SERIES TEMPORALES Y MINERÍA DE FLUJOS DE DATOS

MÁSTER EN CIENCIA DE DATOS E INGENIERÍA DE COMPUTADORES

Evaluación teórica de Minería de Flujo de Datos

Autora
Lidia Sánchez Mérida

Curso 2021 - 2022

Granada, Abril de 2022

a) Explica en qué consisten los diferentes modos de evaluación/validación para clasificación en flujos de datos (máximo 1 página).

Existen diversas técnicas aplicables al estudio de la capacidad de generalización de un modelo previamente entrenado. Una de las más tradicionalmente utilizadas es el ***holdout***, consistente en dividir el conjunto de instancias completo en dos subconjuntos, uno para entrenamiento con el que se genera el clasificador, y un segundo para test con el que evaluar su bondad de predicción. Dependiendo del problema que se desee resolver, se puede realizar una partición personalizada de acuerdo al contexto asociado o a los objetivos que se persiguen. En caso de no disponer de esta información, se puede optar por generar una partición tradicional y aleatoria para dividir el conjunto de datos en 80% o 90% para entrenamiento, mientras que la parte restante se reserva para test.

Otra de las técnicas más populares para evaluar modelos en clasificación es la conocida como ***cross validation***. Si bien existen diferentes versiones de esta metodología, en todas ellas se comienza dividiendo el conjunto de datos total en entrenamiento y test aplicando, por ejemplo, *Holdout*.

- ***K-cross validation***. Se trata de una de las variantes más utilizadas consistente en construir *K* modelos particionando el conjunto de entrenamiento en un primer subconjunto con el que generar el clasificador y un segundo para validar su rendimiento. La precisión final se suele calcular como la media de las tasas de aciertos de los *K* modelos.
- ***Leave one out cross validation.*** En esta versión se persigue la generación de *N* clasificadores a partir del conjunto de entrenamiento completo a excepción de una única instancia con la que realizar la fase de validación.
- ***Time series cross validation***. Si existen dependencias temporales en el conjunto de datos es importante respetar la cronología asociada para el entrenamiento y la correcta validación de los clasificadores. Nuevamente, existen diferentes enfoques con los que modelar series temporales aunque el más sencillo consiste en aplicar *holdout* sin aleatoriedad para conseguir un conjunto de entrenamiento con una temporalidad anterior al de validación. Así, el modelo puede aprender los patrones existentes en los datos a partir de información ubicada en el pasado para luego ser evaluado a partir de muestras situadas en momentos futuros.

Un tercer proceso de evaluación de modelos en problemas de clasificación se denomina ***Interleaved Test-Then-Train*** que consiste en utilizar las instancias del flujo de datos entrante para validar el clasificador antes de emplearlas en entrenamiento. El cálculo de la precisión se realiza de manera incremental conforme aparecen nuevos ejemplos en el tiempo. La principal ventaja con respecto a las técnicas anteriores es que se aprovecha al máximo el conjunto de datos puesto que todas las muestras se utilizan tanto para verificar la bondad del modelo, como para mejorar su capacidad de predicción. Sin embargo, durante las etapas iniciales de entrenamiento el porcentaje de errores de clasificación será considerablemente más elevado, ya que el modelo no dispondrá de la información necesaria para identificar correctamente las nuevas instancias.

Finalmente, una variante de esta última técnica es la conocida como ***Prequential***, cuya validación se aplica únicamente con los datos más recientes. Para ello se puede utilizar una ventana de datos temporal adaptable al progreso del entrenamiento del modelo y al número máximo de instancias prefijado, o mediante el cálculo de factores de decaimiento que replican este mismo comportamiento aunque invirtiendo una menor cantidad de recursos de almacenamiento de datos.

b) Describe tres algoritmos de clasificación en flujos de datos (máximo 2 páginas).

Los dos primeros algoritmos de clasificación seleccionados para este apartado pertenecen a la familia de técnicas *Hoeffding,* capaces de generar árboles de decisión a partir de un flujo de datos. El primero de ellos es conocido como **VFDT** (*Very Fast Decision Tree*), que se caracteriza por generar varios nodos a partir de uno ya existente utilizando el conjunto de ejemplos disponibles. Existen diferentes métricas que permiten calcular si existe suficiente evidencia estadística como para continuar la exploración de una rama del árbol. Una de las más popularmente utilizadas es la **ganancia de información**, cuya formulación depende de la naturaleza del flujo de datos entrante. Si se trata de información estática, entonces se analiza el conjunto de entrenamiento completo verificando todos los atributos disponibles para tomar una decisión respecto a la expansión de un nodo. Por el contrario, si el problema es incremental se puede aplicar la **cota de *Hoeffding*** para seleccionar el atributo con el que considerar la división de un nodo. En este caso se compara la ganancia de información obtenida de cada característica a partir de la diferencia entre la media real y la estimada tras un conjunto de N observaciones independientes con el objetivo de elegir aquella que presente un mayor valor.

El algoritmo VFDT dispone de un procedimiento ***Test-Then-Train*** que le permite validar el clasificador con cada nueva instancia que aparezca en el flujo de datos, previo a su empleo en la fase de entrenamiento. En esta etapa, si la categoría del nuevo ejemplo es conocida, se propaga desde el nodo raíz hacia la rama con mayor probabilidad de acierto para posteriormente evaluar si se continúa expandiendo el árbol a partir del cálculo de la ganancia de información incluyendo la nueva instancia. A pesar de ser un enfoque considerablemente más eficiente que los árboles de decisión tradicionales, presenta dos principales desventajas, y es que esta técnica no es capaz de procesar **atributos continuos** ni tampoco detectar **cambios de concepto** por su imposibilidad de almacenar datos pasados.

VFDTc es una variante del algoritmo anterior que surge para intentar paliar las limitaciones mencionadas previamente. Esta técnica construye árboles binarios introduciendo un nuevo concepto denominado **puntos de corte**, cuyo objetivo consiste en representar a cada uno de los **atributos continuos** mediante valores concretos seleccionados a partir de sus respectivos rangos de datos. Posteriormente, estos elementos participan en el cálculo de la **ganancia de información** de cada característica con la que evaluar la calidad de la expansión de un nodo estableciendo dicho atributo como criterio, tal y como sucede en la técnica original.

En caso de que el conjunto de datos disponga de **atributos discretos**, cada nodo podrá disponer de un número de ramificaciones igual a la cantidad de clases diferentes, más una que por defecto representa una etiqueta desconocida. Aquellas instancias cuyas clases no se conozcan pertenecerán a esta última rama.

El procedimiento de clasificación asociado a VFDTc consiste en asociar la clase de los ejemplos más representativos para cada nueva instancia. A diferencia del algoritmo original, en cada nodo hoja se aplica un **clasificador entrenado con *Naive Bayes*** para considerar tanto la distribución de clases de los ejemplos como la probabilidad de pertenencia según los atributos del conjunto de datos. Si bien este método es capaz de trabajar con atributos de distinta naturaleza, en esta última etapa es necesario

discretizar los valores continuos asignando categorías a los intervalos de datos que se generan para cada característica numérica.

Finalmente el tercer algoritmo seleccionado para su estudio se denomina **FACIL** (*Fast and Adaptive Classifier by Incremental Learning*). Se trata de una técnica de aprendizaje incremental basada en **reglas de decisión** que se encuentra específicamente diseñada para la clasificación de flujos de datos con características numéricas. Cada regla dispone de un conjunto de ventanas deslizantes que contienen ejemplos insertados dinámicamente conforme aparecen en el flujo de información. Uno de los inconvenientes que solventa este algoritmo es el estado de **inconsistencia de una regla**, que se produce cuando dispone de un conjunto de ejemplos pertenecientes a diferentes clases. El algoritmo *FACIL* permite que cada regla cubra una muestra relativa a cada una de las categorías existentes, es decir, si consideramos por ejemplo un problema de clasificación binario, una regla de decisión puede tener tanto instancias positivas como negativas siempre y cuando al menos exista **un ejemplo positivo por cada negativo cubierto**. Con la aplicación de esta estrategia evita la regeneración continua de aquellas reglas que dejen de ser consistentes, lo que produce un uso más eficiente de los recursos disponibles, a diferencia de otros algoritmos similares como el *AQ11-PM*.

Cuando un nuevo ejemplo aparece en el flujo de datos se llevan a cabo las siguientes comprobaciones para actualizar el modelo y su conjunto de reglas asociado:

- **Cobertura positiva**. La nueva instancia es cubierta por una o varias reglas que representan la clase con la que se encuentra etiquetada. En este caso se actualizará la regla cuya generalización no supere el máximo crecimiento permitido, priorizando así la incorporación del nuevo ejemplo a aquella regla cuyo número de cambios sea mínimo.
- **Cobertura negativa**. Representa la situación contraria a la anterior en la que la nueva instancia es representada por una o varias reglas con diferente categoría. Para las reglas candidatas se calcula la intersección entre las originales y las nuevas que se generan a partir de la inserción de la nueva muestra. La primera regla cuyo resultado no sea el conjunto vacío es la elegida para añadir el nuevo ejemplo a su conjunto de instancias. A continuación se verifica su nivel de **pureza** en base a la proporción de ejemplos positivos y negativos, si tomamos como ejemplo un problema de clasificación binaria. Si el valor resultante es inferior a un umbral determinado, entonces el algoritmo genera nuevas reglas consistentes a partir del conjunto de instancias disponibles y clasifica la regla como ***dudosa***, de modo que no se tenga en consideración en ninguno de los tres procesos durante futuras iteraciones.
- **Nueva descripción**. El nuevo ejemplo no es contemplado por ninguna de las reglas del conjunto actual. Si existe una regla candidata pero no ha sido seleccionada en ninguno de los dos procesos anteriores, entonces se genera una nueva que para cubrir a la nueva instancia temporalmente.

La etapa de predicción sobre un conjunto de test consiste en encontrar aquella regla ***no dudosa*** que cubra cada uno de los nuevos ejemplos. Dependiendo del estado de la regla y de la casuística explicada anteriormente, si existe una regla que cubre un ejemplo y es consistente se le asigna la su etiqueta vinculada. En caso de ser inconsistente se clasifica a partir de la categoría del vecino más cercano dentro del conjunto de ejemplos de su ventana. Por el contrario, si no existe ninguna regla que cubra la nueva instancia se etiqueta con la categoría de la regla *no dudosa* que menor crecimiento produce al insertar el ejemplo de test.

c) Explica en qué consiste el problema de concept drift y qué técnicas conoces para resolverlo en clasificación, clustering y patrones frecuentes (máximo 2 páginas).

El término ***concept drift*** hace referencia a la posibilidad de que la información proporcionada presente modificaciones notables con respecto a los datos anteriores o aparezcan nuevas características no contempladas, lo que provoca la depreciación de los modelos construidos. Un cambio de concepto es **real** si altera la frontera de decisión con la que se dividen los ejemplos de las diferentes clases, o **virtual** en caso de que únicamente se modifique la distribución de los datos. Algunas de las razones explicativas de este fenómeno se detallan a continuación:

- La aparición de **ruido** en los nuevos ejemplos recibidos.
- **Factores** del contexto del problema o de la información que comienzan a influir considerablemente en el modelado realizado previamente.
- Variación de los **atributos** analizados hasta el momento o la aparición de nuevas características que no se encuentran incluidas en el clasificador.
- Alteraciones en la **estacionalidad** del flujo de datos que producen cambios en la periodicidad con la que suceden los diferentes patrones de información.

En **problemas de clasificación** se han desarrollado diferentes técnicas que permiten modelar flujos de datos y adaptar los modelos a los cambios de concepto que surjan en el tiempo. En primer lugar disponemos del **aprendizaje online** cuyos algoritmos son capaces de actualizar los parámetros del modelo de forma simultánea a la entrada de nuevos datos. Sin embargo, existen ciertas condiciones a cumplir por parte de aquellas técnicas que pertenezcan a esta categoría, como que cada instancia solo se puede emplear una única vez en la fase de entrenamiento, la independencia de los recursos necesarios para construir el modelo y la posibilidad de interrumpir el aprendizaje sin repercutir negativamente en el rendimiento del clasificador.

Un segundo enfoque orientado a abordar esta problemática son los algoritmos que incorporan **ventanas deslizantes** con las que considerar únicamente el subconjunto de datos más recientes. Así, se construye el modelo en base a la información más actualizada de modo que su representación sea lo más precisa posible conforme al flujo de datos vigente. No obstante, el tamaño de la ventana influye considerablemente en el rendimiento de esta técnica, puesto que si se trata de un valor demasiado inferior el subconjunto de datos no es suficientemente representativo ni voluminoso para que el modelo converja a una solución.

Los ***ensembles*** representan una tercera técnica consistente en combinar el conocimiento intrínseco a un conjunto de clasificadores para mejorar la capacidad de predicción sobre un problema determinado. El éxito de este método reside en la diversidad tanto de los diferentes modelos que participan, como en el conjunto de atributos, ejemplos o el contexto utilizados durante las etapas de entrenamiento. Existen diferentes enfoques en los que se contempla mantener un número fijo de clasificadores, mientras que otros se generan nuevos modelos al detectar un cambio de concepto o para reemplazar aquellos clasificadores con peores tasas de aciertos. No obstante, algunas técnicas mantienen los peores modelos obtenidos para almacenar los patrones detectados en el pasado en caso de que sean útiles para modelar los datos futuros o re-aparezcan en el flujo de datos de entrada. Finalmente se pueden aplicar

distintas técnicas para combinar las respuestas de los diferentes modelos, como asignar la clase mayoritaria o un voto ponderado en función de ciertos criterios, como la calidad de los clasificadores.

En **problemas de agrupamiento** cuyo objetivo reside en analizar y describir los datos proporcionados en lugar de generar predicciones, destaca un primer algoritmo denominado ***CluStream*** que se caracteriza por dos enfoques de funcionamiento. El primero es *online* en el que almacena datos estadísticos como *microclusters*, así como información temporal sobre las instancias previamente estudiadas para el modelado del flujo entrante. Cuando aparece un nuevo ejemplo, bien se puede agregar a alguna de las agrupaciones existentes si sus características son similares a las muestras que componen el *microcluster*, o se genera un nuevo grupo eliminando el más antiguo o fusionando dos *microclusters* para controlar la inversión de recursos de almacenamiento. El segundo enfoque disponible es *offline* para generar *clusters*, de forma similar a los algoritmos de agrupamiento tradicionales, aunque utilizando también las estadísticas obtenidas a partir de las instancias del flujo de datos.

DenStream es otra técnica perteneciente a esta segunda categoría de problemas de *clustering* que utiliza vectores de características para agrupar los diferentes ejemplos proporcionados. Para ello dispone de dos conjuntos de *clusters*, los *p-microclusters* que representan los grupos compuestos por las instancias previamente vistas, mientras que los *o-microclusters* contienen las muestras consideradas como *outliers*. Cuando aparece un nuevo dato, se intenta añadir al *p-microcluster* más cercano si se encuentra dentro de los límites de su frontera o, en su defecto, al *o-microcluster* más próximo.

Por último destacamos un tercer algoritmo conocido como ***ClusTree*** cuyas principales diferencias con respecto a las dos técnicas anteriores es su estructura jerárquica, que le permite tanto almacenar las características de los *clusters* generados a partir de las instancias analizadas, así como el almacenamiento de los propios ejemplos para aumentar su robustez frente a interrupciones durante el proceso de *clustering*, además de su capacidad para adaptar su funcionamiento a diferentes volúmenes de datos entrantes.

La detección de **patrones frecuentes** a partir de un flujo de datos entrante todavía representa una área de investigación abierta debido a los diferentes inconvenientes que presentan las técnicas desarrolladas hasta el momento. Dos de los primeros enfoques consisten en extraer *ítems* frecuentes a partir de un umbral de frecuencia o un conjunto de ellos mediante ventanas deslizantes, aunque únicamente pueden tratar con atributos categóricos. Una de las aproximaciones más utilizadas para identificar patrones de información son las **reglas de asociación** cuya metodología comienza con la extracción de *itemsets* frecuentes a partir de un flujo de información *online* para posteriormente generar las reglas de manera *offline.* Como consecuencia surge un requisito fundamental relativo al acceso al conjunto de datos completo para efectuar la última fase de este procedimiento, lo que supone una contradicción con el significado intrínseco de un flujo de datos *online*. Adicionalmente se mantiene el inconveniente de la imposibilidad de trabajar con atributos numéricos, además del enorme coste computacional que supone la extracción de reglas de asociación, siendo prácticamente inviable su aplicación sobre conjuntos masivos de información. Para solventar estas limitaciones surge un nuevo enfoque conocido como ***Association Stream Mining***, que permite extraer asociaciones entre los atributos de un flujo de datos *online,* con independencia de la tasa de llegada y la capacidad de detectar cambios de concepto para adaptar el modelo a las modificaciones que presenten los nuevos datos de entrada.