EECS E6690: SL for Bio & Info
Lecture 7: Boosting and Intro to Support Vector Machines

Prof. Predrag R. Jelenković
Time: Tuesday 4:10-6:40pm
303 Seeley W. Mudd Building

Dept. of Electrical Engineering
Columbia University , NY 10027, USA
Office: 812 Schapiro Research Bldg.
Phone: (212) 854-8174
Email: predrag@ee.columbia.edu
URL: http://www.ee.columbia.edu/∼predrag

Last lecture: Tree-based methods

Decision trees

- Models for both: regression and classification

- Idea:
  - Segment the predictor space (the set of possible values for $X_1, \ldots, X_p$) into distinct and non-overlapping regions, $R_1, \ldots, R_j$
  - Regression (classification): Average (majority vote) over segments

Advantages and Disadvantages of Trees

Interpretation vs. Prediction accuracy

Advantages:

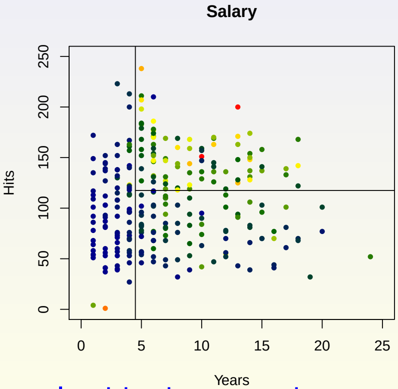
- Easy to explain to people.
  Even easier than linear regression!
- Some believe that decision trees mirror human decision-making.
- Can be graphically displayed and interpreted by non-experts.

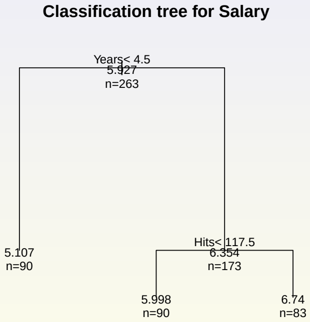
Disadvantages:

- Lower predictive accuracy than other methods.
- Not robust/sensitive: small changes in data can result in large changes in the final decision tree.

Example: Hitters

- Predict `Salary` based on `Years` and `Hits`
- Remove missing values and apply log-transform
- Salary encoding from low to high: blue - green - yellow - red





- Intuitive interpretation
- Prediction

Last lecture: Segmentation

- In general, the regions can have any shape
- Focus on high-dimensional rectangles (boxes)

- Goal: Find $R_1, \ldots, R_J$ that minimize the RSS:

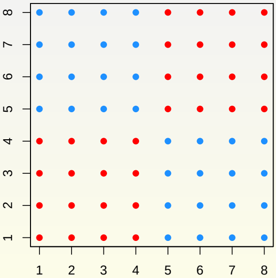
$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

  where $\hat{y}_{R_j}$ is the mean response for the observations in R_j
- Computationally infeasible to consider all partitions

- Top-down, greedy approach: Binary splitting
- Stopping criteria (e.g., max number of observations in a box)

Last lecture: Cannot grow trees sequentially

- Optimal tree size?
  - Training error decreases as the size increases
  - Testing error decreases, but then increases
- Grow the tree only if RSS decreases – poor results
- Example: 2 vales: red and blue
  always same RSS regardless of the cut
  but the next cut - there is (!)



- Alternative: Grow the tree to a large size and then trim/prune it

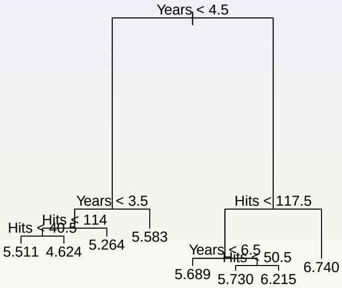
Last lecture: Tree pruning

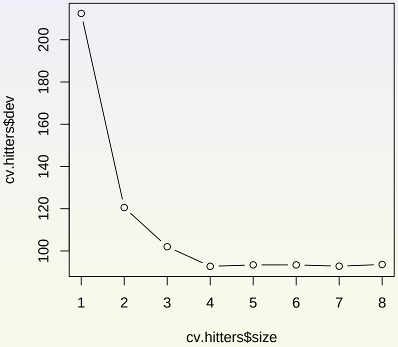
- Start with a large tree T_0
- Cost complexity pruning (weakest link pruning)
- Sequence of trees indexed by α
- For each α:

$$\min_{T \subseteq T_0} \quad \left\{ \sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T| \right\},$$

  where
  - $|T|$ is the number of leafs in T
  - R_m is the box corresponding to the mth leaf
  - $\hat{y}_{R_m}$ is the mean of training observations in R_m
- Parameter α
  - Controls the complexity/fit tradeoff
  - Select $\hat{\alpha}$ using cross-validation

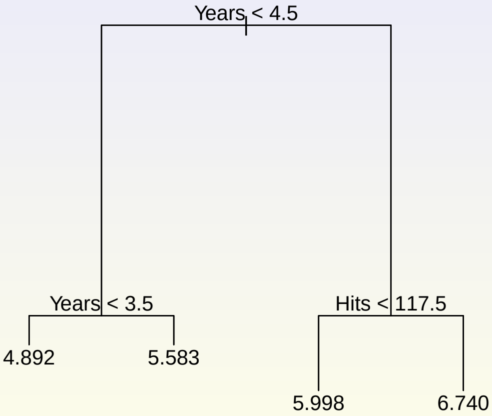
Example: Hitters - build a large tree





Example: Hitters - prunning

```
> prune.hitters<-prune.tree(hitters.fit,best=cv.hitters$size[which.min(cv.hitters$dev)])
```



Last lecture: Classification trees

- Similar to regression trees
- Predict a qualitative response
- Prediction within a box: most commonly occurring class
- Need an alternative to RSS

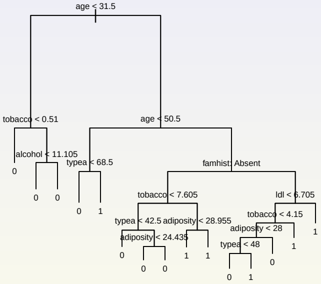
Last lecture: Error measures for classification

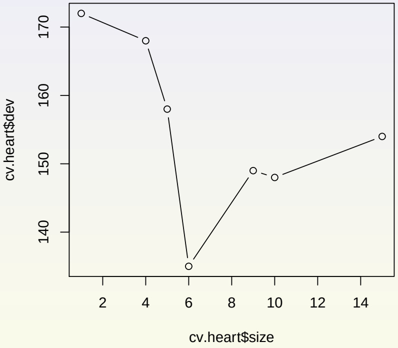
- $\hat{p}_{m,k}$ – proportion of training observations in the mth box that are from class k
- Minimize one of the following measures
  - Classification error rate
    $$E = 1 - \max_k \hat{p}_{m,k}$$
  - Gini index
    $$G = \sum_{k=1}^{K} \hat{p}_{m,k}(1 - \hat{p}_{m,k})$$
  - Entropy (or, deviance$= 2D$)
    $$D = -\sum_{k=1}^{K} \hat{p}_{m,k} \log \hat{p}_{m,k}$$

Example: SA heart disease data - build a large tree

```
> heart.tree<-tree(chd~.,data=SAheart)
> summary(heart.tree)

Classification tree:
tree(formula = chd ~ ., data = SAheart)
Variables actually used in tree construction:
[1] "age"       "tobacco"   "alcohol"   "typea"     "famhist"   "adiposity" "ldl"
Number of terminal nodes:  15
Residual mean deviance:  0.8733 = 390.3 / 447
Misclassification error rate: 0.2078 = 96 / 462

> set.seed(1)
> cv.heart<-cv.tree(heart.tree,FUN=prune.misclass)
> cv.heart
$size
[1] 15 10  9  6  5  4  1

$dev
[1] 154 148 149 135 158 168 172

$k
[1] -Inf    0    1    3    8   10   12

$method
[1] "misclass"

attr(,"class")
[1] "prune"         "tree.sequence"
```

Example: SA heart disease data - after prunning

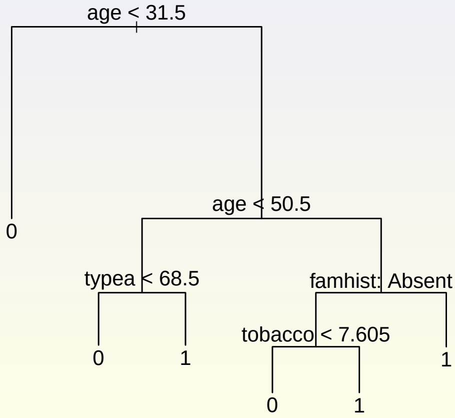




Example: South African heart disease set

```
> heart.prune<-prune.misclass(heart.tree,best=cv.heart$size[which.min(cv.heart$dev)])
> heart.predict<-predict(heart.prune,data=SAheart,type="class")
> table(heart.predict,SAheart$chd)
```

```
heart.predict   0   1
            0 266  70
            1  36  90
```



Bumping

Works for both: classifiers or regressions

- Stochastic search

  avoids getting stuck in a poor solution/local minimum

- Train a classifier or regression model $\hat{f}_0$ on $\boldsymbol{Z}$
- For $b = 1, \ldots, B$:
  1. Draw a bootstrap sample $\boldsymbol{Z}^{*b}$ of size n from training data
  2. Train a classifier or regression model $\hat{f}_b$ on $\boldsymbol{Z}^{*b}$
- Select the best model, e.g.,

$$\hat{b} = \arg\min_{0 \le b \le B} \sum_{i=1}^{n} \left(y_i - \hat{f}_b(\boldsymbol{z}_i) \right)^2$$

Bagging

Works for both: classifiers or regressions

- Bootstrap aggregation/averaging

  reduces the variance/overfitting

- For $b = 1, \ldots, B$:
  1. Draw a bootstrap sample $\boldsymbol{Z}^{*b}$ of size n from training data
  2. Train a classifier or regression model $\hat{f}_b$ on $\boldsymbol{Z}^{*b}$

- For a "new" point $\boldsymbol{x}_0$, compute:

$$\hat{f}_{\text{avg}}(\boldsymbol{x}_0) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}_b(\boldsymbol{x}_0)$$

  - Regression: $\hat{f}_{\text{avg}}(\boldsymbol{x}_0)$ is the prediction
  - Classification: Pick majority

- Example: Bagging trees

Random Forests

Works for both: classifiers or regressions

- Improvement over bagged trees
- Idea: Decorrelated trees
  - Still learn a tree on each bootstrap set
  - To split a region, consider only a subset of predictors/covariates
- Input parameter: $m \leq p$, often $m \approx \sqrt{p}$
- For $b = 1, \ldots, B$
  - Draw a bootstrap sample $\boldsymbol{Z}^{*b}$ of size n from the training data
  - Train a tree classifier on $\boldsymbol{Z}^{*b}$, each split is computed as:
    - Randomly select m predictors/covariates, newly chosen for each b
    - Make the best split restricted to that subsets of covariates
- Similarly as in bagging: for regression prediction

$$\hat{f}_{\text{avg}}(\boldsymbol{x}_0) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b(\boldsymbol{x}_0);$$

for classification: choose the majority vote among B classifiers.

Example: South African heart disease set

```
> library(randomForest)
> set.seed(10)
> train<-sample(1:nrow(SAheart),nrow(SAheart)/2)
> bag.heart<-randomForest(chd~., data = SAheart, subset=train, mtry=9, importance=TRUE)
> bag.heart

Call:
 randomForest(formula = chd ~ ., data = SAheart, mtry = 9, importance = TRUE,      subset = train)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 9

        OOB estimate of  error rate: 36.8%
Confusion matrix:
    0  1 class.error
0 123 31   0.2012987
1  54 23   0.7012987
> table(SAheart$chd[-train],predict(bag.heart, newdata = SAheart[-train,]))

      0   1
  0 118  30
  1  54  29
>
> bag.heart<-randomForest(chd~., data = SAheart, subset=train, mtry=3, importance=TRUE)
> bag.heart

Call:
 randomForest(formula = chd ~ ., data = SAheart, mtry = 3, importance = TRUE,      subset = train)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 33.33%
Confusion matrix:
    0  1 class.error
0 134 20   0.1298701
1  57 20   0.7402597
> table(SAheart$chd[-train],predict(bag.heart, newdata = SAheart[-train,]))

      0   1
  0 128  20
  1  58  25
```

Boosting

Slow learning

- General approach that can be applied to many ML methods
- Focus on trees
  - Works for both regression and classification
- Similar to Random Forests
  - Make a family of weak learners
  - Then, the solution is a combination of many of these weak learners, $\hat{f}^1, \cdots, \hat{f}^B$
  - Note, boosting does not use the bootstrap samples
- Idea:
  - Build trees sequentially in small increments by adding weak learners
  - Use the previous tree and residuals to create a new one

Boosting Algorithm for Regression

1. Set $\hat{f}(\boldsymbol{x}_i) = 0$ and $r_i = y_i$ for all i in the training set
2. For $b = 1, \ldots, B$, repeat:
   2.1 Fit a tree $\hat{f}_b$ with d splits ($d + 1$ terminal nodes) to training data $(\boldsymbol{X}, \boldsymbol{r})$, $\boldsymbol{r} = (r_1, \ldots, r_n)$
   2.2 Update $\hat{f}$ by adding in a shrunken version of the new tree:
   $$\hat{f}(\boldsymbol{x}) \leftarrow \hat{f}(\boldsymbol{x}) + \lambda \hat{f}^b(\boldsymbol{x})$$
   2.3 Update residuals:
   $$r_i \leftarrow r_i - \lambda \hat{f}_b(\boldsymbol{x}_i)$$
3. Output the boosted model:
$$\hat{f}(\boldsymbol{x}) = \sum_{b=1}^{B} \lambda \hat{f}_b(\boldsymbol{x})$$

- Notes:
  - λ is a small positive number (e.g., 0.01 or 0.001)
  - Often $d = 1$ works

Example: Hitters

```
> library(gbm)
> set.seed(1)
> train<-sample(1:nrow(myHitters),nrow(myHitters)/2)
> hitters.boost<-gbm(Salary~.,data = myHitters[train,], distribution = "gaussian", n.trees = 5000,
       interaction.depth = 4 )
> hatSalary<-predict(hitters.boost, newdata=myHitters[-train,], n.trees = 5000)
> mean((myHitters$Salary[-train] - hatSalary)^2)
[1] 0.1802494
>
> hitters.boost<-gbm(Salary~.,data = myHitters[train,], distribution = "gaussian", n.trees = 5000,
       interaction.depth = 4, shrinkage = 0.1 )
> hatSalary<-predict(hitters.boost, newdata=myHitters[-train,], n.trees = 5000)
> mean((myHitters$Salary[-train] - hatSalary)^2)
[1] 0.2967324
```

Classification: AdaBoost

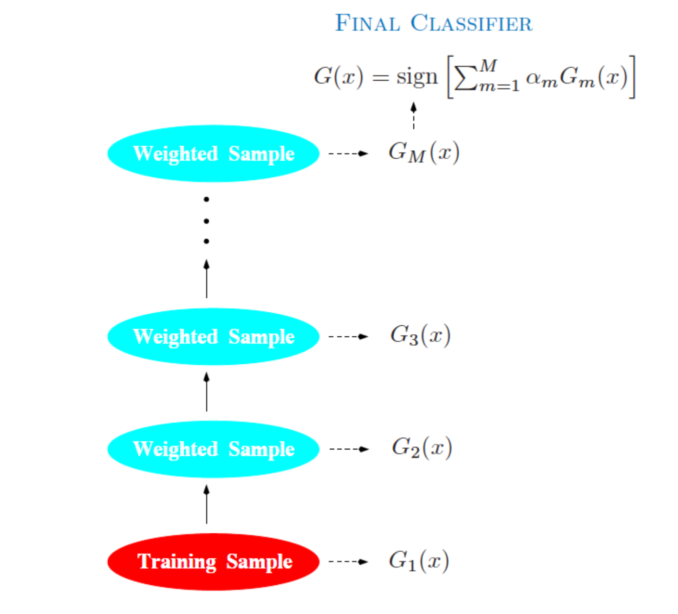
- Due to Freund and Schapire (1997)
- Consider two classes: $Y \in \{-1, 1\}$
- For a classifier $G(x) \in \{-1, 1\}$, the training error is

$$\mathrm{e}_m = \frac{1}{n} \sum_{i=1}^{n} 1_{\{y_i \neq G(x_i)\}}$$

- Main idea: construct weak classifiers
  - Weak classifier: slightly better than random guessing (50% error)
  - Sequentially, construct weak classifiers, $G_m(x)$, on modified training data.
- Final classifier, combination of weak classifiers through a weighted majority vote

$$G(x) = \mathrm{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$$

Classification: AdaBoost schematic



Classification: AdaBoost

1. Set $w_i = 1/n, i = 1, 2, \ldots, n$, where n is the number of training points.
2. For $m = 1, \ldots, M$, repeat:
   (a) Fit a (weak) classifier $G_m(x)$ to training data using wights w_i.
   (b) Compute the weighted error

$$\mathsf{e}_m = \frac{\sum_{i=1}^{n} w_i 1_{\{y_i \neq G_m(x_i)\}}}{\sum_{i=1}^{n} w_i}$$

   (c) Compute $\alpha_m = \log((1 - \mathsf{e}_m)/\mathsf{e}_m)$.
   (d) Update

$$w_i \leftarrow w_i \exp(\alpha_m 1_{\{y_i \neq G_m(x_i)\}}), \quad , i = 1, 2, \ldots, n.$$

3. Final classifier

$$G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$$

Support Vector Machines

Classification methods we learned:

- Logistic
- Gaussian discriminant analysis
- Tree-based

Support Vector Machines (SVM)

- Idea: separate points using surfaces
  - linear surfaces - hyperplanes
  - general nonlinear surfaces

Three types:

- Maximal Marginal Classifier - separator = hyperplane
- Support Vector Classifier - separator = "soft" hyperplane
- Support Vector Machine - separator = nonlinear surface

Hyperplane separator

Hyperplane in p-dimensions is defined by

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = 0$$

Examples:

- 2D hyperplane = line

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

- 3D hyperplane = regular plane

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 = 0$$

Classification decision: Assign $\boldsymbol{x}$ in one of the two classes depending on the sign of

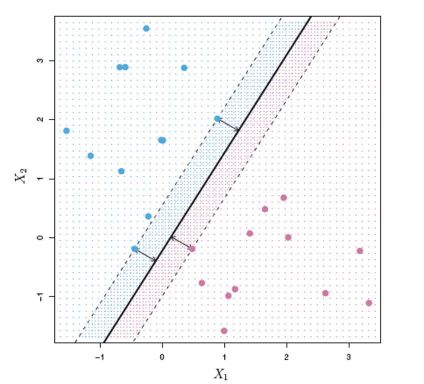
$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p >< 0$$

How do we choose a hyperplane, i.e., $\boldsymbol{\beta}$?

The Maximal Margin Classifier

Optimal hyperplane

- *Margin*: Distance from an observation to the hyperplane
- *Maximal margin hyperplane*: One whose smallest margin is maximal
- *Support vectors*: points that support the maximal margin hyperplane



Understanding geometry: Distance from a hyperplane

Consider a hyperplane in p-dimensions

$$\beta_0 + \langle \boldsymbol{\beta}, \boldsymbol{x} \rangle = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = 0$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$ and $\boldsymbol{x} = (x_1, \ldots, x_p)$

- **Claim:** $\boldsymbol{\beta}$ is a perpendicular vector to this hyperplane

  Proof: Let $\boldsymbol{x}$ and $\boldsymbol{x}'$ be two points on this hyperplane. Then

  $$\beta_0 + \langle \boldsymbol{\beta}, \boldsymbol{x} \rangle - (\beta_0 + \langle \boldsymbol{\beta}, \boldsymbol{x}' \rangle) = \langle \boldsymbol{\beta}, (\boldsymbol{x}' - \boldsymbol{x}) \rangle = 0$$

- Perpendicular unit vector

  $$\frac{\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|}$$

  where $\|\boldsymbol{\beta}\| \equiv \|\boldsymbol{\beta}\|_2$ is the usual euclidian norm.

- Signed distance to the hyperplane: Let $\boldsymbol{x}_0$ be a point outside the hyperplane and $\boldsymbol{x}$ inside (note $\langle \boldsymbol{\beta}, \boldsymbol{x} \rangle = -\beta_0$)

  $$\frac{\langle \boldsymbol{\beta}, (\boldsymbol{x}_0 - \boldsymbol{x}) \rangle}{\|\boldsymbol{\beta}\|} = \frac{\langle \boldsymbol{\beta}, \boldsymbol{x}_0 \rangle + \beta_0}{\|\boldsymbol{\beta}\|}$$

Maximum Margin Classifier: Formal computation

Let the two classes be $y_i \in \{1, -1\}$
Then, the Maximal margin hyperplane is the solution to

$$\max_{\beta_j, M} M \tag{1}$$

$$\text{subject to} \sum_{1}^{p} \beta_j^2 = 1 \tag{2}$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M, \qquad \forall i = 1, \ldots, n \tag{3}$$

Notes:

- (3) requires that each observation is on the correct side of the hyperplane
- The solution M^* is the maximal margin

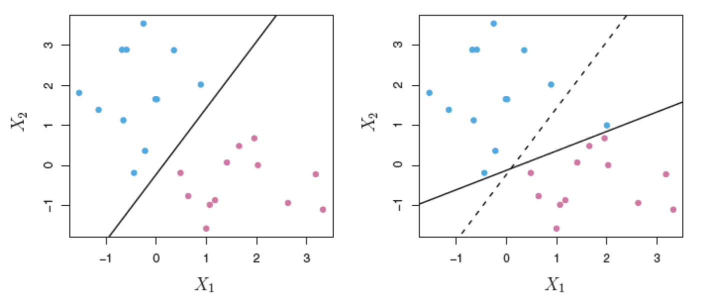
Problems

Non-separable case

- There is no hyperplane that separates the two classes

Highly sensitive to support vectors

- The hyperplane moves if one moves or introduces new support vector points



Need a "softer" separator

Support Vector Classifier

- Greater robustness to individual observations
- More general: Works for most points

The hyperplane is the solution to

$$\max_{\beta_j, \epsilon_j M} M \tag{4}$$

$$\text{subject to} \sum_{1}^{p} \beta_j^2 = 1 \tag{5}$$

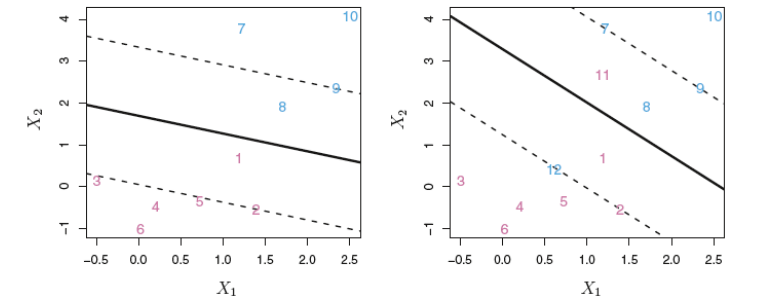
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M(1 - \epsilon_i), \qquad \forall i \tag{6}$$

$$\epsilon_i \geq 0, \sum_{i=1}^{n} \epsilon_i \leq C \tag{7}$$

Notes:

- ϵ_i - *slack variables*
  - $\epsilon_i = 0$ - i-th observation on the correct side of the margin
  - $0 < \epsilon_i < 1$ - i-th observation on the wrong side of the margin
  - $\epsilon_i > 1$ - i-th observation on the wrong side of the hyperplane
- C - budget for slackness

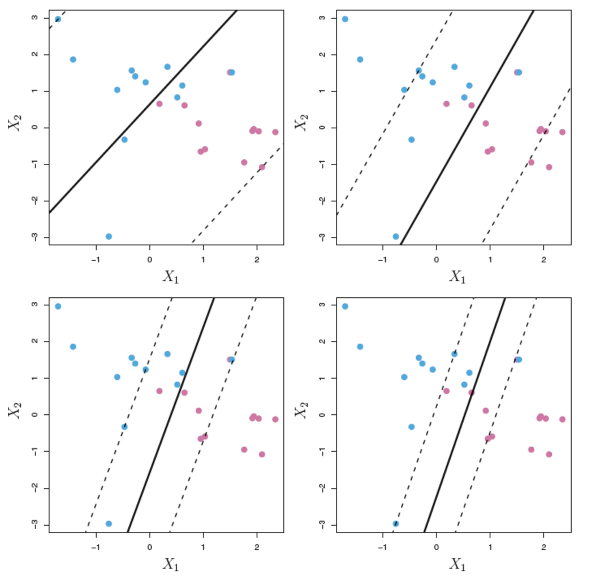
Example



- Left:
  - Purple observations: 3,4,5, and 6 = correct side of the margin; 2 is on the margin, and 1 is on the wrong side of the margin.
  - Blue observations: 7 and 10 = correct side of the margin; 9 is on the margin, and 8 is on the wrong side of the margin.
- Right: Same as left panel with two additional points, 11 and 12. 11 and 12 = wrong side of both the hyperplane and the margin.
- Robustness: the hyperplane on the right did not move much (!)

Example: Decreasing C

Decresing C: From top left to bottom right



Support Vector Machines

Nonlinear decision boundary - example

- p features: $X_1, X_2, \ldots, X_p$
- expand bases - $2p$ features: $X_1, X_1^2, X_2, X_2^2, \ldots, X_p, X_p^2$

Fir the hyperplane to the expanded bases

$$\max_{\beta_j, \epsilon_j M} M \tag{8}$$

$$\text{subject to} \sum_{1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1 \tag{9}$$

$$y_i(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{p} \beta_{j2} x_{ij}^2) \geq M(1 - \epsilon_i), \qquad \forall i \tag{10}$$

$$\epsilon_i \geq 0, \sum_{i=1}^{n} \epsilon_i \leq C \tag{11}$$

- The solutions are in general nonlinear since these are quadratic expressions
- The SVM explores further this idea (using kernels)

Support Vector Machines

Let $\langle a, b \rangle$ be the inner product

$$\langle a, b \rangle := \sum_{i=1}^{p} a_i b_i$$

It turns out that the SV classifier can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$$

with n training parameters, α_i, one per training observation.

- Estimating α_i and β_0: requires $\binom{n}{2}$ inner products $\langle x_j, x_i \rangle$
- α_i is nonzero only for the support vectors (!)
- Let $\mathcal{S}$ be the set of these support points: $|\mathcal{S}| =$ small number
- $f(x)$ simplifies to

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x, x_i \rangle$$

Support Vector Machines: Kernels

Recall, **Kernel** - $K(x_i, x_j)$ is an inner product over expanded basis $x \to \phi(x) : \mathbb{R}^p - \mathbb{R}^d, d > p$, i.e., $K(x_i, x_j) = \langle \phi(x_j), \phi(x_i) \rangle$
Nice property: *we can use Kernel solutions without ever knowing* $\phi(x)$

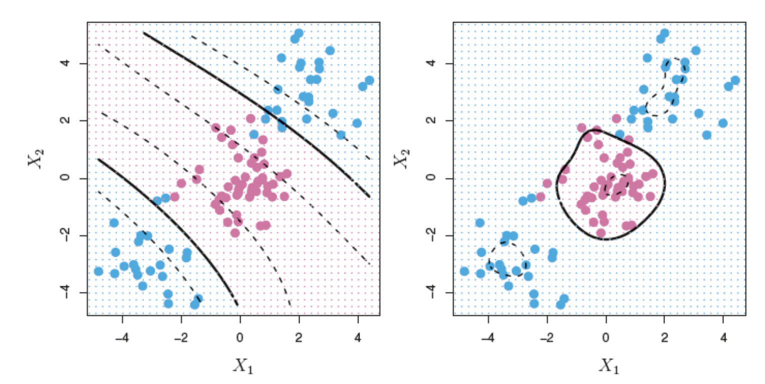
Kernels:

- *Linear*
$$K(x_i, x_j) = \langle x_j, x_i \rangle$$
- *Polynomial* - for positive integer d
$$K(x_i, x_j) = (1 + \langle x_j, x_i \rangle)^d$$
- *Radial* - for $\gamma > 0$
$$K(x_i, x_j) = \exp\left(-\gamma \sum_{k=1}^{p} (x_{ik} - x_{jk} \rangle)^2\right)$$

The resulting classifier is known as **SVM**, with the decision function taking the following nonlinear form in general

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x, x_i)$$

Example



- Left: An SVM with a polynomial kernel of degree 3
- Right: An SVM with a radial kernel
- Either kernel is capable of capturing the decision boundary

Reading on Bootstrap Improvements and Boosting: Bumping, Bagging, Random forests, Boosting

ISL: Section 8.2

ESL: Sections, 8.7, 8.9, 10.1; More advanced reading: Chapters 10 and 15

Reading on Support Vector Machines

ISL: Sections 9.1-9.3

ESL: Section 12.1-12.3 (more advanced)

Homework: HW3 is due: Wed, Oct 19, 11:59pm.
Since this is the last HW, **no late submission will be permitted**, in order to release the solutions on Thu, Oct 20, to give you time to prepare for **midterm**.
Midterms date: Oct 25, during class time.
Closed book, no electronic devices, but one page - both sides - with **handwritten** formulas/facts is permitted.