Predicción del diagnóstico de diabetes a partir de perfiles clínicos de pacientes utilizando aprendizaje automático

Anexo A1: manual de usuario

Leydi Esperanza Pérez Leal

José Alejandro Buitrago Cárdenas

Universidad Antonio Nariño
Facultad de Ingeniería de Sistemas
Bogotá D.C., Colombia
2021

Contenido

Lista de Figuras

Lista de Tablas

Espacio reservado para las Tablas.

1 Guía de usuario

Debido a que el proyecto es de investigación, el manual se compone de dos secciones, que permiten un fácil entendimiento y operación del aplicativo, constan de: Descripción de la aplicación y funcionamiento de la misma.

El presente documente, introduce a los conceptos de programación orientada a objetos, para la creación de un script mediante el uso del intérprete PyCharm Community Edition y Anaconda Navigator para CMD.exe Prompt, para la ejecución.

Este manual contiene ejemplos que ilustran la ejecución del script, cumpliendo con los requerimientos mínimos y exigentes posibles sobre las funciones básicas. Es de suma importancia que usted lea en forma completa las instrucciones, aun cuando no desee llevar el sistema en forma integrada, aunque sólo desee utilizar alguna de sus dos secciones.

1.1. Descripción de la aplicación

Los scripts tienen como finalidad, construir una base de informática completa y organizada, en donde se pueda observar la información del rendimiento de los clasificadores: random forest, neural network y support machine vector.

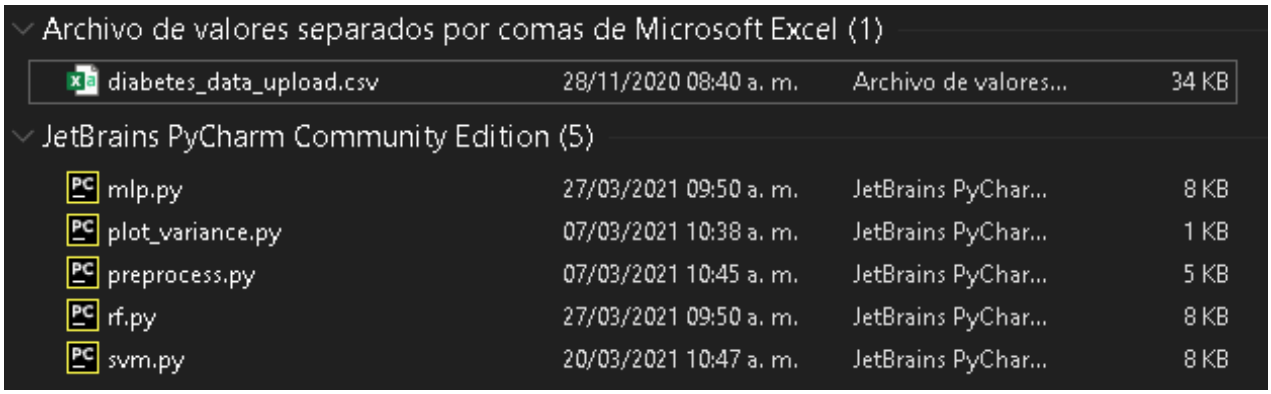
El programa contiene buenas prácticas de documentación, estructuras condicionales, de repetición y estructurado con funciones definidas por el programador, así como la generación de archivo plano separados por comas (en inglés: Comma-separated values, CSV) (archivos de texto que almacena los datos en forma de columnas, separadas por coma y las filas se distinguen por saltos de línea).

1.2. Funcionamiento

Inicio
Para la ejecución del script (código fuente) la computadora debe contar con:

- La instalación de Python 3.7, bibliotecas de Scikit-learn, Docx, entre otras. En la sesión (3.1) se explica cómo realizar las instalaciones

Figura 1-1: Carpeta donde se encuentran los archivos a utilizar.



Fuente: Elaboración propia.

- La clase: preprocess.py, mlp.py, rf.py, svm.py, plot_variance.py y su respectivo dataset en formato CSV diabetes_data_upload.csv, deben guardarse en archivos independientes con el fin de llevar una buena estructura, como se observa en la figura 1-1.

Teniendo en cuenta los requerimientos mencionados anteriormente, se accede a la aplicación anaconda, como se observa en la imagen 1-2 en la opción launch, ingresa a la consola de comandos cmd. Se accede al directorio donde se encuentran alojados todos los archivos .py, para nuestro caso la ruta es: C:/Users/Usuario/Desktop/Codigopy, como muestra la figura 1-3. Una vez ubicado se procede a ingresar la variable de entorno Python seguido del nombre de la clase "preprocess.py" y el nombre del dataset "diabetes_data_upload.csv", por último un archivo par a alojar el histórico de ejecución como buena práctica (ver Figura 1-4). Luego se presiona "Enter". Este proceso se realizará con las siguientes líneas de ejecución, en donde estaremos haciendo uso de los algoritmos propuestos en este trabajo de investigación

- La instalación de Python 3.7, bibliotecas de Scikit-learn, Docx, entre otras. En la sesión (3.1) se explica cómo realizar las instalaciones

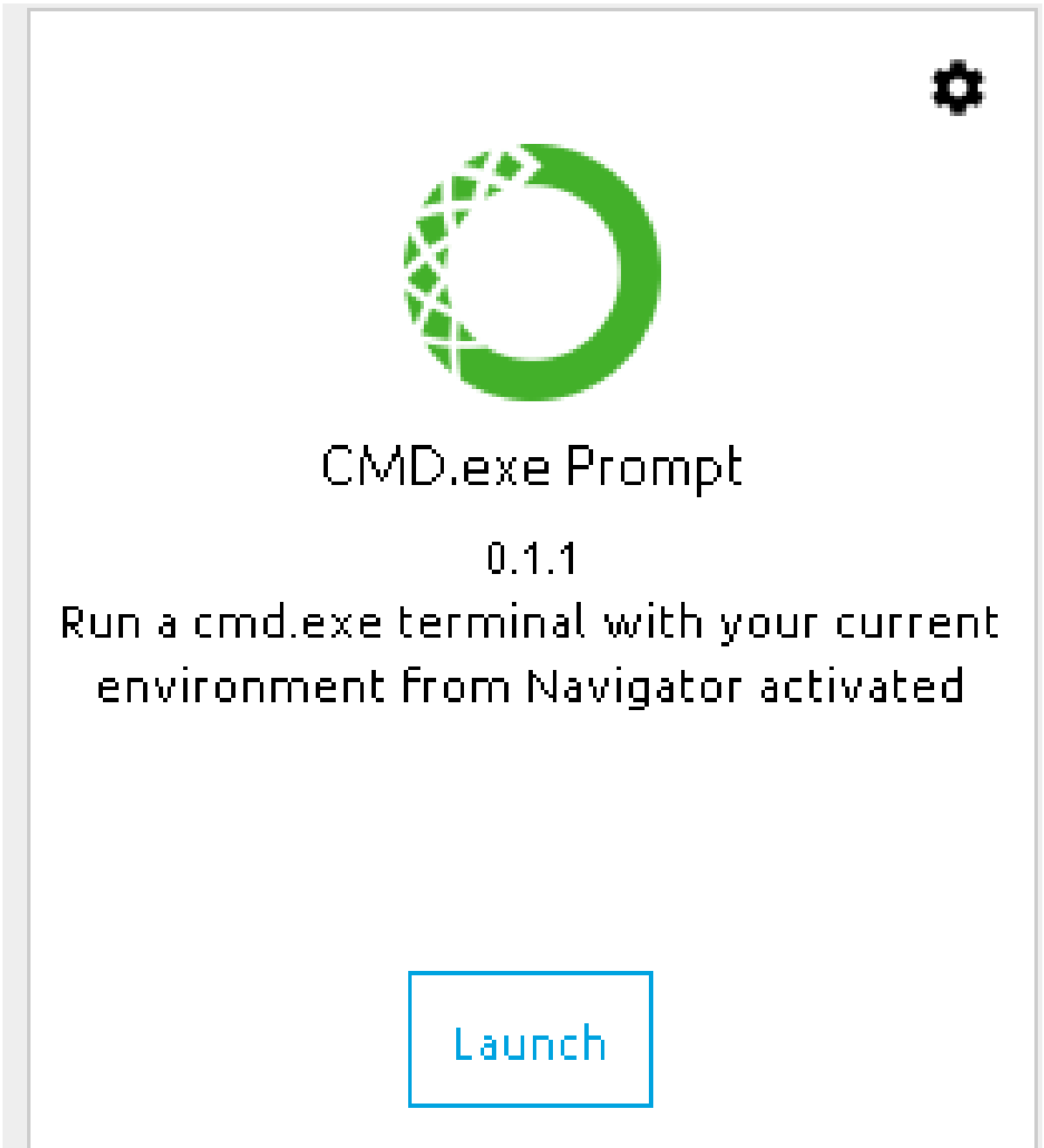
- La clase: preprocess.py, mlp.py, rf.py, svm.py, plot_variance.py y su respectivo dataset en formato CSV diabetes_data_upload.csv, deben guardarse en archivos independientes con el fin de llevar una buena estructura, como se observa en la figura 1-1.

- Python svm.py preprocessed_data.csv svm_performance_metrics.csv model_selection_results.csv 11 svm.log. ver imagen 1-5

- python mlp.py preprocessed_data.csv mlp_performance_metrics.csv model_selection_results.csv 11 svm.log

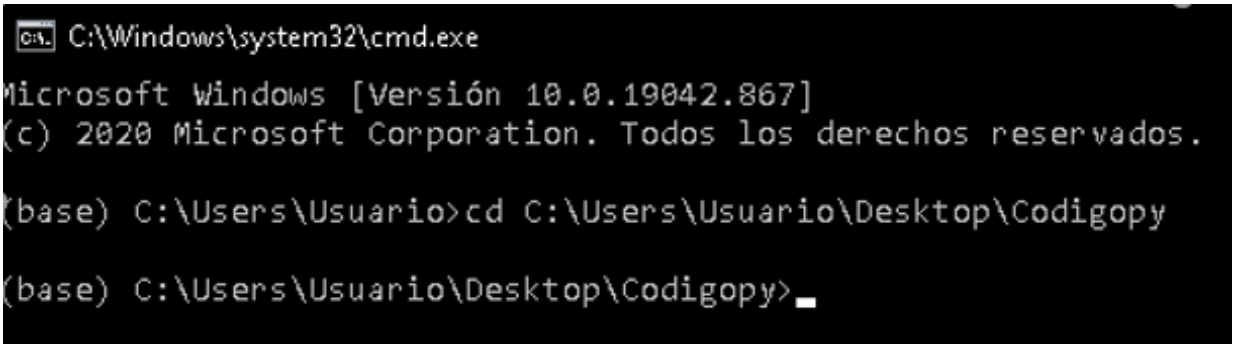
- python rf.py preprocessed_data.csv rf_performance_metrics.csv model_selection_results.csv 11 rf.log

Figura 1-2: Imagen de la interfaz de acceso a anaconda.



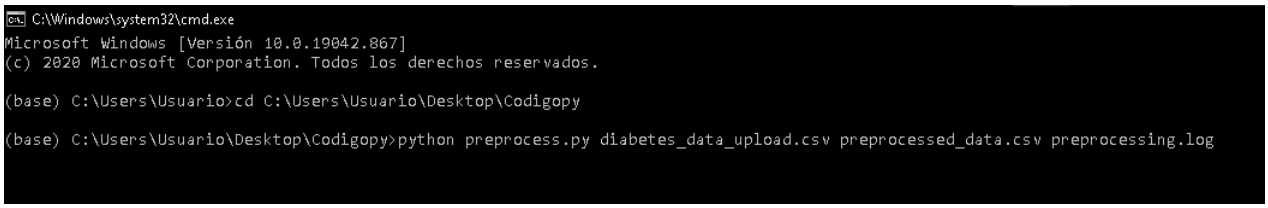
Fuente: Tomado de página oficial Anaconda.

Figura 1-3: ruta de acceso a los archivos .py



Fuente: Elaboración propia.

Figura 1-4: Comando para ingresar a la variable de entorno, con histórico.

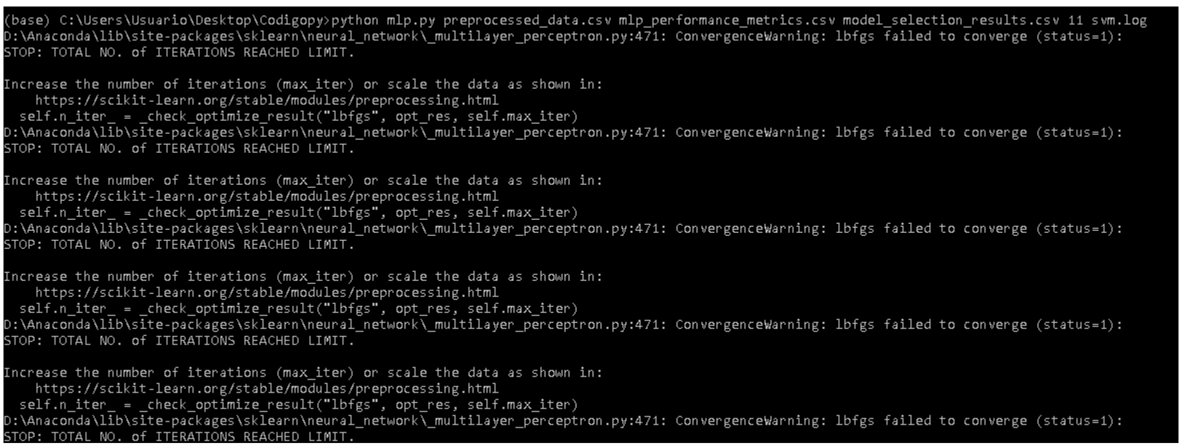


Fuente: Elaboración propia.

Figura 1-5: Implementación de modelo SVM.

Fuente: Elaboración propia.

Figura 1-6: Implementación de modelo MLP.



```
(base) C:\Users\Usuario\Desktop\Codigopy>python mlp.py preprocessed_data.csv mlp_performance_metrics.csv model_selection_results.csv 11 svm.log
D:\Anaconda\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:471: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
  self.n_iter_ = _check_optimize_result("lbfgs", opt_res, self.max_iter)
D:\Anaconda\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:471: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
  self.n_iter_ = _check_optimize_result("lbfgs", opt_res, self.max_iter)
D:\Anaconda\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:471: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
  self.n_iter_ = _check_optimize_result("lbfgs", opt_res, self.max_iter)
D:\Anaconda\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:471: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
  self.n_iter_ = _check_optimize_result("lbfgs", opt_res, self.max_iter)
D:\Anaconda\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:471: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

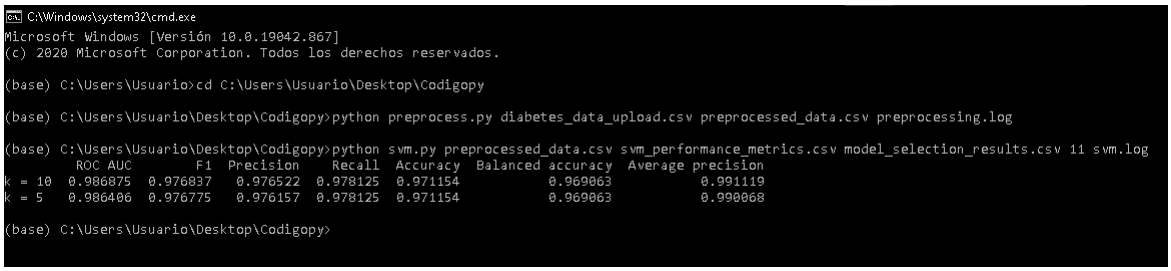
Fuente: Elaboración propia.

Figura 1-7: Implementación de modelo rf.

```
(base) C:\Users\Usuario\Desktop\Codigopy>python rf.py preprocessed_data.csv rf_performance_metrics.csv model_selection_results.csv 11 rf.log
```

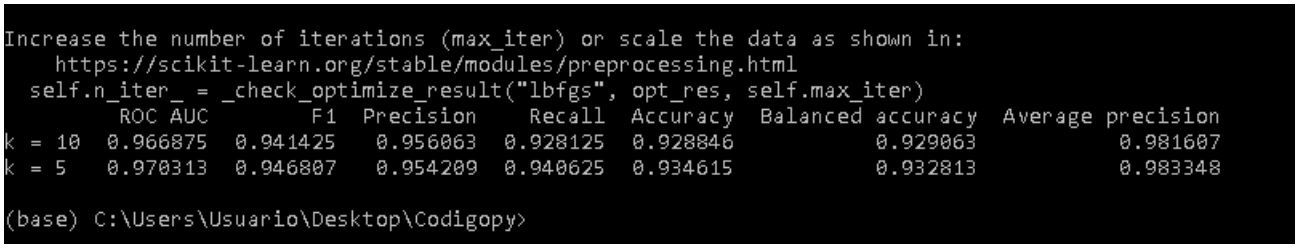
Fuente: Elaboración propia.

Figura 1-8: Comando para solicitar métricas de modelo SVM.



Fuente: Elaboración propia.

Figura 1-9: Comando para solicitar métricas de modelo MLP.



Fuente: Elaboración propia.

1.3. Muestra de resultados

Luego de presionar "Enter", para la ejecución de la segunda línea de código el sistema realiza los cálculos necesarios y procede a mostrar la información por medio de consola (Figura 1-8, 1-9, 1-10) ó en archivos de formato CSV (Figura 1-11). Este proceso se deberá repetir para la segunda y tercera línea de ejecución. La imagen de la figura 1-10 muestra los resultados para las métricas del modelo SVM.

La imagen de la figura 1-10 muestra los resultados para las métricas del modelo MLP.

La imagen de la figura 1-10 muestra los resultados para las métricas del modelo RF.

Luego de obtener los resultados el programa automáticamente descarga un archivo con extensión .CSV en el cual registra los resultados óptimos para las métricas.

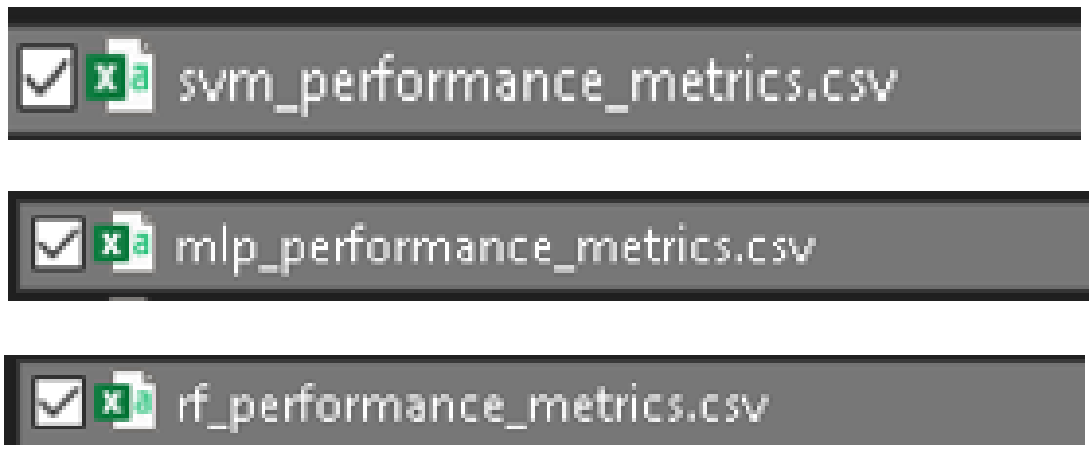
Figura 1-10: Comando para solicitar métricas de modelo RF.

```
(base) C:\Users\Usuario\Desktop\Codigopy>python rf.py preprocessed_data.csv rf_performance_metrics.csv model_selecti
         ROC AUC        F1  Precision    Recall  Accuracy  Balanced accuracy  Average precision
k = 10  0.989062  0.972513   0.964810  0.984375  0.965385           0.965625           0.993536
k = 5   0.984453  0.975523   0.967402  0.987500  0.963462           0.961250           0.986265

(base) C:\Users\Usuario\Desktop\Codigopy>_
```

Fuente: Elaboración propia.

Figura 1-11: Imagen de los archivos extensión .csv de resultados.



Fuente: Elaboración propia.

2 Requisitos de hardware y de software

2.1. Requisitos mínimos

Esta sesión se indican las recomendaciones de hardware y software para la ejecución del programa creado en Python 3.9.2. En la siguiente lista se dan a conocer los requerimientos de hardware y software mínimos necesarios para ejecutar la versión de 32 bits del programa realizado.

- Sistema operativo: Microsoft Windows Xp, Vista, Seven, 8, 10.
- Procesador: Intel 486 o superior.
- Memoria RAM: 2 GB
- Disco duro: 1.16 MB.

2.2. Hardware y Software recomendados

En el item anterior se dan a conocer los requerimientos mínimos, sin embargo, estos no siempre son la mejor opción para obtener un rendimiento ideal en la ejecución del programa. A continuación, se da a conocer la lista recomendada para optimizar la ejecución del programa del trabajo de grado en el intérprete: Python 3.9.2.

- Procesador Pentium, a velocidades de 200MHz (o superior).
- Memoria RAM: 4 GB.
- Disco duro: 4 MB.

3 Instalación y ejecución

Preliminares

El presente capítulo está destinado a proveer la información necesaria sobre la instalación del intérprete Python 3.9. Los ejercicios que se presentan a continuación permiten al usuario familiarizarse con la funcionalidad mencionada anteriormente y la operatoria en que trabaja la aplicación.

3.1. Instrucciones de instalación

Como preparar el desarrollo de entorno para la ejecución de la aplicación.

1. ¿Instalar Python para Windows?.
   a) Escoger una instalación de Python.
   Paso 1. Identificar la versión de Windows que se ejecuta en su computadora. Se tiene que saber si está ejecutando Windows XP, Vista, Windows 7 o Windows 10.
   Siga los siguientes pasos a continuación para ver la información del sistema en Windows 10:
      * Hacer clic en el Menú de inicio y elija Configuración.
      * Hacer clic en Sistema.
      * Seleccione Acerca de

   Paso 2. Decida si necesitará el código fuente de Python. Instalar la fuente es opcional ya que Python es un software de código abierto, lo que significa que el código está disponible para los programadores.

   Paso 3. Dirigirse a la página web Python Welcome to Python.org todas las distribuciones oficiales del programa se pueden encontrar en dicho enlace, incluyendo el instalador para Windows.

   Paso 4. Hacer clic en el enlace "Descargar". Una lista de archivos aparecerá, cada archivo es una distribución de Python para una plataforma específica, ver Figura 3-1.
   Paso 5. Hacer clic en el enlace de descarga. Puede descargar el archivo y luego ejecutarlo,

Figura 3-1: Imagen de descarga de Python.

Fuente: Tomado página oficial de python.

o bien hacerlo automáticamente. Es recomendable descargarlo para tener una copia de uso futuro (ver figura 3-2).

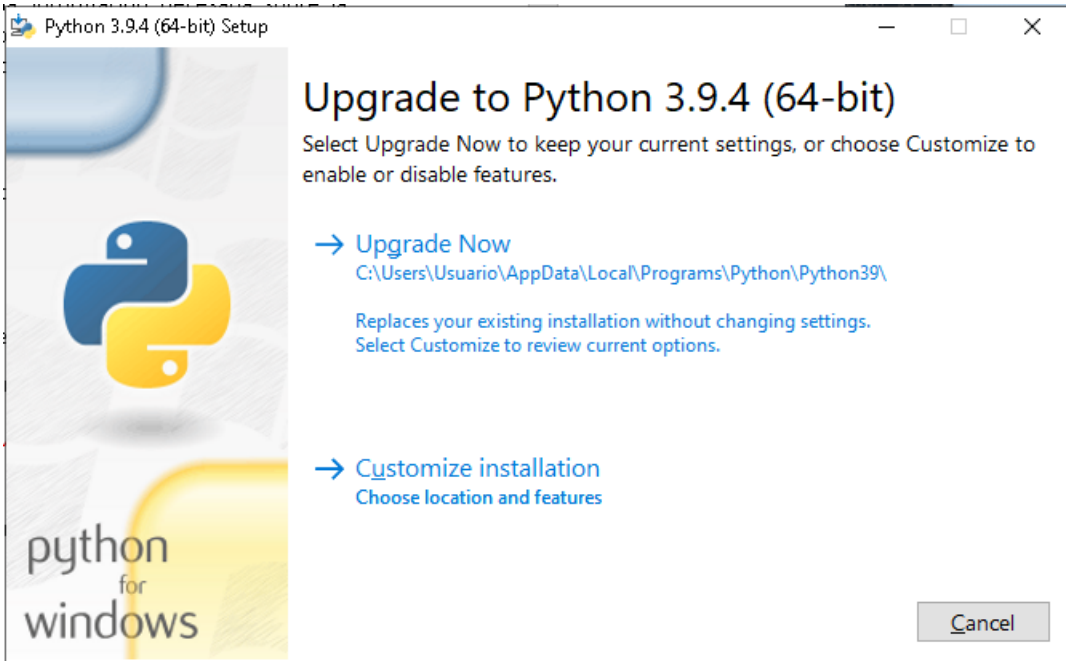
b) Ejecutar el instalador.

Paso 1. Ejecutar el instalador, si no lo ejecutó automáticamente desde el diálogo de descarga. Busque el archivo usando Windows explorer y ejecútelo. Un programa de instalación se abrirá. Hacer clic en "Instalar para todos los usuarios" y luego en "Instalar ahora". Si ya contempla una versión de Python, habilitara la opción Upgrade Now (ver imagen 3-3).

Figura 3-2: Imagen de ejecutable de python.

Fuente: Elaboración propia.

Figura 3-3: Imagen de ejecutable de python.



Fuente: Elaboración propia.

Paso 2. Escoja un directorio de instalación para Python. El valor por defecto, (/AppData/Local/Programs/Python/Python39), se recomienda no cambiar esta ruta, ya que pueden darse situaciones en la que será útil para que escriba la ruta completa al intérprete de Python desde la línea de comandos. Un nombre de directorio corto en la unidad (C: /) es más fácil de escribir.

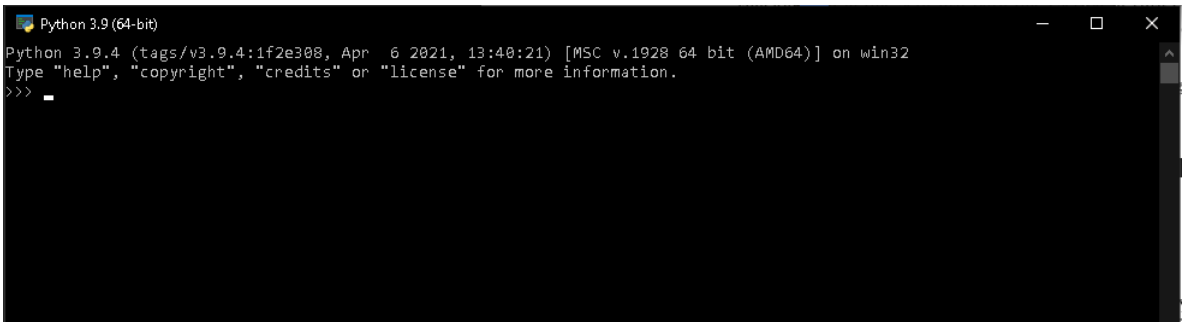
Paso 3. Elija las funciones que desea instalar y hacer clic en "Siguiente" para iniciar la instalación. Espere unos minutos para que el proceso se complete. Una vez que se haya instalado Python, hacer clic en "Finalizar" para cerrar el instalador.

Paso 4. Hacer clic en el Menú de inicio, luego escribir "Python3,9" para probarlo desde el menú Inicio de Windows. Una ventana en negro y blanco se abrirá con un comando interactivo de Python del sistema. Una vez que haya confirmado que el programa está instalado correctamente (ver Figura 3-4 y 3-5), cierre la ventana.

Figura 3-4: Confirmación de instalación correcta de python por aplicaciones.

Fuente: Elaboración propia.

Figura 3-5: Confirmación de instalación correcta de python por comando.



Fuente: Elaboración propia.

3.1.1. Ver reporte

El reporte es una forma de proporcionar la estructura de los resultados generado por la aplicación con el fin de guiar al portador para su respectivo el análisis. A continuación, se muestra los detalles del reporte.
mlp_performance_metrics: figura 3-6
svm_performance_metrics: figura 3-7
rf_performance_metrics: figura 3-8

Figura 3-6: Reporte de métrica para modelo MLP.

	ROC AUC	F1	Precision	Recall	Accuracy	Balanced acc	Average precision
k = 10	0.966875	0.9414253	0.95606261	0.928125	0.92884615	0.9290625	0.9816071
k = 5	0.9703125	0.94680691	0.95420863	0.940625	0.93461538	0.9328125	0.98334816

Fuente: Elaboración propia.

Figura 3-7: Reporte de métrica para modelo SVM.

A	B	C	D	E	F	G	H	I
	ROC AUC	F1	Precision	Recall	Accuracy	Balanced acc	Average precision	
k = 10	0.986875	0.97683719	0.97652167	0.978125	0.97115385	0.9690625	0.9911185	
k = 5	0.98640625	0.97677527	0.97615708	0.978125	0.97115385	0.9690625	0.99006754	

Fuente: Elaboración propia.

Figura 3-8: Reporte de métrica para modelo RF.

A	B	C	D	E	F	G	H	I
	ROC AUC	F1	Precision	Recall	Accuracy	Balanced acc	Average precision	
k = 10	0.9890625	0.97251294	0.96480991	0.984375	0.96538462	0.965625	0.99353634	
k = 5	0.98445312	0.97552302	0.96740245	0.9875	0.96346154	0.96125	0.98626519	

Fuente: Elaboración propia.

4 Presentación y manejo de módulos del sistema

Desarrollo de aplicación Para la aplicación de las técnicas de ML propuestas en este proyecto se utilizó la librería scikit-learn versión 0.24.1 y las demás mencionadas en las sesiones anteriores. El ambiente de desarrollo o entorno de desarrollo integrado (en inglés: Integrated Development Environment, IDE) que se utilizó fue PyCharm y Anaconda.

La aplicación cuenta con un archivo CSV: diabetes_data_upload que corresponde al dataset que se usa en el proyecto de grado. Este archivo se encuentra en el repositorio de Git Hub público.