TRABAJO 3: Preguntas de Teoría

Sergio Aguilera Ramirez

25 de mayo de 2019

1. **¿Podría considerarse Bagging como una técnica para estimar el error de predicción de un modelo de aprendizaje?. Diga si o no con argumentos. En caso afirmativo compárela con validación cruzada**.

   Sol:
   Si, ya que bagging se basa en la obtención de diferentes hipótesis a partir de varios conjuntos de entrenamiento usando bootstraping. Para obtener estas hipótesis entrenamos los conjuntos de datos con el modelo de aprendizaje seleccionado. La predicción final es el resultado de realizar el promedio de las diferentes predicciones obtenidas en la validación, permitiendo obtener una varianza reducida del conjunto de datos.

   Estas dos técnicas permiten la estimación del error de un conjunto de datos mediante un algoritmo de aprendizaje, bagging utiliza el método out-of-bag error para calcular el error del conjunto y validación cruzada obtiene el error a partir de la media de los errores obtenidos en las diferentes particiones. En comparación con bagging, la validación cruzada divide el conjunto de datos de forma fija, es decir, los datos que componen los diferentes conjuntos no podrán estar repetidos en ninguno de los conjuntos restantes, en cambio bagging selecciona los conjuntos mediante la técnica bootstrapping la cual remuestrea de forma aleatoria y con reemplazamiento el conjunto de datos pudiendo aparecer datos repetidos en los distintos conjuntos. Bagging obtiene el resultado final por medio del promedio de los distintos resultados obtenidos por la validación de los conjuntos de datos, a diferencia de la validación cruzada que realiza el ajuste final del mejor resultado obtenido de los conjuntos de datos.

2. **Considere que dispone de un conjunto de datos linealmente separable. Recuerde que una vez establecido un orden sobre los datos, el algoritmo perceptron encuentra un hiperplano separador iterando sobre los datos y adaptando los pesos de acuerdo al algoritmo**

   1: Entradas: $(\mathbf{x}_i, y_i), i = 1, \ldots, n, \mathbf{w} = 0, k = 0$
   2: repeat
   3 : $k \leftarrow (k+1)$ mód n
   4: if sign $(y_i) \neq$ sign $\left(\mathbf{w}^T \mathbf{x}_i\right)$ then
   5: $\mathbf{w} \leftarrow \mathbf{w} + y_i \mathbf{x}_i$
   6 : end if
   7: until todos los puntos bien clasificados

   Modificar este pseudo-código para adaptarlo a un algoritmo simple de SVM, considerando que en cada iteración adaptamos los pesos de acuerdo al caso peor clasificado de toda la muestra. Justificar adecuadamente/matemáticamente el resultado, mostrando que al final del entrenamiento solo estaremos adaptando los vectores soporte.

Sol:
Como indica el enunciado en este caso vamos a adaptar el vector de pesos respecto al dato peor clasificado.
Los cambios realizados sobre el algoritmo son la eliminación de la condicional de adaptación de pesos y la condición de parada. Debemos modificar el vector de pesos respecto al peor de los puntos clasificados ya que si adaptásemos los pesos solo a los que están mal clasificados estaríamos redireccionando el vector de pesos al punto óptimo, lo cuál no seria cierto para llevar a cabo el método SVM donde este busca un hiperplano donde la distancia al punto mas cercano de ambas clases sea máxima (la distancia a ambos vectores soporte debe ser igual). Modificamos la condición de parada, ya que si parasemos cuando todas los datos estén bien clasificados, el hiperplano podría no quedar situado en la posición óptima, por lo cual afirmamos que a un mayor número de iteraciones obtendremos un hiperplano cuyos margenes sean óptimos. Establecemos un número de iteraciones a realizar por el algoritmo. Una vez que todas los datos se clasifiquen de forma correcta, en las iteraciones restantes el algoritmo adaptará los vectores soporte, esto permite aumentar los margenes respectivos a ambas clases.

3. **Considerar un modelo SVM y los siguientes datos de entrenamiento: Clase-1:(1,1),(2,2),(2,0),Clase-2:(0,0),(1,0),(0,1)**

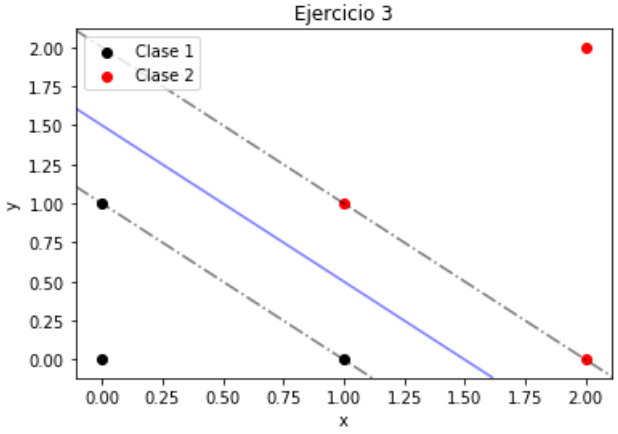
   a) **Dibujar los puntos y construir por inspección el vector de pesos para el hiperplano óptimo y el margen óptimo**

   b) **Cuáles son los vectores soporte**

   c) **Construir la solución en el espacio dual. Comparar la solución con la del apartado (a)**

Sol:
a) Podemos deducir que la recta que une los puntos (1,1) y (2,0) de la clase-1 siendo este recta y = 2 - x tiene la misma pendiente que la recta que une los puntos (1,0) y (0,1) de la clase-2 siendo esta y = 1 - x, es decir estos vectores son equivalentes. Estos puntos son los límites de cada clase respectivamente, con esto afirmamos que el hiperplano estará situado en el espacio comprendido entre estos dos vectores. Estos vectores son denominados vectores soporte del hiperplano. El hiperplano a encontrar debe dejar una distancia máxima entre estos vectores soporte, a esta distancia se le llama margen, siendo la misma distancia en ambos casos.

Podemos calcular el vector intermedio de estos dos vectores de la forma $h = \left(\frac{2-x}{2}\right) + \left(\frac{1-x}{2}\right) = \frac{(2-x)+(1-x)}{2} = \frac{3-2*x}{2}$.




```python
# -*- coding: utf-8 -*-
"""
Created on Tue May 21 23:50:08 2019

@author: SERGIO
"""

import matplotlib.pyplot as plt
from sklearn.svm import SVC
import numpy as np

np.random.seed(1)


datos = [(1,1),(2,2),(2,0),(0,0),(1,0),(0,1)]
etiq = [0,0,0,1,1,1]

datos = np.array(datos, np.int64)
etiq = np.array(etiq, np.int64)

svm = SVC(gamma='auto',kernel='linear',C=100, degree=2,tol=1e-10)
svm.fit(datos,etiq)


def plot_decision_boundary(svm):
    ax = plt.gca()
    xlim = ax.get_xlim()
    ylim = ax.get_ylim()

    x = np.linspace(xlim[0],xlim[1],30)
    y = np.linspace(ylim[0],ylim[1],30)
    Y, X = np.meshgrid(y,x)
    xy = np.vstack([X.ravel(), Y.ravel()]).T
    P = svm.decision_function(xy).reshape(X.shape)

    ax.contour(X,Y,P, colors=['k','b','k'], levels=[-1,0,1],alpha=0.5,linestyles=['-.','-','-.'])

    ax.set_xlim(xlim)
    ax.set_ylim(ylim)

clase1 = [(0,0),(1,0),(0,1)]
clase2 = [(1,1),(2,2),(2,0)]

clase1 = np.array(clase1,np.int64)
clase2 = np.array(clase2,np.int64)

plt.scatter(clase1[:,0],clase1[:,1],c='k', label='Clase 1')
plt.scatter(clase2[:,0],clase2[:,1],c='r', label='Clase 2')

plot_decision_boundary(svm)
plt.title('Ejercicio 3')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```

b) Los vectores soporte son los puntos que comprenden las líneas de los márgenes del hiperplano, estos son [(1,0),(0,1),(1,1),(2,0)]

c)

4. **¿Cuál es el criterio de optimalidad en la construcción de un árbol? Analice un clasificador en árbol en términos de sesgo y varianza. ¿Que estrategia de mejora propondría?**

   Sol:

   El principal criterio de optimalidad de un árbol es la búsqueda de modelos simples, es decir, árbol este compuesto por modelos los cuales no tengan una complejidad alta, de forma que consigamos optimizar de manera compensada entre la complejidad simple del modelo y la precisión del modelo sobre los datos de entrenamiento. Los árboles de decisión se caracterizan por tener un bajo sesgo, ya que sugieren pequeños cambios en la estimación de la función objetivo con cambios en el conjunto de datos, por otro lado los árboles de decisión tienen una alta varianza por lo que estos necesitan grandes cambios en la estimación de la función objetivo. Como sabemos el objetivo de cualquier algoritmo de Machine Learning es lograr un sesgo alto y una baja varianza por lo que propondría una compensación Bias-Varianza o Trade-Off, esto se basa en conseguir una mayor precisión de predicción frente a tener una gran consistencia de los modelos de entrenamiento.

5. **¿Cómo influye la dimensión del vector de entrada en los modelos: SVM, RF, Boosting and NN?**

   Sol:
   En los modelos SVM y NN la dimensión del vector de entrada para que estos modelos sean óptimos debe ser elevada. SVM ajusta los vectores soporte mediante los datos de entrada por lo que a un mayor número de estos mejor ajuste podrá realizar sobre el hiperplano. Las redes neuronales se caracterizan por soportar y realizar un buen modelo sobre vectores de gran dimensionalidad, aunque esto suponga un aumento del tiempo de entrenamiento.

   Respecto a los modelos RF y boosting trabajan de forma limitada sobre vectores de alta dimensionalidad, esto es debido a que estos modelos podrían llegar a sobreajustar la muestra obteniendo un mal ajuste sobre los datos de la muestra, lo que sería erróneo a la hora de estimar predicciones fuera de la muestra.

6. **El método de Boosting representa una forma alternativa en la búsqueda del mejor clasificador respecto del enfoque tradicional implementado por los algoritmos: PLA, SVM, NN, etc.**

   a) **Identifique de forma clara y concisa las novedades del enfoque**

   b) **Diga las razones profundas por las que la técnica funciona produciendo buenos ajustes (no ponga el algoritmo)**

   c) **Identifique sus principales debilidades**

   d) **¿Cuál es su capacidad de generalización comparado con SVM?**

Sol:
a) Boosting se centra en construir un único clasificador fuerte como adición de múltiples clasificadores débiles, el conjunto de datos original cambia en cada iteración. Boosting construye un clasificador de forma secuencial a partir de todas las muestras generadas del conjunto original, estos conjuntos se eligen de forma aleatoria permitiendo disminuir el sesgo.
b) Boosting es una técnica que consiste en ajustar secuencialmente múltiples clasificadores de manera adaptativa, cada modelo ajusta dotando una mayor importancia a las observaciones del conjunto de datos de la secuencia anterior que el modelo no clasificó de forma correcta. Cada nuevo modelo obtenido enfoca sus esfuerzos en las observaciones mas difíciles de ajustar hasta el momento, por lo que al final del proceso obtendremos un clasificador robusto y sólido, con un sesgo bajo.
c)Unos de las principales debilidades de este método es la aparición de sobreajuste cuando los clasificadores débiles son demasiados complejos, esto ocurre por la varianza que presentan estos clasificadores.
d) La generalización mediante el método boosting aumenta cuando la complejidad de los clasificadores aumenta, en cambio la generalización en SVM aumenta cuando mayor es el número de vectores soporte.

7. **Discuta pros y contras de los clasificadores SVM y Random Forest (RF). Considera que SVM por su construcción a través de un problema de optimización debería ser un mejor clasificador que RF. Justificar las respuestas**

   Sol:
   Beneficios SVM: este se basa en la búsqueda del hiperplano óptimo del conjunto de datos, esto supone que si tenemos datos que son separables linealmente, este obtendrá el hiperplano que mejor represente dicho conjunto. Este clasificador situa el hiperplano en la posición donde la distancia de este a los respectivos vectores soporte sea la mayor posible permitiendo una mayor flexibilidad a la hora de clasificar datos fuera de la muestra, también permite una mejora sobre los datos con ruido, ya que contra mayor sea el margen entre los vectores soporte y el hiperplano este permite una posible mejor clasificación sobre datos con ruido.

   Desventajas SVM: en los clasificadores SVM no es fácil encontrar los hiperparámetros adecuados para un conjunto de datos. Cuando el conjunto de datos de entrenamiento es demasiado grande el modelo SVM requiere largos tiempos de entrenamiento sobre dichos datos.

   Beneficios RF: Uno de los beneficios de RF es que, los clasificadores simples que lo componen no están correlados entre sí por lo que esto permite obtener una mayor diversidad entre los conjuntos de datos, además la obtención del resultado final de RF se realiza a partir del promedio de los diferentes clasificadores simples de este, permitiendo disminuir la varianza.

   Desventajas RF: la precisión de la predicción del modelo Random Forest en los problemas complejos es costosa. Random Forest es difícil de interpretar, ya que cada arból subyacente de Random Forest se visualiza como una secuencia de decisiones. RF no adopta un buen comportamiento sobre datos con ruido lo cual podría conducirnos a un sobreajuste del modelo.

8. **¿Cuál es a su criterio lo que permite a clasificadores como Random Forest basados en un conjunto de clasificadores simples aprender de forma más eficiente? ¿Cuales son las mejoras que introduce frente a los clasificadores simples?¿Es Random Forest óptimo en algún sentido? Justifique con precisión en las contestaciones**

   Sol:
   La principal razón por la que RF aprende de forma eficiente es que este se basa en la idea de bagging. Este construye un número determinado de árboles de decisión sobre muestras de boostrap. Random Forest a diferencia de bagging, decorrela los árboles simples que lo forman permitiendo la obtención de una mayor reducción de la varianza.

   Una de las mejores introducidas en RF es la obtención de la predicción final a partir del calculo promedio de las diferentes predicciones obtenidas en los clasificadores simples que lo componen, de esto modo permite reducir la varianza de su resultado.

9. **En un experimento para determinar la distribución del tamaño de los peces en un lago, se decide echar una red para capturar una muestra representativa. Así se hace y se obtiene una muestra suficientemente grande de la que se pueden obtener conclusiones estadísticas sobre los preces del lago. Se obtiene la distribución de peces por tamaño y se entregan las conclusiones. Discuta si las conclusiones obtenidas servirán para el objetivo que se persigue e identifique si hay algo que lo impida.**

   Sol:
   Esto no es totalmente cierto, ya que si se da el caso de que cuando echamos la red obtenemos peces del mismo tipo y tamaño, aunque la muestra sea los suficientemente grande, la diversidad dentro de esta muestra sería relativamente baja por lo que esto supondría un problema respecto a las conclusiones, ya que los resultados estarían basados en un determinado tipo de pez. Esto supone que la muestra recogida no tiene una distribución representativa sobre los peces del lago, lo que nos llevaría a errores fuera de la muestra.

10. **Identifique que pasos daría y en que orden para conseguir el menor esfuerzo posible un buen modelo de red neuronal a partir de una muestra de datos. Justifique los pasos propuestos, el orden de los mismos y argumente que son adecuados para conseguir un buen óptimo. Considere que tiene suficientes datos tanto para el ajuste como para el test**

    Sol:
    1- Preprocesamos los datos de entrada, reduciendo la dimensionalidad y disminuyendo la complejidad de los datos, esto permitirá un aprendizaje mas acertado y fácil sobre los datos.
    2- Elegimos el modelo de clasificación, debemos elegir un modelo de acuerdo al conjunto de datos que poseemos el cual represente de forma adecuada el conjunto de datos.
    3- Inicializacion del vector de pesos. Este paso es uno de los mas importantes en la construcción de una red neuronal. La forma más eficiente sería iniciar el vector a valores próximos a 0.
    4- Determinar el número de capas que compondran nuestra red neuronal.
    5- Definir una condición de parada óptima.