Óscar Fernández Sánchez
Sergio Orejón Pérez
Jordy Ivan San Martín Ponce
Carlos Garrido Junco

PL3 INTELIGENCIA ARTIFICIAL

1.Describir brevemente en qué consiste el problema (1 punto, 10 líneas de texto máximo)

El problema en está práctica consiste en desarrollar varios modelos de machine learning de clasificación supervisada y compara los resultados en una matriz de confusión para cual de esto es el mejor

El problema consiste en crear una red neuronal artificial con los problemas que ello conlleva(conseguir el número de neuronas en las capas ocultas, el propio número de capas ocultas y conseguir el resultado esperado en la mayor parte de los casos.) Finalmente comparamos todos los modelos con las matrices de confusión y podremos seleccionar el que mejor nos venga para solucionar el problema presentado, en nuestro caso el número de ataques al corazón.

2 Sobre el conjunto de datos original proponer algún tratamiento justificado de los datos, alternativas posibles son el escalado de los datos, la compleción de la base de datos para evitar desbalance entre las clases, la eliminación de observaciones influyentes u otras más complejas como la generación mediante Montecarlo de observaciones subrogadas (2 puntos, 30 líneas de texto)

En este caso con el dataset que utilizamos no tenemos el problema de escalado de datos y ni compleción de datos, pero igualmente para el caso de que faltará datos hemos incorporado un algoritmo, que revisa si falta algún datos y si falta, utiliza knn con knnimputer para rellenar los datos faltantes.
Hemos considerado también generar muestras con montecarlo, para solucionar el caso del balanceo, ya que tenemos más hombres que mujeres, pero al final lo hemos descartado, ya que aunque las variables son ortogonales, es decir son independientes las unas de las otras, la variable target depende de estas, y para generarla tendría que usar nuestro modelo, lo cual produciría un sobre ajuste porque al final terminaría aprendiendo a partir de la predicción y no de los datos reales.

Para el escalado de datos, tendríamos que aplicar de alguna forma una estrategia lógica de asignación de pesos en las variables, ya que en nuestro caso en particular no puede ocurrir que por ejemplo la variable edad tenga más peso que la variable de los valores de un electrocardiograma en parado. Esto nos ayudará a obtener unos resultados más ajustados, y posiblemente más justos para los casos “ambiguos”.

3. Dividir la nueva base de datos generada considerando algún criterio razonable en tres conjuntos: entrenamiento, test y validación y motivar dicha división. En el resto de la práctica, en lo relativo a la selección de modelos óptimos nunca se emplearán los conjuntos de test y validación. (1 punto, 10 líneas de texto máximo)

Hemos utilizado una división 80 10 10 , es decir hemos cogido el 80% para entrenamiento, con lo que tendríamos un entrenamiento consistente. Un 10% para test y otro 10% para validación, con esto queremos conseguir evitar el sobreajuste y a la vez tener datos de entrenamiento suficientes para realizar unas buenas predicciones. Al fin y al cabo tendríamos 241 datos para entrenar nuestros modelos y otros 62 para probarlos y validar todo. Con esta división de datos hemos comprobado que nuestros modelos funcionan mejor que con otras (eg. 60% entrenamiento, 20%test y 20% validación.)

4.Seleccionar, empleando exclusivamente el conjunto de entrenamiento los modelos óptimos para cada uno de los modelos siguientes, los hiper parámetros a considerar aparecen entre paréntesis: k-nn(número de vecinos), Bayes naïve (parámetro de suavizado), árboles de clasificación (profundidad del árbol) y redes alimentadas hacia delante de (número de capas -máximo 3- y número de neuronas por capa). Para dicha selección se aplicará necesariamente algún procedimiento de re-muestreo como validación cruzada (2 puntos, 20 líneas máximo).

Empezando por vecinos próximos, empleando validación cruzada al principio usamos 10 vecinos y cuando nuestro algoritmo pasa selecciona el mejor número de vecinos posible.. Para finalizar utilizamos validación cruzada con “K Fold” para comprobar que el número de vecinos es el correcto comparándolo con el número de vecinos anterior.

En árboles, según nuestras pruebas (hemos ido probando con distintos valores de profundidad eg. 4,5,6,7,8,9) hemos conseguido saber que el número de profundidad ha de ser 3. Ya que no tenemos un alto número de accuracy en entrenamiento (para no tener sobreajuste) y la validación cruzada nos da la razón.

En Naive Bayes, hemos ido probando varios parámetros de suavizado, desde el estándar que es 1e-9 hasta el que hemos elegido 1e-6. Con la validación cruzada podemos ver que el valor que hemos elegido nos aporta resultados por encima de la media..

Finalmente en las redes neuronales, hemos ido probando también el número de neuronas y junto con la validación cruzada K-Fold hemos podido comprobar que el número de neuronas por capa oculta debe ser 25 y con el “solver” = lbfgs que nos completa los otros hiper parámetros, momentum y tasa de aprendizaje conseguimos unos datos estables de accuracy.

5.Comparar, con criterios basado en la matriz de confusión los resultados obtenidos con los distintos modelos sobre el conjunto de test. Se deberá seleccionar el mejor modelo (knn, Bayes, etc.) indicando claramente los motivos por los cuales se hace dicha elección (2 puntos, 20 líneas máximo)

En la matriz de confusión lo que buscamos es que el número de falsos negativos sea el menor posible porque nos interesa que podamos predecir el número de enfermos reales con seguridad y no tener falsos no enfermos .
En los vecinos próximos tiene un número alto de falsos negativos por lo tanto es malo porque mucha gente que se le decía que no estaba enferma, realmente lo está.
Viendo los árboles podemos ver que también tiene un número alto de falsos negativos por lo tanto sigue siendo malo, pero en el resto de predicciones es algo mejor que knn.
Con bayes mejoran mucho nuestras predicciones ya que tiene pocos falsos negativos, por lo que puede ser una buena elección como modelo en este caso.
Finalmente las redes neuronales, son las que menor número de falsos negativos y falsos positivos tienen.
Por lo visto anteriormente, escogeremos la red neuronal porque nos devuelve mejores resultados y también es un modelo mucho más polivalente que los otros ya que puedes modificar el número de neuronas en las capas ocultas, la función de activación y hasta el “solver”, en nuestro caso lbfgs(ya que nuestro dataset es “pequeño” y ADAM no es viable) y conseguir mejores resultados.

6.Empleando el conjunto de validación indicar si la elección realizada anteriormente es consistente explicando, en caso positivo o negativo los resultados obtenidos y las posibles mejoras al procedimiento. (1 punto, 10 líneas máximo)

Hemos escogido las redes neuronales artificiales, ya que comparando con los otros métodos nos ha proporcionado datos más estables a lo largo de las distintas pruebas que hemos ido realizando, ya que vecinos próximos en ningún momento nos ha devuelto una precisión aceptable teniendo en cuenta el contexto de datos de nuestro dataset y bayes y los árboles varían bastante sus resultados dependiendo del conjunto de datos que se obtenga en el primer bloque de código.
Después de aplicarle el conjunto de validación podemos ver que nuestros resultados no varían mucho, es decir que nuestra red neuronal tiene cierta consistencia en los resultados que nos aporta.

7.Emplear un modelo lineal considerando las características del problema dado (no continuidad de la variable objetivo) y empleando algún criterio de selección de parámetros óptimos necesariamente explicado en clase. Comparar los resultados con el modelo óptimo seleccionado considerando las métricas propuestas anteriormente (1 punto, 10 líneas máximo).

Debido a que es un problema de clasificación supervisada, no podemos hacerlo con un modelo de regresión lineal. por lo tanto hemos decido utilizar un perceptrón el cual solo puede clasificar problemas lineales.Los parámetros que coge el perceptrón son las diferentes columnas, del dataset los cuales tienen unos pesos que irán cambiando en cada iteración.Además como hiper parámetro se le pone un máximo de iteraciones por si no consigue converger.
Como se puede observar los resultados obtenidos por el perceptrón son bastante más imprecisos llegando a tener una precisión de un (60%) de aciertos comparando con la red neuronal que tiene un 80% de precisión y muchos errores en la matriz de confusión. Esto se debe a que el problema no es fácil de clasificar linealmente.