Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación

Técnicas de inteligencia artificial

Reporte: Práctica III del perceptrón multicapa

Docente: Abraham Sánchez López

Alumno	Matrícula
Taisen Romero Bañuelos	202055209

Práctica III del perceptrón multicapa

La práctica va de observar el rendimiento de una RNA. En palabras simples, el rendimiento fue adecuado sin muchos ajustes. Sería interesante ver si el rendimiento es el mismo con una RNA de dos capas o más, ya que según he visto, una mayor cantidad de capas suele ayudar a detectar relaciones sutiles, aunque siendo honesto, este dataset no es lo suficientemente complejo como para que valga la pena estudiar esa comparación. Por otro lado, seguí la sugerencia de comparar con un método de minería, y como ayer noté un comportamiento inusual de Random Forest con un dataset pequeño decidí probar de nuevo con un dataset pequeño (el de los vinos de este ejercicio).

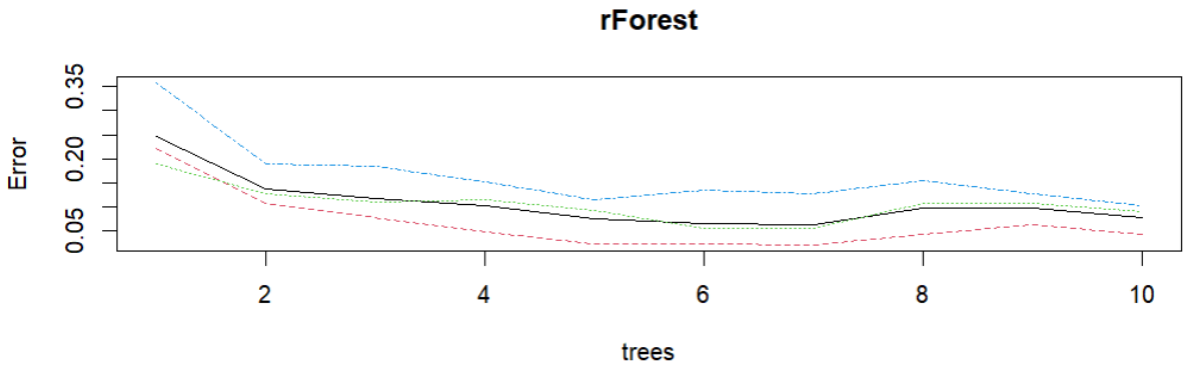
Para aplicar RF no hubo mucha complicación, solo convertí a factor la variable objetivo y apliqué el modelo con una partición 80:20 (ayer usé 90:10, entonces tal vez por eso se comportó de forma inusual). Cuestión, el modelo de RF también dio buenos resultados.

```
> print(rForest)

Call:
 randomForest(formula = label ~ ., data = train, maxnodes = 10,      ntree = 10)
               Type of random forest: classification
                     Number of trees: 10
No. of variables tried at each split: 3

        OOB estimate of  error rate: 7.75%
Confusion matrix:
   1  2  3 class.error
1 45  2  0  0.04255319
2  3 51  2  0.08928571
3  0  4 35  0.10256410
> kappa(rForest$confusion)
[1] 1.472894
```

Sigo sin entender por qué el Kappa es mayor a uno si todavía no es perfecta la precisión, pero si nos basamos en la matriz de confusión podemos notar que tiene un muy buen rendimiento. Como es lógico, cuantos más árboles es menor la tasa de errores.



En esta prueba usé 10 árboles, veamos que pasa con 20.

```
> print(rForest)

Call:
 randomForest(formula = label ~ ., data = train, maxnodes = 10,      ntree = 20)
               Type of random forest: classification
                     Number of trees: 20
No. of variables tried at each split: 3

        OOB estimate of  error rate: 2.8%
Confusion matrix:
   1  2  3 class.error
1 47  0  0  0.00000000
2  1 55  1  0.03508772
3  1  1 37  0.05128205
> kappa(rForest$confusion)
[1] 1.28711
```

Como vemos, el resultado fue considerablemente mejor. Podría ser interesante hacer evaluaciones del rendimiento del modelo, pero no se si es posible aplicar alguna técnica interesante al modelo RF y también a la RNA. Independientemente de eso, podemos reafirmar que hay una competencia fuerte entre los métodos de minería y las redes neuronales.