Universidad del País Vasco Euskal Herriko Unibertsitatea

Sistemas de Ayuda a la Decisión

Práctica 4

Índice

1. Objetivos y Descripción de los contenidos

Esta práctica tiene como objetivo adquirir las competencias que serán evaluadas en el proyecto. Por lo tanto esta tarea se enmarca dentro de la evaluación continua pero no dentro de los hitos evaluables.

Las **competencias** que el alumno deberá haber adquirido tras realizar la práctica son:

1. Capacidad para desarrollar prototipos de clasificación empleando el algoritmo: Naive Bayes.
2. Capacidad para adaptar al empleo de Naive Bayes las plantillas que el propio alumno ha generado para otros algoritmos.
3. Capacidad para evaluar la bonanza de cada modelo y seleccionar el mejor.
4. Capacidad para dado un conjunto de instancias *nuevas* se clasifiquen estas empleando el mejor modelo seleccionado.

2. Materiales disponibles

El siguiente material será empleado para el desarrollo de las tareas que se proponen y que tiene como objetivo alcanzar las competencias enumeradas anteriormente.

1. Datos: Se podrán encontrar en eGela dos juegos de datos (iris.csv y SantanderTraHalfHalf.csv). Con cada uno se realizará una experimentación completa.

2. Transparencias sobre el algoritmo Naive Bayes.
3. Documentación sobre el algoritmo en scikit-learn en sus dos versiones:
   - para valores categoriales (https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.CategoricalNB.html#sklearn.naive_bayes.CategoricalNB)
   - para valores contínuos (https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html#sklearn.naive_bayes.GaussianNB).

   el alpha o (Additive (Laplace/Lidstone) smoothing parameter) es la unidad (o lo que se estime) que hay que añadir a aquellos atributos que no aparecen con una determinada clase. Así evitamos que esa probabilidad a 0 arrastre todo el resultado a 0. El min_categories podéis dejarlo con el default default=None porque el mismo se encarga de calcularlo. Este parámetro se aplica cuando sabemos positivamente que un determinado atributo puede tomar X valores. De forma que si en la muestra de entrenamiento solo aparecen $X-2$ valores, generara 2 valores sintéticos adicionales a los que les asignará el alpha 1. Si es none mirá en su conjunto para ambas clases los valores posibles y si para alguna clase faltase algún valor se encarga la máquina misma de añadirlo.

3. Tareas

Para obtener los objetivos buscados en este proyecto se proponen las siguientes **tareas**:

1. Experimento Naïve Bayes:
   - Adaptar alguna de las plantillas del alumno y modificarla para que se ejecute el algoritmo Naive Bayes.
   - Probar dos versiones:
     - Convertir los atributos continuos en discretos (discretizar con https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.KBinsDiscretizer.html)
     - Probar la versión mixed-naive-bayes https://pypi.org/project/mixed-naive-bayes/
   - Evaluar ambos los modelos y generar automáticamente un .csv con las figuras de mérito Accuracy, Precision, Recall y F-score para ambos. Almacenar los modelos empleando pickle.
   - Generar un programa .py con cada versión.
   - Seleccionar el modelo que mejores resultados obtenga. Recordad que normalmente es difícil comparar modelos en base a precisión, recall, porque suele pasar que el que es mejor en recall, es peor en precisión,

Combinación	Precisión	Recall	F_score(Mac/Mic/Avg/None)
Discretizando	0.XX	0.XX	0.XX
mixedNaive	0.XX	0.XX	0.XX

y la accuracy como vimos suele estar sesgada hacia la clase mayoritaria. Así pues, la figura de mérito que se suele emplear es el f1_score (o f_score) que es una media entre la precisión y el recall. Sklearn nos permite obtener esta figura de mérito invocando: f1_score(y_true, y_pred)

- Si estamos realizando una tarea de clasificación binaria: f1_score(y_true, y_pred average=None), porque solo tenemos dos clases.
- Si estamos realizando una clasificación multiclase (en el ejemplo del iris):
  - Macro: f1_score(y_true, y_pred, average='macro')
  - Micro: f1_score(y_true, y_pred, average='micro')
  - Weighted: f1_score(y_true, y_pred, average='weighted')

  Consultar la documentación: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

  Consultar los apuntes de clase sobre figuras de mérito.

- Regenerar el mejor modelo según la figura de mérito seleccionada (volver a ejecutar el programa que genera el modelo pero con los hiperparametros que mejores resultados nos han dado y salvarlo empleando pickle (Consultar al final del documento).
- Dada una nueva muestra con nuevas instancias que no están clasificadas, obtener la clasificación de las mismas empleando el modelo salvado (Consultar al final del documento).

4. Receta para salvar modelos

Receta para salvar un modelo en disco:

import pickle

nombreModel = "nombreParAlmacenar.sav"

saved_model = pickle.dump(**clf**, open(nombreModel,'wb')) **clf** o como se llame la clase que contiene el modelo

print('Modelo guardado correctamente empleando Pickle')

Receta para recargar un modelo del disco para poder clasificar nuevas instancias:

import pickle

X_nuevo = pd.read_csv(iFile) contendrá instancias nuevas sin la clase p.q. eso es lo que queremos predecir

nombreModel = "nombreParAlmacenar.sav"

clf = pickle.load(open(nombreModel, 'rb'))

resultado = clf.predict(X_nuevo)

Referencias

[1] Dataset Santander Customer Satisfaction: Solo se ha empleado una parte de estos datos. Es importante decir que se emplearán exclusivamente para hacer la práctica y que el alumno se compromete a no ponerlos en su github. Si el alumno quisiera poner su código en lo referente a los datos deberá nombrar la fuente de los datos para facilitar su replicabilidad, pero no podrá subir los datos dado que su uso está restringido:

'Data' means the Data or Datasets linked from the Competition Website for the purpose of use by Participants in the Competition. For the avoidance of doubt, Data is deemed for the purpose of these Competition Rules to include any prototype or executable code provided to Participants by Kaggle or Competition Sponsor via the Website. Participants must use the Data only as permitted by these Competition Rules and any associated data use rules specified on the Competition Website.

Unless otherwise permitted by the terms of the Competition Website, Participants must use the Data solely for the purpose and duration of the Competition, including but not limited to reading and learning from the Data, analyzing the Data, modifying the Data and generally preparing your Submission and any underlying models and participating in forum discussions on the Website. Participants agree to use suitable measures to prevent persons who have not formally agreed to these Competition Rules from gaining access to the Data and agree not to transmit, duplicate, publish, redistribute or otherwise provide or make available the Data to any party not participating in the Competition. Participants agree to notify Kaggle immediately upon learning of any possible unauthorized transmission or unauthorized access of the Data and agree to work with Kaggle to rectify any unauthorized transmission. Participants agree that participation in the Competition shall not be construed as having or being granted a license (expressly, by implication, estoppel, or otherwise) under, or any right of ownership in, any of the Data.

[2] Iris Dataset UCI Machine Learning Repository https://archive.ics.uci.edu/ml/datasets/iris

[3] Competencias asociadas de la tarea: Se han tomado como referencia los apuntes de SAD 2020-2021 (fuente: Alicia Pérez) para así coordinar que la práctica y las competencias a obtener sea lo más similar posible a lo solicitado en años anteriores.