Agregación de Modelos

1EST17 - Aprendizaje Estadístico I

Mg. Enver Gerald Tarazona Vargas

enver.tarazona@pucp.edu.pe

Maestría en Estadística

Escuela de Posgrado

Resumen I

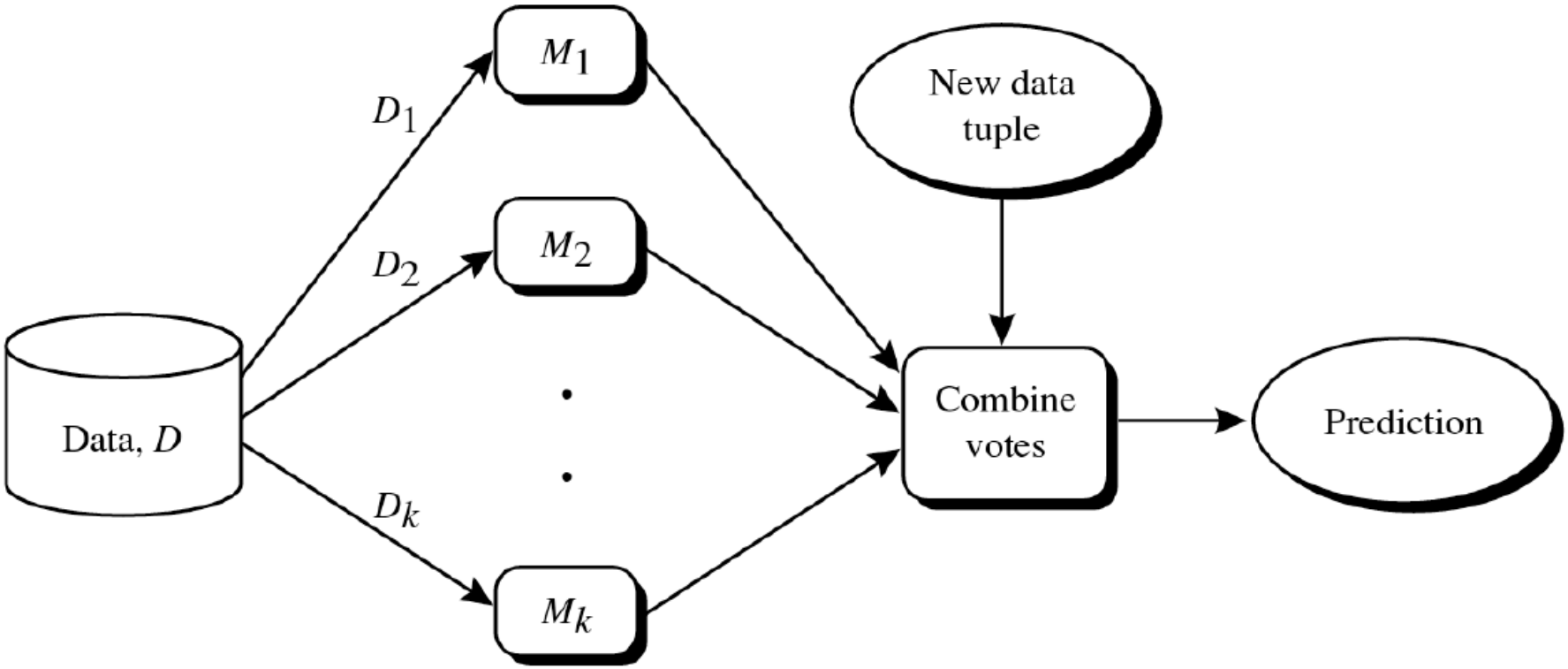
Introducción

- Métodos conjuntos (ensemble methods).
- Modelo compuesto formado por una combinación de clasificadores.
- Los clasificadores individuales votan y la clase predicha se basa en la colección de votos.
- Estos métodos tienden a ser más exactos que los clasificadores que lo componen.

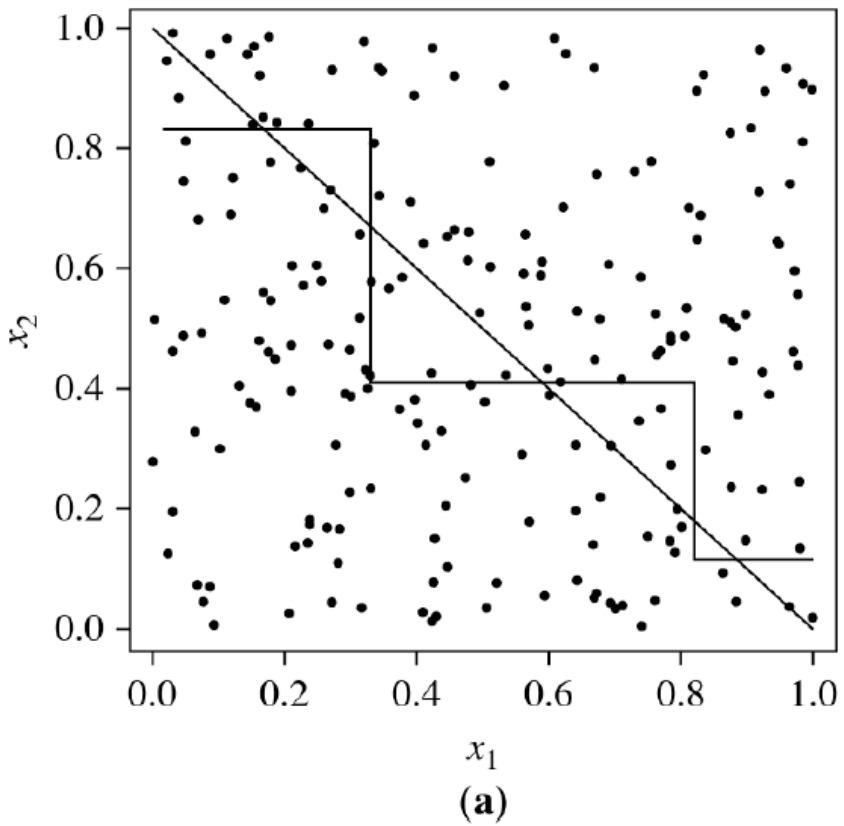
Introducción

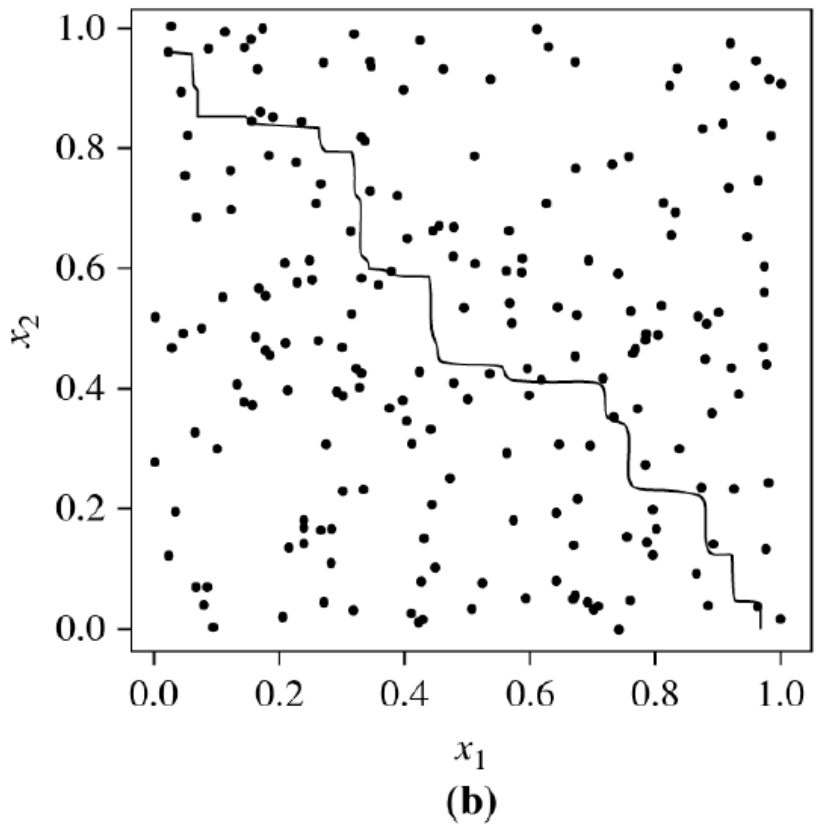
- Los clasificadores conjuntos combinan una serie de k clasificadores, $M_1, M_2, ..., M_k$ con el objeto de crear un modelo de clasificación compuesto $M*$.
- Un conjunto de datos D se usa para crear k conjuntos de datos de trabajo $D_1, D_2, ..., D_k$, donde D_i $(1 \leq i \leq k-1)$ se usa para generar el clasificador M_i.
- Dado un nuevo registro, cada clasificador entrega una clase predicha.
- El conjunto entrega su predicción basado en los votos de los clasificadores bases.

Introducción



Introducción





Comentarios Generales

La idea básica de este método puede ejemplificarse en base a un diagnóstico médico.

- Pedir más de una opinión y quedarse con el diagnóstico que ocurre más frecuentemente.
- El diagnóstico se basa en el voto de la mayoría, donde cada doctor entrega un voto.
- Una votación en base a un grupo más grande de especialistas puede ser más confiable que una basada en un grupo menor.
- `Bagging = bootstrap aggregation.`

Comentarios Generales I

- Bootstrap es usado en muchas situaciones en las cuales es complicado o imposible calcular directamente la desviación estándar de una cantidad de interés.
- Algunos modelos, como los árboles de decisión, padecen por tener una gran varianza.
- Esto quiere decir que si nosotros dividimos los datos de entrenamiento en dos partes de manera aleatoria y ajustamos un árbol de decisión en ambas mitades, los resultados pueden ser bastante diferentes.
- *Bootstrap aggregation*, or, *bagging*, es un procedimiento que busca reducir la varianza en un método de aprendizaje estadístico.
- Recordar: Dada un conjunto de observaciones independientes $Z_1, \ldots, Z_n$, cada uno con varianza σ^2, la varianza de la media $\overline{Z}$ de las observaciones esta dada por $\frac{\sigma^2}{n}$.

Comentarios Generales II

- En otras palabras, el promediar un conjunto de observaciones reduce la varianza.
- Una forma natural de reducir la varianza e incrementar la precisión en la predicción de un método de aprendizaje estadístico es tomar muchas muestras de entrenamiento de la población, construir un modelo predictivo por separado usando cada conjunto de entrenamiento y promediar los resultados de las predicciones..
- En otras palabras, podemos calcular $\hat{f}^1(x), \hat{f}^2(x), \ldots, \hat{f}^B(x)$ usando B conjuntos de entrenamiento separados, y promediarlos para obtener un modelo de aprendizaje estadístico de baja varianza dado por:

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

Comentarios Generales III

- Obviamente este enfoque no es práctico porque generalmente no tenemos acceso a múltiples conjuntos de entrenamiento.
- De manera alternativa, podemos aplicar bootstrap, tomando muestras repetidas de un sólo conjunto de entrenamiento.
- Bajo este enfoque generamos B diferentes muestras de entrenamiento por bootstraping.
- Se entrena el método en la b-ésima muestra de entrenamiento para obtener $\hat{f}^{*b}(x)$ y finalmente promediar todas las predicciones obteniendo:

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x)$$

El cual es llamado bagging

Comentarios Generales IV

- El método anterior es utilizado en regresión. Para el caso de clasificación, se registra la clase predicha para cada observación en cada uno de los B árboles y se toma aquella con mayor frecuencia (voto mayoritario).

Bagging: Algoritmo I

Dado un conjunto D, de d unidades, el `bagging` funciona de la siguiente manera:

1. Para la iteración i $(i = 1, 2, \ldots, , k)$, se muestrea un conjunto D_i de d unidades con reemplazo de las unidades originales D. Cada conjunto de datos de trabajo es una muestra bootstrap.
2. Un modelo M_i es entrenado para cada conjunto de datos de trabajo D_i
3. Para clasificar una unidad desconocida **X**, cada M_i entrega su predicción , que cuenta como un voto.
4. El clasificador conjunto, $M*$, cuenta los votos y asigna a **X** la clase con mayor votación.

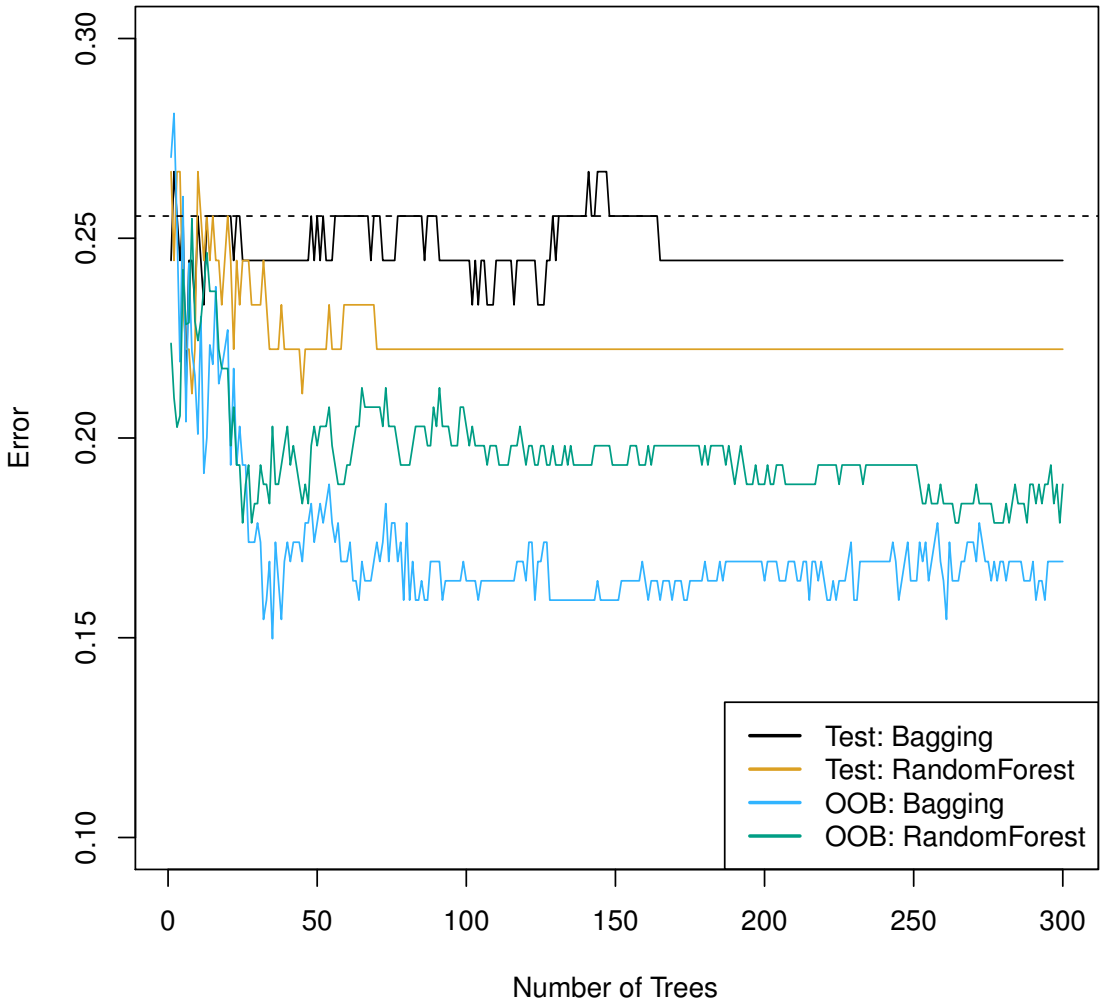
Error de Estimación *Out-of-Bag* I

- Hay una forma sencilla de estimar el error de estimación de un modelo agregado por bagging sin la necesidad de usar validación cruzada o usar el enfoque de usar una muestra de validación.
- Es posible demostrar que, en promedio, cada árbol agregado por bagging usa en promedio alrededor de dos tercios de las observaciones (ver James et al, capítulo 5).
- El tercio restante de las observaciones que no son usada para ajustar el árbol son conocidas como las observaciones out-of-bag (OOB).
- Podemos predecir la respuesta para la i-ésima observación usando cada uno de los árboles en que dicha observación estaba OOB.
- Esto da alrededor de $B/3$ predicciones para la i-ésima observación.
- Para obtener una predicción individual de la i-ésima observación se usa la estrategia de voto mayoritario.

Error de Estimación *Out-of-Bag* II

- La predicción OOB puede ser obtenida de esta forma para cada una de las n observaciones, para los cuales se calcula el error de clasificación.
- El error OOB resultante es una estimación válida del error de prueba para el modelo agregado por bagging, dado que la respuesta para cada observación es predicha usando solamente los árboles en los cuales la observación no fue usada para el ajuste.

Gráfica de Error en la clasificación



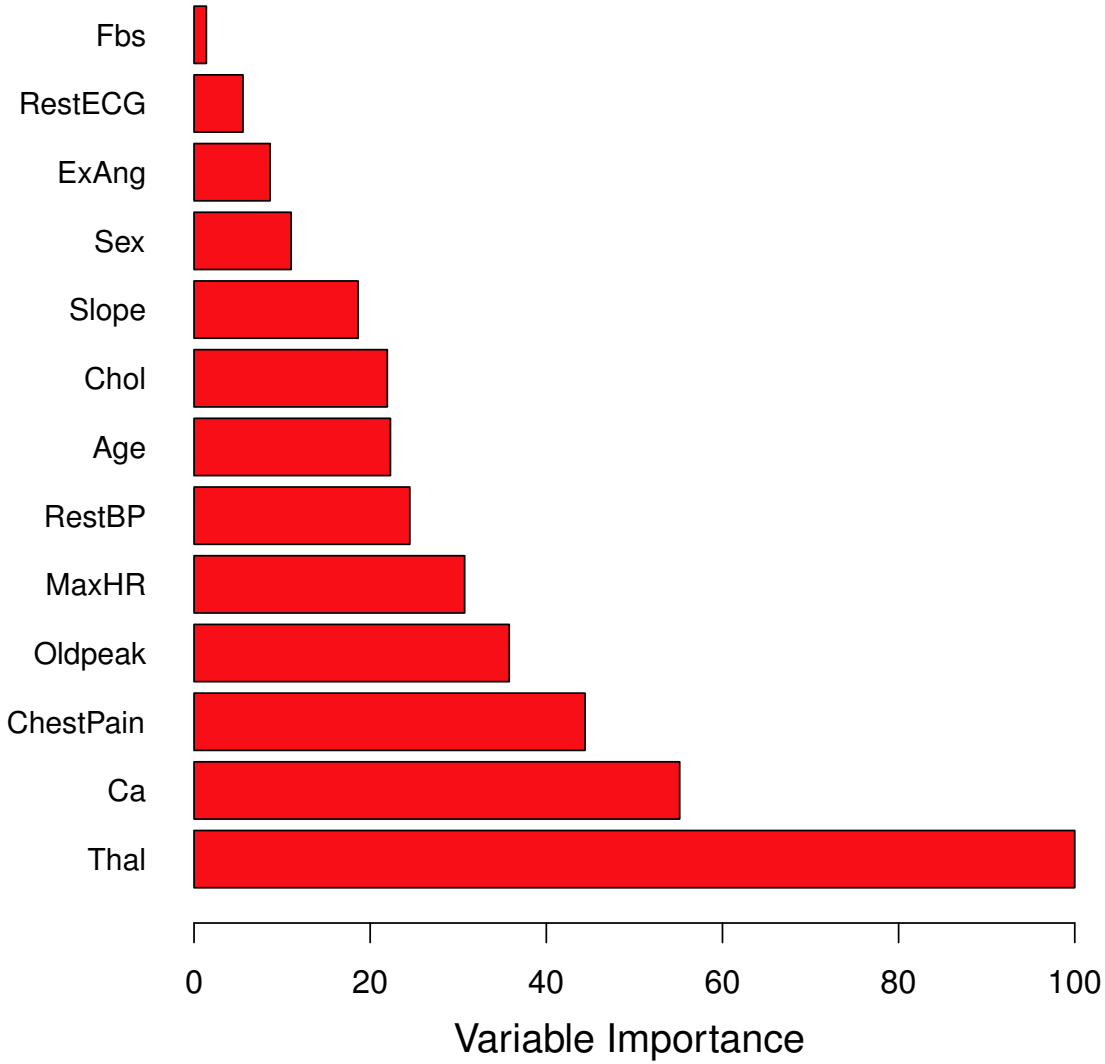
Medidas de la Importancia de las Variables I

- Bagging típicamente resulta en una mejora de la precisión sobre la predicción usando un sólo árbol.
- Desafortunadamente, puede ser difícil de interpretar el modelo resultante.
- Recordar que una de las ventajas de los árboles de decisión era la facilidad para interpertar el diagrama resultante.
- Cuando se agregan un gran número de árboles, no es posible representar el procedimiento estadístico de aprendizaje usando un sólo árbol, y no queda tan claro cuales son la variables más importantes para el procedimiento.
- En otras palabras, bagging mejora la precisión en la predicción a expensas de la interpretabilidad.

Medidas de la Importancia de las Variables II

- A pesar de que la colección de árboles agregados es más difícil de interpretar que un sólo árbol, es posible obtener un resumen de la importancia de cada predictor usando el índice de Gini.
- El resumen puede obtenerse añadiendo la cantidad total en que el índice de Gini es reducido por las divisiones sobre el predictor, promediado sobre todos los B árboles.

Gráfica de Importancia de las Variables



Random Forest I

- Random forests proporcionan una mejora sobre los árboles agregados por bagging realizando un pequeño ajuste que descorrelaciona a los árboles.
- De manera similar al bagging, se construye un número de árboles de decisión realizando un bootstrapping de las muestras de entrenamiento.
- Cuando se construye el árbol de decisión, cada vez que se considera una división, una muestra aleatoria de m predictores es seleccionada como candidatos para la división dentro del conjunto total de p predictores.
- La división es permitida de usar solamente uno de los m predictores.
- Una nueva muestra de m predictores es tomada en cada división, y típicamente se elige $m \approx \sqrt{p}$.
- En otras palabras, en cada división del árbol, el algoritmo no permite que se consideren a la mayoría de los predictores.

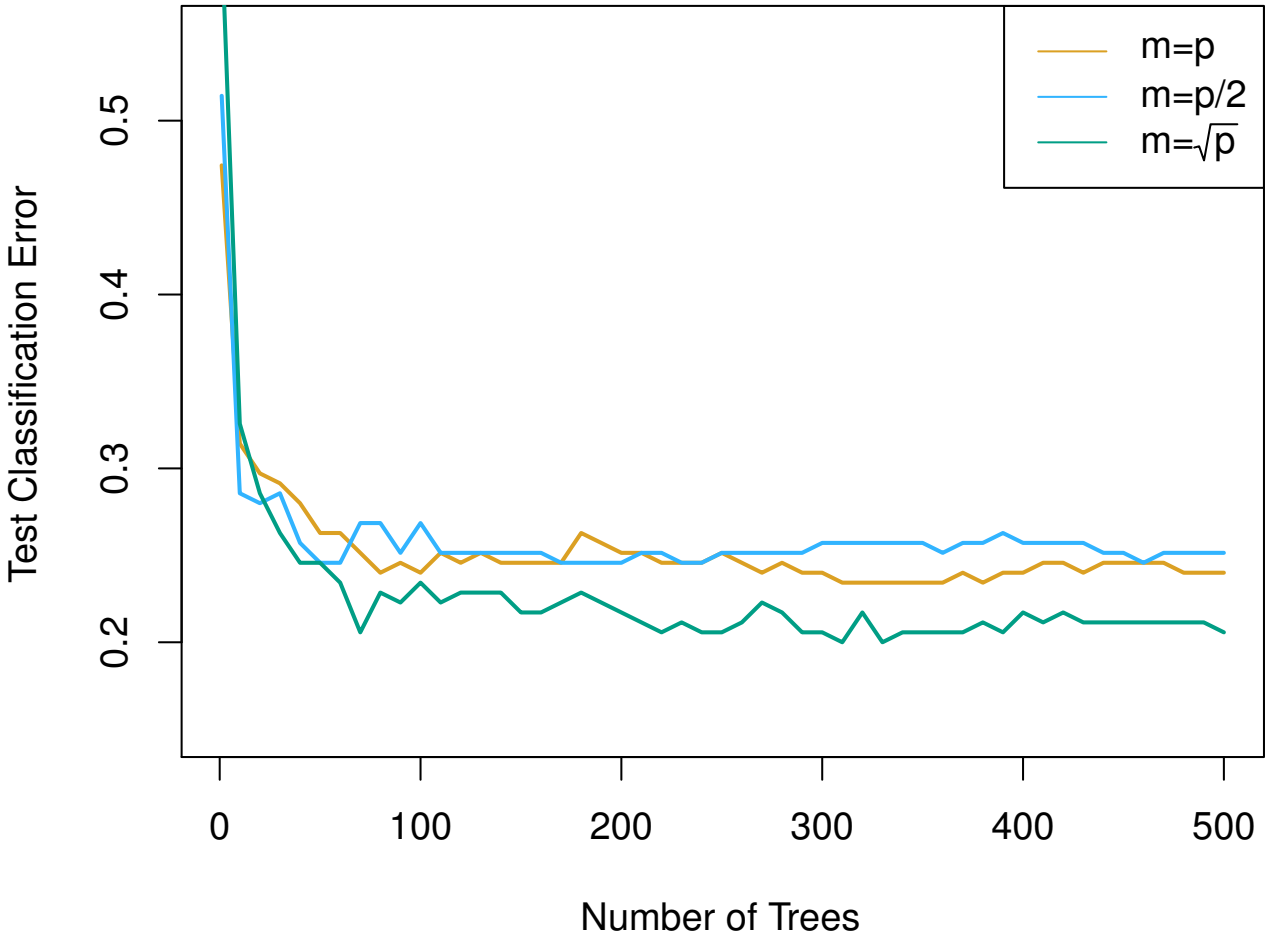
Random Forest II

- Si existiera un predictor demasiado fuerte en el conjunto de datos, mientras el resto de los predictores tiene una fuerza moderada, entonces la mayoría de árboles agregados por bagging usarán este predictor al inicio de la división.
- Como consecuencia la mayoría de los árboles lucirán similares unos a los otros y las predicciones estarán altamente correlacionadas.
- Desafortunadamente, promediar muchas cantidades altamente correlacionadas no ayudan a conseguir una gran reducción de la varianza.
- en otras palabras, bagging no logrará una substancial reducción de la varianza en comparación con un sólo árbol en situaciones de este tipo.
- Random forests supera este problema forzando a que cada partición considere solamente un subconjunto de predictores.

Random Forest III

- En promedio, $(p-m)/p$ de la divisiones no considerarán el predictor más fuerte, de tal forma que otros predictores tienen la oportunidad de ser seleccionados.
- Podemos pensar en este proceso como una descorrelación de los árboles, haciendo que el promedio de los árboles sea menos variable y más fiables.
- La principal diferencia entre bagging y random forests es la elección del subconjunto de predictores m.
- Si random forest es construido con $m = p$ es similar a realizar bagging.
- Usar un valor pequeño de m en la construción de random forest será de gran ayuda cuando se tiene una gran cantidad de predictores correlacionados.

Resultados con Random Forests



Comentarios Generales

La lógica del `boosting` es la siguiente:

- Suponga que tienen ciertos síntomas y en vez de consultar a un doctor, visita muchos.
- Suponga que asigna pesos al valor del diagnóstico de cada uno, en base a la precisión de diagnósticos previos que han hecho.
- El diagnóstico final es una combinación ponderada de los diagnósticos.

Boosting y AdaBoost

La lógica del `boosting` es la siguiente:

- En `boosting` se asignan pesos a las unidades de entrenamiento.
- Se entrena iterativamente una serie de k clasificadores.
- Una vez que se ha entrenado el clasificador M_i , se actualizan los pesos de manera que el siguiente clasificador, M_{i+1} ponga más atención a las unidades de entrenamiento que fueron mal clasificadas por M_i .
- El clasificador final, $M*$, combina los votos de cada clasificador individual, donde el peso del voto de cada clasificador es una función de su precisión.

AdaBoost (Adaptative Boosting)

- Suponga que queremos aumentar la precisión de un método de aprendizaje.
- Se tiene un conjunto de datos D, de d unidades clasificadas,

$$(\boldsymbol{X}_1, y_1), (\boldsymbol{X}_2, y_2), \ldots, (\boldsymbol{X}_d, y_d)$$

  donde y_i es la clase de la unidad $\boldsymbol{X}_i$.
- Inicialmente, `AdaBoost` asigna un peso de $1/d$ a cada unidad de D.
- En la iteración i, las unidades de D son muestreadas para formar un conjunto de trabajo de tamaño d, D_i.
- La chance de ser seleccionada de cada unidad está determinada por su peso.

AdaBoost (Adaptative Boosting)

- Se deriva un clasificador M_i y se calcula su error con utilizando D_i como datos de validación.
- Se ajustan los pesos de las unidades de entrenamiento de acuerdo a cómo fueron clasificadas:
  - mal clasificada $\Longrightarrow$ aumenta su peso.
  - bien clasificada $\Longrightarrow$ disminuye su peso.

El peso de cada unidad refleja su dificultad de clasificación.
A mayor peso, más veces ha sido clasificada incorrectamente.

- Estos pesos se utilizan para generar las muestras de entrenamiento para el siguiente clasificador.

AdaBoost (Adaptative Boosting)

- La idea es centrarse más en las unidades con clasificación incorrecta en el paso anterior.
- Algunos clasificadores son mejores para clasificar unidades “difíciles” por lo que se construye una serie de clasificadores que se complementa.

AdaBoost: Algoritmo

Input:

- D: datos con d unidades clasificadas.
- k: número de iteraciones (se genera un clasificador por iteración)

Output: modelo compuesto.

Paso 1: Inicializar el peso de cada unidad en $1/d$.

Para cada iteración:

Paso 2: Muestrear D con reemplazo de acuerdo a los pesos para obtener D_i.

Paso 3: Usar D_i para derivar M_i.

AdaBoost: Algoritmo

Paso 4: Calcular:

$$error(M_i) = \sum_{j=1}^{d} w_j \times err(\boldsymbol{X}_j)$$

donde $err(\boldsymbol{X}_j)$ indica si la unidad $\boldsymbol{X}_j$ fue mal clasificada.

Paso 5: Si $error(M_i) > 0.5$, volver al paso 2.

Paso 6 Para cada unidad en D_i bien clasificada, multiplicar su peso por $\frac{error(M_i)}{(1-error(M_i)}$.

Paso 7 Actualizar los pesos y normalizarlos.

AdaBoost

- `Boosting` asigna un peso a cada voto de los clasificadores, de acuerdo a su rendimiento.
- A menor tasa de error, más exacto, por lo tanto más vale su voto.
- El peso del voto del clasificador M_i es

$$log\left(\frac{1 - error(M_i)}{error(M_i)}\right)$$

- Para cada clase, c, se suman los pesos de cada clasificador que asignó la clase c a $\boldsymbol{X}$.
- "Gana" la clase con la suma mayor y corresponde a la predicción para $\boldsymbol{X}$.

Bagging vs Boosting

- Como `boosting` se focaliza en las unidades mal clasificadas, corre el riesgo de sobreajustar el modelo resultante a esos datos.
- Algunas veces el modelo resultante puede ser menos preciso que un modelo único para los mismos datos.
- `Bagging` es menos susceptible a un sobreajuste del modelo.
- Aunque ambos pueden mejorar significativamente la exactitud del modelo en comparación a un modelo único, `boosting` tiende a lograr mayor precisión.

Gradient Boosting I

- Gradient Boosting es una generalización del algoritmo AdaBoost que permite emplear cualquier función de coste, siempre que esta sea diferenciable.
- La flexibilidad de este algoritmo ha hecho posible aplicar boosting a multitud de problemas (regresión, clasificación con más de dos clases?).
- Para cada uno de ellos, el algoritmo de Gradient Boosting es ligeramente distinto, pero, para todos, la idea es la misma: dada una función de coste (por ejemplo, residuos cuadrados para regresión) y un weak learner (por ejemplo, árboles), el algoritmo trata de encontrar el modelo que minimiza la función de coste.
- Suele iniciarse con la mejor aproximación de la variable respuesta (la media en el caso de regresión), se calculan los residuos y con ellos se ajusta un nuevo weak learner que intente minimizar la función de coste.

Gradient Boosting II

- Este proceso se repite M veces, de forma que cada nuevo modelo minimiza los residuos (errores) del anterior.
- Dado que el objetivo de Gradient Boosting es ir minimizando los residuos iteración a iteración, es susceptible de overfitting. Una forma de evitar este problema es empleando un valor de regularización, también conocido como learning rate (λ), que limite la influencia de cada modelo en el conjunto del ensemble. Como consecuencia de esta regularización, se necesitan más modelos para formar el ensemble pero se consiguen mejores resultados.
- `https://www.youtube.com/watch?v=wPqtzj5VZus`

Stochastic Gradient Boosting I

- Tiempo después de la publicación de algoritmo de Gradient Boosting, se le incorporó una de las propiedades de bagging, el muestreo aleatorio de observaciones de entrenamiento.
- En concreto, en cada iteración del algoritmo, el weak learner se ajusta empleando únicamente una fracción del set de entrenamiento, extraída de forma aleatoria y sin reemplazo (no con bootstrapping).
- Al resultado de esta modificación se le conoce como Stochastic Gradient Boosting y aporta dos ventajas: consigue mejorar la capacidad predictiva y permite estimar el out of bag error, lo que facilita mucho la optimización de hiperparámetros.

Gradient Boosting Monótono I

- Los modelos basados en Gradient Boosting son capaces de aprender interacciones complejas entre los predictores y la variable respuesta.
- Esto puede convertirse en una desventaja cuando, debido al ruido de los datos, el modelo aprende comportamientos que no reflejan la realidad.
- Una estrategia para evitar este tipo de problemas consiste en forzar a que la relación de cada predictor (numérico) con la variable respuesta sea monótona, es decir tenga siempre la misma dirección (positiva o negativa).
- Aunque el algoritmo es más complejo, a grandes rasgos, se consigue una relación monótona respecto a una determinada variable si, además de otros criterios, solo se permiten divisiones de nodos cuando el valor medio del nodo hijo derecho es mayor que el nodo hijo izquierdo (o a la inversa si se quiere una relación negativa).