Práctica 4: Clasificación semi-supervisada

Clasificación no convencional(CNC)

Máster Universitario en Inteligencia Computacional e Internet de las Cosas

Universidad de Córdoba, EPSC

2022/2023

UNIVERSIDAD
D
CÓRDOBA

Autor:

Antonio Gómez Giménez (i72gogia@uco.es)

Índice:

1. Introducción:

Para la realización de dicha práctica, se pretendió realizar una comparativa entre distintos tipos de dataset diferentes para ver el funcionamiento de los mismos frente a un algoritmo de aprendizaje semi-supervisado. Para realizar distintos tipos de pruebas, se repitió el mismo experimento, probando cuánto afecta la falta de etiquetas en cada uno de los dataset, probando distintas cantidades.

Por último, también se pretende comparar, cuánto es la mejora (en el caso de que exista), entre aplicar un algoritmo de aprendizaje semi-supervisado con datos sin etiquetar frente al algoritmo básico eliminando las instancias que no contienen etiquetas.

2. Experimentos:

Para poder realizar dichos experimentos, se dividió en tres partes, donde cada parte es un dataset distinto:

2.1. Primera parte:

El dataset utilizado en esta parte es el dataset iris que consta de 100 instancias y 4 atributos, con un total de 3 clases, siendo un dataset relativamente pequeño.

Una vez escogido y cargado el dataset, se creó una lista que contiene los porcentajes de etiquetas que vamos a eliminar siendo de 10%, 30%, 60% y 90%.

```
[ ]  #lista con los % de etiquetas a eliminar
```

```
[17] listaPorcentajes = [0.1,0.3,0.6,0.9]
```

Posteriormente se aplicó el siguiente código:

```
for i in listaPorcentajes:
  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)#cargo el dataset en las variables
  print(("number of instances -> " + str(X_train.shape[0])))#instancias
  print(("number of attributes -> " + str(X_train.shape[1])))#atributos
  print("\n")

  #código para eliminar % de datos
  rng = np.random.RandomState(42)
  random_unlabeled_points = rng.rand(y_train.shape[0]) < i
  y_train[random_unlabeled_points] = -1
  y_train_sin_patrones = list(filter(lambda x: x != -1,y_train))
  x_train_sin_patrones = []
  for w in range(len(X_train)):
    if not random_unlabeled_points[w]:
      x_train_sin_patrones.append(X_train[w])
  #modelo de autoentrenamiento
#-------------------------------------------------------
  svc = SVC(probability=True, gamma="auto")
  self_training_model = SelfTrainingClassifier(svc)
  self_training_model.fit(X_train, y_train)
  #resultados obtenidos de dicho modelo
  pred = self_training_model.predict(X_test)
  cm = confusion_matrix(y_test,pred)
  print("Matriz de confusión para", i, "aplicando semi-supervisado")
  print(cm)
  print("Accuracy score ->", accuracy_score(y_test, pred))
  print("\n")
#-------------------------------------------------------
  svc = SVC(probability=True, gamma="auto")
  svc.fit(x_train_sin_patrones, y_train_sin_patrones)
  #resultados obtenidos de dicho modelo
  pred = self_training_model.predict(X_test)
  cm = confusion_matrix(y_test,pred)
  print("Matriz de confusión para", i, "eliminando los datos sin etiqueta")
  print(cm)
  print("Accuracy score ->", accuracy_score(y_test, pred))
  print("\n")
```

En dicho código, primeramente se aplica el porcentaje especificado anteriormente para cada una de las combinaciones posibles, creando así un conjunto de datos con un x% de etiquetas faltantes. También se crea un conjunto nuevo donde se elimina dicho porcentaje de patrones para poder realizar la comparativa entre SVM y SVM pero aplicando self_training(para el conjunto de datos que contiene instancias sin etiquetar).

Finalmente tras realizar el entrenamiento con cada uno, se realiza la predicción y se obtiene la matriz de confusión para cada uno y el valor de Accuracy.

2.2. Segunda parte:

El dataset utilizado en esta parte es el dataset wine que consta de 119 instancias y 13 atributos, con un total de 3 clases, siendo un dataset relativamente pequeño.

Una vez escogido y cargado el dataset, se crea, igual que en el caso anterior, una lista que contiene los porcentajes de etiquetas que vamos a eliminar siendo de 10%, 30%, 60% y 90%.

El resto del código es similar al explicado en la primera parte.

2.3. Tercera parte:

El dataset utilizado en esta parte es un dataset artificial creado a través de la función moon, que consta de 13400 instancias y 2 atributos, con un total de 2 clases, siendo un dataset de un tamaño mayor a los anteriores. Dicho dataset se creó añadiendo un ruido del 50% a los datos.

Una vez escogido y cargado el dataset, se crea, igual que en el caso anterior, una lista que contiene los porcentajes de etiquetas que vamos a eliminar siendo de 10%, 30%, 60% y 90%.

El resto del código es similar al explicado en la primera parte.

3. Resultados y Conclusiones:

Los resultados obtenidos fueron los siguientes:

Dataset iris:

```
Matriz de confusión para 0.1 aplicando semi-supervisado
[[19  0  0]
 [ 0 15  0]
 [ 0  0 16]]
Accuracy score -> 1.0


Matriz de confusión para 0.1 eliminando los datos sin etiqueta
[[19  0  0]
 [ 0 15  0]
 [ 0  0 16]]
Accuracy score -> 1.0


Matriz de confusión para 0.3 aplicando semi-supervisado
[[19  0  0]
 [ 0 15  0]
 [ 0  0 16]]
Accuracy score -> 1.0


Matriz de confusión para 0.3 eliminando los datos sin etiqueta
[[19  0  0]
 [ 0 15  0]
 [ 0  0 16]]
Accuracy score -> 1.0

Matriz de confusión para 0.6 aplicando semi-supervisado
[[19  0  0]
 [ 0 15  0]
 [ 0  1 15]]
Accuracy score -> 0.98


Matriz de confusión para 0.6 eliminando los datos sin etiqueta
[[19  0  0]
 [ 0 15  0]
 [ 0  1 15]]
Accuracy score -> 0.98


Matriz de confusión para 0.9 aplicando semi-supervisado
[[19  0  0]
 [ 0 15  0]
 [ 0  7  9]]
Accuracy score -> 0.86


Matriz de confusión para 0.9 eliminando los datos sin etiqueta
[[19  0  0]
 [ 0 15  0]
 [ 0  7  9]]
Accuracy score -> 0.86
```

Dataset wine:

```
Matriz de confusión para 0.1 aplicando semi-supervisado
[[ 1 19  0]
 [ 0 24  0]
 [ 0 15  0]]
Accuracy score -> 0.423728813559322


Matriz de confusión para 0.1 eliminando los datos sin etiqueta
[[ 1 19  0]
 [ 0 24  0]
 [ 0 15  0]]
Accuracy score -> 0.423728813559322



Matriz de confusión para 0.3 aplicando semi-supervisado
[[ 1 19  0]
 [ 0 24  0]
 [ 0 15  0]]
Accuracy score -> 0.423728813559322


Matriz de confusión para 0.3 eliminando los datos sin etiqueta
[[ 1 19  0]
 [ 0 24  0]
 [ 0 15  0]]
Accuracy score -> 0.423728813559322



Matriz de confusión para 0.3 aplicando semi-supervisado
[[ 1 19  0]
 [ 0 24  0]
 [ 0 15  0]]
Accuracy score -> 0.423728813559322


Matriz de confusión para 0.3 eliminando los datos sin etiqueta
[[ 1 19  0]
 [ 0 24  0]
 [ 0 15  0]]
Accuracy score -> 0.423728813559322



Matriz de confusión para 0.9 aplicando semi-supervisado
[[ 0 20  0]
 [ 0 24  0]
 [ 0 15  0]]
Accuracy score -> 0.4067796610169492


Matriz de confusión para 0.9 eliminando los datos sin etiqueta
[[ 0 20  0]
 [ 0 24  0]
 [ 0 15  0]]
Accuracy score -> 0.4067796610169492
```

Dataset moons:

```
Matriz de confusión para 0.1 aplicando semi-supervisado
[[2774  597]
 [ 548 2681]]
Accuracy score -> 0.8265151515151515

Matriz de confusión para 0.1 eliminando los datos sin etiqueta
[[2774  597]
 [ 548 2681]]
Accuracy score -> 0.8265151515151515

Matriz de confusión para 0.3 aplicando semi-supervisado
[[2775  596]
 [ 549 2680]]
Accuracy score -> 0.8265151515151515

Matriz de confusión para 0.3 eliminando los datos sin etiqueta
[[2775  596]
 [ 549 2680]]
Accuracy score -> 0.8265151515151515

Matriz de confusión para 0.6 aplicando semi-supervisado
[[2798  573]
 [ 577 2652]]
Accuracy score -> 0.8257575757575758

Matriz de confusión para 0.6 eliminando los datos sin etiqueta
[[2798  573]
 [ 577 2652]]
Accuracy score -> 0.8257575757575758

Matriz de confusión para 0.9 aplicando semi-supervisado
[[2867  504]
 [ 623 2606]]
Accuracy score -> 0.8292424242424242

Matriz de confusión para 0.9 eliminando los datos sin etiqueta
[[2867  504]
 [ 623 2606]]
Accuracy score -> 0.8292424242424242
```

Se crearon las siguientes tablas donde se escoge solo el accuracy score, para tener una mejor comparativa entre los distintos resultados:

Dataset iris:

Aplicando semi-supervisado				
Patrones sin etiquetar	10%	30%	60%	90%
Accuracy Score	1.0	1.0	0.98	0.86

Sin aplicar semi-supervisado				
Patrones sin etiquetar (eliminados)	10%	30%	60%	90%
Accuracy Score	1.0	1.0	0.98	0.86

Dataset wine:

Aplicando semi-supervisado				
Patrones sin etiquetar	10%	30%	60%	90%
Accuracy Score	0.4237	0.4237	0.4237	0.4067

Sin aplicar semi-supervisado				
Patrones sin etiquetar (eliminados)	10%	30%	60%	90%
Accuracy Score	0.4237	0.4237	0.4237	0.4067

Dataset moons:

Aplicando semi-supervisado				
Patrones sin etiquetar	10%	30%	60%	90%
Accuracy Score	0.8265	0.8265	0.8257	0.8292

Sin aplicar semi-supervisado				
Patrones sin etiquetar (eliminados)	10%	30%	60%	90%
Accuracy Score	0.8265	0.8265	0.8257	0.8292

Como podemos apreciar con los datos observados de los distintos dataset, a medida que el % de patrones sin etiquetar aumenta, el accuracy score disminuye ya que con el clasificador self_training se introduce ruido al modelo y con svm, se va reduciendo el número de patrones al eliminar aquellos sin etiqueta.

También nos damos cuenta que estos dataset son muy sencillos ya que, da igual usar svm o self_training, ya que el hecho de usar más instancias (sin etiquetar), no está mejorando el resultado ya que el modelo converge con muy pocas instancias y prácticamente no es necesario aplicar clasificadores de semi-supervisado, para aumentar así la cantidad de instancias.