Introducción al Procesamiento de Lenguaje Natural

Grupo de PLN – InCo

Clasificación

Clasificación

- Dado un objeto y un conjunto de clases, quiero saber a qué clase pertenece el objeto.
- Muchas tareas de PLN pueden verse como problemas de clasificación
    - Idioma de un documento
    - Saber si una palabra lleva tilde diacrítico
    - Saber si una oración es especulativa
    - Saber si una oración es un *weasel*

Iris Data Set

- Iris dataset:
  - 150 flores de la especie Iris
  - Identificadas por largo y ancho de sépalo y pétalo (reales)
  - Tres clases: Iris Setosa, Iris Versicolor, Iris Virginica

```
Feature names:['sepal length (cm)', 'sepal width (cm)', 'petal
length (cm)', 'petal width (cm)']
Target classes:['setosa' 'versicolor' 'virginica']
First instance features:[ 5.1  3.5  1.4  0.2]
```

Titanic Data Set

Pasajeros del Titanic y qué pasó con ellos

```
['row.names' 'pclass' 'survived' 'name'
'age' 'embarked' 'home.dest' 'room'
 'ticket' 'boat' 'sex']
['1' '1st' '1' 'Allen, Miss Elisabeth
Walton' '29.0000' 'Southampton'
 'St Louis, MO' 'B-5' '24160 L221' '2'
'female'] 1
```

Clasificación

- Dos grandes líneas
  - Reglas
  - Aprendizaje automático

  - … en general, los mejores clasificadores combinan ambas (*métodos híbridos*)

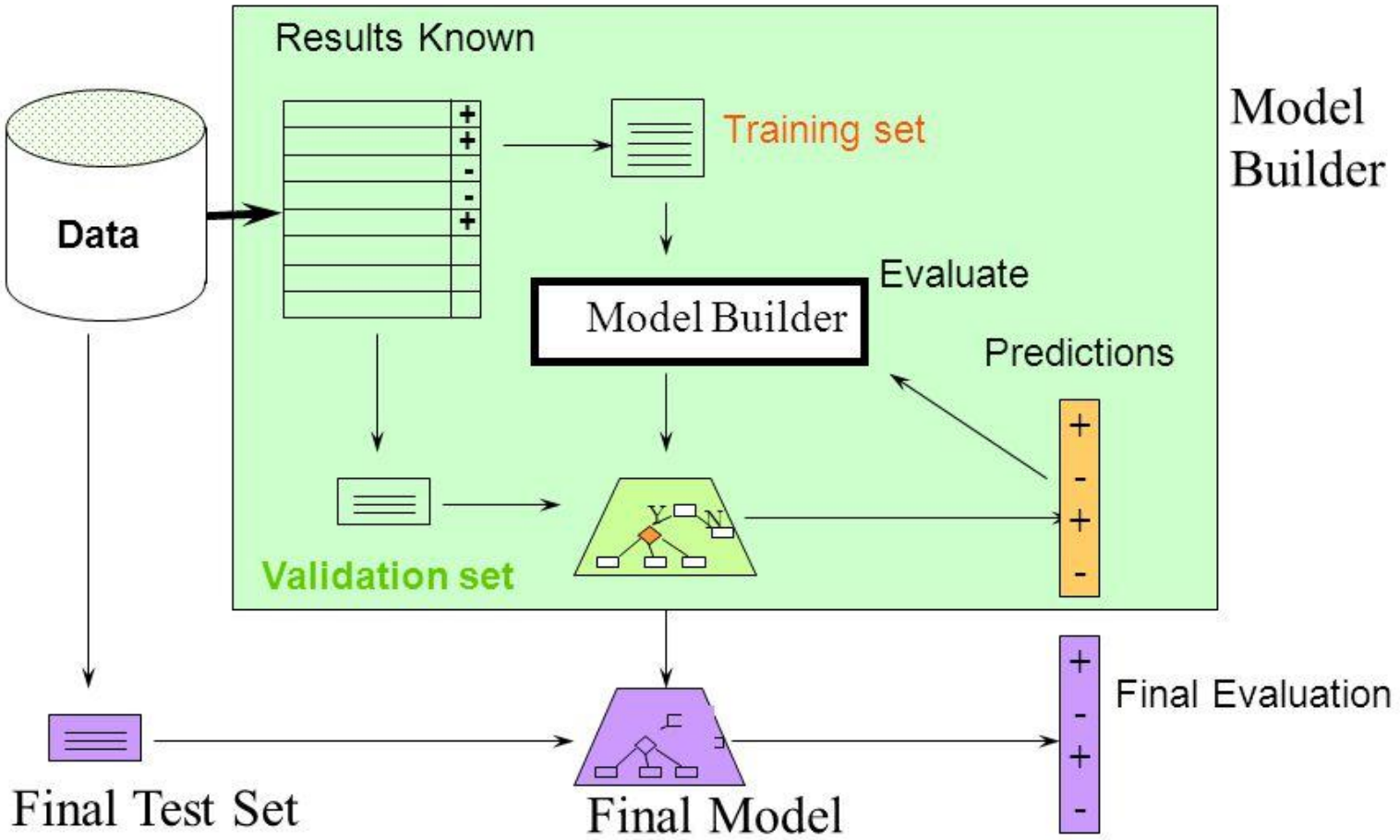
Aprendizaje automático

- Tenemos ejemplos anotados (conjunto de entrenamiento)
  - D= { <d,c> / <d,c> \in X x C}
  - X es el espacio de instancias
  - C es el conjunto de clases
- Una función de clasificación mapea documentos a clases:
  - Y: X -> C
- Un método de **aprendizaje supervisado** recibe los ejemplos anotados y devuelve la función de clasificación

Corpus de aprendizaje

- Dividimos el corpus en dos partes:
    - Corpus de entrenamiento (80%?)
    - Corpus de evaluación (20%?)
- Para evaluar, nuestro conjunto de evaluación debe ser *diferente* al de entrenamiento
- … pero deberían tener la misma distribución

Classification: Train, Validation, Test split



Fuente: **Evaluation and Credibility How much should we believe in what was learned?**
Published by Monica Dorsey

Iris Data Set

- Iris dataset:
  - 112 en el corpus de entrenamiento (75%)
  - 38 en el corpus de evaluación (25%)
  - Instancias elegidas al azar!

Esquema general

1) Ingeniería de atributos
2) Selección de atributos
3) Selección de modelo (ajuste de parámetros)
4) Aprendizaje
5) Evaluación

Ingeniería de atributos

- En general, los métodos de clasificación trabajan sobre conjuntos de pares atributo/valor
  - Atributos para palabras: forma de superficie, lema, terminación, POS, pertenencia a una lista, cantidad de letras, etc, etc.
  - Atributos para oraciones: incluye palabra de una lista, número de veces que aparece una palabra, incluye expresión, largo, etc, etc.
  - Atributos para documentos: largo, ocurrencias de cada palabra, etc, etc.
- Uno de los atributos es la clase objetivo (*target class*)
- La ingeniería de atributos es precisamente obtener esos atributos a partir de nuestra forma original

Ingeniería de atributos

- Algunas estrategias
  - Para convertir categorías a números: one-hot-encoding
    - `idioma: {español, inglés, francés}` ➔
      - `Es_español:{0,1}`
      - `Es_inglés:{0,1}`
      - `Es_francés:{0,1}`
  - Para convertir números a categorías:
    - Dividir en rangos
      - Iguales
      - Según distribución en el corpus de ent.

Ingeniería de atributos

- ¿Cómo…
  - Extraer atributos de árboles?
    - Distancia a la raíz, secuencia de categorías desde la raíz a cierta hoja, componente padre de una hoja, profundidad del árbol, etc, etc.
  - Extraer atributos de grafos?
    - Conectividad, número de relaciones de cierto tipo, etc., etc.
- Reglas de conocimiento
  - Intentan incorporar conocimiento experto como atributo
  - Idea: a)construir una regla determinista que tome una instancia y prediga una clase. b) Incorporar el resultado de la regla como un nuevo atributo
    - Ej: Si un correo incluye la palabra “viagra” y lo envía un tal “Sltionsforyou”, entonces el correo es spam.

Titanic Data Set

```
New feature names:
['age', 'sex', 'first_class',
'second_class', 'third_class']
Values: [ 29.   0.   1.   0.   0.]
```

Selección de Atributos

- Encontrar el menor número de atributos que permitan caracterizar al corpus

- Razones
  - A veces, demasiados atributos pueden producir sobreajuste
  - Performance

Selección de Atributos

- Métodos
  - Estadísticos (atributos muy correlacionados con la clase objetivo, poco correlacionados entre ellos)
    - Chi-square test (descartamos atributos independientes de la clase objetivo)
  - Performance en el corpus de entrenamiento
    - ¡Podemos sobreajustar!
      - Corpus *held-out*
      - *Cross-validation*

Selección de Modelo

- Los modelos de aprendizaje generalmente incluyen *hiperparámetros.*

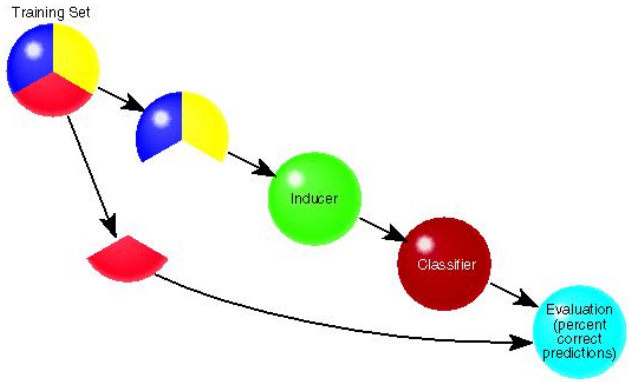
class sklearn.tree.DecisionTreeClassifier(*criterion='gini'*, *splitter='best'*, *max_depth=None*, *min_samples_split=2*, *min_samples_leaf=1*, *min_weight_fraction_leaf=0.0*, *max_features=None*, *random_state=None*, *max_leaf_nodes=None*, *class_weight=None*, *presort=False*)

Detalles

Selección de Modelo

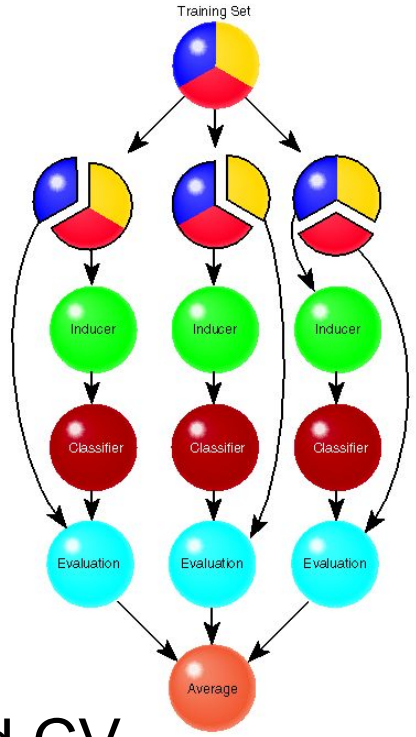
- ¿Cómo ajustamos los parámetros?
  - Corpus held-out
  - Cross Validation
- Corpus held-out
  - Separamos una parte del corpus de entrenamiento y lo utilizamos para evaluar
- Cross-Validation
  - Divido el corpus de entrenamiento en k partes
  - Entreno sobre (k-1) partes y evalúo en la restante
  - Repito para cada parte, y calculo la media

Selección de Modelo



Held out corpus

Tomado de [Mineset'S User Guide]



3-fold CV

Medidas de evaluación

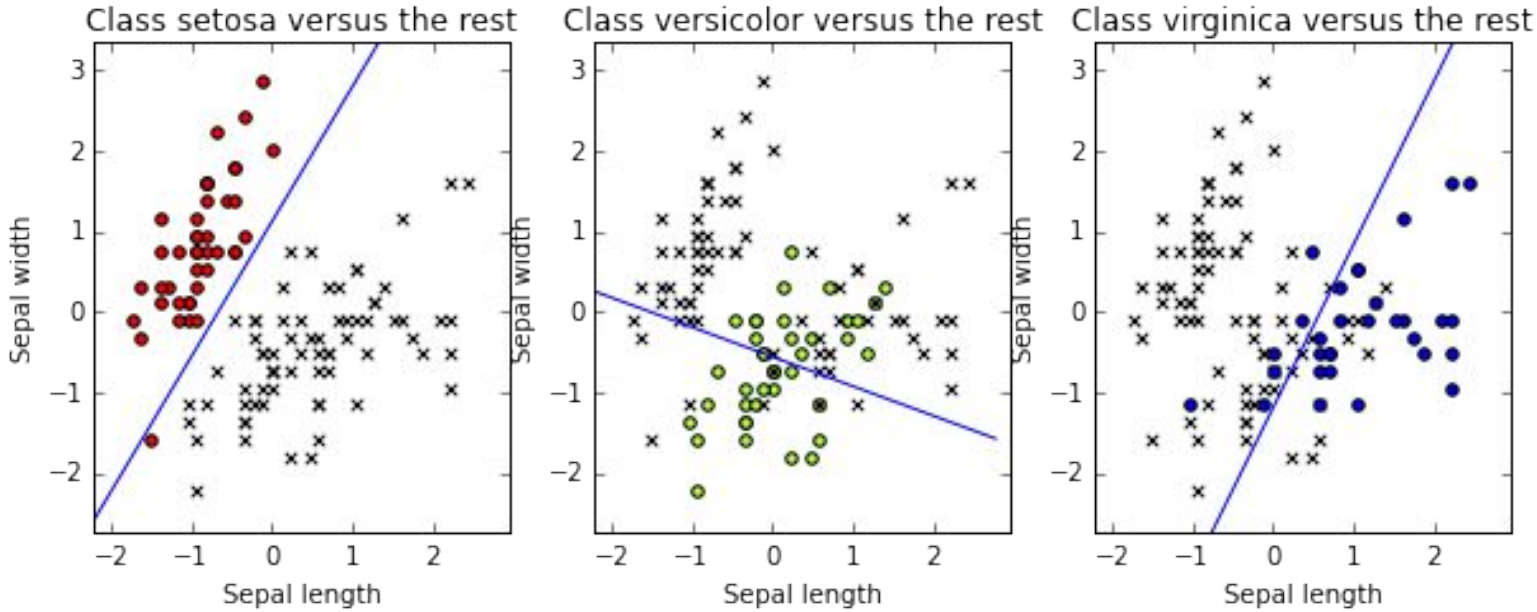
- Accuracy (Precisión)
  - Fracción de las instancias clasificadas correctamente
- Las usuales en el PLN son:
  - Precisión (Precision!)
    - Instancias positivas que se clasificaron correctamente / Total de Instancias Positivas
    - TP / (TP + FP) [TP= True Positives, FP = False Positives]
  - Exhaustividad (*recall*)
    - Fracción de las instancias positivas que se clasificaron correctamente / Fracción de las instancias que son positivas
    - TP/(TP + FN) [FN = False Negatives]
  - Medida-F
    - Media armónica entre Precisión y Recall: 2*Precision*Recall/ (Precision + Recall)

Aprendizaje

- Utilizamos el corpus de entrenamiento para generar un *modelo* (función de clasificación)
- Utilizamos la función de clasificación para calcular la clase objetivo para cada instancia del corpus de evaluación
- Evaluamos performance sobre el corpus de evaluación (que no vimos nunca antes!)

Iris Dataset

- Problema de clasificación multiclase: lo transformamos en tres problemas uno-contra-todos (¿es setosa? ¿es versicolor? ¿es virginica?). Elegimos la más votada, o la más "segura"(según el método)



Medidas de evaluación

- Accuracy (Precisión)
  - Fracción de las instancias clasificadas correctamente
- Las usuales en el PLN son:
  - Precisión (Precision!)
    - Instancias positivas que se clasificaron correctamente / Total de Instancias Positivas
    - TP / (TP + FP) [TP= True Positives, FP = False Positives]
  - Exhaustividad (*recall*)
    - Fracción de las instancias positivas que se clasificaron correctamente / Fracción de las instancias que son positivas
    - TP/(TP + FN) [FN = False Negatives]
  - Medida-F
    - Media armónica entre Precisión y Recall: 2*Precision*Recall/ (Precision + Recall)

Iris Dataset

Medimos accuracy sobre el corpus de entrenamiento: 0.83

- Matriz de confusión:

```
[[41  1  0]
 [ 1 25 13]
 [ 0  4 27]]
```

This matrix includes, in row i and column *j* the number of instances of class *i* that were predicted to be in class *j*.

Iris Dataset

~~Medimos accuracy sobre el corpus de entrenamiento: 0.83~~ Error!

Medimos accuracy sobre el corpus de evaluación: 0.68

Matriz de confusión:

```
[[ 8  0  0]
 [ 0  3  8]
 [ 0  4 15]]
```

This matrix includes, in row i and column *j* the number of instances of class *i* that were predicted to be in class *j*.

Ejemplo

- Matriz de confusión

	setosa	versicolor	virginica
setosa	8	0	0
versicolor	0	3	8
virginica	0	4	15

- Reporte de clasificación (P/R/F)
  - Setosa: 1.0/1.0/1.0 (support=8)
  - Versicolor: 0.43 (3/7)/0.27 (3/11)/0.33 (support=11)
  - Virginica: 0.65 / 0.79 /0.71 (support=19)
  - Promedio: 0.66 / 0.68 / 0.66

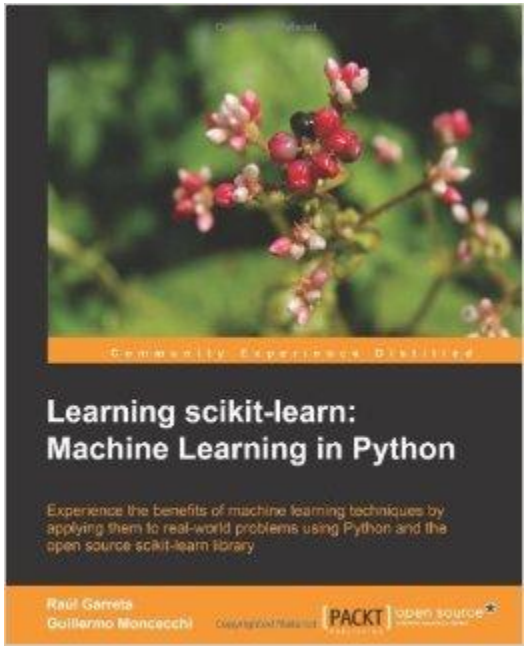
Mejoras

- ¿Cómo podríamos mejorar?
  - Cambiar el método
  - Elegir los atributos (son muy pocos…)
  - Ajustar parámetros del modelo

Ejemplos

http://gmonce.github.io/scikit-learn-book/

(Capítulos 1 y 2)



Clasificación secuencial

- En lugar de un objeto, tengo una secuencia de objetos, y a cada uno quiero clasificarlo.
    - La función de clasificación tiene secuencias como dominio y como codominio
- Ejemplos en PLN:
    - POS-tagging
    - Reconocimiento de Entidades con Nombre
    - Chunking
    - … etc!

Clasificación secuencial

- Aproximación computacional al reconocimiento de *spans* de texto:
  - BIO:
    - El primer elemento del span tiene clase B
    - Los restantes en el span tienen clase I
    - El resto es O (Other)

Clasificación secuencial

- Aproximación computacional al reconocimiento de *spans* de texto:
  - BIO (NER– CoNLL 2002):
    - Wolff /B-PER ,/O currently/O a/O journalist/O in/O Argentina/B-LOC ,/O played/O with/O ...

Clasificación secuencial

- Aproximación computacional al reconocimiento de *spans* de texto:
  - BIO (Hedge Cues – CoNLL 2010):

Token	Surface Form	Hedge
1	Cotransfection	O
2	studies	O
3	with	O
4	this	O
5	cDNA	O
6	indicate	B
7	that	I
8	it	O
9	can	B
10	repress	O
11	basal	O
12	promoter	O
13	activity	O
14	.	O

Clasificación secuencial

- Aproximación computacional al reconocimiento de *spans* de texto:
  - FOL(Alcance de Hedge Cues – CoNLL 2010):

Token	Word	Lemma	POS	Hedge	Scope
1	This	This	DT	O	O
2	finding	finding	NN	O	O
3	suggests	suggest	VBZ	O	O
4	that	that	IN	O	O
5	the	the	DT	O	F
6	BZLF1	BZLF1	NN	O	O
7	promoter	promoter	NN	O	O
8	may	may	MD	B	O
9	be	be	VB	O	O
10	regulated	regulate	VBN	O	O
11	by	by	IN	O	O
12	the	the	DT	O	O
13	degree	degree	NN	O	O
14	of	of	IN	O	O
15	squamous	squamous	JJ	O	O
16	differentiation	differentiation	NN	O	L
17	.	.	.	O	O

Clasificación secuencial

- Medidas de Evaluación
  - Las medidas son las mismas, pero…
    - … ¿qué se considera un TP?
    - En general, se asume *exact match*, aunque hay variantes.

Referencias

- Steven Abney, Semisupervised Learning for Computational Linguistics
- C.Manning, Introduction to Information Retrieval
- R. Garreta, G. Moncecchi, Learning Scikit-learn: Machine Learning in Python