UNIVERSIDAD
D
CÓRDOBA

INTRODUCCIÓN A LA MINERÍA DE DATOS

Práctica 2: Clasificación y evaluación de modelos

4º Curso - Grado en Ingeniería Informática

UCO - Escuela Politécnica Superior de Córdoba

Manuel Casas Castro - 31875931R

i72cascm@uco.es

1.1

Obtenga 10 datasets:

Datasets seleccionados para esta práctica:

-iris

-diabetes

-glass

-wine

-ecoli

-contact_lens

-winequality_red

-winequality_white

-zoo

-breastTisue

1.2.

Seleccione al menos 3 clasificadores dentro de los disponibles en Scikit. Se recomienda elegir tres de entre los siguientes: árboles de decisión, k vecinos más cercanos, máquinas de vectores soporte y clasificador Naïve de Bayes. No use combinaciones (ensembles) de modelos que serán objeto de una práctica posterior.

Los clasificadores utilizados durante los siguientes experimentos serán árboles de decisión, k vecinos más cercanos y SVM.

1.3.

Para cada uno de los problemas seleccionados realice las siguientes tareas:

3.1.

Seleccione como método para obtener el error la validación cruzada de 10 particiones o el método hold out.

Seleccione dos métricas de las estudiadas en teoría.

3.2.

Entrene cada clasificador seleccionado y anote el valor de las métricas estudiadas. Fije los hiperparámetros de forma razonable

En la siguiente imagen se mostrará el código utilizado para obtener la función k Fold con 10 particiones totales, aplicándose a cada uno de los datasets para posteriormente calcular la media de los mismos.

```
aux = cross_val_score(tree, x, y, cv=10)
dec_tree.append(aux.mean())
aux = cross_val_score(vec, x, y, cv=10)
knn.append(aux.mean())
aux = cross_val_score(svm, x, y, cv=10)
aux_svm.append(aux.mean())
```

Los resultados de la métrica empleada (media) utilizada para los 10 siguientes:

```
Media svm: 0.6027167087300354
Media k vec: 0.6243148258834239
Media tree : 0.7320489931927752
```

Podemos deducir que hemos obtenido buenos resultados observando dichas métricas.

1.4.

Use el test de Wilcoxon de comparación de dos algoritmos sobre N problemas y aplíquelo a dos de los algoritmos anteriores. Obtenga el rango de Friedman para cada clasificador y configuración y represente gráficamente los resultados. Aplique el test de Iman-Davenport sobre los tres clasificadores.

Aplicaremos el test de Wilcoxon en los clasificadores árboles de decisión y k vecinos sobre los datasets:

```
BreastTissue.csv
Wilcoxon T value: 0.0
Wilcoxon p value: 1.0
No existe diferencia: p>=0.05
Friedman T value: 2.0
Friedman p value: 0.36787944117144245
No existe diferencia: p>=0.05
Davenport:  1.0
No existen diferencias significativas.
```

```
contact-lens.csv
Wilcoxon T value: 0.0
Wilcoxon p value: 0.5
No existe diferencia: p>=0.05
Friedman T value: 3.7142857142857144
Friedman p value: 0.15611804531597104
No existe diferencia: p>=0.05
Davenport:  2.0526315789473686
No existen diferencias significativas.
```

```
ecoli.csv
Wilcoxon T value: 1.0
Wilcoxon p value: 0.5
No existe diferencia: p>=0.05
Friedman T value: 3.8181818181818183
Friedman p value: 0.14821506633752016
No existe diferencia: p>=0.05
Davenport:  2.123595505617978
No existen diferencias significativas.
```

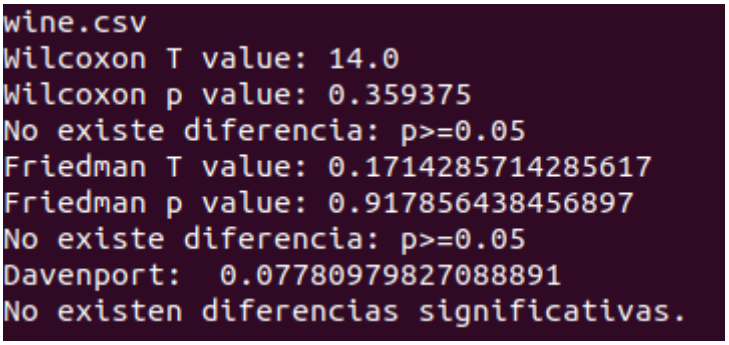
```
glass.csv
Wilcoxon T value: 1.0
Wilcoxon p value: 0.25
No existe diferencia: p>=0.05
Friedman T value: 5.733333333333333
Friedman p value: 0.056888238346101516
No existe diferencia: p>=0.05
Davenport:  3.6168224299065423
Existen diferencias significativas.
```

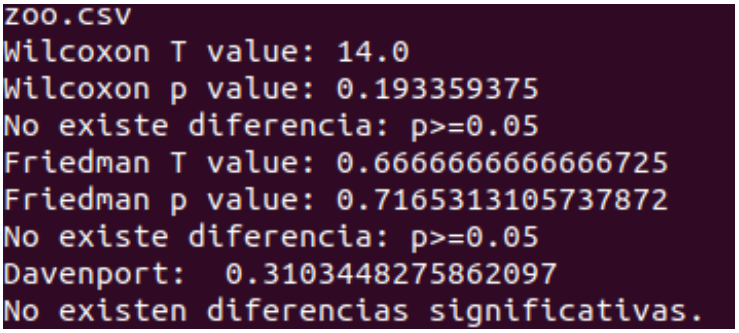
```
diabetes.csv
Wilcoxon T value: 3.0
Wilcoxon p value: 0.3125
No existe diferencia: p>=0.05
Friedman T value: 2.2105263157894752
Friedman p value: 0.33112373295101094
No existe diferencia: p>=0.05
Davenport:  1.1183431952662732
No existen diferencias significativas.
```

```
iris.csv
Wilcoxon T value: 6.0
Wilcoxon p value: 0.4375
No existe diferencia: p>=0.05
Friedman T value: 0.6086956521739081
Friedman p value: 0.7376042638218283
No existe diferencia: p>=0.05
Davenport:  0.28251121076232943
No existen diferencias significativas.
```

```
winequality-red.csv
Wilcoxon T value: 9.0
Wilcoxon p value: 0.46875
No existe diferencia: p>=0.05
Friedman T value: 0.2222222222222159
Friedman p value: 0.8948393168143727
No existe diferencia: p>=0.05
Davenport:  0.10112359550561506
No existen diferencias significativas.
```

```
winequality-white.csv
Wilcoxon T value: 14.0
Wilcoxon p value: 0.640625
No existe diferencia: p>=0.05
Friedman T value: 0.06451612903225806
Friedman p value: 0.9682566771439105
No existe diferencia: p>=0.05
Davenport:  0.029126213592233007
No existen diferencias significativas.
```





Podemos observar para los distintos test de wilcoxon que en todos ellos no existen diferencias significativas debido a que su p value en todos los casos es igual o superior a 0.05.

Por otra parte, podemos observar para los tests de Iman-Davenport que dependiendo del dataset empleado existen diferencias significativas o no entre los diferentes clasificadores.

1.7.

Para uno de los clasificadores elegidos utilice una validación de los hiperparámetros con grid search y compare su rendimiento con el método con hiperparámetros fijados a priori.

```
sol_grid = GridSearchCV(vecinos_grid, param_grid=grid_values, scoring='accuracy')
sol_grid.fit(X_train, Y_train)
accuracy = accuracy_score(y_test, sol_grid.predict(X_test))
sol_vecinos_grid.append(accuracy)
```

```
Media: 0.7195313858737504
```

Utilizando el clasificador de vecinos más cercanos, la media obtenida es la de la imagen anterior es notoriamente mayor a la del tercer ejercicio (0.62), por lo que podemos deducir que aplicando GridSearchCV los resultados han mejorado.

2.1.

Seleccione al menos tres conjuntos de datos de 2 clases. Es preferible conjuntos pequeños con variables nominales.

Los datasets elegidos para esta práctica son:

-haberman.csv

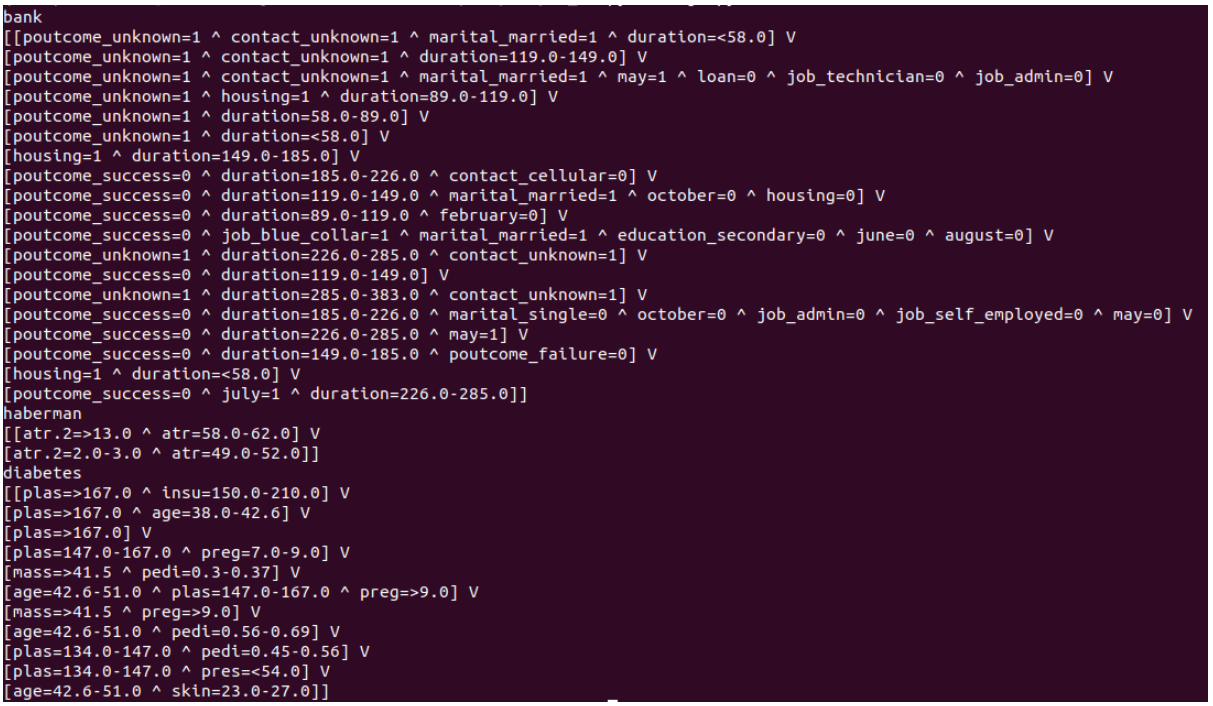
-diabetes.csv

-bank.csv

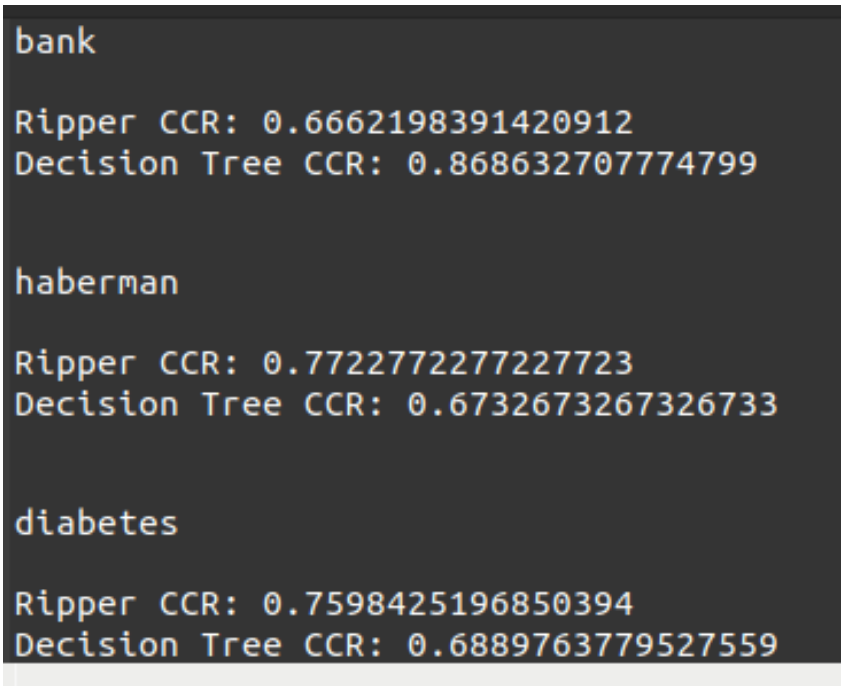
2.2.

Para cada uno de estos conjuntos aplique el método RIPPER del módulo “wittgenstein” (ejemplo en ripper.py).
Evalúe los modelos obtenidos y compárelos con un árbol de decisión.

Se muestran en la siguiente imagen los resultados obtenidos de aplicar el método RIPPER:



CCR obtenidos comparados :



2.3.

Trate de interpretar los modelos obtenidos.

Podemos observar cómo aplicar la metodología RIPPER nos permite obtener un mejor CCR en los datasets de haberman y diabetes con respecto al método utilizado con anterioridad (árboles de decisión).
Respecto al dataset bank, obtenemos mejores resultados utilizando el árbol de decisión que este nuevo método, por lo que no podemos deducir que el método RIPPER sea siempre el más óptimo.

2.4.

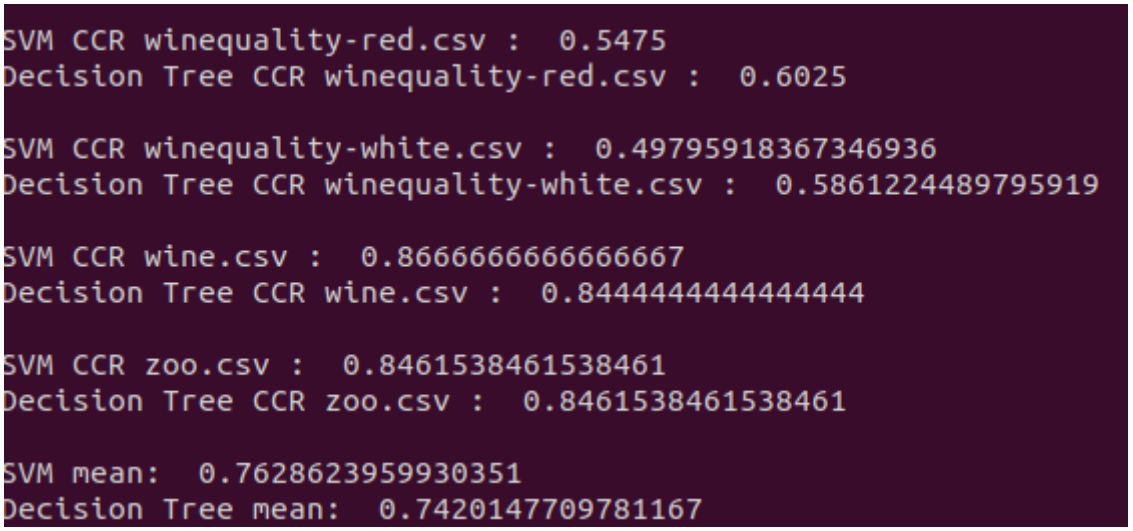
Seleccione al menos 10 conjuntos de datos. Genere un clasificador SVM con un kernel lineal y un valor fijo de C=1. Compare el rendimiento con un árbol de decisión.

Función SVM con C=1 y kernel lineal:

```
svm = SVC(C = 1, kernel = 'linear')
svm.fit(X_train, y_train)
predict = svm.predict(X_test)
score= accuracy_score(y_true = y_test, y_pred =predict)
```

Los resultados obtenidos realizando las pruebas en los datasets utilizados en la práctica anterior son los siguientes:

```
SVM CCR BreastTissue.csv :  0.7407407407407407
Decision Tree CCR BreastTissue.csv :  0.5185185185185185

SVM CCR contact-lens.csv :  0.8333333333333334
Decision Tree CCR contact-lens.csv :  0.8333333333333334

SVM CCR ecoli.csv :  0.9880952380952381
Decision Tree CCR ecoli.csv :  0.9404761904761905

SVM CCR glass.csv :  0.5740740740740741
Decision Tree CCR glass.csv :  0.6296296296296297

SVM CCR diabetes.csv :  0.7604166666666666
Decision Tree CCR diabetes.csv :  0.6979166666666666

SVM CCR iris.csv :  0.9736842105263158
Decision Tree CCR iris.csv :  0.9210526315789473
```



Comparando los scores de los diferentes clasificadores, podemos observar que dependiendo del dataset que pongamos en uso será más eficaz un clasificador u otro.

En el caso de estos datasets, podemos observar que predomina el clasificador SVM como el que en mayor número de datasets muestra mejores resultados pero no podemos concluir que sea un mejor clasificador ya que existe un número considerable de datasets que obtiene mejores resultados con árboles de decisión.

3.1.

Seleccione como métodos base un árbol de decisión y una máquina de vectores soporte.

Se utilizarán los métodos indicados junto a los diez datasets indicados en la práctica 2.1.

3.2.

Para cada uno de estos dos métodos de clasificación realice los siguientes pasos usando validación cruzada de 10 particiones:

- Aplique el método base a cada uno de los conjuntos y anote los resultados obtenidos.

```
solve=cross_val_score(DecisionTreeClassifier(), train, test, cv=10)
solve_arbol.append(solve.mean())

solve=cross_val_score(SVC(C=1, kernel='linear'), train, test, cv=10)
solve_svm.append(solve.mean())

print(f"Arboles mean: {sum(solve_arbol)/len(solve_arbol)}")
print(f"SVM mean: {sum(solve_svm)/len(solve_svm)}")
```

```
Arboles mean: 0.7348505359307215
SVM mean: 0.7574075327522001
```

Tras la ejecución del código indicado con una validación cruzada de 10, obtenemos las medias de los dos clasificadores empleados.

-Aplique el método de combinación de clasificadores Bagging a cada uno de los conjuntos y anote los resultados obtenidos:

Realizamos la codificación del método Bagging para obtener las medias de los resultados obtenidos para los clasificadores SVM y árboles de decisión:

```
arbol = BaggingClassifier(DecisionTreeClassifier(), n_estimators=5)
solve = cross_val_score(arbol, train, test, cv=10)
STreeBg.append(solve.mean())
svm = BaggingClassifier(SVC(C=1, kernel='linear', random_state=1), n_estimators=5)
solve = cross_val_score(svm, train, test, cv=10)
SSVMBg.append(solve.mean())
```

Media obtenida de los resultados obtenidos de aplicar Bagging a los diez datasets:

```
Decision Tree mean begg : 0.7634655278593674
SVM begg mean: 0.7522044428883896
```

Podemos destacar que no existen diferencias significativas respecto a los resultados de la media obtenida del clasificador SVM tanto en Bagging como método base.

Por otro lado, podemos observar que el clasificador árboles de decisión ha experimentado una mejora considerable en este nuevo método respecto al método base en el ejercicio anterior.

-Seleccione dos algoritmos de Boosting y aplique estos algoritmos a cada uno de los conjuntos y anote los resultados obtenidos.

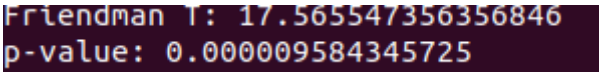
Los algoritmos utilizados para este ejercicio serán adaboost y gradient tree:

```
gradiente = GradientBoostingClassifier()
solve = cross_val_score(gradiente, train, test, cv=10)
solve_gradient.append(solve.mean())

ada = AdaBoostClassifier()
solve = cross_val_score(ada, train, test, cv=10)
solve_ada.append(solve.mean())
```

Media obtenida de los resultados obtenidos de aplicar adaboost y gradient tree a los 10 datasets:

```
GTree mean Boosting: 0.7757189926477611
AdaBoost mean: 0.6852864109730062
(base) i72cascm@i72cascm-System-Product-
```

-Compare si hay diferencias significativas entre ellos usando el test de Iman-Davenport. Si es así, aplique el procedimiento de Wilcoxon para comparar cada método de agrupación con el clasificador base.



Podemos concluir que no existen diferencias significativas en la utilización de estos clasificadores con respecto al conjunto de datasets utilizados.

3.3.

Enuncie las conclusiones del estudio indicando la influencia del clasificador base en el rendimiento de las agrupaciones de clasificadores:

Como conclusión a los resultados obtenidos durante la ejecución de los anteriores experimentos con los clasificadores, podemos destacar que para una metodología base no existen diferencias significativas entre el clasificador árbol de decisión y SVM.

Respecto al método Bagging, no existen diferencias con respecto al método SVM pero podemos destacar una notoria mejoría en el árbol de decisión.

Por último, cabe destacar en los dos algoritmos escogidos, que el método GTree boost no experimenta una gran mejoría con respecto a los anteriores, mientras que en el método AdaBoost obtenemos un notable empeoramiento.

4.1

Seleccione un algoritmo de clasificación de los disponibles en scikit que sea capaz de resolver problemas de más de dos clases y al menos 10 conjuntos de datos de más de 2 clases (puede reusar los de prácticas anteriores).

Datasets seleccionados para esta práctica (similar a prácticas anteriores):

-iris

-diabetes

-glass

-wine

-ecoli

-contact_lens

-winequality_red

-winequality_white

-zoo

-breastTisue

Utilizaremos el clasificador de árboles de decisión.

4.2.

Aplique el clasificador base a cada uno de los conjuntos y anote los resultados obtenidos.

Aplicando el clasificador base sobre los datasets:

```
Media arbol: 0.7303686979393815
```

Como podemos observar, el resultado es similar a aplicar el clasificador base de árboles de decisión en anteriores prácticas debido a que el conjunto de datasets no ha variado.

4.3.

Aplique los métodos multiclase one-vs.-one (OVO), one-vs.all (OVA) y error correcting output codes (ECOC) a cada uno de los conjuntos de datos y anote los resultados obtenidos.

Codificación de los métodos OVO, OVA y ECOC.

```
ovo = OneVsOneClassifier(DecisionTreeClassifier())
solve = cross_val_score(ovo, train, test, cv=10)
solve_ovo.append(solve.mean())
ova = OneVsRestClassifier(DecisionTreeClassifier())
solve = cross_val_score(ova, train, test, cv=10)
solve_ova.append(solve.mean())
ecoc = OutputCodeClassifier(DecisionTreeClassifier(), code_size=3)
solve = cross_val_score(ecoc, train, test, cv=10)
solve_ecoc.append(solve.mean())
```

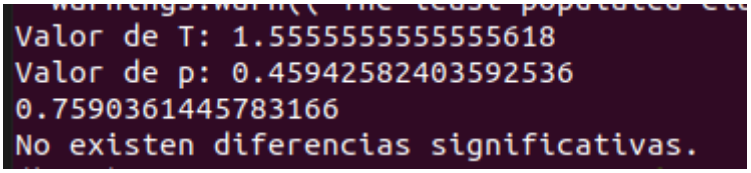
Los resultados obtenidos son:

```
OVO mean: 0.7395671459780118
OVA mean: 0.7041919305724033
ECOC mean: 0.7442960164290728
```

4.5.

Compare si hay diferencias significativas entre ellos usando el test de Iman-Davenport. Si es así, aplique el procedimiento de Wilcoxon para comparar cada método multiclase con el clasificador base y los diferentes métodos entre ellos:

Realizamos los diferentes Tests para comprobar si existen diferencias significativas:



4.7.

Enuncie las conclusiones del estudio:

Durante el estudio de este método, podemos observar que tanto ECOC como OVO obtienen un rendimiento similar respecto al clasificador árbol de decisión base.

En cuanto al método OVA, podemos observar que muestra un peor rendimiento con respecto al resto de métodos.