

# Reporte Weka y Jupyter Notebook | Inteligencia Artificial

Marco Ferraro | B82957

Grabiel Revillat | B86524

Steven Nuñez | B95614

## Enunciado

1. Vea el video de SVM usando Weka que se agregó.
2. Durante el tiempo de clase ( o antes si lo desean) ustedes llevarán a cabo en grupo ( con su equipo de trabajo ) lo siguiente:
  - a) Seleccionarán un archivo de datos .csv creado para la Tarea de Naive Bayes por alguno de los tres integrantes del grupo.
  - b) Siguiendo la información del video, habilitarán LibSVM en Weka para crear un modelo con la colección de Training y Evaluarán sobre la de Testing que eligieron.
  - c) Comparen los resultados de: Naive Bayes contra SVM ( kernel lineal, radial y poli), ajusten los hiperparámetros vistos en la presentación de la clase del martes antes de evaluar sobre el data set de testing. Reporten tanto la calidad del modelo como los resultados de evaluación sobre el testing.
3. Usando Jupiter Notebook carguen los datos de training y usando el cuaderno visto el martes, modifíquelo para evaluar el data set de testing con los mismos hiperparámetros que usaron con Weka.
4. Deben hacer una síntesis y un análisis de los resultados obtenidos en general.

## Bayes y Máquinas de Soporte Vectorial en Weka

Para este trabajo se utiliza el set de datos de películas y su país de origen. El *target variable* es USA o Germany.

Para la primera parte de esta tarea, volvimos a probar el modelo de Bayes en los datos de prueba y entrenamiento.

The screenshot shows the Weka Explorer interface with the NaiveBayes classifier selected. The 'Test options' section on the left indicates 'Cross-validation' with 'Folds' set to 5. The 'Classifier output' pane on the right displays the following results:

Time taken to build model: 1.34 seconds

=== Stratified cross-validation ===  
=== Summary ===

Metric	Value	Percentage
Correctly Classified Instances	10729	94.3458 %
Incorrectly Classified Instances	643	5.6542 %
Kappa statistic	0.3619	
Mean absolute error	0.102	
Root mean squared error	0.2299	
Relative absolute error	76.0325 %	
Root relative squared error	88.764 %	
Total Number of Instances	11372	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.241	0.002	0.912	0.241	0.381	0.453	0.758	0.396	Germany
	0.998	0.759	0.944	0.998	0.970	0.453	0.758	0.967	USA
Weighted Avg.	0.943	0.704	0.942	0.943	0.928	0.453	0.758	0.925	

=== Confusion Matrix ===

```
a      b  <-- classified as
198   624 |  a = Germany
 19 10531 |  b = USA
```

The 'Result list' on the bottom left shows two entries: '17:50:49 - bayes.NaiveBayes from file 'BayesModel.model'' and '17:50:53 - bayes.NaiveBayes', with the latter selected. The 'Status' bar at the bottom indicates 'OK'.

Ilustración 1 Resultados Training con 5 Cross Folds Bayes

Weka Explorer

Preprocess

Classify

Cluster

Associate

Select attributes

Visualize

Choose

NaiveBayes

Test options

☐ Use training set

☒ Supplied test set
 

Set...

☐ Cross-validation
 

Folds5

☐ Percentage split
 

%66

More options...

(Nom) country

Start

Stop

Result list (right-click for options)

17:50:49 - bayes.NaiveBayes from file 'BayesModel.model'

17:50:53 - bayes.NaiveBayes

17:51:40 - misc.InputMappedClassifier

Classifier output

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.67 seconds

=== Summary ===

Correctly Classified Instances	4727	93.3821 %
Incorrectly Classified Instances	335	6.6179 %
Kappa statistic	0.0626	
Mean absolute error	0.1133	
Root mean squared error	0.248	
Relative absolute error	86.4997 %	
Root relative squared error	98.1495 %	
Total Number of Instances	5062	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.035	0.000	1.000	0.035	0.067	0.180	0.664	0.180	Germany
	1.000	0.965	0.934	1.000	0.966	0.180	0.664	0.953	USA
Weighted Avg.	0.934	0.899	0.938	0.934	0.904	0.180	0.664	0.900	

=== Confusion Matrix ===

a

b

<-- classified as

12	335		a = Germany
0	4715		b = USA

Status

OK

Log

x0

*Ilustración 2 Resultados Testing con el modelo de Bayes de Ilustración 1*

weka.gui.GenericObjectEditor

weka.classifiers.functions.LibSVM

coef0	0.0
cost	1.0
debug	False
degree	3
doNotCheckCapabilities	False
doNotReplaceMissingValues	False
eps	0.001
gamma	0.0
kernelType	linear: u*v
loss	0.1
modelFile	Weka-3-8-5
normalize	False
Whether to normalize the data	
nu	0.5
numDecimalPlaces	2
probabilityEstimates	False
seed	1
shrinking	True
weights	

Open... Save... OK Cancel

*Ilustración 3 Definición SVM con kernel lineal*

Weka Explorer

Preprocess **Classify** Cluster Associate Select attributes Visualize

**Classifier**

Choose **LibSVM** -S 0 -K 0 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -model D:\AppData\Weka\Weka-3-8-5 -seed 1

**Test options**

☐ Use training set  
☐ Supplied test set Set...  
☒ Cross-validation Folds **5**  
☐ Percentage split % 66  
 More options...

(Nom) country

Start Stop

**Result list (right-click for options)**

17:36:59 - functions LibSVM

**Classifier output**

```

Time taken to build model: 1.58 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      10816      95.1108 %
Incorrectly Classified Instances     556       4.8892 %
Kappa statistic                    0.4862
Mean absolute error                 0.0489
Root mean squared error             0.2211
Relative absolute error             36.4332 %
Root relative squared error         85.3874 %
Total Number of Instances          11372

=== Detailed Accuracy By Class ===

          TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
          0.347    0.002    0.938     0.347    0.506     0.554    0.672    0.372    Germany
          0.998    0.653    0.951     0.998    0.974     0.554    0.672    0.951    USA
Weighted Avg.    0.951    0.606    0.950     0.951    0.940     0.554    0.672    0.910

=== Confusion Matrix ===

      a    b  <-- classified as
285  537 |    a = Germany
 19 10531 |    b = USA
  
```

**Status**

OK Log x 0

*Ilustración 4 Resultados Training con 5 Cross Folds kernel lineal*

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

**Classifier**

Choose LibSVM - S 0 -K 0 -D 3 -G 0.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -model D:\AppData\Weka\Weka-3-8-5 -seed 1

**Test options**

☐ Use training set  
☒ Supplied test set Set...  
☐ Cross-validation Folds 5  
☐ Percentage split % 66  
 More options...

(Nom) country

Start Stop

**Result list (right-click for options)**

17:36:59 - functions.LibSVM  
17:38:24 - misc.InputMappedClassifier

**Classifier output**

```

=== Evaluation on test set ===

Time taken to test model on supplied test set: 2.96 seconds

=== Summary ===

Correctly Classified Instances      4739      93.6191 %
Incorrectly Classified Instances    323       6.3809 %
Kappa statistic                    0.1534
Mean absolute error                 0.0638
Root mean squared error             0.2526
Relative absolute error             48.7136 %
Root relative squared error         99.9557 %
Total Number of Instances          5062

=== Detailed Accuracy By Class ===

      TP Rate  FP Rate  Precision  Recall   F-Measure  MCC    ROC Area  PRC Area  Class
      0.092    0.002    0.800     0.092    0.165     0.258   0.545    0.136    Germany
      0.998    0.908    0.937    0.998    0.967     0.258   0.545    0.937    USA
Weighted Avg.   0.936    0.846    0.928    0.936    0.912     0.258   0.545    0.882

=== Confusion Matrix ===

  a    b  <-- classified as
32  315 |   a = Germany
 8 4707 |   b = USA
  
```

**Status**

OK Log x 0

Ilustración 5 Resultados Testing kernel lineal

weka.gui.GenericObjectEditor

weka.classifiers.functions.LibSVM

**About**

A wrapper class for the libsvm library.

More

Capabilities

SVMTType C-SVC (classification)

batchSize 100

cacheSize 40.0

coef0 0.0

cost 1.0

debug False

degree 3

doNotCheckCapabilities False

doNotReplaceMissingValues False

eps 0.001

gamma 5.0

kernelType radial basis function:  $\exp(-\gamma \|u-v\|^2)$

loss 0.1

modelFile Weka-3-8-5

normalize False

nu 0.5

numDecimalPlaces 2

probabilityEstimates False

seed 1

shrinking True

weights

Open... Save... OK Cancel

Ilustración 6 Definición SVM con kernel radial y gamma 5

Weka Explorer

Preprocess **Classify** Cluster Associate Select attributes Visualize

**Classifier**

Choose **LibSVM** -S 0 -K 2 -D 3 -G 5.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -model D:\AppData\Weka\Weka-3.8-5 -seed 1

**Test options**

☐ Use training set  
☐ Supplied test set Set...  
☒ Cross-validation Folds **5**  
☐ Percentage split % 66  
 More options...

(Nom) country

Start Stop

**Result list (right-click for options)**

17:47:50 - functions LibSVM

**Classifier output**

```

Time taken to build model: 7.2 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      10723           94.293 %
Incorrectly Classified Instances      649           5.707 %
Kappa statistic                    0.3709
Mean absolute error                  0.0571
Root mean squared error              0.2389
Relative absolute error              42.5272 %
Root relative squared error          92.2526 %
Total Number of Instances          11372

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.254    0.003    0.853     0.254    0.392     0.447    0.625    0.271    Germany
               0.997    0.746    0.945     0.997    0.970     0.447    0.625    0.945    USA
Weighted Avg.   0.943    0.692    0.938     0.943    0.928     0.447    0.625    0.896

=== Confusion Matrix ===

      a    b  <-- classified as
    209  613 |    a = Germany
     36 10514 |    b = USA
  
```

**Status**

OK Log x 0

*Ilustración 7 Resultados Training con 5 Cross Folds kernel radial*



Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

**Classifier**

Choose **LibSVM** -S 0 -K 2 -D 3 -G 5.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -model D:\AppData\Weka\Weka-3-8-5-seed 1

**Test options**

☐ Use training set  
☒ Supplied test set **Set...**  
☐ Cross-validation Folds 5  
☐ Percentage split % 66  
**More options...**

(Nom) country

**Start** **Stop**

**Result list (right-click for options)**

17:47:50 - functions.LibSVM

17:49:19 - misc.InputMappedClassifier

**Classifier output**

```

=== Evaluation on test set ===

Time taken to test model on supplied test set: 27.41 seconds

=== Summary ===

Correctly Classified Instances      4725      93.3426 %
Incorrectly Classified Instances    337      6.6574 %
Kappa statistic                    0.092
Mean absolute error                0.0666
Root mean squared error            0.258
Relative absolute error            50.825 %
Root relative squared error       102.0989 %
Total Number of Instances         5062

=== Detailed Accuracy By Class ===

      TP Rate  FP Rate  Precision  Recall  F-Measure  MCC  ROC Area  FRC Area  Class
      0.055    0.002    0.679    0.055    0.101    0.180  0.526    0.102    Germany
      0.998    0.945    0.935    0.998    0.965    0.180  0.526    0.935    USA
Weighted Avg.   0.933    0.881    0.917    0.933    0.906    0.180  0.526    0.878

=== Confusion Matrix ===

  a    b  <-- classified as
19  328 |  a = Germany
 9  4706 |  b = USA

```

**Status**

OK **Log** x 0

*Ilustración 8 Resultados Testing con kernel radial*

weka.gui.GenericObjectEditor

weka.classifiers.functions.LibSVM

**About**

A wrapper class for the libsvm library.

More

Capabilities

SVMType: C-SVC (classification)

batchSize: 100

cacheSize: 40.0

coef0: 0.0

cost: 1.0

debug: False

degree: 3

doNotCheckCapabilities: False

doNotReplaceMissingValues: False

eps: 0.001

gamma: 5.0

kernelType: polynomial:  $(\gamma u \cdot v + \text{coef0})^{\text{degree}}$

loss: 0.1

modelFile: Weka-3-8-5

normalize: False

nu: 0.5

numDecimalPlaces: 2

probabilityEstimates: False

seed: 1

shrinking: True

weights:

Open... Save... OK Cancel

*Ilustración 9 Definición SVM con kernel polinomial y gamma 5*

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

**Classifier**

Choose LibSVM -S 0 -K 1 -D 3 -G 5.0 -R 0.0 -N 0.5 -M 40.0 -C 1.0 -E 0.001 -P 0.1 -model D:\AppData\Weka\Weka-3-8-5 -seed 1

**Test options**

☐ Use training set  
☐ Supplied test set Set...  
☒ Cross-validation Folds 5  
☐ Percentage split % 66  
 More options...

(Nom) country

Start Stop

**Result list (right-click for options)**

17:43:39 - functions.LibSVM

**Classifier output**

```

Time taken to build model: 0.68 seconds

=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances      10720          94.2666 %
Incorrectly Classified Instances     652           5.7334 %
Kappa statistic                    0.4416
Mean absolute error                 0.0573
Root mean squared error             0.2394
Relative absolute error             42.7238 %
Root relative squared error         92.4656 %
Total Number of Instances          11372

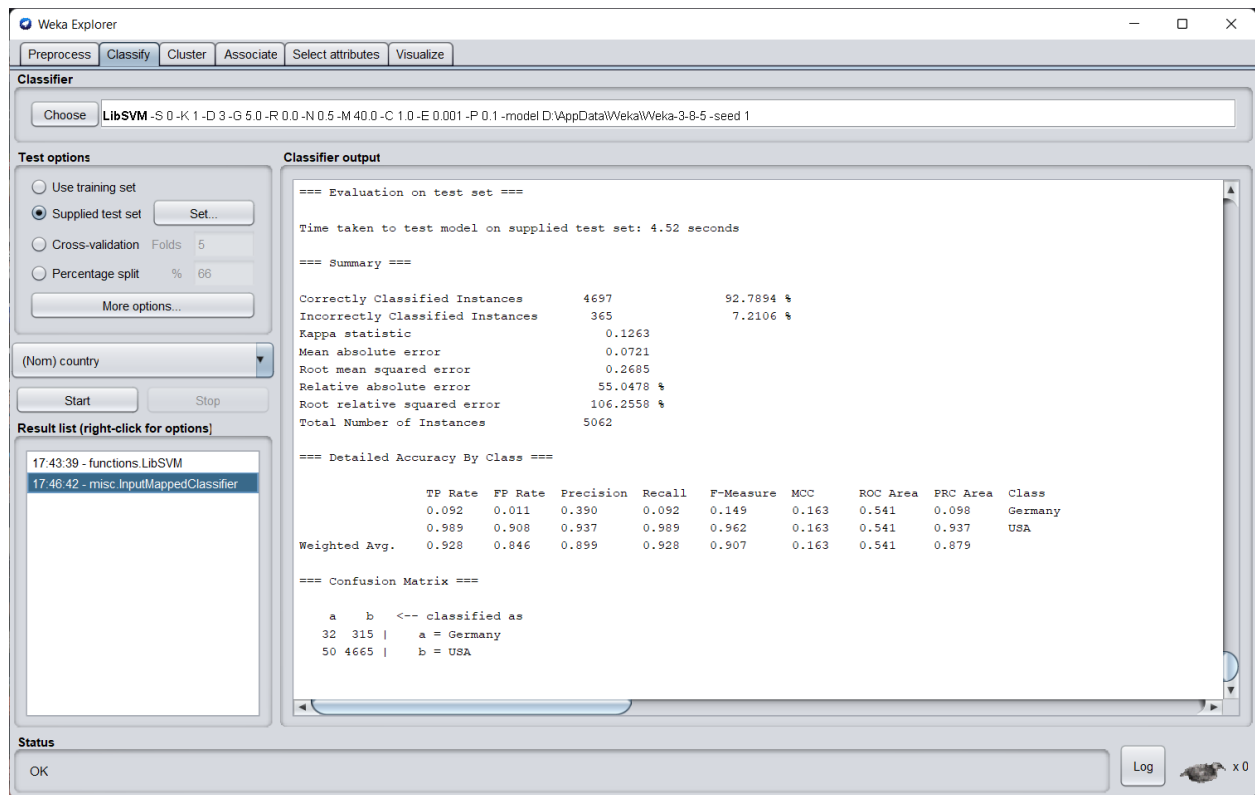
=== Detailed Accuracy By Class ===
               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.349    0.011    0.710     0.349    0.468     0.473    0.669    0.295    Germany
               0.989    0.651    0.951     0.989    0.970     0.473    0.669    0.951    USA
Weighted Avg.   0.943    0.605    0.934     0.943    0.933     0.473    0.669    0.904

=== Confusion Matrix ===
      a    b  <-- classified as
      287  535 |    a = Germany
      117 10433 |    b = USA
  
```

**Status**

OK Log x 0

Ilustración 10 Resultados training con kernel polinomial



*Ilustración 11 Resultados Testing con kernel polinomial*

```

In [14]: # Ligar el dataset con el clasificador
SVM = svm.SVC(C=1, kernel='linear', degree=3, gamma='auto')
SVM.fit(Train_X_Tfidf, Train_Y)
# Hacer las predicciones con el conjunto de prueba
predictions_SVM = SVM.predict(Test_X_Tfidf)

```

*Ilustración 12 Definicion kernel lineal en Python*

```

In [18]: # Ligar el dataset con el clasificador
SVM = svm.SVC(C=1, kernel='rbf', degree=3, gamma='scale')
SVM.fit(Train_X_Tfidf, Train_Y)
# Hacer las predicciones con el conjunto de prueba
predictions_SVM = SVM.predict(Test_X_Tfidf)

```

*Ilustración 13 Definicion kernel radial en Python*

```
In [16]: # Ligar el dataset con el clasificador
SVM = svm.SVC(C=1, kernel='poly', degree=3, gamma='scale')
SVM.fit(Train_X_Tfidf, Train_Y)
# Hacer las predicciones con el conjunto de prueba
predictions_SVM = SVM.predict(Test_X_Tfidf)
```

*Ilustración 14 Definición kernel polinomial en Python*

```
In [15]: print("SVM Accuracy Score -> ", skm.accuracy_score(Test_Y, predictions_SVM)*100)
print("SVM Recall ", etiqueta0, skm.recall_score(Test_Y, predictions_SVM, pos_label=0))
print("SVM Recall ", etiqueta1, skm.recall_score(Test_Y, predictions_SVM, pos_label=1))
print("SVM Precision ", etiqueta0, skm.precision_score(Test_Y, predictions_SVM, pos_label=0))
print("SVM Precision ", etiqueta1, skm.precision_score(Test_Y, predictions_SVM, pos_label=1))
print("SVM F ", etiqueta0, skm.f1_score(Test_Y, predictions_SVM, pos_label=0))
print("SVM F ", etiqueta1, skm.f1_score(Test_Y, predictions_SVM, pos_label=1))
print("SVM Confusion Table: \n", skm.confusion_matrix(Test_Y, predictions_SVM))

SVM Accuracy Score -> 94.15250888976689
SVM Recall Germany 0.15273775216138327
SVM Recall USA 0.999575821845175
SVM Precision Germany 0.9636363636363636
SVM Precision USA 0.9412822049131216
SVM F Germany 0.26368159203980096
SVM F USA 0.9695535897963382
SVM Confusion Table:
[[ 53 294]
 [ 2 4713]]
```

*Ilustración 15 Resultados kernel lineal en Python*

```
In [19]: print("SVM Accuracy Score -> ", skm.accuracy_score(Test_Y, predictions_SVM)*100)
print("SVM Recall ", etiqueta0, skm.recall_score(Test_Y, predictions_SVM, pos_label=0))
print("SVM Recall ", etiqueta1, skm.recall_score(Test_Y, predictions_SVM, pos_label=1))
print("SVM Precision ", etiqueta0, skm.precision_score(Test_Y, predictions_SVM, pos_label=0))
print("SVM Precision ", etiqueta1, skm.precision_score(Test_Y, predictions_SVM, pos_label=1))
print("SVM F ", etiqueta0, skm.f1_score(Test_Y, predictions_SVM, pos_label=0))
print("SVM F ", etiqueta1, skm.f1_score(Test_Y, predictions_SVM, pos_label=1))
print("SVM Confusion Table: \n", skm.confusion_matrix(Test_Y, predictions_SVM))

SVM Accuracy Score -> 93.71789806400632
SVM Recall Germany 0.0893371757925072
SVM Recall USA 0.999575821845175
SVM Precision Germany 0.9393939393939394
SVM Precision USA 0.9371644462119706
SVM F Germany 0.16315789473684209
SVM F USA 0.9673645320197044
SVM Confusion Table:
[[ 31 316]
 [ 2 4713]]
```

*Ilustración 16 Resultados kernel radial en Python*

```

In [17]: print("SVM Accuracy Score -> ", skm.accuracy_score(Test_Y, predictions_SVM)*100)
print("SVM Recall ", etiqueta0, skm.recall_score(Test_Y, predictions_SVM, pos_label=0))
print("SVM Recall ", etiqueta1, skm.recall_score(Test_Y, predictions_SVM, pos_label=1))
print("SVM Precision ", etiqueta0, skm.precision_score(Test_Y, predictions_SVM, pos_label=0))
print("SVM Precision ", etiqueta1, skm.precision_score(Test_Y, predictions_SVM, pos_label=1))
print("SVM F ", etiqueta0, skm.f1_score(Test_Y, predictions_SVM, pos_label=0))
print("SVM F ", etiqueta1, skm.f1_score(Test_Y, predictions_SVM, pos_label=1))
print("SVM Confusion Table: \n", skm.confusion_matrix(Test_Y, predictions_SVM))

SVM Accuracy Score -> 93.16475701303833
SVM Recall Germany 0.005763688760806916
SVM Recall USA 0.9997879109225875
SVM Precision Germany 0.6666666666666666
SVM Precision USA 0.9318047044870528
SVM F Germany 0.011428571428571429
SVM F USA 0.9645999590750972
SVM Confusion Table:
[[ 2 345]
 [ 1 4714]]

```

Ilustración 17 Resultados kernel polinomial en Python

## Conclusiones

Para analizar los resultados hay que tener claro que las maquinas de Soporte Vectorial en Jupyter y en Weka tienen que analizarse sin relacionar los resultados entre estas mismas, ya que, al ser Máquinas de Soporte Vectorial diferentes representan una variable de bloqueo.

## Análisis Weka

En los rendimientos del modelo, entre las Maquinas de Soporte Vectorial vemos que la que tiene mejor rendimiento es la de kernel lineal, ya que tiene más precisión para predecir las X que pertenecen a **Germany**. Dicho esto, hay algo peculiar; por alguna razón, el modelo de Naive Bayes tiene precisión del 100 en el *target variable* **Germany**. Podemos decir que es por la naturaleza de los datos, pero es algo que llama poderosamente la atención, y más porque acierta todos los *True Negative*.

## Análisis Jupyter

Para trabajar esta parte, se modificó el código de Python para respetar la separación de los datos de prueba y de entrenamiento. No se separan con los métodos de *sci-kit learn*, si no que se importan como csv aparte (el notebook viene adjunto con esta tarea). Dentro de los resultados que genera *sci-kit learn* vemos que el kernel lineal tiene un mejor resultado, pero marginalmente. De conclusión, para analizar datos con 1 X y 1 *target variable* con maquinas de soporte vectorial recomendamos utilizar el kernel lineal, ya que no es tan complejo y parece que su umbral de decisión es más preciso que los otros kernel.

