Texto

Descripción generada automáticamenteUniversidad de los Andes

Ingeniería de Sistemas y Computación

**ISIS3301 Inteligencia de Negocios**

**Proyecto 1-Etapa 2**

**Elegibilidad de un paciente para Ensayos clínicos (Diseño de aplicación)**

Daniel Alejandro Ángel – 201911345

Jairo Adolfo Céspedes Plata – 201912008

Bogotá D.C, 9 de may. de 22

Tabla de contenido

[1 Objetivos del negocio (Parte del Proyecto 1) 3](#_Toc103017138)

[2 Estado de los datos y procesamiento realizado 4](#_Toc103017139)

[3 Solución y Modelo propuesto para su utilización en el Pipeline 6](#_Toc103017140)

[4 Generación de Pipelines inspirados en la preparación de datos y el modelo descrito. 6](#_Toc103017141)

[5 Realización de la página web mediante el API 8](#_Toc103017142)

[6 Resultados encontrados 8](#_Toc103017143)

[7 Métricas de los modelos 9](#_Toc103017144)

[8 Conclusiones para el negocio 11](#_Toc103017145)

[9 Conclusiones del trabajo en equipo 11](#_Toc103017146)

[10 Referencias 12](#_Toc103017147)

# Objetivos del negocio (Parte del Proyecto 1)

Determinar la elegibilidad de un paciente para ensayos clínicos de cáncer a partir de texto descriptivo.

**Label 0**: es elegible para ensayos clínicos contra el cáncer basado en los datos

**Label 1**: no es elegible para ensayos clínicos basados en los datos

|  |  |
| --- | --- |
| Oportunidad/problema Negocio: | El negocio quiere determinar, a partir de la información proporcionada de pacientes de cáncer, la elegibilidad de estos en ensayos clínicos para combatir el cáncer. Teniendo esto en cuenta, no todos los pacientes están habilitados para estos ensayos, dependiendo del tipo de cáncer que tienen y la información del estudio que se le realizó a cada uno de los pacientes. Poder clasificar a los pacientes en gran escala entre aquellos que son elegibles para el ensayo clínico o no le facilitará al negocio poder agilizar el proceso de selección de aquellos pacientes que pueden aplicar al ensayo clínico optimizando así el proceso de tratamiento contra el cáncer o, en dado caso, la búsqueda de otras alternativas para combatir el cáncer en los pacientes. |
| Descripción del requerimiento desde el punto de vista de aprendizaje de máquina: | Se desea tener una tarea de aprendizaje de máquina que permita clasificar o agrupar los pacientes, según los datos proporcionados en la variable study\_and\_condition, para determinar cuáles son elegibles para ensayos clínicos. Teniendo en cuenta que la variable study\_and\_condition es la que contiene un texto descriptivo de los estudios realizados y del tipo de cáncer del paciente, se sabe que inicialmente se debe hacer un tratamiento de texto para obtener únicamente aquella información y palabras que aporten valor al modelo, es decir, se deben eliminar/omitir del análisis los signos de puntuación, conjunciones y proposiciones. Por otra parte, dado que los datos que se nos presentan están etiquetados y que tenemos la variable label que nos dice si un paciente es elegible para los ensayos clínicos, descartamos el aprendizaje de maquina no supervisado. Por esto, definimos la problemática del negocio como una tarea de clasificación. Dicho esto, se sabe que se debe realizar una limpieza de aquellos datos duplicados, valores nulos y valores atípicos dado su poco aporte al valor del modelo. Por otra parte, al realizar el análisis del texto, se debe realizar un TfidVectorizer para poder saber la cantidad de veces que aparece una palabra en cada una de las descripciones y según ese número de apariciones, cruzar la información con la elegibilidad de esos pacientes y de esta manera poder saber la influencia de cada una de las palabras en la elegibilidad de un paciente con Cáncer para ensayos clínicos. Por otra parte, para eliminar posibles complicaciones con la comparación de palabras al realizar el proceso de TfidVectorizer, se define que también se debe convertir todos los valores de los registros de la variable que contiene la descripción a minúscula para poder realizar las comparaciones de una manera más certera y precisa. |

# Estado de los datos y procesamiento realizado

(Se inicia con el mismo procedimiento del Laboratorio 1 Etapa 1 excepto porque ahora se usará un TfidVectorizer en lugar )

En cuanto al análisis inicial de los datos, se obtuvo que se tienen un total de 12001 registros en los datos usados para general el modelo y con los cuales se ejecutó el modelo y se realizaron las pruebas. Para estos registros, se tenían 2 variables: label y study\_and\_condition. Esta primera contiene la información de aquellos pacientes que históricamente fueron elegidos para ensayos clínicos basados en la información de sus estudios y su tipo de cáncer. Afortunadamente, luego de un análisis exhaustivo de los datos, se encontró que para esta columna no se debe realizar una limpieza significativa. Esto teniendo en cuenta que los valores se encuentran estructurados, esta columna solo tiene dos valores posibles: \_\_label\_\_0 y \_\_label\_\_1, de los cuales ninguno de los 12001 eran valores atípicos. Por esto, se decidió convertir los valores de la variable label en valores binarios para facilitar el análisis del modelo y el ahorro de recursos en la máquina.

Texto

Descripción generada automáticamente

Por otra parte, para la segunda variable, se tiene que se puede realizar un procesamiento de datos facilitado gracias a la estructura que presentan los datos. De esta manera, se tienen que los datos de study\_and\_condition está separada por un punto, donde antes del punto se tiene información acerca de los estudios de cada uno de los pacientes y, luego del punto, se tiene información acerca del tipo y el estado en el que está el cáncer en el paciente. En este caso, se realizó un análisis para más del 90% de los datos donde se encontró que se cumplía esta estructura, por lo que se decidieron aplicar estrategias de procesamiento de texto teniendo en cuenta la estructura dada. Dicho esto, estas fueron las estrategias de limpieza y procesamiento de texto aplicadas:

* Conversión de todo el texto a minúscula para facilitar la comparación de palabras y el posterior proceso de TfidVectorizer para analizar la aparición de palabras en los textos. Esto se realizó gracias al método. lower().

nuevo\_texto = texto.lower()

* Eliminación de los signos de puntuación que aparecen en el texto. Dada la posibilidad de que muchas de la palabras que aparecen no tengan un signo de puntuación contiguo y otras si, la comparación de palabras sería errónea, por lo que se deciden eliminar todos los signos de puntuación mostrados a continuación. Esto se realizó gracias al método .sub()

regex = '[\\!\\"\\#\\$\\%\\&\\\'\\(\\)\\\*\\+\\,\\-\\.\\/\\:\\;\\<\\=\\>\\?\\@\\[\\\\\\]\\^\_\\`\\{\\|\\}\\~]'

nuevo\_texto = re.sub(regex , ' ', nuevo\_texto)

* Eliminación de conjunciones, proposiciones y otras palabras innecesarias. Este método también se realizó gracias al método .sub() donde se reemplazaron las palabras especificadas por espacios en blanco.

conj = [" and "," are "," for "," of "," with "," to "," as "," or "," do "," the "," in "," than ", "study ", " interventions "]

    for i in conj:

        nuevo\_texto = re.sub(i, ' ', nuevo\_texto)

* Eliminación de los números en la cadena de texto.

# Eliminación de números

nuevo\_texto = re.sub("\d+", ' ', nuevo\_texto)

* Finalmente, se realizó la eliminación de los espacios blancos múltiples que se encuentran en el texto, estos se pudieron haber dado gracias a la posibilidad de conjunciones, proposiciones, signos de puntuación y espacios en blanco contiguos.

# Eliminación de espacios en blanco múltiples

nuevo\_texto = re.sub("\\s+", ' ', nuevo\_texto)

Por lo que, al finalizar la limpieza de datos y el procesamiento de texto para facilitar el análisis en el modelo de clasificación, se presentan los primeros 5 datos del dataframe como muestra para analizar el correcto funcionamiento de la función de limpieza de datos y analizar si es necesario realizar nueva limpieza.

Texto

Descripción generada automáticamente

Finalmente, luego de analizar un número significativo (>100) registros, se determinó que la limpieza de datos se realizó correctamente y se puede proceder a la ejecución del proceso de TfidVectorizer y consecuentemente la creación y ejecución de los modelos de clasificación explicados a continuación.

# Solución y Modelo propuesto para su utilización en el Pipeline

Para esta etapa del proyecto y basados en los resultados de la etapa 1 ofrecemos al usuario final los 3 modelos posibles:

|  |  |  |
| --- | --- | --- |
| **Detalles de la actividad de minería de datos** | | |
| Tipo aprendizaje | Tarea de aprendizaje | Algoritmo e hiper-parámetros utilizados (con la justificación respectiva) |
| Supervisado | Clasificación | **Decision tree (criterion='entropy', random\_state=0, max\_depth=180)**. Dado que nosotros queremos medir la calidad de los datos a través de la ganancia de información que se presenta en el modelo, seleccionamos el criterio de los datos como entropy en vez del criterio Gini que define la pureza de los datos. El random state define la aleatoriedad del estimador, por lo que, si se define como un entero, se tiene que se desea que sea determinístico el modelo. Finalmente, un max Depth define que la profundidad máxima en la que el árbol va a bajar a través de sus nodos será de 180, este número se definió gracias a pruebas usando búsqueda de los mejores hiper-parámetros. Sin embargo, dado el tiempo de ejecución la búsqueda de los mejores hiper-parámetros se decidió únicamente incluir el resultado de los hiper-parámetros mas no la búsqueda de estos. |
| Supervisado | Clasificación | **Random Forest(n\_estimators=100).** En este caso, dada la naturaleza de este modelo, únicamente se definió n\_estimators como 100, dado que queremos que nuestro modelo cree y ejecute 100 árboles de decisión para poder tener una mayor precisión de los resultados. Este número también fue obtenido gracias a pruebas con búsqueda de hiper-parámetros. |
| Supervisado | Clasificación | **Logistic regression (random\_state=0, max\_iter=1000).** Para el caso de regresión logística definimos nuevamente un random\_state de 0 definiendo una aleatoriedad del solucionador baja y tomando un entero para definir el modelo como determinístico. Por otra parte, el max\_iter define el número máximo de iteraciones que demoran los solucionadores en converger, ambos parámetros fueron hallados a través de pruebas de ejecución para obtener las mejores métricas. |

# Generación de Pipelines inspirados en la preparación de datos y el modelo descrito.

Ya con la preparación de datos y los modelos ya ajustados a los datos de entrenamiento procederemos a hacer una limpieza similar, pero en forma de pipeline de manera que el modelo pueda ser utilizado sin necesidad de realizar el procedimiento del notebook otra vez.

* Como son datos basados en texto, procedemos a realizar los mismos pasos que se mostraron en el literal 2 a manera de una función de transformación automatizable (Así que no se va repetir la explicación de los pasos de la eliminación de diferentes caracteres y cadenas de texto ya que es la misma función del literal 2):

Texto

Descripción generada automáticamente

* Después de esto, le aplicamos al pipeline el TfidVectorizer, aclarando que se debe hacer un pre-proceso de los datos (con la función mostrada anteriormente) y finalmente aplicamos el modelo al pipeline

Captura de pantalla de computadora

Descripción generada automáticamente

* Lo exportamos y revisamos sus estadísticas:

Random Forest:

Calendario

Descripción generada automáticamente

Regresión logística:

Calendario

Descripción generada automáticamente

Árbol de decisión:

Calendario

Descripción generada automáticamente

# Realización de la página web mediante el API

# Resultados encontrados

Luego de la ejecución de los modelos mediante la API desarrollada, se realizó un análisis del proceso TfidVectorizer, en el que se pudo examinar la cantidad de veces que aparecía cada una de las palabras para cada uno de los posibles resultados de la variable Label. Es decir, se realizó un análisis de las palabras que más aparecieron para los casos en los que los pacientes de cancer eran elegibles para ser incluidos en el proceso de un ensayo clínico y se realizó un análisis de las palabras que más aparecieron en el caso de los que no eran elegibles.

Luego de realizar este análisis, se obtuvieron los siguientes resultados:

Gráfico, Gráfico de barras

Descripción generada automáticamente

Gráfico, Gráfico de barras

Descripción generada automáticamente

Resultados en los que cabe destacar la aparición de las palabras greater, one, must, less para el caso de los pacientes que son elegibles para los ensayos clínicos. Además, teniendo en cuenta el contexto médico, se puede realizar un análisis más profundo del contexto de cada una de las palabras para hallar los patrones que definen una mayor influencia en la elegibilidad de un paciente para los ensayos clínicos.

Por otra parte, en el caso de aquellos pacientes que no son elegibles, se destaca la aparición de las palabras prior, carcinoma, within y recurrent. Palabras que no aparecieron en una cantidad significativa en el primer escenario. Según esto, y de la misma manera que en el anterior escenario, se puede realizar un análisis más exhaustivo para poder encontrar mucho más valor y aumentar la precisión del modelo. Sin embargo, para este caso se obtuvieron buenas métricas de desempeño para los modelos desarrollados. Estas métricas serán presentadas a continuación.

# Métricas de los modelos

# Conclusiones para el negocio

Finalmente, se puede determinar que el modelo y, por consiguiente, la aplicación tiene un buen desempeño luego de haber realizado varios análisis a las métricas arrojadas tras la ejecución. Además, se tiene que, gracias a las métricas mostradas, se puede notar que nuestro modelo tiene una mayor tendencia a incurrir en falsos negativos, es decir, clasificar a pacientes que sí son elegibles como pacientes que no son elegibles para los ensayos clínicos. Caso por el cual, dado el contexto médico, puede ser de mayor preferencia dada la delicadeza de los ensayos clínicos que se practican a los pacientes y, teniendo en cuenta, que muchos de ellos dadas sus condiciones no son aptos por el alto riesgo vital que representa para ellos, por lo cual, clasificarlos como elegibles cuando en realidad puede ser mortal, llegaría a ser un error significativamente peor que incurrir en un mayor número de falsos negativos.

Por otra parte, también se tiene que el mejor modelo de los que se desarrolló fue Random Forest. Esto se puede ver justificado dada la naturaleza del modelo como un conjunto de modelos para los cuales se evalúan n arboles de decisión en la búsqueda de aquel que arroje la mejor precisión de predicciones. Siento finalmente las métricas de la clasificación de aquellos pacientes elegibles: precisión de 84% y un recall de 82%.

Tomando en cuenta, ya la aplicación como tal

# Conclusiones del trabajo en equipo

Jairo Cespedes:

* + Líder de proyecto
  + Ingeniero de datos
  + Limpieza y preparación para el procesamiento de datos
  + Inclusión del modelo escogido
  + Análisis del estado de datos
  + Programación de tareas

Tiempo estimado total de trabajo (7 horas)

Daniel Angel:

* + Ingeniero de software responsable del diseño de la aplicación y resultados
  + Ingeniero de software responsable de desarrollar la aplicación final
  + Inclusión del modelo escogido
  + Análisis de resultados de los modelos escogidos
  + Interpretación de resultados de cara al negocio
  + Verificación de la calidad del proyecto y la aplicación

Tiempo estimado total de trabajo (8 horas)

Por otra parte, las reuniones planteadas en el enunciado del proyecto se realizaron luego de cada una de las clases de los jueves. Tiempo en el cual se llevó seguimiento del avance del proyecto y se repartieron las tareas según la carga académica de cada uno de los integrantes dependiendo de la semana en la que se encontraban. Se definió como compromiso para el siguiente proyecto continuar con el proceso que se ha llevado a cabo por ahora y mantener el compromiso de ambos integrantes con el equipo.

Repartición de puntos:

Total:100

Jairo Céspedes: 50

Daniel Angel: 50

# Referencias

[Scikit-learn: Machine Learning in Python](http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html), Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.