

**Elvira**

# Proyecto Elvira

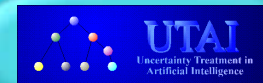
## Clasificadores En Elvira (definición y validación)

Francisco Javier García Castellano

[fjgc@decsai.ugr.es](mailto:fjgc@decsai.ugr.es)

Uncertainty Treatment in Artificial Intelligence Research Group  
Department of Computer Science and Artificial Intelligence

Granada University (Spain)



# Classifier



**Interfaz: Classifier.java**  
(elvira/learning/classification/)

**Métodos :**

**learn: Construye el modelo**

**(parámetros y estructura)**

**classify: Categoriza una instancia.**

# Constructor



**Constructor propio de cada clasificador**



**En el constructor se especifican la configuración a utilizar**



**Ejemplo:**

```
Naive_Bayes nb=new Naive_Bayes();
```

```
CMutInfTAN tan=new CMutInfTAN(casos, false);
```



**Void learn (DataBaseCases dbc, int classnumber);**


**Aprende el clasificador a partir del conjunto que se le da como argumento. Se le indica también que variable es la clase.**

**Ejemplo:**

```
nb.learn(casos, clase);
```

```
tan.learn(casos, clase);
```


# Classify



**Vector classify (configuration instance, int classnumber)**



**Categoriza la instancia.**



**Devolviendo un vector con probabilidades para cada clase**



**Ejemplo:**

```
Vector salida=nb.learn(instancia, clase);  
for (int i=0;i<salida.size();i++)  
    System.out.println("Clase "+i+  
        "probabilidad"+((Double)salida.elementAt(i)))
```

# Validacion



**Clase: ClassifierValidation.java**

( elvira/learning/classification)

**Utiliza ConfusionMatrix.java**

**Métodos de Validación:**

Train & Test

K-fold Cross-validation

Leave One Out

Test de rangos con signos de  
Wilcoxon para muestras pareadas



# Constructor



**ClassifierValidator (Classifier c,  
DataBaseCases datos, int clase)**



**validador con un clasificador no  
construido.**



**Base de datos con todos los casos**



**Ejemplo:**

```
ClassifierValidator validator= new ClassifierValidator(  
(Classifier)nb, casos, clase);
```

# Train & Test



**Vector trainAndTest ()**

**Parte los datos en :  
entrenamiento y test (2/3 y1/3)**

**Devuelve 2 matrices de  
confusión en un Vector**

**Construye el clasificador y lo  
valida**

**Ejemplo:**

```
Vector resultv=validator.trainAndTest();  
  
cm=(ConfusionMatrix)resultv.elementAt(1);  
System.out.println(" Train Error="+cm.getError()+"  
Varianza:"+cm.getVariance());  
cm.print();  
cm=(ConfusionMatrix)resultv.elementAt(1);  
System.out.println(" Test Error="+cm.getError()+"  
Varianza:"+cm.getVariance());  
cm.print();
```

# K-fold



**ConfusionMatrix  
kFoldCrossValidation (int k)**

**K particiones**

**Devuelve una matriz de confusión**

**Construye el clasificador y lo valida k veces**

**Ejemplo:**

```
cm=validator.kFoldCrossValidation(k);  
cm=(ConfusionMatrix)resultv.elementAt(1);  
System.out.println(" K-Fold Error="+cm.getError()+"  
Varianza:"+cm.getVariance());  
cm.print();
```



# Leave one out



**ConfusionMatrix  
leaveOneOut ()**

**K-Fold Cross Validation con  
k=n, n número de casos**

**Devuelve una matriz de confusión**

## **Ejemplo:**

```
cm=validator.leavingOneOut();  
cm=(ConfusionMatrix)resultv.elementAt(1);  
System.out.println(" Leave One Out Error="+  
    cm.getError()+" Varianza:"+cm.getVariance());  
cm.print();
```



# Wilcoxon



double  
wilcoxonPairedSignedRankTest  
(Vector lista1, Vector lista2);



Éste método aplica el test de signos por rangos de Wilcoxon para muestras pareadas.



Se aplica a 2 vectores de matrices de confusión (usa sus errores).



El tamaño de la muestras tiene que estar entre 6 y 25, ambos incluidos.



Devuelve el nivel de significación (0.05, 0.02, 0.01 ó 0)

# Wilcoxon 2



Se usa K-Fold Cross-Validation

Se hace la misma prueba para los dos clasificadores. Devolviendo Vectores.

Se cambia de clasificador

```
ClassifierValidator validator= new ClassifierValidator(  
(Classifier)nb, casos, clase);
```

```
Vector
```

```
matricesNB=validator.kFoldCrossValidation_Vector(k);
```

```
validator.setClassifier((Classifier)tan);
```

```
Vector
```

```
matricesTAN=validator.kFoldCrossValidation_Vector(k);
```

# Wilcoxon 3



De la lista de matrices se puede sacar la matriz media

Se realiza el test

Si el resultados es distinto de 0, los resultados son significativos

```
ConfusionMatrix mediaNB= new ConfusionMatrix  
((FiniteStates)casos.getVariables().elementAt(clase));  
mediaNB.average(matricesNB);
```

```
ConfusionMatrix mediaTAN= new ConfusionMatrix  
((FiniteStates)casos.getVariables().elementAt(clase));  
mediaTAN.average(matricesTAN);
```

```
double salida=validator.wilcoxonPairedSignedRankTest  
(matricesNB, matricesTAN);
```

