Uso de Máquina de Soporte Vectorial para la valoración de calidad de pruebas de espirometría

Angela Bravo Córdoba, Jofre A. Manchola Martínez, Jairo A. Ruiz Sáenz
Departamento de Ingeniería de Sistemas y Computación
Universidad de los Andes, Bogotá, Colombia
{a.bravoc, ja.mancholam, ja.ruiz907}@uniandes.edu.co

***Resumen*— En el Laboratorio de Función Pulmonar de la Fundación Neumológica Colombiana se realizan diversas pruebas de función pulmonar, las cuales deben ser evaluadas de manera individual para establecer un indicador de calidad de ejecución de pruebas y verificar si se requiere realizar de nuevo. Por una alta demanda de pruebas no es posible evaluar todas las pruebas.**

Dado lo anterior, utilizando una Máquina de Soporte Vectorial (SVM) se implementó un algoritmo de clasificación con clases no balanceadas que permite evaluar la calidad de las pruebas de espirometría. Como resultado se obtuvo un recall del 70.45% permitiendo a la Fundación Neumológica tener un mejor conocimiento y control sobre la calidad de las pruebas de espirometría, ya que podrá identificar aquellas pruebas que requiera repetir y así ahorrar tiempo en el proceso de evaluación.

***Abstract*-- In the Pulmonary Function Laboratory of the Colombian Pneumology Foundation various pulmonary function tests are performed, which must be evaluated individually to establish an indicator of quality of execution of tests and verify if it is necessary to perform again. Due to a high demand for tests it is not possible to evaluate all the tests.**

Given the above, using a Support Vector Machine (SVM), a classification algorithm with unbalanced classes was implemented to evaluate the quality of spirometry tests. As a result, a recall of 70.45% was obtained allowing the Pneumology Foundation to have a better knowledge and control over the quality of spirometry tests, since it will be able to identify those tests that require repeating and thus save time in the evaluation process.

I. Introducción

Este documento presenta el desarrollo de un ejercicio académico de estudiantes de la universidad de los Andes en el uso de técnicas de machine learning para la solución de problemas en el análisis y procesamiento de información, se trabajo en conjunto al Laboratorio de Función Pulmonar de la Fundación Neumológica Colombiana, quienes realizan diversas pruebas de función pulmonar y dentro de los indicadores de calidad de la fundación se tiene la calidad de la ejecución de dichas pruebas, estas deben ser evaluadas de manera individual por los terapeutas del laboratorio lo cual implica un reto para la fundación dado el tiempo requerido para esta tarea por la alta demanda de procedimientos de este tipo.

Se pretende entrenar un modelo de machine learning que permita al Laboratorio de Función Pulmonar de la Fundación Neumológica Colombiana identificar oportunamente las pruebas de espirometría que presenten deficiencias en la calidad del procedimiento efectuado con el propósito de mejorar la calidad del servicio ofrecido y reducir los tiempos dedicados a la verificación de calidad de estos procedimientos.

Objetivo de negocio: Poder identificar las pruebas de espirometría realizadas con probabilidad de estar mal ejecutadas con el fin de revisarlas por parte de un profesional y de ser necesario repetirlas.

Objetivo de machine learning: Implementar un algoritmo de clasificación con clases no balanceadas.

II. Análisis exploratorio y preparación de datos

A. *Análisis exploratorio*

La fundación neumológica hizo entrega de dos datasets los cuales fueron anonimizados por contener datos sensibles de los pacientes, el primero (*df_pruebas*) correspondiente a los resultados de las lecturas de los sensores del equipo[1] de espirometrías con 332.143 maniobras[2] realizadas desde el 5 de marzo de 2006 hasta el 2 de marzo de 2019 y el segundo (*df_calidad*) corresponde a las pruebas calificadas por un profesional de la fundación neumológica.

Se identifica que el dataset df_pruebas tiene un alto porcentaje de faltantes, consultando con el experto nos expresa que el dataset contiene información relacionada con varios tipos de pruebas respiratorias, y solo debemos utilizar las de tipo “FVL” o pruebas de capacidad vital forzada.

Dada la cantidad de registros fue necesario importarlos en una base de datos PostgreSQL para realizar la unión de las variables medidas por el equipo de espirometría, y el conjunto de datos etiquetado por los expertos (*df_calidad*) sobre 153 pruebas. Como resultado se obtuvo un conjunto de datos con 287 registros y 185 variables correspondientes a las mediciones realizada antes y después del uso de broncodilatador inhalado que se utiliza durante la prueba de espirometría.

La razón por la cual no se obtuvieron 306 (153 * 2 [pre y post

[1] El equipo de espirometría se refiere al aparato utilizado para realizar la medición durante la prueba.

[2] En una prueba de espirometría se debe realizar el procedimiento varias veces en caso de que existan anomalías en la medición, cada una de estas repeticiones recibe el nombre de maniobra.

broncodilatador inhalado]) registros es porque hay pruebas donde no se cuenta con la medición después del broncodilatador inhalado y porque 8 pruebas calificadas no fueron identificadas en la base de datos del equipo de espirometría.

B. Preparación de datos

Teniendo en cuenta el objetivo del negocio y el objetivo de machine learning a resolver, se propone implementar un algoritmo de SVM (Máquina de Soporte Vectorial), para lo cual fue necesario hacer una preparación de datos, se debe mencionar que se hizo la preparación de datos en dos etapas, la primera siendo una preparación en *Python* mediante *jupyter notebook* y para la segunda etapa se utilizó *Rapidminer*, entre los pasos llevados a cabo vale la pena destacar:

1. Al ser un dataset de procedimientos médicos, este contiene datos sensibles y de carácter confidencial, por tanto, se procedió en realizar un proceso de anonimización de los datos (se usó un método hash llamado hash224) con el propósito de conservar la privacidad de los pacientes.
2. Se procedió a unir los dataset entregados por la fundación, para obtener una tabla única de datos que contienen las mediciones de los sensores al momento de realizar las pruebas junto a la calificación dada por el profesional, para esto se tuvo en cuenta el número de identificación de los pacientes y el día de realización de la prueba (se debió ajustar los formatos de las fechas previamente).
3. Se filtraron las pruebas por tipo "FVL", ya que corresponden a las pruebas con calificación por parte de los expertos.
4. Una de las variables en el dataset es terapeuta, la cual corresponde al nombre de quien realizó la prueba, se identificaron variaciones en los nombres como vane/vanessa, por tanto fue necesario modificar esta variable para unificar los nombres.
5. Ya que el algoritmo SVM no trabaja bien con valores faltantes (nulos), se filtraron aquellas variables donde la totalidad de los registros tienen valores ausentes, de un total de 168 variables se eliminaron 92 variables, para ello se utilizó el operador "*Remove Useless Attributes*" de Rapidminer.
6. Como resultado del análisis exploratorio de datos, se detectaron muchas variables numéricas que son generadas por los equipos de medición que pueden tener alta correlación, por lo tanto se eliminaron 30 variables que presentaban correlación superior a 0.95 con otra variable del conjunto de datos. Se utilizó el operador "*Removed Correlated Attributes*" de *Rapidminer*.
7. Se realizó una selección de variables teniendo en cuenta el operador "*Weight by Information Gain Ratio*" de *Rapidminer*, como resultado el operador selecciona 18 de las 46 variables del conjunto de datos.
8. Ya que el algoritmo SVM no trabaja con variables nominales, las variables categóricas seleccionadas fueron transformadas a tipo numéricas con el operador "*Nominal to Numerical*" de *Rapidminer* con tipo de codificación dummy. Finalmente se obtiene un conjunto de datos con 51 variables donde las variables "QualityGrade", "Terapeuta" y "equipo" fueron convertidas a variables binarias.
9. En las variables numéricas todavía se encuentran valores faltantes, por lo tanto se realiza tratamiento a estos imputando el valor cero usando el operador "*Replace Missing Values*".

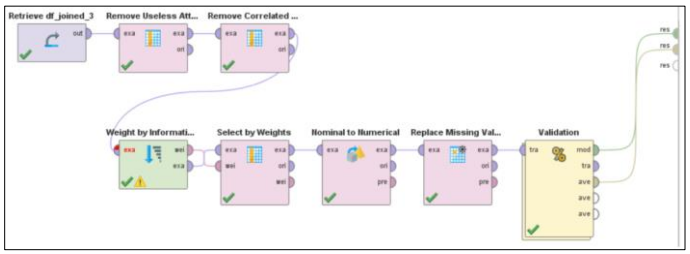

Figura 1: Operadores usados en Rapidminer para la preparación de datos

III. ENTRENAMIENTO DEL MODELO Y ENTONACIÓN DE PARÁMETROS E HIPERPARÁMETROS.

A. Entrenamiento del modelo

Como solución al problema planteado se entrenó un algoritmo de clasificación con clases no balanceadas utilizando una Máquina de Soporte Vectorial (SVM), para construir el modelo se utilizó el nodo "*Support Vector Machine*" de *Rapidminer*, además se utiliza el nodo "*Split Validation*" para dividir la muestra de forma aleatoria en casos de *entrenamiento* y *testing*. Internamente se realiza la medición del desempeño del modelo usando el nodo "*Perfomance*" el cual permite evaluar la calidad de las pruebas de espirometrías.

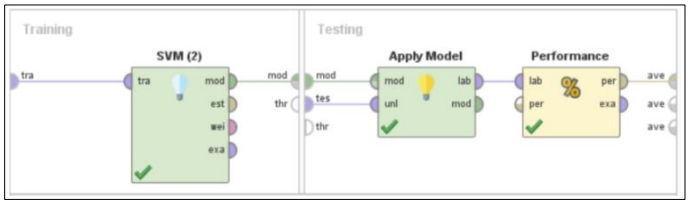

Figura 2: Operadores usados en Rapidminer para entrenamiento y validación del modelo

B. Entonación de parámetros e hiperparámetros

Para la entonación de parámetros se utilizó el nodo *optimize parameter grid* de *Rapidminer*, donde se probaron 1.705 combinaciones resultado de combinar los siguientes parámetros:

1. **Tipo Kernel**: donde se probaron los tipos dot, radial, polynomial, neural y anova.
2. **Kernel gamma**: Se probaron valores entre 0 y 100, se seleccionaron 30 tramos.
3. **Parámetro C**: Se probaron valores entre -1 y 20, se seleccionaron 10 tramos.

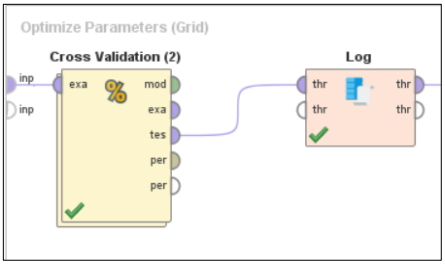

Figura 3: Proceso interno del nodo optimize parameter (grid)

Internamente se utilizó el nodo *cross validation* para validar los resultados obtenidos en el entrenamiento del modelo y descartar sobreajuste de los modelos generados, adicionalmente se utilizó un nodo log para almacenar los resultados obtenidos en cada una de las combinaciones.

IV. Resultados de la evaluación del modelo.

Se analizaron tres medidas de desempeño para verificar la mejor combinación de parámetros para entrenar el modelo:

1. **Accuracy**: el valor más alto obtenido es de 0,926, usando los parámetros: Tipo kernel=dot, Kernel gamma=26,667 y C=-1.
2. **Precisión**: el valor más alto obtenido es de 0,893, usando los parámetros: Tipo kernel=Polynomial, Kernel gamma=0 y C=-1.
3. **Recall**: el valor más alto obtenido es de 0,752, usando los parámetros: Tipo kernel=dot, Kernel gamma=80, C=5,3.

Dado que es importante para el laboratorio poder detectar aquellos resultados que no son confiables o que presentan fallas en la toma del examen, es decir, se espera que el porcentaje de casos falsos negativos sea bajo, se selecciona el modelo en el que se obtuvo el valor de recall más alto.

A continuación se analizan los resultados obtenidos y qué parámetro tiene mayor influencia para que el modelo obtenga un valor de recall alto:

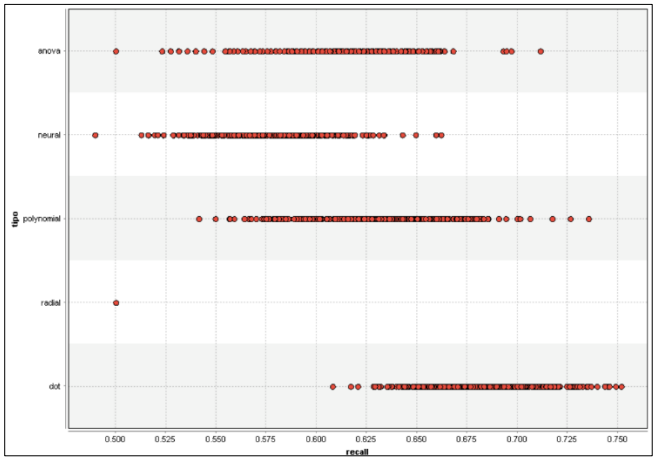

Figura 4: Recall obtenido para los diferentes tipos de kernel

Analizando los tres parámetros se detecta que el parámetro que mayor influencia tiene en el valor de recall del modelo es el tipo de kernel usado, tal como se observa en la Figura *4*, los modelos entrenados con el tipo de kernel dot son los que en promedio obtienen el recall más alto.

También se realizó el entrenamiento del modelo usando los parámetros seleccionados en un 80% de la muestra y se validaron los resultados en un 20% de la muestra que no fue usado en el entrenamiento.

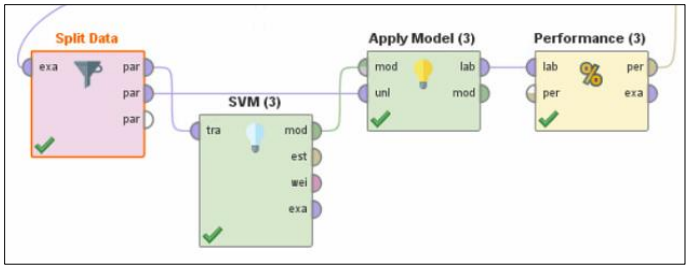

Figura 5: Proceso de evaluación del modelo entrenado

Se obtuvo un recall del 70.45% como se puede observar en la Figura *6*.

weighted_mean_recall: 70.45%, weights: 1, 1

	true AAA	true 1	class precision
pred. AAA	50	4	92.59%
pred. 1	1	3	75.00%
class recall	98.04%	42.86%	

Figura 6: Matriz de confusión del modelo entrenado

V. DESPLIEGUE.

A pesar de la flexibilidad y facilidad de uso de la herramienta *Rapidminer*, hay que tener en cuenta una limitante que presenta la herramienta, esta no permite realizar un despliegue del proceso implementado, razón por la cual se optó por utilizar el framework de desarrollo *Django* junto a la librería *scikit learn* de *Python* para desarrollar una aplicación web que le permita al usuario evaluar las pruebas de espirometrías.

Teniendo en cuenta los parámetros de entonación encontrados en la etapa de entonación de parámetros e hiperparámetros, se procedió a realizar una implementación del modelo SVM en *scikit learn* para así poder exportar un archivo *pickle* del modelo, el cual será utilizado como insumo en el script de evaluación de las pruebas en *Django*.

En cuanto a la aplicación web, se desarrolló una interfaz que incluye una descripción del proyecto desarrollado, funcionalidad de autenticación de usuarios y la opción de carga de archivos *.csv* para la evaluación masiva de pruebas, se planteó incluir una vista para la evaluación individual de pruebas mediante el diligenciamiento de un formulario, sin embargo, dada la gran cantidad de variables por diligenciar se decidió no incluirla en el prototipo final.

Una vez evaluadas las pruebas del archivo cargado, se habilita una opción para descargar un nuevo archivo de resultados en formato *.xlsx* que incluye una nueva columna con la respectiva clasificación del modelo para cada registro.

La aplicación desarrollada fue desplegada en *Heroku* y puede acceder a esta ingresando al link: *https://espirometria.herokuapp.com/*

VI. CONCLUSIONES.

Los resultados obtenidos en el modelo son muy buenos, lo cual permitirá al personal médico de la Fundación Neumológica Colombiana tener mayor certeza en el resultado de las pruebas de espirometría y de este modo reducir los tiempos en la toma del examen y mejorar la calidad del servicio ofrecido.

Se recomienda al laboratorio realizar una prueba piloto donde se pueda testear el modelo usando un porcentaje de las pruebas realizadas y comparar los resultados obtenidos contra el diagnostico final de los terapeutas, de esta manera se podrá reentrenar el modelo y mejorar el rendimiento de este.