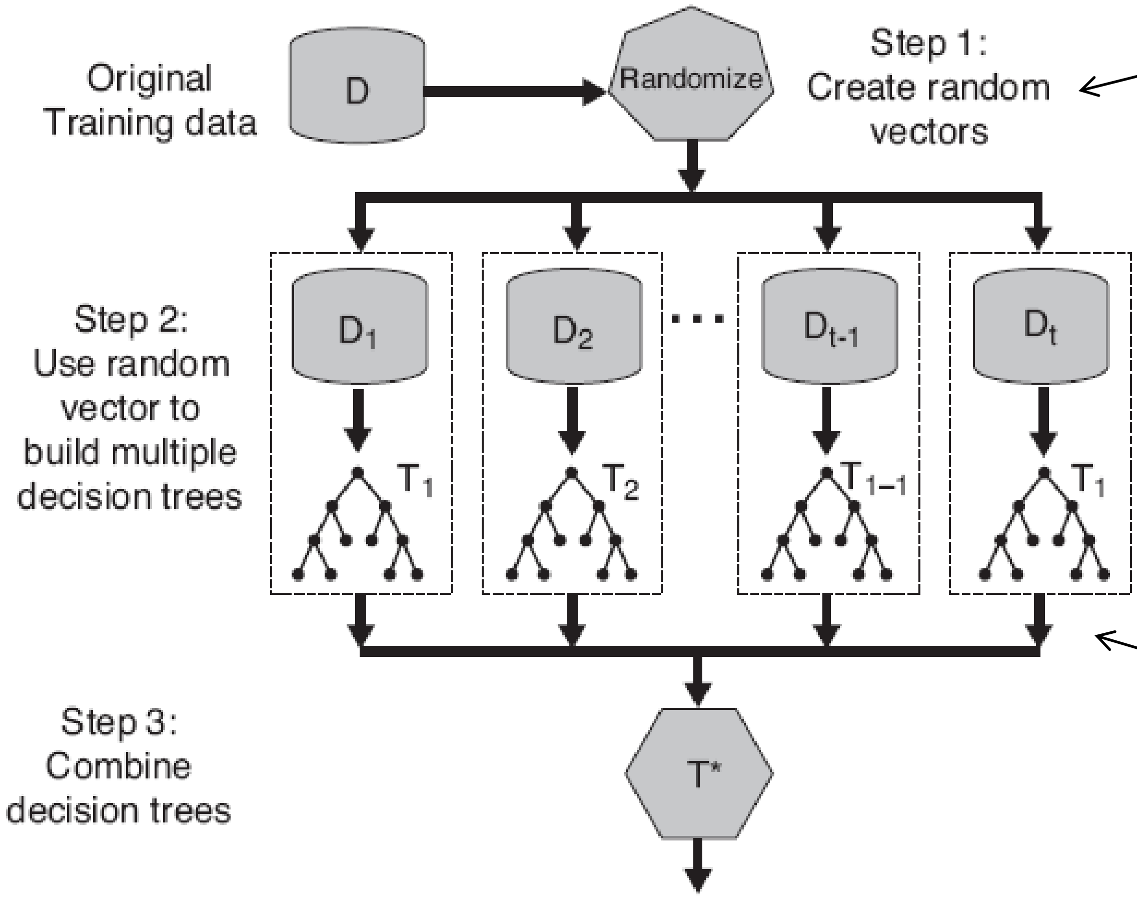
Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

- Example 4 is hard to classify
- Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds

Goal is to make sure model can learn the correct label for this example

Syracuse University
School of Information Studies

Random Forest


Original Training data
D
Randomize
Step 1: Create random vectors
Step 2: Use random vector to build multiple decision trees
D1
D2
Dt-1
Dt
T1
T2
T1–1
T1
Step 3: Combine decision trees
T*

Randomly select subset of features

Model with too many trees tends to overfit

Figure 5.40. Random forests.

Syracuse University School of Information Studies

Exercise: kNN

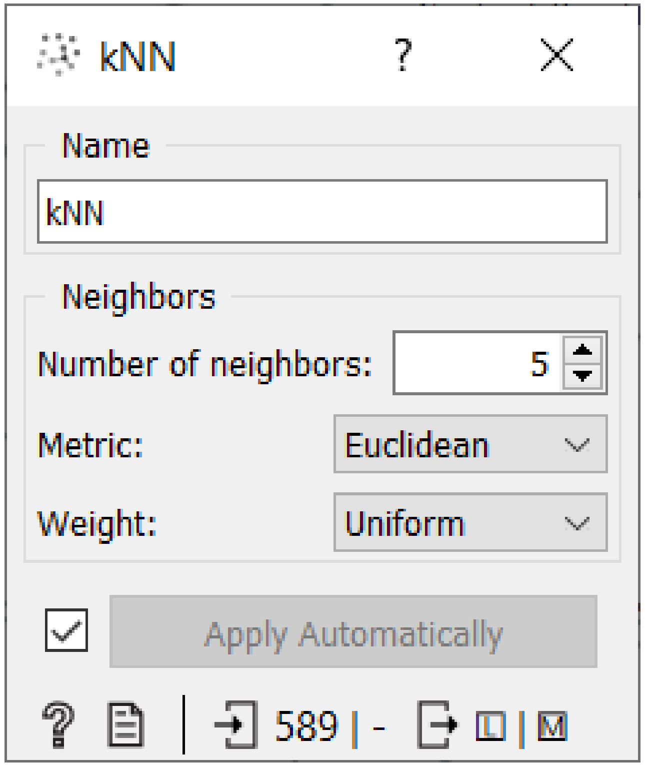
Using the Titanic data:

Create a kNN model

Report accuracy when number of neighbors = {1, 3, 5, 7}.

Pick the best k value and then change the weighted distance metric.

Report if distance weighting helps or not.



kNN

Name: kNN

Neighbors

Number of neighbors: 5

Metric: Euclidean

Weight: Uniform

Syracuse University
School of Information Studies

Exercise: SVMs

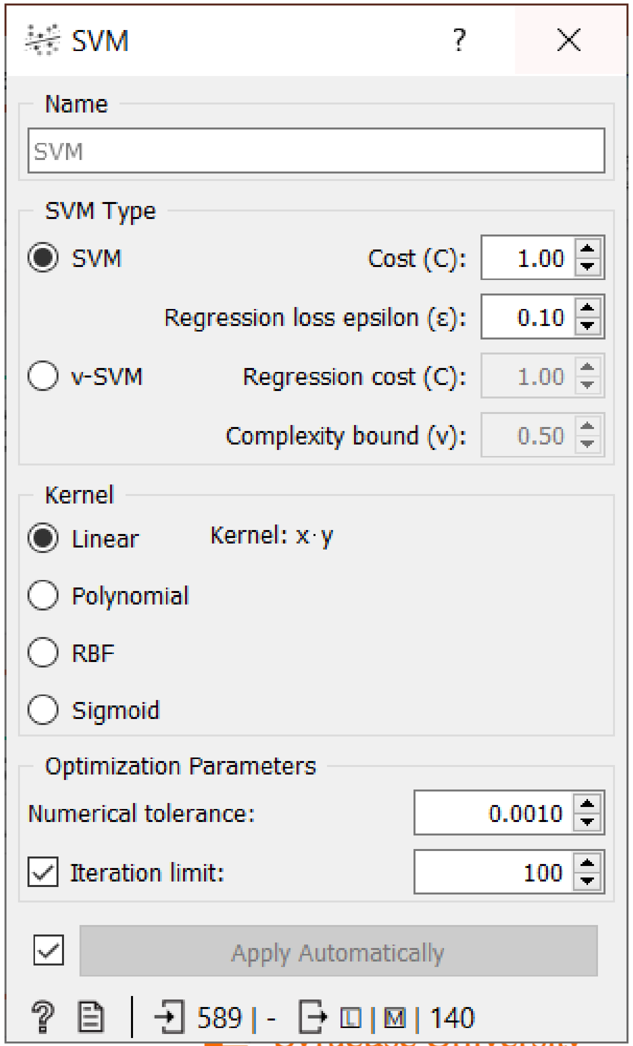
Using the Titanic data:

Create an SVM model

Report and compare accuracies of each kernel.

After picking the best kernel, tune C = {1, 0.8, 0.5, 0.3, 0.1}

Report accuracies and summarize if C tuning helps or not.



SVM

Name: SVM

SVM Type

[✓ SVM] Cost (C): 1.00

Regression loss epsilon (ε): 0.10

v-SVM Regression cost (C): 1.00

Complexity bound (ν): 0.50

Kernel

[✓ Linear] Kernel: x·y

Polynomial

RBF

Sigmoid

Optimization Parameters

Numerical tolerance: 0.0010

[✓] Iteration limit: 100

[✓] Apply Automatically

589 | - 140

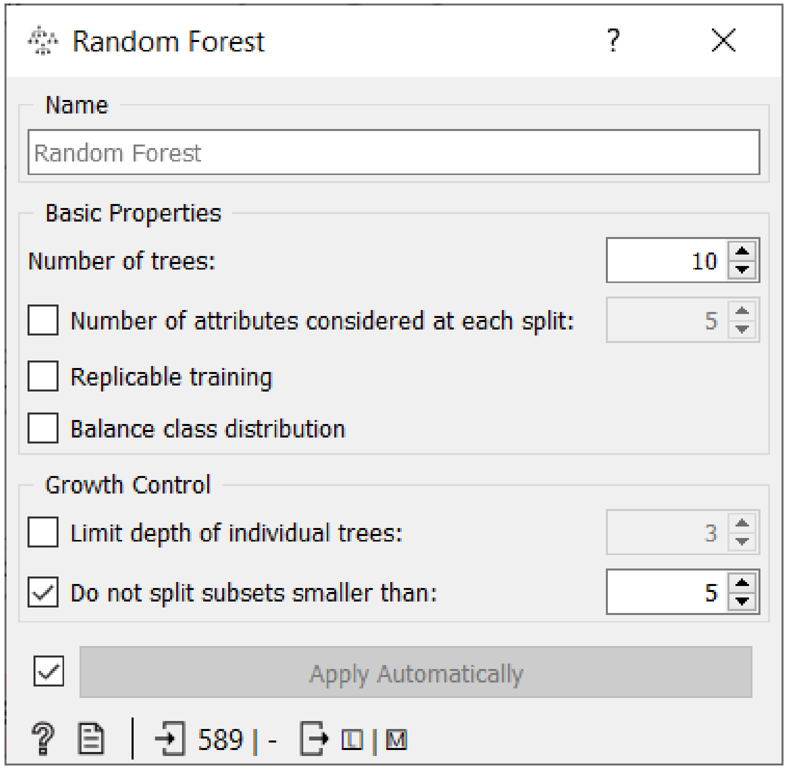
Exercise: random forest

Using the Titanic data:

Create a random forest

Tune the number of trees = {5, 10, 25, 50, 100}.

Which random forest is more likely to overfit?



Random Forest

Name: Random Forest

Basic Properties

Number of trees: 10

☐ Number of attributes considered at each split: 5

☐ Replicable training

☐ Balance class distribution

Growth Control

☐ Limit depth of individual trees: 3

☑ Do not split subsets smaller than: 5

☑ Apply Automatically

589 | -

Exercise: Stacking

Stacking

Use the stacking module to combine your previous models (KNN, SVM, random forest) into one

How does your stacked model perform compared to the individual models?

- In what ways does changing the individual models (k, C, number of trees, etc.) impact this performance?

Exercise: Gradient boosting

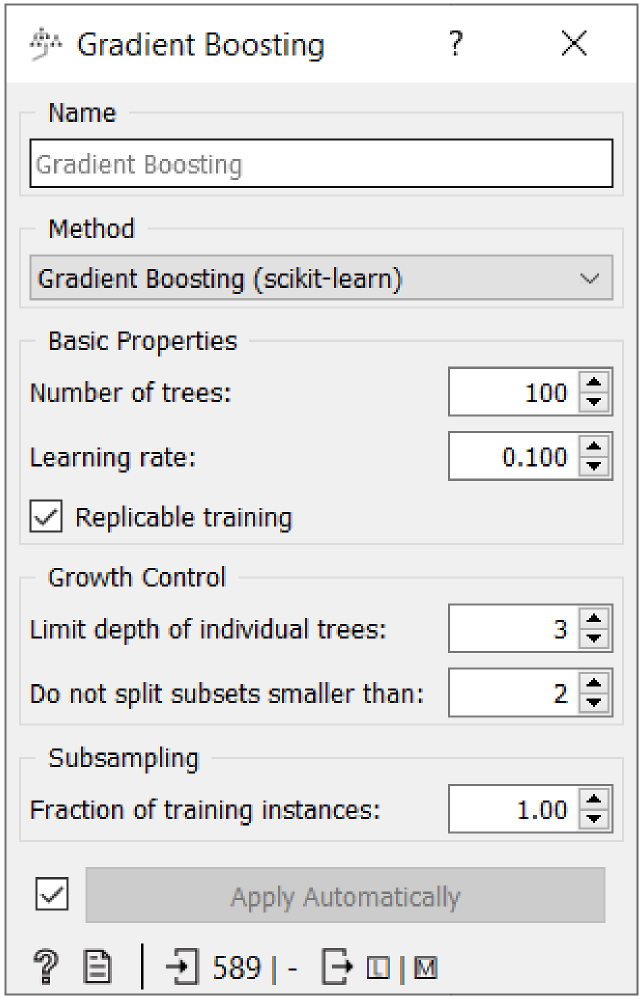
Using the Titanic data:

•Create a gradient boosting model

•Report and compare the accuracies for each method

•How does this model perform when compared to the others?

- How do changes in the number of trees or lambda-value affect performance?



Gradient Boosting

Name: Gradient Boosting

Method: Gradient Boosting (scikit-learn)

Basic Properties

Number of trees: 100

Learning rate: 0.100

✓ Replicable training

Growth Control

Limit depth of individual trees: 3

Do not split subsets smaller than: 2

Subsampling

Fraction of training instances: 1.00

✓ Apply Automatically

589 | -

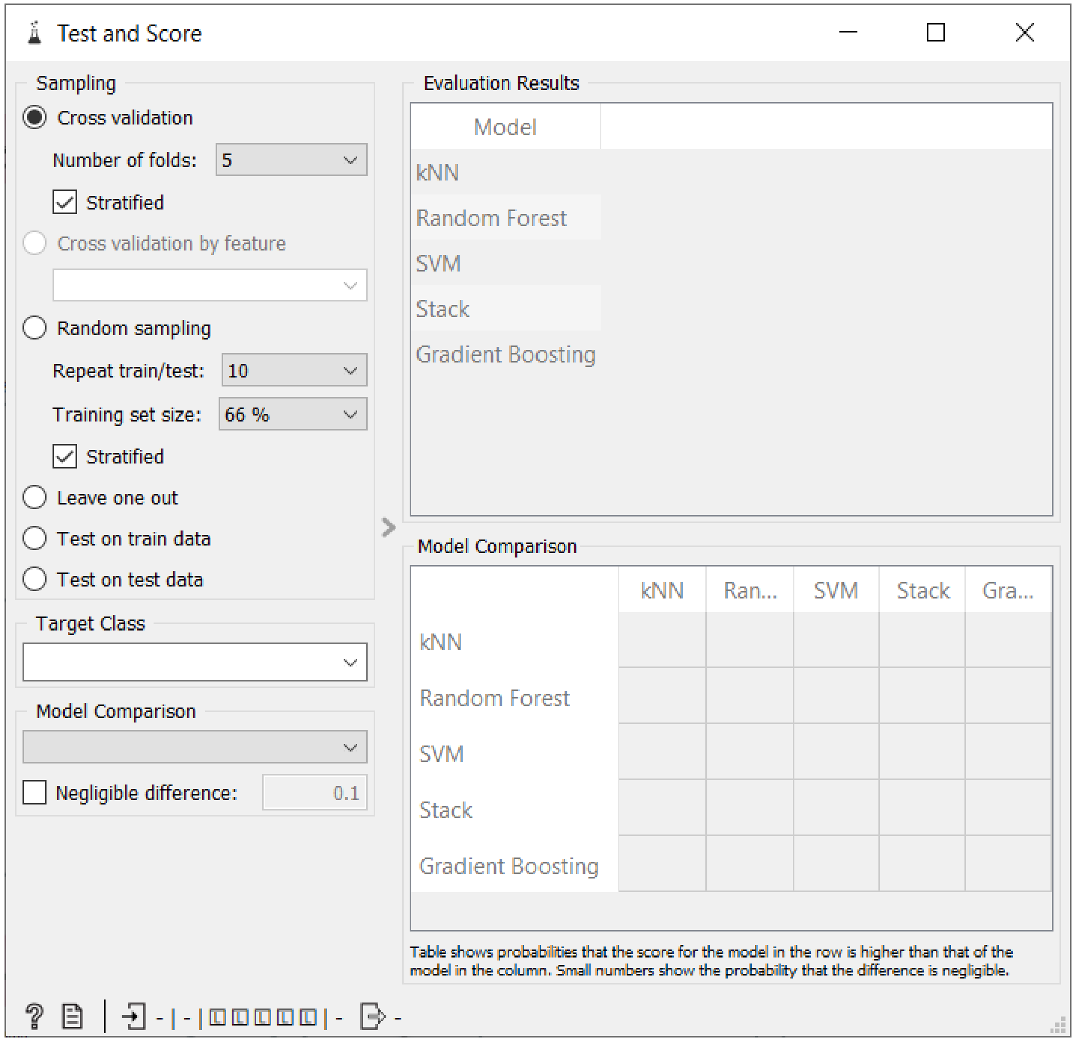
Syracuse University
School of Information Studies

Exercise: algorithm comparison

Compare algorithm performance on Titanic task.

- Same data preprocessing and 5-fold CV

Which algorithm is the best? Why?


Test and Score

Sampling
- [✓ Cross validation]
  - Number of folds: 5
  - [✓ Stratified]
- Cross validation by feature
- Random sampling
  - Repeat train/test: 10
  - Training set size: 66 %
  - [✓ Stratified]
- Leave one out
- Test on train data
- Test on test data

Target Class

Model Comparison

Negligible difference: 0.1

Evaluation Results

Model
kNN
Random Forest
SVM
Stack
Gradient Boosting

Model Comparison

	kNN	Ran...	SVM	Stack	Gra...
kNN					
Random Forest					
SVM					
Stack					
Gradient Boosting					

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.