Predicción de posibles enfermos mentales según su entorno laboral

Lucía Calzado Piedrabuena
Minería de Datos
Ciudad Real
lucia.calzado1@alu.uclm.es

1. INTRODUCCIÓN

En este trabajo se pretende desarrollar, mediante el proceso KDD, un patrón que permita averiguar si un empleado cualquiera de una compañía tiene una enfermedad mental a partir de una serie de preguntas sobre cómo lleva la empresa el trato de las enfermedades mentales.

Hoy en día, las enfermedades mentales siguen siendo tema tabú en la sociedad, sobre todo en el entorno laboral. Mediante este proyecto quiero investigar si fuera posible saber si una persona tiene una enfermedad mental dependiendo de las respuestas que da a ciertas preguntas sobre cómo es el ambiente en el trabajo respecto a la salud mental.

Recursos

Para las partes que he necesitado escribir código, por ejemplo, la adecuación de la base de datos y la transformación de estos, he usado el lenguaje de programación Python. Los scripts que he escrito están adjuntos a este proyecto, y explicados brevemente más adelante.

También he modificado un programa con el algoritmo KNN para adaptarlo a mi dataset, igualmente en Python.

Una vez pasada la fase de transformación, decidí usar una herramienta web, BigML, para realizar casi todo el proceso de Minería de datos (exceptuando el algoritmo KNN mencionado anteriormente). Esta página me ha permitido lanzar varios modelos y evaluarlos rápidamente, para ver cuál de ellos es el que mejor se adapta a mi dataset.

Además, cuenta con una serie de gráficos que facilitan mucho la tarea de interpretar los datos y decidir qué hacer, si fuera necesaria, en la siguiente iteración del proceso KDD.

2. PROCESO KDD

2.1 Primera iteración

2.1.1 Seleccionar el conjunto de datos

Esta es la primera fase del proceso KDD donde se ha elegido una base de datos, que es la siguiente: https://www.kaggle.com/ekwiecinska96/mental-health-in-techology-survey-2014-and-2016#survey_2016.csv

Al ser una base de datos de *Kaggle*, no es necesario convertir su formato, por lo que podemos comenzar a trabajar en la segunda fase inmediatamente.

A continuación, se procede a explicar la base de datos y el trabajo que se realizará sobre ella.

Esta base de datos contiene 63 preguntas realizadas a 1433 personas sobre cómo son tratadas las enfermedades mentales en su empresa, cómo creen que reaccionarían los compañeros de trabajo ante una enfermedad mental y si ellos tienen una enfermedad mental, entre otras preguntas del mismo estilo.

Los motivos por los que he decidido adecuar de este modo el dataset (las columnas borradas son mostradas en el script `quitarColumnas`, explicado más adelante) son los siguientes:

- Muchas de ellas no están relacionadas con el tema de cómo ven los empleados cómo se llevan las enfermedades mentales en su empresa *("Is your employer primarily a tech company/organization?"*, por ejemplo, o las relacionadas con la familia *"Do you have a family history of mental illness?"*)
- Las preguntas de *"¿Por qué no?"* son de texto libre, y aparte de la imposibilidad de interpretarlas para Minería de datos, no interesan para este ejercicio.
- Están relacionadas con una empresa anterior (hay aproximadamente 20 preguntas referidas a experiencias con empresas anteriores), y eso no interesa para el tema del ejercicio, que se tiene que centrar solo en la empresa actual.
- La geolocalización de los encuestados no tiene tampoco cabida para el tema principal del ejercicio, así como su puesto de trabajo.

Hay algunas columnas que, a pesar de parecer a primera vista que no tienen relación con el tema principal, puede ser interesante tenerlas en cuenta. Estas columnas son:

- *Teletrabajo*. Pienso que el hecho de trabajar remotamente puede tener impacto en el ambiente de trabajo (en este caso, si trabajaría esa persona sola o en la oficina), y me gustaría ver si tiene consecuencias en el hecho de pensar que tiene una enfermedad mental.
- *Edad y género*. Creo que estas dos columnas pueden ser útiles al final, sobre todo la de edad.
- *Número de empleados de la compañía*. creo que es interesante saber lo que piensa la persona del ambiente en la compañía dependiendo de los empleados que tenga.

He cogido la columna "*Do you currently have a mental health disorder?"* en vez de *"Have you been diagnosed with a mental health condition by a medical professional?*", porque lo que me interesa es saber si la persona piensa que tiene un desorden mental, no si ha ido a que un profesional le afirme que lo tiene, puesto que si la persona piensa que lo tiene es que no se encuentra bien, y es eso lo que quiero intentar averiguar.

Obtengo la tarjeta de datos en esta fase.

2.1.2 *Pre-procesado*

Tratamiento de datos nulos, blancos y outliers

Aquí se limpia, ordena y se prepara la base de datos para su posterior utilización por el algoritmo. Hay tres puntos básicos que voy a tratar:

- Datos en blanco

Los datos en blanco del dataset se han convertido en datos nulos al pasarlo a Python. Estos datos son respuestas de la encuesta que se han dejado en blanco. En la mayoría de los casos, las he decidido interpretar dependiendo del contexto de la pregunta, asignándoles el valor "quizá", puesto que he considerado que si se deja una pregunta de sí/no en blanco en una encuesta es porque la respuesta sería "quizá".

- Datos nulos, o que se salen de los parámetros

He filtrado toda la tabla dependiendo de la primera columna, *"Are you self-employed?",* porque si el empleado no pertenece a una empresa, su encuesta es inútil para el ejercicio.

Dado que es una encuesta y la mayoría de respuestas son binarias o están dentro de ciertos grupos cerrados, solo he encontrado un par de "outliers". Estos son, por ejemplo, los valores "3", "323" y "99" en el campo de edad. He decidido borrar las entradas enteras de la base de datos, porque es imposible saber la edad real de esas personas.

En el caso de la pregunta *"Have your observations of how another individual who discussed a mental health disorder made you less likely to reveal a mental health issue yourself in your current workplace?",* los datos nulos los he tratado como "no aplica", porque si no han contestado es probable que sea porque no han observado nada aplicable a esa pregunta.

- Datos mal escritos

En esta sección explicaré cómo he tenido que modificar la tabla para poder pasarla por los scripts de Python.

He tenido varios problemas:

- ❖ Los nombres de las columnas estaban entre comillas… Pero no todos. He solucionado este problema quitando las comillas dobles a toda la tabla.
- ❖ En los nombres de algunas columnas había comas. Por ejemplo, en la pregunta *"If you have a mental health issue, do you feel that it interferes with your work when being treated effectively?"* hay una coma, lo que genera una nueva columna al ser el `csv` un archivo delimitado por comas. Para ello he decidido reemplazar en uno de los scripts los nombres de esas columnas por el mismo nombre, pero sin coma.

Scripts en Python

Para poder realizar todo lo anteriormente mencionado, he escrito unos scripts en Python que hay que ejecutar para adecuar la base de datos para el siguiente paso. Estos son los archivos:

- Archivo `quitarColumnas`

Este archivo contiene las columnas que se van a quitar de la base de datos. Es el primero que debe ejecutarse.

Aquí soluciono también el problema del carácter extraño 'Â', que no fue muy complicado ya que la columna que lo contenía, igualmente, había que borrarla en el proceso de selección. Para esto, simplemente he copiado y pegado el nombre de la columna en el archivo que las borra.

- Archivo `quitarComillas`

El archivo `csv` tenía dentro varios errores de escritura en los identificadores de las columnas: un carácter extraño ('Â') y dos comillas dobles ' " ' al final de una de las cadenas. Para solucionar esto, he escrito un script en Python para quitar todas las comillas de los identificadores de las columnas.

En este script también se suprimen las comas dentro de los nombres de las columnas que generaban más columnas de la cuenta.

- Archivo `limpiarFilas`

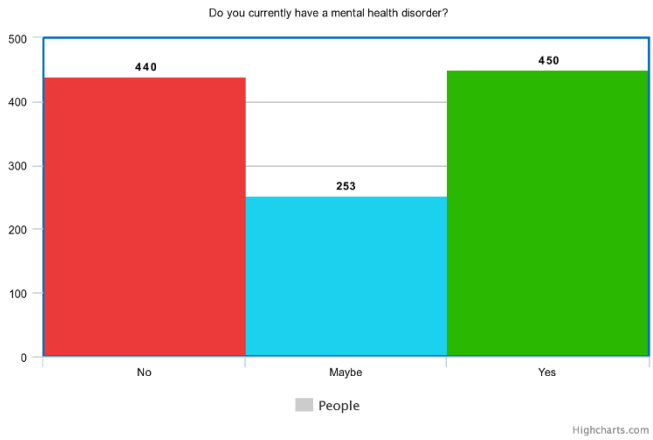
Este archivo quita de la base de datos las filas con los outliers y los datos nulos que no se han podido rellenar.

- Archivo `reemplazarColumnas`

Este archivo es muy importante: es el que contiene todo el proceso de adecuación de la base de datos. En él, se pasan a números y grupos los datos de la encuesta, para posterior interpretación por los algoritmos de Minería de datos.

Para todo el código que he escrito en Python (tanto este de pre-procesado como el archivo con el algoritmo KNN) he usado las librerías `Pandas` (para tratar con la base de datos) y `sklearn` (para el algoritmo). También me he apoyado en algunas como `numpy` y `matplotlib`.

Tras el pre-procesado, la columna *label* se ha quedado así:



2.1.3 *Data Mining*

Para realizar esta fase he decidido hacer uso en gran parte de la página BigML. Esta herramienta me permite subir mi base de datos limpia y transformada, y evaluar diferentes modelos en ella.

Lo primero que hice fue cargar el dataset y apartar los datos para entrenamiento y testing, en proporción de 90% / 10%. Ahora, para

los datos de entrenamiento, elegí varios modelos supervisados, que expongo a continuación.

La medida de rendimiento que vamos a estar teniendo en cuenta va a ser la *accuracy*, porque en este caso, interesa más clasificar bien los datos que entran que tener precisión en una sola clase.

Modelos y evaluaciones

He realizado cuatro modelos diferentes para esta iteración, a fin de encontrar el que mejor resultados diera:

1. Árbol de decisión

El árbol mostró la pregunta más decisiva para la clasificación: *"If you have a mental health issue do you feel that it interferes with your work when NOT being treated effectively?"*, seguida por *"If you have a mental health issue do you feel that it interferes with your work when being treated effectively?"*. Estas dos preguntas son importantes, porque muestran la importancia que la gente da a las enfermedades mentales en su trabajo, y la respuesta que den podría denotar la experiencia que tienen con enfermedades mentales.

Al evaluar el árbol, he obtenido los siguientes resultados:

ACTUAL VS. PREDICTED	0	1	2
0	25	7	6
1	7	9	20
2	8	8	25

Estos muestran un 51.3% de *accuracy* del modelo escogido. Es una cifra bastante mejorable.

2. Algoritmo KNN

También realicé el algoritmo KNN con este conjunto de datos. Estos son los resultados:

Actual VS. Predicted	0	1	2
0	36	3	4
1	11	2	15
2	2	2	40

Este modelo ha salido bastante mejor en las clases 0 ("No") y 2 ("Sí"), pero la clase 1 ("Quizás") ha tenido en general, menos verdaderos positivos, y algunos falsos negativos.

3. Random forest

El algoritmo random forest es llamado "Ensemble" en BigML. Los resultados de este modelo fueron ligeramente mejores que los del árbol de decisión, con una *accuracy* del 54.8%.

ACTUAL VS. PREDICTED	0	1	2
0	28	2	8
1	3	6	27
2	5	7	29

En este modelo se puede observar una reducción general de los fallos en la predicción de la clase 0 (el "No"), pero un aumento considerable de los fallos al evaluar la clase 2 ("Sí") como clase 1 ("Quizás")

4. Regresión logística

BigML tiene una opción, llamada OptiML: un proceso de optimización para la selección y parametrización de los modelos que encuentra automáticamente el mejor de ellos para resolver problemas de clasificación y regresión. El modelo con mejores resultados para mi dataset fue el de regresión logística:

ACTUAL VS. PREDICTED	0	1	2
0	29	2	7
1	7	8	21
2	5	4	32

Scripts

Para esta primera iteración he modificado un script en Python con el algoritmo KNN para adaptarlo a mi dataset. Este script se llama `knn.`

Conclusiones de la iteración

Al ver estos resultados me di cuenta de lo siguiente: el modelo ha fallado demasiado prediciendo como "quizás" las personas respondieron que "Sí" creen que tienen una enfermedad mental (clase 2 predicha como clase 1). Esto me hizo reflexionar sobre la propia pregunta de la encuesta: si a una persona se le pregunta "¿Crees que tienes una enfermedad mental?" y responde "quizás", es que siente que la tiene, pero no está completamente segura. Considero entonces que el algoritmo, al predecir los "sí" como "quizás", no se ha equivocado del todo, porque una persona completamente sana siempre responderá rotundamente "no", pero si ve que tiene algunos síntomas y no está segura de ello dirá que "quizás".

Decidí entonces hacer otra iteración del KDD, juntando los valores de "Sí" y "Quizás" en un solo valor: el 1, que ahora representaría a las personas que, mentalmente, no se encuentran sanas.

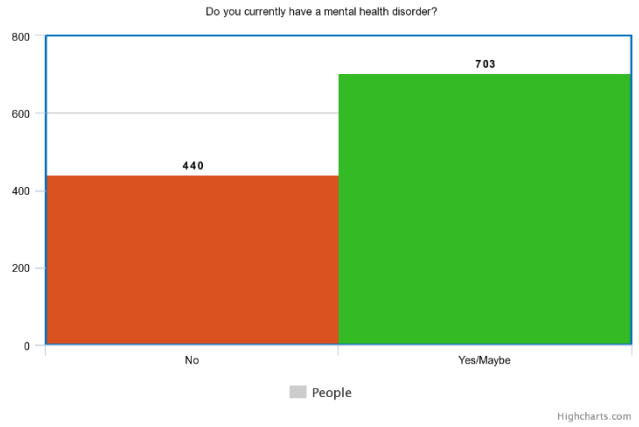
2.2 Segunda iteración

2.2.1 Pre-procesado

El cambio propuesto para esta segunda iteración es el siguiente:

- Juntar las clases 1 y 2 de la *label*, correspondientes a las respuestas "Sí" y "Quizás" de la encuesta.

Tras el cambio, los valores de la columna *label* han quedado así:



2.2.2 Data Mining

Modelos y evaluaciones

1. Árbol de decisión

Este es el modelo resultado del árbol de decisión para esta segunda iteración:

ACTUAL VS. PREDICTED	0	1
0	32	22
1	10	51

En general, el árbol de decisión tiene bastante mejores resultados en la clase 1, pero sigue teniendo demasiados falsos negativos (22 falsos negativos respecto a los 32 verdaderos positivos) en la clase 0.

2. Algoritmo KNN

A continuación, se muestra la matriz de confusión del algoritmo KNN con dos clases:

Actual VS. Predicted	0	1
0	28	15
1	5	67

Podemos observar que el algoritmo tiene problemas clasificando los valores "No", puesto que ha clasificado 15 valores "No" como "Sí". Básicamente lo mismo que le pasaba al árbol de decisión.

3. Random forest

ACTUAL VS. PREDICTED	0	1
0	33	21
1	5	56

Una vez más, el random forest obtiene mejores resultados en el rendimiento… Pero en la clase 1. La clase 0 sigue fallando casi un 40% de las predicciones que hace.

4. Regresión logística

De nuevo, he vuelto a usar la herramienta OptiML para ver qué modelo con qué parámetros tiene un mejor resultado, y de nuevo ha sido la regresión logística.

Sin embargo, los resultados que da son muy parecidos a los del random forest, con tan solo UNA buena predicción más de cada clase.

ACTUAL VS. PREDICTED	0	1
0	32	22
1	4	57

Conclusiones de la iteración

Aunque generalmente ha dado mejores resultados, se han debido a que, en vez de tres clases, ahora tengo dos, y considero que una proporción de acierto de poco más de 2/3 es bastante mejorable. Sin embargo, como los resultados tampoco son malos del todo, he decidido darle otra oportunidad y hacer una tercera iteración, esta vez, pasando la *label* a las preguntas que vienen después de esta:

- *If yes, what condition(s) have you been diagnosed with?*
- *If maybe, what condition(s) do you believe you have?*

Ahora, el objetivo será predecir si la persona cree que tiene una enfermedad mental, y en el caso de que tenga, averiguar cuál.

En realidad, para los algoritmos lo que predeciremos será qué enfermedad mental tiene la persona, porque las clases serán las enfermedades mentales posibles, y una clase aparte con las personas que han respondido que "No" a la pregunta anterior, *"Do you currently have a mental health disorder?"*

Para ello, voy a juntar las dos columnas anteriormente mencionadas. Las respuestas de estas dos columnas dependen directamente de las dadas en la de *"Do you currently have a mental health disorder?"*: si han dicho que "No" tienen una enfermedad mental, en la columna resultado de sumar esas dos habrá un valor nulo. Si han dicho que "Sí" o "Quizás", en la columna resultado saldrá la enfermedad que tienen/creen tener.

2.3 Tercera iteración

2.3.1 Pre-procesado

En el script `reemplazarColumnas` de esta iteración he dividido las enfermedades mentales en 9 grupos:

- Anxiety disorder - 0
- Attention Deficit Hyperactivity Disorder - 1
- Autism Spectrum Disorder – 2
- Eating disorder - 3
- Mood disorder - 4
- Obsessive-Compulsive Disorder - 5
- Personality Disorder - 6
- Post-traumatic Stress Disorder - 7
- Psychotic Disorder - 8

- Otros (aquí he puesto las enfermedades mentales que solo había puesto una persona, puesto que no merecía hacer un grupo solo para esa) – 9
- Ninguna enfermedad mental (respondió "No" a la pregunta *"Do you currently have a mental health disorder?")*

Además, he borrado la columna que anteriormente era la *label*, puesto que agrupar a las personas en la clase 10 sería instantáneo, ya que estas dos nuevas columnas dependen de ella.

Como no soy una experta en este tema, he hecho los grupos después de una breve investigación en internet sobre las distintas enfermedades mentales. Además, como son respuestas múltiples y para reducir la complejidad de los grupos, he agrupado las respuestas que tienen varias enfermedades según la primera. Por ejemplo, "'Mood Disorder (Depression, Bipolar Disorder, etc)|Personality Disorder (Borderline, Antisocial, Paranoid, etc)'," la he puesto en "Mood disorder". Esta aproximación me ha parecido correcta porque, en una lista, siempre se tiende a decir lo más importante primero, así que considero que esa persona tiene más parte de "Mood disorder" que de "Personality disorder".

2.3.2 Data Mining

Modelos y evaluaciones

1. Árbol de decisión

	0.0	1.0	10.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0
0.0	30	2	7	0	0	10	0	0	1	0	1
1.0	2	0	2	0	0	0	0	0	0	0	0
10.0	4	0	30	0	0	2	0	0	0	0	0
2.0	0	0	0	0	0	0	0	0	0	0	0
3.0	0	0	0	0	0	1	0	0	0	0	0
4.0	6	2	3	0	0	5	0	0	0	0	1
5.0	0	0	1	0	0	0	0	0	0	0	0
6.0	0	0	0	0	0	0	0	0	0	0	0
7.0	0	0	0	0	0	0	0	0	0	0	0
8.0	1	0	0	0	0	0	0	0	0	0	0
9.0	1	1	0	0	0	2	0	0	0	0	0

Este nuevo enfoque tiene una *accuracy* del 69.6%... Es mejorable, pero no es un mal resultado. Sin embargo, ocurre lo esperado: no clasifica bien ninguna de las clases que apenas tienen datos.

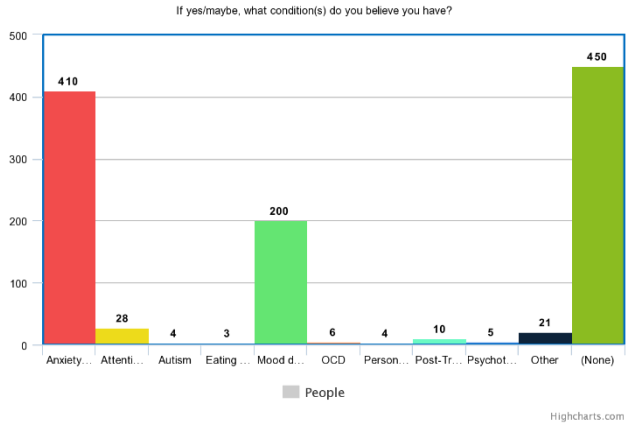
Scripts

Para hacer la clasificación de los grupos, he modificado el script `reemplazarColumnas`, creando a partir de él uno nuevo: `reemplazarColumnas2Ej`, donde he borrado la anterior columna *label ("Do you currently have a mental disorder?")* y he renombrado las dos columnas unidas como la nueva *label*.

He creado una versión para la segunda ejecución de todos los scripts.

Conclusiones de la iteración

Nada más ver la matriz de confusión del árbol de decisión me he dado cuenta de que esta iteración no ha sido del todo buena idea. Tal y como muestra el histograma, hay demasiadas clases, y muchas de ellas apenas tienen datos:



He decidido no sacar ningún modelo más, y reducir drásticamente los grupos de las enfermedades mentales a 3:

- Anxiety disorder / mood disorder
- Otra enfermedad
- Ninguna enfermedad

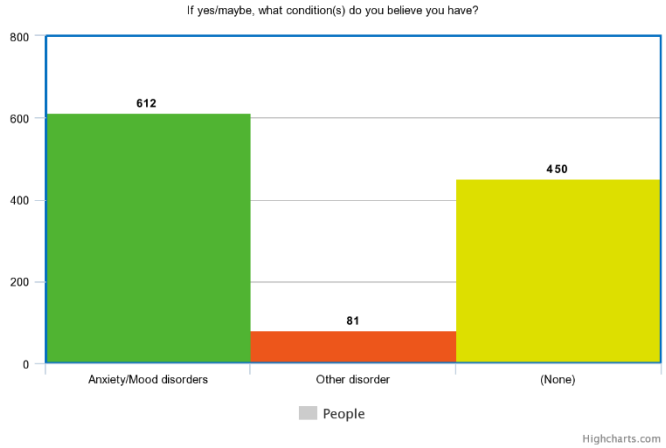
2.4 Cuarta iteración

2.4.1 Pre-procesado

He agrupado las enfermedades mentales en los tres grupos mencionados en la conclusión de la iteración anterior:

- Ansiedad/Desorden del humor - 0
- Otra enfermedad – 1
- Ninguna enfermedad – 2

Así, los datos de la columna *label* han quedado repartidos de la siguiente forma:



Nada más ver el histograma de los datos totales que se quedan en cada grupo, creo que los del grupo "Other disorder" no los va a clasificar bien. Sin embargo, he decidido dar una oportunidad a los algoritmos, ya que el pre-procesado está terminado.

2.4.2 *Data Mining*

Modelos y evaluaciones

1. Árbol de decisión

ACTUAL VS. PREDICTED	0.0	1.0	2.0
0.0	39	7	20
1.0	4	0	1
2.0	9	1	34

Podemos ver que no se ha predicho ningún dato correctamente de la clase 1.

2. Algoritmo KNN

Esta es la matriz de confusión resultado de la ejecución del algoritmo KNN en este tercer dataset:

Actual VS. Predicted	0	1	2
0	58	0	5
1	7	0	1
2	12	0	32

Parece que, efectivamente, y tal y como pasa en el árbol de decisión, el algoritmo tiene problemas clasificando como "No" varias de las personas con "Anxiety/Mood disorder". Además, no ha conseguido clasificar bien ninguno del grupo "Other mental disorders".

O bien el KNN no es capaz de diferenciar entre ansiedad/trastorno del humor y otras enfermedades mentales, o bien faltan datos (esta pregunta está claramente desbalanceada). Yo me inclinaría por la segunda opción, puesto que, aunque he agrupado los datos, sigue habiendo una diferencia abismal entre la cantidad de datos de los dos grupos.

3. Random forest

Estos son los resultados de la ejecución del random forest:

ACTUAL VS. PREDICTED	0.0	1.0	2.0
0.0	59	0	7
1.0	5	0	0
2.0	7	0	37

Esta vez, el random forest ha dado una *accuracy* del 83.5%, ¡la mejor hasta ahora! Personalmente pienso que una *accuracy* del 83.5% en una encuesta con tan pocos datos está bastante bien, aunque seguimos con el mismo problema: no sabe clasificar los datos de la clase 1: "*Other mental health disorder*s".

4. Regresión logística

Esta vez, la regresión logística que ha sacado el OptiML no ha sido la mejor opción, sino, en esta iteración, el random forest, que ha acertado una predicción más en la clase 2.

ACTUAL VS. PREDICTED	0.0	1.0	2.0
0.0	59	0	7
1.0	5	0	0
2.0	8	0	36

Scripts

He hecho una nueva versión del script `reemplazarColumnas` de la iteración anterior, porque no quería sobrescribirlo, para juntar la *label* en las 3 clases.

Conclusiones de la iteración

Creo que puedo afirmar, sin duda, que este nuevo enfoque es el que mejor funciona, con una *accuracy* en la regresión logística del 83.5%. Sin embargo, me sigue preocupando que no clasifique bien ningún dato de la clase 1, aunque entiendo que es porque está completamente desbalanceada.

3. Conclusiones generales

Al principio comencé este trabajo porque me parecía muy interesante intentar saber si, dependiendo de lo que pensaba una persona sobre cómo se llevan las enfermedades mentales en su ambiente laboral, se podría saber si tiene una enfermedad mental. Preguntas como "Le contarías a tu supervisor que crees que tienes una enfermedad mental", que denotan la confianza que se tiene para hablar de enfermedades mentales en el ambiente laboral; o "¿Tu empresa tiene planes de concienciación sobre las enfermedades mentales?", que denotan el compromiso que tiene la empresa con este tipo de enfermedades.

Al principio todas las preguntas me parecían bastante buenas, hasta que vi la pregunta más decisiva sacada por el primer árbol de decisión:

"If you have a mental health issue do you feel that it interferes with your work when NOT being treated effectively?"

Esto me hizo reflexionar bastante sobre cómo iban a funcionar los algoritmos con la base de datos. Si se piensa, una pregunta así denota **experiencia**. Las personas que responden que sí, piensan que la enfermedad mental interfiere con el trabajo cuando no se trata, es porque quizá lo estuvieran sufriendo en ese momento. ¡Puede que no!, pero considero que esa pregunta es un buen comienzo.

En este trabajo he hecho dos acercamientos para aplicar Minería de datos a la base de datos:

1) Clasificar por "*Do you currently have a mental disorder?"*

Es la primera idea que se me ocurrió, y de la que iba a ir en principio todo el proceso.

Esta clasificación no dio resultados perfectos, pero se acercó mucho gracias a la segunda iteración, que ya dio resultados deseables: la *accuracy* del mejor modelo, la regresión logística, fue 77.4%.

Como no consideraba que los resultados fueran malos, quería probar si podía clasificar el tipo de enfermedad mental de la gente que creyera tener alguna.

2) Clasificar juntando las dos columnas *"If yes, what condition(s) have you been diagnosed with?"* y *"If maybe, what condition(s) have you been diagnosed with?"*

Me parecía interesante ver si se podía diferenciar entre las distintas enfermedades mentales que la gente creía tener, puesto que es un acercamiento bastante parecido al anterior, solo que, vaya, con más de dos clases.

También hice dos iteraciones, una con las 10 clases que suponía clasificar las distintas respuestas que dio la gente, y otra juntando las dos clases con más datos y todas las demás en una.

El principal problema que vi con este enfoque fue la clasificación de las respuestas en la fase de pre-procesado de la tercera iteración. Ciertamente, era un campo múltiple donde la persona podía poner tantas respuestas como considerara oportunas. Entonces, ¿en qué clase debía meter una respuesta denotada como *"Mood disorder | Anxiety disorder | Eating disorder"*? Pasé las respuestas múltiples a las clases según la primera enfermedad, sin embargo, un/a profesional de la psicología probablemente sabría, de esas tres, cuál es la más importante o la posible desencadenante de las demás, y ponerla entonces en la clase que verdaderamente le corresponde.

Por eso, pienso que el problema de que las clases estuvieran desbalanceadas era porque yo no hice bien la clasificación en primera instancia. Aunque también era debido a que había pocos datos de la mayoría de las clases, porque son enfermedades mentales mucho más raras que la ansiedad y el desorden del humor.

Aquí notamos, de nuevo, cómo de desbalanceadas estaban las clases en las dos últimas iteraciones.

Efectivamente, eso hizo que los modelos no clasificaran bien ninguna de los desórdenes mentales que no fueran ni ansiedad ni trastornos del humor.

Puedo concluir, entonces, que aunque me hubiera gustado que los resultados hubieran pasado del 90% en *accuracy*, creo que no son del todo malos y que el hecho de cómo se ven las enfermedades mentales en el ambiente laboral, dice mucho de cómo se sienten las personas que trabajan en él.

Finalmente, me gustaría terminar comentando que este trabajo me ha parecido muy interesante. He aprendido mucho, y me ha parecido increíble lo que se puede sacar de una base de datos gracias a la Minería de datos. ¡Es muy útil!

A continuación, muestro unas ideas para trabajos futuros sobre esta misma base de datos.

4. Propuestas futuras

Tras terminar este proyecto, tengo varias propuestas para futuros desarrollos.

4.1 Aumentar el tamaño de la base de datos

Después de haber visto la evolución del proyecto, he llegado a la conclusión de que la base de datos inicial no era pequeña, pero ahora tampoco creo que fuera adecuada para pasar por un proceso de Minería de datos.

Dado que es evidente que la encuesta se hizo on-line a un grupo grande de personas, propongo rehacerla contando con el triple, o incluso con el cuádruple de personal.

4.2 Hacer más iteraciones

Mi segunda propuesta es que se podrían hacer más iteraciones, puesto que los algoritmos no dan un resultado ejemplar. Se podría estudiar pasar, desde la última iteración, la clase que contiene tanto "Mood disorder" como "Anxiety disorder" de la label a clases distintas. Sin embargo, reitero la palabra "estudiar", porque creo que debido a los pocos datos que se tienen de "Other mental disorders" no creo que ello aumentase mucho la confianza de los algoritmos a la hora de clasificar.

4.3 Clasificar mejor algunas columnas

Como he mencionado en el pre-procesado de la iteración 3, hay algunas preguntas que permiten respuestas múltiples, y eso conlleva problemas en su clasificación: ¿en qué grupo se pueden meter? Hay varias preguntas con este problema:

- If yes, what condition(s) have you been diagnosed with?
- If maybe, what condition(s) do you believe you have?
- Which of the following best describes your work position?

La tercera de estas preguntas iba a ser una columna a tener en cuenta desde la primera ejecución. Sin embargo, no había forma de poder ajustar correctamente los datos a clases, porque hay personas que han respondido con 5 puestos de trabajo diferentes.