**Autores**

**JAMES – CAPITULO 8**

**HASTIE – CAPITULO 9, 10**

**Clasificadores Combinados y Métodos a Base de Árboles**

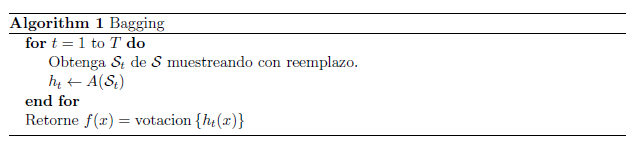
**Bagging (Boostrap)**

Técnica para mejorar precisión es estimativos/parámetros

Remuestreo con reemplazo

Los clasificadores se entrenan con muestras bootstrapeadas de los datos de entrenamiento

Algoritmo Bagging



**Boosting**

Aprendibilidad PAC: encontrar h que satisfaga para cualquier distribución de datos

Condición de aprendibilidad débil

Algoritmo Boosting Fuerte

Texto

Descripción generada automáticamente

Algoritmo Boosting Adaptativo

Dada la distribución de datos con y

Asociar los datos a un vector de pesos D con una distribución

Clase de hipótesis base:

Error pesado de una hipótesis de acuerdo con D

Asumir acceso a aprendizaje débil A que recibe y retorna

AdaBoost procede en serie de rondas en las que obtiene hipótesis

En la primera ronda se llama a A con la distribución uniforme

En la siguiente ronda se modifica la distribución D

Se itera el procedimiento cambiando los pesos D en cada ronda a partir de la ronda anterior

**AdaBoost**

Construir para minimizar

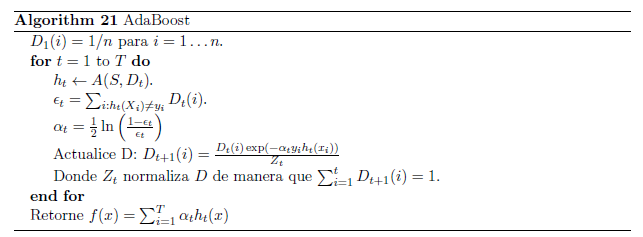
Encontrar tal que minimicen

+

Los pesos aprendizaje débil

Con h fija se encuentra derivando e igualando a cero

**Algoritmo AdaBoost**

****

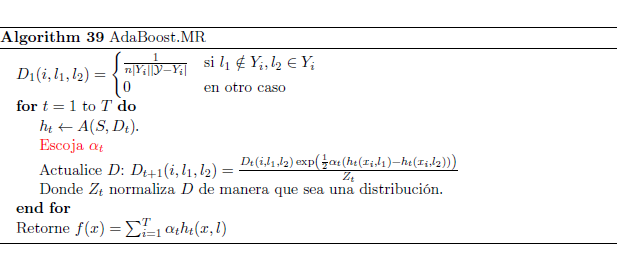
El error pesado de con respecto a

Si el error empírico decrece exponencialmente

Si el error empírico llega a cero en un numero finito de pasos

AdaBoost minimiza función de costo del margen

**Algoritmo AdaBoost.MR**

****

**Gradient Boosting**

Clases de funciones

Espacio de combinaciones lineales span

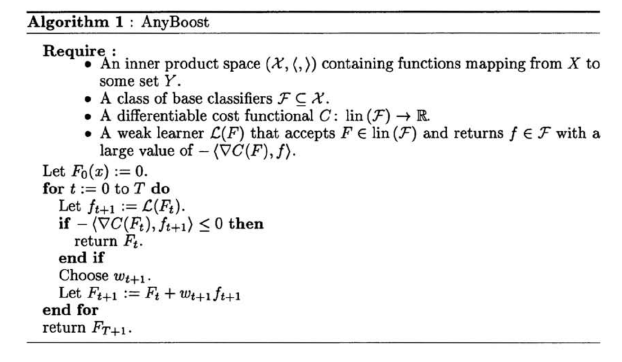
Producto punto en span F

Funcional de costo a minimiza

Dada encontrar f tal que

Buscar dirección f que maximiza descenso

**Algoritmo Gradient Boosting**

****

Los modelos a base de árboles consisten en segmentar (estratificar) el espacio de variables independientes/predictores en regiones

Para generar la predicción se usa el promedio(regresión) o moda(clasificación) entre las regiones

**Arboles de Decisión**

**Regresión**

1. Dividir espacio de variables independientes (predictores) en R regiones no sobrelapadas
2. Para cada observación en región se hace la misma predicción, que es el promedio de la respuesta de los datos de entrenamiento en

El espacio de variables independientes/predictores en rectángulos o cajas

Encontrar cajas/subespacios de regiones que minimicen RRS

respuesta promedio de datos de entrenamiento en región J

Aplicar Splitting Binario (Recursivo): Inicia en la cima del árbol (región donde todas las observaciones están en una única región inicial) y hace splits del espacio de variables independientes (espacio de predictores)

1. Seleccionar variable independiente/predictor y punto de corte tal que hacer Split del espacio de variables independientes que minimice el RRS  
   Considerar todas las variables independientes y todos los posibles valores de punto de corte para cada variable, escoger predictor y punto de corte que minimice el RRS

En cada (binary) Split escoger valor de variable y punto de corte que minimice:

es promedio de respuesta de datos de entrenamiento en región

es promedio de respuesta de datos de entrenamiento en región

1. Iterar en el proceso de (binary) Split de las regiones, encontrando la variable y punto de corte que minimice RRS en cada iteración
2. Criterio de parada (hiper parámetros)
3. Con las regiones generadas por Split binario se predice la respuesta para los datos de prueba usando el promedio de los datos de entrenamiento en la región en la que pertenecen los datos de prueba

**Prunning (Podar)**

Generar árbol con menos splits para reducir varianza y mejorar interpretación, con costo de mayor sesgo

Generar árbol muy profundo y luego podar(prune) para generar subárbol

Prunning Costo Complejidad (weakest link pruning) subárboles sintonizados por parámetro

**Algoritmo de Árbol de Regresión con Prunning Costo-Complejidad**

1. Aplicar (binary) splitting para generar árbol profundo con los datos de entrenamiento. Criterio de parada cuando regiones terminales/hojas tengan mínimo número de observaciones
2. Aplicar prunning Costo-Complejidad a árbol profundo para obtener una secuencia de subárboles como función de parámetro
3. Aplicar técnica de remuestro K-fold Validación Cruzada para sintonizar parámetro

* Dividir datos de prueba en K Folds. Para cada Fold
* Aplicar 1 (binary splitting exhaustivo) y 2 (pruning) para folds
* Evaluar la función de error MSE en los datos con el fold, como función de
* Calcular promedio del valor de parámetro y escoger que minimiza el error promedio

1. Escoger subárbol de 2. Con parámetro con menor MSE promedio

El parámetro de prunning controla el trade-off entre la complejidad de subárbol y el ajuste con los datos de entrenamiento

Con el subárbol es igual al árbol profundo de base

Cuando aumenta se podan las ramas de los arboles

Se selecciona con menor MSE promedio en los datos de prueba por remuestreo con Validación Cruzada

**Similar a regularización Lasso** para controlar complejidad de modelo y controlar sobreajuste a datos de entrenamiento

**Clasificación**

En clasificación la predicción de una observación se genera por la moda de la clase más frecuencia en los datos de prueba en la región que pertenece

Calcular la proporción de clase de datos de entrenamiento en una región

En clasificación se busca minimizar la función de Proporción de Error de Clasificación en cada Split

Proporción de Error de Clasificación: fracción de datos de prueba en región que no pertenecen a la clase más común

**proporción de datos de entrenamiento en región que pertenecen a clase**

En la práctica se usan 2 funciones de Error para generar árboles de clasificación:

Indice de Gini

Varianza total de las clases

Medida de pureza de nodo

Valores que tienden a 0 indican que nodo tienen observaciones de una única clase predominante

Entropía Cruzada

Para clasificación se usan Índice de Gini o Entropía Cruzada para minimizar función de error en cada Split

* Arboles de Decisión son fáciles de interpretar
* Se asemejan a procesos de decisión de humanas
* Interpretación grafica
* Soportan variables independientes categóricas sin necesidad de crear variables dummies (codificación)
* Arboles de decisión tienen performance bajo
* Alta varianza (cambio de pesos/parámetros a variaciones de datos de entrenamiento)

**Bagging (Bootstrap Agregación)**

Aplicar técnicas de Bootstaping (remuestreo con reemplazo) sobre los datos de entrenamiento para reducir la varianza del modelo

Promediar un set de las observaciones reduce la varianza

Generar múltiples sets de datos de entrenamiento, ajustar modelo con cada set Bootstrap y promediar los resultados de las predicciones, para obtener un único modelo de baja varianza

Generar B sets/muestras de los datos de entrenamiento Bootstrappeadas (remuestreo con reemplazo). Ajustar modelos con B sets Bootstrapped de datos de entrenamiento para obtener , promediar las predicciones

Arboles no tienen Prunning α

Cada árbol de Bagging tiene alta varianza, pero sesgo bajo

Promediar los B arboles reduce la varianza

Para clasificación se toma la moda o clase más frecuente predicha por los B arboles

Predicciones de árboles de Bagging son altamente correlacionadas

**Estimación del Error Out-of-Bag**

Los árboles ajustados por Bagging (Bootstrapping) usan de datos de entrenamiento

de datos de entrenamiento no usados durante el entrenamiento son Out-of-Bag

**Random Forest (random Split de m variables de p)**

Decorrelacionar los arboles.

Quitar correlación entre B arboles

Quitar correlación entre predicciones

Quitar similitud entre la estructura de los arboles

Aplicar Bootstrapping sobre datos de entrenamiento para generar arboles

Al aplicar el Binary Split se considera una muestra aleatoria (random sample) de las m variables independientes(predictores) en vez de usar todos los p variables

Como regla general se usa un nuevo sample de variables en cada split

En cada Split el modelo Random Forest no puede considerar la mayoría de las variables

Todos los árboles Bagging(Bootstrapping) son parecidos porque escogen primero las mismas variables fuertes. Predicciones son altamente correlacionadas

En Random Forest en promedio de los Splits al ajustar cada árbol no va a considerar el predictor fuerte en el random sample de variables m

Mejora el desempeño del ajuste para variables correlacionadas

**Boosting**

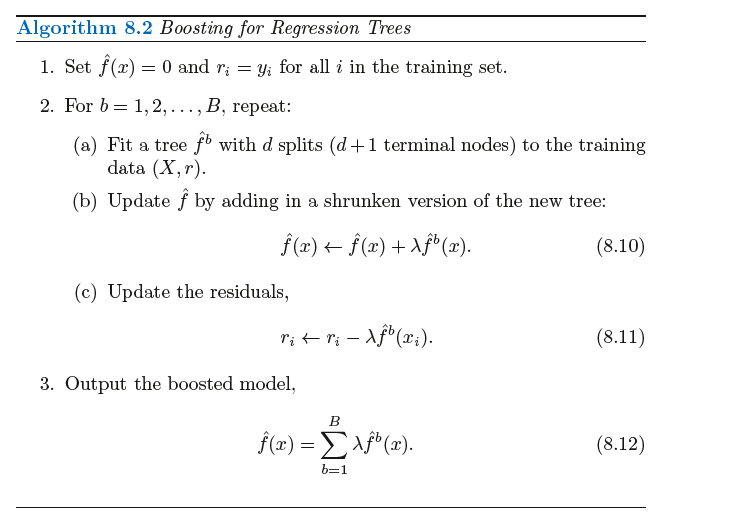
Cada árbol se genera secuencial

Cada árbol se genera usando información de aprendizaje de la iteración anterior

NO aplica Boostraping de los datos de entrenamiento

Cada árbol se ajusta con una versión modificada de los datos de entrenamiento originales

Algoritmo Boosting



Aumentar gradualmente el modelo del árbol a partir del cálculo de residuales en interacciones pasadas

Mejorar desempeño de regiones con mal desempeño

Actualizar función de residuales (error) en la predicción

En Boosting las siguientes iteraciones de ajuste dependen significativamente de iteraciones anteriores

**Parámetros de Boosting**

1. Numero de árboles: Se puede sobreajustar si el número de árboles es muy alto. Aplicar remuestro por Validación Cruzada para seleccionar n número de arboles
2. tasa de aprendizaje (parámetro de encogimiento): Controla el ritmo de aprendizaje. Valores típicos entre .  
   Valores bajos de requieren muchos arboles
3. Numero de Split d: Control de complejidad de árbol de ensamble

Se puede sobreajustar si arboles son muchos