| DATOS PERSONALES | FIRMA |
| --- | --- |
| |  |  | | --- | --- | | Nombre: Juan José | DNI: 0604067256 | |  |
| Apellidos: Flores Fiallos |

| ESTUDIO | ASIGNATURA | CONVOCATORIA |
| --- | --- | --- |
| MÁSTER UNIVERSITARIO EN INGENIERÍA MATEMÁTICA Y COMPUTACIÓN (PLAN 2016) | 4391020006.- TÉCNICAS MULTIVARIANTES | Ordinaria Número periodo 1823 |

| FECHA | MODELO | CIUDAD DEL EXAMEN |
| --- | --- | --- |
| 14-16/01/2022 | Modelo - C |  |

| Etiqueta identificativa |
| --- |
|  |

**INSTRUCCIONES GENERALES**

1. Ten disponible tu documentación oficial para identificarte, en el caso de que se te solicite.
2. Si tu examen consta de una parte tipo test, indica las respuestas en la plantilla según las características de este.
3. Debes contestar en el documento adjunto, respetando en todo momento el espaciado indicado para cada pregunta. Si este es en formato digital, los márgenes, el interlineado, fuente y tamaño de letra vienen dados por defecto y no deben modificarse. En cualquier caso, asegúrate de que la presentación es suficientemente clara y legible. Entrega toda la documentación relativa al examen, revisando con detenimiento que los archivos o documentos son los correctos. El envío de archivos erróneos o un envío incompleto supondrá una calificación de “no presentado”.
4. Durante el examen y en la corrección por parte del docente, se aplicará el Reglamento de Evaluación Académica de UNIR que regula las consecuencias derivadas de las posibles irregularidades y prácticas académicas incorrectas con relación al plagio y uso inadecuado de materiales y recursos.
5. No está permitido el uso de Internet ni ningún tipo de comunicación con otra persona.Durante todo el examen tu teléfono móvil debe estar en modo avión.
6. La parte principal de cada pregunta consiste en interpretar y comentar los resultados obtenidos. Si te limitas a hacer los cálculos no vas a poder superar el examen.
7. Es fundamental que las respuestas estén debidamente redactadas, de forma clara y precisa y sin faltas de ortografía.
8. Para hacer el examen puedes utilizar los apuntes del curso y los scripts que hayas preparado y Python para hacer los cálculos.

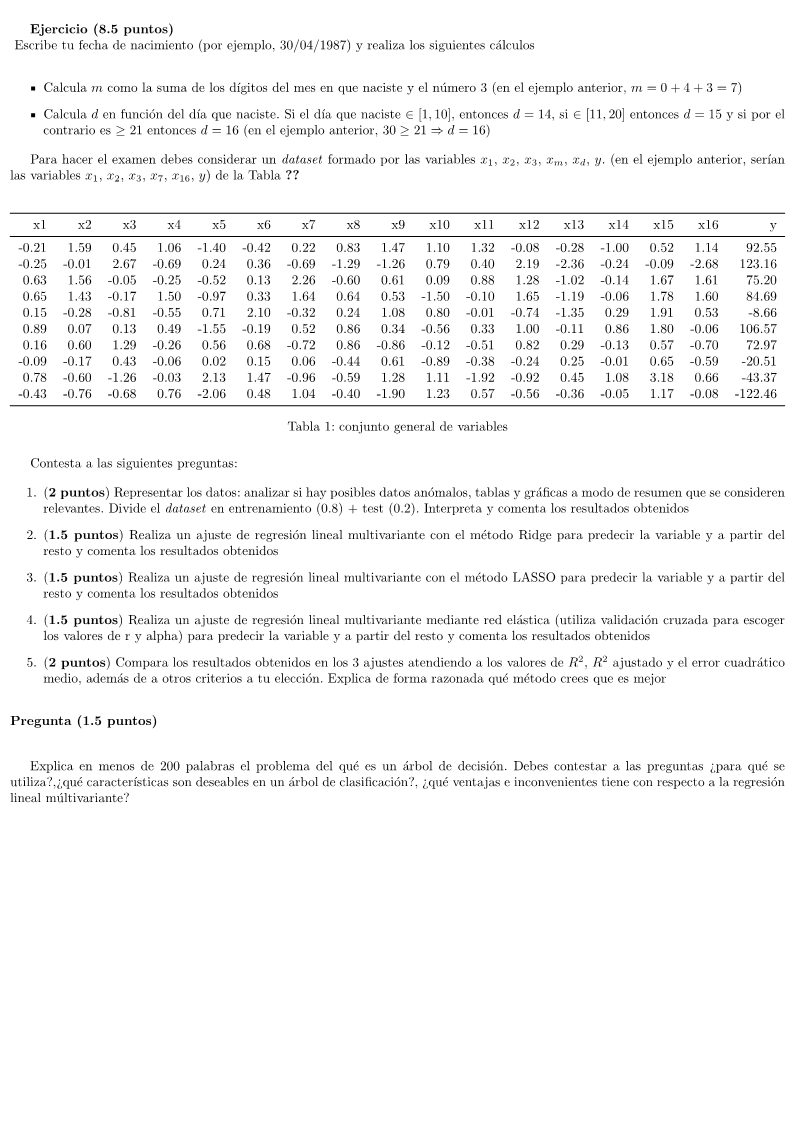
**Puntuación**

**Preguntas**

* Puntuación máxima 10.00 puntos

El examen constará de un ejercicio práctico (8,5 puntos) y una pregunta teórica (1,5 puntos). Los enunciados están en la página 14 y el espacio para responder el examen está entre las práginas 4 y 13.  
  
**1.** Pregunta

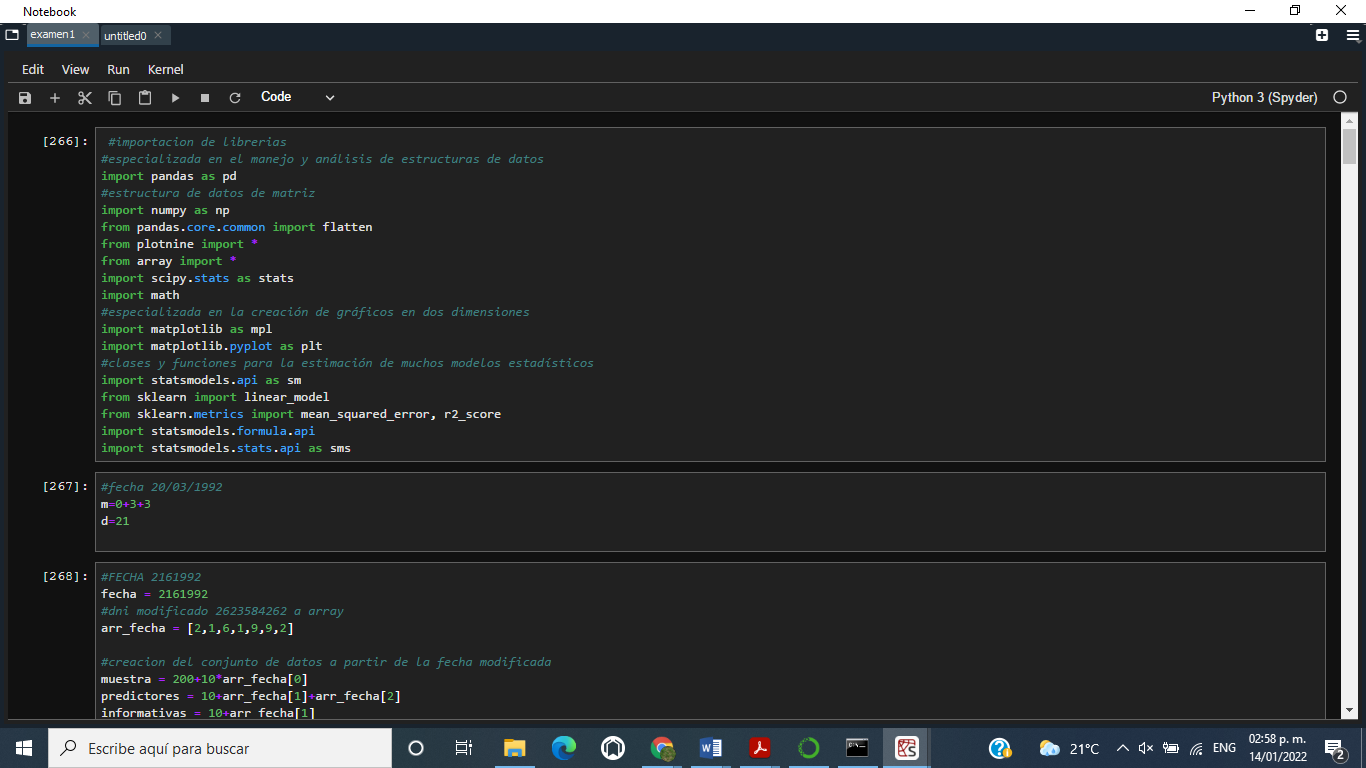
 (Responder en 10 caras)



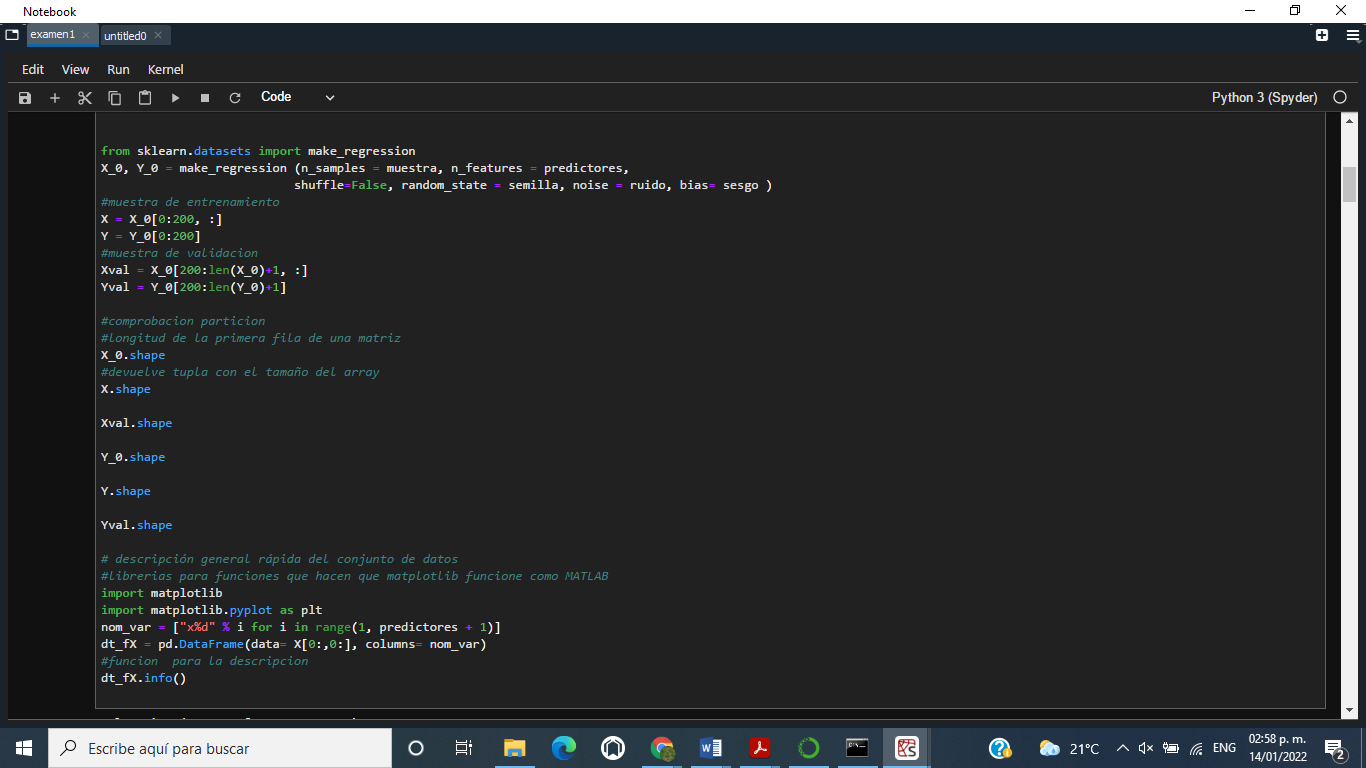
Pregunta 1

Fecha de nacimiento 20/03/1992

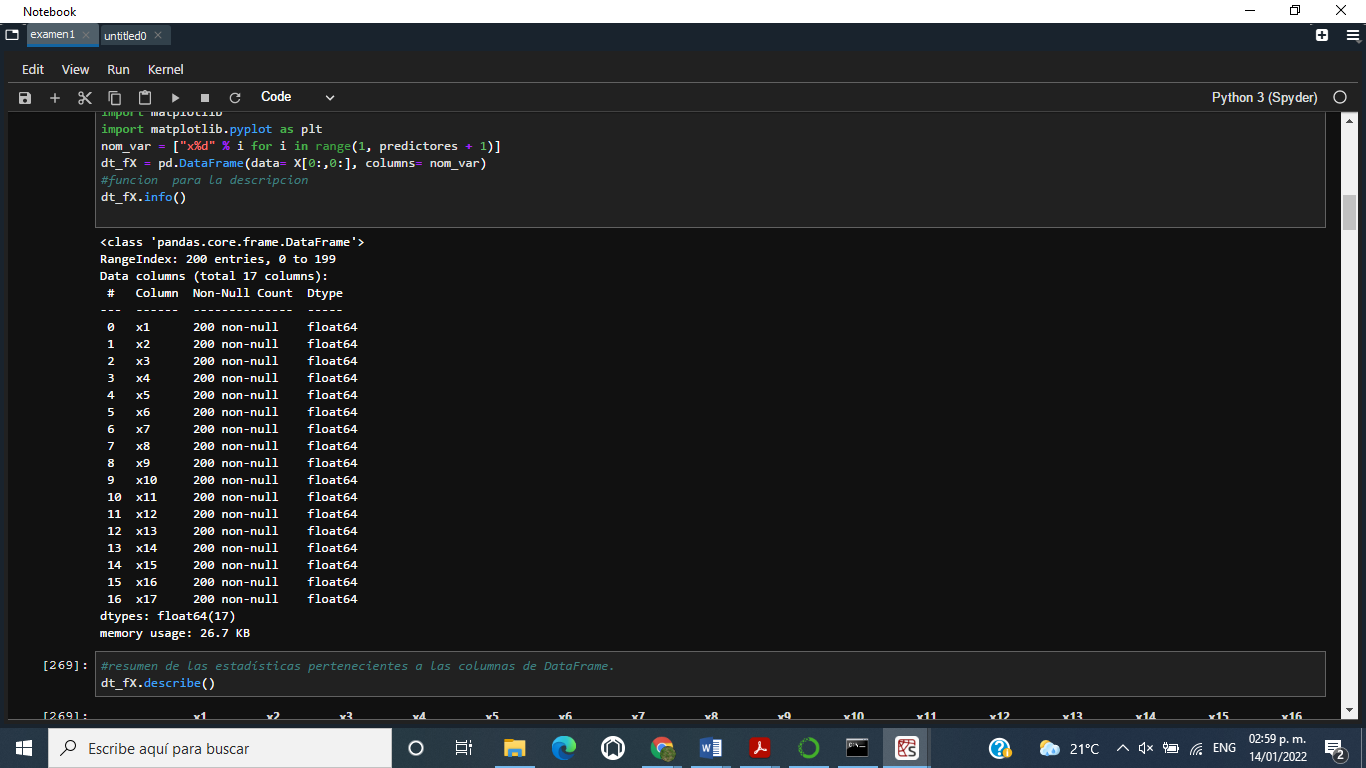
Creacion de dataframe mediane fecha, mes y dia



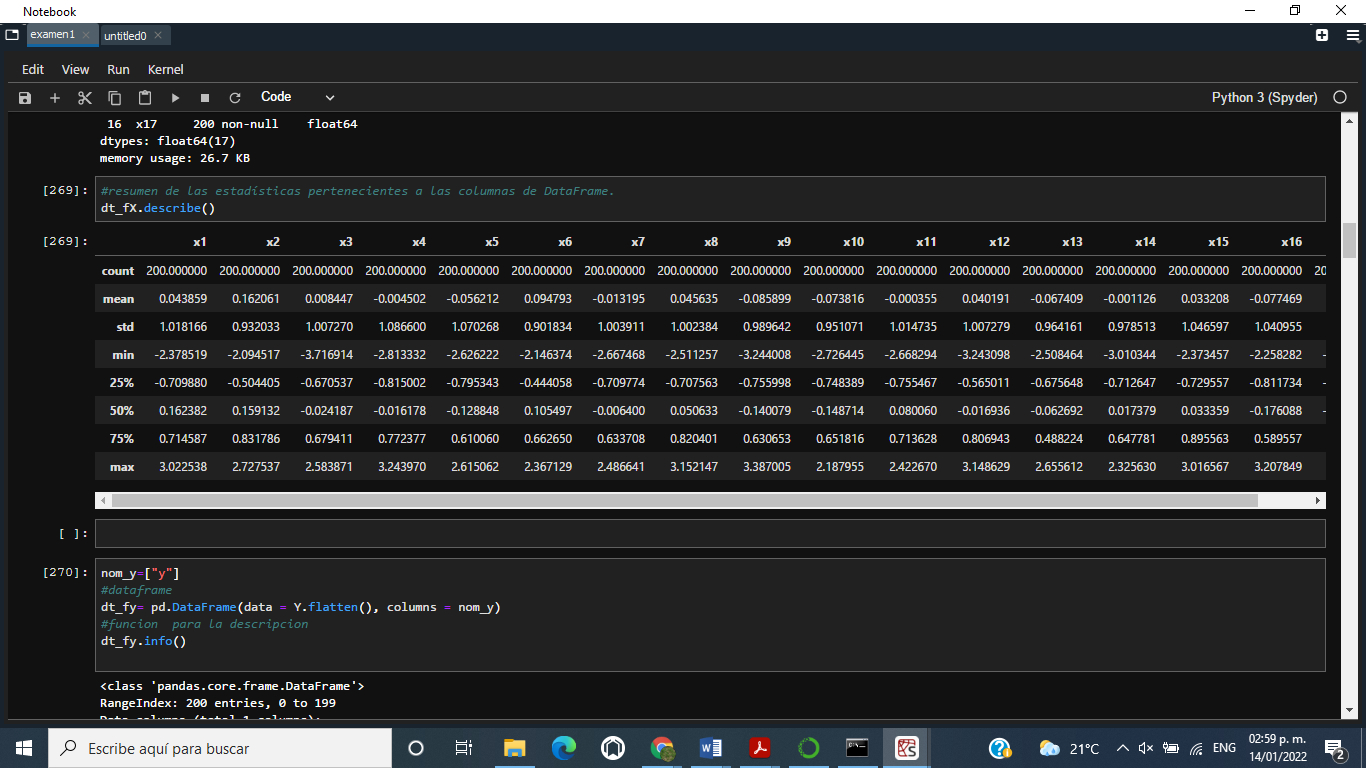
Realizacion de la muestra con entrenamiento y prueba se toma en cuenta la determinacion de los datos previos obtenidos de la generacion del dataframe



Una vez obtenido los resultados imprimimos losm mismos tomando en cuenta las variables predictoras mas optimas



Descriptivo de la tabla del dataframe creado



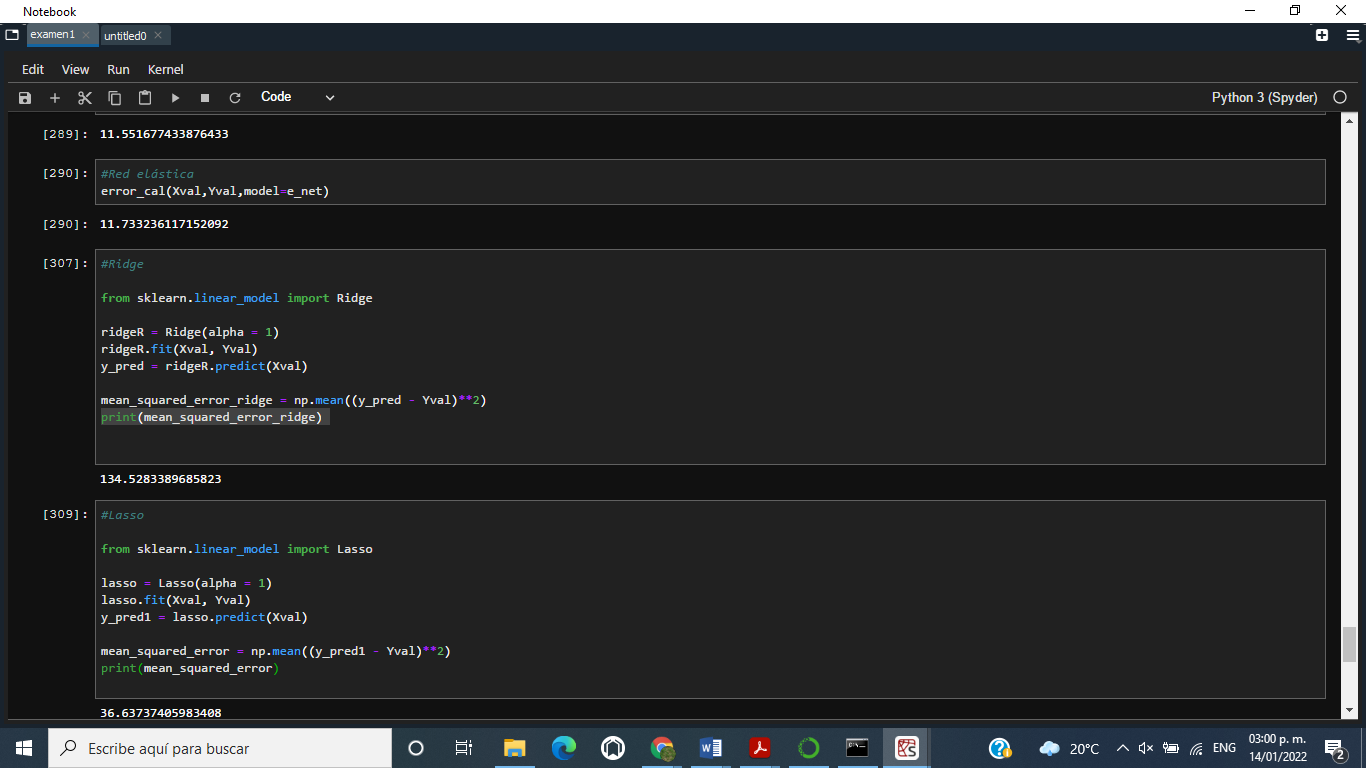
Representacion grafica de los datos obtenidos



2

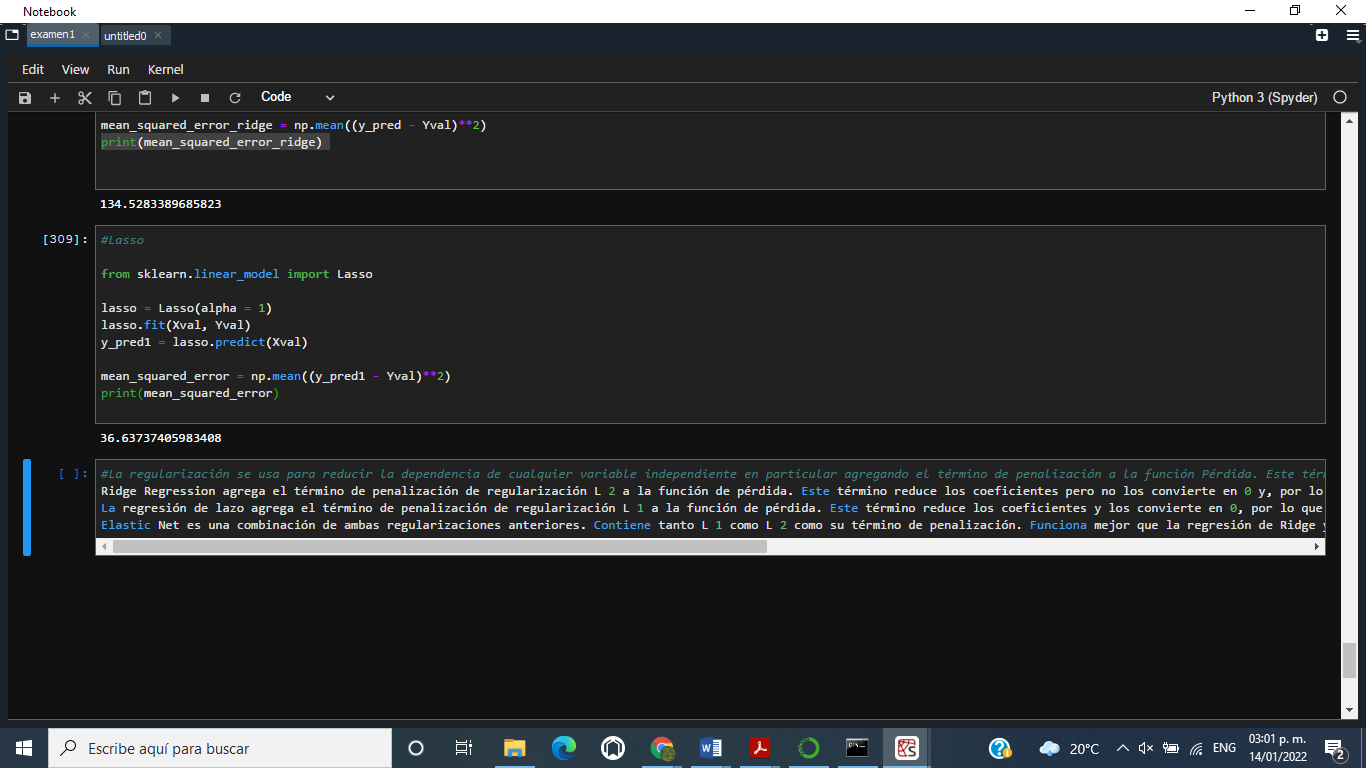
Ridge

Metodo de regresion de Ridge para toma de errores



3 Lasso

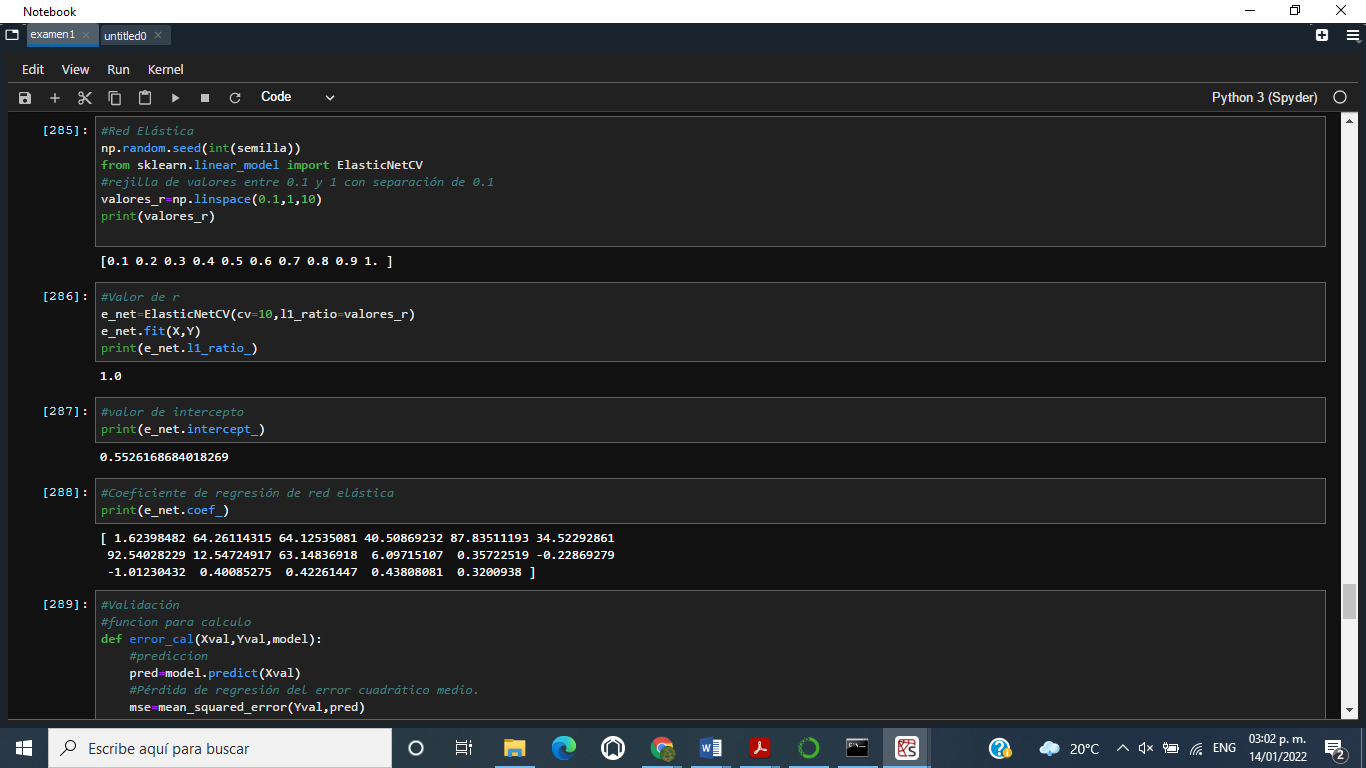
Metodo de regresion de Laso



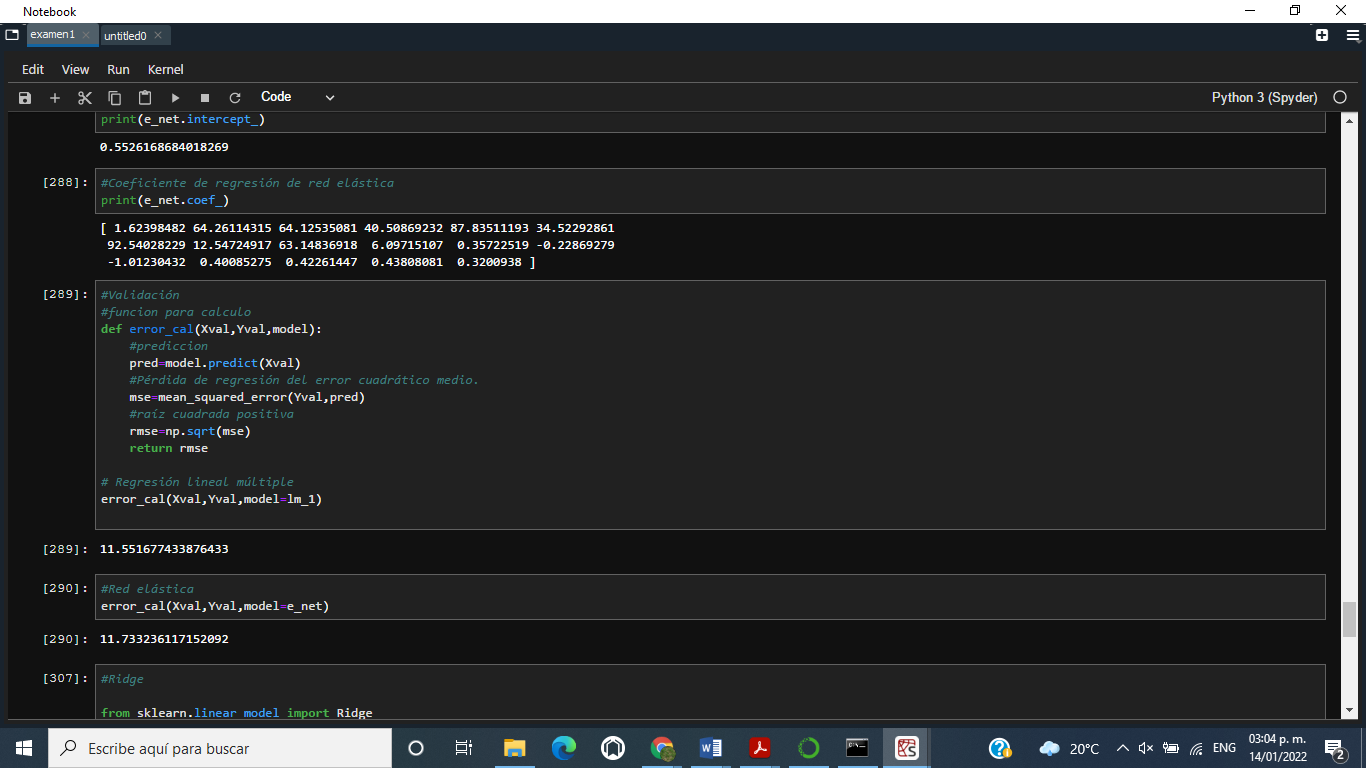
4.

Elastica

Estimaciones realizadas para encontrar el coeficiente de regresion lineal



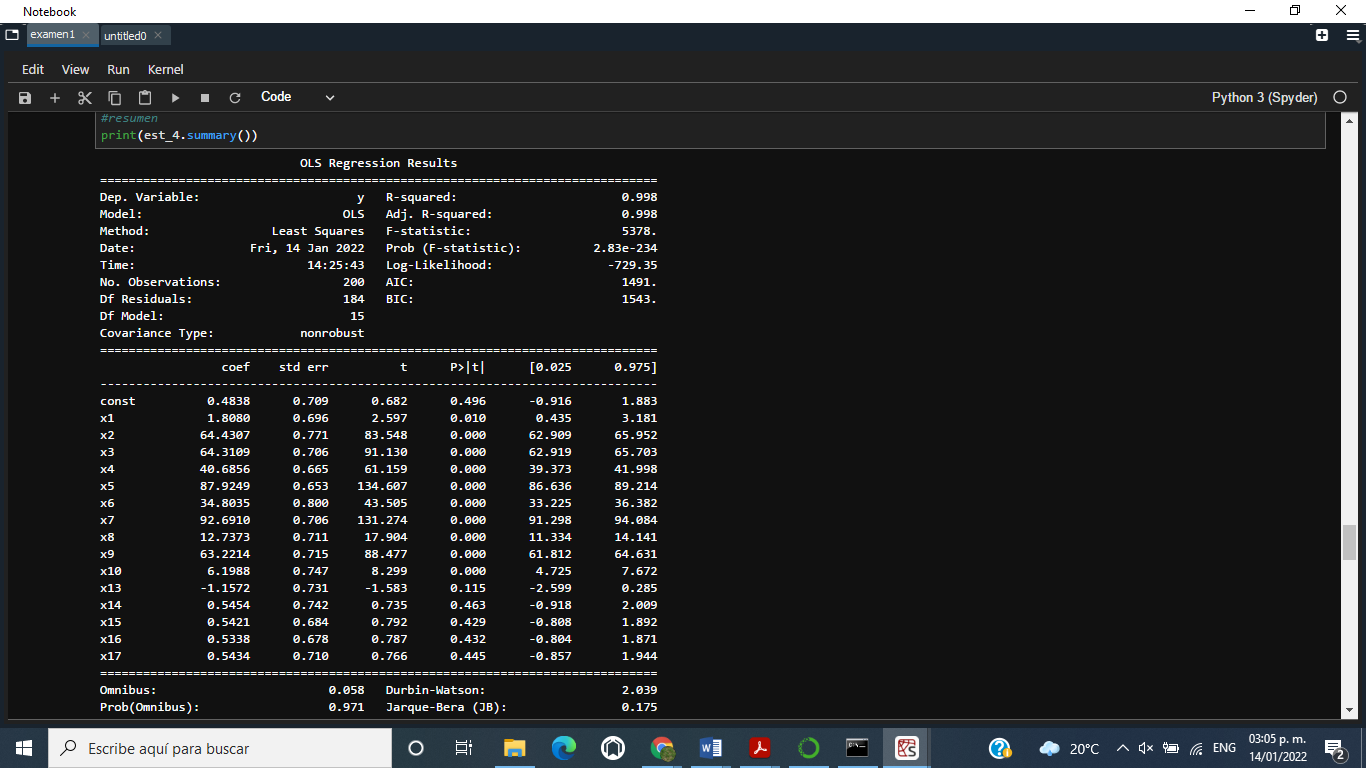
Comparacion de la red elastica para la validacion de la regresion lineal



5.

MEDIANTE EL CALCULO DE R2

Una vez implementados los modelos el calculo del r2 nos da una estimacion aproxima a 1 con un valor de 0.998 el cual brinda una gran aceptacion de los caluclos establecidos a raiz del dataframe



Tambien cabe acotar que entre los procedimientos realizados mediante los 3 metodos de ajustes, ridge , laso, elastica, la red elastica brinda mejores resultados en los calculos realizados

Pregunta 2

Arboles

Un árbol de decisión consiste en secuencias de condiciones sobre distintas variables predictoras y su relación con la variable de respuesta. Estas condiciones se van sucediendo en varios caminos del árbol, y por tanto, las posibilidades se dividen en ramas. Estas ramas representan las divisiones del espacio de los predictores en diferentes regiones.

Una vez definido el árbol, para poder realizar predicciones se emplea medidas de tendencia como son: la media o la moda de la región a la cual pertenece dicha observación. Los métodos basados en árboles de decisión son simples y son fáciles de interpretar, aunque suelen tener unas capacidades de ajuste y de predicción peores que otro tipo de métodos como pueden ser la regresión penalizada. Sin embargo, para mejorar esta capacidad predictiva se usan métodos de ensamble, en los que se combinan estos árboles, obteniéndose resultados de predicción mucho mejores (aunque a expensas de la interpretabilidad del modelo). Las ventajas que presenta es que son sencillos de implementar, fáciles de entender e interpretar, versátiles y permiten resolver tanto problemas de clasificación como de regresión, las desventajas serían que son sensibles ante rotaciones de los datos de la muestra de entrenamiento.

Anexos

#importacion de librerias

#especializada en el manejo y análisis de estructuras de datos

import pandas as pd

#estructura de datos de matriz

import numpy as np

from pandas.core.common import flatten

from plotnine import \*

from array import \*

import scipy.stats as stats

import math

#especializada en la creación de gráficos en dos dimensiones

import matplotlib as mpl

import matplotlib.pyplot as plt

#clases y funciones para la estimación de muchos modelos estadísticos

import statsmodels.api as sm

from sklearn import linear\_model

from sklearn.metrics import mean\_squared\_error, r2\_score

import statsmodels.formula.api

import statsmodels.stats.api as sms

#fecha 20/03/1992

m=0+3+3

d=21

#FECHA 2161992

fecha = 2161992

#dni modificado 2623584262 a array

arr\_fecha = [2,1,6,1,9,9,2]

#creacion del conjunto de datos a partir de la fecha modificada

muestra = 200+10\*arr\_fecha[0]

predictores = 10+arr\_fecha[1]+arr\_fecha[2]

informativas = 10+arr\_fecha[1]

sesgo = 2

ruido = 10\* arr\_fecha[3]

semilla=int(dni)

from sklearn.datasets import make\_regression

X\_0, Y\_0 = make\_regression (n\_samples = muestra, n\_features = predictores,

shuffle=False, random\_state = semilla, noise = ruido, bias= sesgo )

#muestra de entrenamiento

X = X\_0[0:200, :]

Y = Y\_0[0:200]

#muestra de validacion

Xval = X\_0[200:len(X\_0)+1, :]

Yval = Y\_0[200:len(Y\_0)+1]

#comprobacion particion

#longitud de la primera fila de una matriz

X\_0.shape

#devuelve tupla con el tamaño del array

X.shape

Xval.shape

Y\_0.shape

Y.shape

Yval.shape

# descripción general rápida del conjunto de datos

#librerias para funciones que hacen que matplotlib funcione como MATLAB

import matplotlib

import matplotlib.pyplot as plt

nom\_var = ["x%d" % i for i in range(1, predictores + 1)]

dt\_fX = pd.DataFrame(data= X[0:,0:], columns= nom\_var)

#funcion para la descripcion

dt\_fX.info()

#resumen de las estadísticas pertenecientes a las columnas de DataFrame.

dt\_fX.describe()

nom\_y=["y"]

#dataframe

dt\_fy= pd.DataFrame(data = Y.flatten(), columns = nom\_y)

#funcion para la descripcion

dt\_fy.info()

#resumen estadistico

dt\_fy.describe()

#Traza un histograma

dt\_fy.hist(bins=50, color="red" );

plt.xlabel("Valor Y");

plt.ylabel("n");

#mostrar figura

plt.show();

#Regresión lineal por mínimos cuadrados ordinarios

from sklearn.linear\_model import LinearRegression

#ajustar el modelo

lm\_1= LinearRegression()

lm\_1.fit(X,Y)

LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=False)

print(lm\_1.intercept\_)

print(lm\_1.coef\_)

#Cargar funciones

import statsmodels.api as sm

# intercepto

X1 = sm.add\_constant(dt\_fX)

#Comprobar df

X1.head()

#Ajustar el modelo

est\_x1= sm.OLS(Y,X1)

#Ver Ajuste

est\_1 = est\_x1.fit()

print(est\_1.summary())

X2=X1.drop("x11",axis=1)

X2.head()

est\_X2=sm.OLS(Y,X2)

est\_2=est\_X2.fit()

print(est\_2.summary())

X3=X2.drop("x12",axis=1)

X3.head()

est\_X3=sm.OLS(Y,X3)

est\_3=est\_X3.fit()

print(est\_3.summary())

cols=['x13','x14','x15','x16','x17']

X4=X3.drop(cols,axis=1)

X4.head()

#Ajuste

est\_X4=sm.OLS(Y,X3)

est\_4=est\_X4.fit()

#resumen

print(est\_4.summary())

from sklearn.linear\_model import LinearRegression

#ajustar el modelo

X\_lm=X4.drop("const",axis=1)

#instancia de un modelo de regresión lineal

lm\_4= LinearRegression()

#ajústa los datos con la función fit ()

lm\_4.fit(X\_lm,Y)

LinearRegression()

#Imprime la intersección:

print(lm\_4.intercept\_)

#Coeficiente de regresión

print(lm\_4.coef\_)

#Red Elástica

np.random.seed(int(semilla))

from sklearn.linear\_model import ElasticNetCV

#rejilla de valores entre 0.1 y 1 con separación de 0.1

valores\_r=np.linspace(0.1,1,10)

print(valores\_r)

#Valor de r

e\_net=ElasticNetCV(cv=10,l1\_ratio=valores\_r)

e\_net.fit(X,Y)

print(e\_net.l1\_ratio\_)

#valor de intercepto

print(e\_net.intercept\_)

#Coeficiente de regresión de red elástica

print(e\_net.coef\_)

#Validación

#funcion para calculo

def error\_cal(Xval,Yval,model):

#prediccion

pred=model.predict(Xval)

#Pérdida de regresión del error cuadrático medio.

mse=mean\_squared\_error(Yval,pred)

#raíz cuadrada positiva

rmse=np.sqrt(mse)

return rmse

# Regresión lineal múltiple

error\_cal(Xval,Yval,model=lm\_1)

#Red elástica

error\_cal(Xval,Yval,model=e\_net)

#Ridge

from sklearn.linear\_model import Ridge

ridgeR = Ridge(alpha = 1)

ridgeR.fit(Xval, Yval)

y\_pred = ridgeR.predict(Xval)

mean\_squared\_error\_ridge = np.mean((y\_pred - Yval)\*\*2)

print(mean\_squared\_error\_ridge)

#Lasso

from sklearn.linear\_model import Lasso

lasso = Lasso(alpha = 1)

lasso.fit(Xval, Yval)

y\_pred1 = lasso.predict(Xval)

mean\_squared\_error = np.mean((y\_pred1 - Yval)\*\*2)

print(mean\_squared\_error)