SEDE: ORELLANA

FACULTAD: INFORMÁTICA Y ELECTRÓNICA

**CARRERA:** INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

**MINERIA DE DATOS**

**NIVEL:** SEXTO

**PARALELO:** A

## Informe

1. **DATOS GENERALES:**

**NOMBRE: estudiantes CÓDIGOS: de estudiantes**

Byron Bonifaz 2831

Aracely Miranda 2812

**FECHA DE REALIZACIÓN: FECHA DE ENTREGA:**

24/06/2023 24/06/2023

1. **TEMA:**

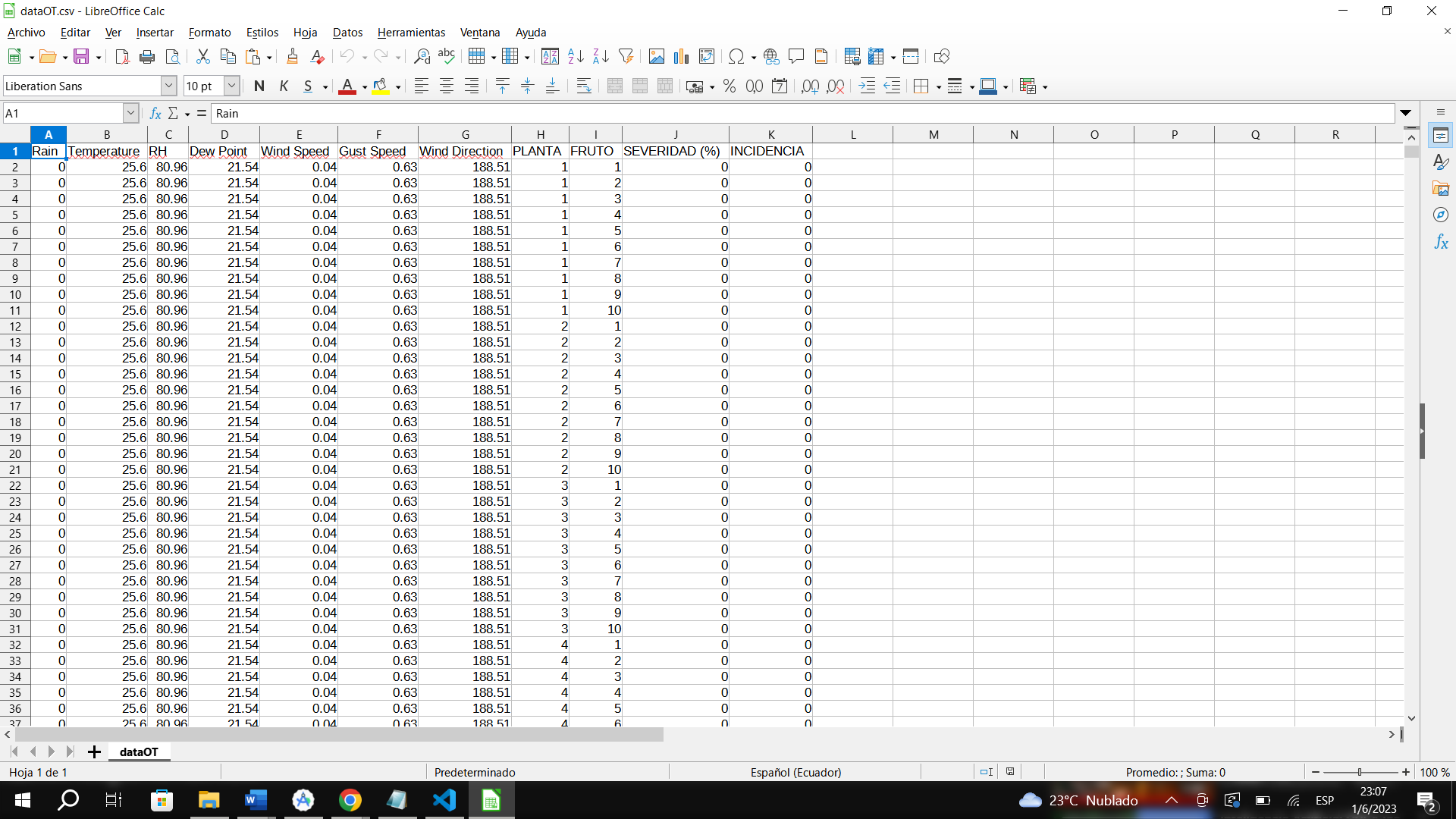
Algoritmos de aprendizaje supervisado

1. **DESARROLLO:**

Para realizar un aporte conciso y practico en el articulo que se esta redactando sobre el desarrollo de un modelo predictivo en producción mediante aplicaciones web progresivas para pronosticar la moniliasis en el cacao en la joya de los sacha, se pretende realizar primero este documento donde se detalla con claridad y se realiza un análisis de todos los resultados que se obtuvieron con todos los modelos realizados y una vez terminado dicho análisis se seleccionara la información con mayor relevancia para colocar en el artículo.

Para obtener un modelo predictivo que pronostique la monoliasis en los frutos del cacao con el mayor porcentaje de fiabilidad, se realiza la siguiente comparación entre los diferentes tipos de algoritmos de Machine Learning donde se les clasifico en 4 grupos los cuales se detallan más adelante y básicamente estos algoritmos ayudan a reducir la complejidad de los posibles problemas que podria llegar a tener este modelo predictivo.

Por otra parte, se realizó un dataset general donde se llevó a cabo la unión de los datos obtenidos de los sensores y de los reportes de datos de las plantas del cacao obtenidos manualmente, después de esto se realizó un tratamiento a dichos datos dando lugar así a tener el siguiente archivo “[dataset](https://liveespochedu-my.sharepoint.com/personal/aracely_miranda_espoch_edu_ec/Documents/Sensores/dataOT.csv)” con extensión .csv, como se observa a continuación:



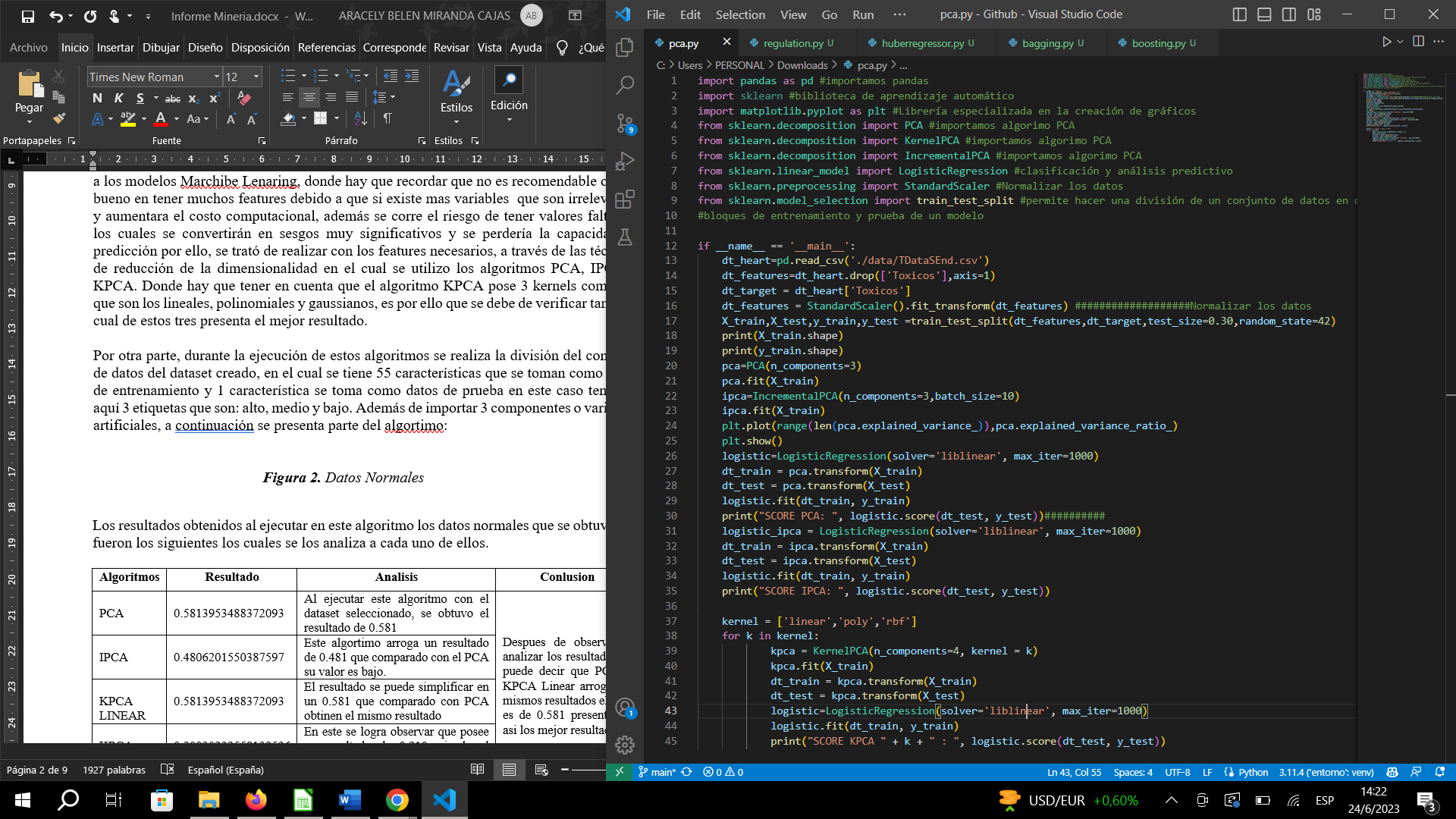
**Figura 1.** Dataset

Entonces los métodos que se realizo para que este modelo evite caer en problemas de complejidad o de simplesa y de los cuales se dividen en 4 grupos son:

1. **Técnicas de la reducción de la dimensionalidad**

Primero, de todo dataset se necesita saber o identificar cual de todos los feaures afectan a los modelos o algoritmos de Machine Learning, donde no es recomendable tener muchos features debido a que existen variables que pueden llegar a ser irrelevantes las cuales aumentara el costo compuracional, además se corre el riesgo de tener valores faltantes los cuales se convierten en sesgos muy significativos y se perdera la capacidad de prediccion. Por ello se aplica las tecnicas de reduccion de la dimensionalidad, donde se utilizo los siguientes algortimos: PCA, IPCA y KPCA, este ultimo posee 3 kernels comunes, que son los lineales, polinomiales y gaussianos, donde de igual manera se va a elegir cual es le mejor de ellos.

Por otra parte, durante la ejecucion de estos algoritmos se realiza la division del conjunto de datos del dataset creado, en el cual se tiene 10 caracteristicas que se tomara como datos de entrenamiento y 1 caracteristica como datos de prueba. Tambien importamos 3 componentes o variables artificiales, a continuacion se presenta parte del codigo:



**Figura 2.** Algotimos PCA, IPCA y KPCA

Si se desea observar a detalle los algotirmos desarrollados se puede ingresar al siguiente link <https://github.com/Abel272000/skict-01> el cual es un repositorio creado en GitHub.

Los resultados obtenidos con los tres algortimos fueron los siguientes, los cuales fueron implementados o realizados con datos normales, datos normalizados y datos discretizados, donde sorprendentemente se obtuvieron los mismos resultados en cada uno de ellos, estos resultados se muestran a continuacion:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmos** | **Resultado** | **Analisis** | **Conlusion** |
| PCA | **0.9675376088677752** | Al ejecutar este algoritmo con el dataset seleccionado, se obtuvo el resultado de 0.9675 | Al analisar estos resultados se concluye que tanto el algortimo PCA, IPCA y KPCA Linear obtienen el mejor y mismo resultado con un 0.9675 |
| IPCA | **0.9675376088677752** | Este algortimo arroga un resultado de 0.9675 que comparado con el PCA su valor es identico a este. |
| KPCA LINEAR | **0.9675376088677752** | El resultado se puede simplificar en un 0.9675 el cual obtiene el mismo resultado que PCA y IPCA |
| KPCA POLY | 0.9588281868566905 | En este se logra observar que posee un resultado de 0.9588 |
| KPCA RBF | 0.6595407759303247 | Se obtiene un resultado de 0.6595, siendo este el resultado mas bajo. |

**Tabla 1.** Datos normales, normalizados y discretizados

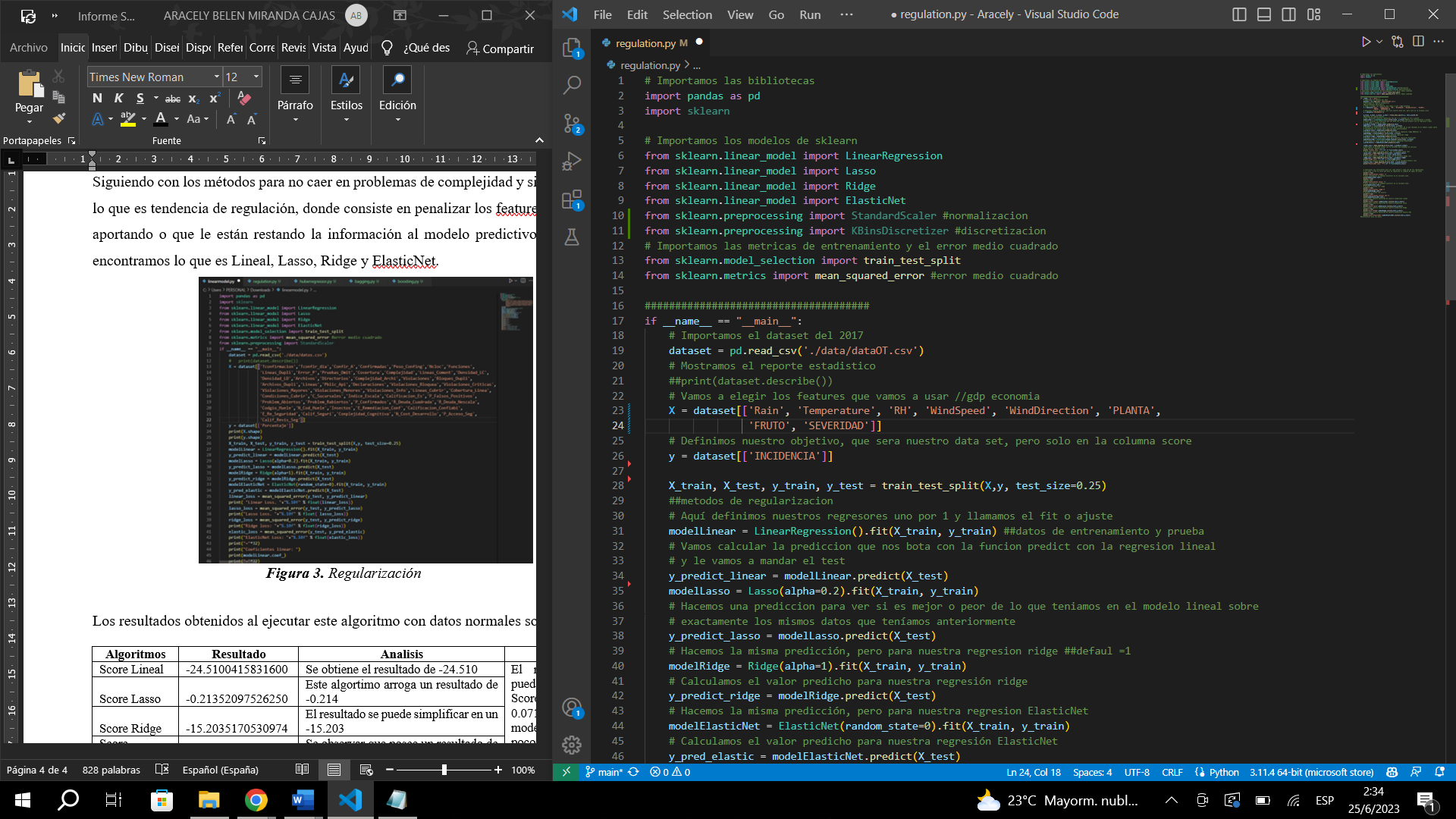
**Análisis**

Después de todas las pruebas realizadas y los datos analizados a través de la tabla presentada se puede decir que:

* Trabajando con datos normales, normalizados y discretizados se obtiene un mejor resultado con PCA, IPCA y KPCA Linear, así que se puede elegir y trabajar con cualquiera de estos tres algoritmos para el entrenamiento de un modelo predictivo.

1. **Tendencia de regulación**

Siguiendo con los métodos para no caer en problemas de complejidad y simpleza, tenemos lo que es tendencia de regulación, donde consiste en penalizar los features que no le están aportando o que le están restando la información al modelo predictivo, en esta técnica encontramos lo que es Lineal, Lasso, Ridge y ElasticNet.



**Figura 3.** Regularización

Por otra parte, en esta técnica nos arroga también datos los cuales nos indican cuál de los features que posee nuestro dataset nos arroga la variable que tiene el mayor peso. Los resultados obtenidos tanto con datos normales, normalizados y discretizados nos arrogan el resultado presentado a continuación:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algoritmos** | **Resultado** | **Variables** | **Analisis** | **Conlusion** |
| Score Lineal | [ **7.23217088e-01**  **1.31698394e-02**  -2.47729213e-03  -1.03106128e-01  -3.01107162e-05  -3.08045390e-03  -3.28433586e-03  **1.04077604e-02** ] | **Rain**  **Temperature**  RH  WindSpeed  WindDirection  Planta  Fruto  **Severidad** | Con los resultados obtenidos se observa que son 3 las variables que con el Score lineal obtienen mayor peso la cual la variable Rain o lluvia. | El resultado que se puede elegir es el de Score Lineal debido a que arrogar 3 variables con mayor peso al igual que Score Ridge con las mismas 3 variables, obteniendo la variable Rain o lluvia un valor de 72321708. |
| Score Lasso | [ 0.  0.  -0.  0.  -0.  -0.  -0.  **0.01056977** ] | Rain  Temperature  RH  WindSpeed  WindDirection  Planta  Fruto  **Severidad** | En el Score Lasso obtenemos una sola variable con mayor peso en este caso es la Severidad |
| Score Ridge | [**6.44883771e-01**  **1.36400101e-02**  -1.98320984e-03  -8.04306108e-02  -3.77166101e-06  -3.04767364e-03  -3.29294017e-03  **1.04229830e-02**] | **Rain**  **Temperature**  RH  WindSpeed  WindDirection  Planta  Fruto  **Severidad** | El resultado de Score Ridge posee de igual manera que el Score Lineal con 3 variables con mayor peso siendo de igual forma la variable Lluvia con mayor puntaje |
| Score ElasticNet | [ 0.  0.  -0.  0.  -0.  -0.  -0.  **0.010379** ] | Rain  Temperature  RH  WindSpeed  WindDirection  Planta  Fruto  **Severidad** | Se observar que posee una sola variable con mayor peso en este caso es severidad. |

Por otra parte, los siguientes datos que nos arroga dicho algoritmo con datos normales son los siguientes:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmos** | **Resultado** | **Analisis** | **Conlusion** |
| Score Lineal | 0.8188688560839352 | Se obtiene el resultado de 0.8189 | El resultado que se puede elegir es el de Score Elastic Net con 0.8137 debido a que este modelo que se ajusta un poco mejor al dataset. |
| Score Lasso | 0.8139025669211409 | Este algortimo arroga un resultado de 0.8139 |
| Score Ridge | 0.8192720262719972 | El resultado se puede simplificar en un 0.8193 |
| Score ElasticNet | **0.8133631615947737** | Se observar que posee un resultado de -0.8137 |

**Tabla 2.** Datos normales

Ahora para verificar si al normalizar los datos el resultado de cada uno de estos algoritmos mejora, se normalizo los datos de la data set, en el cual nos dieron el siguiente resultado:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmos** | **Resultado** | **Analisis** | **Conlusion** |
| Score Lineal | 0.83098209018589 | Al normalizar los datos este algoritmo obtiene el resultado de 0.8310 | Al analisar estos datos normalizados se observa que el modelo que obtiene menor errores es el Score ElasticNet con un 0.8205 |
| Score Lasso | 0.8210287669429691 | Este algortimo arroga un resultado de 0.8210 |
| Score Ridge | 0.8306285519199366 | El resultado se puede simplificar en un 0.8306 |
| Score ElasticNet | **0.8204538371623756** | En este se logra observar que posee un resultado de 0.8205 |

**Tabla 3.** Datos normalizados

En el caso anterior al normalizar los datos los resultados cambiaron. Entonces, ¿qué pasa si discretizamos los datos en esta técnica también? A continuación, se realizó esa interrogante, y estos fueron los resultados arrogados:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmos** | **Resultado** | **Analisis** | **Conlusion** |
| Score Lineal | 0.8168445746092333 | Con datos discretizados obtenemos 0.8168 | El resultado que se obtuvo al discretizar los datos que presenta menor errores es el de Score ElasticNet con 0.80 |
| Score Lasso | 0.8086331011252863 | Este algortimo arroga un resultado de 0.8086 |
| Score Ridge | 0.8167191749791375 | El resultado se puede simplificar en un 0.8167 |
| Score ElasticNet | **0.8076917959407464** | En este se logra observar que posee un resultado de 0.8077 |

**Tabla 4.** Discretizados

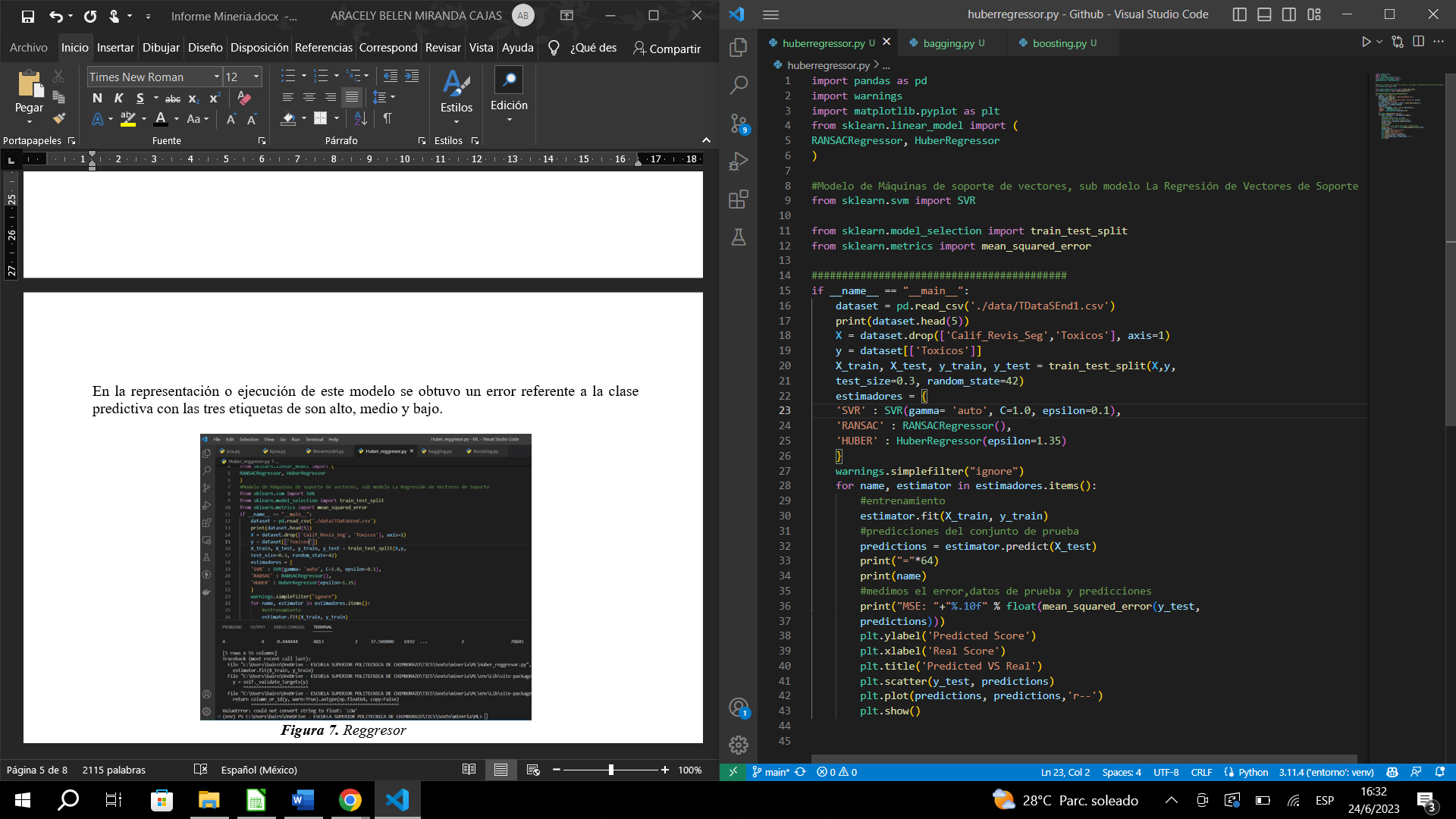
Culminando con las pruebas y el análisis de los datos en cada tabla realizada se llega a que:

* Trabajando con datos normales se considera que el resultado que arroga menos errores es el de Score ElasticNet con 0.8137
* Al normalizar los datos se elige a Score ElasticNet con un resultado de 0.8205.
* Y al discretizar los datos se obtienen que Score ElasticNet con 0.009 el que presenta el resultado con menor error.

Esto nos lleva a que con la técnica de regularización se obtiene resultados más convenientes trabajando con Score ElasticNet ya sea con datos normales, normalizados o discretizados este algoritmo va a arrogar el resultado con menores errores posibles. Por otra parte, si verificamos que variable es la que este está dando mayor peso son al campo de lluvia, temperatura y severidad

1. **Valores atípicos con regresiones robustas**

Hay que tener en cuenta los valores atípicos los cuales pueden estar presentes en nuestro data set, debido a que esto puede generar sesgos importantes en estos modelos, estos valores atípicos se los puede identificar a través de las regresiones robustas, en donde Sci-kit learn nos brinda RANSAC y Huber Regressor para abordar la existencia de los valores atípicos en nuestro data set, el modelo se presenta a continuación:



**Figura 4.** Reggresor

En este modelo se realizó las pruebas con los datos normales, normalizados y discretizados donde los resultados en los tres casos fueron los mismos y se presenta a continuación en forma general:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmos** | **Resultado** | **Analisis** | **Conlusion** |
| SVR | **0.1699674007** | Se obtiene el resultado de 0.1699 | El algoritmo SRV es quien indica un mejor rendimiento en terminos en los que el valor o resultado menor es el mejor. |
| RANSAC | 0.3404592241 | Este algortimo arroga un resultado de 0.3405 |
| HUBER | 0.1952528881 | El resultado se puede simplificar en un 0.1953 |

**Tabla 5.** Datos Normales, Normalizados y Discretizados

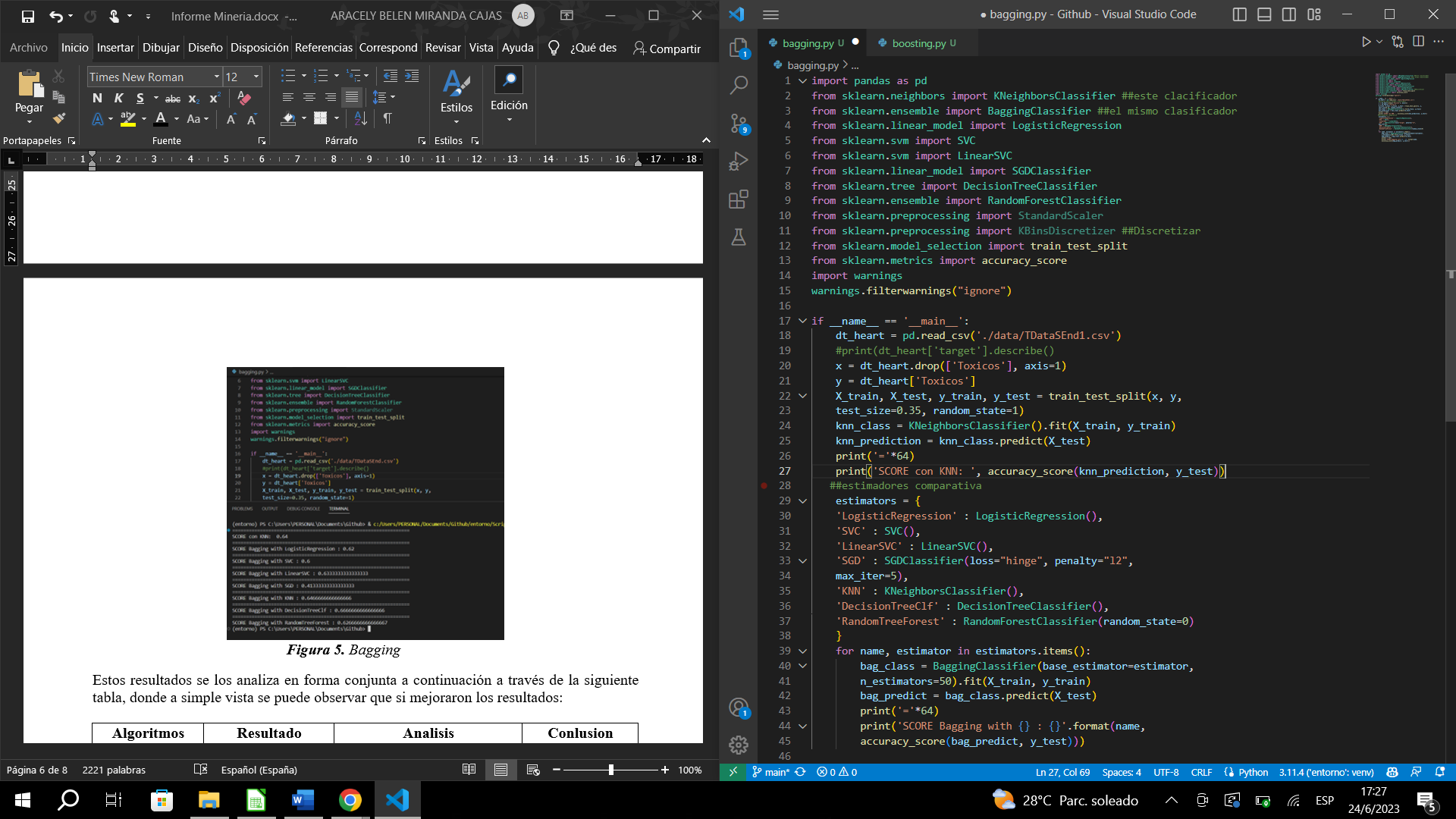
En los modelos de regresiones robustas el modelo con el cual se obtiene mejor rendimiento en términos en los que el valor o resultado menor es el mejor es SVR con un valor de 0.169

1. **Métodos de ensamble**

Lo siguiente es emplear los métodos de ensamble, el cual combina diferentes métodos con diferentes configuraciones donde aplica un método para lograr un consenso, posee dos estrategias.

**Estrategia Bagging**

En esta estrategia se reduce la varianza de los clasificadores individuales combinándolos a continuación se presenta estrategia:



**Figura 5.** Bagging

Los resultados obtenidos se los analiza a continuación:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmos** | **Resultado** | **Analisis** | **Conlusion** |
| SCORE con KNN | 0.9776119402985075 | Se obtiene el resultado de 0.9776 | Despues de observar y analizar los resultados se puede decir que en este caso el algoritmo que nos arroga el mejor resultado es RandomTreeForest con 0.9864 |
| SCORE Bagging with LogisticRegression | 0.9681139755766621 | Este algortimo arroga un resultado de 0.9681 |
| SCORE Bagging with SVC | 0.9694708276797829 | Este algoritmo con el dataset seleccionado, obtiene el resultado de 0.9695 |
| SCORE Bagging with LinearSVC | 0.9606512890094979 | Este algortimo arroga un resultado de 0.9607 |
| SCORE Bagging with SGD | 0.9755766621438263 | El resultado se puede simplificar en un 0.9756 |
| SCORE Bagging with KNN | 0.9789687924016283 | En este se logra observar que posee un resultado de 0.9790 |
| SCORE Bagging with DecisionTreeClf | 0.9850746268656716 | En este se obtiene un resultado de 0.9851 |
| SCORE Bagging with RandomTreeForest | **0.9864314789687924** | Y por ultimo este arroga un resultado de 0.9864 |

**Tabla 6.** Datos Normales

Ahora se va analizar los resultados con datos normalizados:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmos** | **Resultado** | **Analisis** | **Conlusion** |
| SCORE con KNN | 0.9776119402985075 | Se obtiene el resultado de 0.9776 | Al momento de normalizar los datos existe un ligero incremento en cada resultado de cada modelo donde RandomTreeForest es quien sigue teniendo el mejor resultado esta ves con 0.9891 |
| SCORE Bagging with LogisticRegression | 0.9687924016282226 | Este algortimo arroga un resultado de 0.9688 |
| SCORE Bagging with SVC | 0.9694708276797829 | Este algoritmo con el dataset seleccionado, obtiene el resultado de 0.9695 |
| SCORE Bagging with LinearSVC | 0.9606512890094979 | Este algortimo arroga un resultado de 0.9607 |
| SCORE Bagging with SGD | 0.9769335142469471 | El resultado se puede simplificar en un 0.9769 |
| SCORE Bagging with KNN | 0.9776119402985075 | En este se logra observar que posee un resultado de 0.9776 |
| SCORE Bagging with DecisionTreeClf | 0.9837177747625508 | En este se obtiene un resultado de 0.9837 |
| SCORE Bagging with RandomTreeForest | **0.989145183175034** | Y por ultimo este arroga un resultado de 0.9891 |

**Tabla 7.** Normalizacion

También se analizan los datos discretizados

|  |  |  |  |
| --- | --- | --- | --- |
| **Algoritmos** | **Resultado** | **Analisis** | **Conlusion** |
| SCORE con KNN | 0.9776119402985075 | Se obtiene el resultado de 0.9776 | Con datos discretizados se concluye que con RandomTreeForest se siguie obteniendo el mejor resultado esta vez con 0.9864 |
| SCORE Bagging with LogisticRegression | 0.9687924016282226 | Este algortimo arroga un resultado de 0.9688 |
| SCORE Bagging with SVC | 0.9728629579375848 | Este algoritmo con el dataset seleccionado, obtiene el resultado de 0.9729 |
| SCORE Bagging with LinearSVC | 0.9626865671641791 | Este algortimo arroga un resultado de 0.9627 |
| SCORE Bagging with SGD | 0.9748982360922659 | El resultado se puede simplificar en un 0.9749 |
| SCORE Bagging with KNN | 0.9789687924016283 | En este se logra observar que posee un resultado de 0.9790 |
| SCORE Bagging with DecisionTreeClf | 0.9850746268656716 | En este se obtiene un resultado de 0.9851 |
| SCORE Bagging with RandomTreeForest | **0.9864314789687924** | Y por ultimo este arroga un resultado de 0.9864 |

**Tabla 8.** Discretizacion

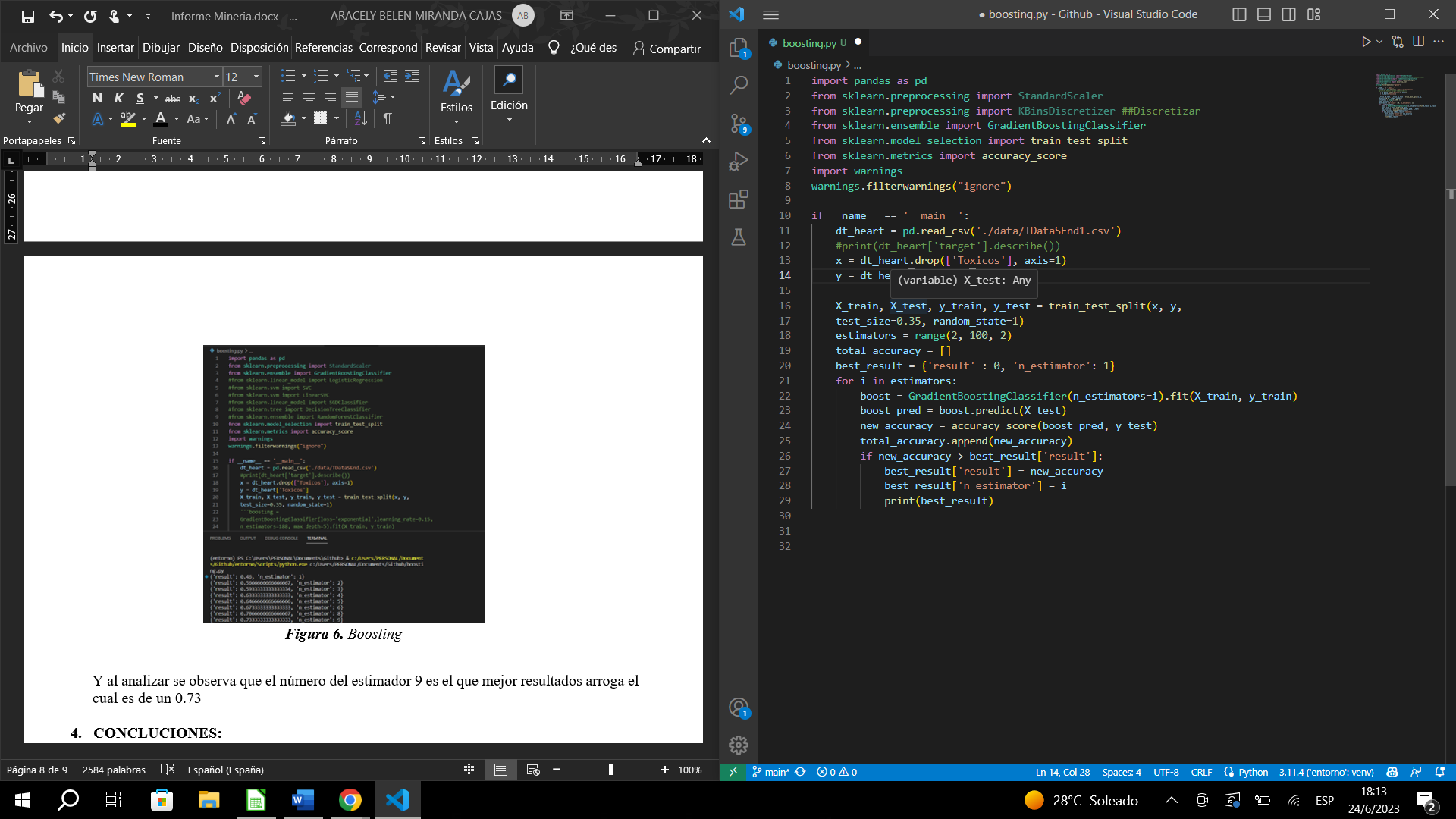
Una vez terminado con el análisis de cada tabla realizada se puede decir que:

* Trabajando con datos normales se considera que el mejor resultado obtenido es de RandomTreeForest con 0.9864
* Al normalizar los datos el mejor resultado fue de es RandomTreeForest con 0.9891
* Y al discretizar los datos se obtiene como mejor resultado al es RandomTreeForest con 0.9864 al igual al resultado arrogado con datos normales

Esto nos lleva a que con la estrategia de Bagging se obtiene resultados más convenientes trabajando con datos normalizados donde el RandomTreeForest arroja en los tres casos los mejores resultados o lo resultados más altos.

**Estrategia Boosting**

La otra estrategia es Boosting el cual es un método secuencial donde busca fortalecer gradualmente un modelo de aprendizaje usando siempre el error residual de las etapas anteriores donde el resultado final también se consigue por consenso entre todos los modelos, esta estrategia se presenta a contracción:



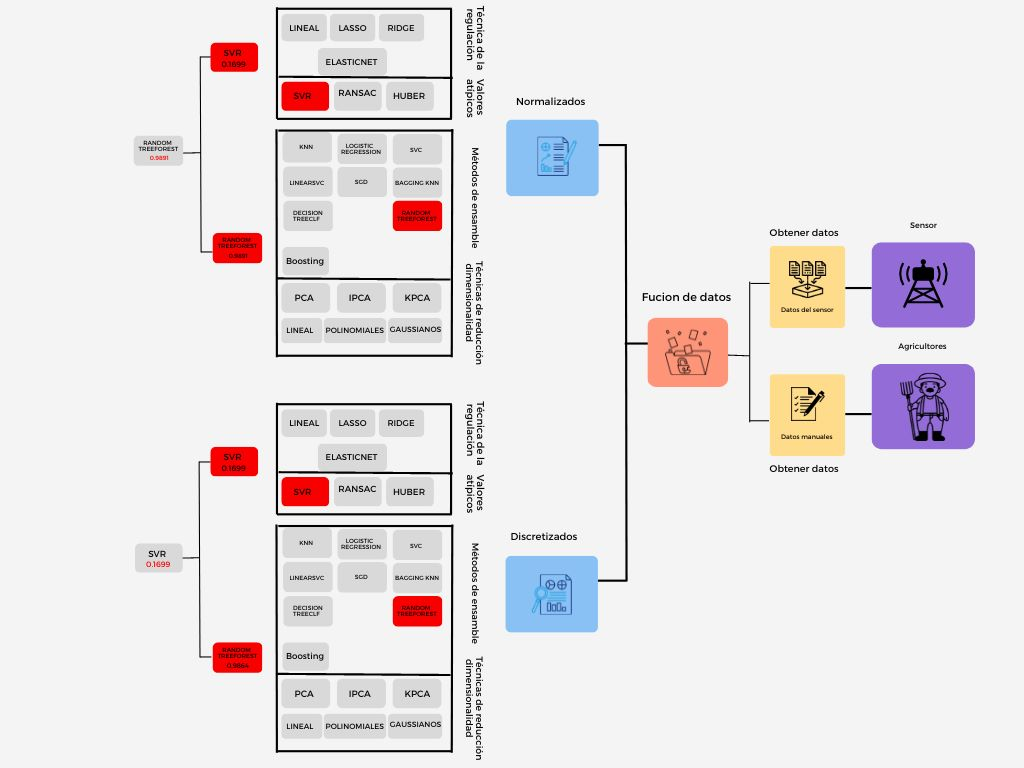
**Figura 6.** Boosting

Los resultados que se obtuvieron con los datos fueron:

|  |  |  |  |
| --- | --- | --- | --- |
| **Estimador** | **Resultado** | **Analisis** | **Conlusion** |
| **Datos Normales** | | | Al comparar los resultados de los tres estimadores finales, se observa que en los tres casos se obtine 0.9905 |
| Estimador 4 | **0.9905020352781547** | Se obtiene el resultado de 0.9905 |
| **Datos Normalizados** | | |
| Estimador 4 | **0.9905020352781547** | Este algortimo arroga un resultado de 0.9905 |
| **Datos Discretizados** | | |
| Estimador 4 | **0.9905020352781547** | Este algoritmo con el dataset seleccionado, obtiene el resultado de 0.9905 |

**Tabla 9.** Resultados Generales

Así se llega a la conclusión de que al trabajar con Bosting, ya sea con datos normales, normalizados o discretizados se va a obtener los mismos resultados.

**Análisis Global de todos los resultados obtenidos**

(6)

(5)

(4)

(3)

(1)

(2)

**DETALLE**

**Acorde a la numeración realizada en la gráfica anterior:**

1. Como se detalló al principio de este informe, para la realización de este modelo predictivo se necesitó los datos que arrogan un sensor y los datos que nos brindan los agricultores, estos promotores se encuentran en el anterior grafico en los recuadros de color morado.
2. La información que arroga cada uno de estos se almacena en documentos en Excel donde por parte del sensor tenemos los datos como la lluvia, velocidad de viento, temperatura, velocidad de ráfaga, etc. Por otra parte, está la información que los agricultores nos proporcionan algunos de ellos es el número de la planta, el número de frutos la severidad y la incidencia la cual en esta variable la vamos a declarar como la clase predictiva, esta información o datos los puede identificar en el recuadro de color amarillo.
3. Luego de esto tenemos un recuadro de color anaranjado encontramos lo que es la unión de estos datos los cuales pasan por un proceso o un tratamiento a dichos datos dando lugar así a tener el data set final.
4. Una vez obtenidos estos datos ejecutamos de forma normalizada los datos, por cada algoritmo que posee cada grupo formado como se logra observar en el recuadro superior izquierda, en el cual encontramos los algoritmos de Tendencia de regulación (Lineal, Lasso, Ridge y ElasticNet) y Valores atípicos con regresiones robustas (RANSAC y Huber Regressor ), aquí el algoritmo que nos arroga el menor resultado es SVR con el valor 0.1699, en el recuadro de abajo encontramos lo que es las Técnicas de la reducción de la dimensionalidad (PCA, IPCA y KPCA: lineales, polinomiales y gaussianos) y Métodos de ensamble (Bagging y Boosting), donde el algoritmo que nos arroga el mejor resultado es el RandomTreeForest con 0.9891.
5. Ahora al discretizar los datos se observa que con los algoritmos de Tendencia de regulación (Lineal, Lasso, Ridge y ElasticNet) y Valores atípicos con regresiones robustas (RANSAC y Huber Regressor ), aquí el algoritmo que nos arroga el menor resultado es SVR con el valor 0.1699, en el recuadro de abajo encontramos lo que es las Técnicas de la reducción de la dimensionalidad (PCA, IPCA y KPCA: lineales, polinomiales y gaussianos) y Métodos de ensamble (Bagging y Boosting), donde el algoritmo que nos arroga el mejor resultado es el RandomTreeForest con 0.9864.
6. Llegando así a la conclusión que se va a utilizar los datos normalizados debido a que nos arroga mejores resultados con el algoritmo RandomTreeForest con 0.9864 y por parte de los datos discretizamos se puede trabajar con SVR con el valor 0.1699 debido a que brinda el menor resultado de error.

**VALIDACIÓN DE MODELOS**

Después de entrenar un modelo de Machine Learning con datos etiquetados, se supone que tiene que funcionar con nuevos datos. El proceso de validación consiste en decidir si los resultados digitales que cuantifican las relaciones hipotéticas entre las variables son aceptables como descripciones de los datos.

algunos aspectos que tienes es que:

* La última palabra la tienen los datos
* Necesitamos mentalidad de testeo
* No todos los modelos son malos, solamente que algunos resultan útiles.

Una de las técnicas más empleadas para probar la eficacia de un modelo de Machine Learning es la «cross-validation» o validación cruzada. Este método también es un procedimiento de «re-sampling» que permite evaluar un modelo incluso con datos limitados.

**Las principales técnicas de validación cruzada**

1. **Dividir nuestros datos en Entrenamiento / prueba (Hold-On)**

Aspectos de esta técnica

* Se requiere un prototipado rápido
* No se tiene mucho conocimiento
* No se cuenta con abundante poder de cómputo

1. **Usar validación Cruzada (K-Folds)**

Justamente, permite garantizar que todas las observaciones de la serie de datos original tengan la oportunidad de aparecer en la serie de entrenamiento y en la serie de prueba. En caso de datos de entrada limitados, resulta uno de los mejores enfoques. Primero se empieza separando la serie de datos de manera aleatoria en K folds. El valor de K no debe ser ni demasiado bajo ni demasiado alto y, por lo general, se elige un valor comprendido entre 5 y 10 en función de la envergadura de la serie de datos.

Por ejemplo, si K=10, la serie de datos se dividirá en 10 partes. El proceso se repite hasta que cada K-fold sirva dentro de la serie de entrenamiento. La función cross\_val\_score indica la puntuación de cada fold de prueba, mientras que la función cross\_val\_predict indica la puntuación predicha para cada observación de la serie de datos de entrada cuando formaba parte de la serie de prueba. De este modo, los datos de los folds de entrenamiento y de prueba se distribuyen de manera equitativa.

Los folds se pueden utilizar como iteradores o en un bucle para entrenar en un marco de datos de Pandas.

**Aspectos importantes**

* Recomendable en la mayoría de los casos
* Se cuenta con un equipo suficiente para desarrollar ML
* Se requiere la integración con técnicas de optimización paramétrica
* Se tiene más tiempo para las pruebas

1. **Validación Cruzada (LOOCV)**

La validación cruzada, o validación cruzada de pliegue k, es un procedimiento utilizado para estimar el rendimiento de un algoritmo de aprendizaje automático cuando se hacen predicciones sobre datos no utilizados durante el entrenamiento del modelo.

La validación cruzada tiene un solo hiperparámetro «k«que controla el número de subconjuntos en los que se divide un conjunto de datos. Esto significa que la validación cruzada del pliegue k implica el ajuste y la evaluación k modelos. Esto, a su vez, proporciona k estimaciones del rendimiento de un modelo en el conjunto de datos, que se pueden comunicar utilizando estadísticas resumidas como la media y la desviación estándar.

**Aspectos importantes**

* Se tiene gran poder de computo
* Se cuenta con pocos datos como para dividir por Training / Test
* Personas que sufren de TOC y quieren probar todos los casos posibles.

**OPTIMIZACIÓN PARAMÉTRICA**

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

En general, la optimización paramétrica se realiza para encontrar valores de un conjunto de

parámetros (o variables de decisión) que optimizan alguna medida de rendimiento

(generalmente, minimizan un costo o maximizan una recompensa). Antiguamente, en el

campo de la investigación de operaciones, la optimización paramétrica es realizada usando

programación matemática sea el ejemplo lineal, no lineal e programación integral.

La optimización paramétrica a menudo es llamada optimización estática, porque la solución

es un conjunto de parámetros preestablecidos “estáticos” para todos los estados.

Se realiza para encontrar valores de un conjunto de parámetros o variables de decisión que optimizan alguna medida de rendimiento. Antiguamente, en el campo de la investigación de operaciones, la optimización paramétrica es realizada usando programación matemática sea el ejemplo lineal, no lineal y programación integral. La optimización paramétrica a menudo es llamada optimización estática, porque la solución es un conjunto de parámetros preestablecidos "estáticos" para todos los estados.

**Enfoques**

1. **Optimización por grilla de parámetros. GridSearchCV**

La grilla de parámetros nos define grupos de parámetros que serán probados en todas sus combinaciones, un grupo a la vez. Se ocupa cuando se quiera realizar un estudio a fondo sobre las implicaciones de los parámetros, y además se tenga el tiempo y el poder de procesamiento requerido

**Aspectos**

* Definir una o varias métricas que queremos optimizar
* Identificar los posibles valores que pueden tener los parámetros
* Crear un diccionario de parámetros
* Usar Cross Validation
* Entrenar el modelo

1. **Optimización por búsqueda Aleatorizada. RandomizedSearchCV**

En este método, definimos escalas de valores para cada uno de los parámetros seleccionados, el sistema probara varias iteraciones y mostrara la mejor combinación encontrada. Se utiliza cuando se quieran explorar posibles optimizaciones, además cuando haya poco tiempo o poco poder de procesamiento.

**Aspectos**

* Definir una o varias métricas que queremos optimizar.
* Identificar los rangos de valores que pueden tomar ciertos parámetros.
* Crear un diccionario de rangos de valores.
* Usar Cross Validation.
* Entrenar el modelo.

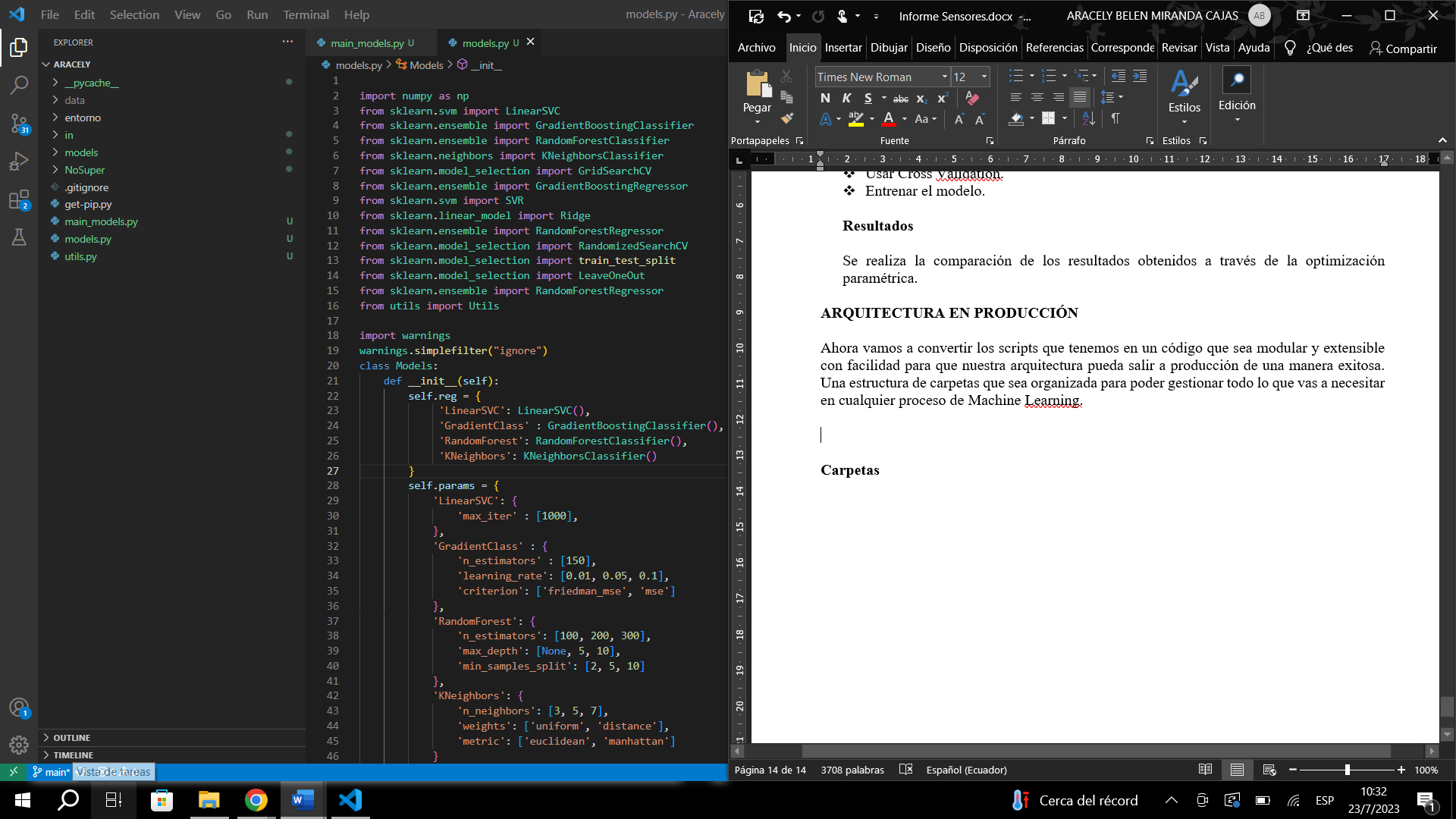
**Resultados**

A continuación, se realizará la comparación de los resultados que se obtienen de las tres validaciones, donde con la validación de K-Folds se utiliza los enfoques de optimización paramétrica.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tecnicas** | | **Resultado** | **Analisis** | **Conlusion** |
| Hold-On | | 0.9595065491332347 | Esta tecnica arroja un resultado de 0.96 en el cual en la mayoria son aceptables las variables. | Despues de esta comparación, ya se logra tener a un ganadar, y este es la tecnica de K-Folds ejecutando por busqueda aleatoria el cual tiene el mayor resultado de 0.979 en las acaptación de variables. |
| K-Folds | Grid Search CV | 0.9776740115300343 | K-Folds ejecuntando por grillas de parametros, nos da un resultado de 0.978 |
| Randomized Search CV | 0.9790995282798561 | K-Folds ejecuntando por busqueda Aleatoria, nos da un resultado de 0.979 |
| LOOCV | | 0.9040576854573047 | En LOOCV arroja un resultado de 0.90 en el cual tiene un porcenteaje menor que los anteriores sobre las variables aceptables. |

**ARQUITECTURA EN PRODUCCIÓN**

Ahora vamos a convertir los scripts que tenemos en un código que sea modular y extensible con facilidad para que nuestra arquitectura pueda salir a producción de una manera exitosa. Una estructura de carpetas que sea organizada para poder gestionar todo lo que vas a necesitar en cualquier proceso de Machine Learning.



**Figura 7.** Arquitectura

**Carpetas:**

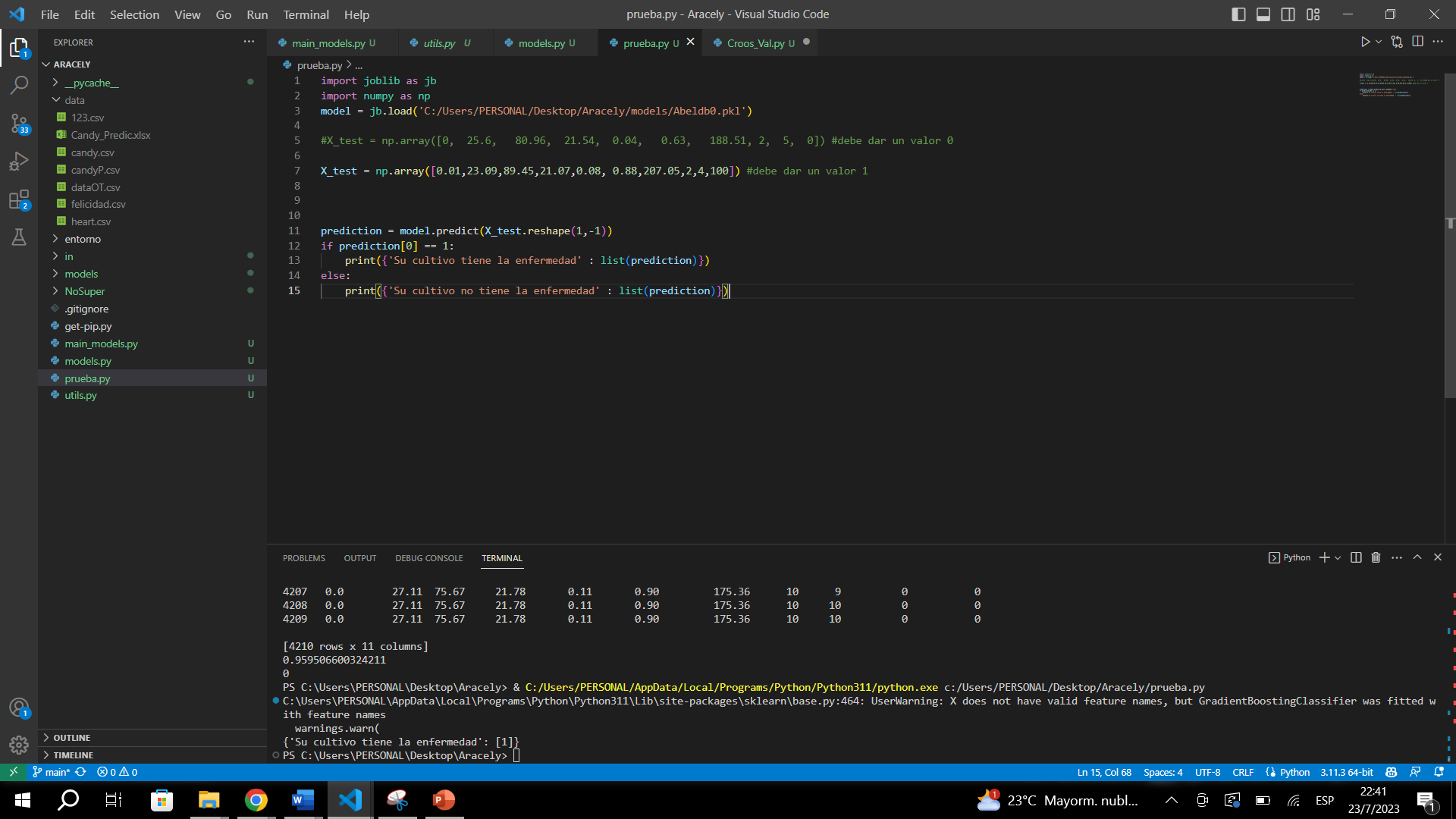
* **data:** Esta carpeta contiene el resultado de la exportación de nuestros modelos, visualizaciones, datos en Excel o csv, etc.
* **entorno:**  se encuentran las librerías y los scripts.
* **in:** Contiene archivos de entrada, datos que alimentarán a nuestros modelos.
* **models:** Aquí se encontrarán los modelos.

**Archivos:**

* **main\_models.py:** Este es el método principal de ejecución. Ejecutará todo el flujo de datos. Se encargaría de controlar el flujo de todo el código de principio a fin.
* **utils.py:** Todos los métodos que se reutilizaran una y otra vez, escalamiento
* **models.py:** Irá toda la parte de Machine Learning como tal.

**VARIFICACIÓN**

En un documento con extensión py importamos la librería joblib y numpy en la cual la librería joblib se ocupa para cargar el archivo de predicción que anteriormente se creó, se ingresa una de las filas de datos que posee el data set sin incluir el dato de la variable predictiva, para así verificar que el modelo arroje resultados correctos.



**Figura 8** *Verificación*

Y como se observa su resultado es el correcto, el modelo funciona correctamente.