Decisión en Clasificación Binaria

Decisión en Clasificación Binaria

¿Qué queremos hacer con el resultado de nuestras predicciones en una futura aplicación real?

El resultado de aplicar un algoritmo predictivo en clasificación binaria sobre nuevos datos suele venir dado en forma de tabla de probabilidades. Es decir, para cada observación a predecir se obtiene la probabilidad estimada por el algoritmo de que esa observación pertenezca a la clase de interés (que consideraremos de aquí en adelante como la clase minoritaria, y denotaremos como y=1):

	observacion	prob
2902	2902	0.002253886
1685	1685	0.191795764
1093	1093	0.675927032
1112	1112	0.607122317
728	728	0.860130218
1636	1636	0.241558307
937	937	0.880905823
840	840	0.804763614
1987	1987	0.062973749
104	104	0.992507514
2521	2521	0.019578365
107	107	0.990536413
2870	2870	0.001515387
1438	1438	0.457457862
2149	2149	0.020783208

Por ejemplo, la probabilidad estimada/predicha de que y=1 en la observación 2902 es 0.0022, mientras que la probabilidad predicha en la observación 1093 es 0.6759.

(Una excepción es el algoritmo SVM que produce valores duros 0 o 1 en lugar de probabilidades, pero habitualmente los paquetes que desarrollan SVM aplican después la llamada corrección de Platt que arroja resultados en forma de probabilidades como los restantes algoritmos).

Conocimiento del contexto en el que trabajamos

Dependiendo del contexto en que estemos trabajando, las decisiones de qué hacer con una observación dependiendo de su probabilidad predicha pueden ser muy variadas. Algunos ejemplos pueden ser los siguientes:

1) Se desea realizar una campaña de emailing o llamadas telefónicas a observaciones con alta probabilidad de y=1. Cada llamada origina un coste y si se obtiene una respuesta positiva por parte del cliente, se obtiene un beneficio. Pero también podemos ir tirando de la lista

© Javier Portela 2020

ordenada de mayor probabilidad a menor e irlos llamando, según el tiempo y recursos disponibles.

2) Estamos creando una app móvil para clasificar imágenes, por ejemplo distinguir un pastor alemán de cualquier otro perro. En este caso la aplicación es automática y debe originar una respuesta Sí/No.

3) En un ingreso hospitalario, se realiza un triaje, en el cual se deriva un paciente a consulta urgente o no urgente, por ejemplo. Aquí la decisión tiene que ser Sí/No.

4) Se estudia si una reclamación de seguros es fraudulenta. Dependiendo de la probabilidad estimada se pueden realizar diferentes acciones de investigación.

5) En un programa de antivirus se marca un mensaje como spam o no, derivándolo a otras carpetas o acciones sucesivas.

6) En medicina, dependiendo de la probabilidad de y=1, se realiza un diagnóstico concreto (clasificación dura) o bien se pasa a realizar otras pruebas.

7) Se predice un resultado en tenis, y las apuestas son uno a uno (si se acierta, se obtiene exactamente la cantidad que se apuesta).

Por lo tanto, dependiendo del contexto se nos abren diferentes posibilidades de actuación. Un escenario ideal sería tener creada a priori una tabla de coste/beneficio asociada a las acciones que tomamos y dependiendo del caso en el que estemos en cada observación, que pueden ser cuatro, según sea VP,FN,FP o FN (verdadero positivo, negativo, falso positivo o negativo).

Esta situación suele ser muy poco habitual; es difícil en la práctica estimar los costes y beneficios, y a menudo no tiene demasiado sentido (por ejemplo es difícil estimar el coste de equivocarse en el triaje hospitalario o en enviar un mensaje a la carpeta spam).

Lo que sí queda claro a través de estos ejemplos es que a menudo es necesario decidirse por un punto de corte en la probabilidad estimada, a partir del cual realizamos unas acciones u otras. Incluso, aunque se sale de este tema, se puede tener en cuenta más de un punto de corte, para realizar más de dos acciones diferentes, como en los ejemplos 4) o 6) .

Un caso particular, que se puede llamar caso básico, es cuando el coste de equivocarse en uno u otro sentido (FP y FN) es el mismo, y también el beneficio al acertar en ambos sentidos (VP o VN). Es el caso 7) pero podría ser el caso de muchas aplicaciones automáticas como el caso 2). En estos casos el punto de corte correcto para la probabilidad estimada es 0.5, y cualquier otro punto de corte generaría pérdidas esperadas (estamos suponiendo siempre que el algoritmo predice razonablemente bien). En estos ejemplos, utilizar accuracy o tasa de fallos con punto de corte 0.5 como criterio, sería razonable.

Pero en la práctica suele existir cierta asimetría en nuestros aciertos y fallos, puede ser más grave o generar más costes un FP que un FN por ejemplo, o viceversa. Igualmente puede ocurrir con los beneficios. Además en ciertas aplicaciones reales, y sin tener en cuenta los costes relativos, si la prevalencia es muy baja y hay mala separación, a menudo fijar el punto de corte en 0.5 puede originar realizar pocas acciones (predecir pocos positivos) o incluso ninguna, lo cual es poco práctico.

En esos casos es necesario determinar qué punto de corte en las probabilidades estimadas origina unas acciones u otras.

La decisión del punto de corte no es nada más que eso: una **decisión** del investigador, y aunque existen muchísimos métodos que pretenden "optimizar" ese punto de corte (como por ejemplo el punto de corte de Youden, que sería el que diera máximo valor a especificidad+sensitividad, o utilizar medidas combinadas de diagnóstico como la F-M o F1), en la mayoría de los casos se toman decisiones intuitivas y prácticas teniendo en cuenta factores diversos como son el tiempo, costes, capacidad predictiva de nuestros algoritmos, y la estructura numérica de nuestro problema, de la cual se hablará en el siguiente apartado.

Otro aspecto práctico relativo a la toma de decisiones es que podemos adoptar la decisión comentada en el ejemplo 1): "ir tirando de lista ordenada de mayor probabilidad a menor, según el tiempo y recursos disponibles".

En principio es buena opción adoptar esta estrategia, pues evita decisiones sobre puntos de corte basados en probabilidades predichas, dado que estos puntos de corte están basados en estimaciones, siempre sujetas a variabilidad muestral. Variantes posibles asociadas a esta estrategia mencionada incluirían "ir tirando de lista hasta obtener un 20% de éxitos" o planteamientos similares. También hay que mencionar que esta estrategia no es perfecta, cuando hay baja sensitividad encontraremos muchos falsos positivos aún en las observaciones con mayor probabilidad predicha.

Esta última estrategia tampoco sirve en muchas situaciones prácticas, por ejemplo cuando se debe predecir no una lista de observaciones, sino simplemente una sola observación, y hay que tomar una decisión directamente sobre ella. Como en los ejemplos 3), 5), 6), etc.

Estructura numérica del problema

Los planteamientos a priori de qué acciones deberíamos tomar dependiendo de nuestras probabilidades predichas, partiendo solamente del contexto, son útiles para ponernos en situación pero son insuficientes. Hay que hacer también un estudio de la estructura numérica de nuestro problema para saber a qué situaciones concretas nos llevaría la aplicación de nuestra estrategia.

Por ejemplo, cuántas observaciones esperamos en la práctica que necesitarían aplicar la acción A, aproximadamente en cuantas fallaríamos como FP o FN, y a qué situaciones nos llevaría esto.

a) Un ejemplo reciente es aplicar test PCR para el COVID masivamente en aeropuertos. Suponiendo una incidencia de la enfermedad de 1/1000, en 50.000 individuos habría 50 positivos. Como el test PCR tiene una especificidad de 98%, de los 49950 negativos detectaría solo el 98%, es decir fallaría en 999, de manera que declararía esos 999 como positivos (Falsos positivos). Igualmente, de los 50 positivos reales, al tener el test un 85% de sensitividad, detectaría 43 y daría 7 falsos negativos. Esto sin tener en cuenta la latencia, para más información visitar https://www.eldiario.es/sociedad/madrid-prueba-cribados-masivos-mal-planteados-son-inutiles-agotan-recursos_1_6213472.html

Por lo tanto el coste/beneficio de este cribado en aeropuertos parece demasiado alto, con mucho esfuerzo para poca detección (43) y obteniendo además las complicaciones derivadas de los 999 falsos positivos. Sin embargo a priori las cifras de especificidad y sensitividad de 98% y 85% del test PCR parecen altas; es un buen test, lo que pasa es que la prevalencia de la enfermedad es baja y esto hay que tenerlo en cuenta y ponerse en situación, en el escenario real, y pensar en los posibles resultados numéricos y prácticos de nuestras estrategias.

b) Otro ejemplo dramático sobre un test COVID con demasiado baja Sensitividad (capacidad de detectar positivos) puede verse en https://www.eldiario.es/andalucia/brote-causo-25-fallecidos-centro-mayores-cuestiona-cribado-antigenos-sido-desastroso_1_6425917.html.

Probablemente en la subpoblación donde se iba a probar ese test de antígenos (la residencia), había bastantes infectados y por lo tanto era necesario un test con más sensitividad como el PCR, para no dejar fuera demasiados falsos negativos, debido a las graves consecuencias prácticas de ello. Otra vez más no se reflexionó anteriormente sobre los resultados posibles teniendo en cuenta el tamaño de la muestra donde se iba a aplicar el test, las características de esa muestra y las características predictivas de este test.

Datos sobre los problemas de sensitividad con este test de antígenos pueden verse aquí:

https://maldita.es/malditaciencia/2020/11/18/test-antigenos-como-funcionan-casos-indicados-efectividad-asintomaticos/

En ambos ejemplos a) y b) el test para detectar COVID hace el papel que haría nuestro algoritmo predictivo.

Supongamos que tenemos una muestra A para modelar nuestro algoritmo, y una muestra externa/futura B, desconocida de momento, donde vamos a aplicar en la realidad nuestro modelo para obtener predicciones. De momento vamos a suponer que la prevalencia (porcentaje) de y=1 es aproximadamente similar en nuestra muestra A y en la muestra B donde obtendríamos nuestras predicciones.

Desde el punto de vista de la estructura numérica de nuestro problema hay que tener en cuenta, los siguientes factores:

Relativos a la muestra A:

1) El número de observaciones con y=1 en la muestra

2) El número de observaciones total en la muestra

3) Aspectos relativos al desempeño del algoritmo predictivo:

3.1 Número de variables input

3.2 Auc obtenido en validación cruzada; Accuracy o tasa de fallos obtenida en validación cruzada

3.3 Número de observaciones clasificadas como y=1 con punto de corte básico 0.5

3.4 Sensitividad, especificidad con punto de corte básico 0.5, FP,FN

4) Porcentaje de y=1 en la muestra

Relativos a la futura muestra B:

5) Si se conoce, número de observaciones a predecir en una muestra B, en la aplicación práctica futura del algoritmo. Puede ser una cifra aproximada. El número aproximado de observaciones y=1 que se espera en esta muestra B también es una cifra importante. Referimos a los ejemplos a) y b) anteriores para ilustrar lo importante que es prever, aunque sea superficialmente, la estructura de la futura muestra B.

En algunos contextos las predicciones no se van a aplicar sobre una muestra B de casos, sino en observaciones aisladas una a una. Entonces no aplica lo dicho en este punto 5).

Para hacer una evaluación preliminar del problema al que nos enfrentamos, en el punto 3) basta con una simple aplicación de la regresión logística.

En los ejemplos que vendrán a continuación, aparte de obtener un resultado numérico básico, nos ayudaremos con el paquete visualpred para obtener una visión aproximada de la separación que se obtiene con nuestras variables input y algoritmos, del desequilibrio de la muestra y como se refleja en la separación, y otras cuestiones. No hay trabajo un completo de selección de variables y tuneado pero sirven igualmente de ejemplo.

Ejemplos artificiales básicos

Los siguientes ejemplos se basan en muestras generadas artificialmente, y sirven para centrar algunos conceptos básicos de situaciones que se pueden reproducir en la realidad. En estos ejemplos artificiales solo se usan dos variables input x1 y x2. Los puntos Yes de interés, clase minoritaria, son los puntos rojos.

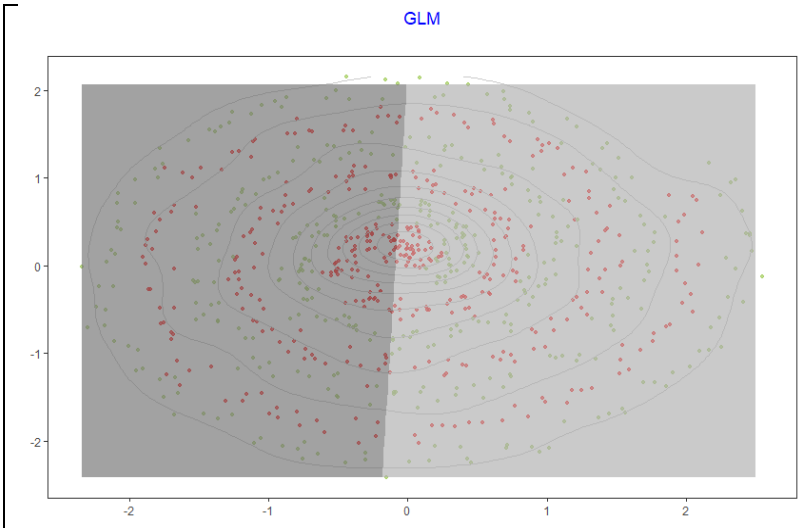
1

obs: 400
obs clase minoritaria: 180
% clase minoritaria: 45 %
tasa fallos: 0
auc: 1
sensitivity: 1
specificity: 1
numero de Yes predichos: 180
FP: 0
FN: 0
VP: 180
VN: 220

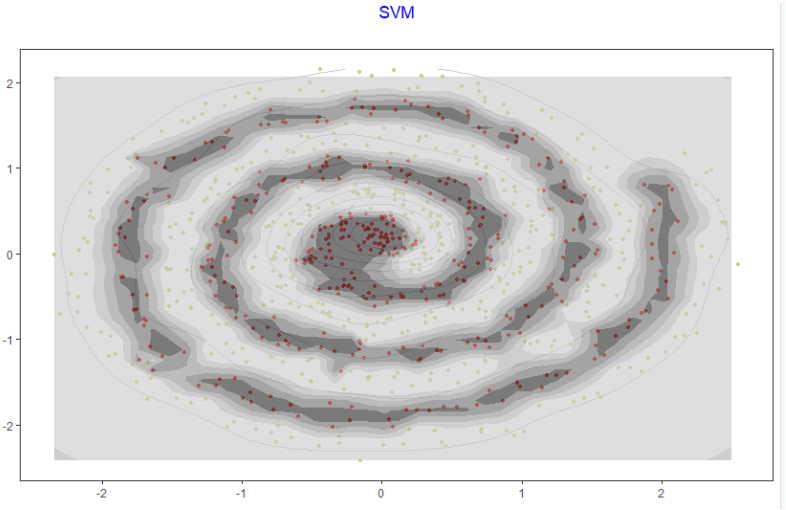
Este es un caso ideal: las variables input permiten una separación lineal perfecta; la regresión logística separa perfectamente las dos clases; aunque las medidas de diagnostico están tomadas en validación cruzada dan valores perfectos.

© Javier Portela 2020

2



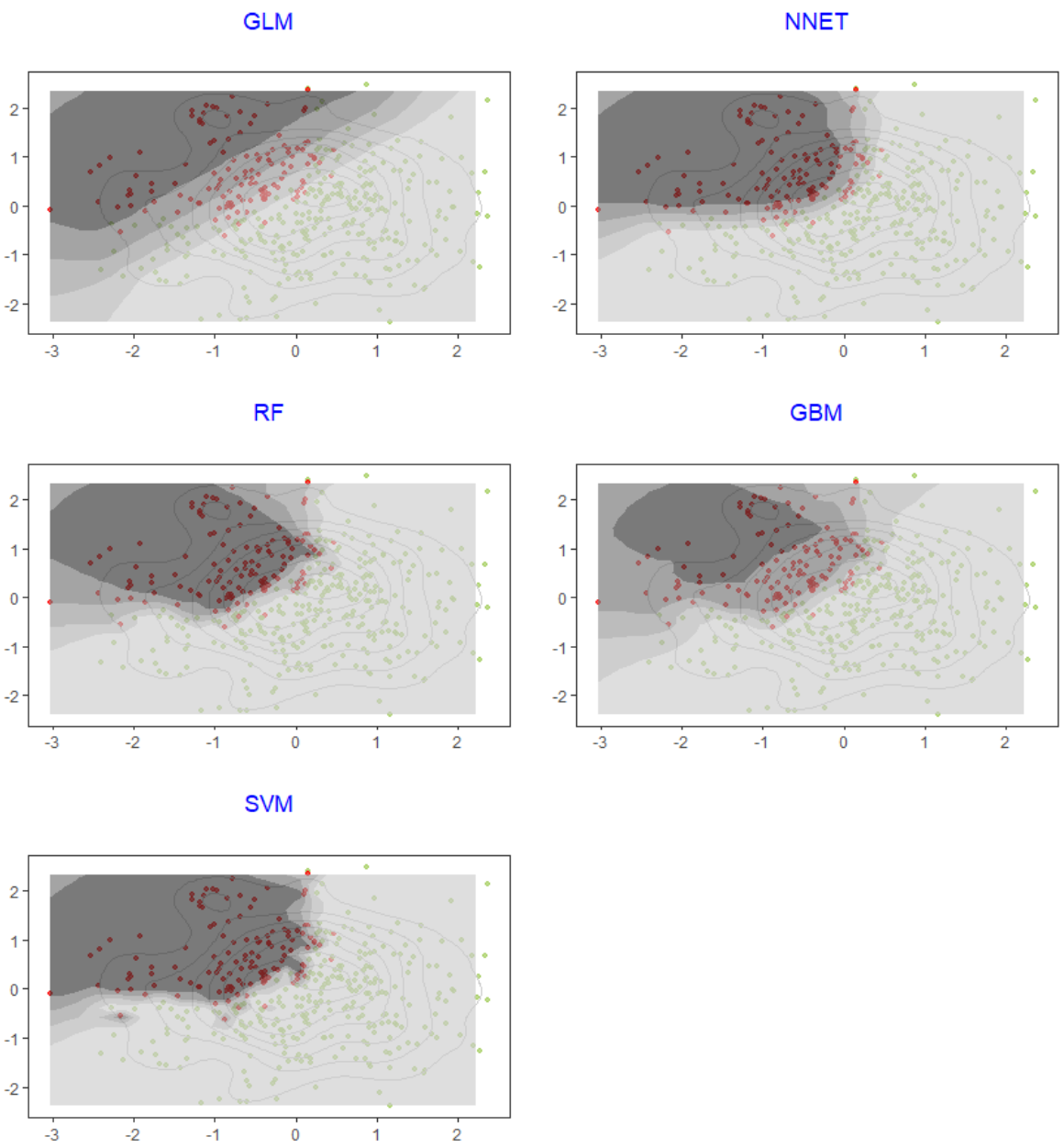
obs: 803
obs clase minoritaria: 401
% clase minoritaria: 50 %
tasa fallos: 0.5554172
auc: 0.5670897
sensitivity: 0.3715711
specificity: 0.5174129
numero de Yes predichos: 343
FP: 194
FN: 252
VP: 149
VN: 208



obs: 803
obs clase minoritaria: 401
% clase minoritaria: 50 %
tasa fallos: 0.06351183
auc: 0.9697708
sensitivity: 0.9376559
specificity: 0.9353234
numero de Yes predichos: 402
FP: 26
FN: 25
VP: 376
VN: 376

En el ejemplo 2 de datos en espiral, se ve como existe separación casi perfecta, pero es no lineal. Aplicar un algoritmo de separación lineal como la logística arroja resultados muy malos en tasa de fallos, auc, etc. mientras que un algoritmo como SVM RBF separa perfectamente las clases. Del mismo modo que elegir el buen algoritmo, un buen tuneado de cada algoritmo puede llegar a mejorar mucho los resultados.

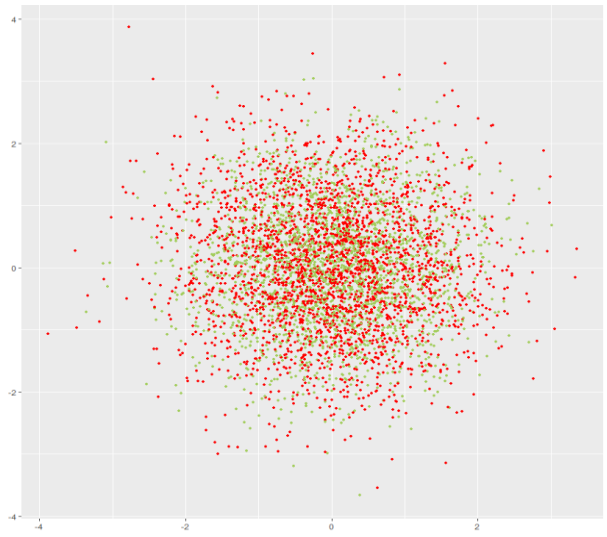
3



El ejemplo 3 muestra datos con separación no lineal más sencilla que los datos en espiral. Exceptuando la logística (glm), que establece incorrectamente separación lineal, los algoritmos de machine learning están preparados para adaptarse a la curva de separación, dando resultados de ajuste parecidos.

© Javier Portela 2020

4

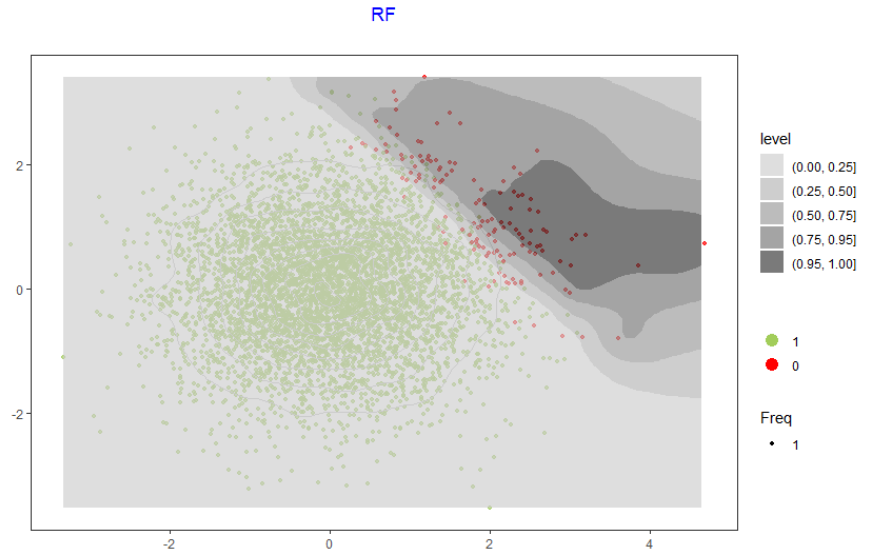


obs: 5000
obs clase minoritaria: 2315
% clase minoritaria: 46 %
tasa fallos: 0.4628
auc: 0.4900898
sensitivity: 0.0004319654
specificity: 1
numero de Yes predichos: 1
FP: 0
FN: 2314
VP: 1
VN: 2685

El ejemplo 4 muestra datos donde las variables input no sirven para separar las clases. La tasa de fallos es similar al porcentaje de la clase minoritaria de la muestra y el auc toma un valor cercano al peor valor posible, que es de 0.5. La sensitividad de 0 significa que la logística directamente clasifica todos los puntos como y=0 y falla en la detección de los y=1, con una especificidad de 1 porque evidentemente detecta todos los ceros.

A recalcar que los valores de y están distribuidos al 50%, ilustrando que un equilibrio entre clases no garantiza nada a priori.

5



obs: 5000
obs clase minoritaria: 138
% clase minoritaria: 3 %
tasa fallos: 0.0136
auc: 0.987026
sensitivity: 0.7028986
specificity: 0.9944467
numero de Yes predichos: 124
FP: 27
FN: 41
VP: 97
VN: 4835

corte 0.4

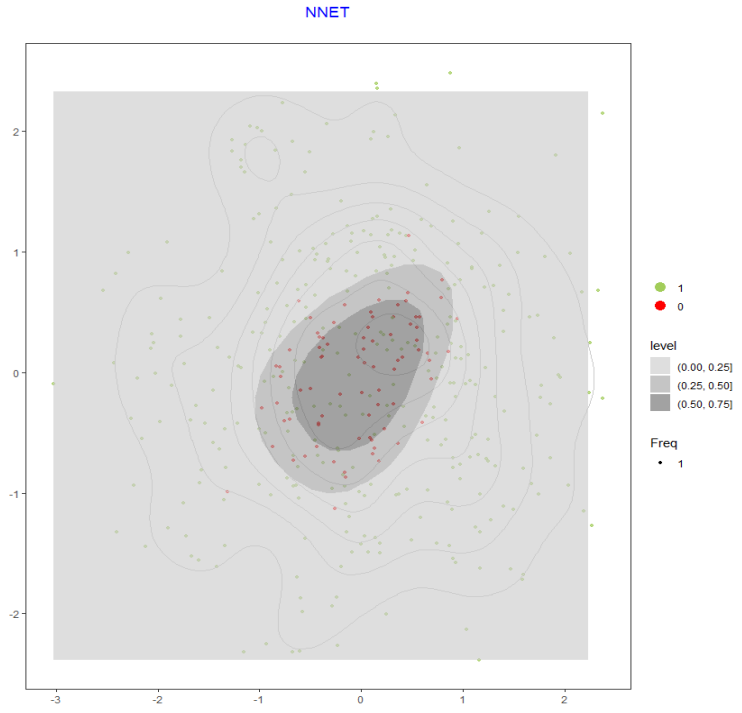
obs: 5000
obs clase minoritaria: 138
% clase minoritaria: 3 %
tasa fallos: 0.0138
auc: 0.987026
sensitivity: 0.7681159
specificity: 0.99239
numero de Yes predichos: 143
FP: 37
FN: 32
VP: 106
VN: 4825

El ejemplo 5 muestra datos donde las variables input permiten separar bien las clases aunque de manera no perfecta, como suele ser normal. Aunque hay desequilibrio de clases (3% para la minoritaria) se consigue una buena separación con un auc de 0.98 y tasa de fallos de 0.01, menor que 3%, con lo que se ve que el desequilibrio de clases no tiene por qué conllevar malas clasificaciones a priori. De las 138 observaciones, se detectan 97 (de los 124 Yes predichos 27 son falsos positivos).

En el gráfico solo las partes más oscuras superan el punto de corte 0.5; el resto de partes ligeramente sombreadas corresponden a probabilidades >0 de Yes, pero menores que 0.5. En este ejemplo se pueden adoptar estrategias de decisión considerando tomar acciones para las observaciones que superen por ejemplo el punto de corte 0.4. Esto mejoraría la detección de positivos pero también llevaría a tomar esas acciones sobre más falsos positivos. Como se ve aplicando el corte 0.4, se detectan 9 positivos más, se falla en 10 falsos positivos más pero se bajan también los falsos negativos en 9.

© Javier Portela 2020

6

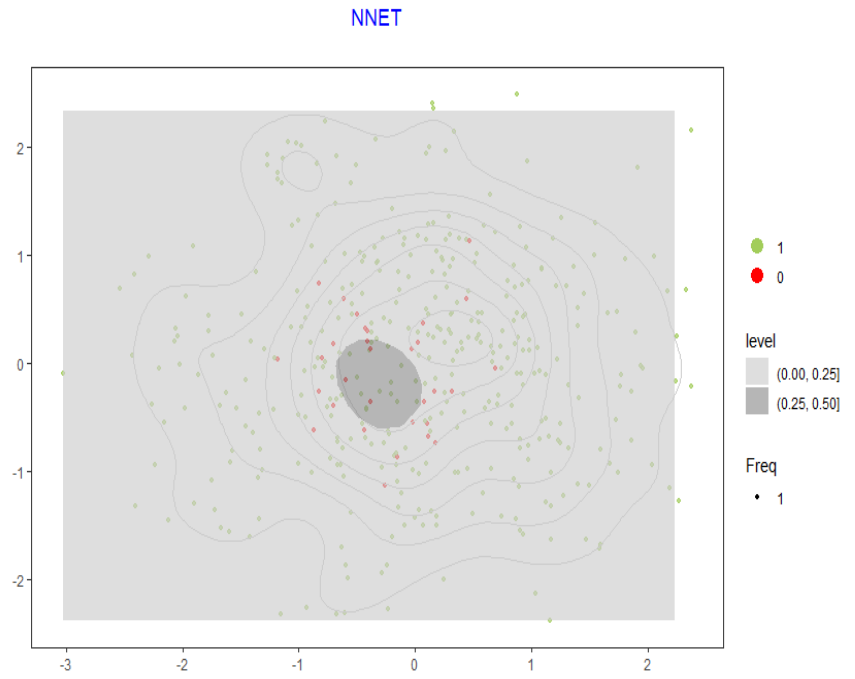


obs: 400
obs clase minoritaria: 80
% clase minoritaria: 20 %
tasa fallos: 0.17
auc: 0.8766797
sensitivity: 0.5625
specificity: 0.896875
numero de Yes predichos: 78
FP: 33
FN: 35
VP: 45
VN: 287

El ejemplo 6 muestra datos de separación no lineal. La red neuronal con 5 nodos permite obtener resultados correctos de separación. En una separación con corte 0.5, todos los puntos del círculo interno serían clasificados como rojos, como aparece en la leyenda relativa a la probabilidad (0.50-0.75). La sensitividad no es demasiado alta pues la zona media que podría permitirnos detectar más puntos rojos tiene probabilidad predicha entre 0.25-0.50 por contener además mayoría de puntos verdes. Queda como una opción de decisión, por ejemplo, realizar acciones para los puntos con probabilidad >0.4, fallando en muchos puntos verdes (falsos positivos, bajando la especificidad) pero detectando más rojos (aumentando la sensitividad).

Hay que destacar también que se predicen 78 Yes, pero de estos, 45 son verdaderos Yes y 33 son falsos positivos, porque en la zona más sombreada hay también muchos puntos verdes.

7



obs: 400
obs clase minoritaria: 31
% clase minoritaria: 8 %
tasa fallos: 0.0775
auc: 0.8009441
sensitivity: 0
specificity: 1
numero de Yes predichos: 0
FP: 0
FN: 31
VP: 0
VN: 369

corte 0.2

obs: 400
obs clase minoritaria: 31
% clase minoritaria: 8 %
tasa fallos: 0.1775
auc: 0.8009441
sensitivity: 0.4516129
specificity: 0.8536585
numero de Yes predichos: 68
FP: 54
FN: 17
VP: 14
VN: 315

El ejemplo 7 muestra datos de separación no lineal . La red neuronal con 5 nodos localiza una zona central de interés en cuanto a separación. Pero en este ejemplo, en comparación con el ejemplo 6 anterior, la zona interior tiene siempre probabilidades inferiores a 0.50. El algoritmo, usando punto de corte 0.5, no predice ninguna observación como Yes de las 31 presentes, resultando en una sensitividad de 0.

Desde el punto de vista práctico, se pueden tomar acciones a partir de un punto de corte menor. Por ejemplo, con corte 0.2 se detectan un 45% de rojos (14) a cambio de tener nada menos que unos 54 falsos positivos…otros puntos de corte se pueden probar dependiendo del interés del investigador. O bien resignarse e intentar conseguir otras variables input más útiles.

Cuando la clase minoritaria es proporcionalmente pequeña suele haber dos consecuencias: un auc relativamente alto, y una especificidad muy alta; si la separación es buena, como en el ejemplo 5) o 6), se pueden tener valores relativamente altos de sensitividad.

Muchas veces esa pequeña proporción de la clase minoritaria viene acompañada de un bajo número de observaciones en esa clase (en este ejemplo 7) hay 31). A menudo un número pequeño de observaciones no permite generar modelos con muchas variables input sin sobreajustar, y como consecuencia es difícil encontrar una buena separación y se llega a casos como este ejemplo 7).

Supongamos por el contrario que hay un número alto de observaciones en la clase minoritaria, aunque tenga una proporción baja de por ejemplo 1%. Si el problema de separación no está

© Javier Portela 2020

bien resuelto, y hay malos valores de sensitividad, la causa de estos bajos valores de sensitividad no es que ese la proporción de la clase minoritaria sea baja (1%), sino que simplemente las variables que tenemos no nos permiten diferenciar más. Solo queda adaptar nuestras decisiones asignando puntos de corte que nos parezcan adecuados, asumiendo un cierto número de falsos positivos. O resignarse y buscar otras variables nuevas que nos puedan ayudar.

Ejemplos con datos reales

Los siguientes ejemplos utilizan datos reales. En la representación se usa el paquete visualpred:

https://cran.r-project.org/package=visualpred

Este paquete utiliza técnicas de reducción de datos (FAMD, MCA) para proyectar sobre dos dimensiones (los dos primeros "factores") los puntos. Ayuda a dar una idea de la capacidad de separación que tenemos con nuestras variables. Algunas cuestiones relativas a la interpretación:

1) La reducción de datos se realiza sin tener en cuenta la variable de clase dependiente. Los factores proyectados son aquellos que representan mejor la información de las variables input. A pesar de ello, si existe una buena separación se verá en el gráfico.

2) Al proyectar una cierta cantidad de variables sobre dos dimensiones , pueden solaparse los puntos, por eso se representan puntos con diferente tamaño relativo a su frecuencia. Esto ocurre a menudo cuando existen muchas variables categóricas en el espacio de variables input.

3) Los sombreados relativos a las regiones de probabilidad predicha se realizan construyendo una rejilla de puntos en el rango de los factores e interpolando la probabilidad estimada en esos puntos ficticios basándose en la predicción sobre los puntos existentes.

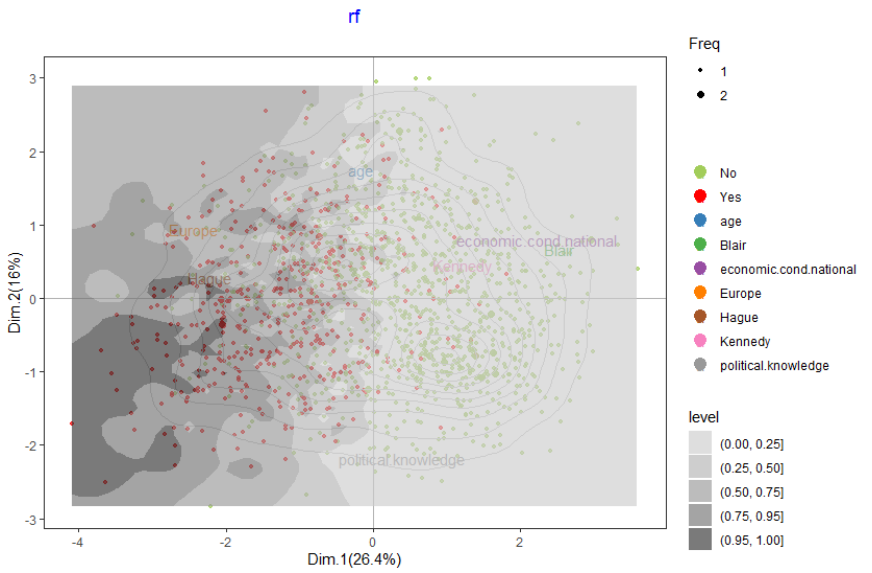
8

BEPS

https://rdrr.io/cran/carData/man/BEPS.html

Datos sobre votantes en elecciones británicas. class=1 indica votante al partido Conservative, class=0 a otros.

Una ejecución rápida con selec=1 y RandomForest permite observar en el gráfico que hay muy buena separación; el modelo stepwise usa 7 variables. Las medidas de diagnóstico también son relativamente buenas aunque la sensitividad no es demasiado alta.



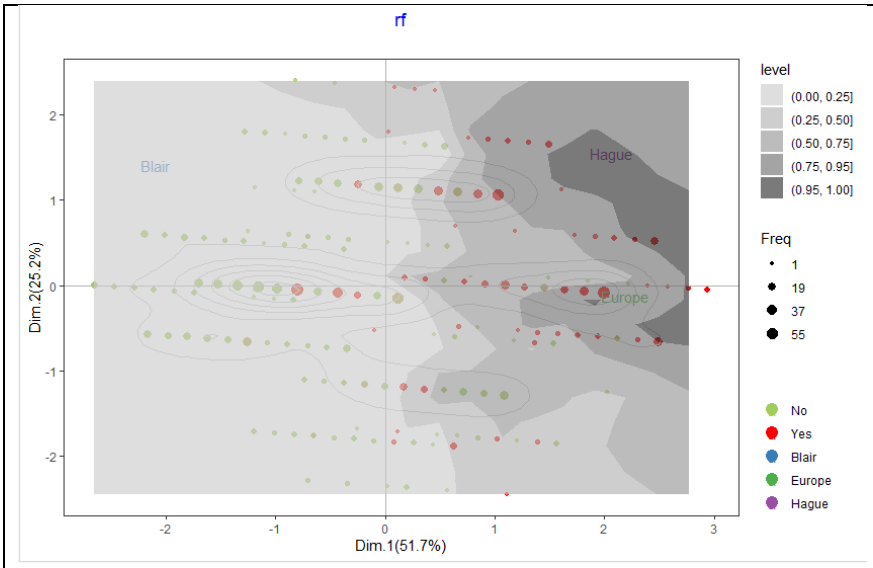
obs: 1525
obs clase minoritaria: 462
% clase minoritaria: 30 %
tasa fallos: 0.1757377
auc: 0.8851938
sensitivity: 0.6450216
specificity: 0.9021637
numero de Yes predichos: 402
FP: 104
FN: 164
VP: 298
VN: 959

corte=0.4

obs: 1525
obs clase minoritaria: 462
% clase minoritaria: 30 %
tasa fallos: 0.1744262
auc: 0.8851938
sensitivity: 0.7467532
specificity: 0.8598307
numero de Yes predichos: 494
FP: 149
FN: 117
VP: 345
VN: 914

Los métodos stepwise de selección de variables no son perfectos (ningún método de selección lo es). Si atendemos al orden por el que se introducen las variables en el algoritmo stepwise observamos que las más evidentes entran primero (Blair, Hague, Europe). Cabe preguntarse si con ellas sería suficiente para tener un buen modelo de separación y si no estamos sobreajustando con nuestras 7 variables. El siguiente gráfico se obtiene con esas tres primeras variables.

© Javier Portela 2020



obs: 1525
obs clase minoritaria: 462
% clase minoritaria: 30 %
tasa fallos: 0.2065574
auc: 0.8431449
sensitivity: 0.6082251
specificity: 0.8739417
numero de Yes predichos: 415
FP: 134
FN: 181
VP: 281
VN: 929

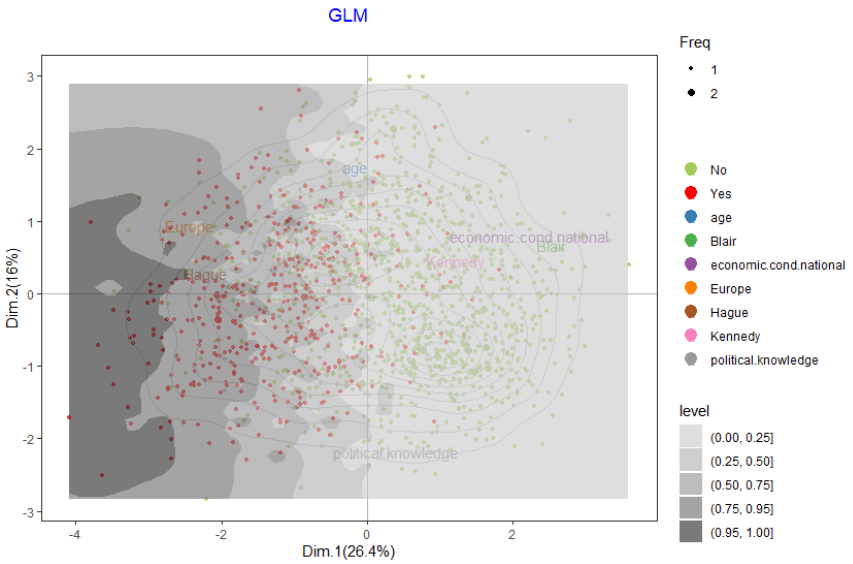
corte 0.4

obs: 1525
obs clase minoritaria: 462
% clase minoritaria: 30 %
tasa fallos: 0.2111475
auc: 0.8431449
sensitivity: 0.6774892
specificity: 0.8372531
numero de Yes predichos: 486
FP: 173
FN: 149
VP: 313
VN: 890

Se ve como al usar solo variables dummies (no continuas como age) la proyección solapa puntos. Las medidas de diagnóstico empeoran en general; pero sirve para ilustrar que se puede obtener algo decente en este ejemplo con solo 3 variables, aunque la capacidad de detección de votantes conservadores Yes (sensitividad) es relativamente baja.

Si quisiéramos aumentar la sensitividad reduciríamos el punto de corte, a costa de tener más falsos positivos. Se ponen como ejemplo las medidas obtenidas al rebajar el punto de corte a 0.4 en ambas versiones con 7 variables y con 3.

Realmente en este ejemplo no era necesario random forest (que parecía sobreajustar con “islas” en el gráfico) , la logística funciona bastante mejor, ante lo que parece una separación más bien lineal :



obs: 1525
obs clase minoritaria: 462
% clase minoritaria: 30 %
tasa fallos: 0.1606557
auc: 0.8882217
sensitivity: 0.6709957
specificity: 0.9125118
numero de Yes predichos: 403
FP: 93
FN: 152
VP: 310
VN: 970

corte 0.4

obs: 1525
obs clase minoritaria: 462
% clase minoritaria: 30 %
tasa fallos: 0.16
auc: 0.8882217
sensitivity: 0.7575758
specificity: 0.8758231
numero de Yes predichos: 482
FP: 132
FN: 112
VP: 350
VN: 931

© Javier Portela 2020

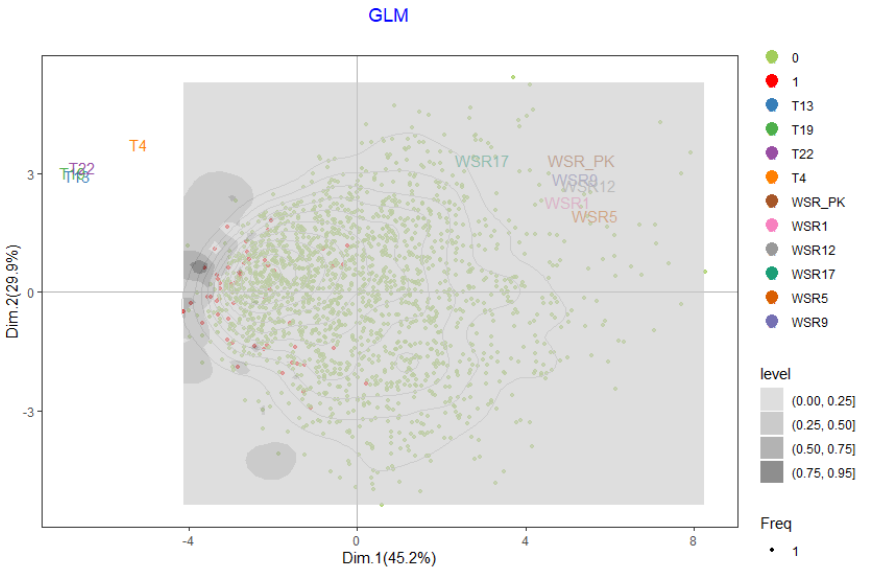
9

OZONO

https://archive.ics.uci.edu/ml/datasets/ozone+level+detection

Datos depurados de predicción de picos de ozono.

Una ejecución básica con GLM y stepwise arroja los resultados representados. A recalcar que la función famdcontour realiza un stepwise backward, mientras que la función seleccionar hace stepwise forward y selecciona menos variables y nos quedamos con este último set.

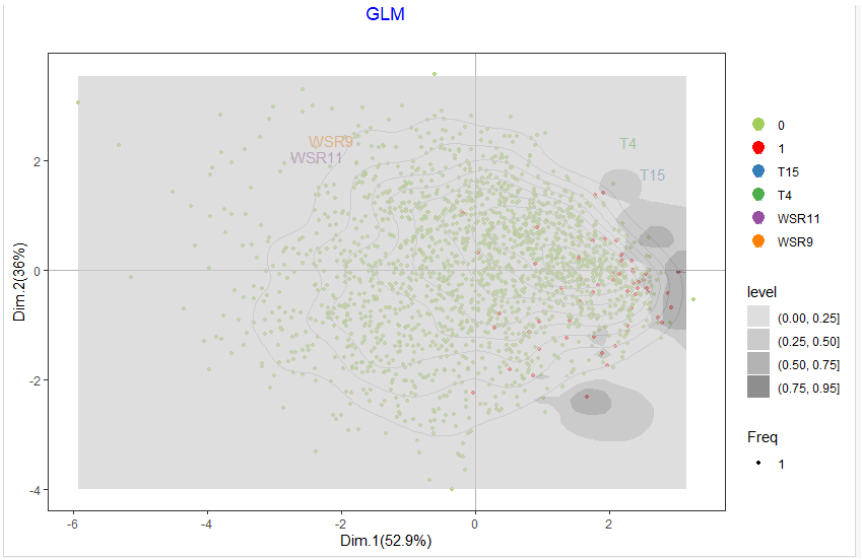


obs: 1848
obs clase minoritaria: 57
% clase minoritaria: 3 %
tasa fallos: 0.02922078
auc: 0.9295307
sensitivity: 0.1929825
specificity: 0.9955332
numero de Yes predichos: 19
FP: 8
FN: 46
VP: 11
VN: 1783

Obviamente se trata de datos problemáticos para un estudio de clasificación. Solo hay 57 observaciones de la clase minoritaria, y el campo de variables input posibles es muy alto, 72 variables. Además parece haber cierta dificultad en encontrar un set de variables consistente, y todo apunta a una selección sobreajustada por parte de los métodos stepwise, seleccionando demasiadas variables (10) cuando solo hay 57 observaciones para representar un perfil de la clase minoritaria. Un ajuste básico da una sensitividad baja, típico de datos con separación insuficiente parecido al ejemplo 6) . Como suele pasar en este tipo de datos, el auc es alto pero eso es simplemente debido a la alta especificidad.

Comenzamos con ilustrar los problemas de selección de variables en este tipo de datos. Una posibilidad para evitar el sobreajuste es quedarse con las 4 primeras seleccionadas por stepwise por ejemplo. Así tendríamos 57 observaciones/4= 14 observaciones en la clase

minoritaria por variable (cada variable origina la estimación de un parámetro en la logística, y al menos 10 observaciones por parámetro son recomendables).



obs: 1848
obs clase minoritaria: 57
% clase minoritaria: 3 %
tasa fallos: 0.02922078
auc: 0.9207735
sensitivity: 0.1052632
specificity: 0.998325
numero de Yes predichos: 9
FP: 3
FN: 51
VP: 6
VN: 1788

corte=0.03

obs: 1848
obs clase minoritaria: 57
% clase minoritaria: 3 %
tasa fallos: 0.1774892
auc: 0.9207735
sensitivity: 0.8421053
specificity: 0.8218872
numero de Yes predichos: 367
FP: 319
FN: 9
VP: 48
VN: 1472

Obviamente la sensitividad baja, pero el aspecto general en el gráfico de la estructura de separación es similar a cuando se utilizaban 10 variables.

Si se trata de capturar positivos, sin importarnos demasiado los falsos positivos, podemos poner un punto de corte muy bajo.

Por ejemplo, si lo fijamos en 0.03, obtenemos una sensitividad de 0.84. Acertamos 48 de los 57, obteniendo una cifra alta de falsos negativos, 319, pero también acertando 1472 negativos de los 1791.

Realmente no podemos más que tomar este tipo de decisiones, pues no se está consiguiendo una separación mejor incluso recurriendo a un modelo sobreajustado.

También hay que tener en cuenta que en casos como este, ante un número tan pequeño de observaciones en la clase minoritaria, cualquier estimación está sujeta a un alto nivel de error. Tanto las variables óptimas a introducir en el modelo, como los parámetros estimados en los modelos, los datos relativos a la sensitividad u otras medidas de diagnóstico varían mucho de muestra a muestra, lo que se conoce en estadística como alta varianza en los estimadores. Manipular el punto de corte afinando demasiado no es razonable, y toda conclusión debe tomarse de manera general e intuitiva.

Por último, y a pesar de los problemas en este ejemplo, hay información valiosa. Está claro que es mejor hacer el modelo, utilizando las variables input, que predecir al azar los Yes. Se

observa que hay regiones claras de puntos verdes donde la probabilidad de Yes es cero. Los puntos No (verdes) se detectan en su mayoría (alta especificidad). También se puede decir que sabemos gracias al modelo y a las variables input, en qué región buscar los puntos rojos. Dentro del contexto del problema, sabemos en términos generales cuándo *puede* haber peligro de pico de ozono (puntos rojos), y cuando no lo hay; aunque cuando decimos que hay peligro nos podamos equivocar dando lugar a falsos positivos.

10

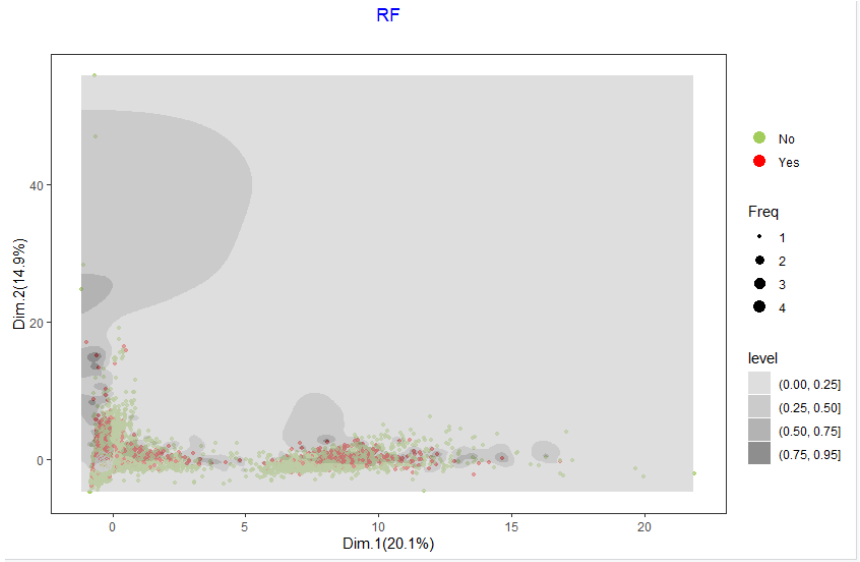
CERVEZA

https://www.kaggle.com/jtrofe/beer-recipes

Se trata de 70871 observaciones sobre tipos de cervezas y sus características. He omitido los missings del archivo original .

Es un archivo útil para plantearse diferentes problemas, pues la variable dependiente "style" es multiclase y se puede elaborar un modelo predictivo binario para cada clase. El número de observaciones en cada clase varía de 2 a 11940 y su separabilidad con las variables input también es variable.

Realizamos el ejemplo uniendo todas las variedades IPA creando la variable estilo que será "Yes" si style es alguna de las cervezas IPA y "No" si no. Igualmente se podrá plantear cualquier problema de clasificación binaria con otras clases o uniones de clases. Hay 16506 observaciones de cervezas IPA.



obs: 70871
obs clase minoritaria: 16506
% clase minoritaria: 23 %
tasa fallos: 0.118116
auc: 0.9213876
sensitivity: 0.6778141
specificity: 0.9438425
numero de Yes predichos: 14241
FP: 3053
FN: 5318
VP: 11188
VN: 51312

corte: 0.3

obs: 70871
obs clase minoritaria: 16506
% clase minoritaria: 23 %
tasa fallos: 0.1296017
auc: 0.9213876
sensitivity: 0.8168545
specificity: 0.886655
numero de Yes predichos: 19645
FP: 6162
FN: 3023
VP: 13483
VN: 48203

Se utiliza random forest, con más o menos buenos valores globales de especificidad (0.94) y sensitividad (0.67). La regresión logística da peores resultados. Claro que todo depende de nuestras expectativas: en el ejemplo se obtienen nada menos que 3053 falsos positivos y 5318 falsos negativos. Si se desea aumentar la sensitividad a costa de obtener más falsos positivos se puede hacer; como ejemplo se prueba con corte=0.3, obteniendo una sensitividad de 0.81 pero bajando la especificidad a 0.88 con 6162 falsos positivos y 3203 falsos negativos.

© Javier Portela 2020

La subcultura del "imbalanced dataset"

Existe un conjunto de teorías que abarca muchos artículos en literatura científica, a menudo en revistas con factor de impacto en el ámbito más bien informático, pero nunca en revistas del ámbito de la probabilidad y estadística. Es una teoría que afirma que tener datos desequilibrados entre las dos clases origina problemas per se.

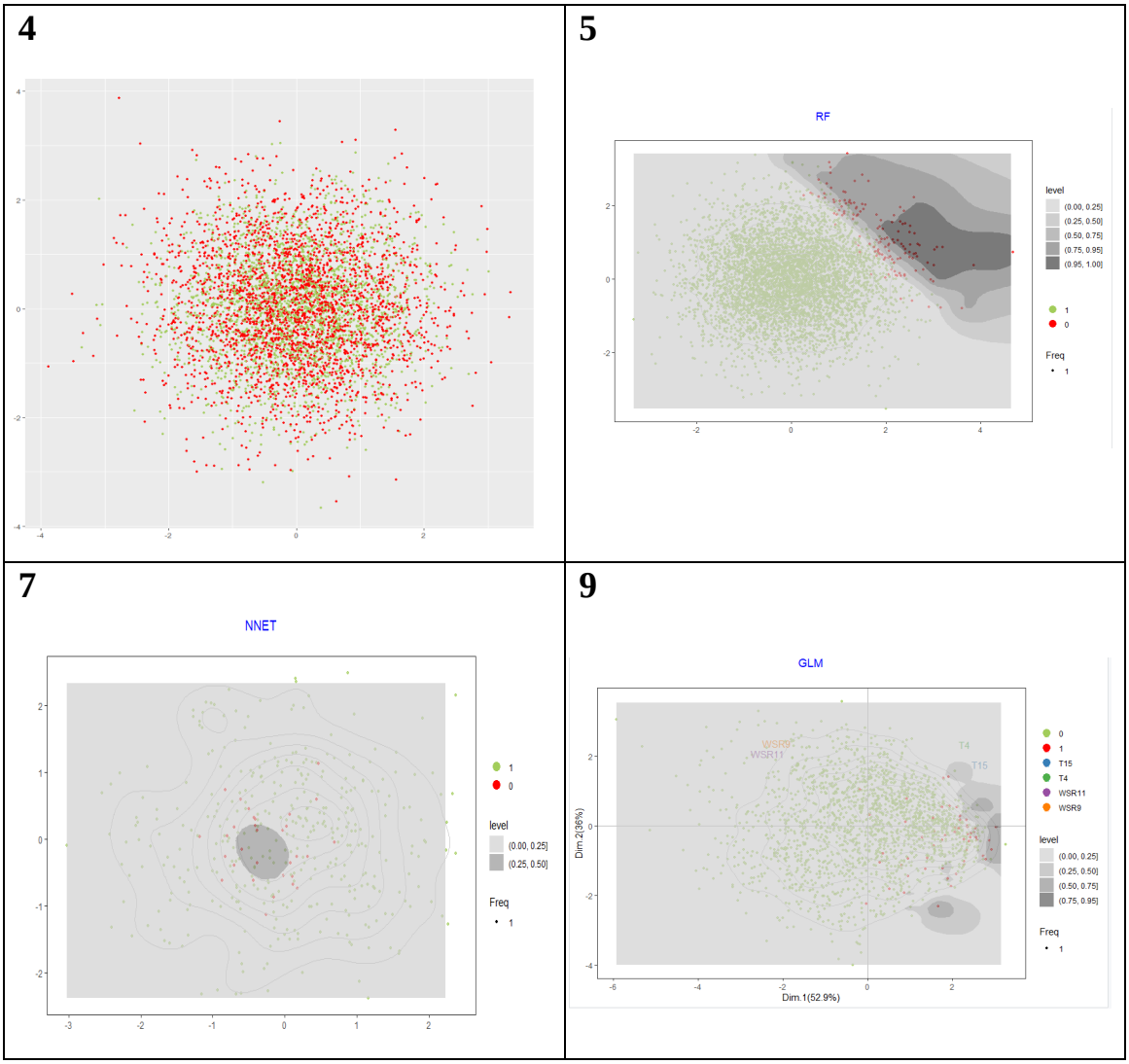
No estoy de acuerdo con la mayor parte de estas teorías. Todo lo que viene a continuación es simplemente mi opinión y no tenéis por qué compartirla.

Dos argumentos básicos sustentan las teorías del imbalanced dataset:

1) Se afirma que los algoritmos tienen tendencia a la clase mayoritaria o se ven afectados por la desproporción entre clases. A veces incluso se afirma que la teoría que subyace bajo los algoritmos asume que los datos están equilibrados y cosas parecidas. O es directamente falso, o está al menos desencaminado; los algoritmos funcionan eficientemente, detectando regiones de separación y estimando correctamente las probabilidades de Yes de cada observación.

Esto se puede ver en el ejemplo 5, donde a pesar del desequilibrio de un 3% se estiman correctamente las probabilidades tanto altas como bajas de rojo, y también en los ejemplos 7 o 9, donde hay mala separación pero el algoritmo detecta correctamente las regiones y también, correctamente, asigna probabilidades menores que 0.5 a muchos puntos rojos, e incluso a todos, porque así debe ser, dada la información de las variables input que tenemos.

2) Otro argumento que aparece en estas teorías son afirmaciones sobre que las medidas de diagnostico habituales están mal planteadas en casos de desequilibrio. Esto puede ser verdad en muchos casos, como hemos visto en varios ejemplos, y en cada caso hay que tomar decisiones sobre puntos de corte, eventualmente asignando costes, etc. pero esto no quiere decir que el desequilibrio entre clases sea un problema en sí. Por ejemplo, en el ejemplo 5 con el punto de corte 0.5 tenemos buenas medidas de diagnostico, porque existe una buena separabilidad. Y nos puede convenir asignar el mismo coste a FP que a FN, es decir, quedarnos con el punto de corte 0.5.



El ejemplo 4 tiene clases perfectamente equilibradas pero eso no tiene por qué conllevar una buena separación. Por otro lado, el desequilibrio de clases en el ejemplo 5 no impide que la separación sea muy buena. En los ejemplos 7 y 9 se ve como los algoritmos detectan correctamente las regiones de interés y asignan probabilidades racionalmente a esas regiones. Los algoritmos no se ven afectados por la desproporción, pues su funcionamiento es correcto teniendo en cuenta la información disponible.

Resumiré por qué pienso que estas teorías no son acertadas:

El problema nunca está en la desproporción, sino en que la naturaleza de nuestros datos (variables input utilizadas) no permite separar, o bien que la escasa representación de la clase minoritaria *en número absoluto de observaciones* no permite un modelo fiable.

Antes de decir que el problema está en la desproporción, observar si hay problema de separabilidad (nuestras variables no permiten separar suficientemente bien), o bien observar si el número de observaciones en la clase minoritaria es demasiado pequeño (20 observaciones son demasiado pocas por ejemplo para adoptar cualquier modelo predictivo, originando modelos con alta varianza).

© Javier Portela 2020

Tampoco se debe decir que el problema es que la desproporción afecta a como predice nuestro algoritmo, pues cuando hay mala separabilidad, el algoritmo en general detecta bien las regiones y estima las probabilidades, y es que simplemente no se puede hacer mejor con esos datos/variables input.

Normalmente en los artículos sobre "imbalanced dataset", cuando presentan un dataset problemático, nunca llegan a distinguir entre el problema de separabilidad y el de desproporción. Simplemente afirman que la falta de separabilidad se debe a la desproporción, y no simplemente a la falta de información útil (ausencia de mejores variables input). O bien afirman que la falta de separabilidad se debe a que el algoritmo tiene problemas con esos datos porque están desequilibrados, y no porque con las variables input disponibles ningún algoritmo lo puede hacer bien. O bien en otros casos afirman que el problema de obtener malas medidas de diagnóstico está en la desproporción cuando lo que pasa es que solo tenemos 10 observaciones de la clase minoritaria y no se puede plantear ningún modelo serio con esos datos.

En todo esto hay un poco de "correlación no implica causalidad" : a menudo los problemas con datos desequilibrados que nos encontramos en la práctica ofrecen problemas de mala separabilidad, y los teóricos del imbalanced dataset interpretan que lo primero (el desequilibrio) lleva a lo segundo (mala separabilidad), cuando lo que pasa es que ocurren simultáneamente. Se puede otorgar el beneficio de la duda al argumento de que es más común en la práctica que no podamos encontrar variables capaces de discernir claramente entre las dos clases cuando hay una gran desproporción, pero no hay desarrollos teóricos al respecto (porque no hay una causalidad directa) y todo se queda en observaciones empíricas.

Defender el argumento de que la desproporción es la que origina la mala separabilidad, y no simplemente la ausencia de variables input de utilidad, no tendría mayores consecuencias, salvo cuando se pretende arreglar esa desproporción manipulando la muestra. Es el caso de los métodos "oversampling" o "undersampling". No contentos con la realidad de nuestros datos, se genera una realidad paralela que coincida con nuestros deseos.

Undersampling y oversampling

Undersampling se llama en general a quedarse con solo una porción de la clase mayoritaria con el objetivo de equilibrar algo más las clases. Esta estrategia no la considero correcta, pues lo único que se conseguirá es la apariencia de más separación y estimaciones incorrectas de las probabilidades de cada punto, de las regiones de separación, y de las medidas de diagnóstico, cuando con esas variables input en realidad no se puede hacer más.

Tampoco tiene sentido hacer undersampling con el objetivo de conseguir que el algoritmo "encuentre" mejor las regiones de separación. El algoritmo ya funciona eficientemente con todos los datos, como se ve en los ejemplos 7) o 9), lo que pasa es que arroja probabilidades estimadas más pequeñas de lo que nos gustaría pero así es la realidad y tenemos que aprender a interpretarla correctamente y a tomar nuestras decisiones de acuerdo con esa realidad.

Análogamente, cualquier técnica de "oversampling" como SMOTE y parecidas, que generan observaciones artificiales en la clase minoritaria para "equilibrar la muestra", no solo parten de una premisa falsa (i.e. que el problema está en la desproporción entre clases en lugar de estar en la separabilidad), sino que pretendiéndolo arreglar usan observaciones que no aparecen en los datos, dando lugar a estimaciones de errores, probabilidades y medidas de diagnóstico alejadas de la realidad.

En ambos casos, undersampling y oversampling, en el mejor de los casos se tendrán probabilidades, regiones y estimaciones de las medidas de diagnóstico similares a las reales. Pero a menudo se obtienen incorrectamente valores más altos de sensitividad que lo real, infraestimando el error y llevando a falsas conclusiones.

Igualmente cuando se generan los modelos, regiones de separación y probabilidades estimadas con estos métodos, al aplicarlos sobre nuevos datos de la población de origen, no vamos a tener mejor separabilidad que construyendo los modelos con los datos originales, salvo simplemente por azar. Los ejemplos empíricos que se presentan en la literatura científica al respecto no son suficiente argumento: no hay desarrollos objetivos en términos teóricos de matemáticas y estadística que soporten estos métodos.

Por último una reflexión. La realidad de la separabilidad de nuestros datos es la que es: intentar afinar demasiado o corregirla porque no estamos satisfechos con ella lleva en la mayoría de los casos al sobreajuste. No conviene forzar demasiado introduciendo variables dudosas, haciendo excesiva feature engineering o utilizando algoritmos con una gran cantidad de parámetros a tunear con el objetivo de aumentar unas centésimas la sensitividad. La mayor parte de las veces los perjuicios, en términos de varianza de nuestros modelos, son mayores que los beneficios obtenidos.

© Javier Portela 2020

Comportamiento comparativo de algoritmos

En esta sección se presentan ejemplos de comportamiento de algoritmos predictivos en diferentes datasets. Se realiza un proceso automatizado básico, con una selección stepwise de variables y un tuneado muy básico de los algoritmos. Se presenta un diagrama de cajas basado en validación cruzada repetida de los diferentes modelos y las gráficas 2d obtenidas con cada algoritmo con el paquete visualpred.

Sin entrar demasiado en el detalle de cada caso, se puede observar la buena o mala separación en cada dataset, además de ver qué algoritmos desempeñan un mejor y peor papel. La observación de las regiones de probabilidades observada en los gráficos de visualpred permite intuir algo el sobreajuste, y realmente que las diferencias entre algoritmos son a menudo pequeñas y ni siquiera se puede o tiene sentido percibirlas en un gráfico. Se recomienda poner el visor del lector en dos páginas para ver a la vez el diagrama de cajas y los gráficos visualpred.

Comentarios iniciales

Antes de revisar los ejemplos hay que tener en cuenta varias cosas:

a) En los diagramas de cajas hay que fijarse bien en los valores numéricos, a menudo las diferencias entre algoritmos son muy pequeñas.

b) Al solo presentar el auc en el diagrama de cajas , esto puede ser engañoso. Hay que recordar que en casos con proporción de clase minoritaria muy pequeña, el auc siempre es alto por la alta especificidad. Y en esos casos esas diferencias pequeñas entre auc sí significan una diferencia importante en los algoritmos.

c) Al ser a menudo no demasiado grandes las diferencias numéricas entre algoritmos (en auc o tasa de fallos), el gráfico equivalente de contour de visualpred a veces no permite reflejar bien las diferencias entre algoritmos; hay que tener cuidado en no “sobrediagnosticar” un sobreajuste, mal o buen funcionamiento o diferencias entre algoritmos básándonos únicamente en la pura observación de los gráficos.

d) Por supuesto se pueden tunear mejor los algoritmos y mejorar la selección de cada ejemplo dando lugar a resultados ligeramente diferentes de los presentados. Igualmente el diagrama de cajas se obtiene de cv repetida con ciertas semillas. En la misma situación cambiar las semillas puede resultar en resultados ligeramente diferentes debido al remuestreo.

Comentarios sobre los ejemplos

1) En general se observa que en los gráficos de cajas la diferencia con la regresión logística no es tan grande, lo que a priori nos inclinaría por ésta por sus virtudes conocidas (robusta y simple , no sobreajusta tanto, está muy estudiada y con medidas de inferencia, y tiene poder explicativo) pero en cada aplicación hay que ver qué es lo que conviene. Si el coste de fallar (en FN o FP) es muy alto y se van a predecir muestras grandes, puede convenir no utilizar la

logística, prefiriendo un algoritmo de machine learning pues en una aplicación particular, por ejemplo, la diferencia en un segundo o tercer decimal puede originar una diferencia de digamos 2000 o 3000 Falsos positivos o negativos con sus costes asociados. También hay que tener en cuenta el comentario b) anterior.

2) Hay ejemplos como acuteinf en los que la presencia de pocas variables y cualitativas origina modelos muy simples y las regiones de predicción son muy parecidas entre los algoritmos.

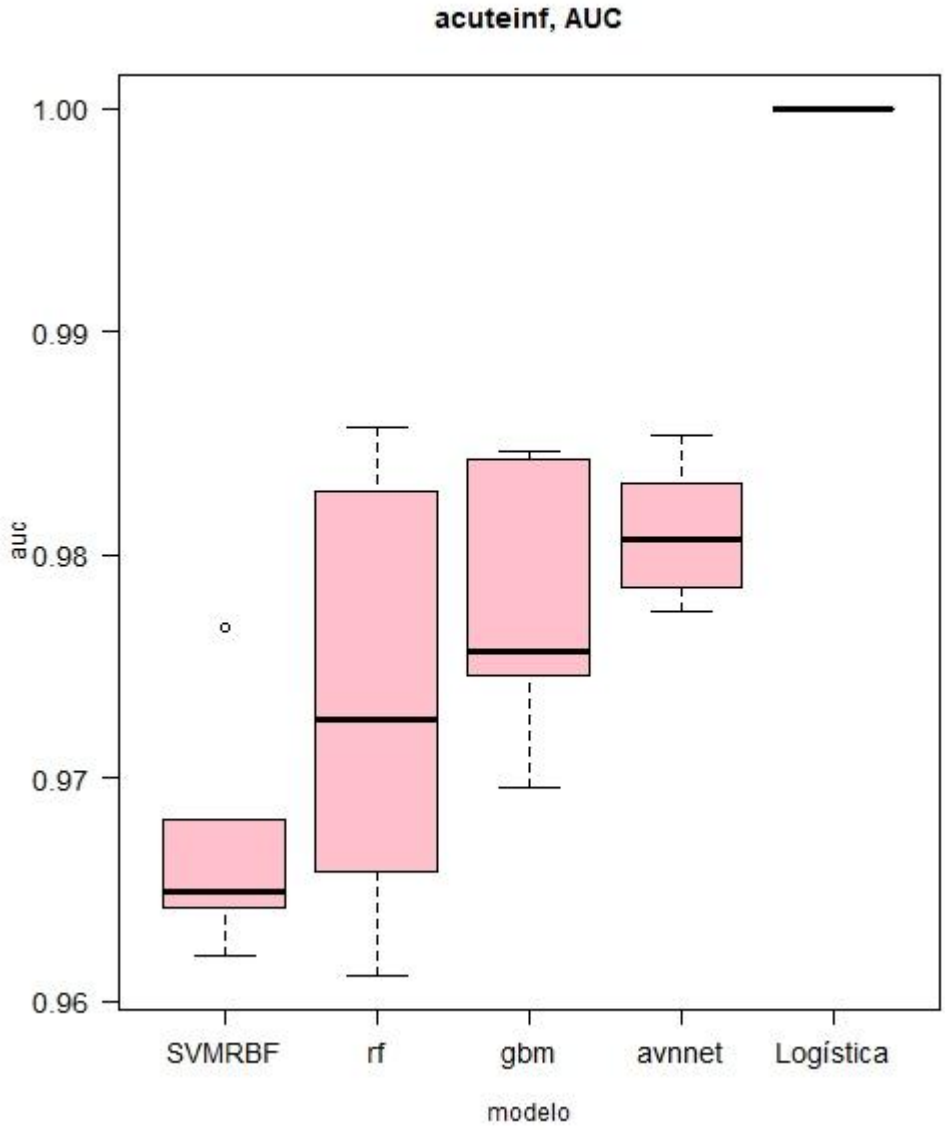
3) En algún caso la separación no es demasiado buena, como el ejemplo caravan, donde el exceso de variables posibles (y ninguna claramente buena) lleva a selección de variables con sobreajuste. Aquí el modelo más simple (logística) es el recomendable. Son datos de un concurso en el cual un método muy simple como naive bayes obtuvo la mejor performance.

4) Hay ejemplos en los que claramente la logística es peor, como bridges-relif, aunque no se aprecia en los gráficos de contour de visualpred, pero subyace que posiblemente haya interacciones que otros métodos detectan mejor.

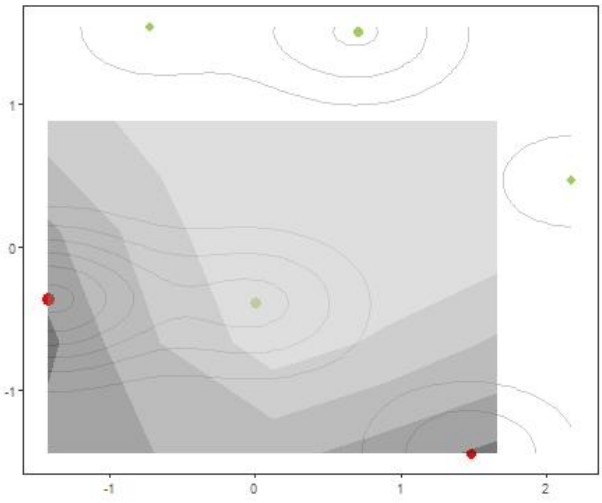
5) En el ejemplo BEPS, la diferencia con logística en auc es pequeña, y cabe preguntarse al ver los gráficos de contour si los otros métodos no estarán sobreajustando (al observarse “islas”, regiones demasiado específicas).

6) En otros ejemplos como bank, da la impresión de que la región construida por gbm está menos sobreajustada que la logística y puede que en la práctica sea mejor.

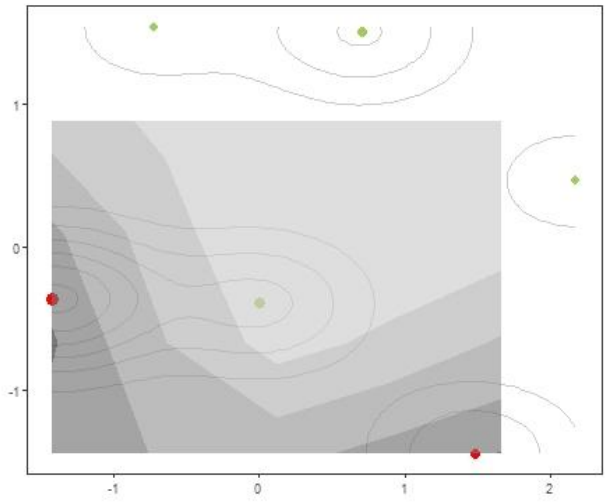
© Javier Portela 2020


acuteinf, AUC
1.00
0.99
0.98
0.97
0.96
auc
SVMRBF
rf
gbm
avnnet
Logística
modelo

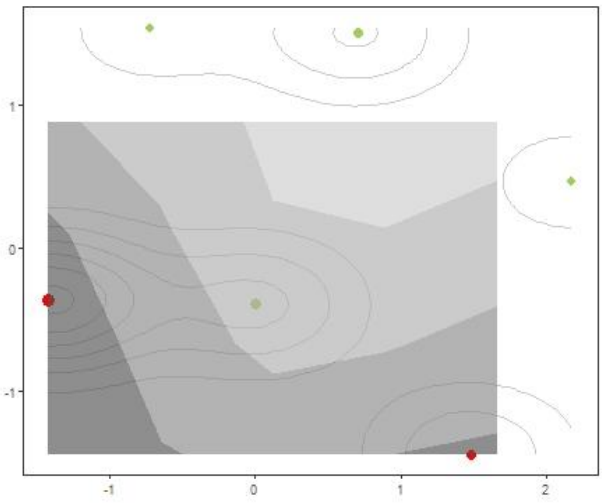
GLM



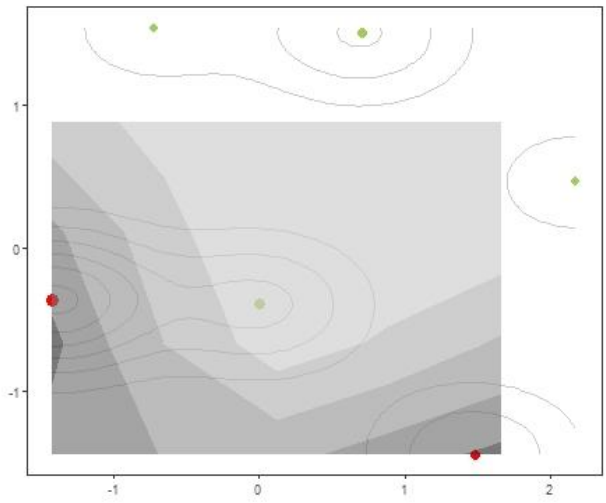
NNET



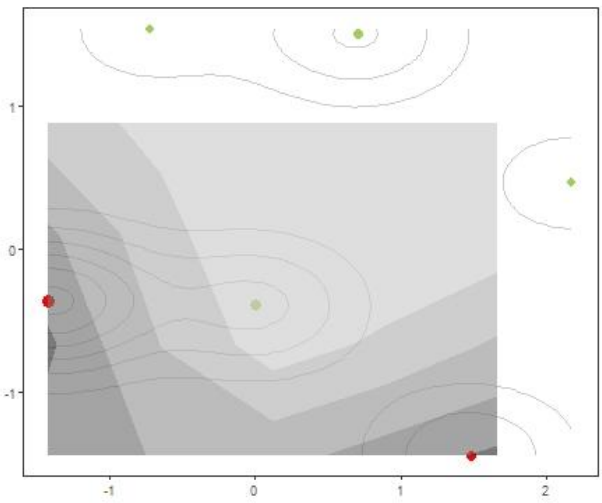
RF



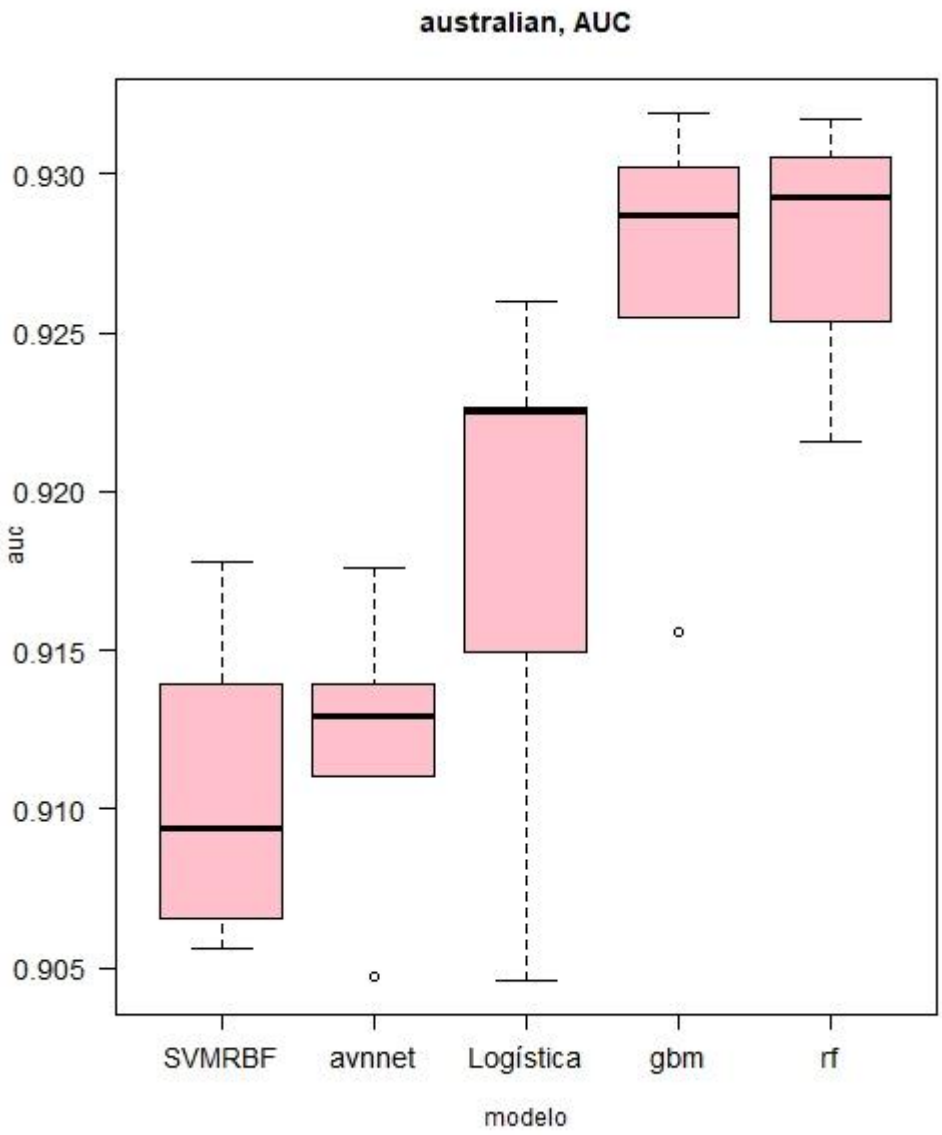
GBM



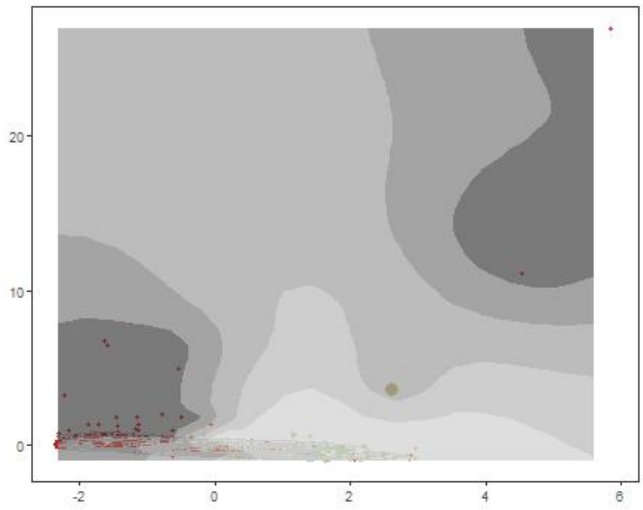
SVM



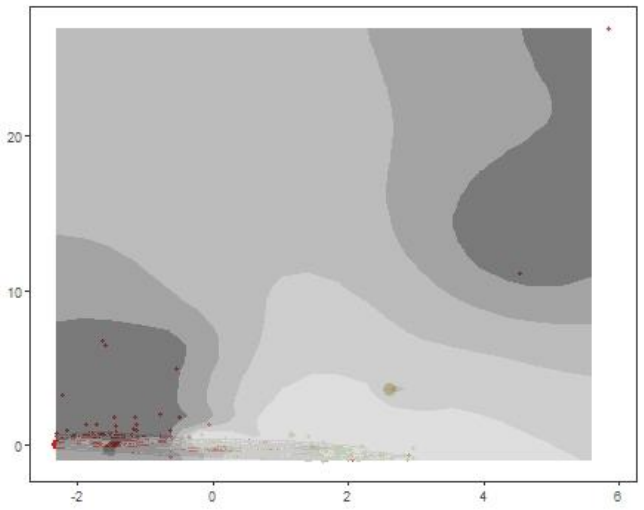
© Javier Portela 2020


australian, AUC
0.930
0.925
0.920
0.915
0.910
0.905
auc
SVMRBF
avnnet
Logística
gbm
rf
modelo

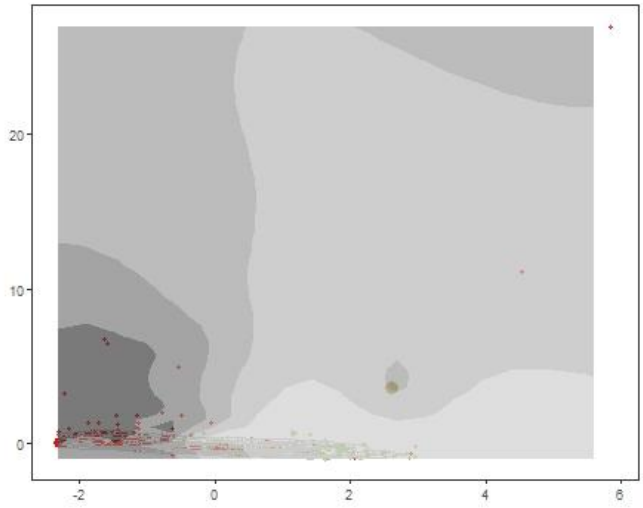
GLM



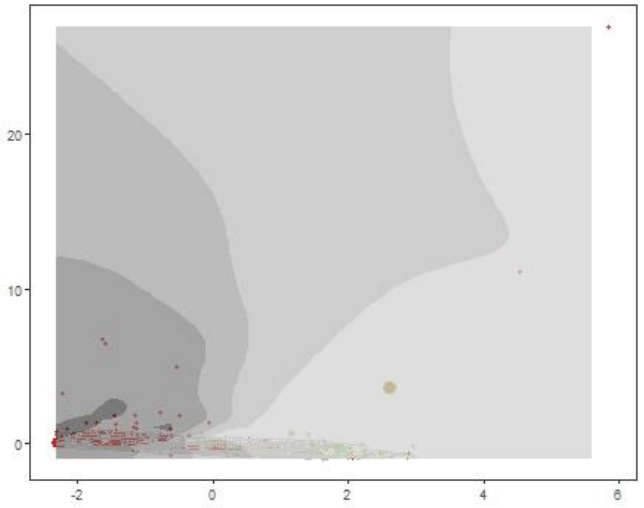
NNET



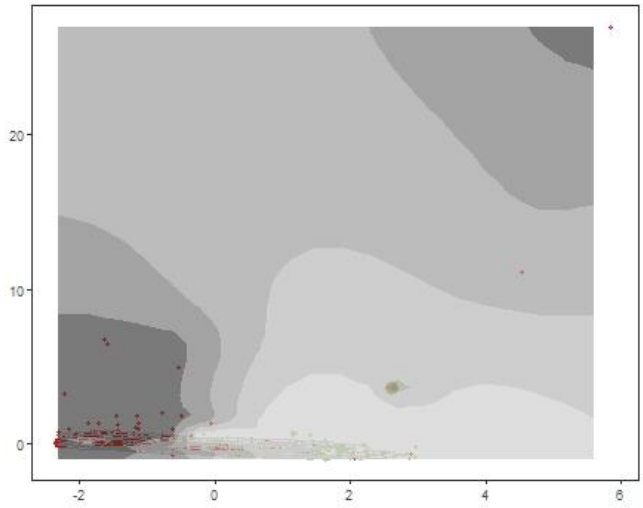
RF



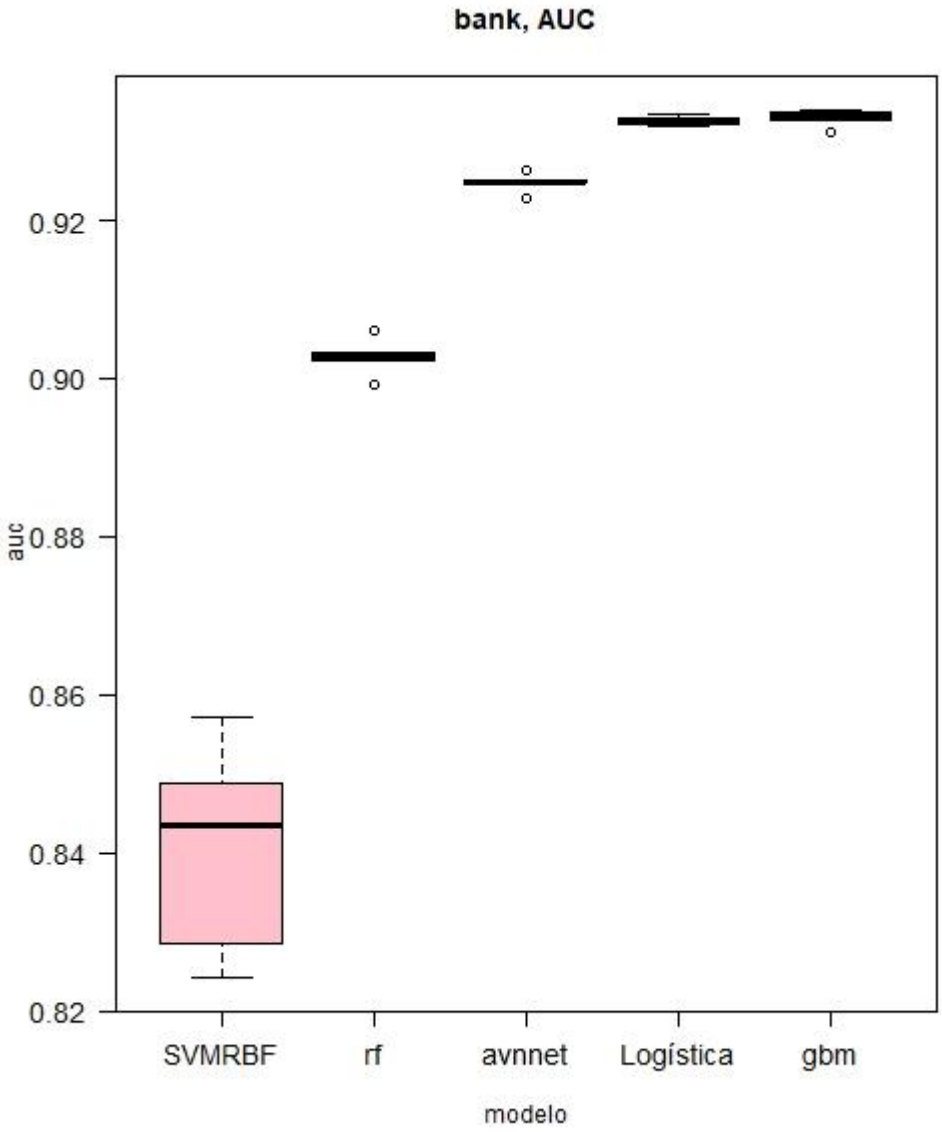
GBM



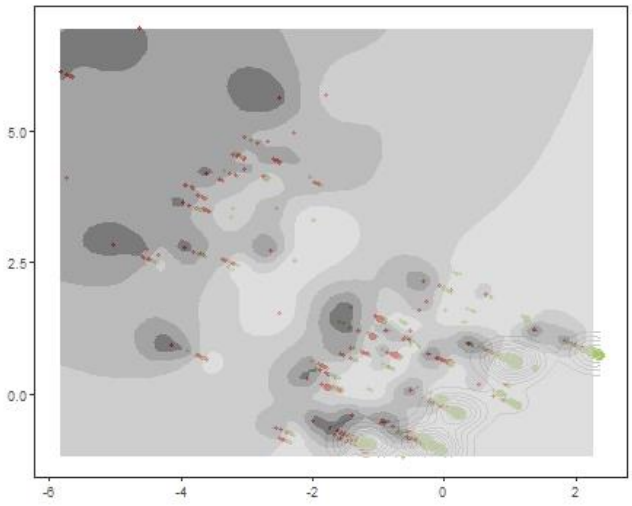
SVM



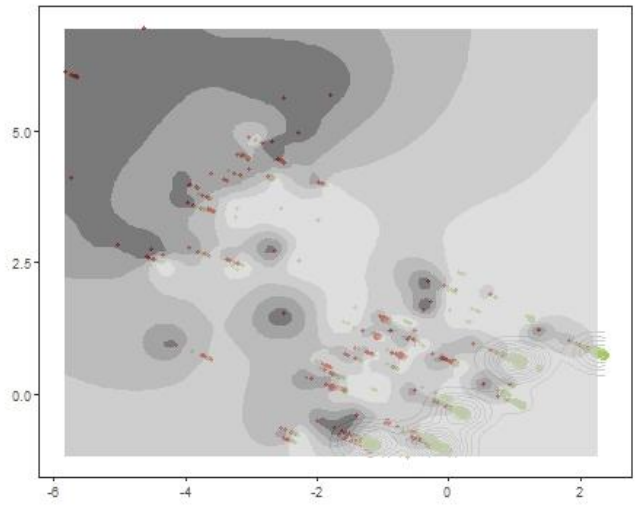
© Javier Portela 2020


bank, AUC
auc
0.92
0.90
0.88
0.86
0.84
0.82
SVMRBF
rf
avnnet
Logística
gbm
modelo

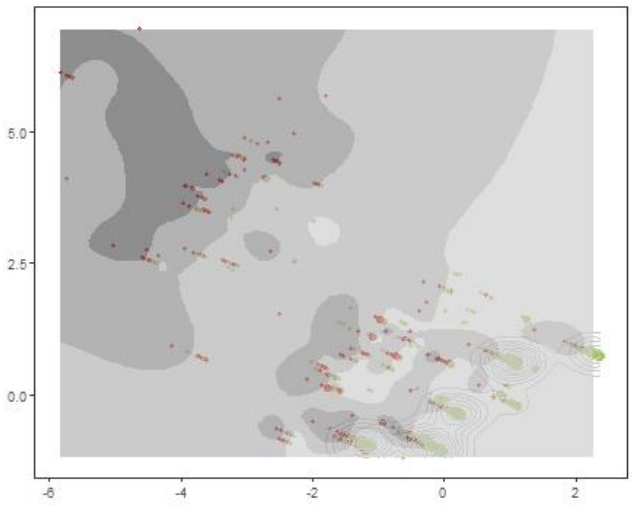
GLM



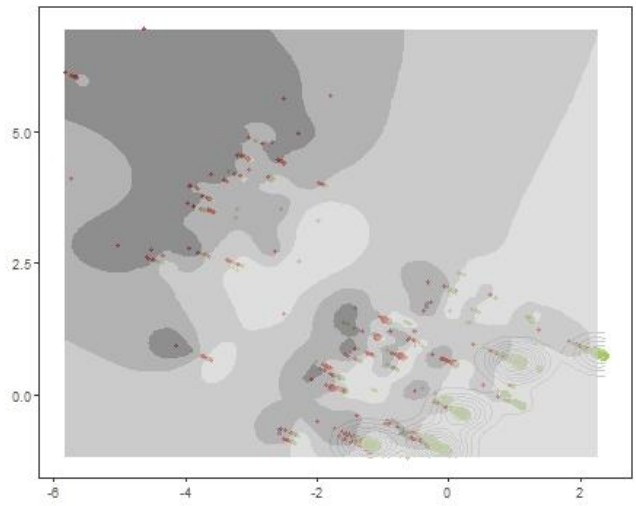
NNET



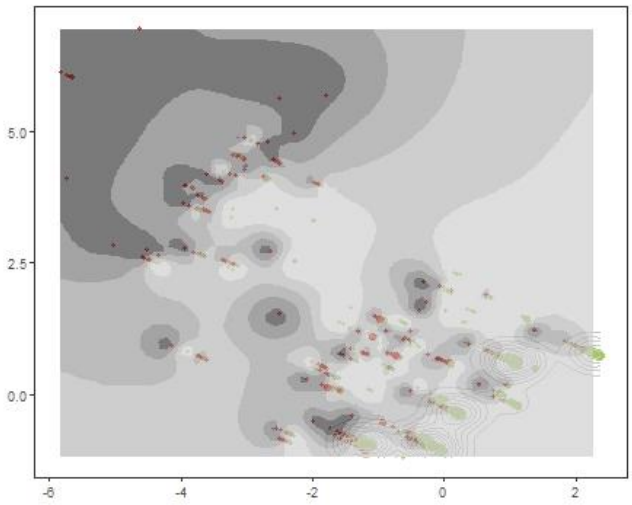
RF



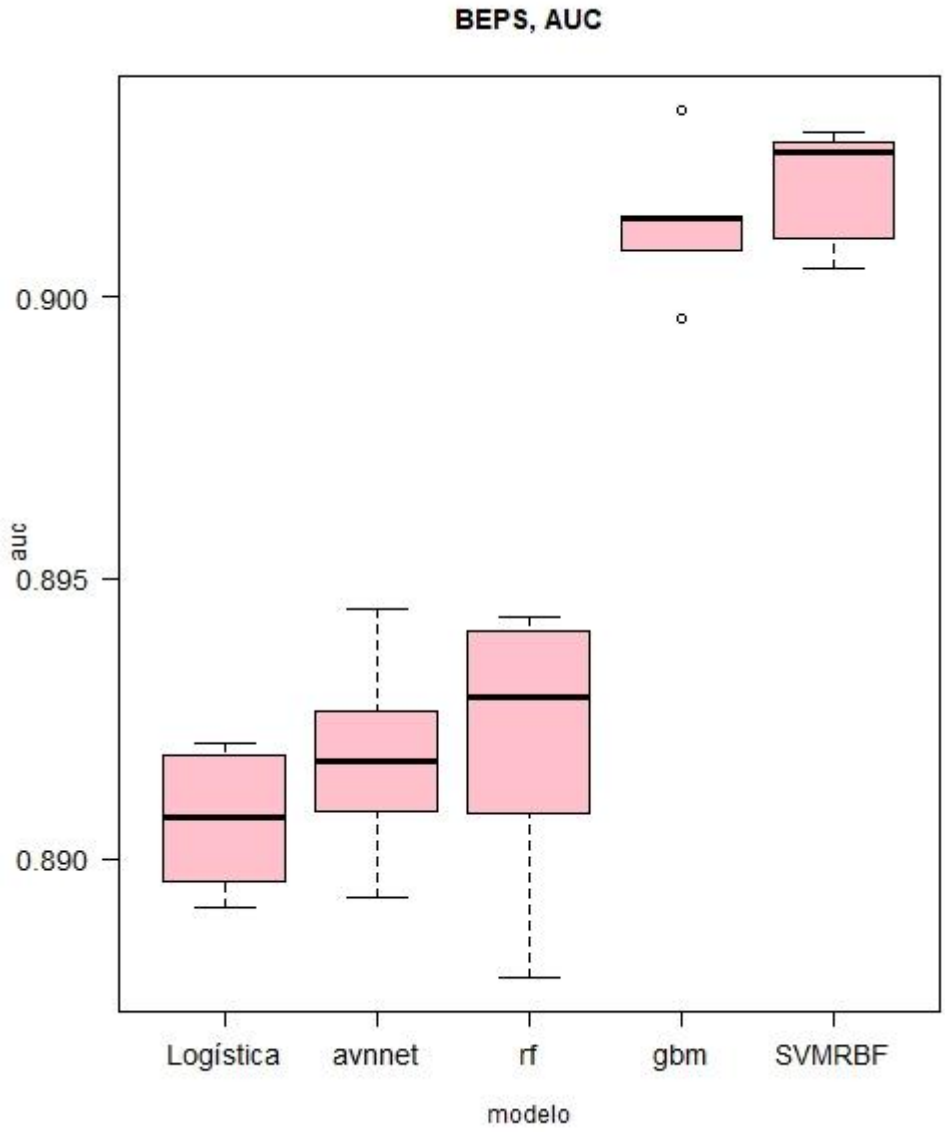
GBM



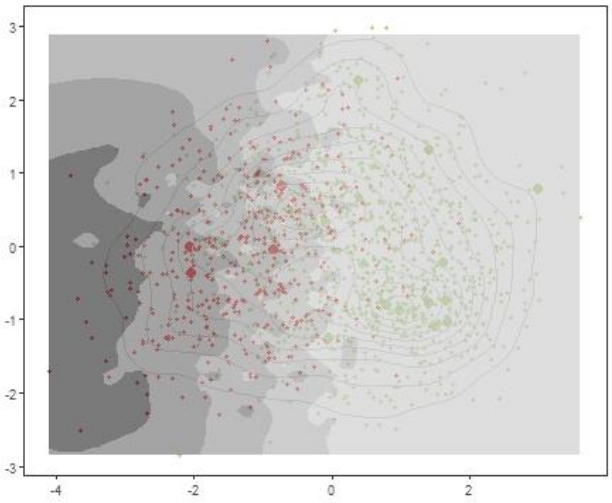
SVM



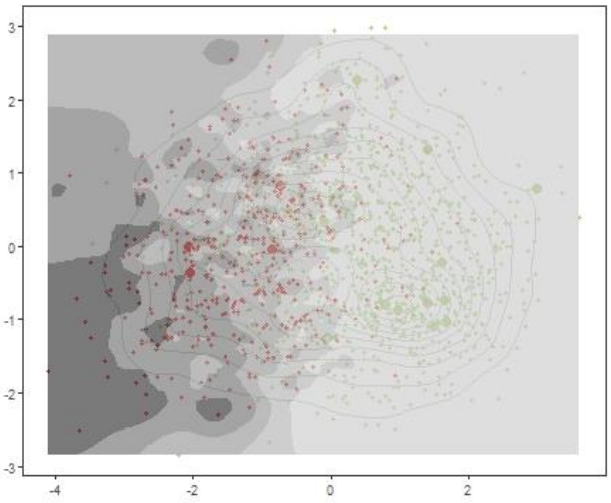
© Javier Portela 2020


BEPS, AUC
0.900
0.895
0.890
auc
Logística
avnnet
rf
gbm
SVMRBF
modelo

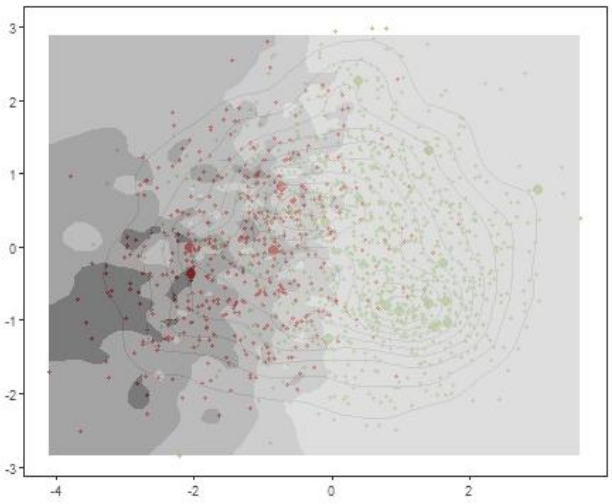
GLM



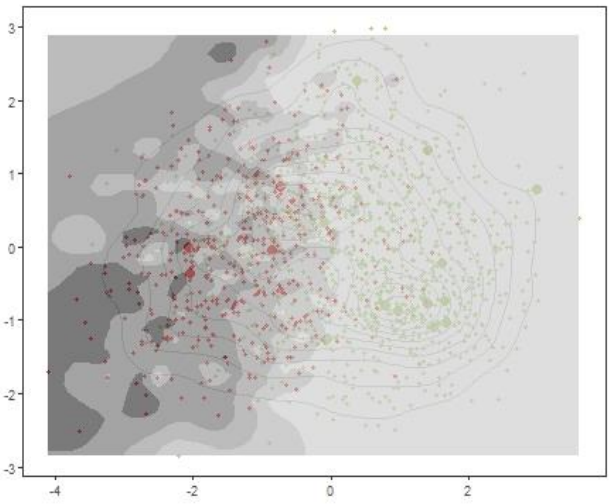
NNET



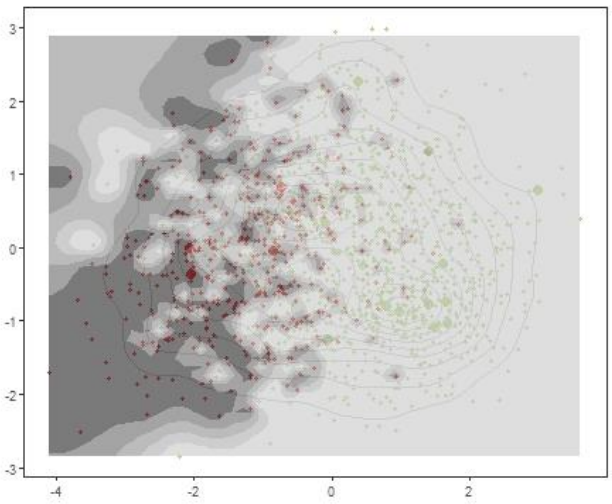
RF



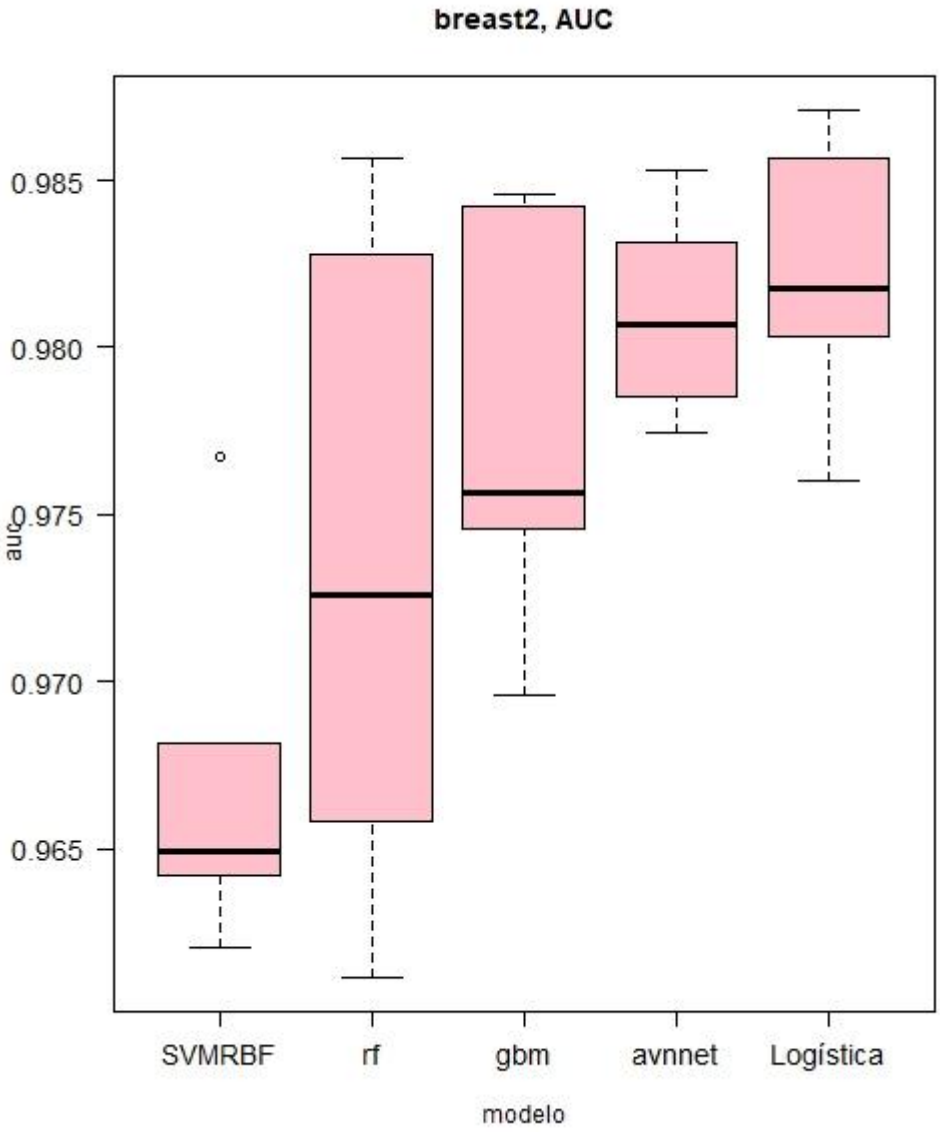
GBM



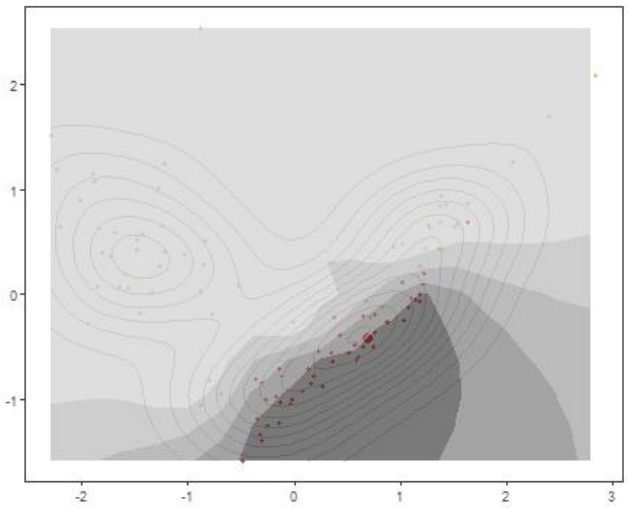
SVM



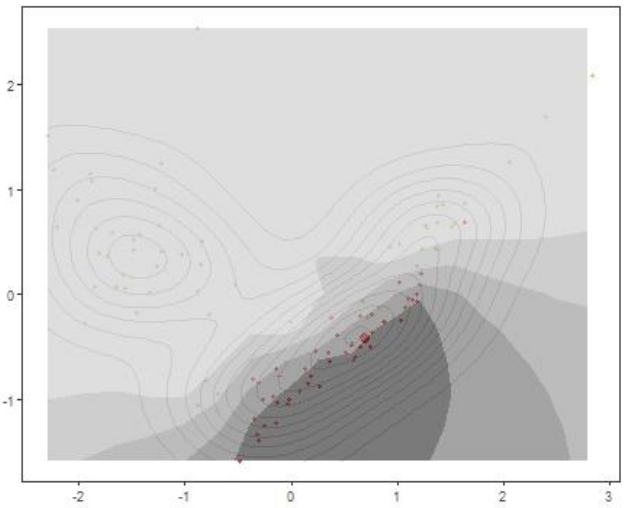
© Javier Portela 2020


breast2, AUC
0.985
0.980
0.975
0.970
0.965
auc
SVMRBF
rf
gbm
avnnet
Logística
modelo

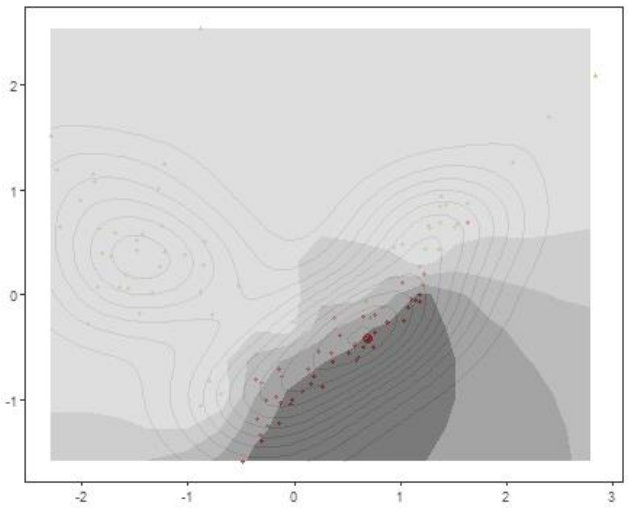
GLM



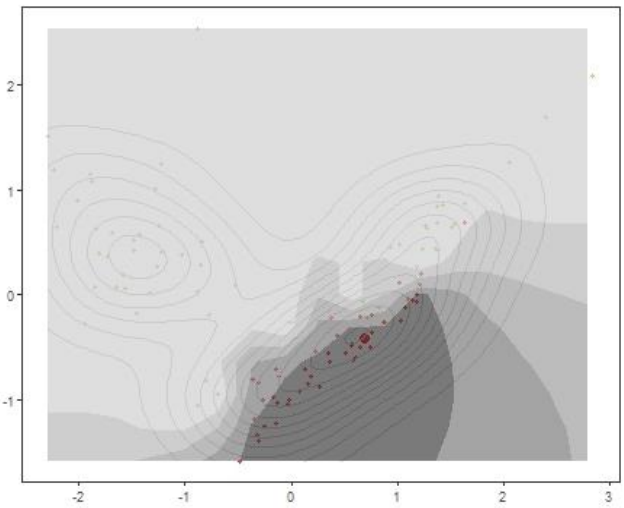
NNET



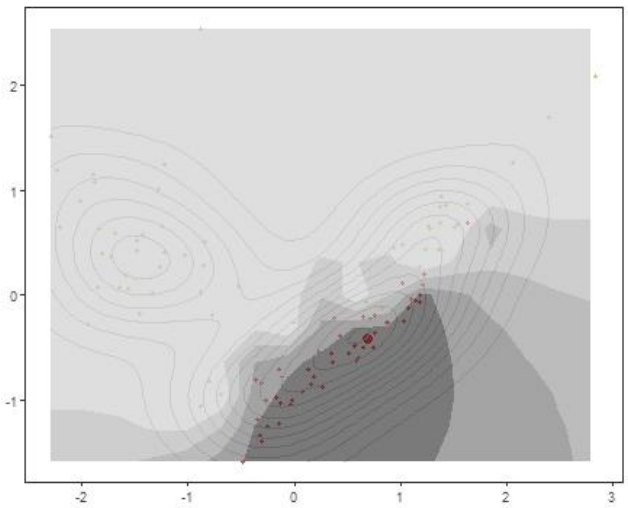
RF



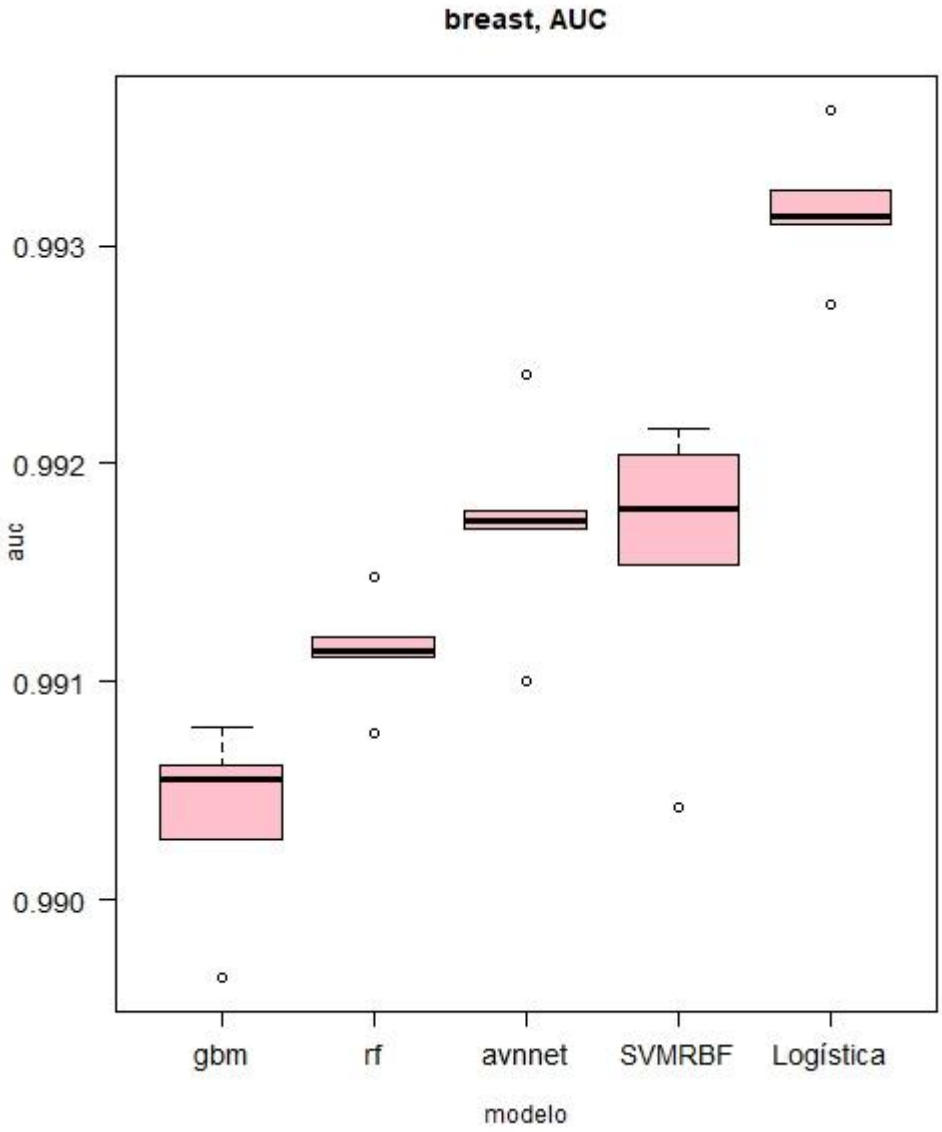
GBM

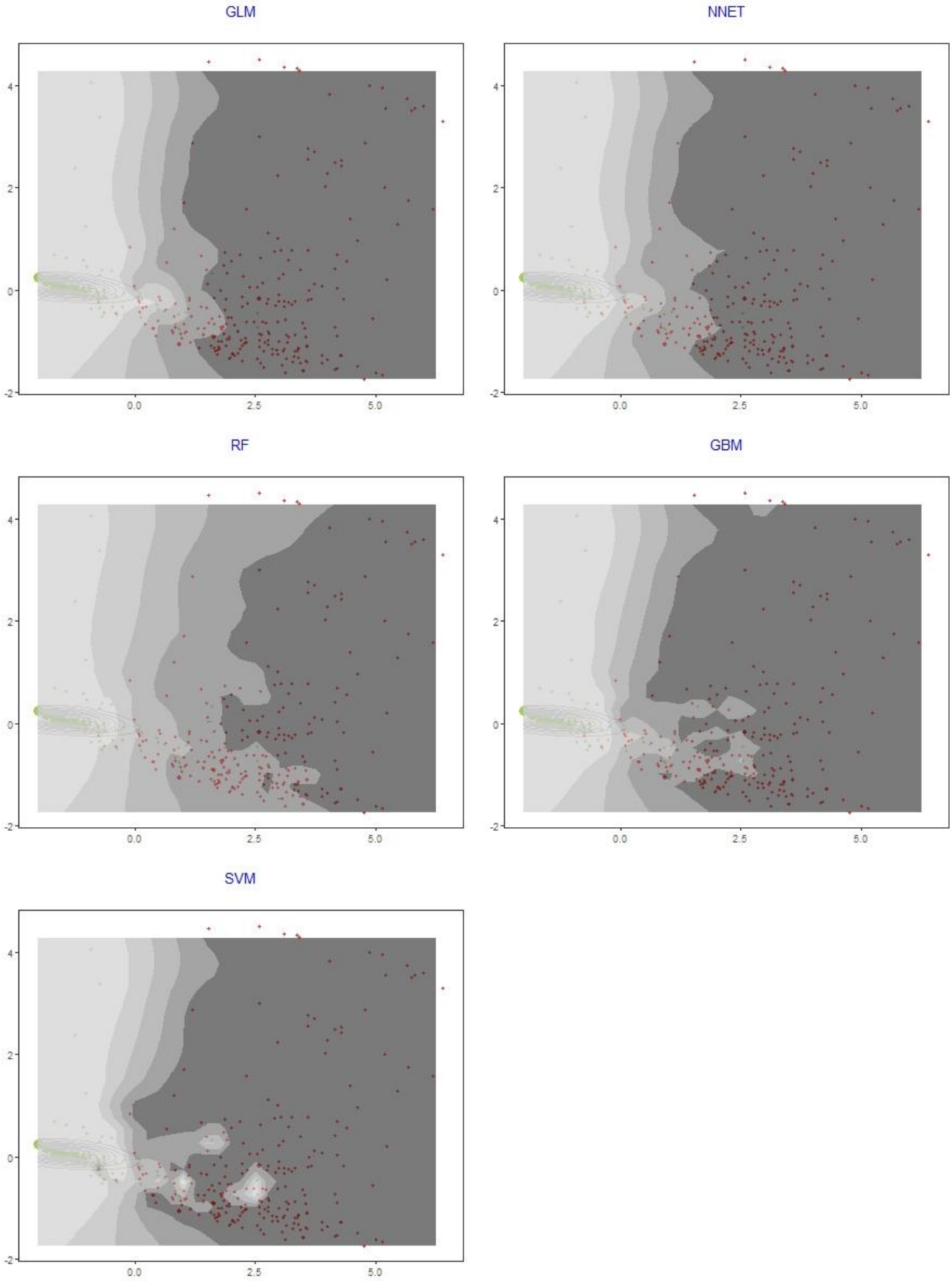


SVM

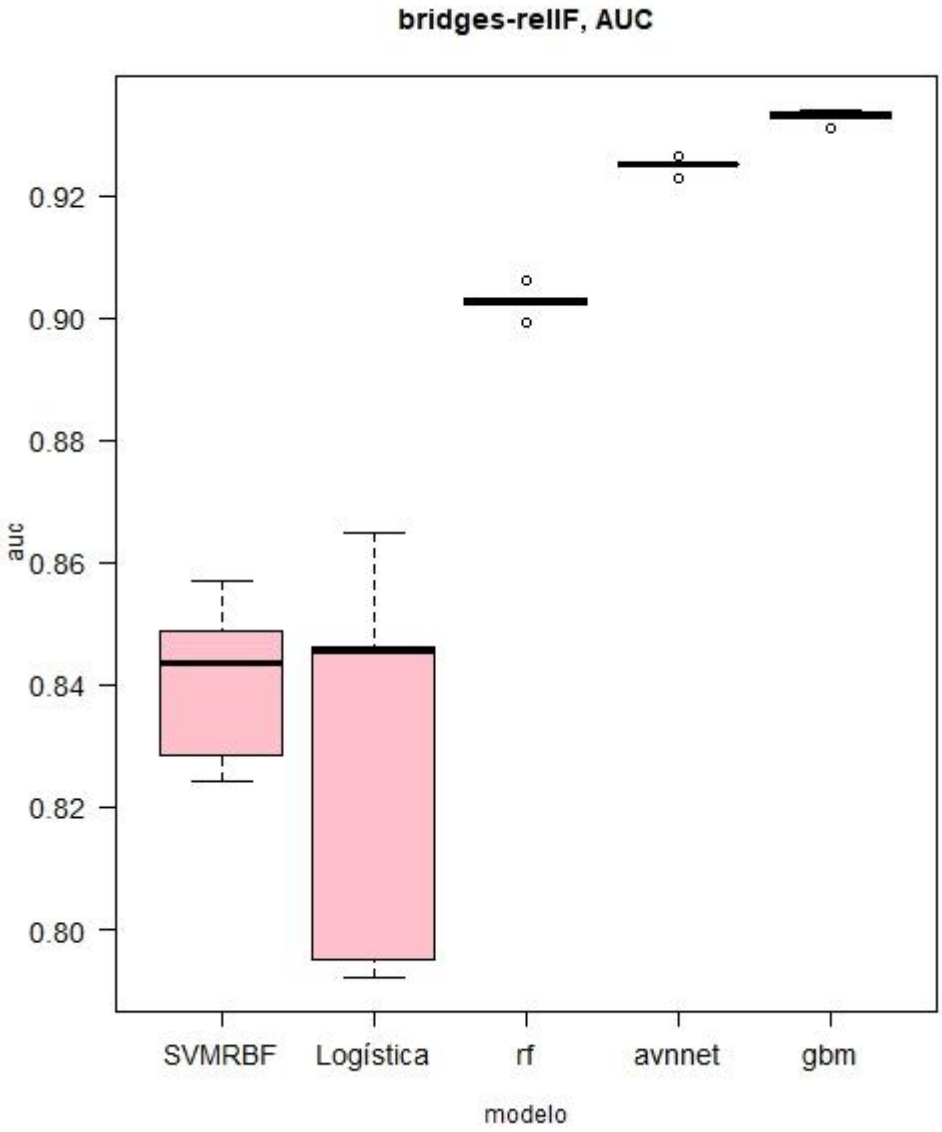


© Javier Portela 2020

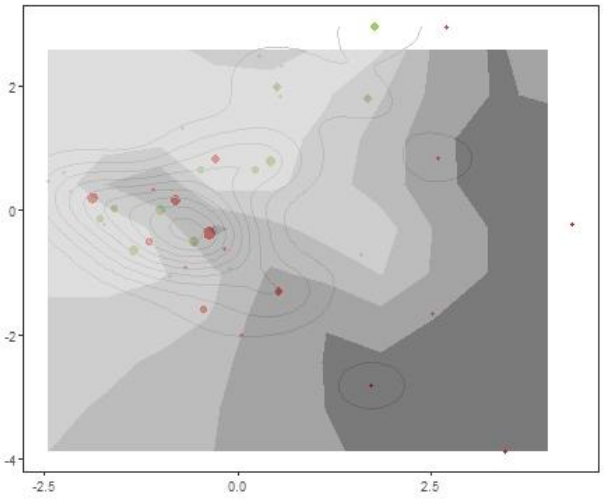

breast, AUC
0.993
0.992
0.991
0.990
auc
gbm
rf
avnnet
SVMRBF
Logística
modelo



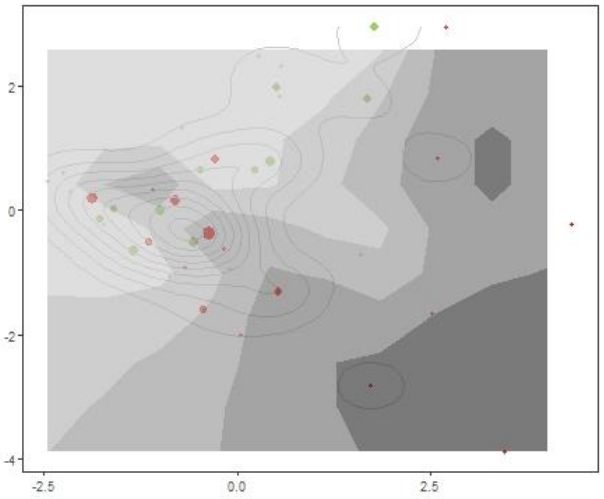
© Javier Portela 2020


bridges-relIF, AUC
auc
0.92
0.90
0.88
0.86
0.84
0.82
0.80
SVMRBF
Logística
rf
avnnet
gbm
modelo

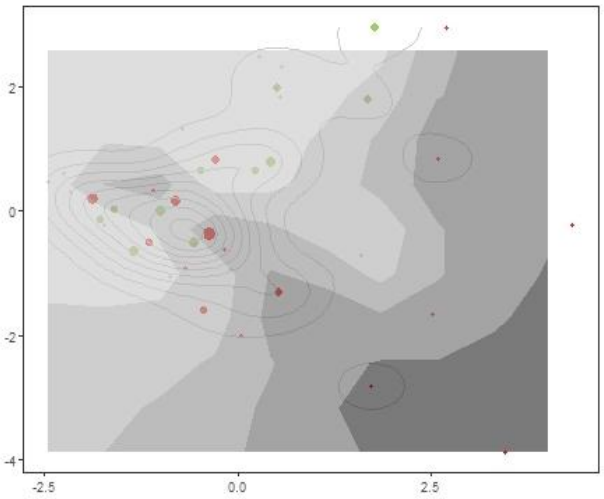
GLM



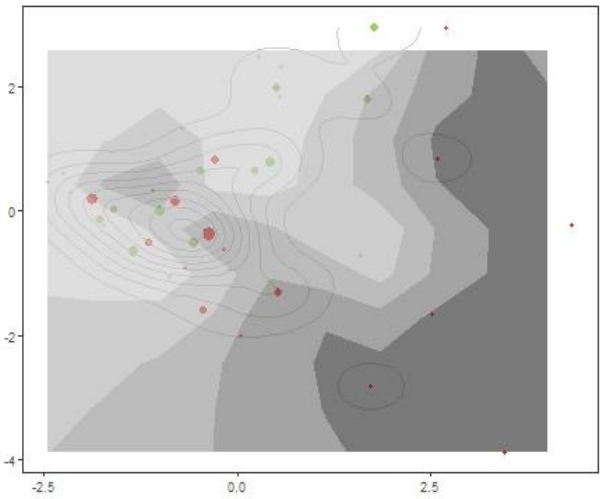
NNET



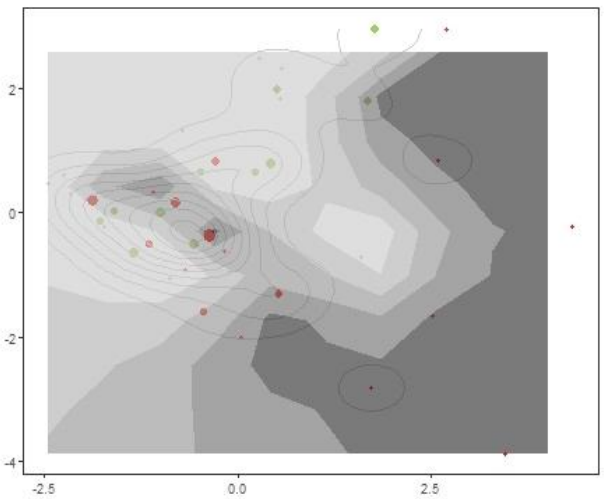
RF



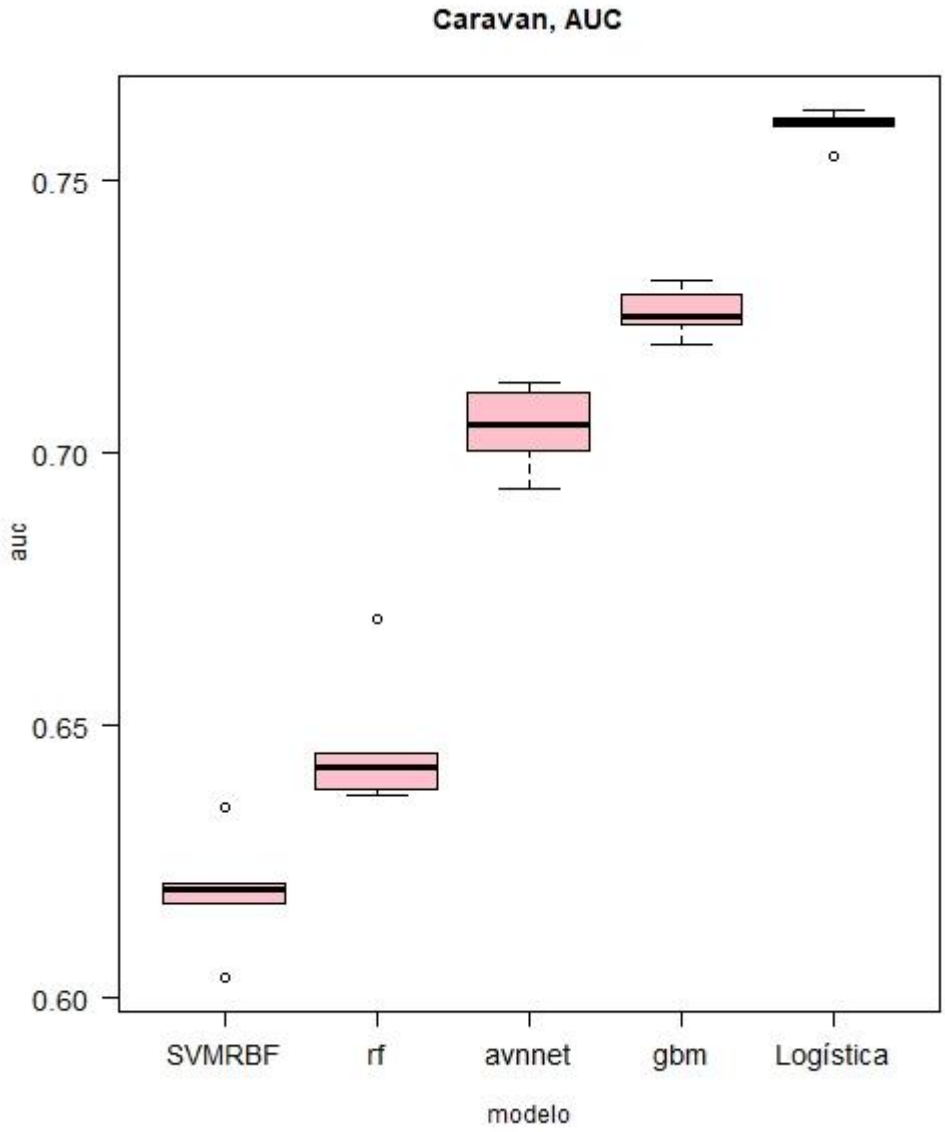
GBM



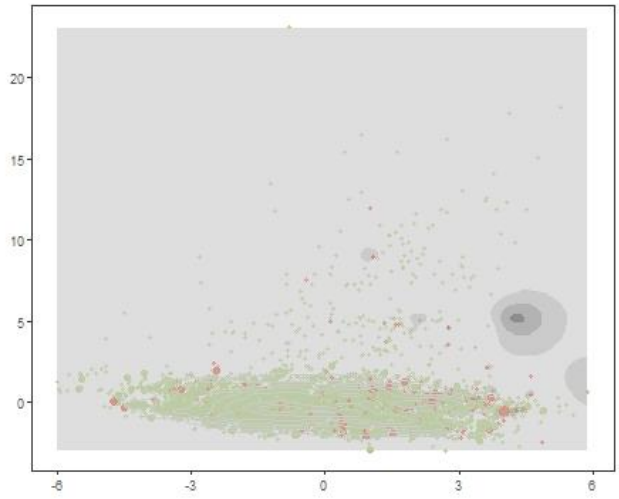
SVM



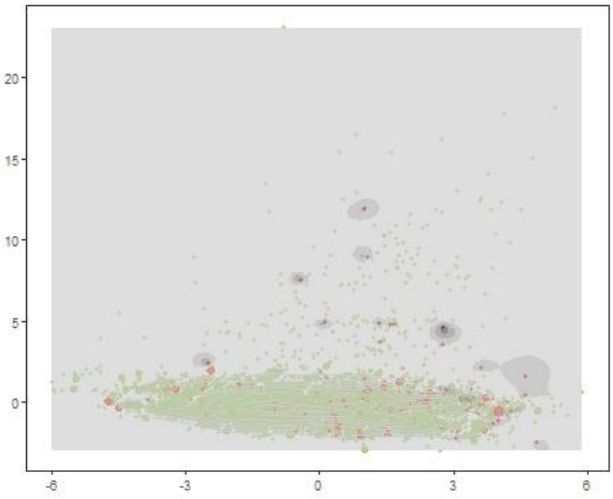
© Javier Portela 2020


Caravan, AUC
0.75
0.70
0.65
0.60
auc
SVMRBF
rf
avnnet
gbm
Logística
modelo

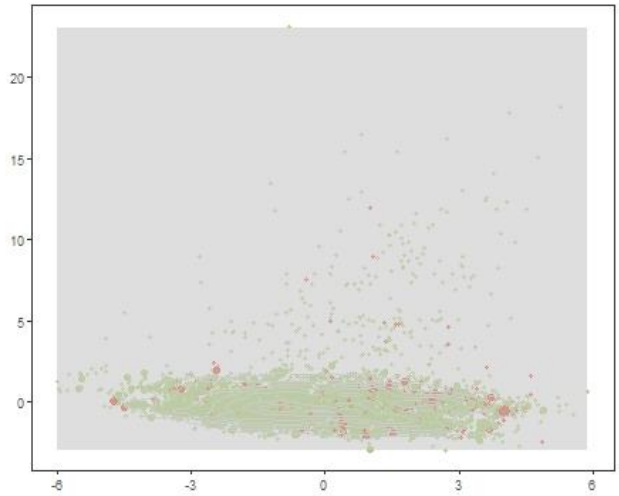
GLM



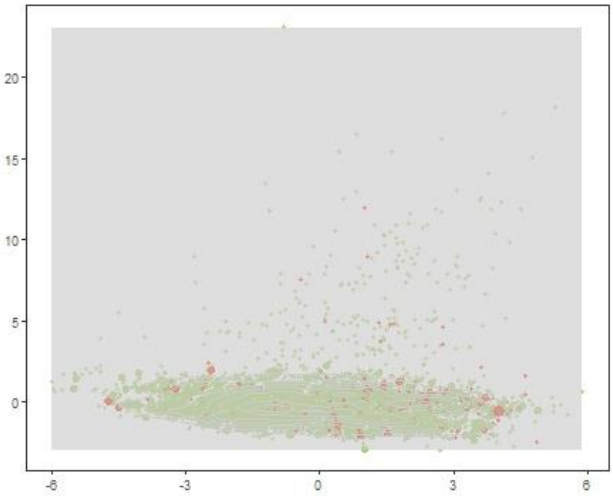
NNET



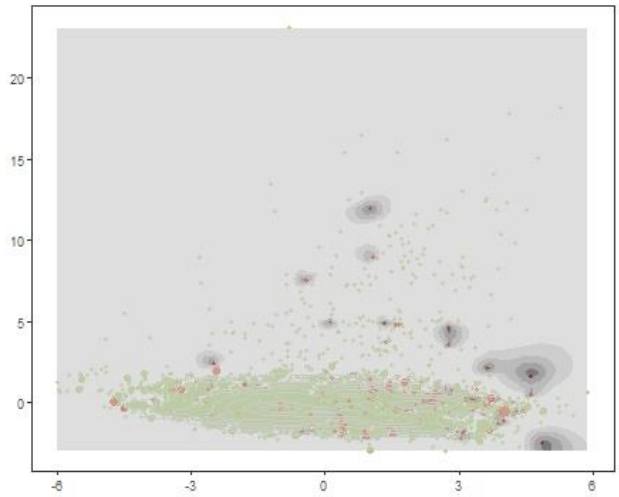
RF



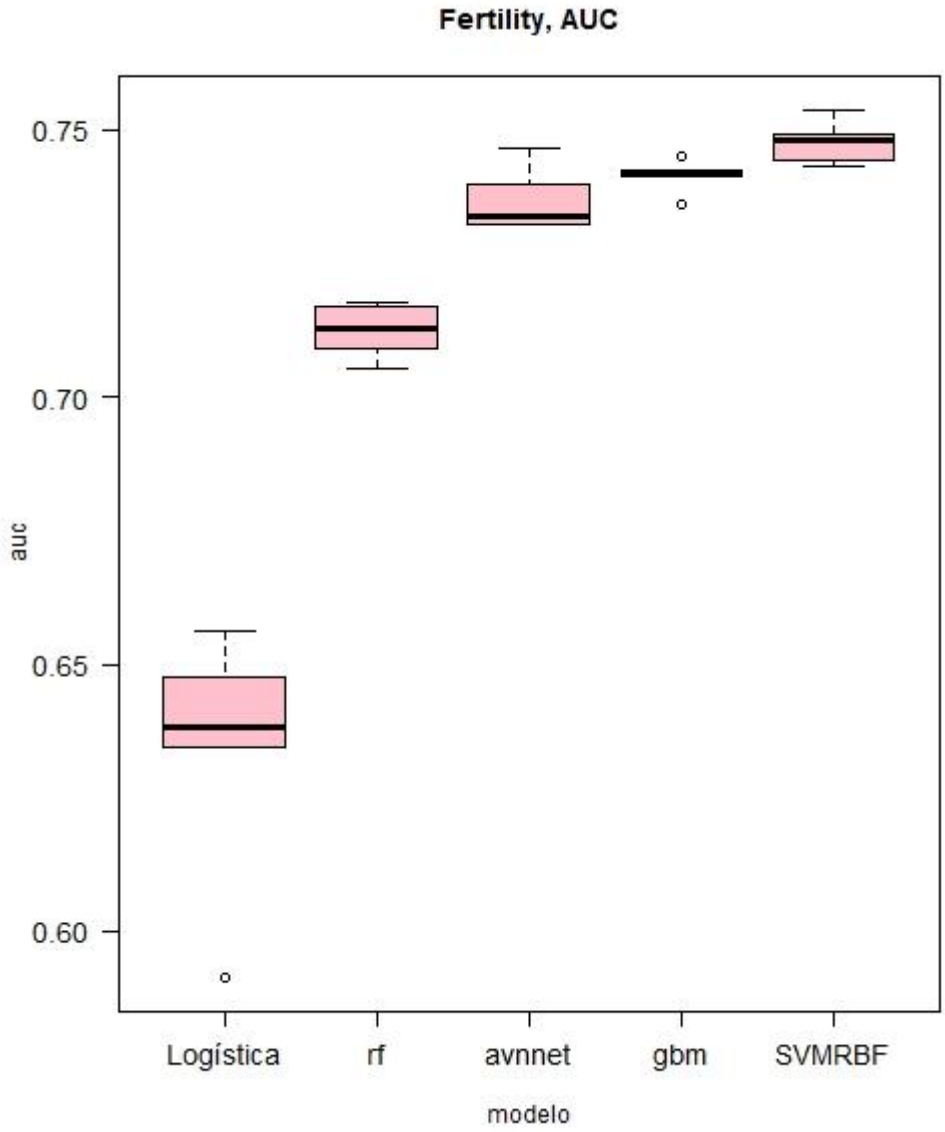
GBM

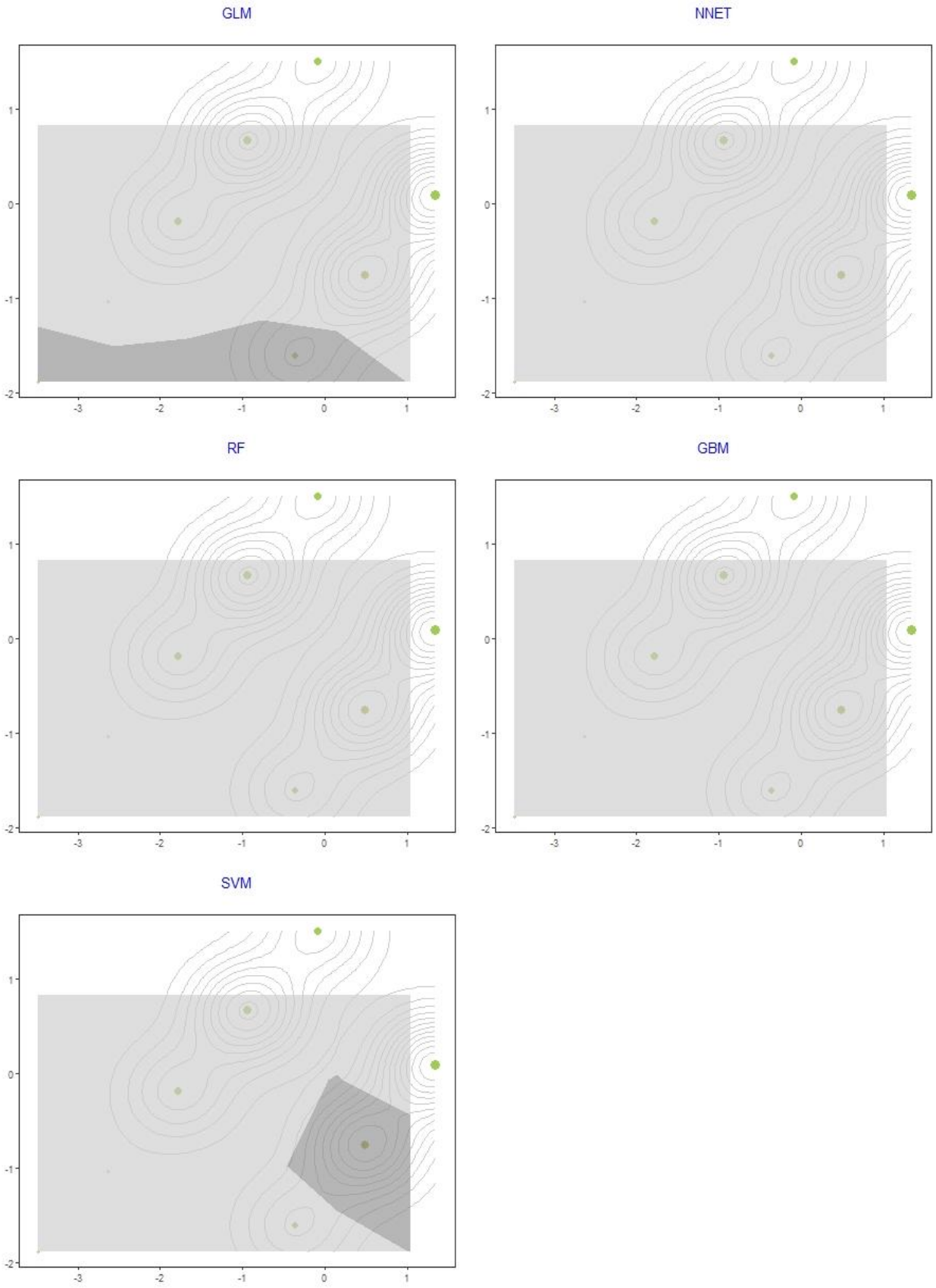


SVM

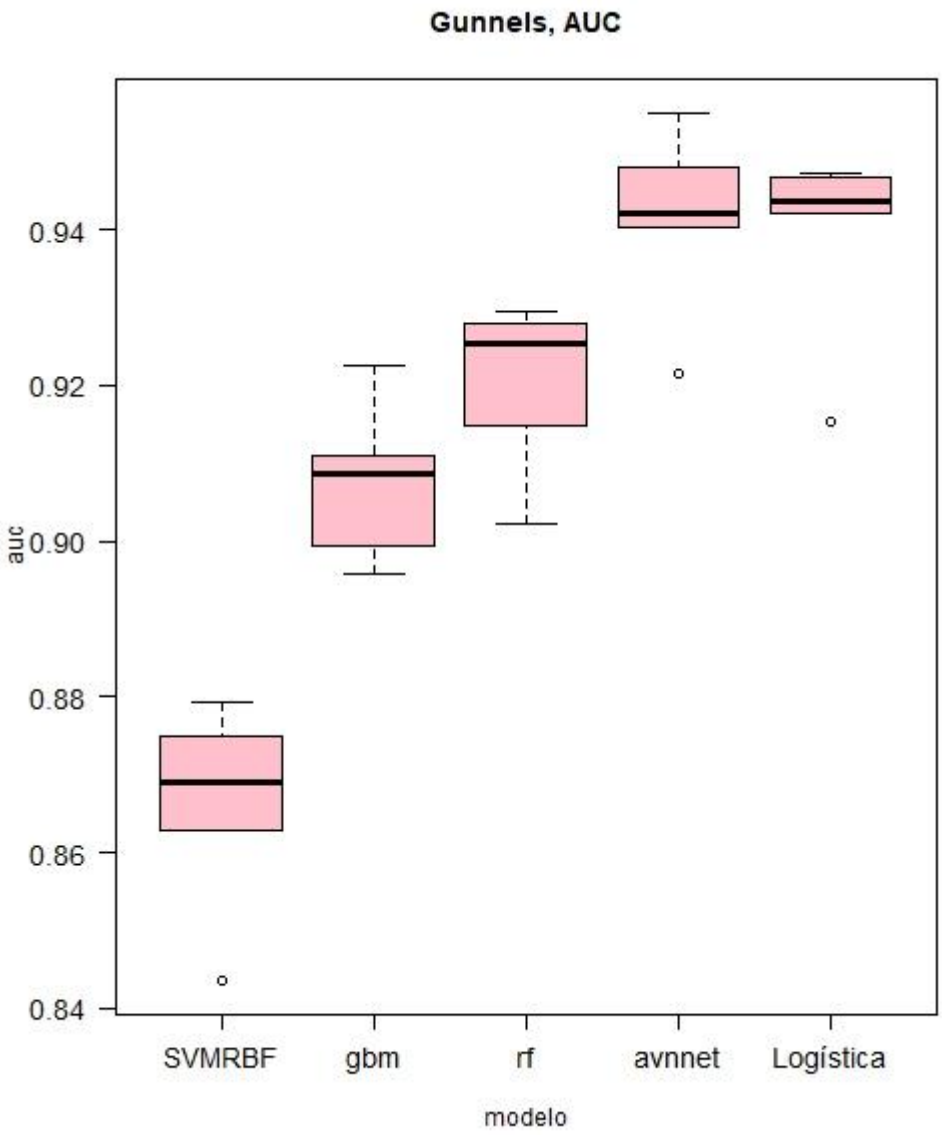


© Javier Portela 2020

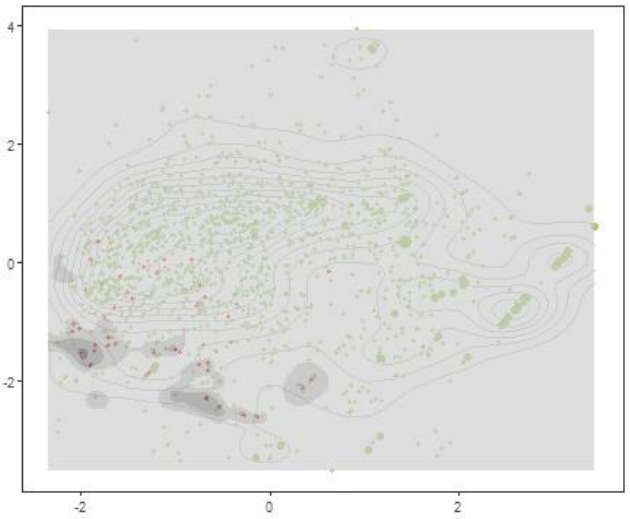

Fertility, AUC
0.75
0.70
0.65
0.60
auc
Logística
rf
avnnet
gbm
SVMRBF
modelo



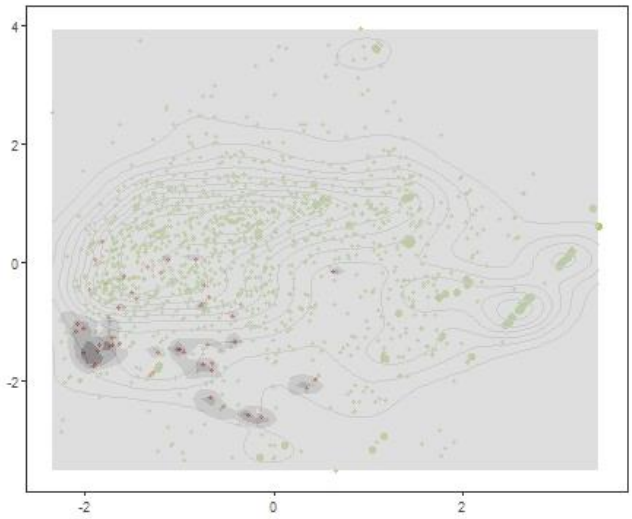
© Javier Portela 2020


Gunnels, AUC
auc
0.94
0.92
0.90
0.88
0.86
0.84
SVMRBF
gbm
rf
avnnet
Logística
modelo

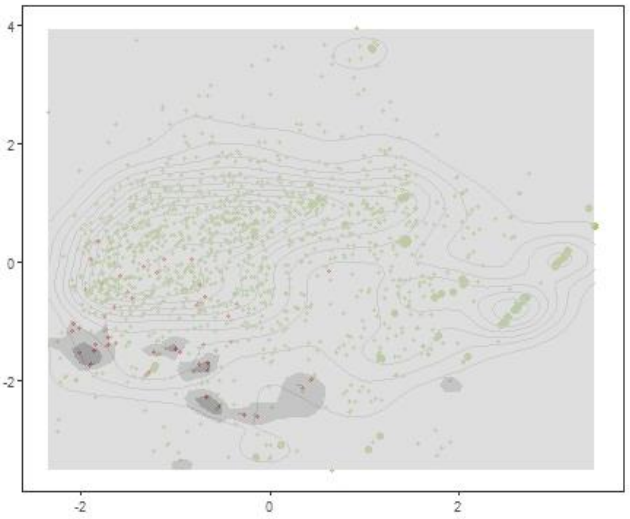
GLM



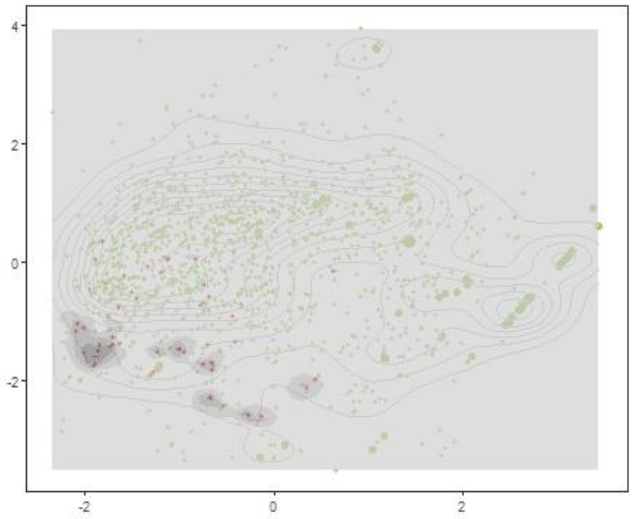
NNET



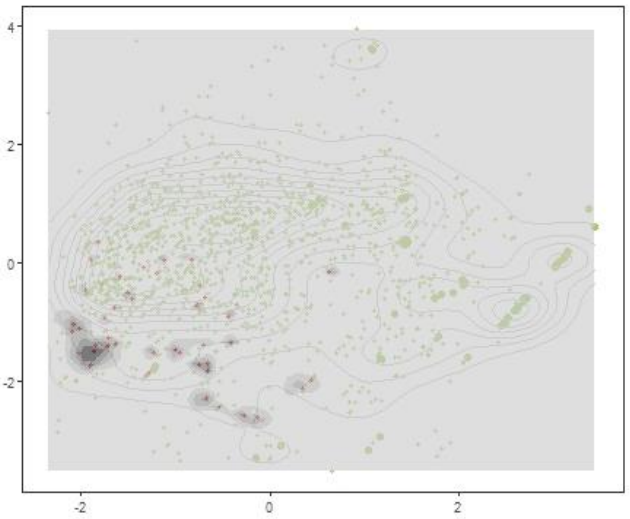
RF



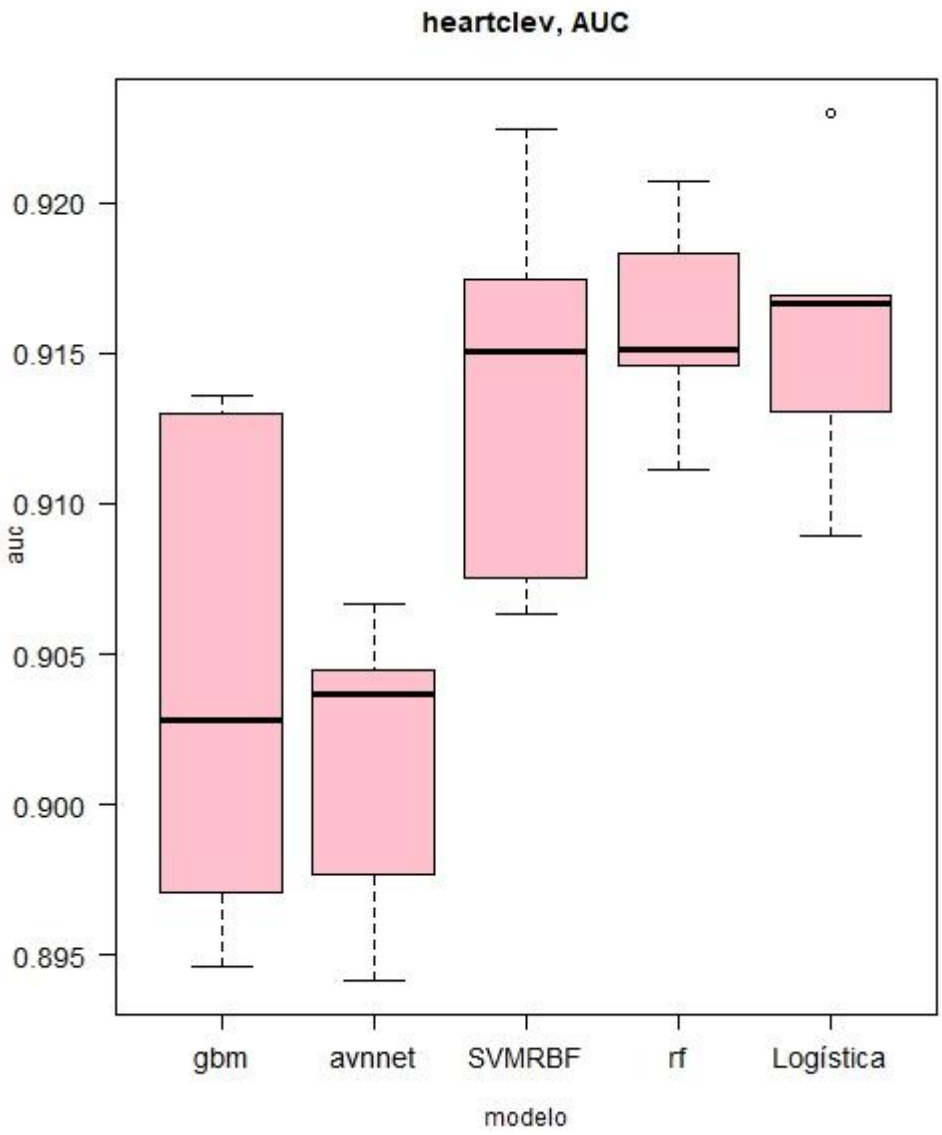
GBM



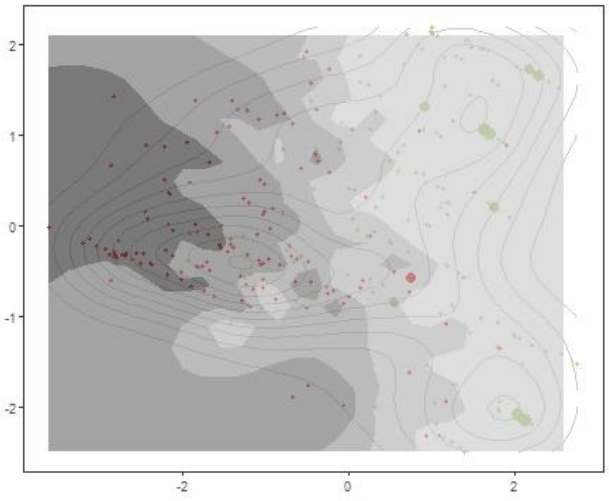
SVM



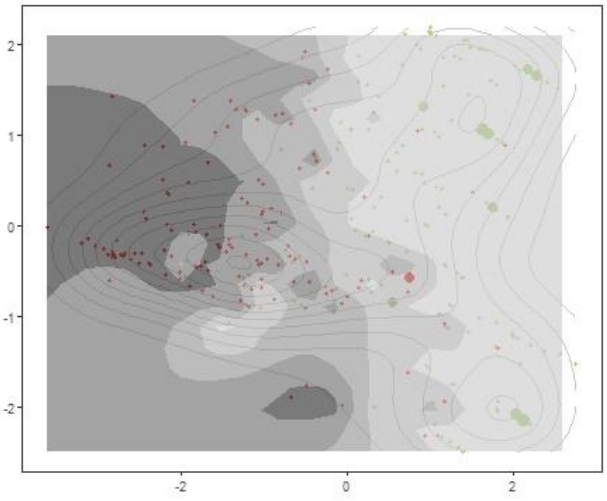
© Javier Portela 2020


heartclev, AUC
0.920
0.915
0.910
0.905
0.900
0.895
auc
gbm
avnnet
SVMRBF
rf
Logística
modelo

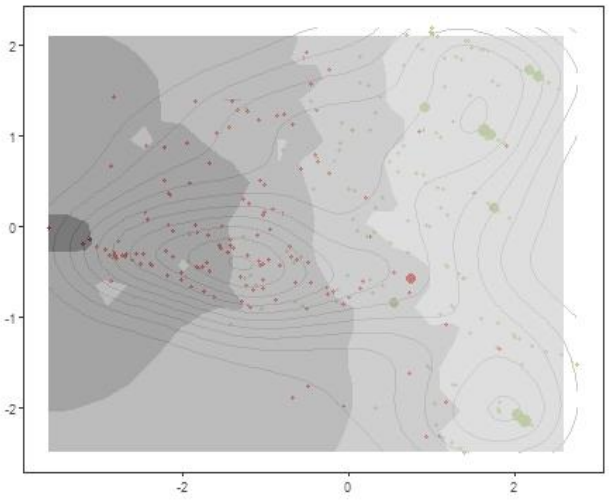
GLM



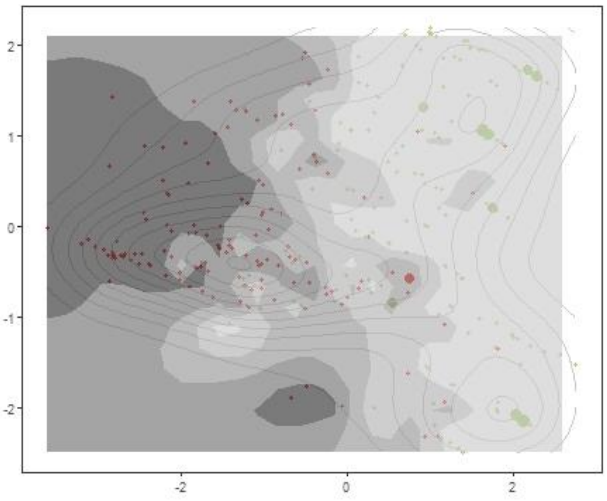
NNET



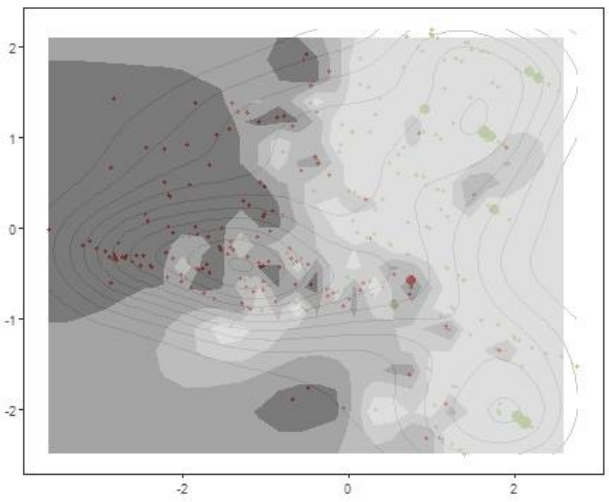
RF



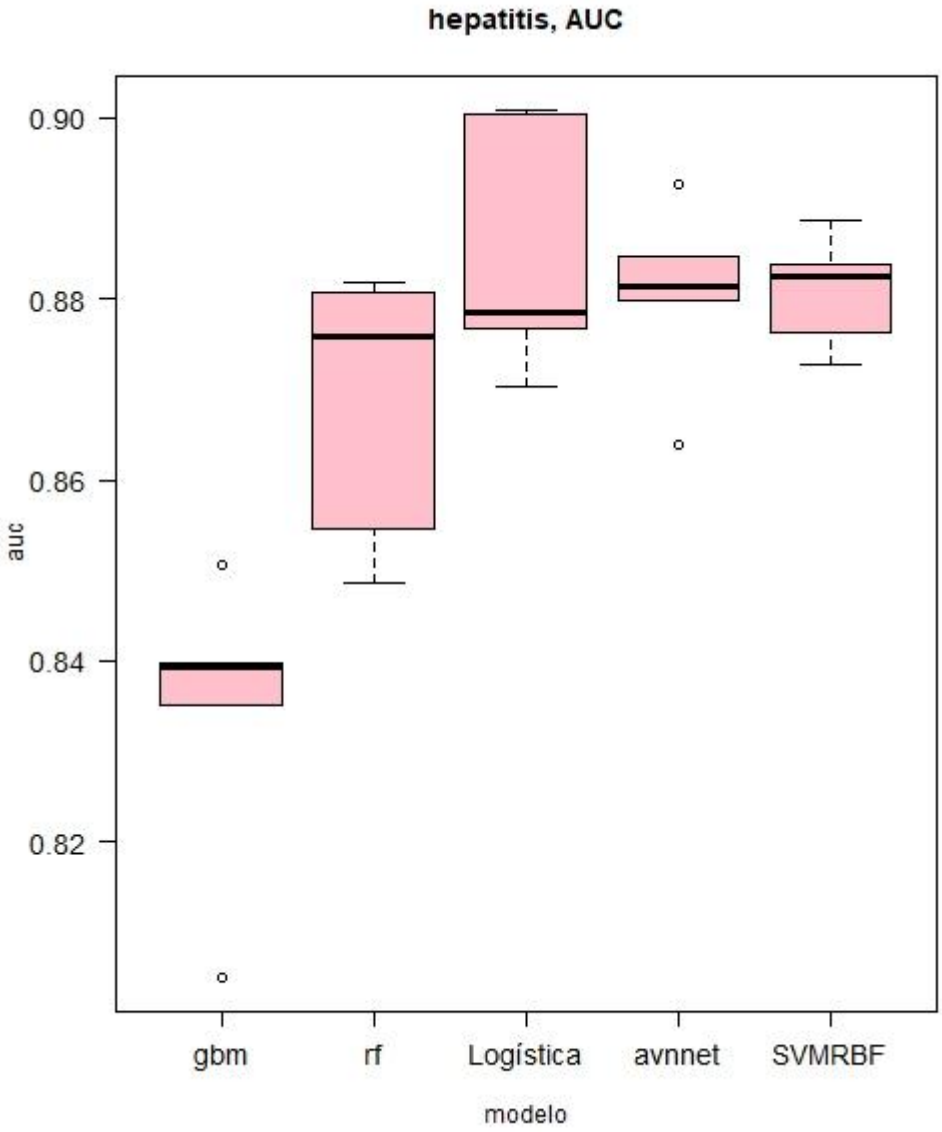
GBM



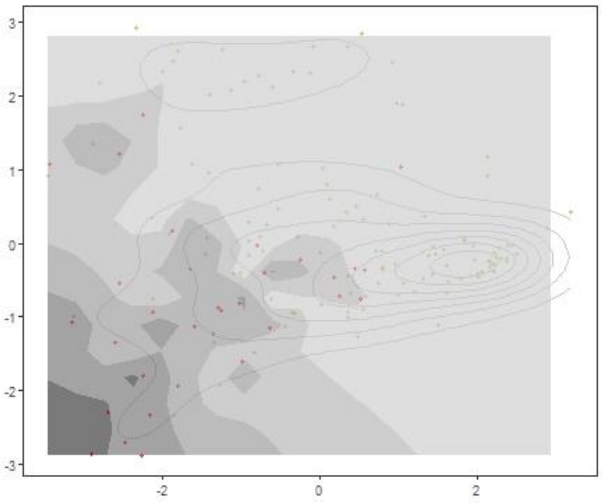
SVM



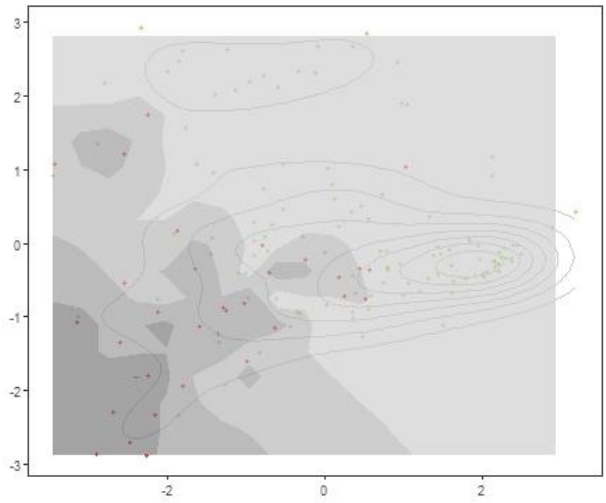
© Javier Portela 2020


hepatitis, AUC
0.90
0.88
0.86
0.84
0.82
auc
gbm
rf
Logística
avnnet
SVMRBF
modelo

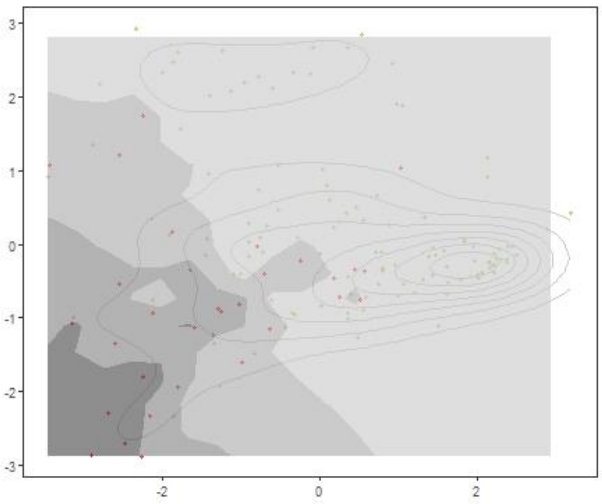
GLM



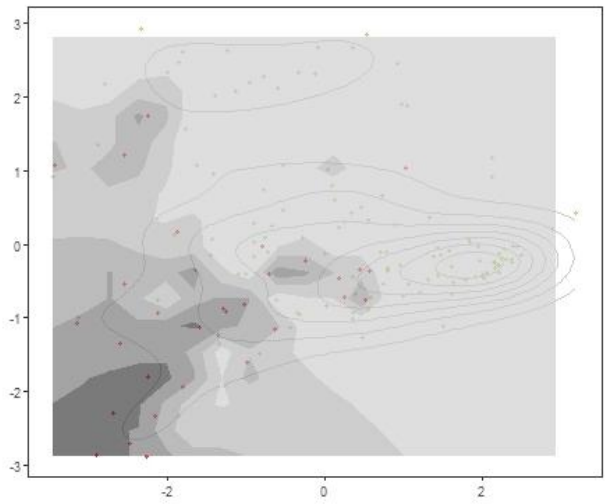
NNET



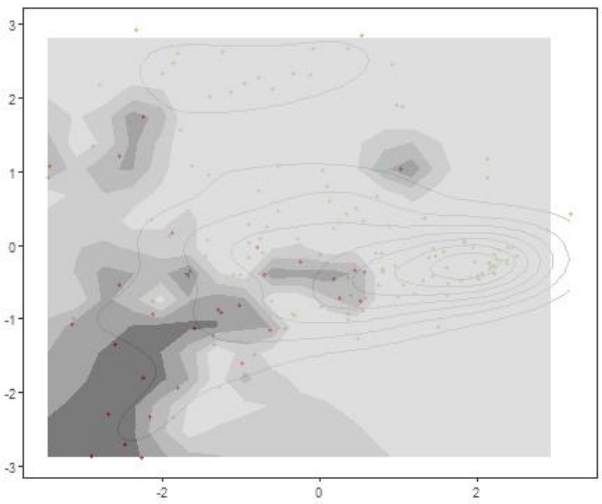
RF



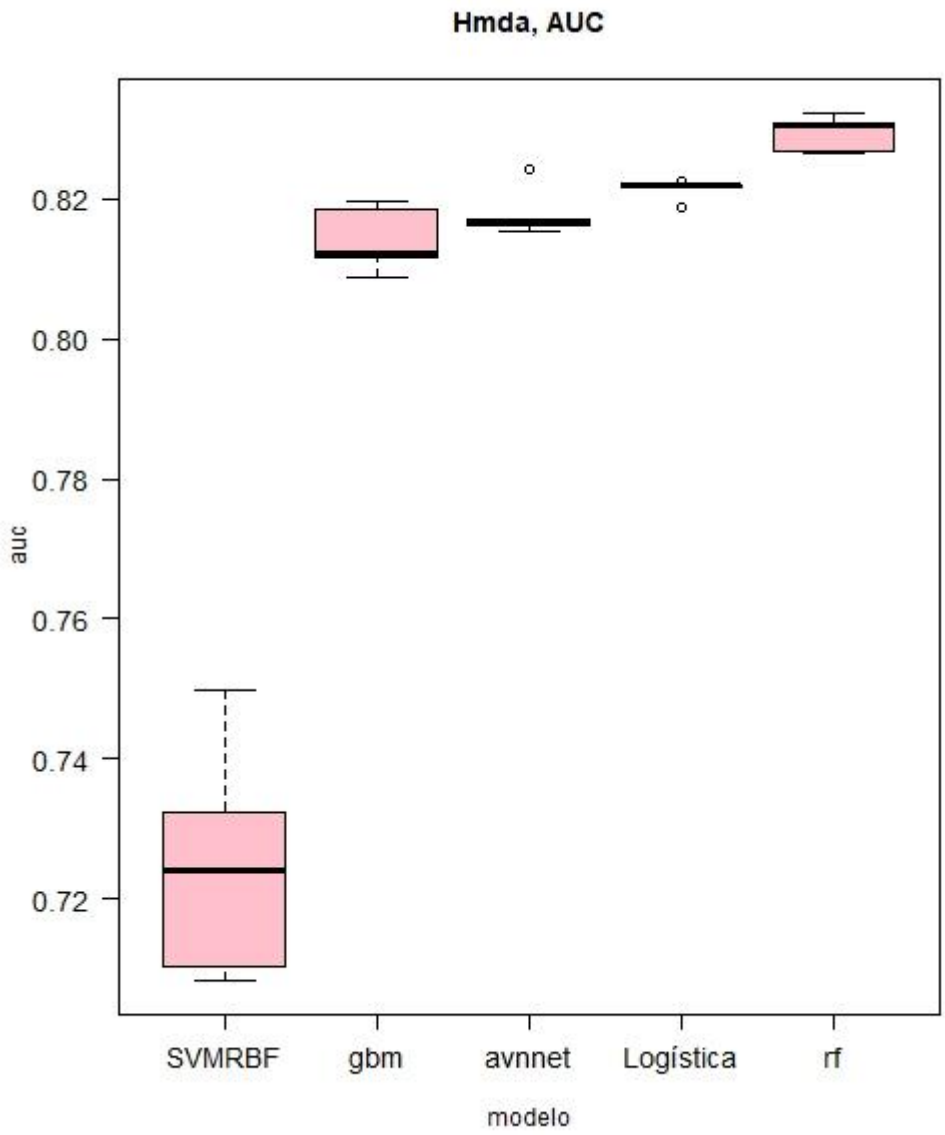
GBM



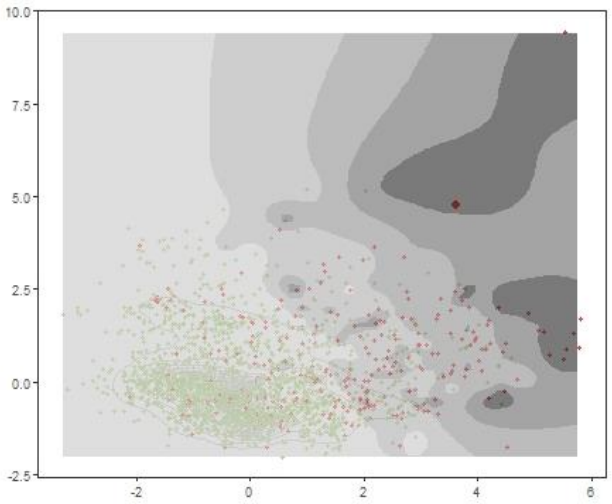
SVM



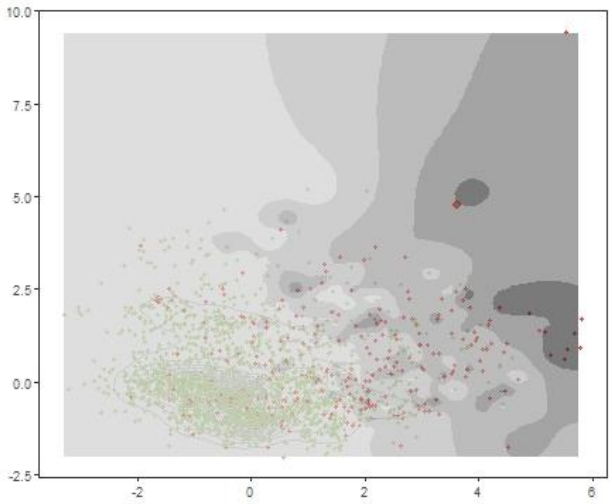
© Javier Portela 2020


Hmda, AUC
0.82
0.80
0.78
0.76
0.74
0.72
auc
SVMRBF
gbm
avnnet
Logística
rf
modelo

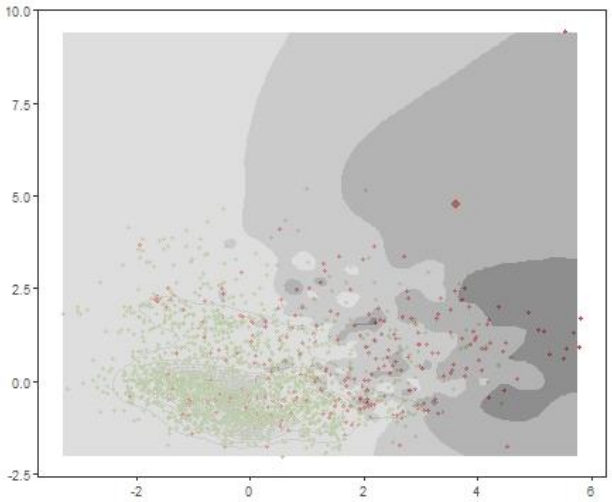
GLM



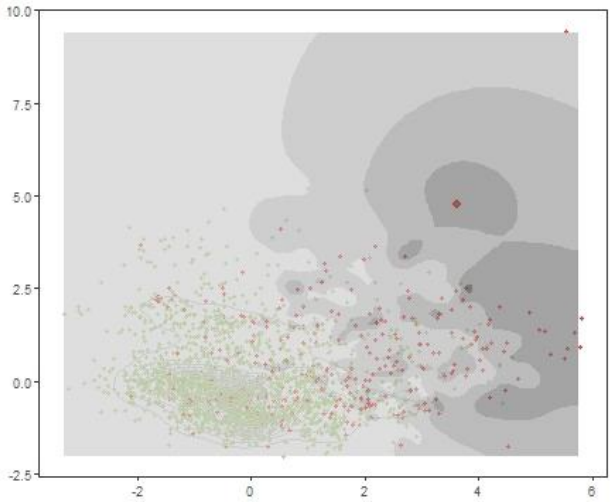
NNET



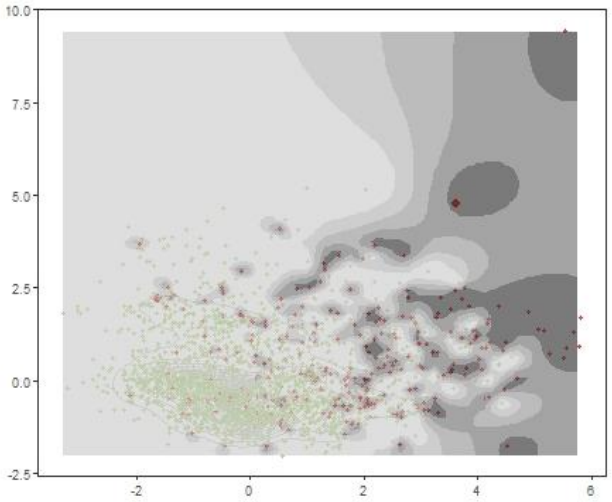
RF



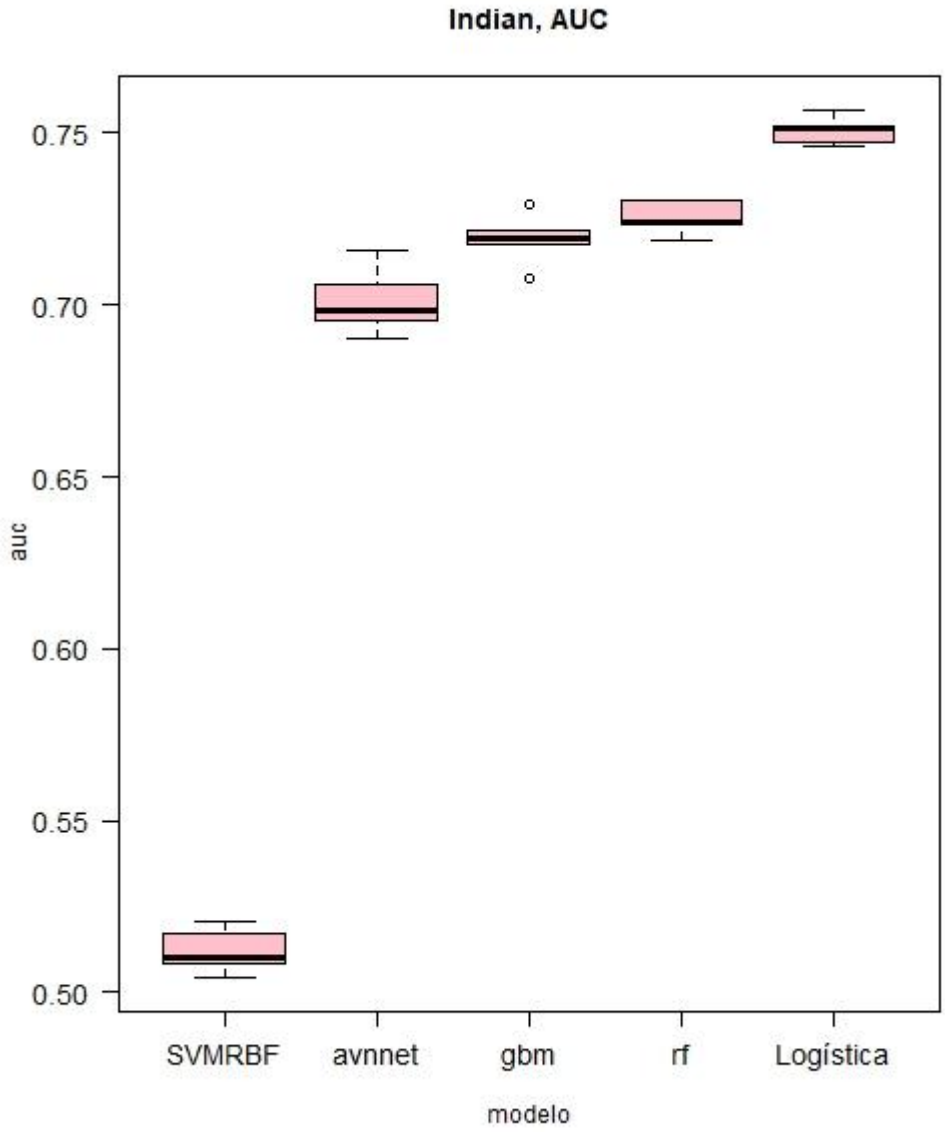
GBM

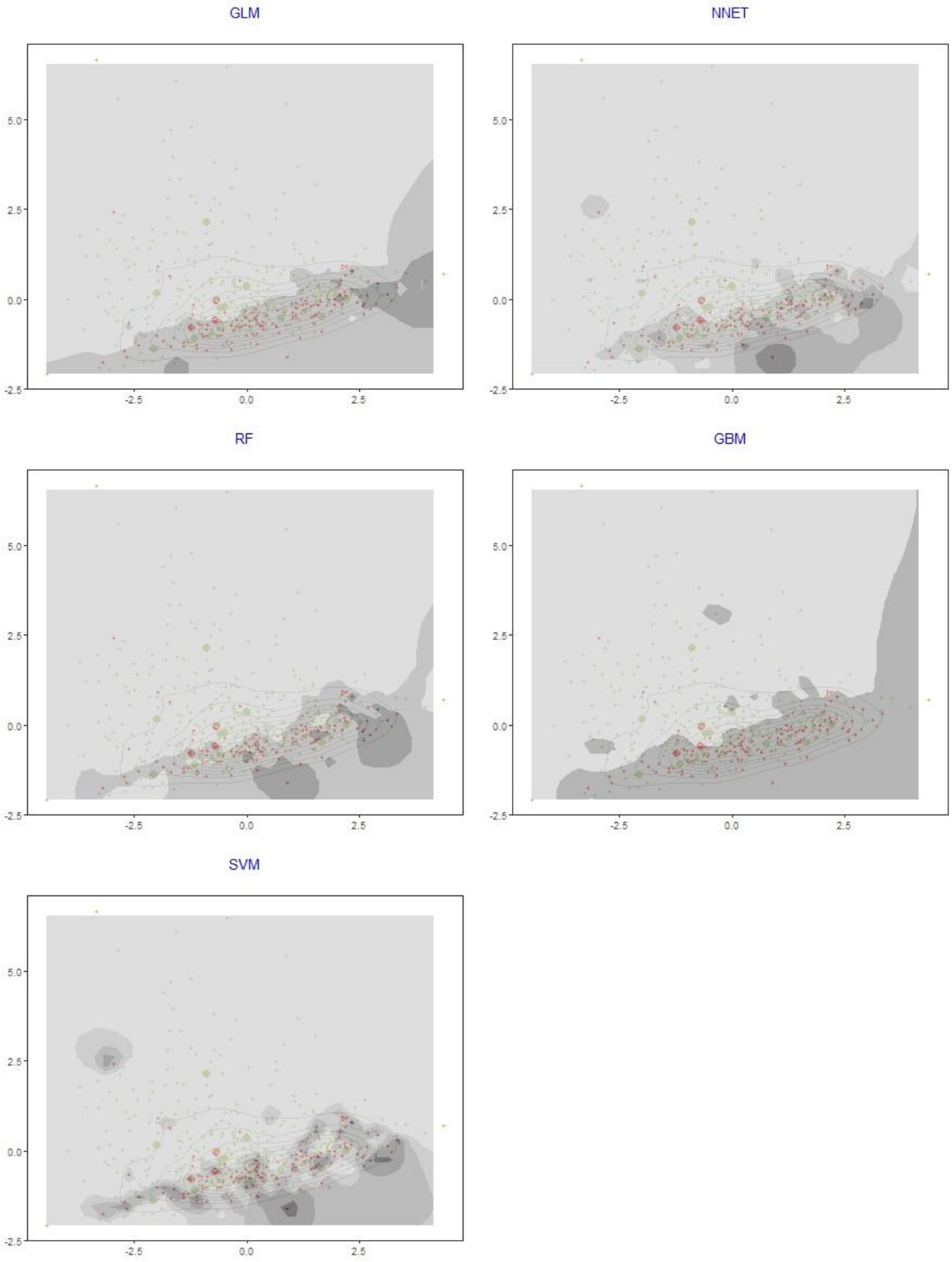


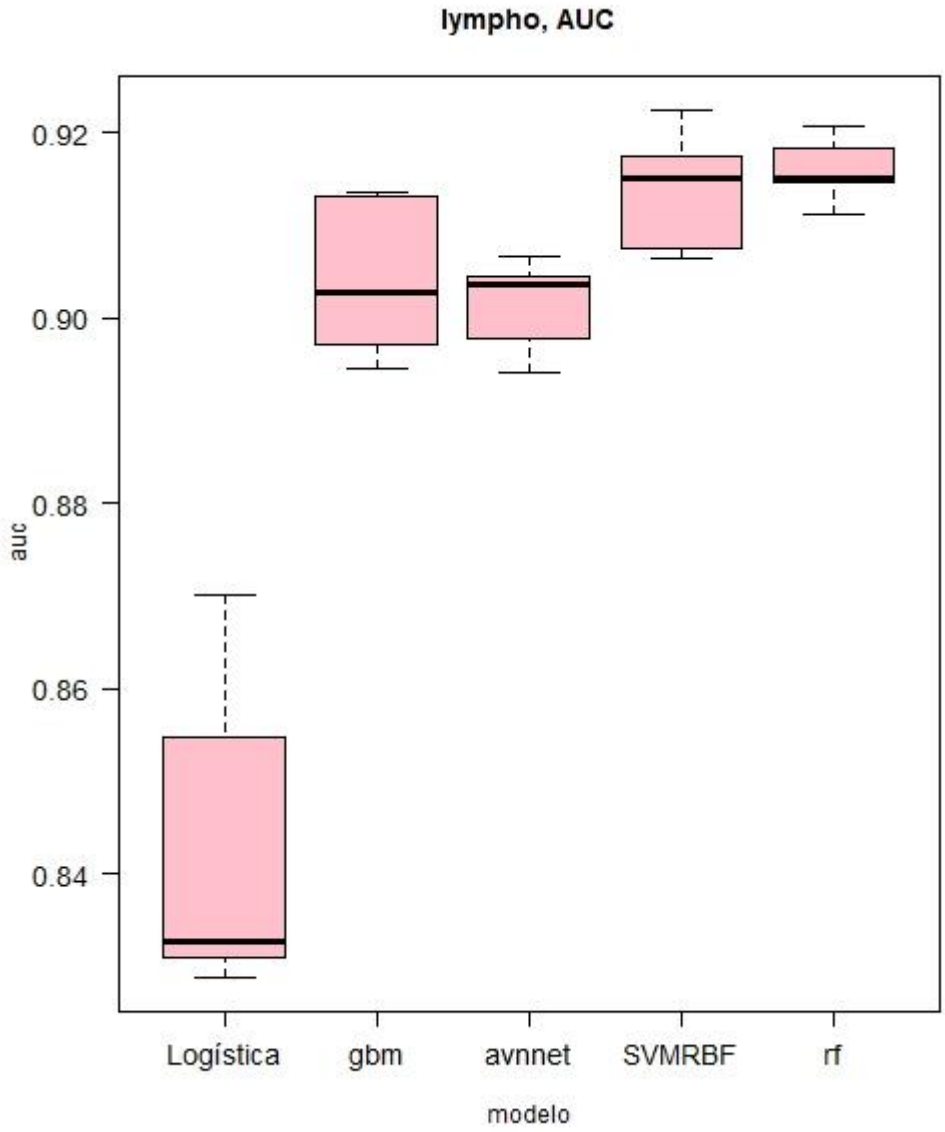
SVM



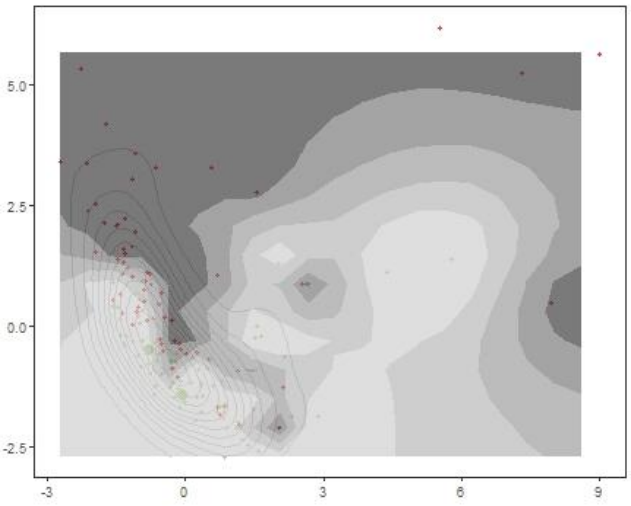
© Javier Portela 2020


Indian, AUC
0.75
0.70
0.65
0.60
0.55
0.50
auc
SVMRBF
avnnet
gbm
rf
Logística
modelo

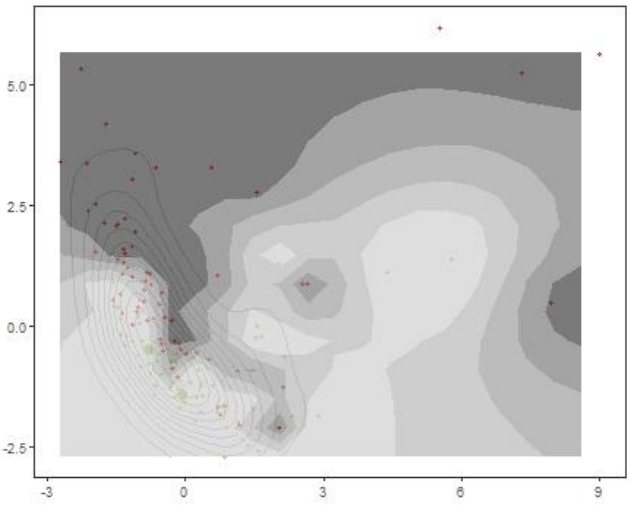

GLM
NNET
RF
GBM
SVM
5.0
2.5
0.0
-2.5
-2.5
0.0
2.5


lympho, AUC
0.92
0.90
0.88
0.86
0.84
auc
Logística
gbm
avnnet
SVMRBF
rf
modelo

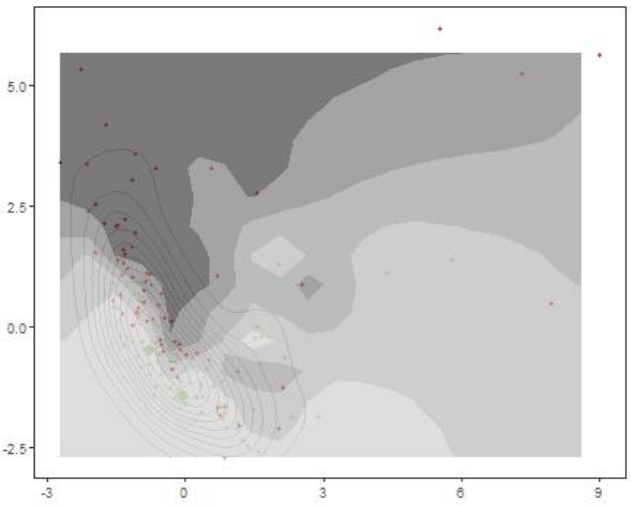
GLM



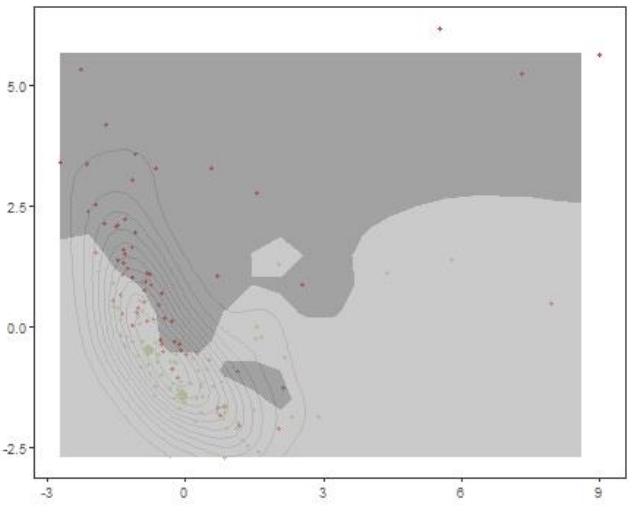
NNET



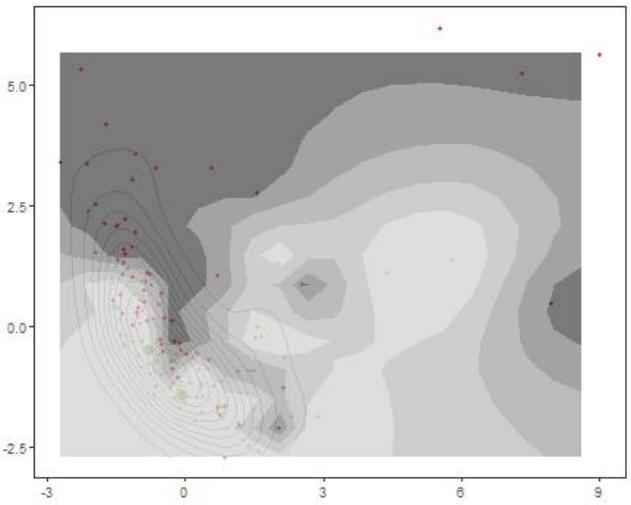
RF



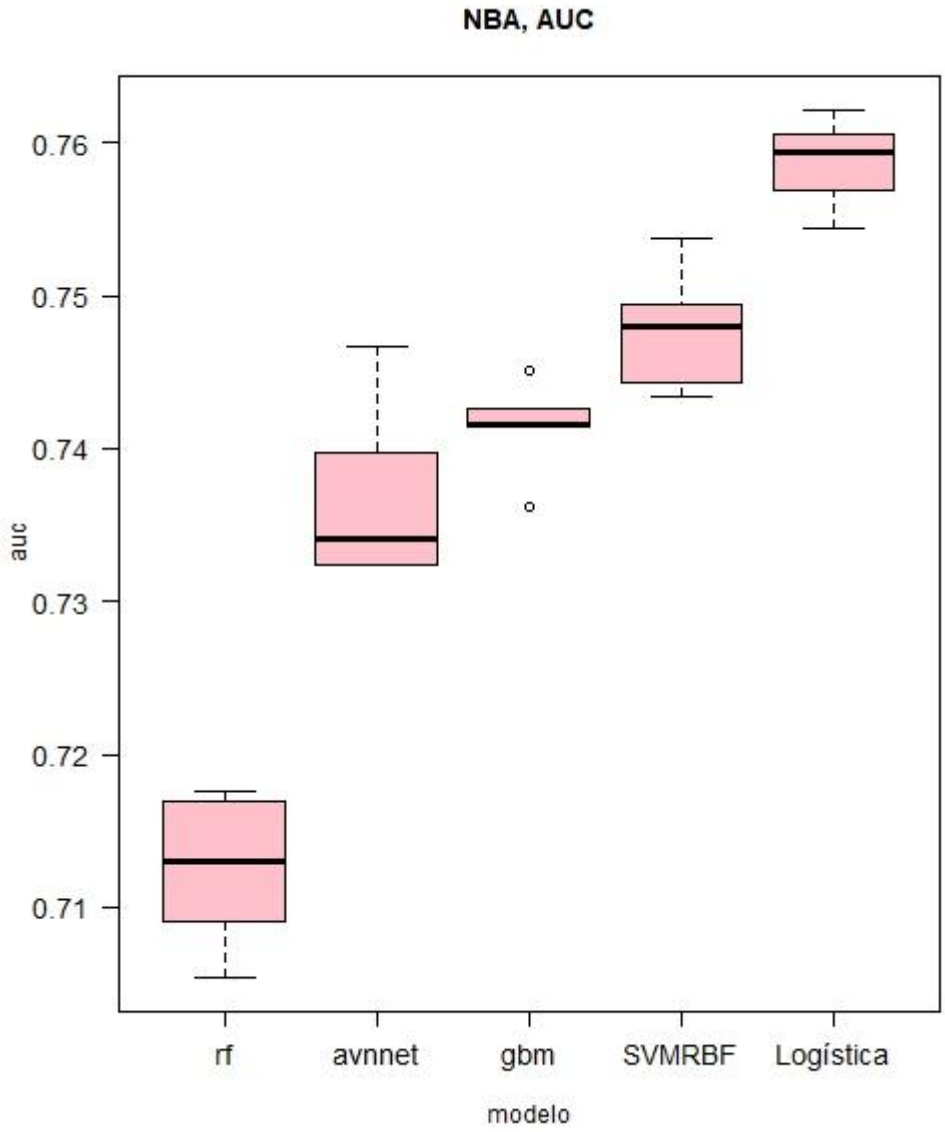
GBM



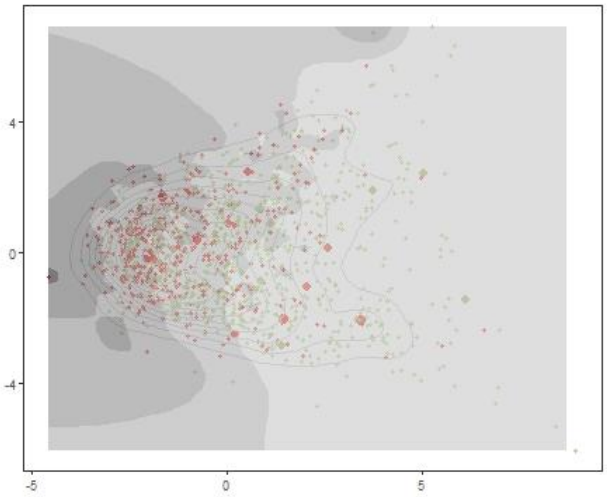
SVM



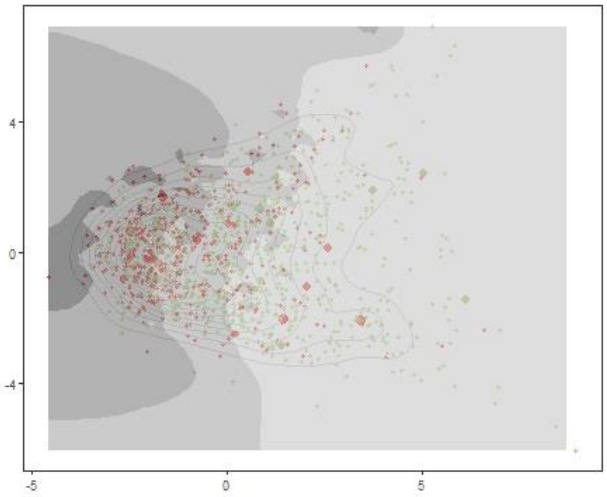
© Javier Portela 2020


NBA, AUC
0.76
0.75
0.74
0.73
0.72
0.71
auc
rf
avnnet
gbm
SVMRBF
Logística
modelo

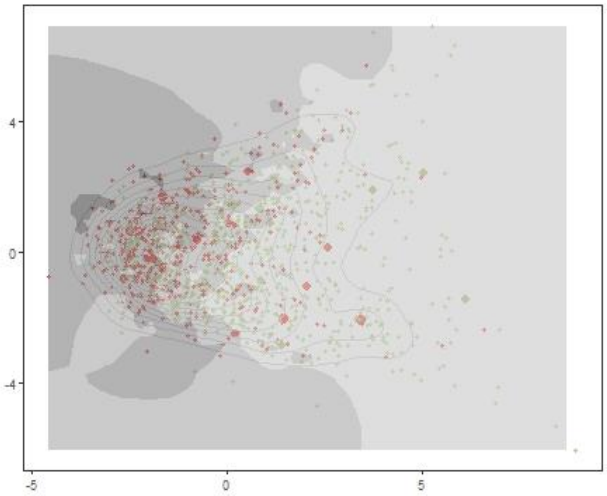
GLM



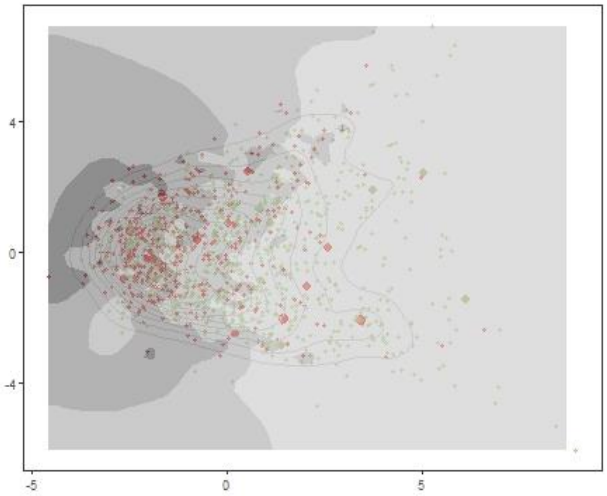
NNET



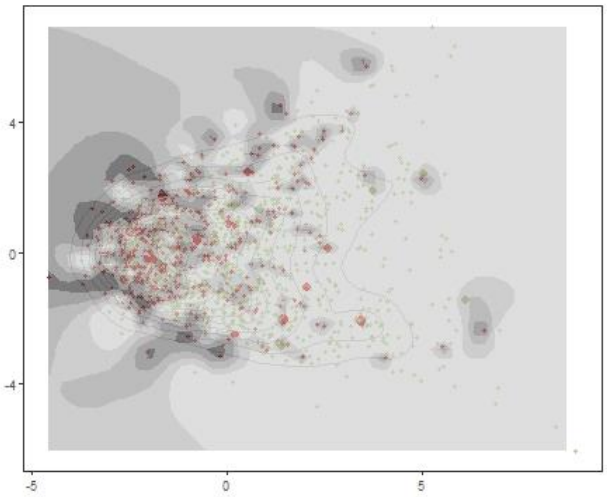
RF



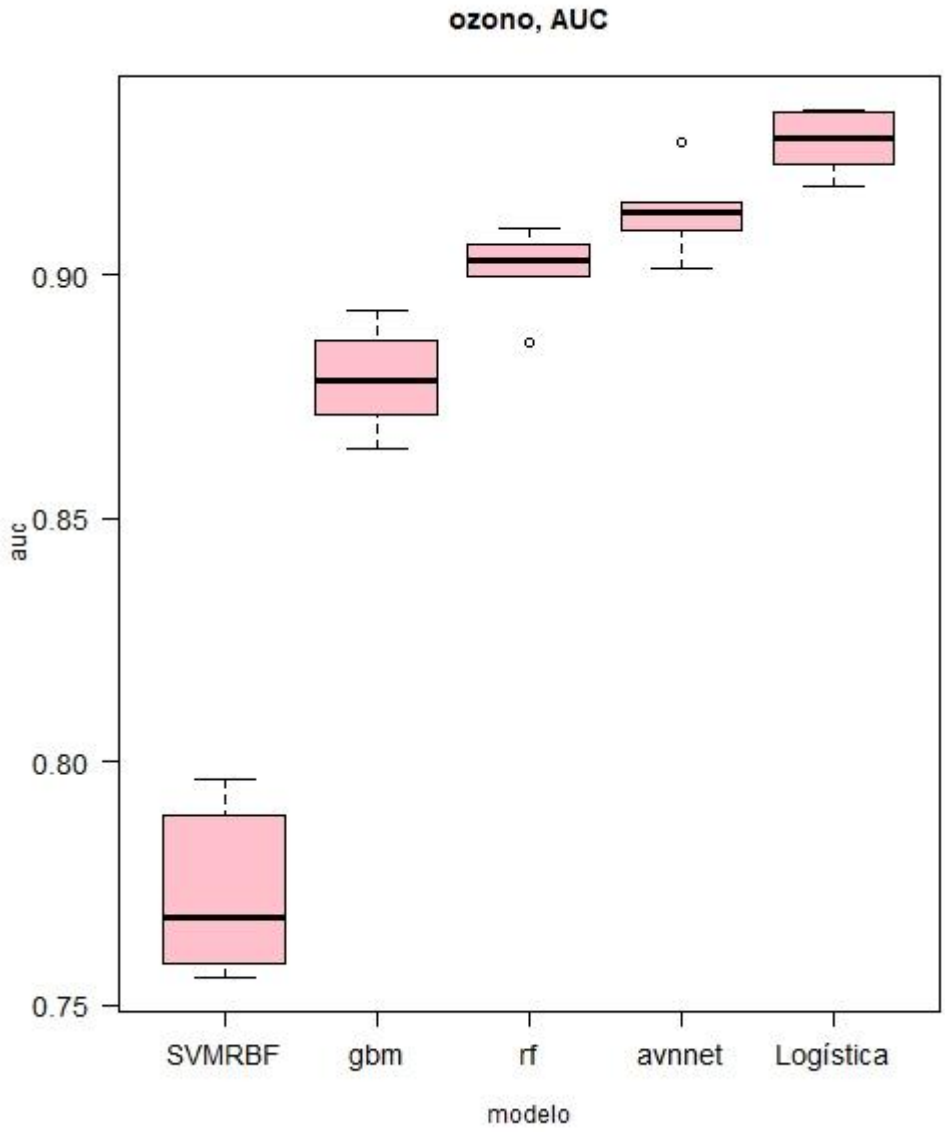
GBM



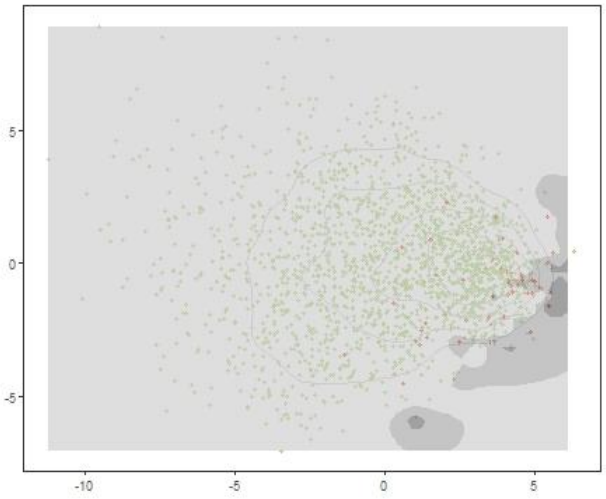
SVM



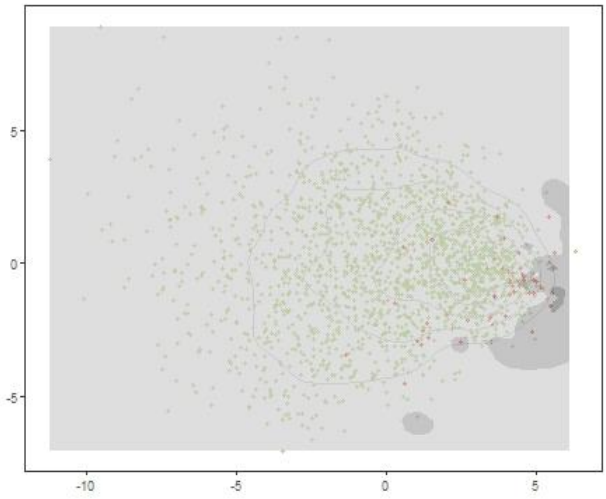
© Javier Portela 2020


ozono, AUC
auc
0.90
0.85
0.80
0.75
SVMRBF
gbm
rf
avnnet
Logística
modelo

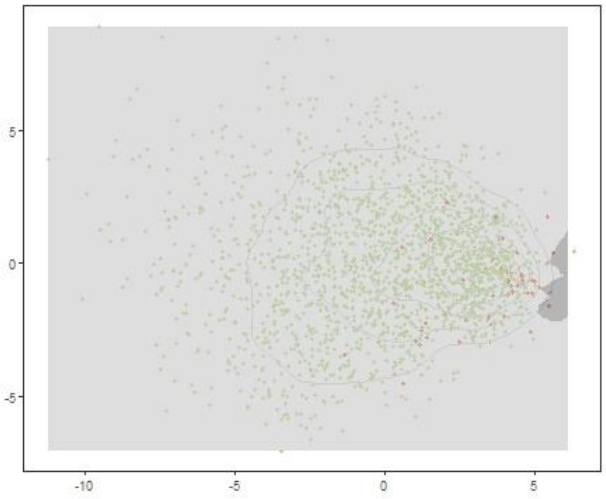
GLM



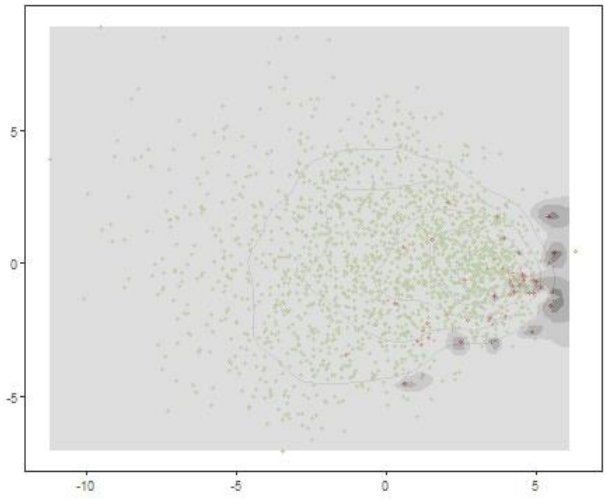
NNET



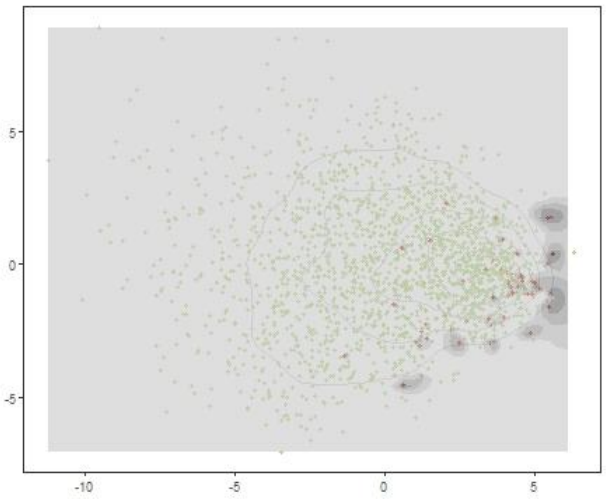
RF



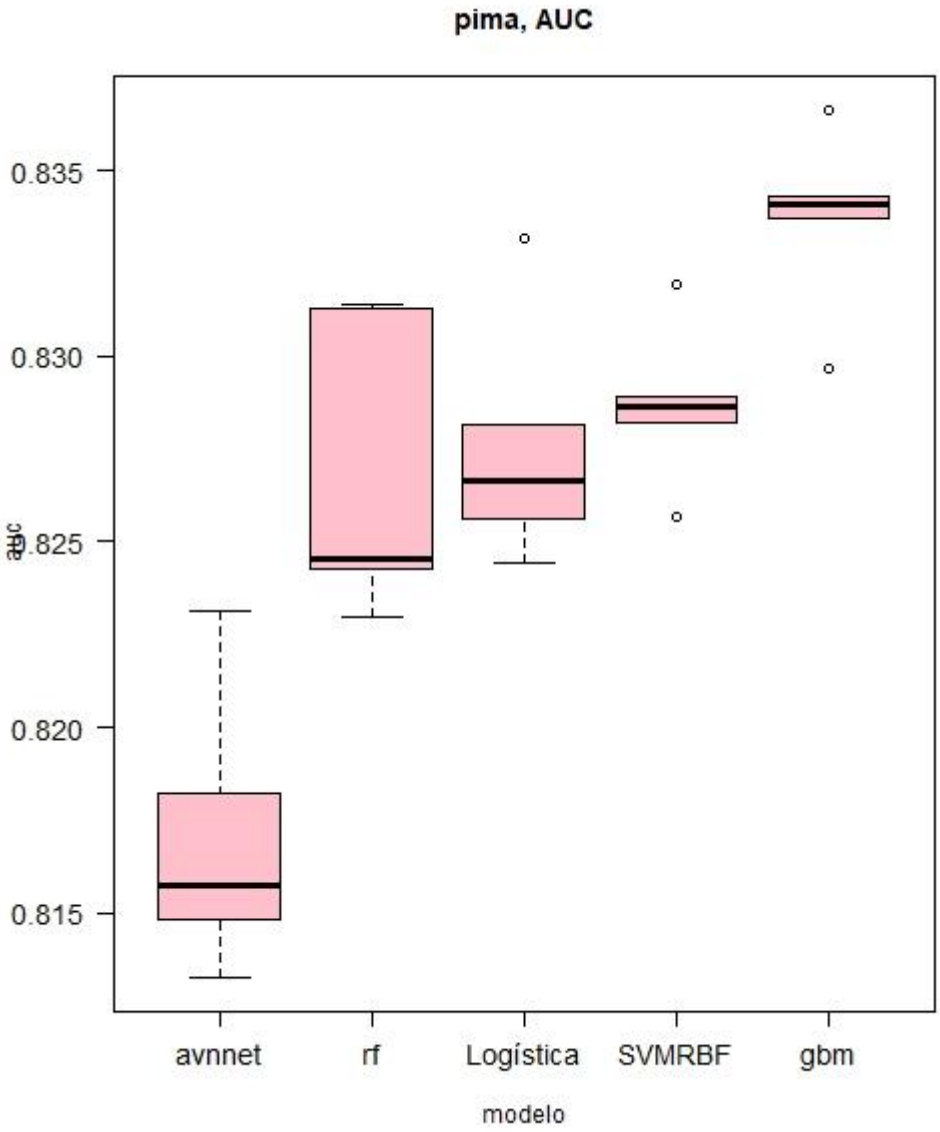
GBM



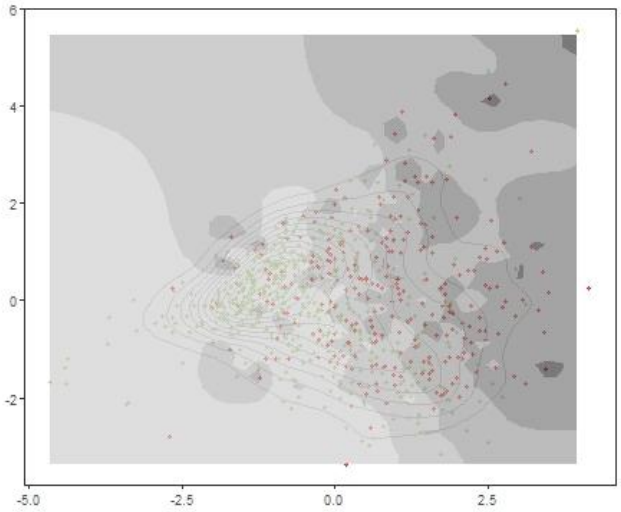
SVM



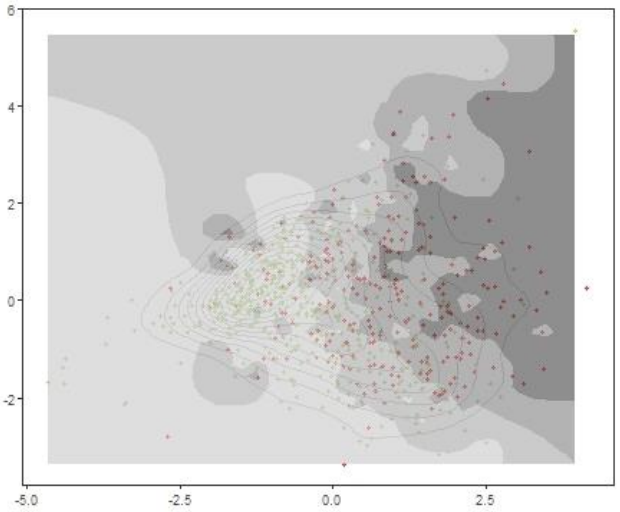
© Javier Portela 2020


pima, AUC
0.835
0.830
0.825
0.820
0.815
auc
avnnet
rf
Logística
SVMRBF
gbm
modelo

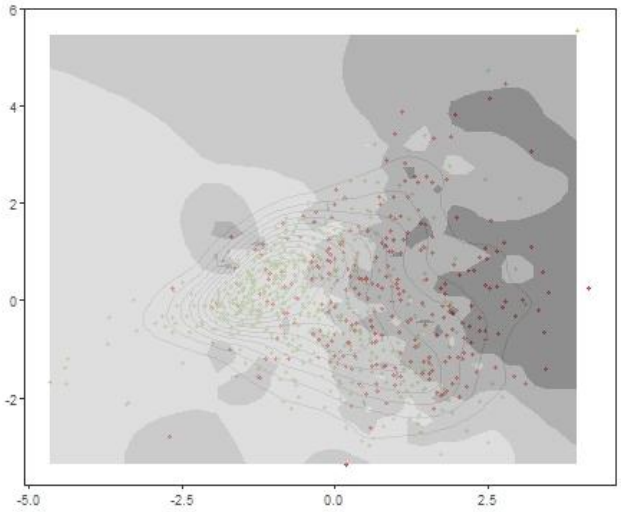
GLM



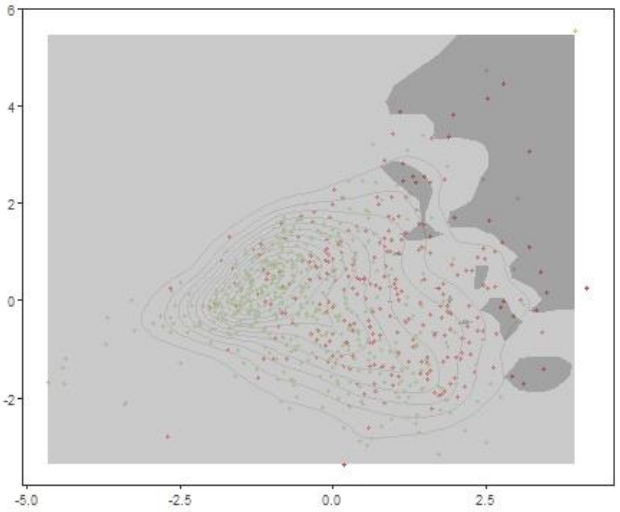
NNET



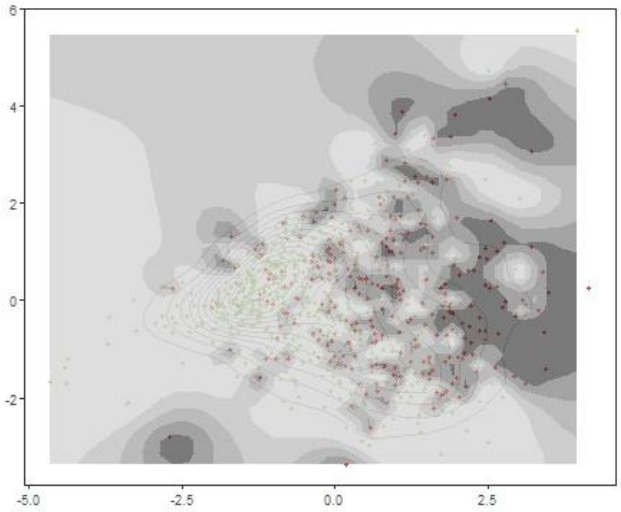
RF



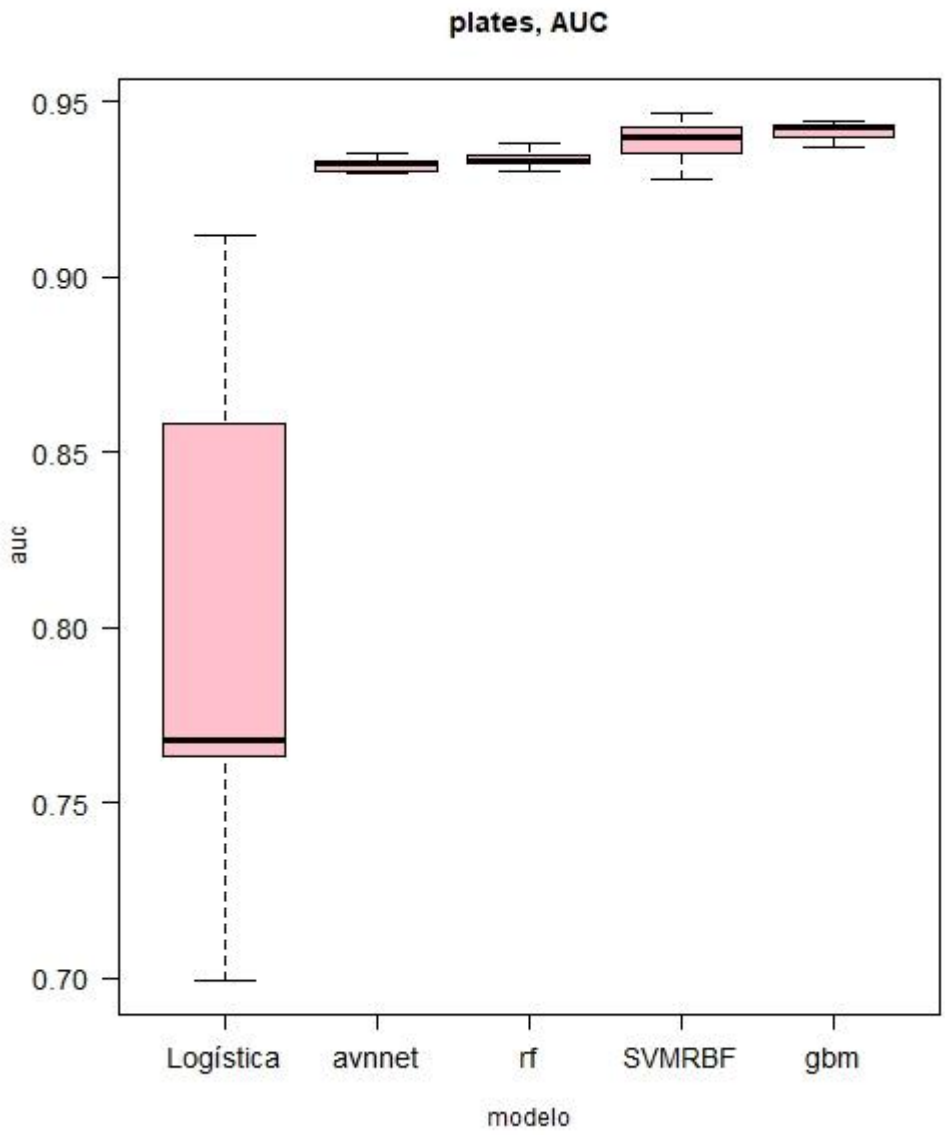
GBM

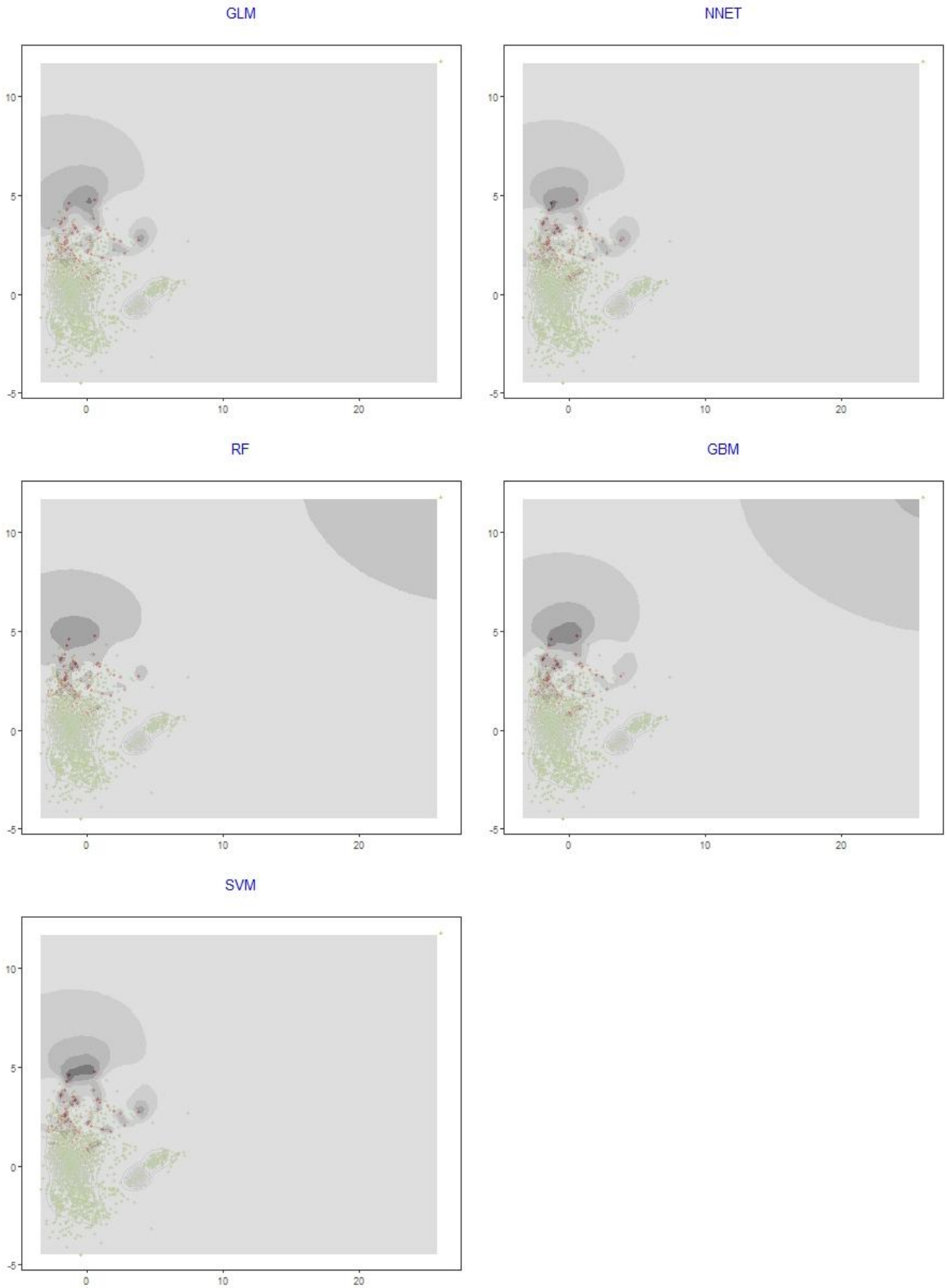


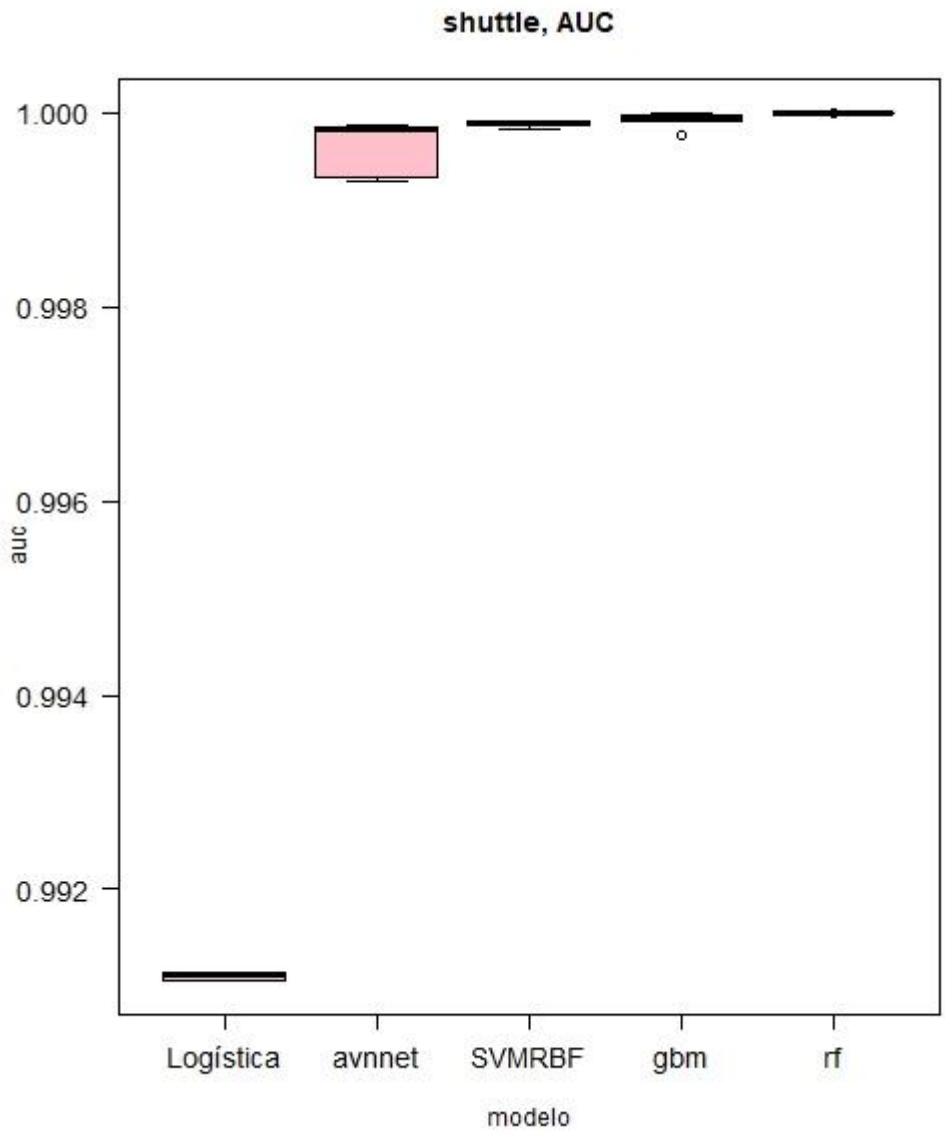
SVM



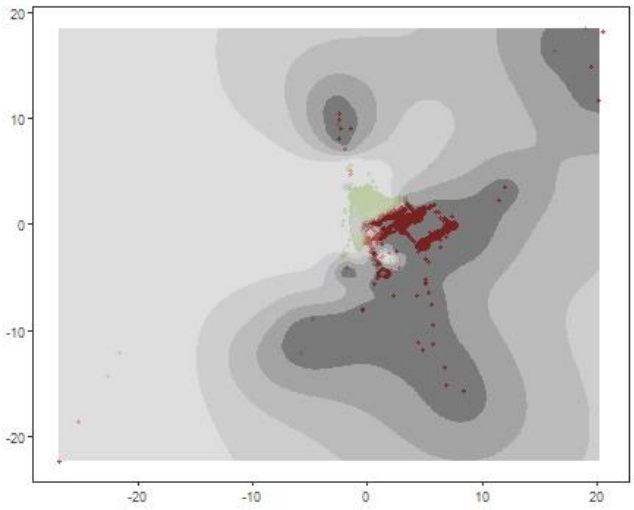
© Javier Portela 2020


plates, AUC
0.95
0.90
0.85
0.80
0.75
0.70
auc
Logística
avnnet
rf
SVMRBF
gbm
modelo

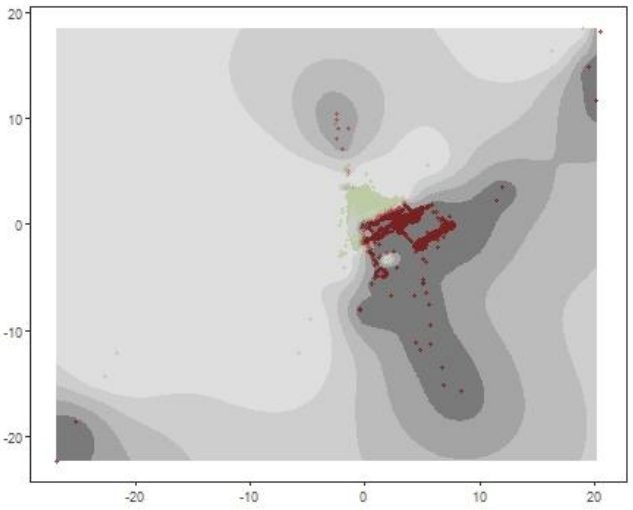

GLM
NNET
RF
GBM
SVM
10
5
0
-5
0
10
20


shuttle, AUC
1.000
0.998
0.996
0.994
0.992
auc
Logística
avnnet
SVMRBF
gbm
rf
modelo

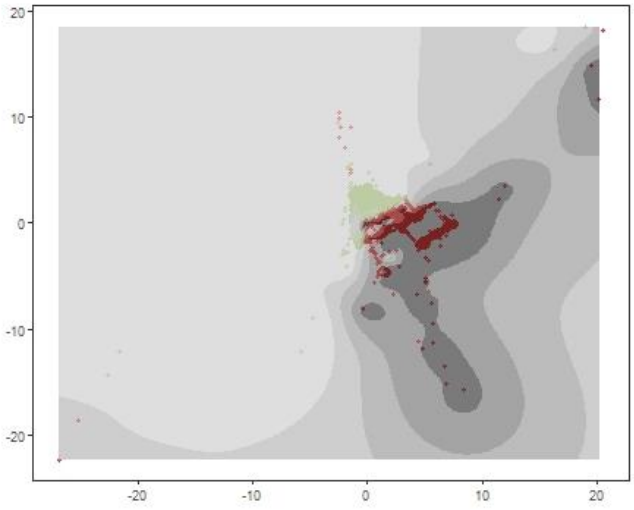
GLM



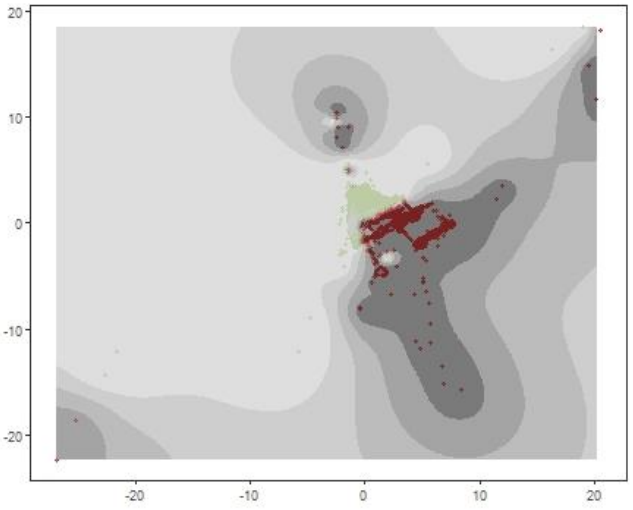
NNET



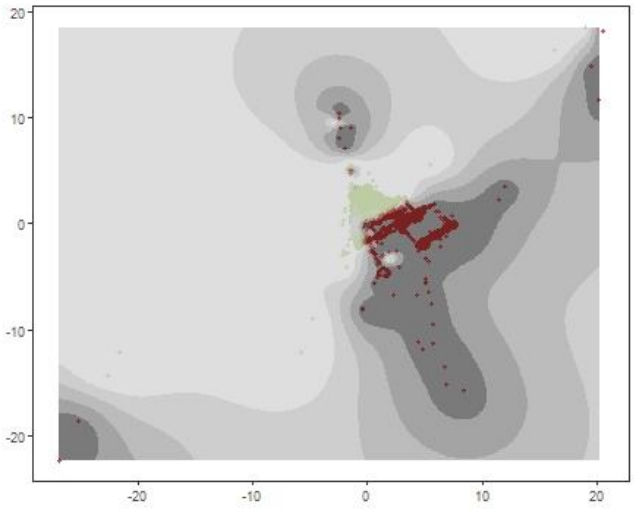
RF



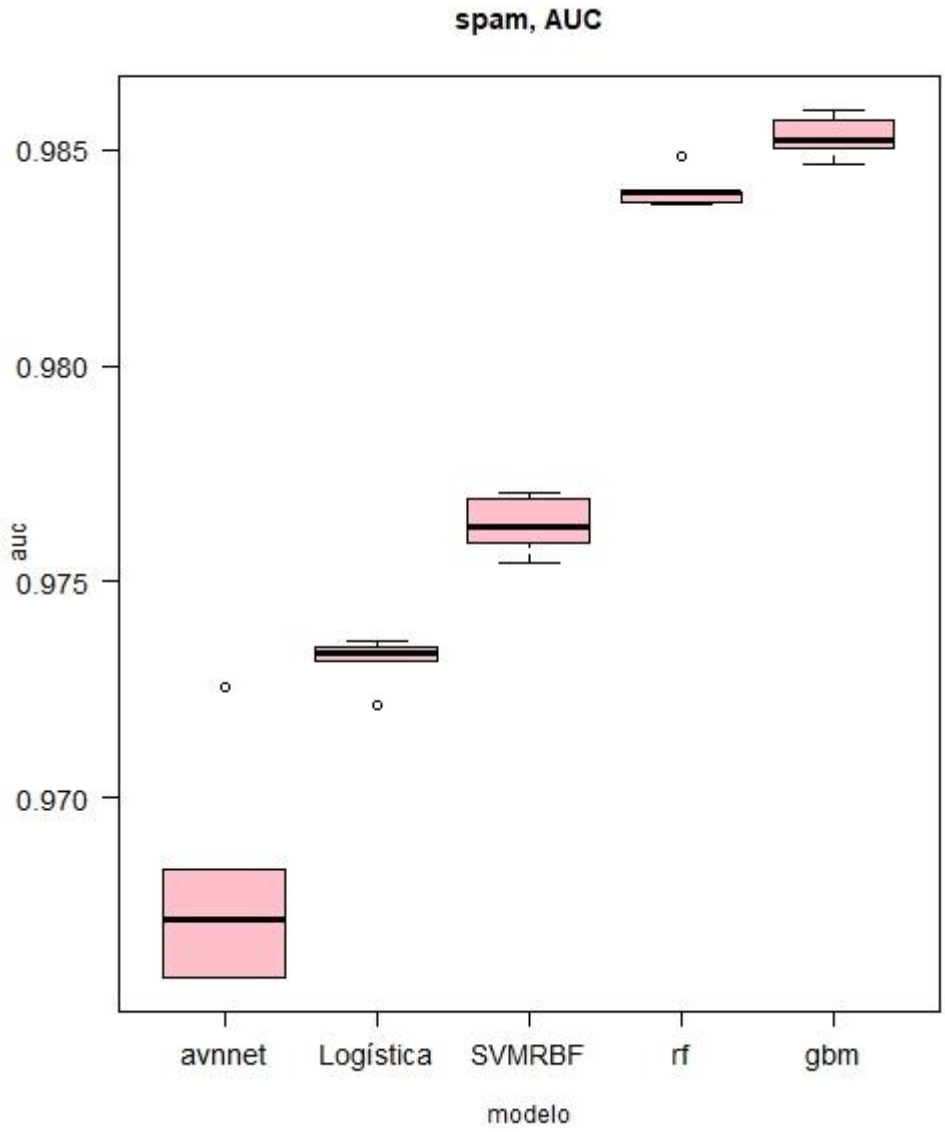
GBM



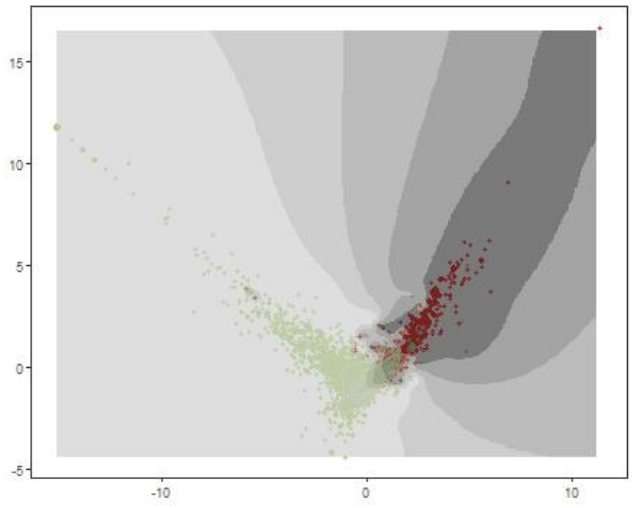
SVM



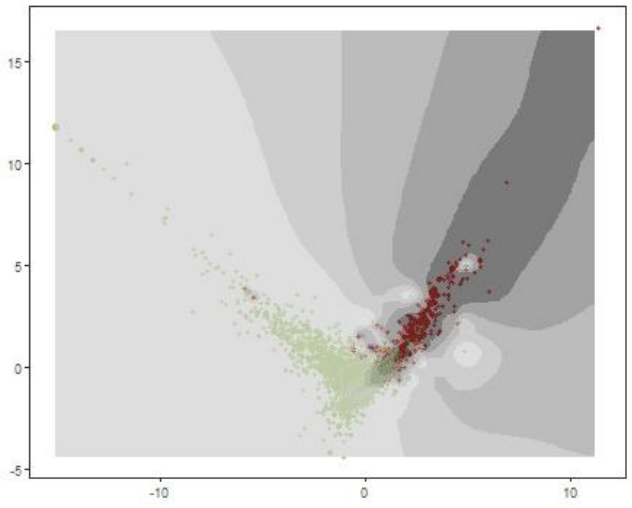
© Javier Portela 2020


spam, AUC
0.985
0.980
0.975
0.970
auc
avnnet
Logística
SVMRBF
rf
gbm
modelo

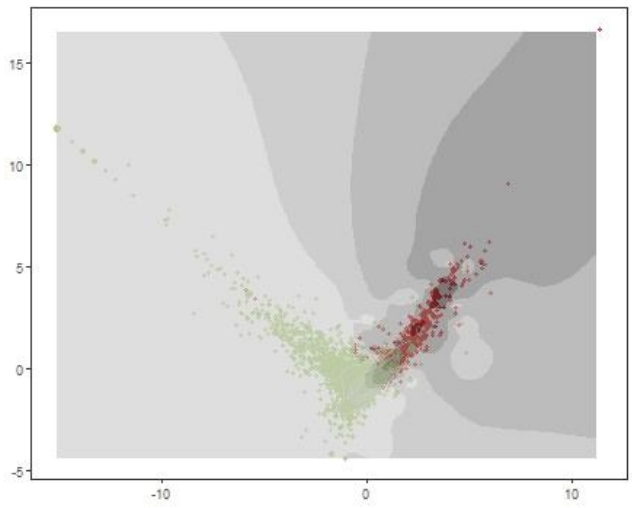
GLM



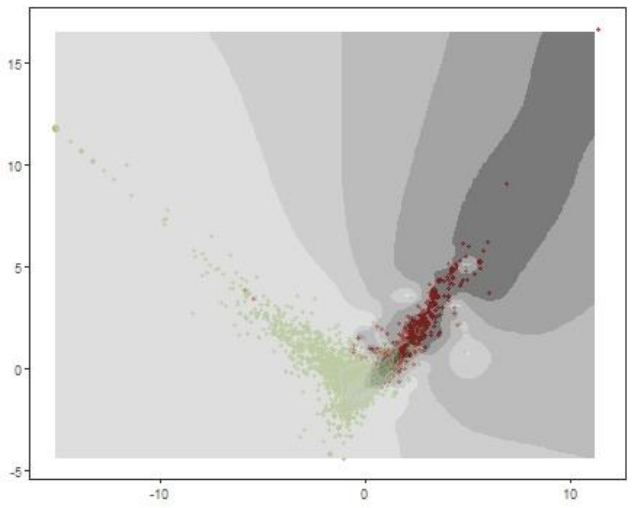
NNET



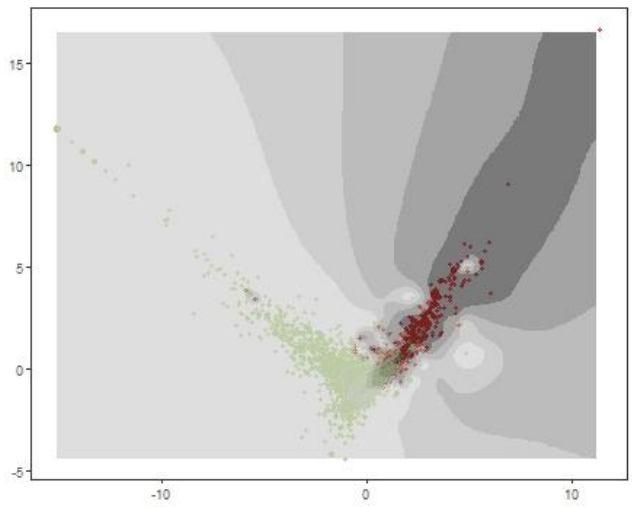
RF



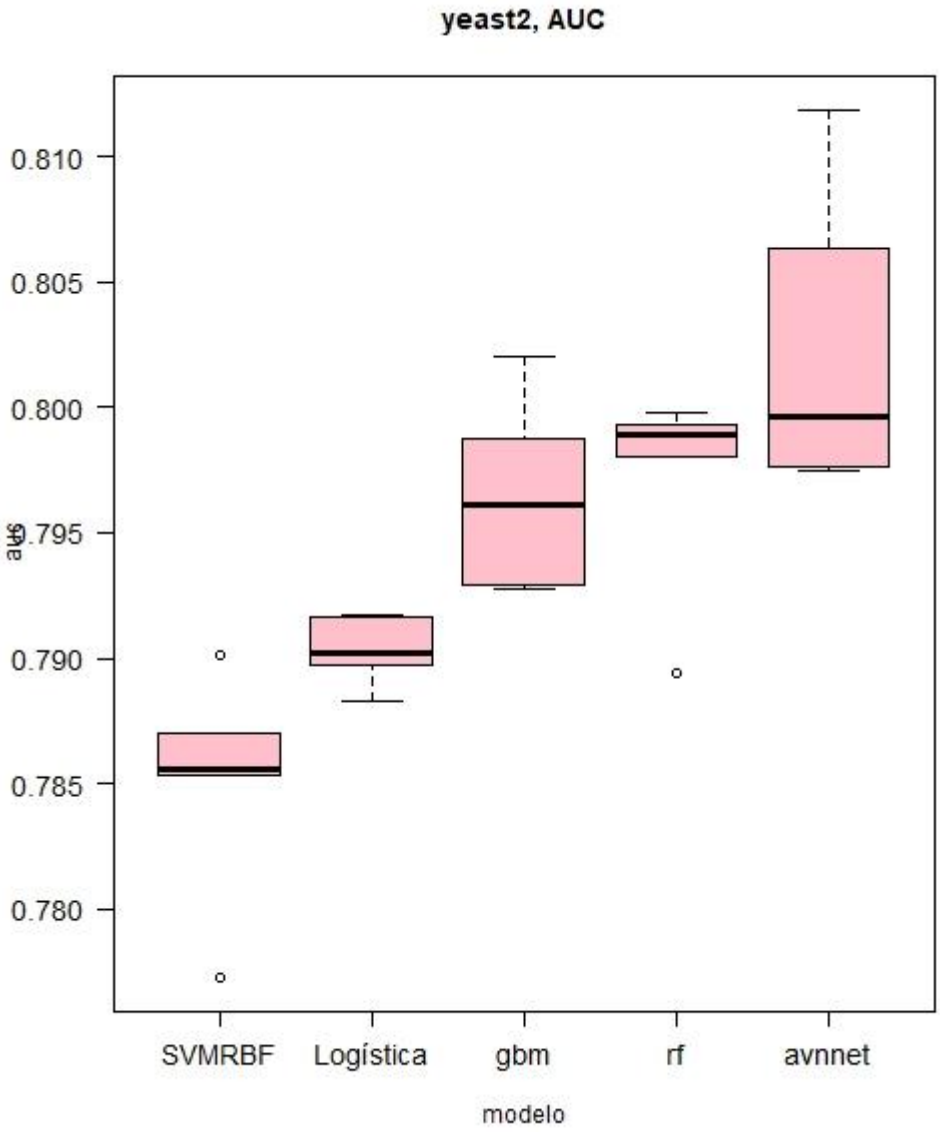
GBM



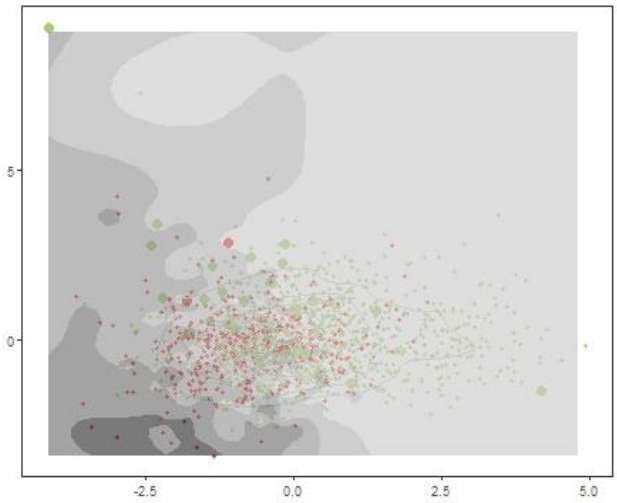
SVM



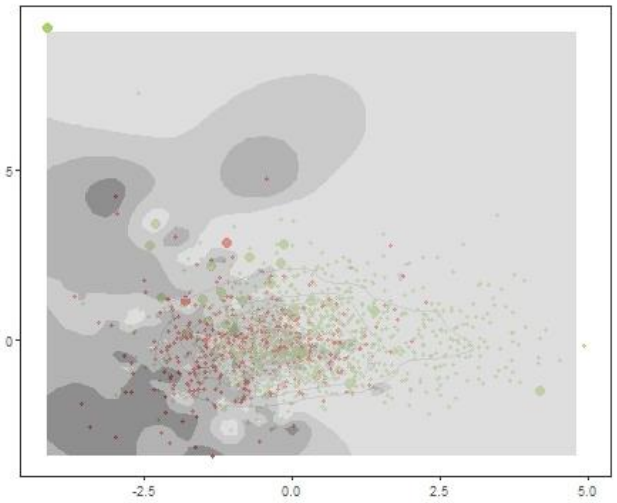
© Javier Portela 2020


yeast2, AUC
0.810
0.805
0.800
0.795
0.790
0.785
0.780
auc
SVMRBF
Logística
gbm
rf
avnnet
modelo

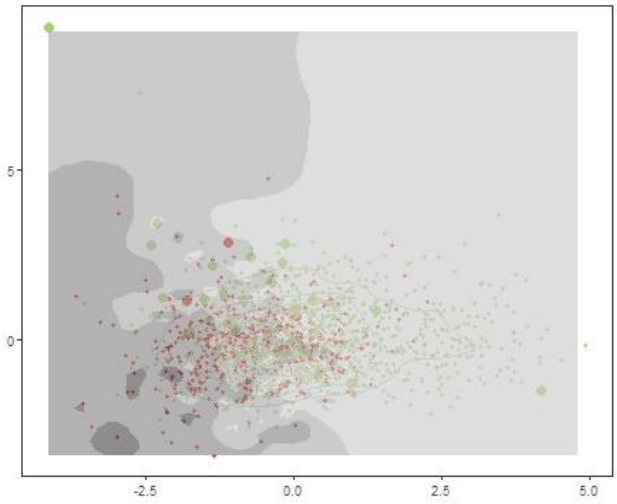
GLM



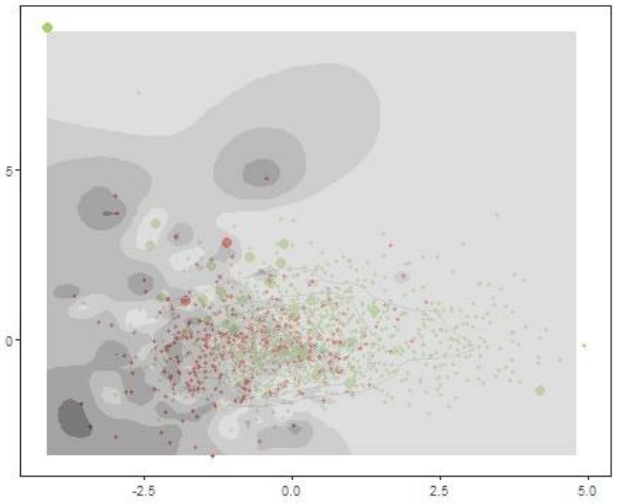
NNET



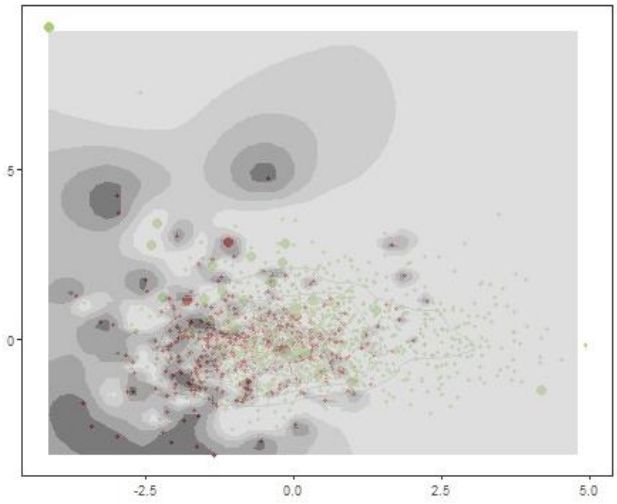
RF



GBM



SVM



© Javier Portela 2020

Anexo I: Medidas básicas de clasificación binaria

Este anexo es un recordatorio de las medidas habituales de clasificación binaria.

Normalmente el algoritmo da como salida, para cada observación, una probabilidad estimada de Yes. Si se establece un punto de corte, que por defecto suele ser 0.5, se asigna Yes a las observaciones con probabilidad predichas mayores que el punto de corte, y No a las restantes. Esta clasificación "dura" origina la llamada matriz de confusión, donde se cruzan nuestras predicciones de Yes, No con los valores reales.

Matriz de Confusión de un método de clasificación

		Observados	
		Yes	No
Predichos	Yes	15	6
	No	8	25

A partir de la matriz de confusión se pueden calcular estas medidas básicas:

VP=Verdaderos positivos: 15 : 27.7% (sobre el total)
VN=Verdaderos negativos: 25 : 46.3%
FP=Falsos positivos: 6 : 11.1%
FN=Falsos negativos: 8 : 14.8%

Sensitividad=Probabilidad de que la predicción sea 1, dado que la observación es realmente 1= capacidad de detectar positivos =VP/(VP+FN)=0.65. También se le puede llamar ***recall***.

Especificidad=Probabilidad de que la predicción sea 0, dado que la observación es realmente 0=capacidad de detectar negativos=VN/(VN+FP)=0.80

Tasa de aciertos=(VP+VN)/N=0.747

Tasa de fallos=1-(VP+VN)/N=0.253

Tasa de verdaderos positivos (también llamada ***precision***)=VP/(VP+FP)=0.71 (cuando digo positivo, qué proporción acierto)

Tasa de verdaderos negativos=VN/(VN+FN) =0.757 (cuando digo negativo, qué proporción acierto)

Otras medidas de diagnóstico conocidas son:

Si nos basamos en la clase Yes, se usan simultáneamente *precision* (tasa de verdaderos positivos) y *recall* (sensitividad) y también la **medida F**, también llamada F1 o F-M que pretende equilibrar ambas medidas:

F=2*(precision*recall)/((precisión+recall)

El índice de Youden es otra medida pretende equilibrar nuestra capacidad de detectar positivos y negativos:

Indice de Youden=especificidad+sensitividad-1

Una medida importante, porque es independiente del punto de corte, es el **área bajo la curva ROC** , llamada **AUC**. El método para obtener este área es:

a) Para cada punto de corte diferente sobre la probabilidad predicha se obtiene una matriz de confusión, que da lugar a una sensitividad y una especificidad. Normalmente se fijan como puntos de corte sucesivamente todas las probabilidades diferentes predichas en nuestros datos.

b) Se genera un gráfico (curva ROC) que plotea sensitividad en eje y, (1-especificidad) en eje x. Este gráfico es una curva situada por encima de la recta y=x.

c) Se calcula por métodos numéricos (por ejemplo trapezoidal) el área bajo esta curva, que siempre está entre 0 y 1. Cuanto mayor sea el área y más se acerque a 1, mejor ajuste general en el problema de clasificación.

Hay más medidas, pero básicamente hay que trabajar con observar la sensitividad y especificidad bajo varios puntos de corte, así como los números absolutos que se obtienen de VP,FP,VN y FN bajo diferentes puntos de corte.

Anexo II: paquete visualpred

El paquete visualpred permite representar en un plot de dos dimensiones varios aspectos relativos a un problema de clasificación binaria. En particular:

Plots de las observaciones coloreadas según las clases.

Plots de contour basado en las probabilidades predichas obtenidas de un algoritmo predictivo

Plots con observaciones extremas identificadas con etiquetas.

El procedimiento interno para la obtención de los gráficos en un problema de clasificación binaria con cierto número de variables input es el siguiente:

1) Se utiliza Análisis de Correspondencias Múltiple (MCA) o bien Análisis Factorial de Datos Mixtos (FAMD) para obtener una proyección o reducción del espacio de variables input en varios factores. En los plots se utilizan dos factores solamente, normalmente los dos factores que explican mayor varianza del espacio de variables input.

2) Los análisis obtenidos en 1 dan lugar a coordenadas para cada observación en todos los factores y permiten presentarlas en un gráfico de dos factores, coloreando cada observación según la clase de la variable output binaria a la que pertenece.

3) Se aplican algoritmos predictivos que dan lugar a una predicción de probabilidad para cada observación.

4) Se construye una rejilla de valores en el rango de los valores que toman los factores, y se da valores a esa rejilla por interpolación, tomando como referencia las predicciones de probabilidad obtenidas en 3.

5) Con los valores interpolados de probabilidades obtenidas en 4 para esa rejilla, se superpone un gráfico de contour sobre los gráficos básicos de puntos.

Para información sobre el paquete:

https://cran.r-project.org/web/packages/visualpred/index.html

Recomiendo visualizar las vignettes, que son ejemplos sobre el paquete:

https://cran.r-project.org/web/packages/visualpred/vignettes/Basic_example.html

https://cran.r-project.org/web/packages/visualpred/vignettes/Comparing.html

https://cran.r-project.org/web/packages/visualpred/vignettes/Outliers.html

https://cran.r-project.org/web/packages/visualpred/vignettes/Advanced.html

Para información sobre la sintaxis y código, ver el manual:

https://cran.r-project.org/web/packages/visualpred/visualpred.pdf

Otras cuestiones técnicas relativas al uso del paquete:

a) No permite valores missing

b) selec=1 obliga a selección stepwise de las variables; en los ejemplos artificiales anteriores por ejemplo se pone selec=0 . selec=1 puede llevar a que ninguna variable sea seleccionada y por lo tanto da error.

Anexo III: funciones y scripts R utilizadas para los ejemplos de este tema

1) Para crear y representar datos artificiales, la función toydata y la función espiral

toydata 2.0.R

2) Para representar puntos y curvas de contour, paquete *visualpred*

3) Para selección de variables simple , función seleccionar:

funcion seleccionar 2.0.R

4) Para calcular medidas de diagnostico bajo cv con diferentes puntos de corte:

funciones *resultadosglm.R*, *resultadosrf.R*, etc.

Estas funciones generan archivos de texto con las medidas en el directorio de trabajo.

5) Código con todos los ejemplos artificiales y de datos reales:

graficos tema decision.R

6) Código relativo a la sección de comportamiento comparativo de algoritmos:

comparaciones básicas.R

Para cada dataset simplemente “descomentar” la parte de código referente a ese archivo y ejecutar con el resto del script.

El código genera dos archivos jpg en el directorio de trabajo, uno relativo al diagrama de cajas y otro a los gráficos de visualpred.

Opcionalmente podréis cambiar los parámetros de tuneo, variables a introducir, etc. pero sobre todo es interesante probarlo con otros dataset diferentes.