Imagen que contiene Texto

Descripción generada automáticamente

Instituto tecnológico de Hermosillo.

maestro hinojosa palafox eduardo antonio.

alumno rodriguez carrillo luis armando.

fecha 15 de diciembre 2022.

Proyecto final.

mineria de datos.

**Introducción:**

Este proyecto final de minería de datos se trata de un problema de clasificación, al buscar varios dataset, se termino eligiendo un dataset de pacientes con diabetes [1], para poder así predecir posibles personas con diabetes.

Se encontrará una breve descripción del problema a desarrollar, una descripción del conjunto de datos que explica el significado de las columnas del dataset, una descripción de la solución propuesta y una descripción del código utilizado en colab o sea código Python.

**Descripción del problema a desarrollar:**

Hoy en día existen muchas personas con diabetes, existen ciertos patrones que presentan los pacientes con diabetes, que hace que esta enfermedad pueda ser predicha por un algoritmo. Lo que busca este algoritmo es que dependiendo de ciertos criterios buscar quien puede tener diabetes, por que puede ser que prediga que tiene diabetes y esta persona se puede hacer estudios y no tener diabetes, por lo tanto, estos pacientes se pueden tratar aun; esto es muy importante ya que la diabetes es una enfermedad crónica y el poder detectarla puede ser de gran ayuda.

**Descripción del conjunto de datos:**

* Pregnancies: Es el número de embarazos.
* Glucose: Nivel de glucosa en sangre.
* BloodPressure: Medida de la presión arterial.
* SkinThickness: El grosor de la piel.
* Insulin: Nivel de insulina en sangre.
* BMI: Índice de masa corporal.
* DiabetesPedigreeFunction: Porcentaje de diabetes.
* Age: La edad.
* Outcome: Variable de clasificación, nos indica si tiene diabetes o no (1 es si y 0 es no).

**Descripción de la solución propuesta:**

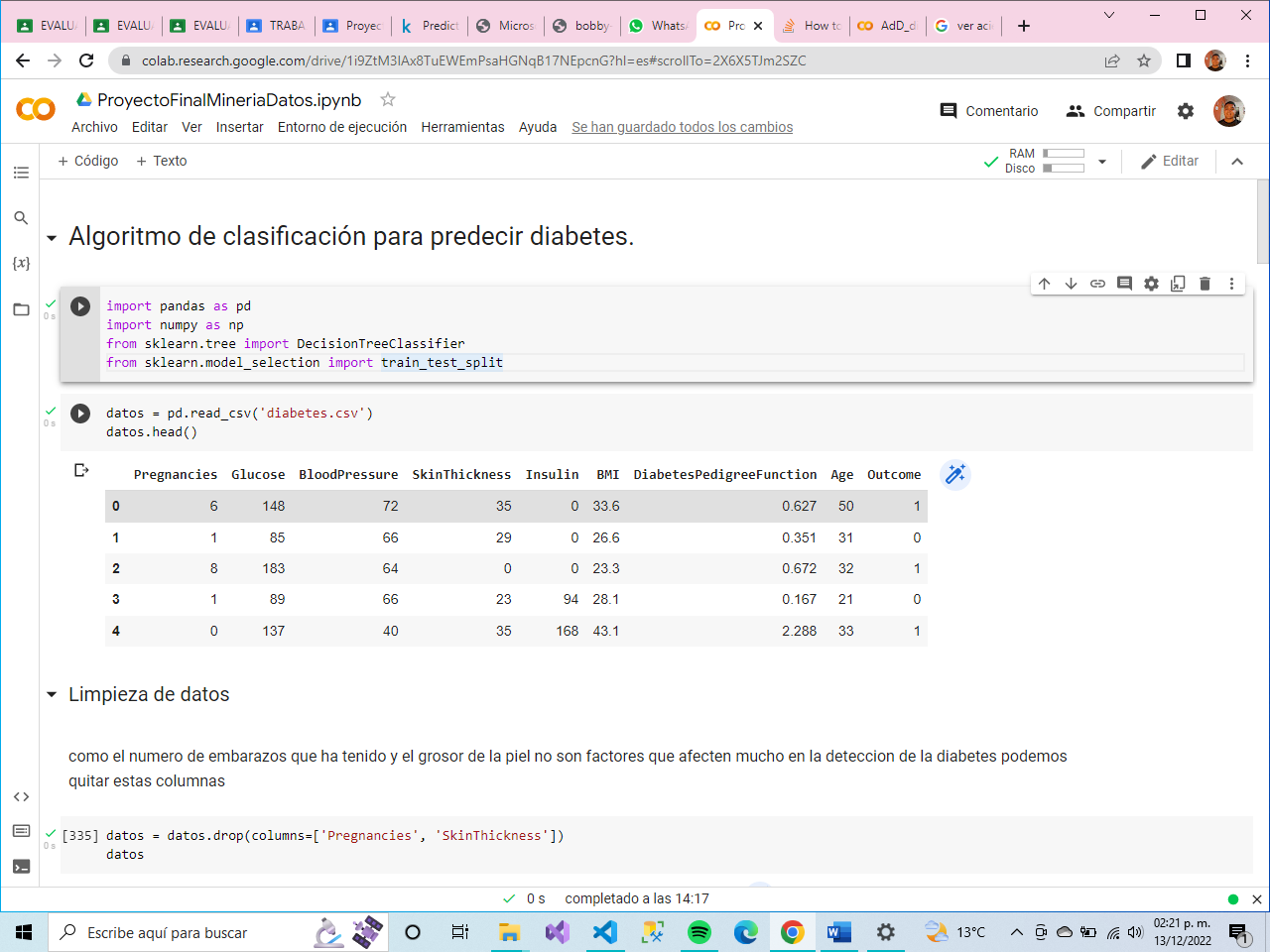
Para poder solucionar este problema de clasificación debemos:

1. Reconocimiento de los datos.
2. Limpieza de los datos.
3. Reconocer y separar nuestros atributos de nuestra variable de clasificación.
4. Segmentar los datos en datos de prueba y de entrenamiento.
5. Implementación del modelo de clasificación para nuestros datos de entrenamiento (en este caso el modelo de árbol de decisión).
6. Visualización de las métricas.
7. Implementar nuestro modelo de clasificación para nuestros datos de prueba.
8. Visualizar las métricas de prueba.
9. Comprobar nuestro modelo usando datos nuestros.

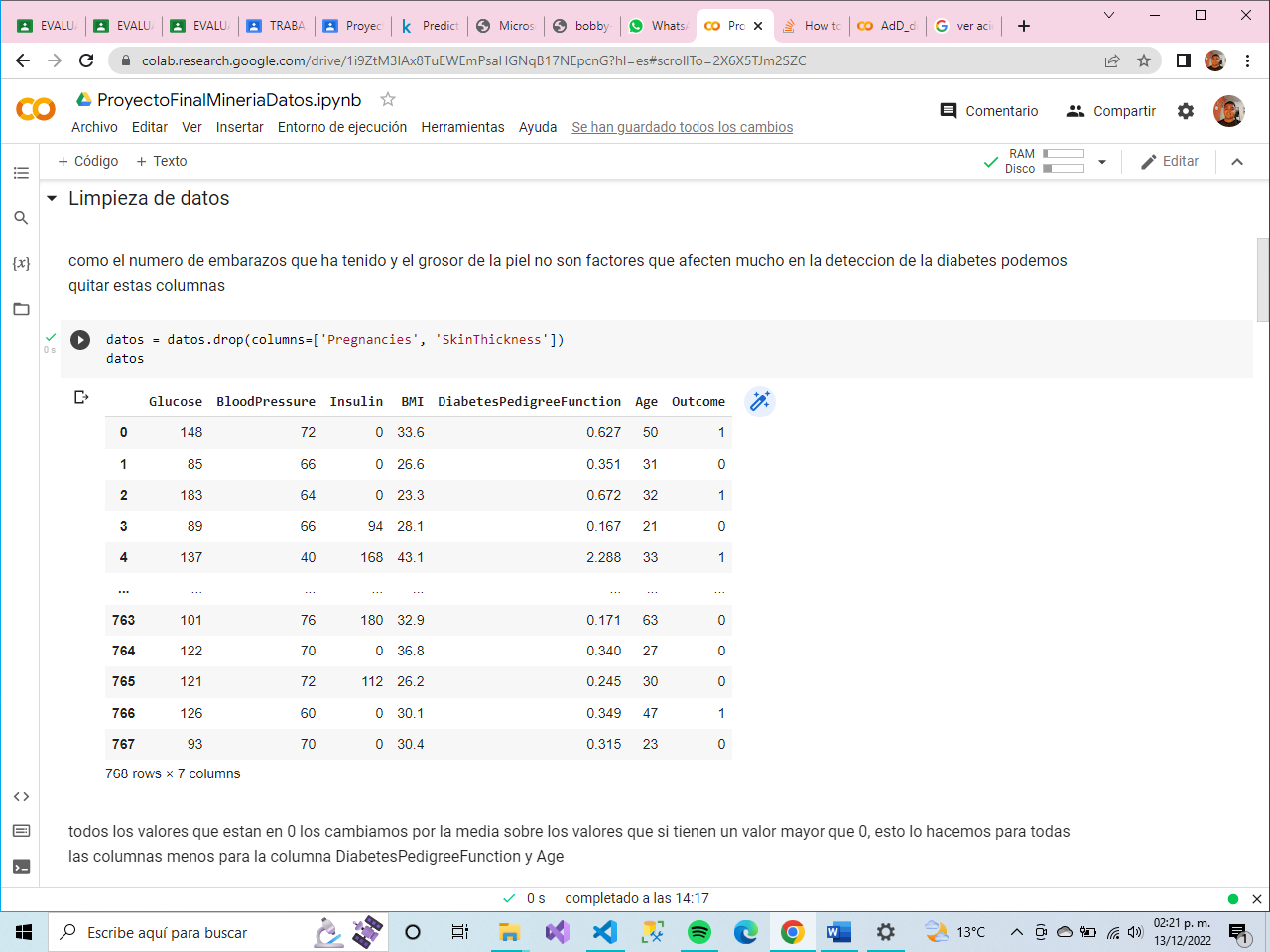
Para poder solucionar este problema utilizaremos el modelo de árbol de decisión que nos ayudara a crear un algoritmo de aprendizaje para poder predecir si una persona tiene diabetes o no.

**Descripción del código utilizado**

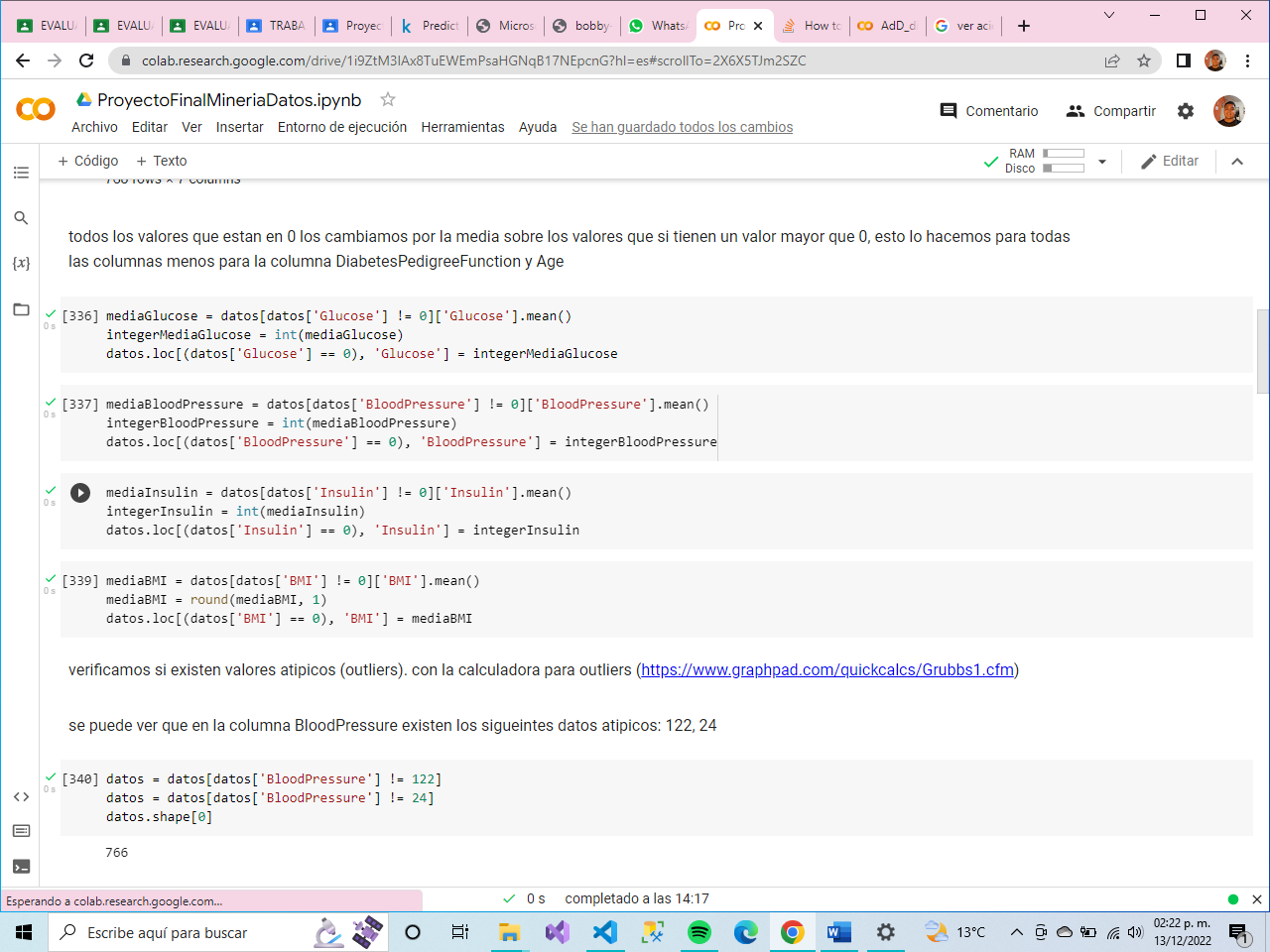
Importamos las librerías de pandas y numpy que nos ayudarán a trabajar con grandes cantidades de datos. Así también importamos



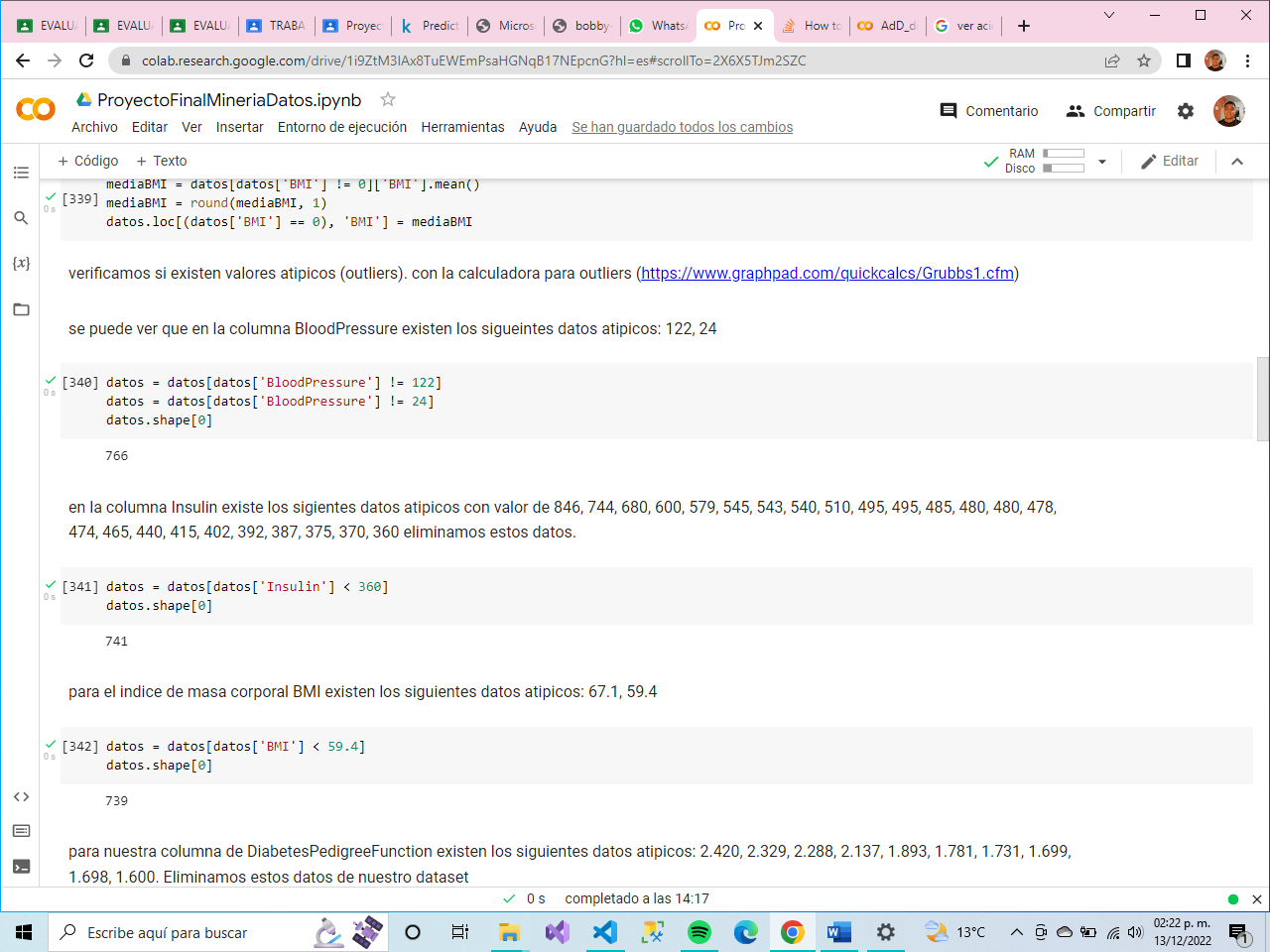
Ahora hacemos una limpieza de nuestro dataset. Primeramente quitamos las columnas Pregnancies y SkinThichness

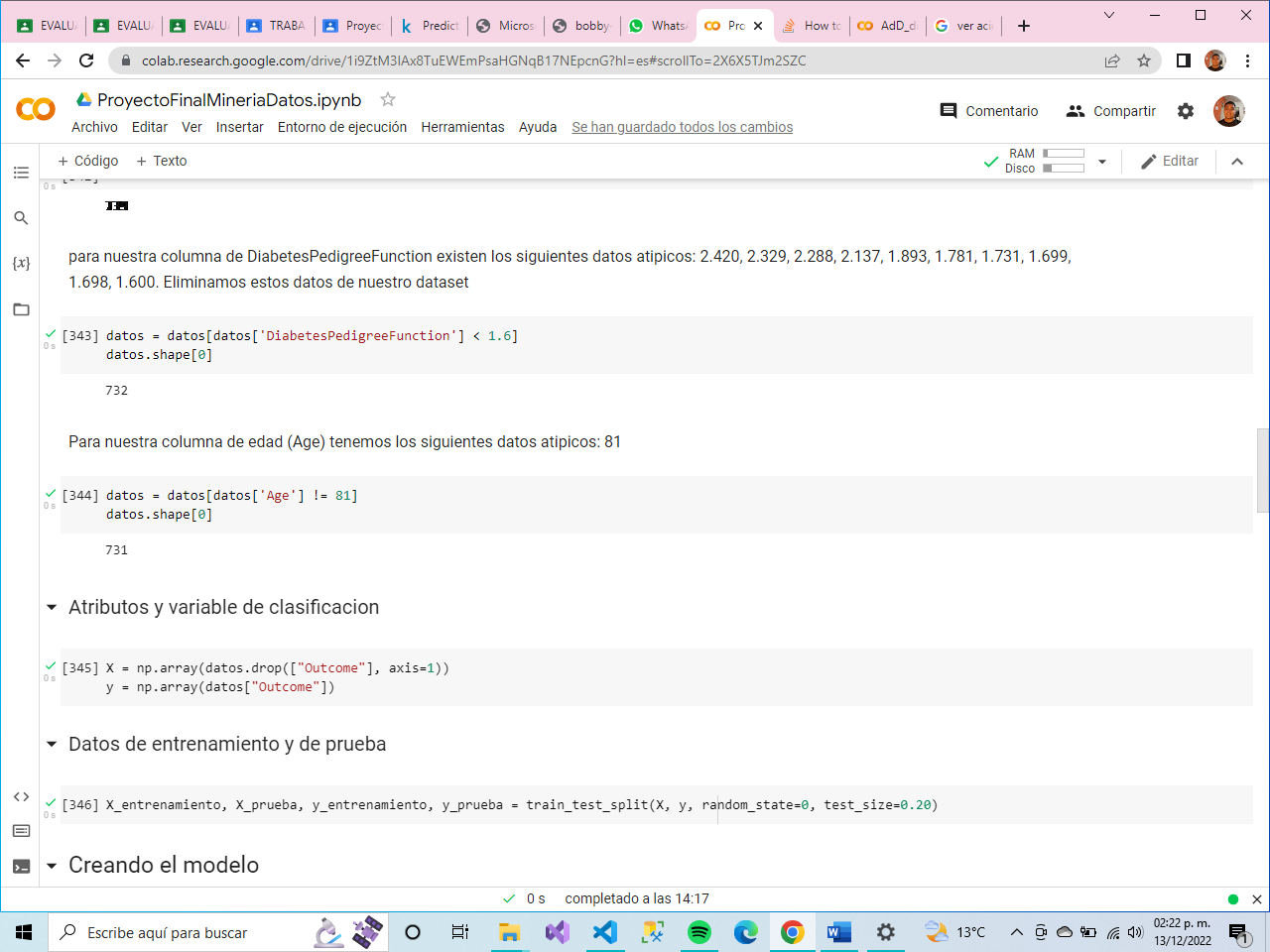


Con el siguiente código, evitamos eliminar los datos que están en 0 y los cambiamos por la media, tomando en cuenta en la media solo los datos que tienen valores diferentes a 0

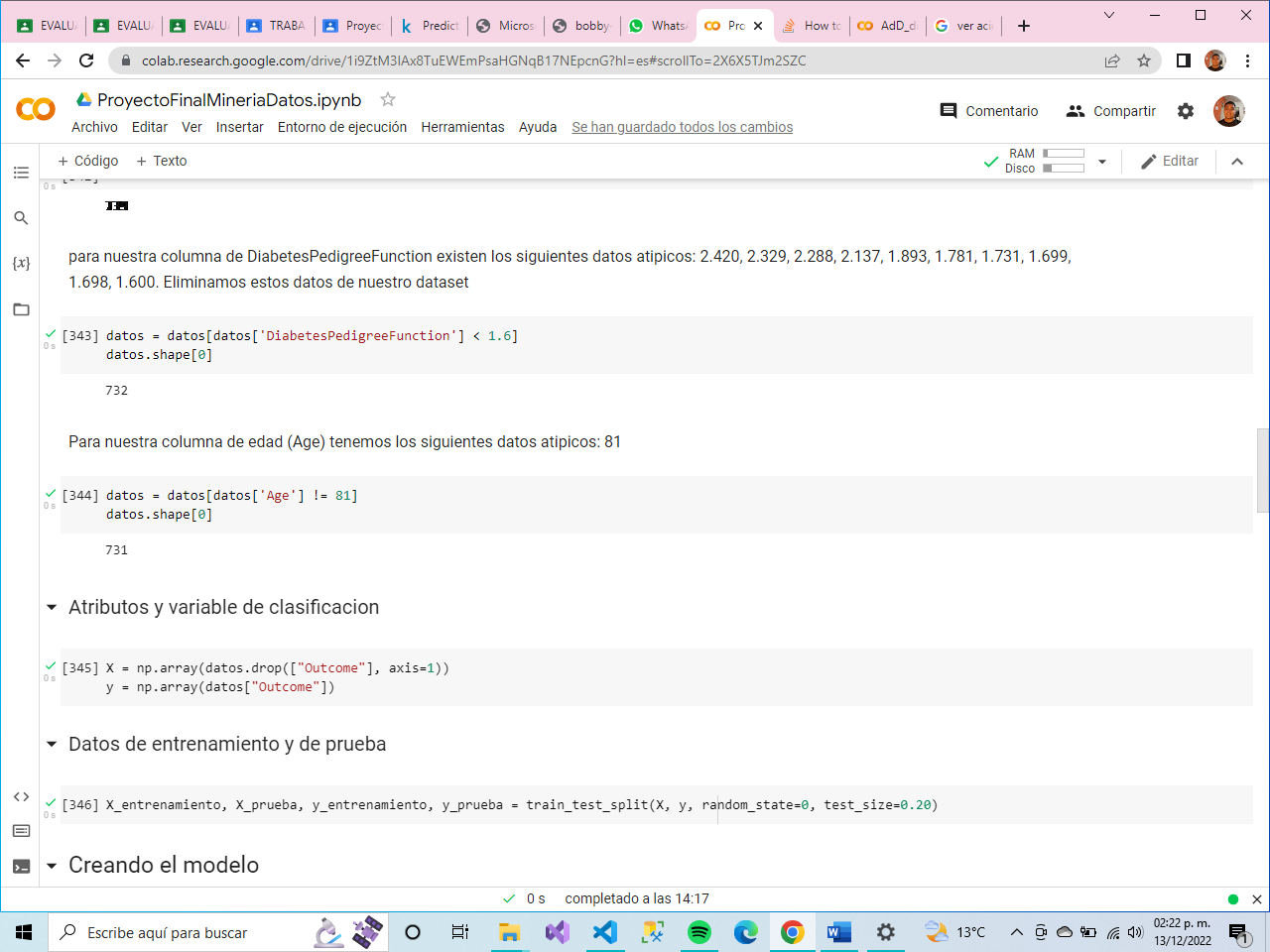


Para buscar valores atípicos utilizamos la calculadora graphpad [2] y eliminamos esos valores como se muestra en el siguiente código

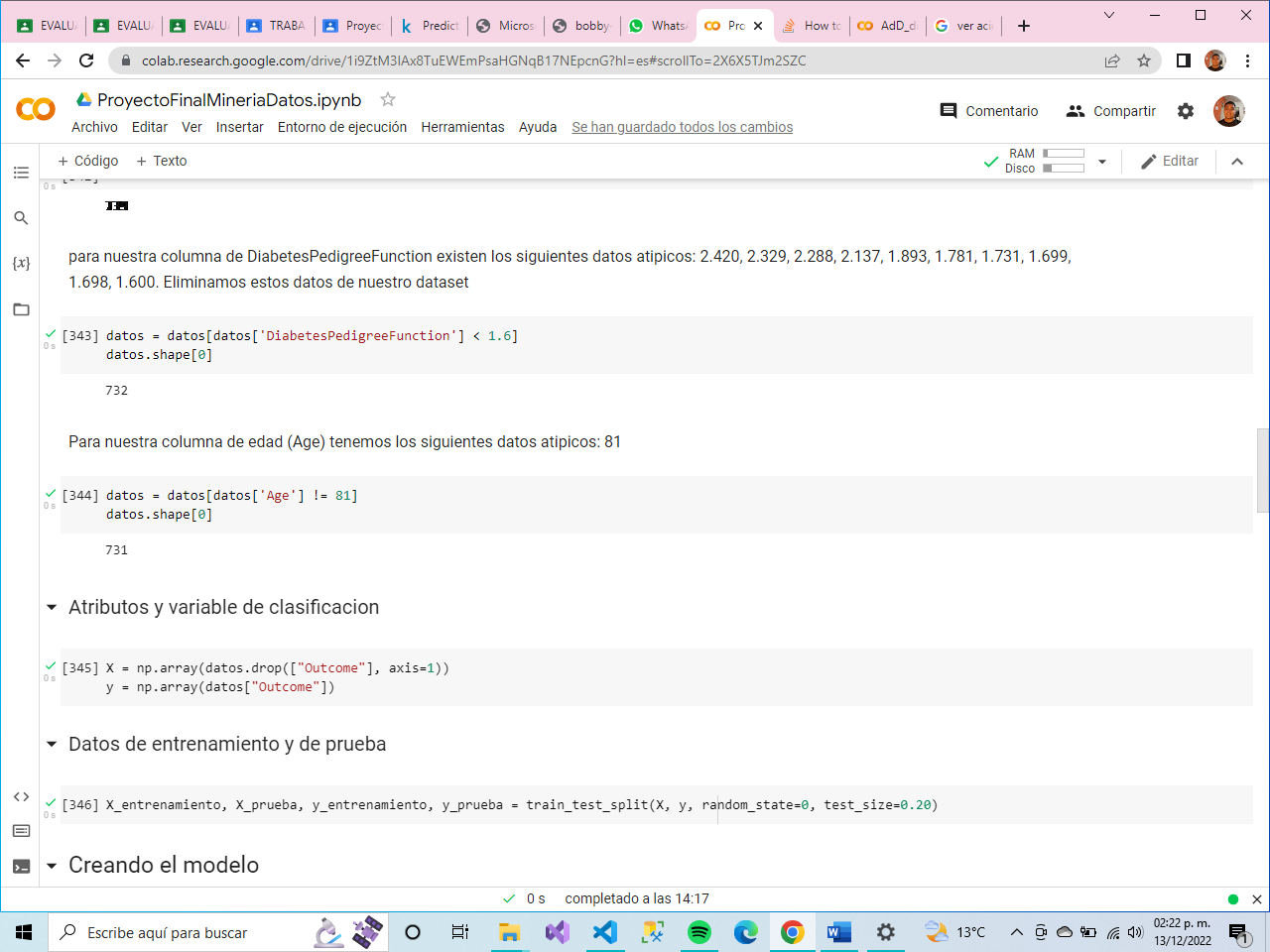




Ahora separamos nuestros atributos y nuestra variable de clasificación, los atributos serán los datos que ocupa el modelo para reconocer patrones o similitudes y predecir y nuestra variable de clasificación será la columna que nos dice si el paciente tiene diabetes o no.



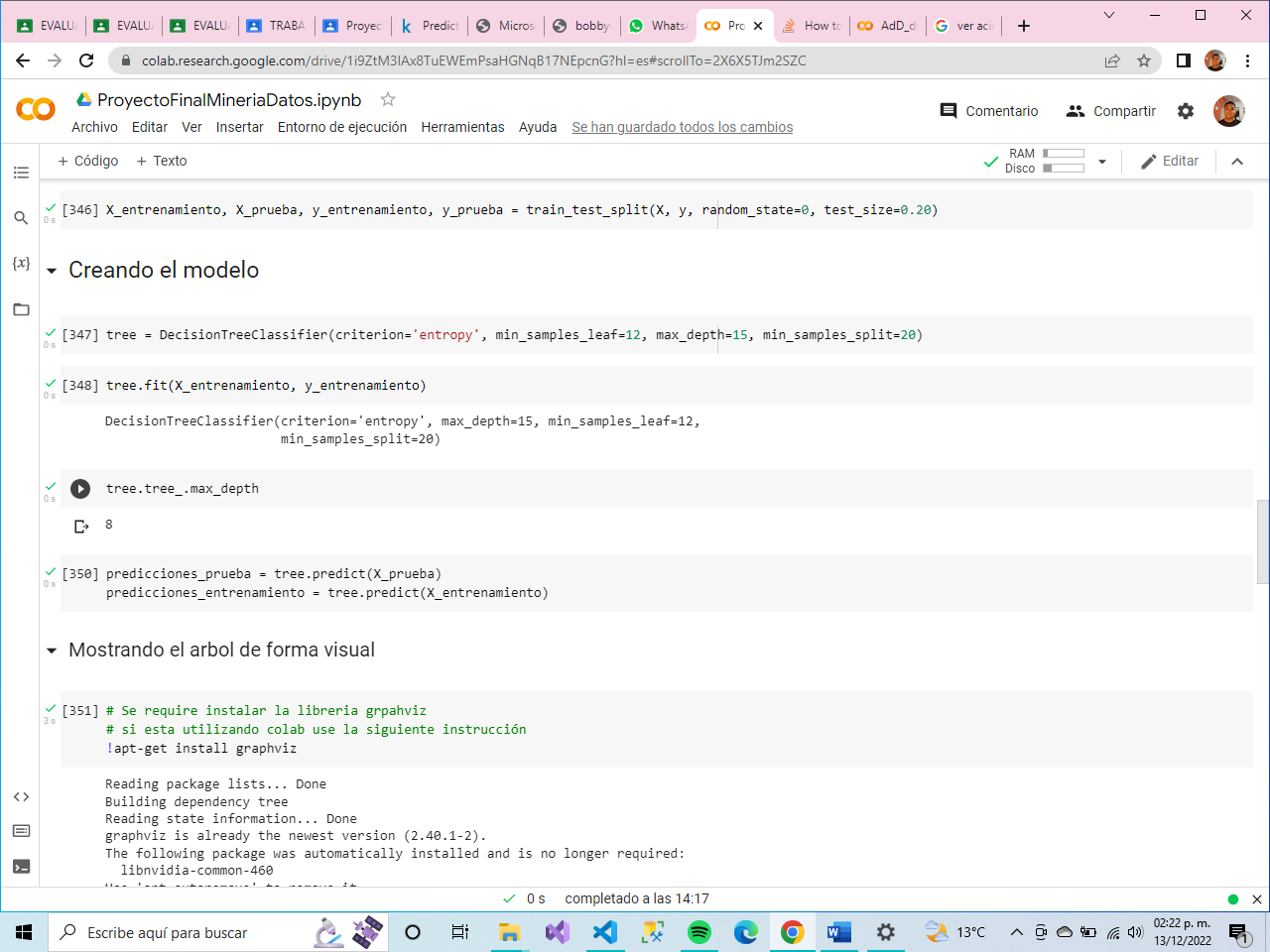
Al utilizar modelos de entrenamiento se recomienda reservar de un 10% a un 30% de los datos para utilizarlos como datos de prueba, es decir, entrenar al modelo con 90%-70% de los datos, para ello utilizaremos la función train\_test\_split() para poder reservar un 20% de los datos de prueba como se muestra en el siguiente código



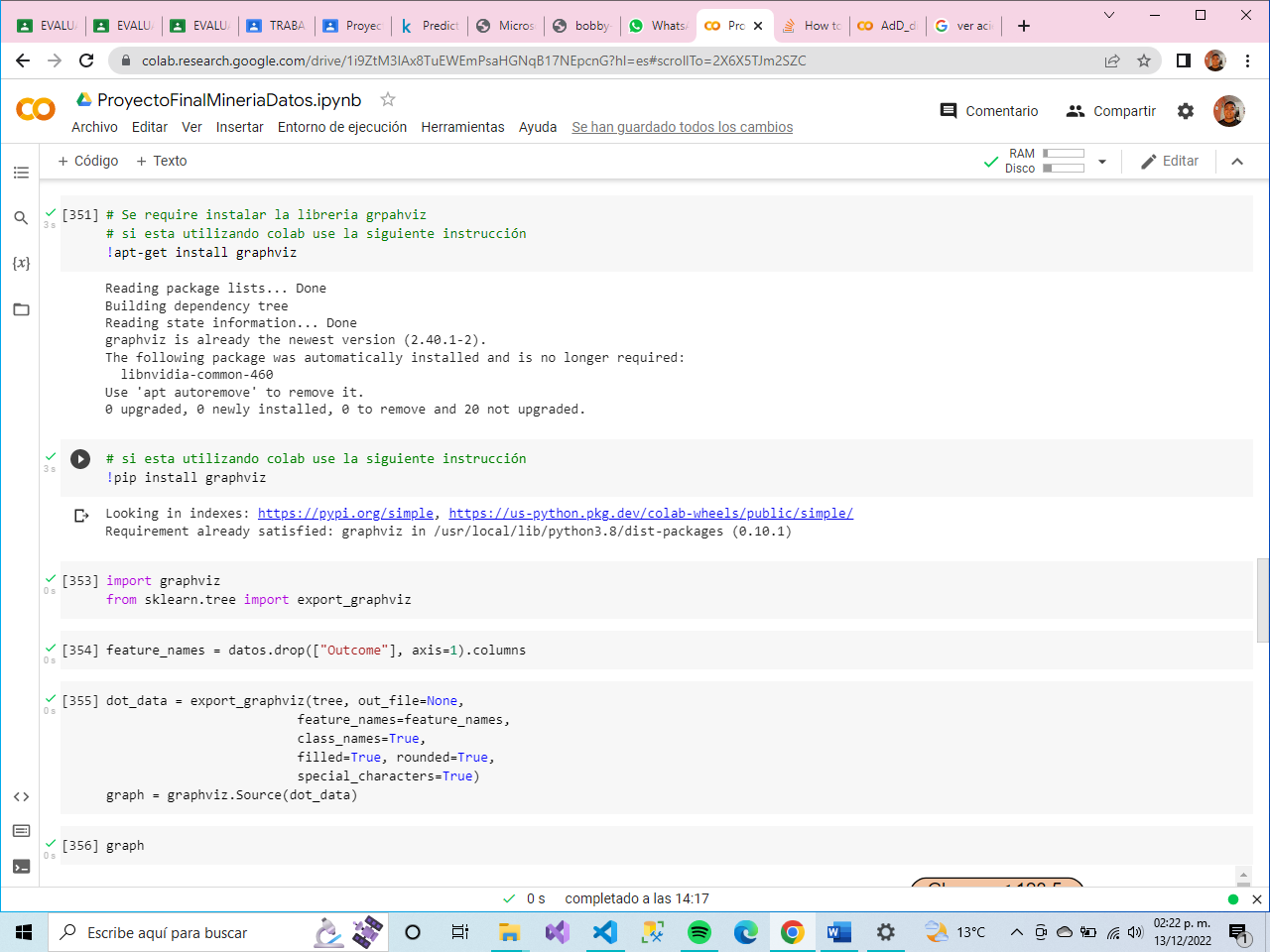
Ahora creamos nuestro modelo de arbol de decisión utilizando la función DecisionTreeClassifier().

Pasandole como parametro criterion (la función para medir la calidad de una división) igual a ‘entropy’, min\_samples\_split (El número minimo de muestras requeridas para estar en un nodo hoja) igual a 12, max\_depth (la profundidad maxima del árbol) igual a 15 y min\_samples\_split (el numero minimo de muestras requeridas para dividir un nodo interno).

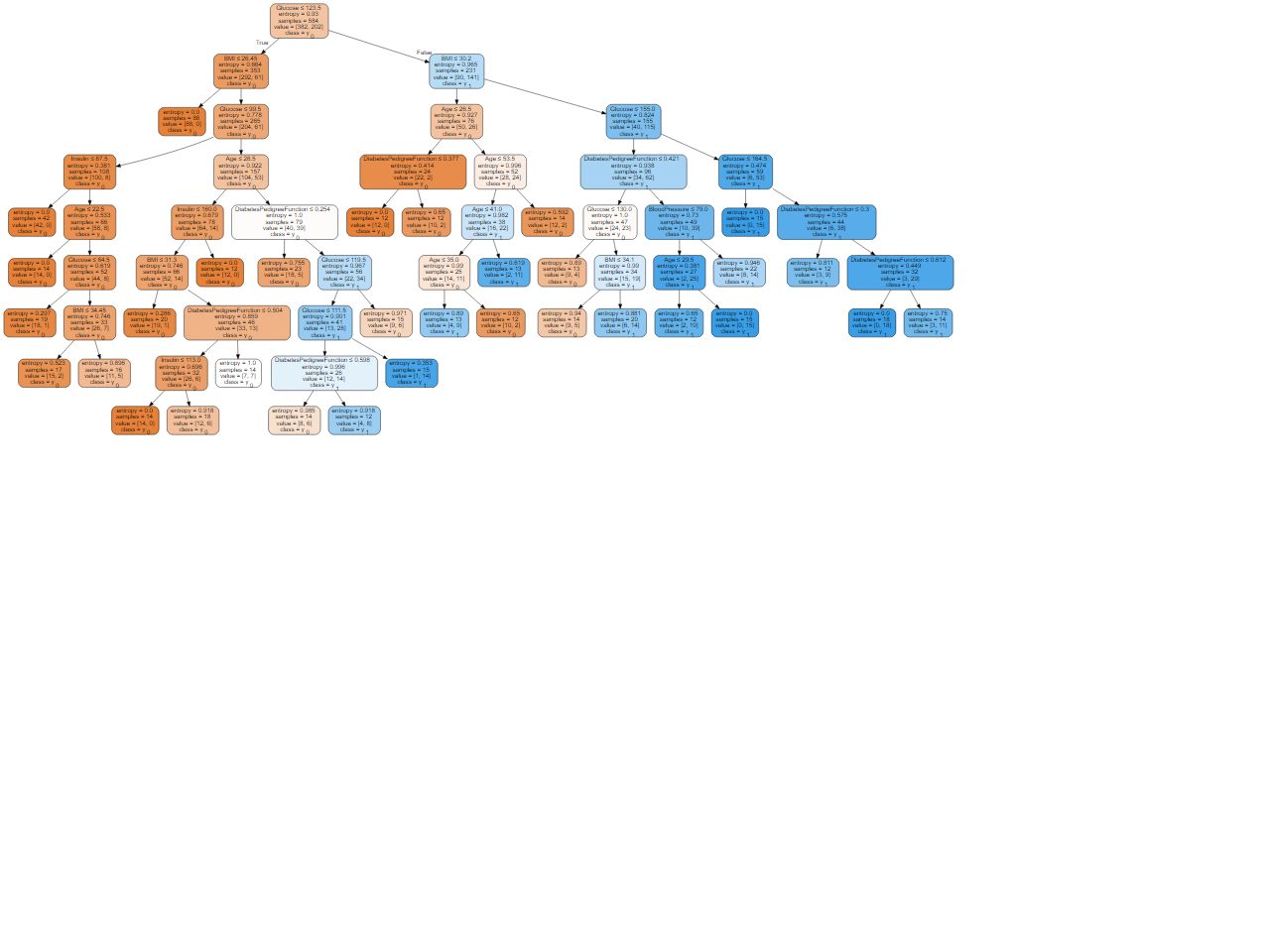
Ajustamos el modelo a nuestros datos de entrenamiento o en otras palabras, entrenamos a nuestro algoritmo para que aprenda con los datos de entrenamiento



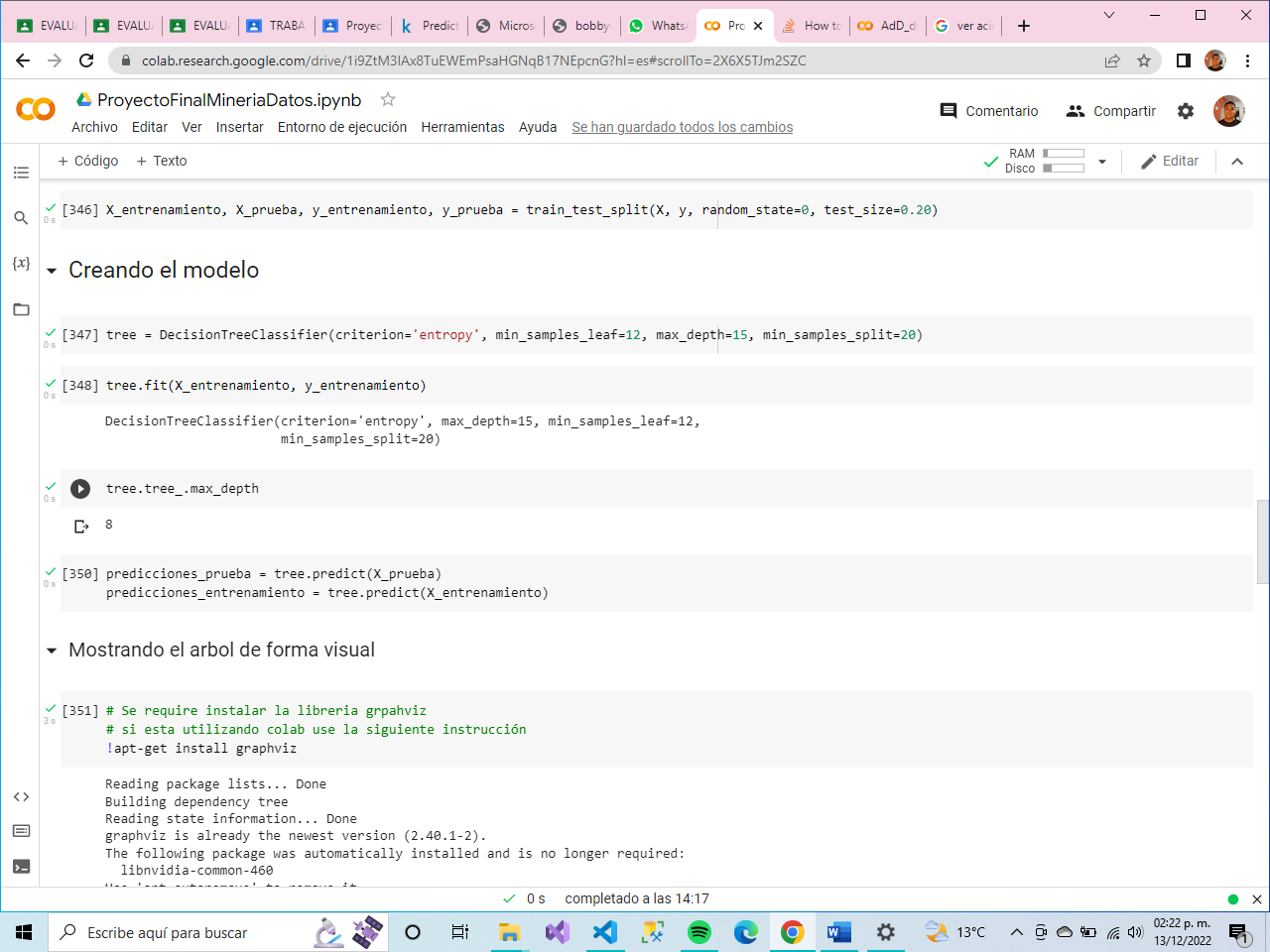
Con el siguiente código podemos ver como quedo nuestro árbol de decisión de forma visual



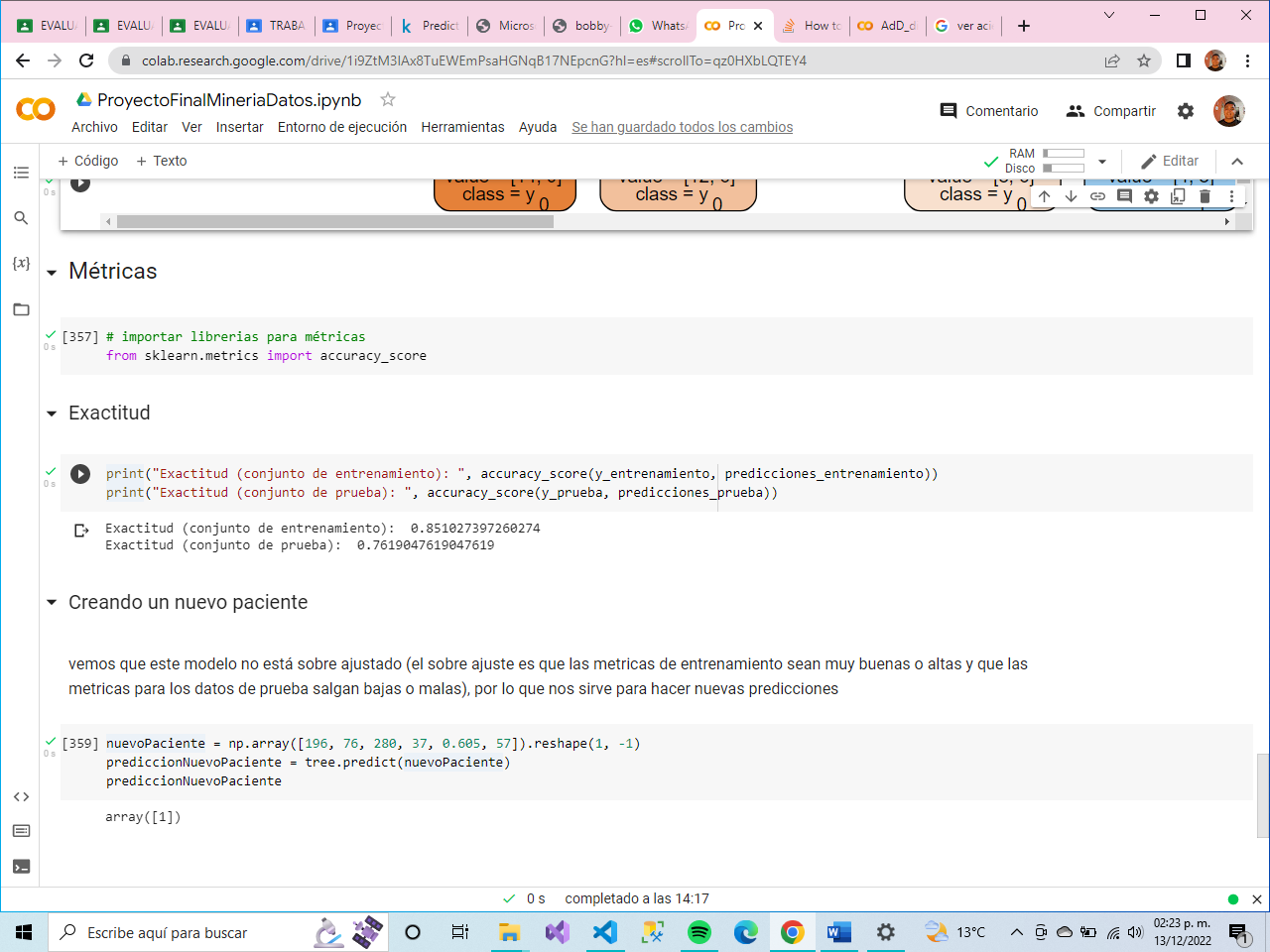
Y el árbol quedaria de la siguiente manera



Ahora con la función predict() podemos predecir los datos para nuestros datos de prueba, pero antes debemos ver una cosa, si nuestro modelo está sobre ajustado, esto quiere decir que las metricas para nuestros valores de entrenamiento son altas pero para los valores de prueba no o en otra palabas a la hora de entrenar nuestro modelo es bueno pero a la hora de predecir tiene poco rendimiento

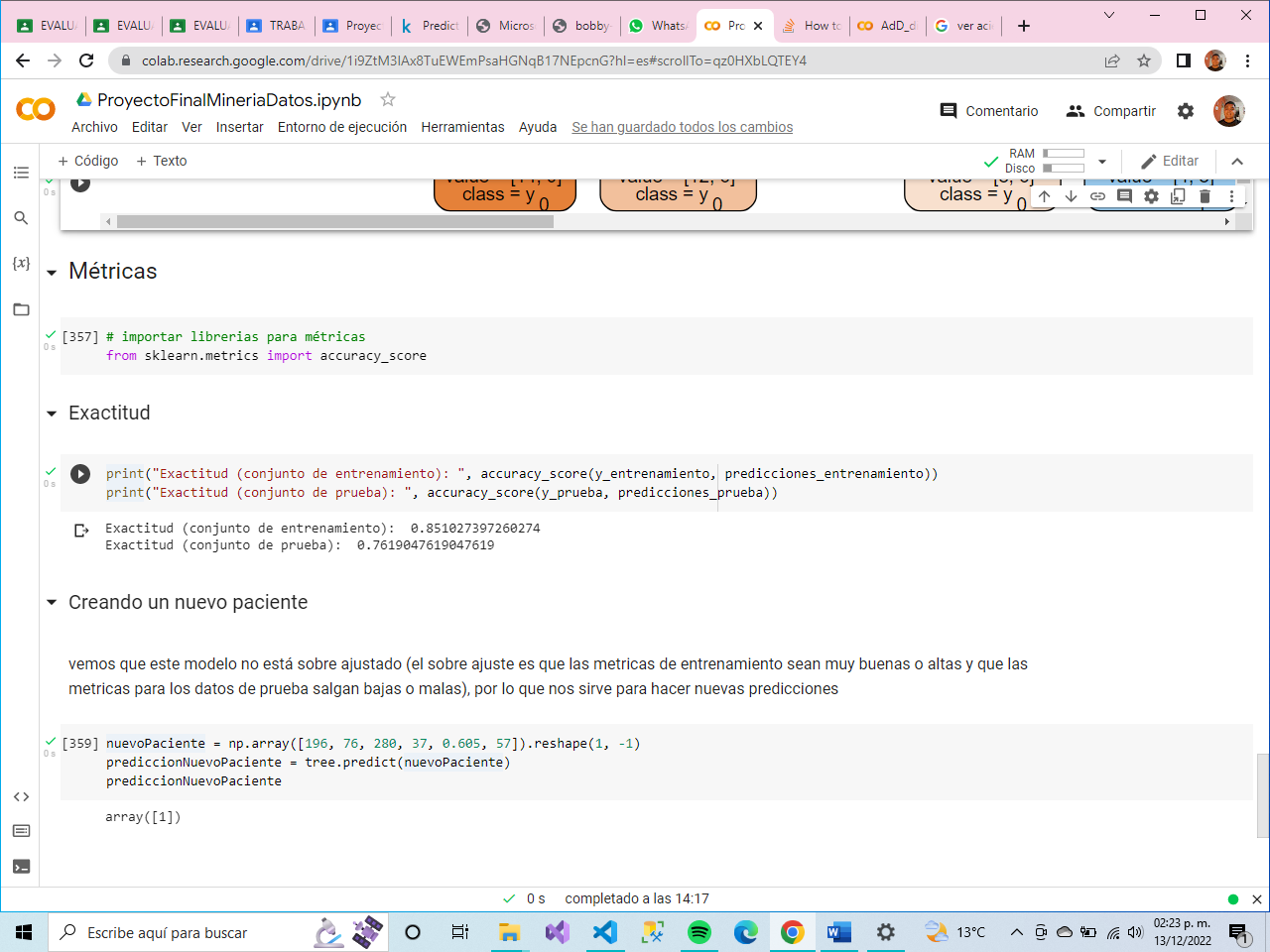


Para esto utilizando la función accurancy\_score() podemos determinar la exactitud para nuestros datos de entrenamiento y prueba



En la imagen pasada vemos que aunque nuestras exactitudes no son muy altas, las dos son un poco parecidas por lo que descartamos la opción de un sobre ajuste.

Sabiendo esto podemos utilizar nuestro modelo para predecir nuevos valores, intentemos esto, crearemos un nuevo paciente se lo pasaremos a nuestro modelo de árbol (tree) en la función predict() y veremos si tiene diabetes o no (siendo 0 no y 1 sí) con el siguiente código



**Resultados:**

Con este modelo de clasificación podemos ver que cuando una persona tiene nivel de glucosa en sangre de 196, una presión arterial de 76, un nivel de insulina en sangre de 280, un índice de masa corporal de 37, un porcentaje de diabetes de 0.605 y una edad de 57 nuestro paciente tiene diabetes.

**Conclusiones:**

Podemos concluir que el modelo de árbol de decisión nos sirve para clasificar, como en este caso que teníamos un dataset con pacientes positivos y negativos para diabetes, con la información que se tenia en el dataset se pudo entrenar al modelo para así poder predecir nuevos pacientes. En conclusión esto es muy importante, un dato que nos gustaría saber mucho es si es si el algoritmo clasifico a la persona como positivo y en realidad no tiene diabetes (falso positivo), estos pacientes son muy importantes, ya que se pueden tratar y evitar tener diabetes, ya que esta enfermedad es crónica, no tiene cura, solo se puede tratar y regular; por otra parte también nos interesan los pacientes que el algoritmo predijo que tenían diabetes y al hacerse una prueba de diabetes realmente dieron positivo (verdaderos positivos), estos pacientes nos interesan ya que se tienen que empezar a tratar ya que “la diabetes tiende a reducir el [colesterol](https://medlineplus.gov/spanish/cholesterollevelswhatyouneedtoknow.html) HDL (el ‘bueno’) y aumentar los triglicéridos (un tipo de grasa en la sangre) y el colesterol LDL (el ‘malo’). Estos cambios pueden aumentar el riesgo de enfermedad cardiaca y de derrame cerebral [3]”. Otros pacientes que nos interesan son los pacientes que el algoritmo clasifico como negativo a diabetes, pero en realidad si tienen diabetes (falsos negativos), siendo estos pacientes los mas en riesgo, ya que pueden confiarse y no atenderse. Por ultimo los que tienen que preocuparse mucho son los pacientes que el algoritmo clasifico como negativo a diabetes y realmente al hacerse pruebas no tienen diabetes (verdaderos negativos). La ventaja de usar algoritmos es que puede ser más fácil y enfocado hacia grandes cantidades de datos, como vimos anteriormente puede ser que el algoritmo no sea del todo preciso pero puede dar cierta idea talvez para hacer pruebas estadísticas de personas propensas a diabetes.

**Referencia:**

[1] Kaggle (Aman Chauhan), “Predict Diabetes”, (s.f.). [Online]. Disponible en: <https://www.kaggle.com/datasets/whenamancodes/predict-diabities>. [13 de diciembre de 2022].

[2] GraphPad, “Outlier calculator”, (s.f.). [Online]. Disponible en: <https://www.graphpad.com/quickcalcs/Grubbs1.cfm>. [12 de diciembre de 2022].

[3] Centros para el Control y la Prevención de Enfermedades, “Póngale el freno a las complicaciones de la diabetes”, (s.f.). [Online]. Disponible en: <https://www.cdc.gov/diabetes/spanish/resources/features/prevent-complications.html#:~:text=La%20diabetes%20tiende%20a%20reducir,cardiaca%20y%20de%20derrame%20cerebral>. [13 de diciembre de 2022].