



# Aprendizaje en Ensamble

ITAM



## ➤ Inspiración

➤ Sabiduría de las masas

## ➤ Bagging

➤ Una técnica exitosa: Random Forest

## ➤ Boosting

➤ Una técnica exitosa: Adaboost

# Inspiración

- Experimento de Galton
- Wisdom of Crowds (Surowiecki)
- Who wants to be a millionaire

# Ensamblas

- Estos métodos construyen múltiples modelos
  - Diversidad de modelos, diversos aspectos de los datos
- Combinan la predicción de los diversos modelos
  - Votaciones, promedios, etc.
- La diferencia entre las diversas técnicas se reduce a diferencias en estos dos puntos

# Bagging

- Bagging (bootstrap aggregating)
- Este método toma diferentes muestras aleatorias de los datos con reemplazo (bootstrap)
- Aquí asumimos que las diferencias entre las muestras llevarán a diferencias en los modelos resultantes
- La predicción final se toma usando alguna regla para combinar las predicciones individuales
  - Votación (mayoría, pluralidad, ponderada, suave..)
  - Promedio (simple y ponderada)

# Bagging

- Como la diversidad es fundamental, otra idea para fomentarla es utilizar diferentes subconjuntos de variables para cada modelo (subspace sampling)
  - Fomenta diversidad
  - Aumenta velocidad
  - Ayuda un poco con la maldición de la dimensionalidad
- Los árboles de decisión son sensibles a variaciones en los datos y pueden aprovechar bien esta técnica
  - Pequeñas variaciones en los datos pueden provocar construcciones distintas
- Esto es el fundamento de los árboles aleatorios

# Random Forest Entrenamiento

- Input: Datos  $D$ , número de árboles  $T$ , número de atributos  $p$
- Output: Un conjunto de árboles
- for  $t=1$  to  $T$  do
  - Crear muestra  $D_t$  a partir de  $D$  con reemplazo
  - Seleccionar  $p$  atributos al azar y eliminar el resto de  $D_t$
  - Crear un árbol  $A_t$  usando  $D_t$  sin podar
- return  $\{A_t \mid 1 \leq t \leq T\}$
- Nota: Se recomienda que  $p = \log(\text{número de atributos en } D)$  o bien  $p = \sqrt{\text{número de atributos en } D}$

# Random Forest Predicción

- Input: Conjunto de Árboles, instancia a etiquetar  $x$ , bandera de si es clasificación o regresión
- Output: Predicción para  $x$
- for  $t=1$  to  $T$  do
  - $y_t = A_t.\text{predict}(X)$
- if classification
  - return  $\text{vote}(\{y_t \mid 1 \leq t \leq T\})$
- else
  - return  $\text{mean}(\{y_t \mid 1 \leq t \leq T\})$



# Boosting

- Surge de la pregunta de si son equivalentes las clases de problemas weakly learnable y strongly learnable
  - La respuesta la dio Schapire en su artículo “The Strength of Weak Learnability”
  - La respuesta es que si son equivalentes y la demostración es por construcción. La construcción es boosting (bootstrap aggregation)
- La idea general:
  - Generar secuencialmente un conjunto de modelos. Cada nuevo modelo es entrenado para corregir los errores de los modelos anteriores. La salida es una combinación de los modelos

# AdaBoost

- Adaboost (adaptive boosting) es una implementación de esta idea. En particular define:
  - Cómo dar peso a los ejemplos de entrenamiento para reflejar los aciertos y errores de los otros modelos
  - Cómo dar peso a cada modelo que refleje lo importante que es para el ensamble final
- Define una función de costo exponencial de la cuál se derivan los valores anteriores
- Toma como entrada cualquier algoritmo de clasificación binaria y supone que las clases están etiquetadas con -1 y 1

# Boosting

- Para cada iteración se calculan los nuevos pesos que debe tener cada ejemplo
  - Esto se puede implementar muestreando los datos de entrenamiento con la nueva distribución
- La idea es adjudicar la mitad del peso a los datos bien clasificados y la mitad a los datos mal clasificados.
  - Dados  $D$  datos al principio cada instancia tiene peso  $1/|D|$
  - Subsecuentemente dado el error de clasificación  $\epsilon$  asignamos la mitad del peso a los ejemplos mal clasificado y la otra mitad a los bien clasificados
    - Por ejemplo si nos equivocamos en 25% de los ejemplos, para el siguiente modelo éstos tendrán el doble de peso, mientras que los bien clasificados se reducirán a 0.66

# Otros métodos

- La ideas principales del aprendizaje en ensamble son
  - Contar con modelos que capturen diferentes aspectos de los datos
  - Contar con una manera de mezclarlos
- Dicho esto podemos pensar en hacer un ensamble de modelos heterogéneos (redes neuronales + regresiones + k-vecinos, etc.) con la idea de que provean diversidad
- Podemos mezclarlos utilizando otro modelo más, por ejemplo una regresión logística
  - Esto se conoce como stacking
  - Ahora el modelo mezclador tiene parámetros que aprender (a diferencia de adaboost) por lo que se necesitan tomar en cuenta esto en el proceso de aprendizaje

# Ejercicio Opcional

- Baje un conjunto de datos de UCI
  - <http://archive.ics.uci.edu/ml/>
    - Sugiero Abalone
  - <http://archive.ics.uci.edu/ml/datasets/Abalone?pagewanted=all>
- Utilice los paquetes de Sklearn y compare el desempeño de un árbol de decisión y un random, forest (comparar con adaboost es opcional)