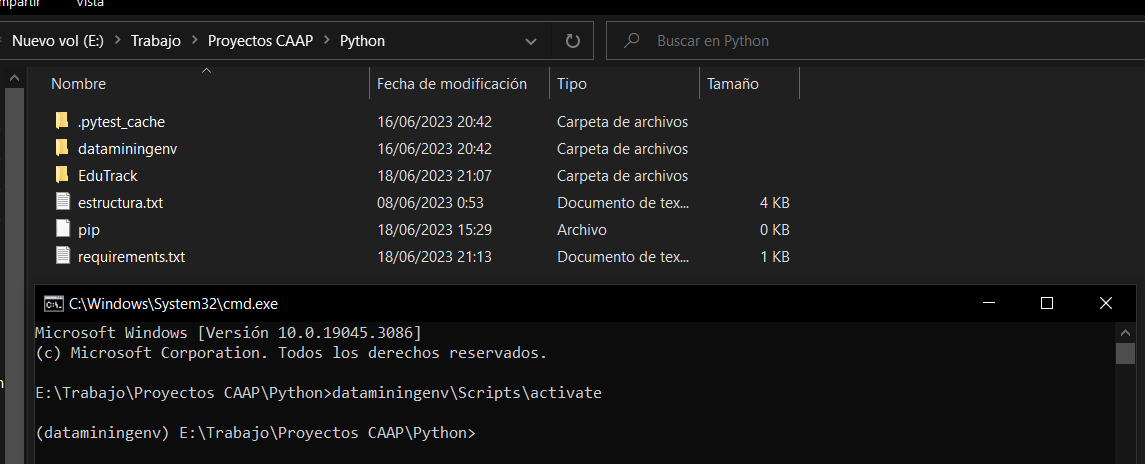
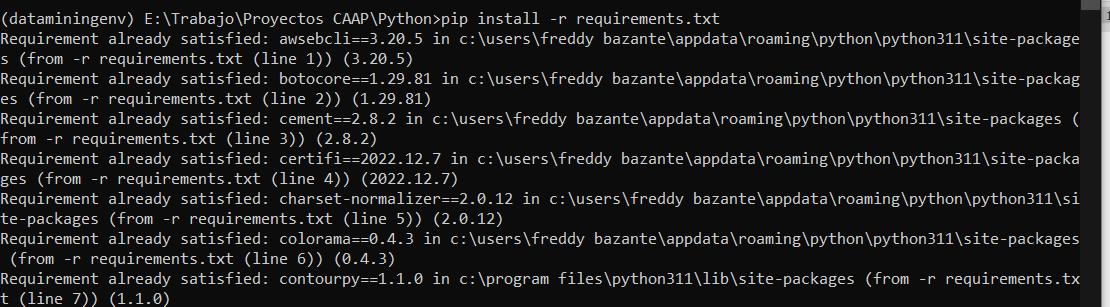
INDICACIONES GENERALES

1. Para poder ejecutar la aplicación, pruebas o comprobación de cualquier archivo se debe gestionar el entorno virtual con el comando dataminingenv\Scripts\activate en el directorio raíz antes del env



1. Una vez dentro del entorno ejecutamos el comando pip install -r requirements.txt para que se instalen todas las dependencias y bibliotecas que se usaron en el proyecto



1. Una vez instalado pasaremos a comprobar el funcionamiento de cada parte del proyecto para esto tenemos una estructura de directorios

Edutrack

| InterfazFinal.py

|

|

+---Historia\_de\_Usuario\_1

| +---codigo\_fuente

| | | main.py

| | | mineria\_datos.py

| | | modelos.py

| |

| +---datos

| | base\_de\_datos.sql

| |

| +---documentacion

| | documentacion\_adicional.md

| | estructura\_directorios.txt

| | Historias\_de\_usuario.docx

| | README.md

| |

| +---Interfaces

| | maininterface.py

| |

| \---pruebas

| | test\_mineria\_datos.py

| | test\_modelos.py

|

+---Historia\_de\_Usuario\_2

| +---codigo\_fuente

| | | main.py

| | | mineria\_datos.py

| | | modelos.py

| |

| +---datos

| | base\_de\_datos.sql

| |

| +---documentacion

| | documentacion\_adicional.md

| | estructura\_directorios.txt

| | Historias\_de\_usuario.docx

| | README.md

| |

| +---Interfaces

| | maininterface.py

| |

| \---pruebas

| | test\_mineria\_datos.py

| | test\_modelos.py

|

+---Historia\_de\_Usuario\_3

| +---codigo\_fuente

| | | main.py

| | | mineria\_datos.py

| | | modelos.py

| |

| +---datos

| | base\_de\_datos.sql

| |

| +---documentacion

| | documentacion\_adicional.md

| | estructura\_directorios.txt

| | Historias\_de\_usuario.docx

| | README.md

| |

| +---Interfaces

| | maininterface.py

| |

| \---pruebas

| | test\_mineria\_datos.py

| | test\_modelos.py

|

+---Historia\_de\_Usuario\_4

| +---codigo\_fuente

| | | main.py

| | | mineria\_datos.py

| | | modelos.py

| |

| +---datos

| | base\_de\_datos.sql

| |

| +---documentacion

| | documentacion\_adicional.md

| | estructura\_directorios.txt

| | Historias\_de\_usuario.docx

| | README.md

| |

| +---Interfaces

| | maininterface.py

| |

| \---pruebas

| | test\_mineria\_datos.py

| | test\_modelos.py

|

\---Historia\_de\_Usuario\_5

+---codigo\_fuente

| | main.py

| | mineria\_datos.py

| | modelos.py

|

+---datos

| base\_de\_datos.sql

|

+---documentacion

| documentacion\_adicional.md

| estructura\_directorios.txt

| Historias\_de\_usuario.docx

| README.md

|

+---Interfaces

| maininterface.py

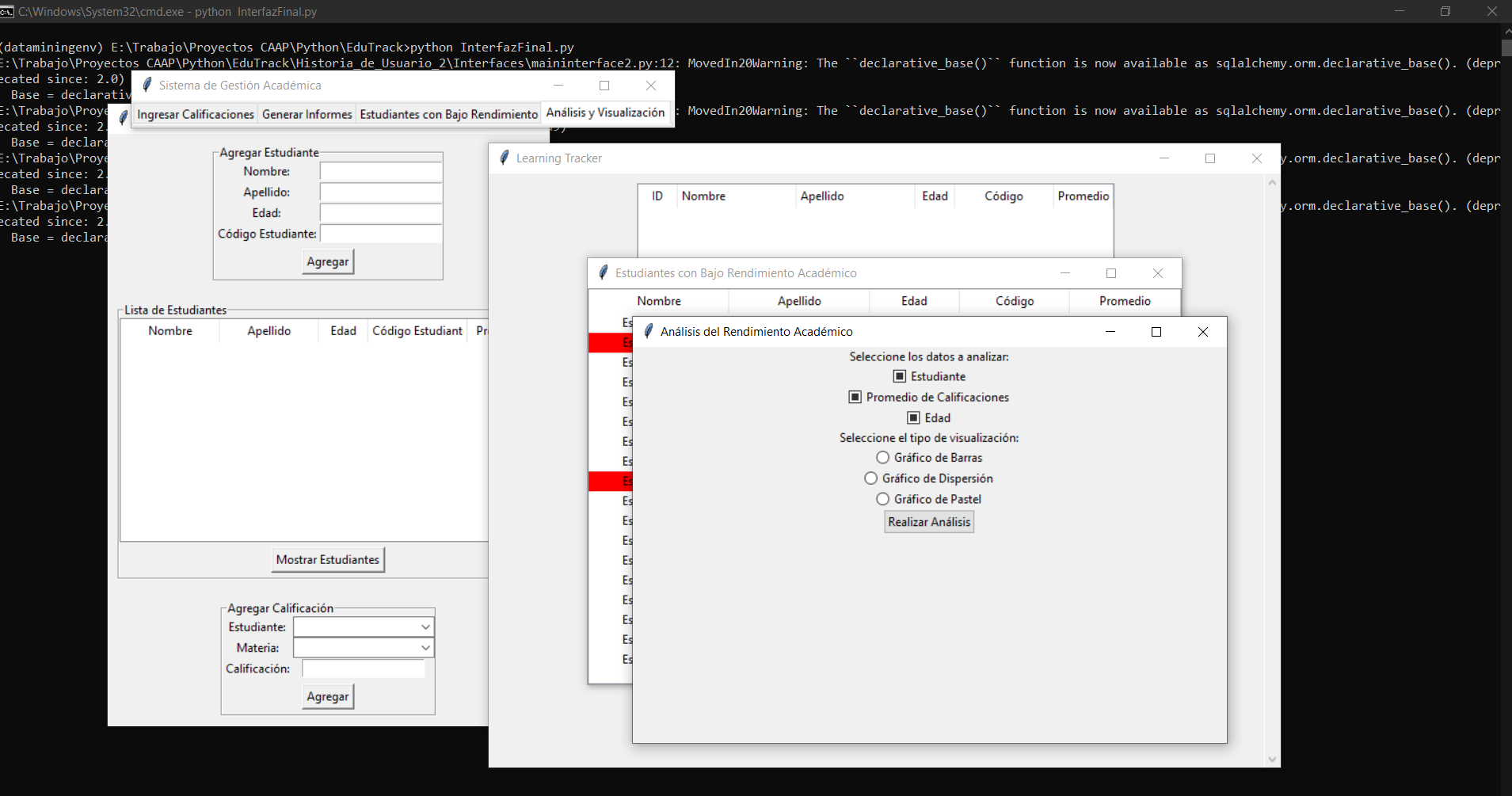
|

\---pruebas

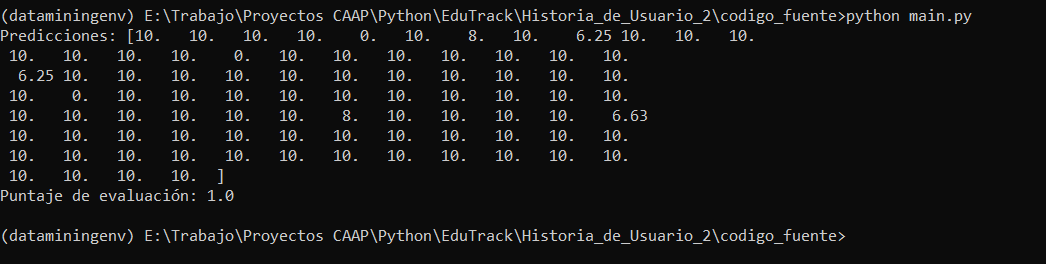
| test\_mineria\_datos.py

| test\_modelos.py

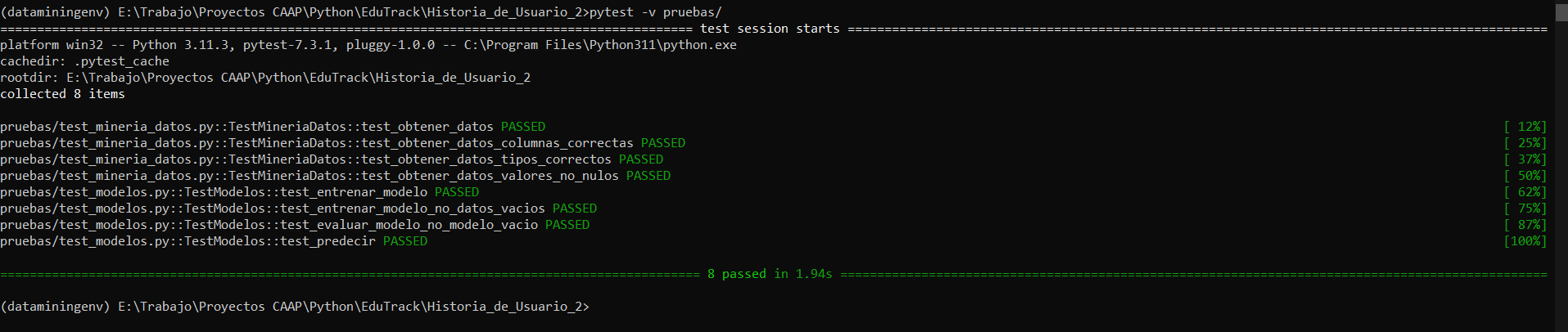
1. Comprobaremos la interfaz principal llamada InterfazFinal.py con el comando python InterfazFinal.py



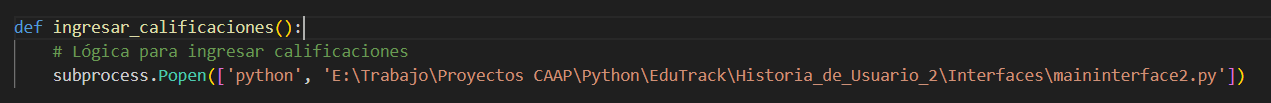
1. Después de interactuar con cada modulo pasaremos a comprobar el modelo de minería de datos basado en árboles de decisión ingresando a cada directorio de historia de usuario y dentro de estas en el directorio codigo\_fuente. Ejecutaremos el comando python main.py y con esto obtendremos las predicciones con respecto al rendimiento



1. Después de ver las predicciones del modelo pasaremos a realizar las pruebas unitarias para ello regresamos al directorio principal de cada historia de usuario y ejecutamos el comando pytest -v pruebas/. Esto nos indicará las pruebas realizadas y de estar correctas tendremos un estado PASSED

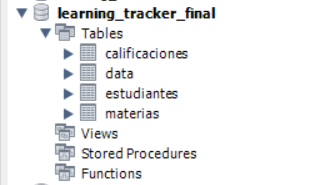


1. Hay que tomar algunas consideraciones en el código de la interfaz principal

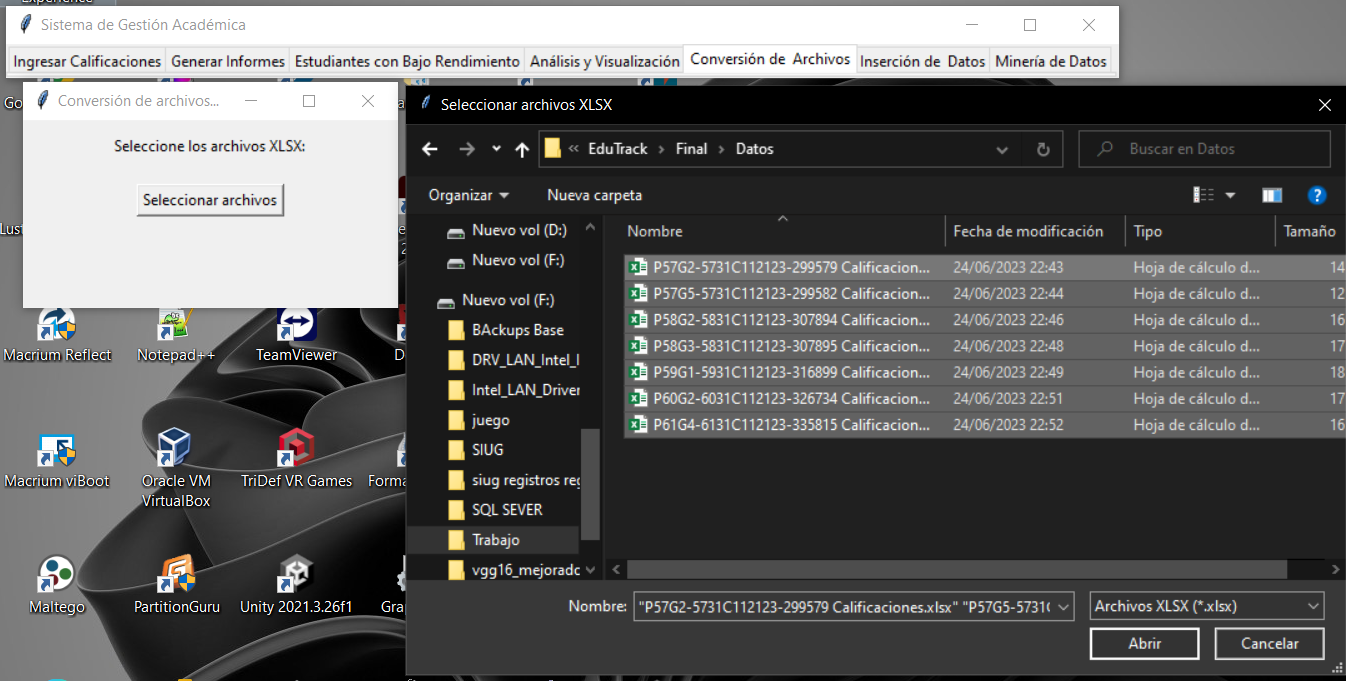


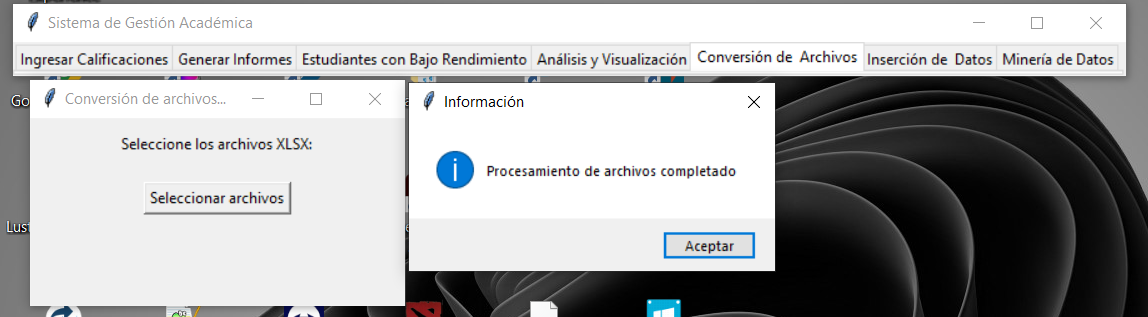
Aquí debemos modificar el directorio de acuerdo a la ubicación de la aplicación

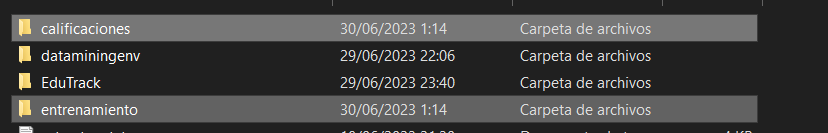
1. También debemos aclarar que se uso como motor de base de datos mysql con el gestor Workbench. Para cargar la base de datos usaremos el script que se encuentra en cada historia de usuario. Para la historia de usuario 1 es un script diferente mientras que para las otras historias de usuario es la misma
2. En la parte final del proyecto se solicitó que se prueben algunos datos almacenados en archivos xlsx de Excel para lo cual se procedió a crear una base de datos llamada learning\_tracker\_final



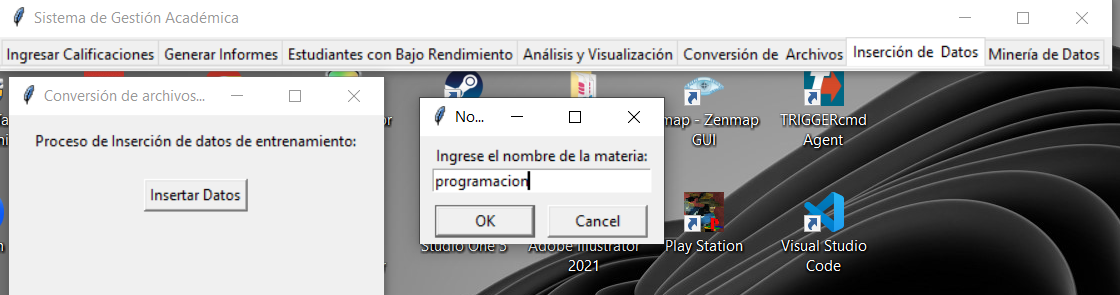
1. Después de esto se procedió a analizar los archivos xlsx, para ello se abre cada archivo y se toma los datos esenciales a analizar y se los normaliza para poder interpretarlos con el código. Las columnas que se tomaron en cuenta son Total Parcial Final Aprovechamiento (Real), Cuestionario: Examen Parcial Final (Real), Cuestionario\_Examen\_Final, Total PF Examen (Real), Total Parcial Final (Real), Total del curso (Real.  
   
2. Una vez normalizados los archivos se procede a convertirlos y almacenarlos en una carpeta para entrenamiento y otra para evaluar datos que se llama calificaciones

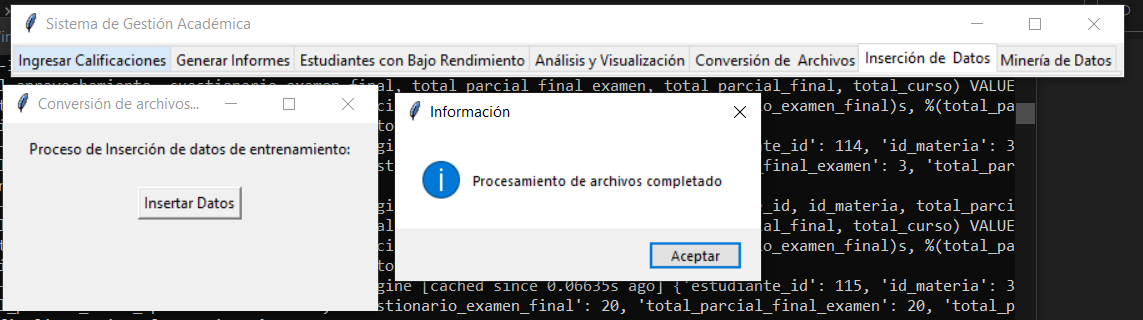




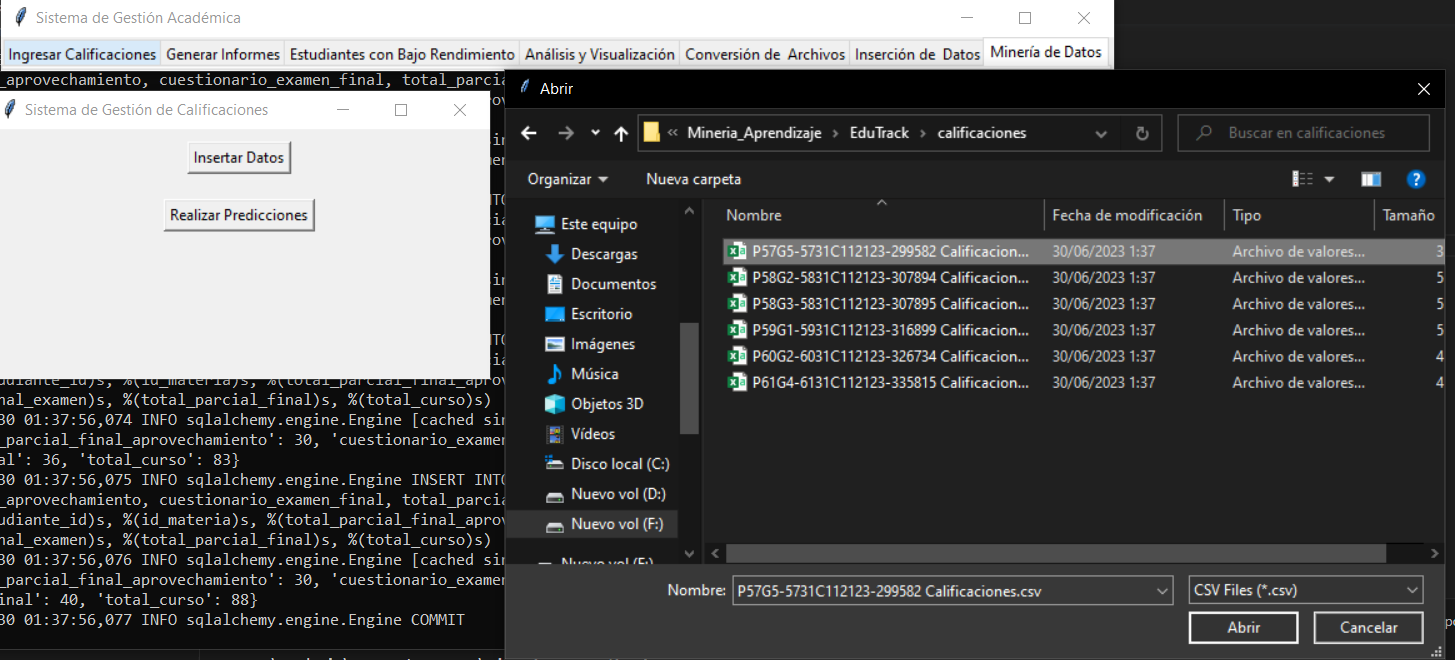


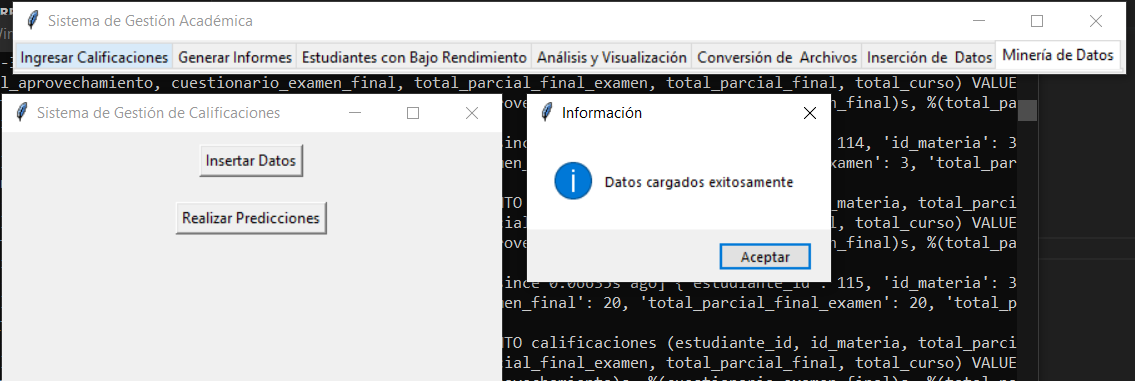
1. Luego se procede a insertar los datos en la base en la pestaña Inserción de archivos. Aquí se pedirá la materia

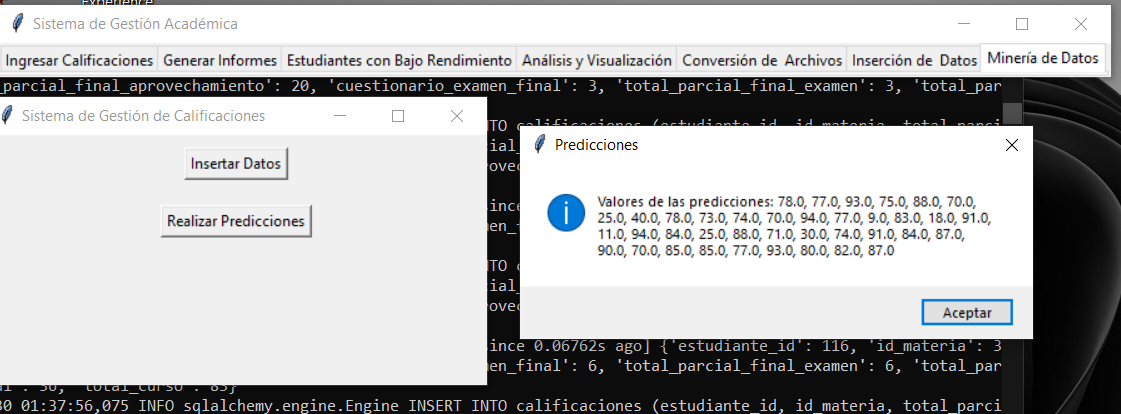


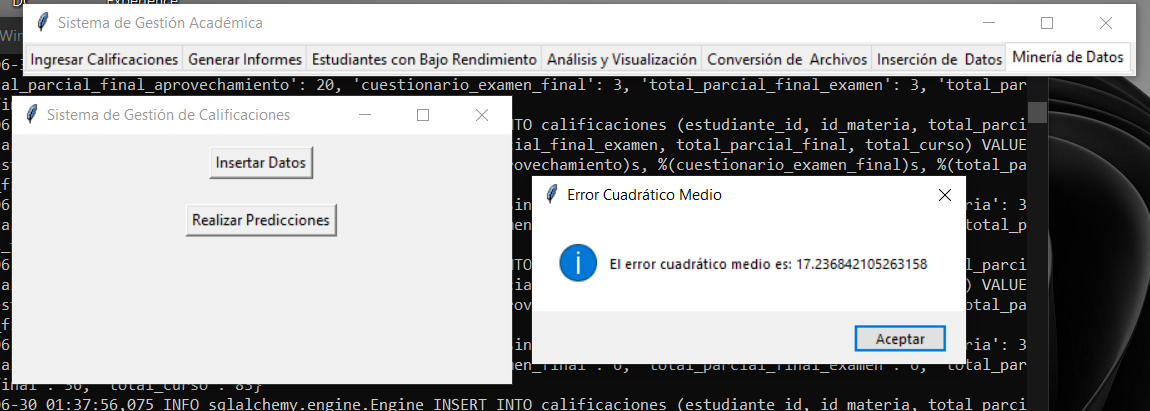


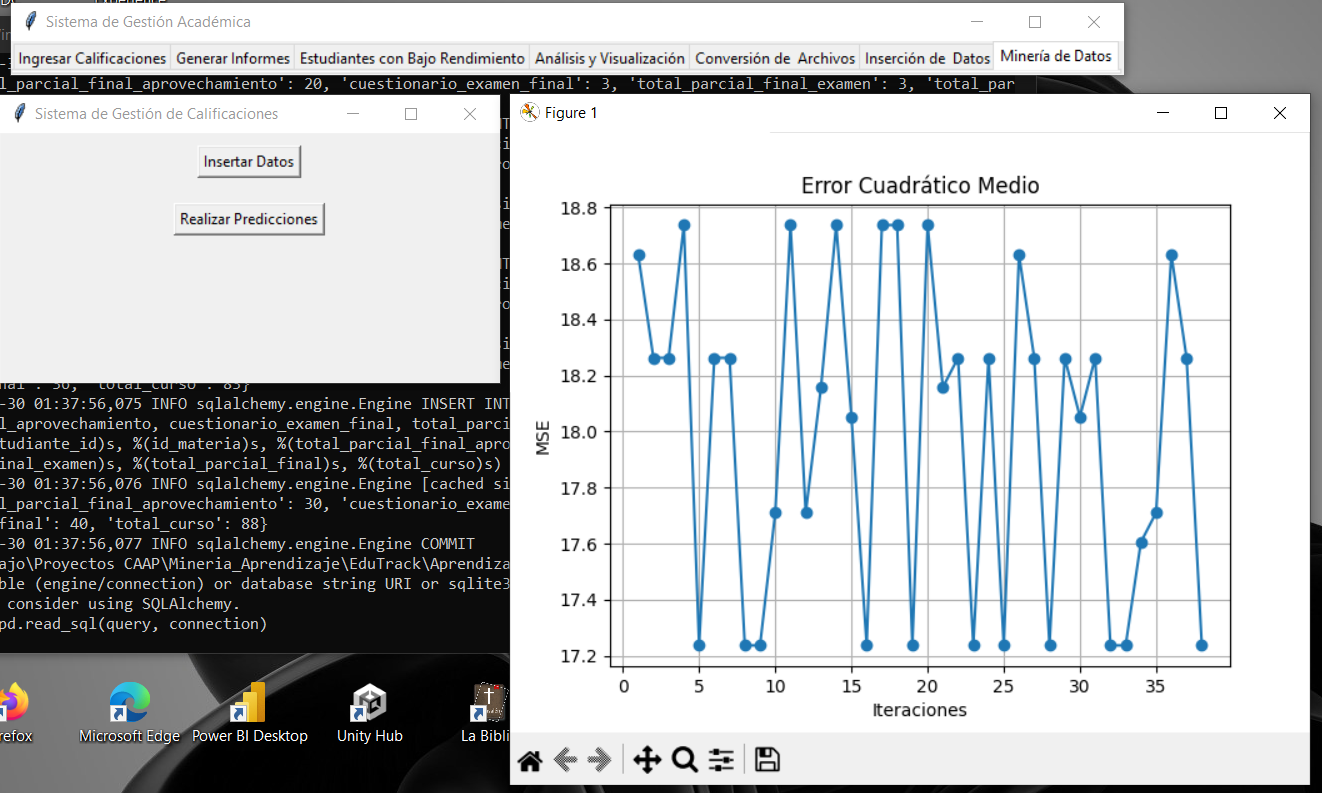
1. YA insertados los datos de entrenamiento se procede a ejecutar el modelo de minería en el que se evaluan datos de prueba y obtendremos datos como predicciones y el rendimiento del algoritmo mediante el erro cuadrático medio



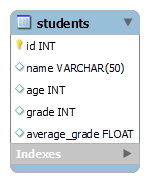




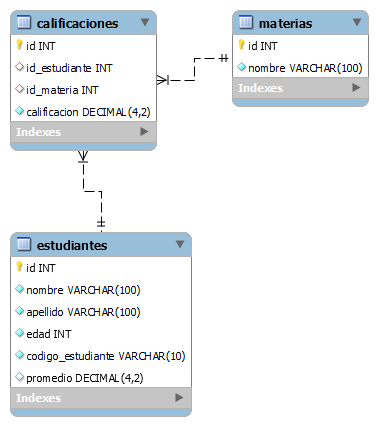




Los modelos entidad relación de la base de datos usadas son los siguientes  
  
Base de datos learning\_tracker\_



Base de datos learning\_tracker\_h2



Base de datos learning\_tracker\_final

