Memoria análisis conjunto de datos.

Julio Cabria, Alejandro Coloma, S. Xiao Fernández, Pablo Izaguirre [1]

1- Escuela Politécnica Superior - Universidad Autónoma de Madrid
C Francisco Tomás y Valiente nº 11 - España

Abstract. Presentamos un análisis del uso de distintos modelos de clasificación para determinar si el objeto medido por una serie de sensores de gas, temperatura y humedad se trata de vino o de un plátano. También creamos un clasificador para medir la presencia de estímulo o no en las mediciones. Alcanzamos a encontrar un modelo para la identificación del objeto con una exactitud de 83.33% haciendo uso de regresión logística y un modelo del estímulo con exactitud de 99.99% mediante un clasificador de Random Forest.

1 Introducción

En este proyecto vamos a identificar si ciertos gases pertenecen a un plátano o a vino a partir de mediciones de sensores que están localizados en una habitación. Se cuenta con un conjunto de datos compuesto por dos archivos: "HT_Sensor_metadata.dat" y "HT_Sensor_dataset.dat" [1]. El primero contiene información sobre la fecha en que se realizó la medición, qué se introdujo en la habitación para generar la medición (vino, plátano o nada) y el tiempo en el que comenzó y duró la medición. El segundo archivo incluye información sobre la medición en sí, incluyendo el tiempo, el valor de resistencia de cada uno de los 8 sensores MOX y las mediciones de temperatura y humedad. Los objetivos del análisis son clasificar las series de mediciones y determinar cuándo hay un estímulo presente y cuál es.

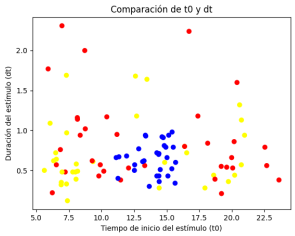
2 Análisis exploratorio de datos

Los datos del proyecto constan de dos archivos: "HT_Sensor_meta-data.dat" y "HT_Sensor_dataset.dat", como explicado anteriormente en la introducción. En el primero, cada fila representa una medición diferente y contiene información como el id de la medición, la fecha, la clase (vino, plátano o ninguno), el tiempo de inicio del estímulo (t0) y la duración del estímulo (dt). Hay un total de 100 mediciones en el dataset, con una distribución de las clases de: vino 35 mediciones, plátano 33 mediciones y ninguno 31 mediciones.

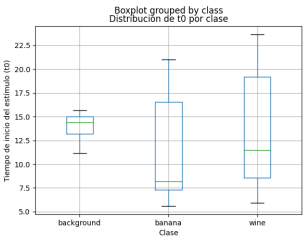
En 1a se muestra la relación entre el tiempo de inicio del estímulo (t0) y la duración del estímulo (dt). Cada punto en el gráfico representa una medición y su color indica a qué clase pertenece, podemos observar cómo se distribuyen los valores de t0 y dt para cada clase de alimento. Por ejemplo, podemos ver que los experimentos con vino tienden a tener un t0 más alto y un dt más corto, mientras que los experimentos con plátano tienden a tener un t0 más bajo y un dt más largo. Esta información es útil para entender cómo se relacionan los

valores de t0 y dt con la clase de alimento utilizada en cada experimento, y puede ser utilizada para mejorar nuestro modelo de clasificación.

Finalmente, en 1b, se muestra la distribución del tiempo de inicio del estímulo (t0) para cada clase. Se puede ver que hay una diferencia en la distribución de t0 entre las diferentes clases. Por ejemplo, se puede observar que la clase de vino tiene una distribución más centrada en torno al t0 medio, mientras que la clase de plátano tiene una distribución más dispersa con valores de t0 más extremos. Esto sugiere que la clase de vino tiene un patrón más predecible de tiempos de inicio de estímulo en comparación con la clase de plátano. Además, el gráfico también nos permite identificar si hay alguna tendencia o patrón en la distribución de t0 para cada clase, lo que puede proporcionar información valiosa para el análisis y la clasificación de los datos.



(a) Gráfico de dispersión para comparar el tiempo de inicio del estímulo (t0) y la duración del estímulo (dt).



(b) Gráfico de caja para mostrar la distribución del tiempo de inicio del estímulo (t0) para cada clase.

Fig. 1

En conclusión, estos gráficos nos ayudan a tener una mejor comprensión de los datos y a identificar patrones y tendencias en el conjunto de datos. Esto puede ser útil para mejorar la precisión de los modelos de clasificación y para tener una mejor comprensión de cómo se están utilizando los alimentos.

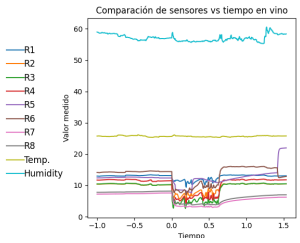
3 Atributos propuestos

Para lograr determinar cuándo está presente un estímulo y qué clase de alimento está siendo utilizado en cada experimento, se analiza el dataset de metadatos y utiliza la información de duración para cada identificador (id) para determinar la clase de cada experimento en el dataset principal. Se asigna el valor 1 si el tiempo de inicio del estímulo (t0) y la duración (dt) para ese id están comprendidos en el intervalo especificado en el dataset de metadatos, y el valor 0 en caso contrario. Además, se ha decidido fijar la clase a 0 para todos los experimentos con id igual o superior a 69, ya que estos han sido identificados como ruido de fondo en el dataset de metadatos.

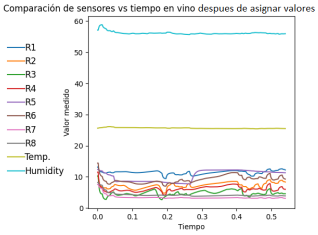
Una vez se han asignado las clases a cada experimento, se ha utilizado esta información para entrenar un modelo de clasificación y determinar cuándo está

presente un estímulo y qué clase de alimento se está utilizando. Para distinguir entre plátano y vino en el segundo dataset, se ha filtrado el dataset de estímulos y se han utilizado solo los datos que contienen un estímulo de clase plátano o vino, es decir, aquellos que tienen clase 1.

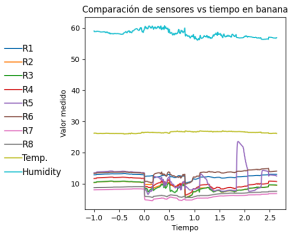
En las siguientes gráficas, se ve como en 2a y 2c hay mas ruido que en 2b y en 2d, ya que en la primera gráfica mencionada, se añaden todos los casos donde la clase es vino o plátano. Sin embargo, en la segunda, vemos como solo se añaden la que tienen clase 1, filtrando el ruido.



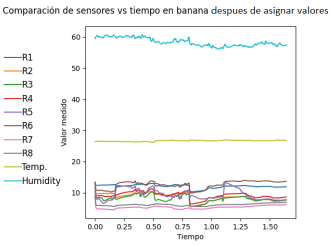
(a) Valor medido por los sensores para cada t.



(b) Valor de los sensores para el intervalo en el que se ha determinado que está presente el estímulo.



(c) Valor medido por los sensores para cada t



(d) Valor de los sensores para el intervalo en el que se ha determinado que está presente el estímulo

Fig. 2

A partir de estos datos, se ha generado uno nuevo que incluye los atributos media, mediana y desviación típica por experimento. Estos atributos se han escogido porque proporcionan información útil sobre la distribución de los datos y nos permiten comparar las mediciones de diferentes experimentos de manera más precisa. Además, la media y la mediana son estadísticas centrales que nos dan una idea del valor "típico" de las mediciones, mientras que la desviación típica nos indica la variabilidad de los datos. Al incluir estos atributos, podemos tener una mejor comprensión de cómo varían las mediciones entre diferentes experimentos y cómo se relacionan con la clase de alimento utilizada.

4 Modelos utilizados

En este experimento, se ha creado una clase llamada "Clasificador" que recibe como entrada un modelo de clasificación, como por ejemplo un árbol de decisión de sklearn, y el nombre de un archivo de datos para realizar predicciones. El modelo de clasificación utilizado debe tener una función "fit" para entrenar el modelo con un conjunto de atributos y clases (ambos en formato de array de numpy) y una función "predict" para hacer predicciones sobre un conjunto de atributos sin clases. La clase "Clasificador" se encarga de generar conjuntos de entrenamiento y prueba a partir del archivo de datos, entrena el modelo con la función "fit" y luego hace predicciones sobre el conjunto de prueba con la función "predict". Posteriormente, se evalúa la precisión del modelo comparando las predicciones con las clases reales del conjunto de prueba.

Por defecto, se ha utilizado una validación simple para evaluar el rendimiento del modelo, pero también se puede utilizar validación cruzada si se desea. Con esta implementación, es sencillo probar diferentes modelos de clasificación y comparar sus resultados de precisión sin tener que realizar más trabajo adicional.

Para determinar qué modelo es el mejor para discriminar, se pueden comparar los resultados de precisión de diferentes modelos y elegir el que tenga un rendimiento más alto. También se pueden ajustar los parámetros de cada modelo para mejorar su rendimiento y luego comparar nuevamente los resultados. Es importante tener en cuenta que los datos de prueba no deben utilizarse para entrenar los modelos, ya que esto puede dar lugar a resultados sesgados. Es necesario utilizar un conjunto de datos independiente para evaluar el rendimiento del modelo y así poder comparar de manera justa los diferentes modelos entrenado. Cuando entrenamos un modelo de clasificación, lo hacemos con el objetivo de que pueda hacer predicciones precisas sobre datos que no ha visto antes. Si utilizamos los datos de prueba para entrenar el modelo, estamos "filtrando" los datos a través del modelo y, por lo tanto, estamos proporcionando al modelo información adicional que podría no estar disponible en el mundo real. Esto puede hacer que el modelo se sobreajuste a los datos de prueba y no sea capaz de generalizar bien a datos nuevos, lo que puede llevar a resultados poco precisos en la práctica.

Por lo tanto, es importante tener un conjunto de datos de prueba independiente y no utilizarlo para entrenar el modelo. De esta manera, podemos evaluar el rendimiento del modelo de manera justa y determinar si es capaz de generalizar bien a datos nuevos. Esto es especialmente importante si queremos utilizar el modelo para hacer predicciones en el mundo real, ya que es probable que no tengamos acceso a todos los datos que se utilizaron para entrenar el modelo. Utilizando un conjunto de datos de prueba independiente, podemos evaluar el rendimiento del modelo en condiciones más cercanas a las del mundo real y obtener una idea más precisa de su rendimiento

4.1 Descripción del protocolo experimental

Hemos implementado la funcionalidad de generar los datasets y aplicar modelos sobre ellos en diferentes scripts de python. Los más importantes son:

- ***dataset_estimulo.py***: Genera un dataset cuya columna de clase toma el valor 1 si el estímulo está presente y 0 si no está presente para cada instante de tiempo.
- ***dataset_clasificar.py***: Genera otro dataset cuya columna de clase toma el valor del tipo de sustancia detectada por los sensores para cada experimento. Incluye atributos que identifican cada experimento por el valor de sus sensores (medios, máximos y mínimos, etc.)
- ***test_clasificadores.py***: Realiza un particionado de los datos en entrenamiento y test y aplica diferentes modelos a ambos datasets, mostrando por pantalla los porcentajes de error.

Hemos implementado también un archivo *Makefile*, que simplifica el proceso de ejecutar los scripts y gestiona el orden y las dependencias entre ellos:

- ***make download***: Descarga los datasets iniciales de la web [2] y los extrae para que puedan ser leídos por los scripts, limpiando los archivos comprimidos una vez termina la extracción.
- ***make convertir-a-csv***: Como su propio nombre indica, realiza una conversión de los conjuntos de datos iniciales, separados por espacios, a un formato separado por comas.
- ***make dataset-estimulo, dataset-clasificar*** y ***test-clasificadores***: Ejecutan los scripts *dataset_estimulo.py*, *dataset_clasificar.py* y *test_clasificadores.py* detallados anteriormente.
- ***make clean***: Para limpiar entre ejecuciones, elimina los datasets y otros archivos temporales.

Para llegar a los resultados expuestos en este trabajo, hemos iterado modificando variables como la presencia de ruido de fondo en *dataset_estimulo.py*, la elección de atributos en *dataset_clasificar.py* o los parámetros de los modelos y el particionado en *test_clasificadores.py*, ejecutando los tests y obteniendo porcentajes de error y gráficas para cada caso.

4.2 Estimación de parámetros

Para elegir los atributos adecuados se han hecho pruebas con los atributos propuestos, con máximos, mínimos, medias, medianas y desviaciones típicas. Las pruebas con mejor porcentaje de acierto eran las que utilizaban máximos y mínimos. Después se hicieron pruebas para eliminar sensores que pudieran añadir ruido. En la prueba con mejor porcentaje de acierto se eliminó la temperatura, la humedad, el sensor 1 y el sensor 5.

Clasificador/Atributos	max, min	max, min, σ, $\bar{x}$, $\tilde{x}$	σ, $\bar{x}$, $\tilde{x}$
Decision Tree	82.98%	79.47%	69.95%
KNeighbors	60.56%	66.74%	65.62%
Random Forest	81.20%	79.29%	75.77%
Gaussian NB	80.70%	75.98%	74.37%
Logistic Regression	80.13%	77.89%	78.80%

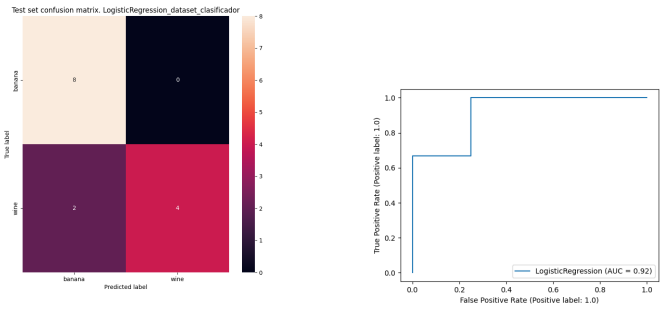
Table 1: Porcentaje de acierto usando todos los sensores, la temperatura y la humedad

Clasificador/Atributos	Max y min
DecisionTreeClassifier	80.56%
KNeighborsClassifier	74.79%
RandomForestClassifier	82.44%
GaussianNB	82.31%
LogisticRegression	83.22%

Table 2: Estos son los resultados del test después de eliminar temperatura, humedad y los sensores 1 y 5.

5 Resultados

Para clasificar los datos en función de las etiquetas vino y plátano, el modelo que hemos encontrado que nos da un mejor resultado es el que utiliza regresión logística, que nos da un porcentaje de aciertos de media sobre el conjunto de datos de prueba de 83.35%. Para analizar mejor el rendimiento de este modelo y como clasifica los diferentes datos hemos realizado las siguientes gráficas representando la matriz de confusión 3a y la curva ROC 3b.

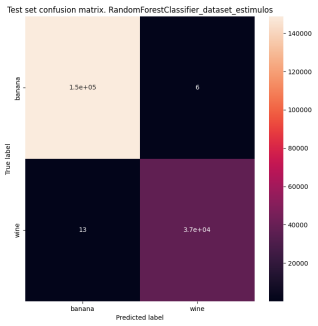


(a) Matriz de confusión. (b) Curva ROC.

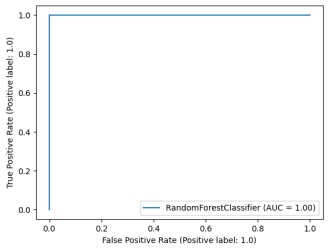
Fig. 3: Gráficas de análisis del clasificador entre vino y plátano usando regresión logística.

La matriz de confusión representa el número de predicciones de cada clase con respecto a su valor real y nos permite analizar la cantidad de verdaderos y falsos positivos y verdaderos y falsos positivos, permitiéndonos mucho mejor análisis que el que nos proporciona simplemente el porcentaje de aciertos. La curva ROC por otro lado, representa la razón entre verdaderos positivos y falsos positivos a medida que vamos cambiando el umbral de discriminación (el valor de la probabilidad a partir del cual clasificamos un dato como positivo). Un modelo perfecto sería aquel que se encuentra en la esquina superior izquierda, en la que no habría ningún falso positivo y ningún falso negativo.

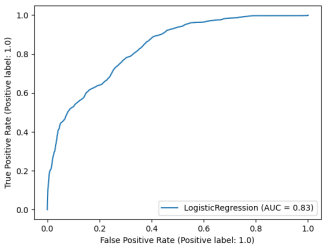
En cambio, para analizar la presencia o no de estímulo, hemos observado que el clasificador de regresión logística no produce buenos resultados, como podemos ver en la figura 4c.



(a) Matriz de confusión de random forest.



(b) Curva ROC de random forest.



(c) Curva ROC de regresión logística.

Fig. 4: Gráficas de análisis de modelos sobre el dataset estímulos.

Por otro lado, obtenemos buenos resultados con Random Forest, que nos da un porcentaje de aciertos de 99.99%. Podemos observar la matriz de confusión 4a y la curva ROC 4b de este modelo sobre los datos de entrenamiento.

6 Discusión y conclusiones

El modelo que ha dado mejores resultados es el de Regresión Logística para clasificar por tipo de sustancia y el de Random Forest para detectar la presencia o ausencia del estímulo. Los atributos que identifican mejor a un experimento son los valores máximos y mínimos de los sensores, eliminando la humedad y la temperatura, así como los sensores 1 y 5. Estos resultados concuerdan con las observaciones del paper de Ramón Huerta [3], en el que estudian la relación entre los sensores de gas y la humedad y temperatura. Nos explican que la reducción en la exactitud al entrenar el modelo con la humedad y temperatura puede deberse a que esta información adicional es redundante.

Utilizamos dos modelos distintos debido a que se trata de dos tipos de problemas diferentes. En el caso de detectar el estímulo, se trata de un problema con solamente dos clases (presente y ausente) y se trabaja con un dataset con una gran cantidad de filas. Dado que se puede apreciar visualmente en las gráficas una diferencia clara en los valores de los sensores dentro del intervalo de presencia del estímulo indicado por los metadatos, es razonable concluir que un Random Forest será capaz de identificar fácilmente las condiciones que determinan la clase.

En cambio, para el caso de clasificar por el tipo de sustancia, es necesario separar linealmente los experimentos en el espacio de atributos seleccionado para dicha tarea. Se trata de un dataset de un tamaño más reducido, con una fila por cada experimento. Para este caso, es una combinación compleja de los valores que toman esos atributos lo que determina el resultado. Por tanto, la aplicación de un modelo de Regresión Logística conseguirá separar ese espacio de atributos de la mejor manera posible y clasificar cada experimento correctamente.

En definitiva, si comparamos los resultados de exactitud obtenidos para los conjuntos de datos de estímulo y clasificación, 99.99% 83.35% y respectivamente, podemos decir que son más que aceptables en comparación con los presentados por otras fuentes como el paper de Ramón Huerta [3], donde logran una exactitud de 78.5% en la tarea de clasificar entre plátano y vino utilizando el mismo conjunto de datos que nosotros.

En este mismo paper dan unos modelos matemáticos para mejorar la exactitud de las mediciones de los sensores de gas realizando un filtrado de la humedad y la temperatura. Nos muestran también que se logra una mejor clasificación entre vino y plátano añadiendo los datos filtrados al modelo. Sería interesante en un futuro crear un modelo de clasificación utilizando las mediciones filtradas mediante estas fórmulas para analizar si conseguimos encontrar un modelo más exacto.

También sería muy positivo contar con más datos de entrenamiento, lo que podríamos lograr replicando el experimento con el que se han obtenido estos datos.

References

[1] Ramon Huerta, Thiago Mosqueiro, Jordi Fonollosa, Nikolai Rulkov, and Irene Rodriguez-Lujan. Gas sensors for home activity monitoring data set, online decorrelation of humidity and temperature in chemical sensors for continuous monitoring. `https://github.com/thmosqueiro/ENose-Decorr_Humdt_Temp`, 2016.

[2] Ramon Huerta, Thiago Mosqueiro, Jordi Fonollosa, Nikolai Rulkov, and Irene Rodriguez-Lujan. Gas sensors for home activity monitoring data set, online decorrelation of humidity and temperature in chemical sensors for continuous monitoring. `https://archive.ics.uci.edu/ml/machine-learning-databases/00362/HT_Sensor_UCIsubmission.zip`, 2016.

[3] Ramon Huerta, Thiago Mosqueiro, Jordi Fonollosa, Nikolai F Rulkov, and Irene Rodriguez-Lujan. Online decorrelation of humidity and temperature in chemical sensors for continuous monitoring. *Chemometrics and Intelligent Laboratory Systems*, 157:169–176, 2016.

[4] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[5] Sklearn.linear_model.logisticregression. `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`.

[6] Sklearn.linear_model.logisticregression. `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html`.