*Para comprobar el código: descomprimir la carpeta “AlgoritmosClasificacion” y ejecutar el archivo main.py

El dataset aportado cuenta con 7 variables, las cuales 6 son atributos y 1 es la clase objetivo, todas las variables categóricas excepto ‘class’ se han mapeado a valores numéricos para un manejo correcto en los algoritmos de clasificación:

Buying: Precio del coche, toma los valores: vhigh -> 4: Precio muy alto / high -> 3: Precio alto / med -> 2: Precio medio / low -> 1: Precio bajo

Maintenance: Precio de mantenimiento, toma los valores: vhigh -> 4: Precio muy alto / high -> 3: Precio alto / med -> 2: Precio medio / low -> 1: Precio bajo

Doors: Es el número de puertas. Toma los valores 2, 3, 4 y 5more -> 5.

Persons: Número de plazas. Toma los valores 2, 4, more -> 5.

lug_boot: Tamaño del maletero. Toma los valores: small -> 1: Pequeño / med -> 2: Mediano / big -> 3: Grande

safety: Valora la seguridad del coche. Toma los valores: high -> 3: Es muy seguro / med -> 2: Normal, en la media / low -> 1: Es poco seguro

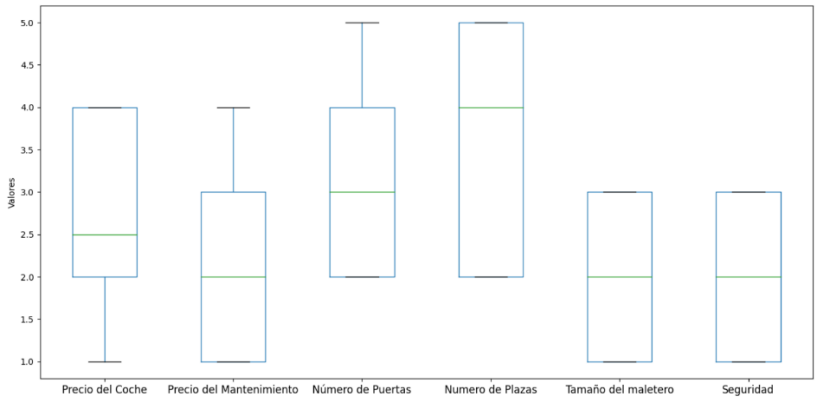
class: Es la variable clase y define si un coche es aceptable o no. Toma los valores: unacc: No es aceptable / acc: Es aceptable / good: Es bueno / vgood: Es muy bueno

Datos en formato Texto (Después de modificar las variables categóricas a numéricas)

	Buying	Maintenance	Doors	Person	lug_boot	safety	**class**
0	4	4	2	2	1	1	unacc
1	4	4	2	2	1	2	unacc
2	4	4	2	2	1	3	unacc
3	4	4	2	2	2	1	unacc
4	4	4	2	2	2	2	unacc
...	...	...	...	...	...	...	...
1745	1	1	5	5	2	3	vgood
1746	1	1	5	5	2	3	vgood
1747	1	1	5	5	3	1	unacc
1748	1	1	5	5	3	2	good
1749	1	1	5	5	3	3	vgood

[1750 rows x 7 columns]

Formato gráfico (Después de modificar las variables categóricas a numéricas):



Como se puede observar en las anteriores figuras, se tienen un total de 1750 instancias, de las cuales pertenecientes a cada clase se tienen las siguientes:

```
unacc    1215
acc       390
good       75
vgood      70
```

Los 6 atributos explicados anteriormente son todos de tipo numérico entero (int64), con el siguiente número de instancias en cada uno de sus valores.

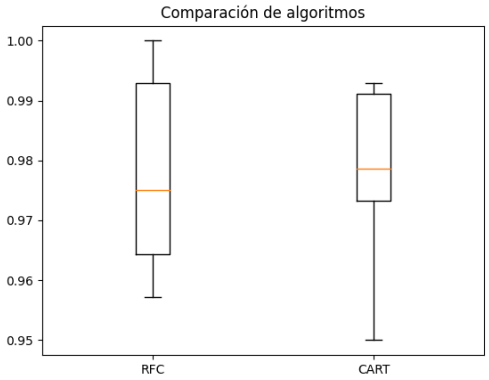
```
Name: Buying, dtype: int64
4    443
2    438
1    437
3    432
1750 en total

Name: Maintenance, dtype: int64
1    447
4    437
2    434
3    432
1750 en total

Name: Doors, dtype: int64
2    444
5    437
3    435
4    434
1750 en total

Name: Person, dtype: int64
4    587
5    585
2    578
1750 en total

Name: lug_boot, dtype: int64
3    585
2    583
1    582
1750 en total

Name: safety, dtype: int64
3    590
2    582
1    578
1750 en total
```

Hay que comprobar si existe algún dato desconocido, se observa que no lo hay:

```
Buying         0
Maintenance    0
Doors          0
Person         0
lug_boot       0
safety         0
class          0
```

Los algoritmos de clasificación que se van a comparar son CART y Random Forest, para ello, se van a escoger un 80% de las instancias para realizar el entrenamiento del clasificador y el 20% restante para los test de predicción. Además, utilizaremos una

validación cruzada estratificada de 10 veces (k-fold) para estimar la precisión del modelo:



Se observa que la mediana de CART se encuentra un poco por encima de la de RFC, en CART el 50% de los datos oscila entre 0.975 y 0.99, el rango intercuartil es de 0.015, mientras que en RFC tenemos un rango de 1-0.957=0.043, se puede observar que en RFC los valores son más dispersos, aunque en ocasiones tenemos una precisión del 100%.

Se ha evaluado el algoritmo frente a una validación cruzada estratificada de 10 veces, de ahí que tengamos una media y una desviación para el valor de la precisión.

```
Clasificador: RFC
precision media:  0.975000 (0.012474)
Precision de una prediccion:  0.9828571428571429

Clasificador: CART
precision media:  0.977857 (0.012556)
Precision de una prediccion:  0.98
```

```
Clasificador: RFC
Matriz de confusion:
 [[ 74   2   0   1]
 [  0  17   0   1]
 [  1   0 244   0]
 [  1   0   0   9]]
```

```
Clasificador: CART
Matriz de confusion:
 [[ 73   3   1   0]
 [  0  18   0   0]
 [  2   1 242   0]
 [  0   0   0  10]]
```

```
Clasificador: RFC
Reporte de clasificacion:
              precision    recall  f1-score   support

         acc       0.97      0.96      0.97        77
        good       0.89      0.94      0.92        18
       unacc       1.00      1.00      1.00       245
       vgood       0.82      0.90      0.86        10

    accuracy                           0.98       350
   macro avg       0.92      0.95      0.94       350
weighted avg       0.98      0.98      0.98       350
```

```
Clasificador: CART
Reporte de clasificacion:
              precision    recall  f1-score   support

         acc       0.97      0.95      0.96        77
        good       0.82      1.00      0.90        18
       unacc       1.00      0.99      0.99       245
       vgood       1.00      1.00      1.00        10

    accuracy                           0.98       350
   macro avg       0.95      0.98      0.96       350
weighted avg       0.98      0.98      0.98       350

Clasificador: RFC
RFC FP: 6
RFC FN: 6
RFC TP: 344
RFC TN: 1044
RFC TPR: 0.950350443207586
RFC TNR: 0.9951918858368185
RFC FPR: 0.004808114163181491
RFC PPV: 0.9216507177033493
RFC NPV: 0.9934115330206829

Clasificador: CART
CART FP: 7
CART FN: 7
CART TP: 343
CART TN: 1043
CART TPR: 0.983951762523191
CART TNR: 0.9927754975947747
CART FPR: 0.007224502405225297
CART PPV: 0.9468499812944258
CART NPV: 0.9893542905692438
```

Conclusiones: Si nos fijamos en todos los datos aportados se puede decir que los 2 clasificadores aportan una gran precisión a la hora de clasificar, en este caso una precisión media ligeramente superior en el clasificador CART (0.978 vs 0.975). Si observamos el F1 score que nos resume la precisión y sensibilidad en una sola métrica, podemos decir que, en macro, CART es ligeramente superior, donde tenemos alta precisión y bajo recall, nuestro modelo no detecta las clases todo lo bien que debería, pero si lo hace de forma confiable, y mejor que RFC. También en cuanto a TP Rate podemos observar valores más cercanos al 100% en el clasificador CART, tenemos un TPR de 0.98 vs 0.95 (diff 0.3) y sin embargo peor FPR de 0.007 vs 0.004 (diff 0.003, bastante menor), teniendo estos datos se puede decir que CART es mejor opción para este caso, la tasa de aciertos positivos es mejor en proporción que la tasa de falsos positivos.