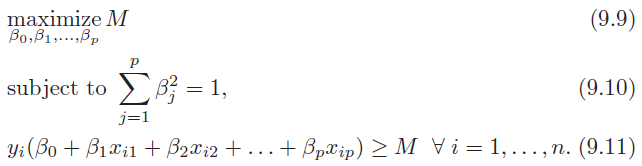
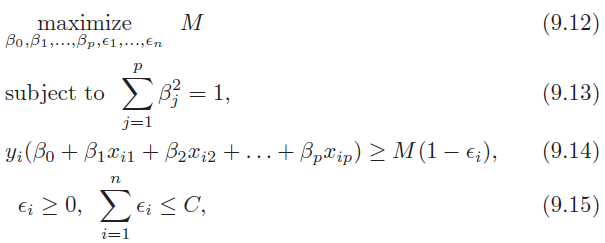
SVM

* Basic to Advanced:
  + Maximal Margin Classifier
  + Support Vector Classifier
  + Support Vector Machine

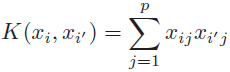
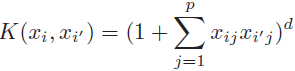
Maximal Margin Classifier

* Hyperplane –
  + Flat subspace of dimension (p-1) in dimension p
  + 2D – hyperplane = line
    - 
  + 3D – hyperplane = plane
* Determine which side of hyperplane point lies on by solving:
  + 
  + Left-side > 0 🡪 assign to class 1
  + Left-side < 0 🡪 assign to class -1
* Maximal margin classifier – farthest *minimum* distance to training observations
  + = large margin
  + Only works when there is a *linear* separating hyperplane
* **Support vectors** – points closest to hyperplane
  + At least one on each side of hyperplane
  + They “support” maximal hyperplane s.t 🡪 their moved slightly -- the max hyperplane will move as well
  + \*\*\* Maximal margin hyperplane on depends on small number of observations
* Maximal margin hyperplane – optimization problem:
  + 
    - M = maximal margin hyperplane
    - (9.11) –
      * yi = (-1 or 1)
      * ensures each point on right side of hyperplane
* Problems with maximal margin hyperplane:
  + Very sensitive to single data points 🡪 overfitting
  + Can’t work unless linear decision boundary

Support Vector Classifier

* 
  + M = width of margin
  + ε1, …, εn = slack variables
    - Allow individual observations to be on wrong side of margin/hyperplane
  + C = non-negative tuning parameter
    - Determines number and severity of violations willing to tolerate
    - C = 0 🡪 maximal margin hyperplane
    - Chosen via CV
    - (+) C 🡪 (+) margin 🡪 (+) violations
    - (-) C 🡪 (+) variance | (-) bias
* Classify test observations based on *sign of*:
  + 
* Observations that affect hyperplane (*support vectors*):
  + (1) lie on hyperplane
  + (2) violate the margin
  + \*\*\* observations correctly classified does NOT affect **support vector classifier**

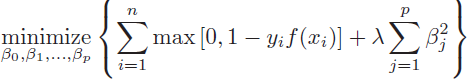
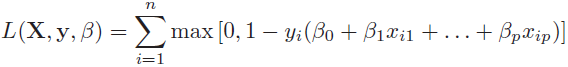
Support Vector Machine

* Support Vector Classifier 🡪 w/ non-linear decision boundary
* Enlarge feature space using *kernels*
  + Linear kernel – Support Vector Classifier
    - 
  + Nonlinear kernel – Support Vector Machine
    - 
* Solve function - 
  + αi = 0 (if not a support vector)
* Limited to only 2 classes

Extending SVM to +2 classes

* 2 methods:
  + (1) One-versus-one
  + (2) One-versus-all

Relationship to Logistic Regression

* Support vector classifier optimization function can be rewritten as:
  + 
  + λ = non-negative tuning parameter
  +  = **Ridge Regression** penalty term
  + Loss function = 
    - **Hinge loss**
    - ONLY observations on wrong side contribute to loss function
      * Logistic regression – all observations contribute
* Logistic vs SVM:
  + Classes well separated – SVM
  + Overlapping regimes – Logistic
* Support Vector Regression –
  + Minimize residuals only larger in absolute value than some positive constant (C)