Práctica 4: Clasificación avanzada

Asignatura: Introducción a la Minería de Datos, 4º Grado de Ingeniería Informática Escuela Politécnica Superior de Córdoba - Universidad de Córdoba 2020 - 2021

Trabajo realizado por:
-Antonio Gómez Giménez (32730338G)
i72gogia@uco.es

Índice:

Ejercicio 2 **2**

Ejercicio 3 **4**

Ejercicio 2

Para todos los apartados se han utilizado las 10 bases de datos usadas en la práctica anterior, también se han realizado un k-fold de 10 como en la práctica anterior.
Los clasificadores usados en esta práctica son árboles de decisiones, máquina de vector soporte y naive Bayes.
He cambiado k-vecinos porque a la hora de realizar AdaBoostClassifier no se puede utilizar como estimador un k-vecinos y usar la función HistGradientBoostingClassifier era complicado ya que se encuentra en un estado experimental.

Para el primer apartado se ha aplicado el método base a cada uno de los conjuntos y los datos obtenidos son los siguientes (primera columna-> base de datos, segunda columna-> arbol de decisión, tercera columna-> svm y última columna-> naive Bayes):

vote_mod.arff	93.73118279569891	96.3978494623656	93.0752688172043
soybean_mod.arff	77.22074468085107	78.50177304964538	76.63563829787235
segment-challenge.arff	94.66666666666666	96.76190476190474	79.9047619047619
wine.arff	91.08974358974359	97.56410256410257	96.85897435897435
ionosphere.arff	82.46666666666667	78.85	85.3
iris.arff	93.0	95.0	95.0
diabetes.arff	67.4283717679944	75.81761006289308	75.4367575122292
glass.arff	61.142857142857146	58.47619047619047	42.90476190476191
labor_mod.arff	86.66666666666666	100.0	86.66666666666666
segment-test_mod.arff	92.78195488721805	94.88408521303256	81.8295739348371

Respecto al segundo apartado, hemos aplicado el método de combinación de clasificadores Bagging a cada uno de los conjuntos, en este caso me he encontrado problemas ya que para ciertas bases de datos puede llegar a aparecer un bug al aplicar bugging a ciertas bases de datos y al aplicar la gaussian falla, por eso se han realizado las modificaciones necesarias en las bases de datos para solucionar este error. Los resultados obtenidos son los siguientes:

vote_mod.arff	96.06451612903227	96.72043010752688	93.3978494623656
soybean_mod.arff	82.17391304347828	80.0	82.3913043478261
segment-challenge.arff	96.38095238095237	96.66666666666666	80.0952380952381
wine.arff	90.32051282051283	95.12820512820512	95.25641025641026
ionosphere.arff	86.14999999999999	81.26666666666667	86.93333333333334
iris.arff	95.0	95.0	95.0
diabetes.arff	71.71907756813417	76.74353598881902	75.44025157232704
glass.arff	69.85714285714285	53.80952380952381	49.61904761904761
labor_mod.arff	85.83333333333334	97.5	89.16666666666666
segment-test_mod.arff	93.83145363408521	94.00375939849623	83.06390977443608

Para el tercer apartado, seleccionamos dos algoritmos de Boosting (AdaBoostClassifier y GradientBoostingClassifier) y aplique estos algoritmos a cada uno de los conjuntos, dando como resultados:

Para AdaBoostClassifier:

vote_mod.arff	94.04301075268818	96.37634408602152	69.40860215053763
soybean_mod.arff	80.21739130434784	64.83348751156336	81.73913043478262
segment-challenge.arff	94.95238095238093	88.0952380952381	71.90476190476193
wine.arff	90.32051282051282	92.82051282051282	74.55128205128206
ionosphere.arff	82.9	79.18333333333332	64.83333333333333
iris.arff	93.0	95.0	95.0
diabetes.arff	67.98392732354996	65.0314465408805	53.77358490566037
glass.arff	61.142857142857146	43.142857142857146	42.95238095238095
labor_mod.arff	79.16666666666666	95.0	79.16666666666666
segment-test_mod.arff	91.9047619047619	86.9642857142857	73.32080200501254

Para GradientBoostingClassifier:

vote_mod.arff	94.05376344086021	96.70967741935485	65.50537634408603
soybean_mod.arff	80.21739130434783	80.86956521739131	23.681776133209986
segment-challenge.arff	95.33333333333331	97.23809523809523	65.80952380952381
wine.arff	90.25641025641025	95.1923076923077	98.3974358974359
ionosphere.arff	82.45	85.73333333333333	87.73333333333333
iris.arff	93.0	93.0	94.0
diabetes.arff	68.91334730957372	75.63941299790356	75.06988120195669
glass.arff	57.095238095238095	65.28571428571429	40.238095238095234
labor_mod.arff	81.66666666666666	100.0	89.16666666666666
segment-test_mod.arff	92.24624060150374	94.3609022556391	64.69924812030075

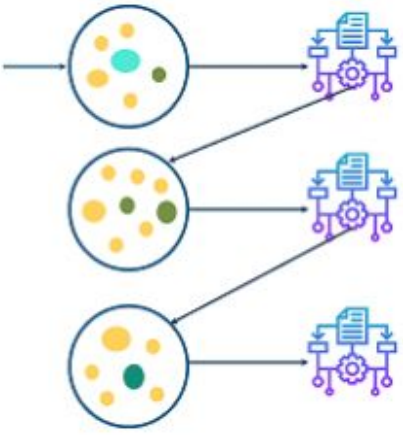
Por último, comparamos si hay diferencias significativas entre ellos usando el test de Iman-Davenport, en este caso como podemos observar si hay diferencias significativas por ello, aplicamos el procedimiento de Wilcoxon para comparar cada método de agrupación con el clasificador base:

```
Statistics_Iman-Davenport= 3.4000000000000057 ,  0.33396524909015995
Statistics_wilcoxon_vbase_y_vbagging= 1.0 ,  0.5
Statistics_wilcoxon_vbase_y_vboosting_AdaBoostClassifier= 0.0 ,  0.25
Statistics_wilcoxon_vbase_y_vboosting_GradientBoostingClassifier= 2.0 ,  0.75
```

Para la comparación del clasificador base respecto a bagging observamos que la probabilidad se encuentra por debajo del punto crítico, por ello, desechamos la hipótesis nula y por tanto, podemos considerar que hay diferencias significativas, con el caso base y AdaBoostClassifier ocurre algo similar, sin embargo, y es algo curioso, al realizar el procedimiento de Wilcoxon entre el clasificador base y GradientBoostingClassifier no se descarta la hipótesis nula, y por tanto no hay diferencias notables. Esto puede ser, debido a que he realizado la media de entre todas las bases de datos para poder comparar esas medias de, árboles de decisión, svm y naive Bayes, de entre todas las pruebas ya sean el clasificador base, bagging y boosting.

Ejercicio 3

Como podemos observar, al realizar bagging frente a linear, se realiza una mejora notable cuando se usa el clasificador de árboles de decisión, cuando se ha realizado con cualquiera de los otros, prácticamente no hay variación y si hay es insignificante.
Respecto a la comparación entre boosting de GradientBoostingClassifier con linear, no hay casi diferencia pero cuando comparamos AdaBoostClassifier frente a linear si hay diferencia y es a peor, esto se debe a que se esta realizando sobreentrenamiento, esto se debe a como funciona boost.



Respecto a un número n, que se haya especificado de iteraciones, se utiliza el conjunto de datos y se extrae un modelo, y ese mismo modelo se vuelve a aplicar al conjunto de datos y así respecto a n.
De tal forma que si el número n es demasiado elevado se puede llegar a sobre entrenamiento, por tanto, al validar con el conjunto de test los ccr van a disminuir como este es el caso.

Cabe destacar, que bagging mejora el clasificador de árboles de decisión porque estos son muy estáticos, es decir, dependen en gran medida del conjunto de datos con los que se hayan entrenado, entonces, como con bagging modificamos parte de ese conjunto de datos permite variar el conjunto de datos mejorando el clasificador, si esto se hace en exceso se podría producir infra-entrenamiento.