latestDataAnalisi

December 6, 2019

1 Cosa andremo a fare oggi?

- Introduzione alla libreria scikit-learn
- Regressione vs Classificazione
- Regressione Lineare
  - Regressione Lineare: Esempio 1: Stima del salario mensile di un individuo basandoci solo sull'etá della persona.
  - Regressione Lineare: Esempio 2: Stima del salario mensile di un individuo prima basandoci sull'etá, sul numero di ore di lavoro mensili e sull'indice (1-10) di esperienza della persona.
  - Regressione Lineare Esempio 3: Utilizzeremo la regressione lineare per approssimare delle curve nello spazio 2D.
- Scikit-Learn datasets Spiegazione, nozioni su come scaricarli applicazione della regressione lineare a un dataset di scikit-learn (diabetes dataset)

2 Introduzione alla libreria scikit-learn

Scikit-learn è una libreria di machine learning. * Sklearn Installation page

```
# Pip packet manager
pip install -U scikit-learn
# Conda packet manager
conda install scikit-learn
```

3 Regressione vs Classificazione

Appartengono entrambi alla cateria di "Supervised Learning" (Apprendimento Supervisionato). Con ciò intendiamo quella situazione in cui io mi creo un dataset (quindi racconto una serie di dati) e con questi dati cerco di risolvere un problema. Posso risolvere due tipi di problema:

3.1 Regressione: Spiegazione

Qualè il valore di?

Quando parliamo di regressione vogliamo stimare, predirre un NUMERO usando dei dati precedenemente raccolti. Quello che vogliamo predirre dipende dagli input che abbiamo.

3.1.1 Esempi Regressione:

- Dato il salario mensile delle persone che lavorano in Google e la loro età vogliamo capire (stimare, predirre) quanto guadagnerà marco che ha 20 anni se venisse assunto domani.
- Dati gli ultimi match dell'inter e la qualità degli ultimi acquisti vogliamo stimare il numero di punti che esso otterrà a fine campionato.
- Vogliamo approssimare una curva nello spazio. Cioè data una curva nella spazio 2D vogliamo trovare una funzione matematica capace di approssimarla.
- Vogliamo stimare il numero di anni che un pc appena comprato durerà, analizzando quando normamente dura.
- Vogliamo stimare la qualità di questo corso (da 1 a 10) sulla base delle vostre valutazioni e del numero di ore del corso.
- Vogliamo predirre il numero di studenti che passerà l'esame analizzando il vostro background (1-10), il numero di domande fatte e il numero di esercitazioni che avete risolto correttamente.

3.2 Classificazione: Spiegazione

Chi tra?

Parliamo invece di classificazione quando volgiamo DISTINGUERE (cioè capire se un qualcosa è a o b). Non ci interessa in questa caso avere una percentuale o un indicazione numerica vogliamo differenziare qualcosa.

3.2.1 Esempi Classificazione:

- Quale squadra di calcio vincerà il campionato? Nel campionato italiano ci sono 20 squadre, tra queste 20 voglio indovinare quale vincerà.
- Dati un insieme di data relativi a un paziente vogliamo capire se ha un raffreddore o semplicemente la febbre. (O uno o l'altro)
- Sulla base delle specifiche vogliamo campire se è più adatto come gaming laptop il mac pro, il dex xps o l'msi. Chi tra questi tre pc è più adatto?
- Sulla base del numero di persone assunte, più altre informazione vogliamo capire se è meglio per uno studente incerto inscriversi a giurisprudenza, oppure a ingegneria.
- Basandoci sulle citazioni più famose dei filosi vogliamo capire chi ha è più conosciuto tra Risultati relativi a Immanuel Kant
- Raccogliendo dati di 100 persone che hanno seguito le seguenti diete (Dieta dissociata, Dieta Mediterranea, Cronodieta, Dieta del biscotto) vogliamo capire quale tra queste è più adatta a me.

4 Regressione Lineare

Significato:

- dato una serie di dati misurati vogliamo trovare una relazione matematica tra i dati misurati e l'output desirato. Vogliamo predirre qualcosa.

4.1 Regressione Lineare: Esempio 1: Stima del salario mensile di un individuo basandoci solo sull'etá della persona.

4.1.1 Spiegazione del Problema

Vogliamo predirre il salario di un lavoratore mensile in euro basandoci sulla sua età.

Definiamo il problema matematicamente (caso lineare):
$Y = w0 * 1 + w1 * X1$

- Y: salario che voglio predire
- X1: età del lavoratore esempio X1=25 anni
- w0 e w1 sono i due parametri da stimare (Sconosciuti)

Il problema di regressione consiste nel determinare le variabili sconosciute (a, b) utilizzando un dataset (insieme dei dati raccolti). Una volta indiviudati i parametri a e b saremo in grado di stimare quanto guandagna una persona a noi sconosciuta semplicemente basandoci sulla sue età.

- Step 1) Dato un insieme di dati raccolti (Y e X1) vogliamo stimare i parametri sconosciuti w0 e w1 (sono due numeri)

```
Dati Raccolti:
Y = Salario al mese in euro
X1 = Età del lavoratore
1) Marco:    Y=1100   X1=19
2) Daniele:  Y=1150   X1=21
3) Davide:   Y=1155   X1=22
4) Marta:    Y=1170   X1=23
6) Alessia:  Y=1200   X1=26
9) Stella:   Y=1750   X1=33
10) Chiara   Y=1640   X1=29
```

Extra Tip: É interesante notare che stiamo affrontando un problema di regressione lineare non é necessario applicare "feature scaling" cioé scalare gli inputs nel range 0 1. Stiamo cercando di trovare un "affine map" che é una composizione lineare. É piú comune scalare i dati quando affrontiamo un problema di classificazione.

- Step 2) Vogliamo rispondere alla seguente domanda:
  Quanto guadagnerá Giada se ha 25 anni?

Vediamo chiaramente che all'aumentare dell'età aumenta il guadagno e ad occhio possiamo dire che Giada guadagnerà intorno ai 1180 euro. Matematicamente ci stiamo chiedendo il valore di Y (salario di giada) conoscendo la sua età (X1=25 anni) e i due parametri w0 e w1 (ottenuti nello step 1)

Andremo a risolvere il sopra citato problema usando scikit-learn library. In particolare ci focalizzero sulla Regressione lineare.

4.1.2 Scikit-Learn Regressione Lineare

Dobbiamo capire come scrivere il seguente problema usando la libreria scikit-learn.

- Linear Model Regression Descrizione
- Scikit-Learn Linear Regression Documentazione

$Y = w0 * 1 + w1 * X1 + w2 * X2 + ...wN * XN$

Dove X0 ... XN rappresentano gli input, nel nostro caso abbiamo solo letá ma avremmo potuto avere anche altri valore come il numero di ore di lavoro etc .. Quindi nel nostro caso avremo X1,X2 ...XN =0 e X1 (sarebbe letá). Quindi la sopra descritta equazione diventa

$Y = w0 * 1 + w1 * X1$

4.1.3 Creiamo il dataset usando numpy

Cominciamo importando entrable le librerie scikit-learn e numpy. Sucessivamente andiamo a scrivere il dataset creato usando numpy.

[2]:
```
#--------------- Importiamo le librerie ---------------------
from sklearn import linear_model
import numpy as np
# ------------------------------------------------------------

#------------- Creiamo il dataset usando numpy --------------
'''
Y = Salario al mese in euro
X1 = Età del lavoratore
Equazione = Y = b + a*X --> Y = w0 + w1*X1
1) Marco:    Y=1100   X1=19  Y=w0*1+w1*X1 --> 1100 = w0*1 + w1*19
2) Daniele:  Y=1150   X1=21  Y=w0*1+w1*X1 --> 1150 = w0*1 + w1*21
3) Davide:   Y=1155   X1=22  Y=w0*1+w1*X1 --> 1155 = w0*1 + w1*22
4) Marta:    Y=1170   X1=23  Y=w0*1+w1*X1 --> 1170 = w0*1 + w1*23
6) Alessia:  Y=1200   X1=26  Y=w0*1+w1*X1 --> 1200 = w0*1 + w1*26
9) Stella:   Y=1750   X1=33  Y=w0*1+w1*X1 --> 1750 = w0*1 + w1*33
10) Chiara   Y=1640   X1=29  Y=w0*1+w1*X1 --> 1640 = w0*1 + w1*29
'''
X= np.array([[1,19],[1,21], [1,22], [1,23], [1,26], [1,33],[1,29]])
Y= np.array([[1100],[1150],[1155],[1170],[1200],[1750],[1640]])
#Y= np.array([1100,1150,1155,1170,1200,1750,1640])
print("X input data: \n",X, X.shape)
print("Y output data: \n",Y, Y.shape)
#------------------------------------------------------------
```

```
X input data:
 [[ 1 19]
 [ 1 21]
 [ 1 22]
 [ 1 23]
 [ 1 26]
 [ 1 33]
 [ 1 29]] (7, 2)
Y output data:
```

```
[[1100]
[1150]
[1155]
[1170]
[1200]
[1750]
[1640]] (7, 1)
```

4.1.4 Implementazione Regressione Lineare scikit-learn

[3]:
```
# fit_intercept = False significa che dovremmo manualmente inseririe 1 se␣
 ↪vogliamo estrarre il parametro b
reg = linear_model.LinearRegression(fit_intercept=False, normalize=False)
reg.fit(X, Y)

print('w0 and w1 ', reg.coef_)
w0 = reg.coef_[0][0]
w1 = reg.coef_[0][1]

# Quanto guadagnerá Giada se ha 25 anni?
X_test = np.array([[1,25]]) # creiamo l'input vector
Y_pred = reg.predict(X_test) # prediciamo il guandagno
Y_pred_ = w0*X_test[0][0] + w1*X_test[0][1] # Y_pred_ = w0*1 + w1*25
print('Giada gudagnerà al mese: ', Y_pred, Y_pred_)
```

```
w0 and w1  [[56.21316306 50.70235756]]
Giada gudagnerà al mese:  [[1323.77210216]] 1323.7721021610998
```

4.1.5 Implementazione Regressione Lineare scikit-learn Classi

[4]:
```
from sklearn import linear_model
import numpy as np
# Implementazione usando fit_intercept=False , piú vicina alla logica␣
 ↪matematica
class Dataset:
    def create(self):
      X= np.array([[1,19],[1,21], [1,22], [1,23], [1,26], [1,33],[1,29]])
      Y= np.array([[1100],[1150],[1155],[1170],[1200],[1750],[1640]])
      return X,Y

class LinearRegressor:
    def __init__(self):
      # Inizializzazione
      self.reg = linear_model.LinearRegression(fit_intercept=False,␣
 ↪normalize=False)
    def train(self,X,Y):
      # Estimate w0, w1 .. wN
```

```
      self.reg.fit(X,Y)
      print(self.reg.intercept_) # Questo deve essere 0 se fit_intercept=false
      return self.reg.coef_
    def predict(self,X_test,coef_):
      Y_pred = self.reg.predict(X_test)
      #Y_pred = np.dot(coef_, X_test.T)
      return Y_pred

myDataset=Dataset()
X,Y = myDataset.create()
myLinearRegressor = LinearRegressor()
coef_ = myLinearRegressor.train(X,Y)
print(coef_)
# Set input Giada 25 anni Y=w0*1 + w1*X1 --> 1 X1=25
X_test = np.array([[1,25]])
Y_pred = myLinearRegressor.predict(X_test,coef_)
print(Y_pred)
```

```
0.0
[[56.21316306 50.70235756]]
[[1323.77210216]]
```

Nella sezione EXTRA sono presenti: * Implementazione usando fit_intercept=True. Questa implementazione permette di semplicare la scrittura degli input in quanto permette di omettere l'1 * Implementazione usando la libreria numpy del Regressore lineare (Least Square-Metodo dei Minimi quadrati). Implementazione piú matematica.

Come doveremmo modificare la creazione dei dati se oltre all'etá della persona avessimo piú informazioni?

Per esempio se conoscessimo anche:

- il numero di ore mensili di lavoro
- indice di esperienza da 1 a 10

4.2 Regressione Lineare: Esempio 2: Stima del salario mensile di un individuo prima basandoci sull'etá, sul numero di ore di lavoro mensili e sull'indice (1-10) di esperienza della persona.

4.2.1 Spiegazione del Problema

Il dataset diventerebbe:

```
Y = Salario al mese in euro
X1 = Età del lavoratore
X2 = Numero di ore mensili di lavoro
X3 = Indice di esperienza da 1 a 10
1) Marco:    Y=1100   X1=19   X2=150  X3=6
2) Daniele:  Y=1150   X1=21   X2=135  X3=8
3) Davide:   Y=1155   X1=22   X2=160  X3=5
```

```
4) Marta:    Y=1170   X1=23   X2=158  X3=7
6) Alessia:  Y=1200   X1=26   X2=155  X3=7
9) Stella:   Y=1750   X1=33   X2=120  X3=10
10) Chiara   Y=1640   X1=29   X2=130  X3=9
```

L' equazione della regressione lineare diventa:
$Y = w0 * 1 + w1 * X1 + w2 * X2 + w3 * X3$

L'obbiettivo é stimare i parametri w0,w1,w2,w3 partendo da un dataset (X1,X2,X3,Y) come il sopra elencato.

Quanto guadagnerá Giada se ha 25 anni, lavora 130 ore al mese ed ha un indice di esperienza uguale a 8?

4.2.2 Creiamo il dataset usando numpy

[5]:
```
#--------------- Importiamo le librerie ---------------------
from sklearn import linear_model
import numpy as np
# ------------------------------------------------------------

#------------- Creiamo il dataset usando numpy --------------
'''
Y = Salario al mese in euro
X1 = Età del lavoratore
X2 = Numero di ore mensili di lavoro
X3 = Indice di esperienza da 1 a 10
Equazione = Y = w0 + w1*X1 + w2*X2 + w3*X3
1) Marco:    Y=1100   X1=19   X2=150   X3=6    Y= w0*1 + w1*X1 + w2*X2 + w3*X3␣
 ↪--> 1100 = w0*1 + w1*19 + w2*150 + w3*6
2) Daniele:  Y=1150   X1=21   X2=135   X3=8    Y= w0*1 + w1*X1 + w2*X2 + w3*X3␣
 ↪--> 1100 = w0*1 + w1*21 + w2*135 + w3*8
3) Davide:   Y=1155   X1=22   X2=160   X3=5    Y= w0*1 + w1*X1 + w2*X2 + w3*X3␣
 ↪--> 1100 = w0*1 + w1*22 + w2*160 + w3*5
4) Marta:    Y=1170   X1=23   X2=158   X3=7    Y= w0*1 + w1*X1 + w2*X2 + w3*X3␣
 ↪--> 1100 = w0*1 + w1*23 + w2*158 + w3*7
6) Alessia:  Y=1200   X1=26   X2=155   X3=7    Y= w0*1 + w1*X1 + w2*X2 + w3*X3␣
 ↪--> 1100 = w0*1 + w1*26 + w2*155 + w3*7
9) Stella:   Y=1750   X1=33   X2=120   X3=10   Y= w0*1 + w1*X1 + w2*X2 + w3*X3␣
 ↪--> 1100 = w0*1 + w1*33 + w2*120 + w3*10
10) Chiara   Y=1640   X1=29   X2=130   x3=9    Y= w0*1 + w1*X1 + w2*X2 + w3*X3␣
 ↪--> 1100 = w0*1 + w1*29 + w2*130 + w3*9
'''

X= np.array([[1,19,150,6],[1,21,135,8], [1,22,160,5], [1,23,158,7],␣
 ↪[1,26,155,7], [1,33,120,10],[1,29,130,9]])
Y= np.array([[1100],[1150],[1155],[1170],[1200],[1750],[1640]])
#Y= np.array([1100,1150,1155,1170,1200,1750,1640])
print("X input data: \n",X, X.shape)
```

```
print("Y output data: \n",Y, Y.shape)
#------------------------------------------------------------
```

```
X input data:
 [[  1  19 150   6]
 [  1  21 135   8]
 [  1  22 160   5]
 [  1  23 158   7]
 [  1  26 155   7]
 [  1  33 120  10]
 [  1  29 130   9]] (7, 4)
Y output data:
 [[1100]
 [1150]
 [1155]
 [1170]
 [1200]
 [1750]
 [1640]] (7, 1)
```

4.2.3 Implementazione Regressione Lineare scikit-learn

[6]:
```
# fit_intercept = False significa che dovremmo manualmente inseririe 1 se␣
 ↪vogliamo estrarre il parametro b
reg = linear_model.LinearRegression(fit_intercept=False, normalize=False)
reg.fit(X, Y)

print('w0, w1, w2, w3 ', reg.coef_)
w0 = reg.coef_[0][0]
w1 = reg.coef_[0][1]
w2 = reg.coef_[0][2]
w3 = reg.coef_[0][3]

# Quanto guadagnerá Giada se ha X1=25 anni, lavora X2=130 ore al mese e ha un
# indice di esperienza X3=8 ?
X_test = np.array([[1,25,130,8]]) # creiamo l'input vector
Y_pred = reg.predict(X_test) # prediciamo il guandagno
Y_pred_ = w0*X_test[0][0] + w1*X_test[0][1] + w2*X_test[0][2] + w3*X_test[0][3]␣
 ↪# Y_pred_ = w0*1 + w1*25 + w2*130 + w3*8
print('Giada gudagnerà al mese: ', Y_pred, Y_pred_)
```

```
w0, w1, w2, w3  [[1989.84032486   43.44020785   -9.68682336  -48.36000551]]
Giada gudagnerà al mese:  [[1429.67844041]] 1429.6784404143273
```

4.2.4 Implementazione Regressione Lineare scikit-learn Classi

[7]:
```
from sklearn import linear_model
import numpy as np
# Implementazione usando fit_intercept=False , piú vicina alla logica␣
 ↪matematica
class Dataset:
    def create(self):
      X= np.array([[1,19,150,6],[1,21,135,8], [1,22,160,5], [1,23,158,7],␣
 ↪[1,26,155,7], [1,33,120,10],[1,29,130,9]])
      Y= np.array([[1100],[1150],[1155],[1170],[1200],[1750],[1640]])
      return X,Y

class LinearRegressor:
    def __init__(self):
      # Inizializzazione
      self.reg = linear_model.LinearRegression(fit_intercept=False,␣
 ↪normalize=False)
    def train(self,X,Y):
      # Estimate w0, w1 .. wN
      self.reg.fit(X,Y)
      print(self.reg.intercept_) # Questo deve essere 0 se fit_intercept=false
      return self.reg.coef_
    def predict(self,X_test,coef_):
      Y_pred = self.reg.predict(X_test)
      #Y_pred = np.dot(coef_, X_test.T)
      return Y_pred

myDataset=Dataset()
X,Y = myDataset.create()
myLinearRegressor = LinearRegressor()
coef_ = myLinearRegressor.train(X,Y)
print(coef_)
# Set input Giada 25 anni Y=w0*1 + w1*X1 --> 1 X1=25
X_test = np.array([[1,25,130,8]])
Y_pred = myLinearRegressor.predict(X_test,coef_)
print(Y_pred)
```

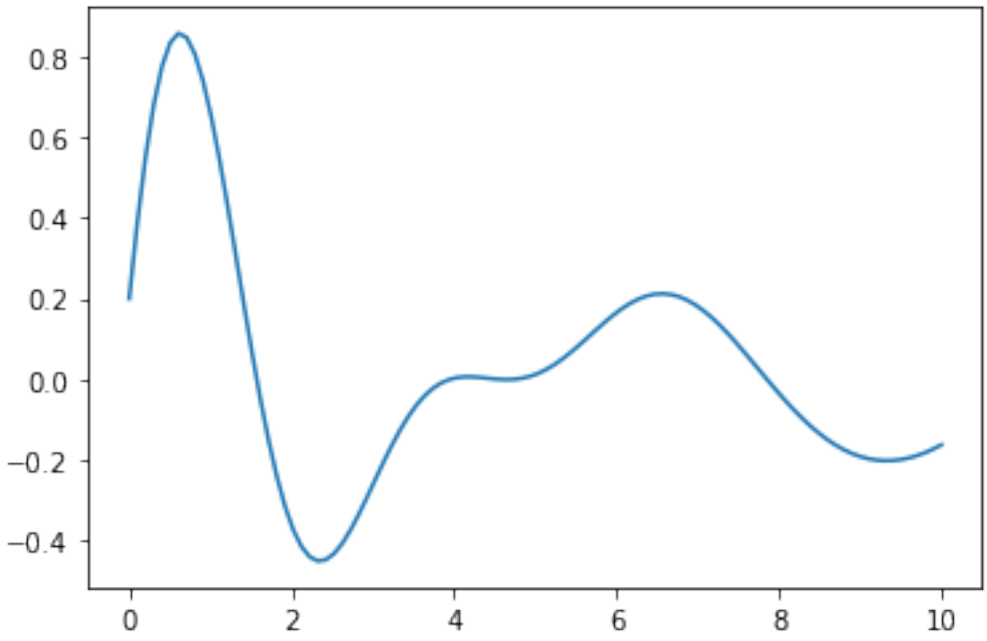
```
0.0
[[1989.84032486   43.44020785   -9.68682336  -48.36000551]]
[[1429.67844041]]
```

4.3 Regressione Lineare Esempio 3: Utilizzeremo la regressione lineare per approssimare delle curve nello spazio 2D.

4.3.1 Visualizzare una curva nello spazio 2D

[8]:
```
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np

X = np.linspace(0,10,100)
Y = np.sin(2*X)*np.exp(-0.5*X) + 0.2*np.cos(X)
plt.plot(X,Y)
```

[8]:
```
[<matplotlib.lines.Line2D at 0x7fd9471034e0>]
```



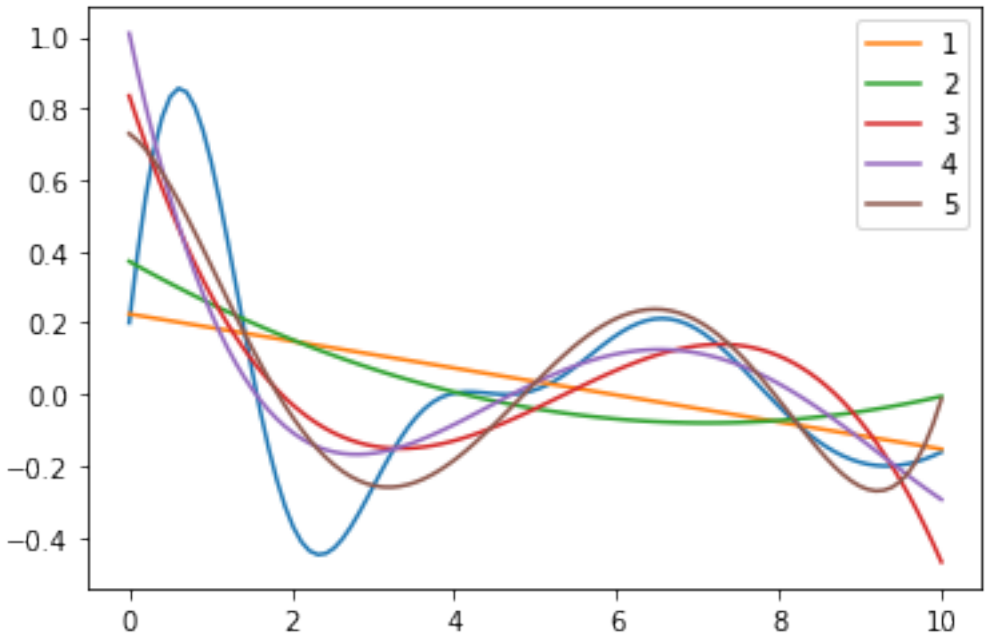
4.3.2 Regressore lineare utilizzato per stimare una curva nello spazio

[9]:
```
from sklearn import linear_model
import numpy as np

class LinearRegressor:
    def __init__(self):
      # Inizializzazione
      self.reg = linear_model.LinearRegression(fit_intercept=False,␣
 ↪normalize=False)
    def train(self,X,Y):
```

```
      # Estimate w0, w1 .. wN
      self.reg.fit(X,Y)
      return self.reg.coef_
    def predict(self,X_test,coef_):
      Y_pred = self.reg.predict(X_test)
      #Y_pred = np.dot(coef_, X_test.T)
      return Y_pred

# Creazione dataset
X1 = np.linspace(0,10,100)
X2 = np.power(X1,2)
X3 = np.power(X1,3)
X4 = np.power(X1,4)
X5 = np.power(X1,5)

X_test1 = np.vstack([np.ones(len(X1)), X1]).T
X_test2 = np.vstack([np.ones(len(X1)), X1, X2]).T
X_test3 = np.vstack([np.ones(len(X1)), X1, X2, X3]).T
X_test4 = np.vstack([np.ones(len(X1)), X1, X2, X3, X4]).T
X_test5 = np.vstack([np.ones(len(X1)), X1, X2, X3, X4, X5]).T

Y = np.sin(2*X1)*np.exp(-0.5*X1) + 0.2*np.cos(X1)

myLinearRegressor1 = LinearRegressor()
myLinearRegressor2 = LinearRegressor()
myLinearRegressor3 = LinearRegressor()
myLinearRegressor4 = LinearRegressor()
myLinearRegressor5 = LinearRegressor()

coef_1 = myLinearRegressor1.train(X_test1,Y)
coef_2 = myLinearRegressor2.train(X_test2,Y)
coef_3 = myLinearRegressor3.train(X_test3,Y)
coef_4 = myLinearRegressor4.train(X_test4,Y)
coef_5 = myLinearRegressor5.train(X_test5,Y)

Y_pred1 = myLinearRegressor1.predict(X_test1,coef_1)
Y_pred2 = myLinearRegressor2.predict(X_test2,coef_2)
Y_pred3 = myLinearRegressor3.predict(X_test3,coef_3)
Y_pred4 = myLinearRegressor4.predict(X_test4,coef_4)
Y_pred5 = myLinearRegressor5.predict(X_test5,coef_5)

#print(Y_pred)
plt.plot(X1,Y)
plt.plot(X1,Y_pred1, label='1')
plt.plot(X1,Y_pred2, label='2')
plt.plot(X1,Y_pred3, label='3')
plt.plot(X1,Y_pred4, label='4')
```

```
plt.plot(X1,Y_pred5, label='5')
plt.legend()
```

[9]: `<matplotlib.legend.Legend at 0x7fd946bf0710>`



5 Scikit-Learn datasets Spiegazione, nozioni su come scaricarli applicazione della regressione lineare a un dataset di scikit-learn (diabetes dataset)

Andremo a vedere quali dataset sono disponibili in scikit-learn, come scaricarli e capirne il contenuto.

- Sklearn dataset page

I dataset disponibili sono i seguenti: * Iris plant dataset * Optical recognition of handwritten digits dataset * Boston houses price dataset * Diabetes dataset * Wine Recognition dataset * Breast cancer wisconsin (diagnostic) dataset * Linnerrud Dataset

5.1 Importare i dataset da scikit-learn

[10]:
```
# Importare i datasets
from sklearn import datasets

iris = datasets.load_iris() # Load iris dataset
digits = datasets.load_digits() # Load digits dataset
```

```
boston = datasets.load_boston() # Load boston dataset
diabetes = datasets.load_diabetes() # Load diabetes dataset
linnerud = datasets.load_linnerud() # Load linnerud dataset
wine = datasets.load_wine() # Load wine dataset
breast_cancer = datasets.load_breast_cancer() # Load breast_cancer dataset

dataset_scelto = diabetes

# Check the dataset diabetes
#print(dataset_scelto)
parametri = dataset_scelto.keys()
valore = dataset_scelto.values()
print(parametri)
```

```
dict_keys(['data', 'target', 'DESCR', 'feature_names', 'data_filename',
'target_filename'])
```

[11]:
```
# Print useful information
for name in parametri:
  print("----------------------------------------")
  print(name , dataset_scelto[name])
  print("----------------------------------------")
```

```
----------------------------------------
data [[ 0.03807591  0.05068012  0.06169621 ... -0.00259226  0.01990842
  -0.01764613]
 [-0.00188202 -0.04464164 -0.05147406 ... -0.03949338 -0.06832974
  -0.09220405]
 [ 0.08529891  0.05068012  0.04445121 ... -0.00259226  0.00286377
  -0.02593034]
 ...
 [ 0.04170844  0.05068012 -0.01590626 ... -0.01107952 -0.04687948
   0.01549073]
 [-0.04547248 -0.04464164  0.03906215 ...  0.02655962  0.04452837
  -0.02593034]
 [-0.04547248 -0.04464164 -0.0730303  ... -0.03949338 -0.00421986
   0.00306441]]
----------------------------------------
----------------------------------------
target [151.  75. 141. 206. 135.  97. 138.  63. 110. 310. 101.  69. 179. 185.
 118. 171. 166. 144.  97. 168.  68.  49.  68. 245. 184. 202. 137.  85.
 131. 283. 129.  59. 341.  87.  65. 102. 265. 276. 252.  90. 100.  55.
  61.  92. 259.  53. 190. 142.  75. 142. 155. 225.  59. 104. 182. 128.
  52.  37. 170. 170.  61. 144.  52. 128.  71. 163. 150.  97. 160. 178.
  48. 270. 202. 111.  85.  42. 170. 200. 252. 113. 143.  51.  52. 210.
  65. 141.  55. 134.  42. 111.  98. 164.  48.  96.  90. 162. 150. 279.
  92.  83. 128. 102. 302. 198.  95.  53. 134. 144. 232.  81. 104.  59.
```

```
 246. 297. 258. 229. 275. 281. 179. 200. 200. 173. 180.  84. 121. 161.
  99. 109. 115. 268. 274. 158. 107.  83. 103. 272.  85. 280. 336. 281.
 118. 317. 235.  60. 174. 259. 178. 128.  96. 126. 288.  88. 292.  71.
 197. 186.  25.  84.  96. 195.  53. 217. 172. 131. 214.  59.  70. 220.
 268. 152.  47.  74. 295. 101. 151. 127. 237. 225.  81. 151. 107.  64.
 138. 185. 265. 101. 137. 143. 141.  79. 292. 178.  91. 116.  86. 122.
  72. 129. 142.  90. 158.  39. 196. 222. 277.  99. 196. 202. 155.  77.
 191.  70.  73.  49.  65. 263. 248. 296. 214. 185.  78.  93. 252. 150.
  77. 208.  77. 108. 160.  53. 220. 154. 259.  90. 246. 124.  67.  72.
 257. 262. 275. 177.  71.  47. 187. 125.  78.  51. 258. 215. 303. 243.
  91. 150. 310. 153. 346.  63.  89.  50.  39. 103. 308. 116. 145.  74.
  45. 115. 264.  87. 202. 127. 182. 241.  66.  94. 283.  64. 102. 200.
 265.  94. 230. 181. 156. 233.  60. 219.  80.  68. 332. 248.  84. 200.
  55.  85.  89.  31. 129.  83. 275.  65. 198. 236. 253. 124.  44. 172.
 114. 142. 109. 180. 144. 163. 147.  97. 220. 190. 109. 191. 122. 230.
 242. 248. 249. 192. 131. 237.  78. 135. 244. 199. 270. 164.  72.  96.
 306.  91. 214.  95. 216. 263. 178. 113. 200. 139. 139.  88. 148.  88.
 243.  71.  77. 109. 272.  60.  54. 221.  90. 311. 281. 182. 321.  58.
 262. 206. 233. 242. 123. 167.  63. 197.  71. 168. 140. 217. 121. 235.
 245.  40.  52. 104. 132.  88.  69. 219.  72. 201. 110.  51. 277.  63.
 118.  69. 273. 258.  43. 198. 242. 232. 175.  93. 168. 275. 293. 281.
  72. 140. 189. 181. 209. 136. 261. 113. 131. 174. 257.  55.  84.  42.
 146. 212. 233.  91. 111. 152. 120.  67. 310.  94. 183.  66. 173.  72.
  49.  64.  48. 178. 104. 132. 220.  57.]
----------------------------------------
----------------------------------------
DESCR .. _diabetes_dataset:

Diabetes dataset
----------------

Ten baseline variables, age, sex, body mass index, average blood
pressure, and six blood serum measurements were obtained for each of n =
442 diabetes patients, as well as the response of interest, a
quantitative measure of disease progression one year after baseline.

**Data Set Characteristics:**

  :Number of Instances: 442

  :Number of Attributes: First 10 columns are numeric predictive values

  :Target: Column 11 is a quantitative measure of disease progression one year
after baseline

  :Attribute Information:
      - Age
      - Sex
```

```
        - Body mass index
        - Average blood pressure
        - S1
        - S2
        - S3
        - S4
        - S5
        - S6

    Note: Each of these 10 feature variables have been mean centered and scaled by
    the standard deviation times `n_samples` (i.e. the sum of squares of each column
    totals 1).

    Source URL:
    https://www4.stat.ncsu.edu/~boos/var.select/diabetes.html

    For more information see:
    Bradley Efron, Trevor Hastie, Iain Johnstone and Robert Tibshirani (2004) "Least
    Angle Regression," Annals of Statistics (with discussion), 407-499.
    (https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf)
    ----------------------------------------
    ----------------------------------------
    feature_names ['age', 'sex', 'bmi', 'bp', 's1', 's2', 's3', 's4', 's5', 's6']
    ----------------------------------------
    ----------------------------------------
    data_filename /usr/local/lib/python3.6/dist-
    packages/sklearn/datasets/data/diabetes_data.csv.gz
    ----------------------------------------
    ----------------------------------------
    target_filename /usr/local/lib/python3.6/dist-
    packages/sklearn/datasets/data/diabetes_target.csv.gz
    ----------------------------------------
```

[12]:
```
# Get Data
data = dataset_scelto['data'] # or  data = iris.get('data')
print(type(data), data.shape)
# Get Target
target = dataset_scelto['target']
print(type(target), target.shape)

# Print data
#print(data)
#print(target)
```

```
<class 'numpy.ndarray'> (442, 10)
<class 'numpy.ndarray'> (442,)
```

5.2 Visualizzare il contenuto del dataset usando pandas

[13]:
```
import pandas as pd
# visualize machine laerning data https://machinelearningmastery.com/
 ↪visualize-machine-learning-data-python-pandas/
# might be cool: https://towardsdatascience.com/
 ↪data-visualization-for-machine-learning-and-data-science-a45178970be7

# Create csv file
df = pd.DataFrame(data=dataset_scelto['data'], columns =␣
 ↪dataset_scelto['feature_names'])
df.to_csv('dataset_scelto.csv', sep = ',', index = False)
df
```

[13]:

	age	sex	bmi	...	s4	s5	s6
0	0.038076	0.050680	0.061696	...	-0.002592	0.019908	-0.017646
1	-0.001882	-0.044642	-0.051474	...	-0.039493	-0.068330	-0.092204
2	0.085299	0.050680	0.044451	...	-0.002592	0.002864	-0.025930
3	-0.089063	-0.044642	-0.011595	...	0.034309	0.022692	-0.009362
4	0.005383	-0.044642	-0.036385	...	-0.002592	-0.031991	-0.046641
..	...	...	...	...	...	...	...
437	0.041708	0.050680	0.019662	...	-0.002592	0.031193	0.007207
438	-0.005515	0.050680	-0.015906	...	0.034309	-0.018118	0.044485
439	0.041708	0.050680	-0.015906	...	-0.011080	-0.046879	0.015491
440	-0.045472	-0.044642	0.039062	...	0.026560	0.044528	-0.025930
441	-0.045472	-0.044642	-0.073030	...	-0.039493	-0.004220	0.003064

[442 rows x 10 columns]

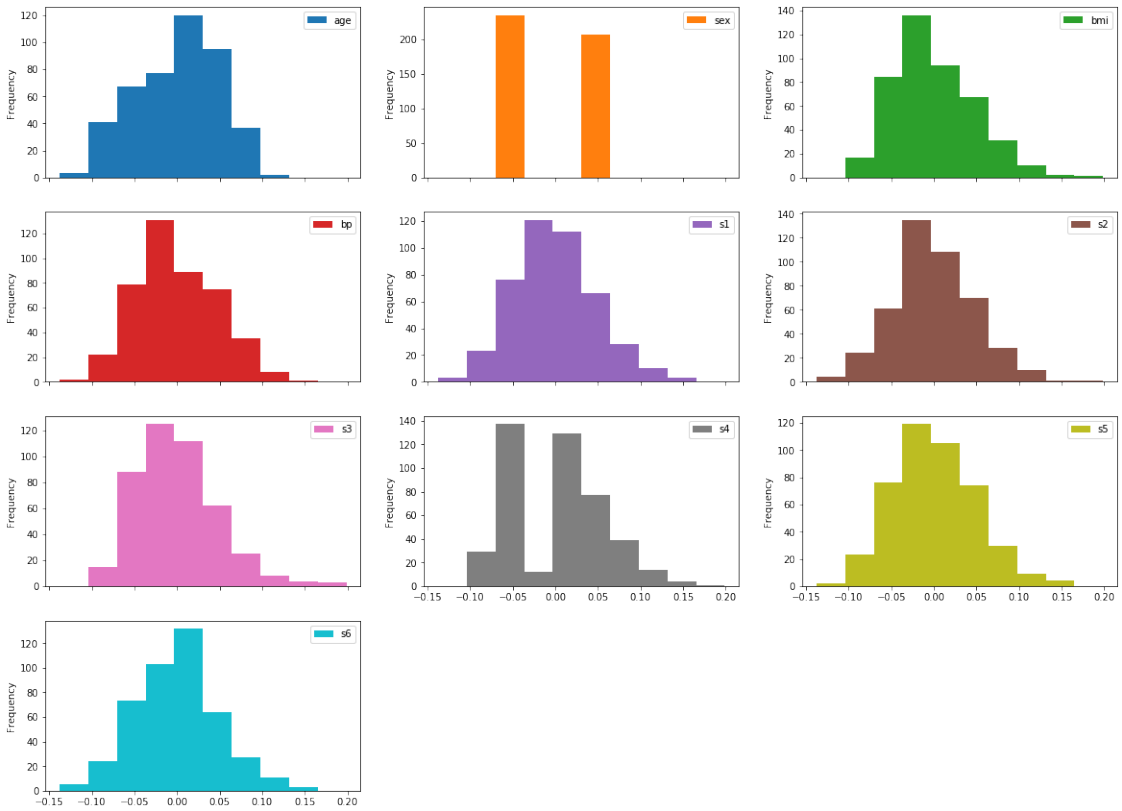
[14]:
```
df.describe()
```

[14]:

	age	sex	...	s5	s6
count	4.420000e+02	4.420000e+02	...	4.420000e+02	4.420000e+02
mean	-3.634285e-16	1.308343e-16	...	-3.830854e-16	-3.412882e-16
std	4.761905e-02	4.761905e-02	...	4.761905e-02	4.761905e-02
min	-1.072256e-01	-4.464164e-02	...	-1.260974e-01	-1.377672e-01
25%	-3.729927e-02	-4.464164e-02	...	-3.324879e-02	-3.317903e-02
50%	5.383060e-03	-4.464164e-02	...	-1.947634e-03	-1.077698e-03
75%	3.807591e-02	5.068012e-02	...	3.243323e-02	2.791705e-02
max	1.107267e-01	5.068012e-02	...	1.335990e-01	1.356118e-01

[8 rows x 10 columns]

[15]:
```
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = [20, 15]
# Plot input data
df.plot(kind='hist', subplots=True, layout=(4,3))
plt.show()
```



5.3 Applicare la regressione lineare al dataset diabetes

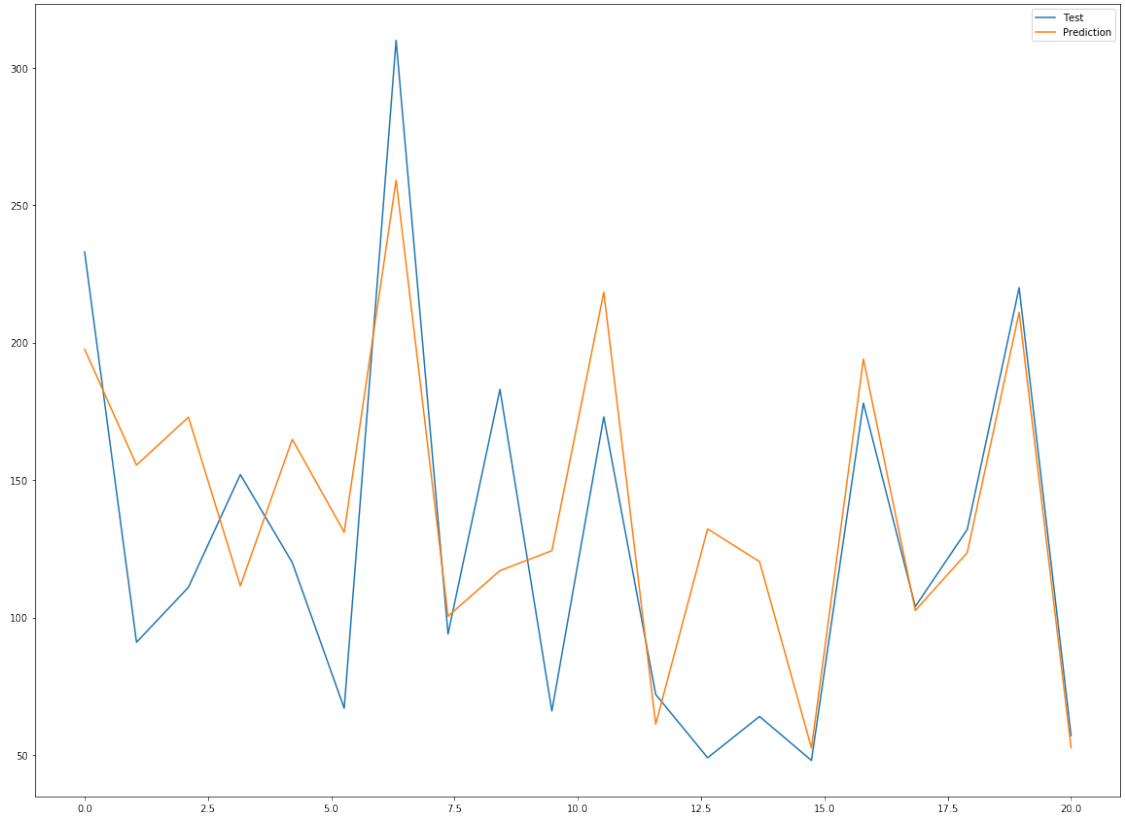
Usiamo fit_intercet=False. L'implementazione con fit_intercept = True la trovate negli Extra

[16]:
```
# Creiamo il regressore lineare
# Esempio interessante: https://scikit-learn.org/stable/auto_examples/
 ↪linear_model/plot_ols.html
# Implementazione Intercept FALSE , see Extra Advance for Intercept True
class LinearRegressor:
    def __init__(self):
      # Inizializzazione
      self.reg = linear_model.LinearRegression(fit_intercept=False,␣
 ↪normalize=False)
    def train(self,X,Y):
      # Estimate w0, w1 .. wN
      self.reg.fit(X,Y)
      return self.reg.coef_
    def predict(self,X_test,coef_):
      Y_pred = self.reg.predict(X_test)
      #Y_pred = np.dot(coef_, X_test.T)
      return Y_pred
```

```
# Creiamoil dataset
data = dataset_scelto['data']
target = dataset_scelto['target']
x_train = data[:-20]

X_train = np.vstack([np.ones(len(x_train)), x_train.T]).T
Y_train = target[:-20]
x_test = data[-20:]
X_test = np.vstack([np.ones(len(x_test)), x_test.T]).T
Y_test = target[-20:]

print(" X Train shape: ",X_train.shape)
print("Y Train shape: ", Y_train.shape)

# Creiamo Regressore Lineare
myLinearRegressor = LinearRegressor()
coef_ = myLinearRegressor.train(X_train,Y_train)
# Cosa vogliamo predirre
Y_pred = myLinearRegressor.predict(X_test, coef_)

# Plot
print(Y_test.shape)
print(Y_pred.shape)
length = Y_pred.shape[0] # 20
index_bar = np.linspace(0,length,length)
plt.plot(index_bar, Y_test, label='Test')
plt.plot(index_bar, Y_pred, label='Prediction')
plt.legend()
```

```
 X Train shape:  (422, 11)
Y Train shape:  (422,)
(20,)
(20,)
```

[16]: `<matplotlib.legend.Legend at 0x7fd945d662b0>`



6 Extra Informazioni (Avanzate)

6.1 Scikit-Learn Regressione Lineare fit_intercept=True senza classi

[17]:
```
#------------------------------------------------------------------
#--------------------   SCIKIT-LEARN REGRESSORE LINEARE  ------------
#------------------------------------------------------------------
#--------------- Implementazione usando fit_intercept=True
# Implementazione piú semplice da scrivere ma meno vicina alla logica␣
 ↪matematica

from sklearn import linear_model
import numpy as np

# Linear Model Regression Description https://scikit-learn.org/stable/modules/
 ↪linear_model.html
# Scikit Documentazione https://scikit-learn.org/stable/modules/generated/
 ↪sklearn.linear_model.LinearRegression.html

# Appraoch Linear model: Y=b+aX
```

```
# fit_intercept = True significa che il parametro b è automaticamente settato a
 ↪1 e la sua stima sarà inserita in intercept_
reg = linear_model.LinearRegression(fit_intercept=True, normalize=False)

# Y e X
x = np.array([19,21,22,23,26,33,29])
#X = np.vstack([np.ones(len(x)),x]).T
X = np.array([[19],[21], [22], [23], [26], [33],[29]])
Y = np.array([1100,1150,1155,1170,1200,1750,1640])
#print(X)
#print(Y)

reg.fit(X, Y)

# Get w0 and w1
w0 = reg.intercept_
w1 = reg.coef_

print('w0: ', reg.coef_)
print('w1: ', reg.intercept_)

X_test = np.array([[25]]) # giada ha 25 anni
Y_pred = reg.predict(X_test)
Y_pred_ = w0 + w1*X_test
print('Giada gudagnerà al mese: ', Y_pred, Y_pred_)
```

```
w0:  [50.70235756]
w1:  56.21316306483277
Giada gudagnerà al mese:  [1323.77210216] [[1323.77210216]]
```

6.2 Scikit-Learn Regressione Lineare fit_intercept=True con classi

[18]:
```
#-------------------------------------------------------------------
#---------------- CLASSE SCIKIT-LEARN REGRESSORE LINEARE ------------
#-------------------------------------------------------------------
#--------------- Implementazione usando fit_intercept=True
# Implementazione piú semplice da scrivere ma meno vicina alla logica
 ↪matematica

from sklearn import linear_model
import numpy as np

class Dataset:
    def create(self):
      X= np.array([[19],[21], [22], [23], [26], [33],[29]])
      Y= np.array([[1100],[1150],[1155],[1170],[1200],[1750],[1640]])
      return X,Y
```

```
class LinearRegressor:
    def __init__(self):
      # Inizializzazione
      self.reg = linear_model.LinearRegression(fit_intercept=True,␣
 ↪normalize=False)
    def train(self,X,Y):
      # Estimate w0, w1 .. wN
      self.reg.fit(X,Y)
      print(self.reg.intercept_) # Questo deve essere diverso da 0 se␣
 ↪fit_intercept=True
      return self.reg.coef_, self.reg.intercept_
    def predict(self,X_test,coef_, intercept_):
      Y_pred = self.reg.predict(X_test)
      #Y_pred = np.dot(coef_, X_test.T) + intercept_
      return Y_pred

myDataset=Dataset()
X,Y = myDataset.create()
myLinearRegressor = LinearRegressor()
# coef_ rappresenta w1,w2, ... wN , intercept_ rappresenta w0
coef_, intercept_ = myLinearRegressor.train(X,Y)
print(coef_, intercept_)
# Set input Giada 25 anni Y=w0*1 + w1*X1 --> 1 X1=25
X_test = np.array([[25]])
Y_pred = myLinearRegressor.predict(X_test,coef_, intercept_)
print(Y_pred)
```

```
[56.21316306]
[[50.70235756]] [56.21316306]
[[1323.77210216]]
```

6.3 Regressione Lineare implementata usando numpy (calcolo matriciale)

[19]:
```
# Implementazione regressore lineare usando solo numpy
#-----------------------------------------------------------------------
#---------------- IMPLEMENTAZIONE REGRESSORE LINEARE NUMPY -----------
#-----------------------------------------------------------------------
# Link Utile Least square with numpy: https://mmas.github.io/
 ↪least-squares-fitting-numpy-scipy

'''
[Y1]       [1 X1]
[Y2]       [1 X2]   [w0]
[Y3]   =   [1 X3]   [w1]
...        ......
[YN]       [1 XN]
```

```
'''
Y = Xbeta
Nx1 = Nx2 x 2x 1
beta = ( (XT*X)-1 )*XT*Y  (Least square)
'''
import numpy as np

# Construct Input Matrices
x = np.array([19,21,22,23,26,33,29])
X = np.vstack([np.ones(len(x)), x]).T
Y = np.vstack(np.array([1100,1150,1155,1170,1200,1750,1640]))
#print(X)
#print(Y)

# Solve Least square Matematicall (Matrix product)
# beta = ( (XT*X)-1 )*XT*Y  (Least square)
coeff = np.dot(np.linalg.inv(np.dot(X.T, X)), np.dot(X.T, Y))
print("Coefficienti: ", coeff.T)
w0 = coeff[0]
w1  = coeff[1]

# Different approach Using Numpy Functions
#coeff_2 = np.linalg.lstsq(X, Y)[0]
#print("Coefficient caso 2: ", coeff_2)
#a  = coeff_2[0]
#b = coeff_2[1]

X_test = 25
Y_pred = w0 + w1*X_test
print('Giada (25 anni) gudagnerà al mese: ', Y_pred)
```

```
Coefficienti:  [[56.21316306 50.70235756]]
Giada (25 anni) gudagnerà al mese:  [1323.77210216]
```

6.4 Come trasformare python dizionari in python array e viceversa

[20]:
```
#-----------------------------------------------------------------------
#---------------- Trasformare dizionari in Array --------- -----------
#-----------------------------------------------------------------------
# Usiamo Le funzioni
# Importare i datasets

from sklearn import datasets

iris = datasets.load_iris() # Load iris dataset
digits = datasets.load_digits() # Load digits dataset
boston = datasets.load_boston() # Load boston dataset
```

```
diabetes = datasets.load_diabetes() # Load diabetes dataset
linnerud = datasets.load_linnerud() # Load linnerud dataset
wine = datasets.load_wine() # Load wine dataset
breast_cancer = datasets.load_breast_cancer() # Load breast_cancer dataset

dataset_scelto = iris

def approccio_1(dataset_scelto):
  # ---------------------- Approccio 1
  # Get dictionar keys, value
  print(dataset_scelto.keys())
  list_keys = []
  list_values = []
  for key in dataset_scelto:
    list_keys.append(key)
    print(key)
    value = dataset_scelto[key]
    list_values.append(value)
  print("All keys inside array ", list_keys)
  #print("All values inside array ", list_values)

  # Convert list to numpy array
  print("-----------------------------------")
  array_keys = np.asarray(list_keys)
  array_values = np.array(list_values)
  print(type(array_keys), array_keys.shape, array_keys[0].shape)
  print(type(array_values), array_values.shape,array_values[0].shape)

  # Going deeper inside data shape
  print("-----------------------------------")
  for i in range(0,len(array_keys)):
    if isinstance(array_values[i],np.ndarray):
      print(array_keys[i], type(array_values[i]), array_values[i].shape )
    else:
      print(array_keys[i], type(array_values[i]))

  # Other Useful Solutions
  '''
  for value in diabetes.values():
    print(value)

  for key, value in diabetes.items():
    print(key, value)
  '''

def approccio_2(dataset_scelto):
  #---------------------- Approccio 2
```

```
  # Convert a dictionary to an array of string
  list_keys = list(dataset_scelto.keys())
  list_values = list(dataset_scelto.values())
  #print(list_keys)
  print(type(list_keys))
  #print(list_values)
  print(type(list_values))
  # Convert list as numpy narray
  array_keys = np.asarray(list_keys)
  array_values = np.array(list_values)
  print(type(array_keys))
  print(type(array_values))
  # Covert back numpy ndarray to list
  new_list_keys = array_keys.tolist()
  new_list_values = array_values.tolist()
  print(type(new_list_keys))
  print(type(new_list_values))
  # Check if the list are equal
  if list_keys == new_list_keys and list_values==new_list_values:
    print ("The lists are identical")
  else :
    print ("The lists are not identical")

print("------------------------------")
print("-------- Approach 1 ---------")
print("------------------------------")
approccio_1(dataset_scelto)
print("------------------------------")
print("-------- Approach 2 ---------")
print("------------------------------")
approccio_2(dataset_scelto)
```

```
------------------------------
-------- Approach 1 ---------
------------------------------
dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_names',
'filename'])
data
target
target_names
DESCR
feature_names
filename
All keys inside array  ['data', 'target', 'target_names', 'DESCR',
'feature_names', 'filename']
---------------------------------
<class 'numpy.ndarray'> (6,) ()
```

```
<class 'numpy.ndarray'> (6,) (150, 4)
---------------------------------
data <class 'numpy.ndarray'> (150, 4)
target <class 'numpy.ndarray'> (150,)
target_names <class 'numpy.ndarray'> (3,)
DESCR <class 'str'>
feature_names <class 'list'>
filename <class 'str'>
-----------------------------
-------- Approach 2 ---------
-----------------------------
<class 'list'>
<class 'list'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'list'>
<class 'list'>
The lists are identical
```

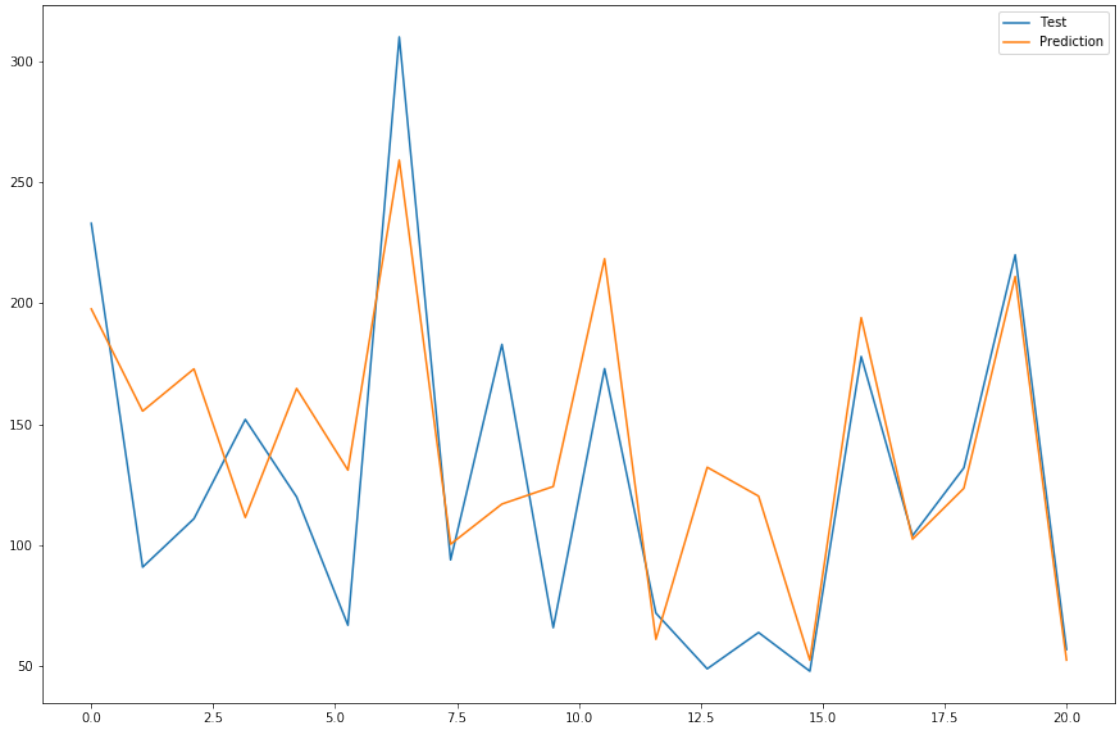
6.5 Scikit-Learn Regressione Lineare applicata al dataset Diabetes

[21]:
```
#-------------------------------------------------------------------------------------------------
#----------------SCIKIT-LEARN REGRESSORE LINEARE APPLICATA A UN DATASET (DIABETES) -------------
#-------------------------------------------------------------------------------------------------
#--------------- Implementazione usando fit_intercept=True
# Implementazione piú semplice da scrivere ma meno vicina alla logica matematica

# Importare i datasets
from sklearn import datasets
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = [15, 10]

iris = datasets.load_iris() # Load iris dataset
digits = datasets.load_digits() # Load digits dataset
boston = datasets.load_boston() # Load boston dataset
diabetes = datasets.load_diabetes() # Load diabetes dataset
linnerud = datasets.load_linnerud() # Load linnerud dataset
wine = datasets.load_wine() # Load wine dataset
breast_cancer = datasets.load_breast_cancer() # Load breast_cancer dataset

dataset_scelto = diabetes

# Check the dataset diabetes
#print(dataset_scelto)
parametri = dataset_scelto.keys()
```

```
valore = dataset_scelto.values()
print(parametri)


# Creiamo il regressore lineare
# Esempio interessante: https://scikit-learn.org/stable/auto_examples/
 ↪linear_model/plot_ols.html
class LinearRegressor:
    def __init__(self):
      # Inizializzazione
      self.reg = linear_model.LinearRegression(fit_intercept=True,␣
 ↪normalize=False)
    def train(self,X,Y):
      # Estimate w0, w1 .. wN
      self.reg.fit(X,Y)
      return self.reg.coef_, self.reg.intercept_
    def predict(self,X_test,coef_, intercept_):
      Y_pred = self.reg.predict(X_test)
      #Y_pred = np.dot(coef_, X_test.T) + intercept_
      return Y_pred

# Creiamoil dataset
data = dataset_scelto['data']
target = dataset_scelto['target']
X_train = data[:-20]
Y_train = target[:-20]
X_test = data[-20:]
Y_test = target[-20:]

print(" X Train shape: ",X_train.shape)
print("Y Train shape: ", Y_train.shape)

# Creiamo Regressore Lineare
myLinearRegressor = LinearRegressor()
coef_, intercept_ = myLinearRegressor.train(X_train,Y_train)
# Cosa vogliamo predirre
Y_pred = myLinearRegressor.predict(X_test, coef_, intercept_)

# Plot
print(Y_test.shape)
print(Y_pred.shape)
length = Y_pred.shape[0] # 20
index_bar = np.linspace(0,length,length)
plt.plot(index_bar, Y_test, label='Test')
plt.plot(index_bar, Y_pred, label='Prediction')
plt.legend()
```

```
dict_keys(['data', 'target', 'DESCR', 'feature_names', 'data_filename',
'target_filename'])
 X Train shape:  (422, 10)
Y Train shape:  (422,)
(20,)
(20,)
```

[21]: `<matplotlib.legend.Legend at 0x7fd9444f9860>`



6.6 Come generare pdf da google-colab

[22]:
```
# Install the necessary packages into the virtual machine with:
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
!pip install pypandoc
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
pandoc is already the newest version (1.19.2.4~dfsg-1build4).
pandoc set to manually installed.
The following packages were automatically installed and are no longer required:
  cuda-cufft-10-1 cuda-cufft-dev-10-1 cuda-curand-10-1 cuda-curand-dev-10-1
  cuda-cusolver-10-1 cuda-cusolver-dev-10-1 cuda-cusparse-10-1
```

```
  cuda-cusparse-dev-10-1 cuda-drivers cuda-license-10-2 cuda-npp-10-1
  cuda-npp-dev-10-1 cuda-nsight-10-1 cuda-nsight-compute-10-1
  cuda-nsight-systems-10-1 cuda-nvgraph-10-1 cuda-nvgraph-dev-10-1
  cuda-nvjpeg-10-1 cuda-nvjpeg-dev-10-1 cuda-nvrtc-10-1 cuda-nvrtc-dev-10-1
  cuda-nvvp-10-1 default-jre dkms freeglut3 freeglut3-dev
  keyboard-configuration libargon2-0 libcap2 libcryptsetup12 libcublas10
  libdevmapper1.02.1 libfontenc1 libgtk2.0-0 libgtk2.0-common libip4tc0
  libjansson4 libnvidia-cfg1-440 libnvidia-common-430 libnvidia-common-440
  libnvidia-decode-440 libnvidia-encode-440 libnvidia-fbc1-440
  libnvidia-gl-440 libnvidia-ifr1-440 libpam-systemd libpolkit-agent-1-0
  libpolkit-backend-1-0 libpolkit-gobject-1-0 libxfont2 libxi-dev libxkbfile1
  libxmu-dev libxmu-headers libxnvctrl0 nsight-compute-2019.5.0
  nsight-systems-2019.5.2 nvidia-compute-utils-440 nvidia-dkms-440
  nvidia-driver-440 nvidia-kernel-common-440 nvidia-kernel-source-440
  nvidia-modprobe nvidia-settings nvidia-utils-440 openjdk-11-jre policykit-1
  policykit-1-gnome python3-xkit screen-resolution-extra systemd systemd-sysv
  udev x11-xkb-utils xserver-common xserver-xorg-core-hwe-18.04
  xserver-xorg-video-nvidia-440
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-texgyre
  libcupsfilters1 libcupsimage2 libgs9 libgs9-common libijs-0.35 libjbig2dec0
  libkpathsea6 libpotrace0 libptexenc1 libruby2.5 libsynctex1 libtexlua52
  libtexluajit2 libzzip-0-13 lmodern poppler-data preview-latex-style rake
  ruby ruby-did-you-mean ruby-minitest ruby-net-telnet ruby-power-assert
  ruby-test-unit ruby2.5 rubygems-integration t1utils tex-common tex-gyre
  texlive-base texlive-binaries texlive-fonts-recommended texlive-latex-base
  texlive-latex-recommended texlive-pictures texlive-plain-generic tipa
Suggested packages:
  fonts-noto poppler-utils ghostscript fonts-japanese-mincho
  | fonts-ipafont-mincho fonts-japanese-gothic | fonts-ipafont-gothic
  fonts-arphic-ukai fonts-arphic-uming fonts-nanum ri ruby-dev bundler
  debhelper gv | postscript-viewer perl-tk xpdf-reader | pdf-viewer
  texlive-fonts-recommended-doc texlive-latex-base-doc python-pygments
  icc-profiles libfile-which-perl libspreadsheet-parseexcel-perl
  texlive-latex-extra-doc texlive-latex-recommended-doc texlive-pstricks
  dot2tex prerex ruby-tcltk | libtcltk-ruby texlive-pictures-doc vprerex
The following NEW packages will be installed:
  fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-texgyre
  libcupsfilters1 libcupsimage2 libgs9 libgs9-common libijs-0.35 libjbig2dec0
  libkpathsea6 libpotrace0 libptexenc1 libruby2.5 libsynctex1 libtexlua52
  libtexluajit2 libzzip-0-13 lmodern poppler-data preview-latex-style rake
  ruby ruby-did-you-mean ruby-minitest ruby-net-telnet ruby-power-assert
  ruby-test-unit ruby2.5 rubygems-integration t1utils tex-common tex-gyre
  texlive texlive-base texlive-binaries texlive-fonts-recommended
  texlive-latex-base texlive-latex-extra texlive-latex-recommended
  texlive-pictures texlive-plain-generic texlive-xetex tipa
0 upgraded, 45 newly installed, 0 to remove and 5 not upgraded.
```

```
Need to get 146 MB of archives.
After this operation, 459 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-droid-fallback
all 1:6.0.1r16-1.1 [1,805 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-lato all 2.0-2
[2,698 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic/main amd64 poppler-data all
0.4.8-2 [1,479 kB]
Get:4 http://archive.ubuntu.com/ubuntu bionic/main amd64 tex-common all 6.09
[33.0 kB]
Get:5 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-lmodern all
2.004.5-3 [4,551 kB]
Get:6 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-noto-mono all
20171026-2 [75.5 kB]
Get:7 http://archive.ubuntu.com/ubuntu bionic/universe amd64 fonts-texgyre all
20160520-1 [8,761 kB]
Get:8 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcupsfilters1
amd64 1.20.2-0ubuntu3.1 [108 kB]
Get:9 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcupsimage2
amd64 2.2.7-1ubuntu2.7 [18.6 kB]
Get:10 http://archive.ubuntu.com/ubuntu bionic/main amd64 libijs-0.35 amd64
0.35-13 [15.5 kB]
Get:11 http://archive.ubuntu.com/ubuntu bionic/main amd64 libjbig2dec0 amd64
0.13-6 [55.9 kB]
Get:12 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libgs9-common
all 9.26~dfsg+0-0ubuntu0.18.04.12 [5,092 kB]
Get:13 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libgs9 amd64
9.26~dfsg+0-0ubuntu0.18.04.12 [2,264 kB]
Get:14 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libkpathsea6
amd64 2017.20170613.44572-8ubuntu0.1 [54.9 kB]
Get:15 http://archive.ubuntu.com/ubuntu bionic/main amd64 libpotrace0 amd64
1.14-2 [17.4 kB]
Get:16 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libptexenc1
amd64 2017.20170613.44572-8ubuntu0.1 [34.5 kB]
Get:17 http://archive.ubuntu.com/ubuntu bionic/main amd64 rubygems-integration
all 1.11 [4,994 B]
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 ruby2.5 amd64
2.5.1-1ubuntu1.6 [48.6 kB]
Get:19 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby amd64 1:2.5.1
[5,712 B]
Get:20 http://archive.ubuntu.com/ubuntu bionic/main amd64 rake all 12.3.1-1
[45.1 kB]
Get:21 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-did-you-mean all
1.2.0-2 [9,700 B]
Get:22 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-minitest all
5.10.3-1 [38.6 kB]
Get:23 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-net-telnet all
0.1.1-2 [12.6 kB]
```

```
Get:24 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-power-assert all
0.3.0-1 [7,952 B]
Get:25 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-test-unit all
3.2.5-1 [61.1 kB]
Get:26 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libruby2.5
amd64 2.5.1-1ubuntu1.6 [3,069 kB]
Get:27 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libsynctex1
amd64 2017.20170613.44572-8ubuntu0.1 [41.4 kB]
Get:28 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libtexlua52
amd64 2017.20170613.44572-8ubuntu0.1 [91.2 kB]
Get:29 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libtexluajit2
amd64 2017.20170613.44572-8ubuntu0.1 [230 kB]
Get:30 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libzzip-0-13
amd64 0.13.62-3.1ubuntu0.18.04.1 [26.0 kB]
Get:31 http://archive.ubuntu.com/ubuntu bionic/main amd64 lmodern all 2.004.5-3
[9,631 kB]
Get:32 http://archive.ubuntu.com/ubuntu bionic/main amd64 preview-latex-style
all 11.91-1ubuntu1 [185 kB]
Get:33 http://archive.ubuntu.com/ubuntu bionic/main amd64 t1utils amd64 1.41-2
[56.0 kB]
Get:34 http://archive.ubuntu.com/ubuntu bionic/universe amd64 tex-gyre all
20160520-1 [4,998 kB]
Get:35 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 texlive-
binaries amd64 2017.20170613.44572-8ubuntu0.1 [8,179 kB]
Get:36 http://archive.ubuntu.com/ubuntu bionic/main amd64 texlive-base all
2017.20180305-1 [18.7 MB]
Get:37 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-fonts-
recommended all 2017.20180305-1 [5,262 kB]
Get:38 http://archive.ubuntu.com/ubuntu bionic/main amd64 texlive-latex-base all
2017.20180305-1 [951 kB]
Get:39 http://archive.ubuntu.com/ubuntu bionic/main amd64 texlive-latex-
recommended all 2017.20180305-1 [14.9 MB]
Get:40 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive all
2017.20180305-1 [14.4 kB]
Get:41 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-pictures
all 2017.20180305-1 [4,026 kB]
Get:42 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-latex-
extra all 2017.20180305-2 [10.6 MB]
Get:43 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-plain-
generic all 2017.20180305-2 [23.6 MB]
Get:44 http://archive.ubuntu.com/ubuntu bionic/universe amd64 tipa all 2:1.3-20
[2,978 kB]
Get:45 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-xetex all
2017.20180305-1 [10.7 MB]
Fetched 146 MB in 11s (13.3 MB/s)
Extracting templates from packages: 100%
Preconfiguring packages ...
Selecting previously unselected package fonts-droid-fallback.
```

```
(Reading database ... 125048 files and directories currently installed.)
Preparing to unpack .../00-fonts-droid-fallback_1%3a6.0.1r16-1.1_all.deb ...
Unpacking fonts-droid-fallback (1:6.0.1r16-1.1) ...
Selecting previously unselected package fonts-lato.
Preparing to unpack .../01-fonts-lato_2.0-2_all.deb ...
Unpacking fonts-lato (2.0-2) ...
Selecting previously unselected package poppler-data.
Preparing to unpack .../02-poppler-data_0.4.8-2_all.deb ...
Unpacking poppler-data (0.4.8-2) ...
Selecting previously unselected package tex-common.
Preparing to unpack .../03-tex-common_6.09_all.deb ...
Unpacking tex-common (6.09) ...
Selecting previously unselected package fonts-lmodern.
Preparing to unpack .../04-fonts-lmodern_2.004.5-3_all.deb ...
Unpacking fonts-lmodern (2.004.5-3) ...
Selecting previously unselected package fonts-noto-mono.
Preparing to unpack .../05-fonts-noto-mono_20171026-2_all.deb ...
Unpacking fonts-noto-mono (20171026-2) ...
Selecting previously unselected package fonts-texgyre.
Preparing to unpack .../06-fonts-texgyre_20160520-1_all.deb ...
Unpacking fonts-texgyre (20160520-1) ...
Selecting previously unselected package libcupsfilters1:amd64.
Preparing to unpack .../07-libcupsfilters1_1.20.2-0ubuntu3.1_amd64.deb ...
Unpacking libcupsfilters1:amd64 (1.20.2-0ubuntu3.1) ...
Selecting previously unselected package libcupsimage2:amd64.
Preparing to unpack .../08-libcupsimage2_2.2.7-1ubuntu2.7_amd64.deb ...
Unpacking libcupsimage2:amd64 (2.2.7-1ubuntu2.7) ...
Selecting previously unselected package libijs-0.35:amd64.
Preparing to unpack .../09-libijs-0.35_0.35-13_amd64.deb ...
Unpacking libijs-0.35:amd64 (0.35-13) ...
Selecting previously unselected package libjbig2dec0:amd64.
Preparing to unpack .../10-libjbig2dec0_0.13-6_amd64.deb ...
Unpacking libjbig2dec0:amd64 (0.13-6) ...
Selecting previously unselected package libgs9-common.
Preparing to unpack .../11-libgs9-common_9.26~dfsg+0-0ubuntu0.18.04.12_all.deb
...
Unpacking libgs9-common (9.26~dfsg+0-0ubuntu0.18.04.12) ...
Selecting previously unselected package libgs9:amd64.
Preparing to unpack .../12-libgs9_9.26~dfsg+0-0ubuntu0.18.04.12_amd64.deb ...
Unpacking libgs9:amd64 (9.26~dfsg+0-0ubuntu0.18.04.12) ...
Selecting previously unselected package libkpathsea6:amd64.
Preparing to unpack .../13-libkpathsea6_2017.20170613.44572-8ubuntu0.1_amd64.deb
...
Unpacking libkpathsea6:amd64 (2017.20170613.44572-8ubuntu0.1) ...
Selecting previously unselected package libpotrace0.
Preparing to unpack .../14-libpotrace0_1.14-2_amd64.deb ...
Unpacking libpotrace0 (1.14-2) ...
Selecting previously unselected package libptexenc1:amd64.
```

```
Preparing to unpack .../15-libptexenc1_2017.20170613.44572-8ubuntu0.1_amd64.deb
...
Unpacking libptexenc1:amd64 (2017.20170613.44572-8ubuntu0.1) ...
Selecting previously unselected package rubygems-integration.
Preparing to unpack .../16-rubygems-integration_1.11_all.deb ...
Unpacking rubygems-integration (1.11) ...
Selecting previously unselected package ruby2.5.
Preparing to unpack .../17-ruby2.5_2.5.1-1ubuntu1.6_amd64.deb ...
Unpacking ruby2.5 (2.5.1-1ubuntu1.6) ...
Selecting previously unselected package ruby.
Preparing to unpack .../18-ruby_1%3a2.5.1_amd64.deb ...
Unpacking ruby (1:2.5.1) ...
Selecting previously unselected package rake.
Preparing to unpack .../19-rake_12.3.1-1_all.deb ...
Unpacking rake (12.3.1-1) ...
Selecting previously unselected package ruby-did-you-mean.
Preparing to unpack .../20-ruby-did-you-mean_1.2.0-2_all.deb ...
Unpacking ruby-did-you-mean (1.2.0-2) ...
Selecting previously unselected package ruby-minitest.
Preparing to unpack .../21-ruby-minitest_5.10.3-1_all.deb ...
Unpacking ruby-minitest (5.10.3-1) ...
Selecting previously unselected package ruby-net-telnet.
Preparing to unpack .../22-ruby-net-telnet_0.1.1-2_all.deb ...
Unpacking ruby-net-telnet (0.1.1-2) ...
Selecting previously unselected package ruby-power-assert.
Preparing to unpack .../23-ruby-power-assert_0.3.0-1_all.deb ...
Unpacking ruby-power-assert (0.3.0-1) ...
Selecting previously unselected package ruby-test-unit.
Preparing to unpack .../24-ruby-test-unit_3.2.5-1_all.deb ...
Unpacking ruby-test-unit (3.2.5-1) ...
Selecting previously unselected package libruby2.5:amd64.
Preparing to unpack .../25-libruby2.5_2.5.1-1ubuntu1.6_amd64.deb ...
Unpacking libruby2.5:amd64 (2.5.1-1ubuntu1.6) ...
Selecting previously unselected package libsynctex1:amd64.
Preparing to unpack .../26-libsynctex1_2017.20170613.44572-8ubuntu0.1_amd64.deb
...
Unpacking libsynctex1:amd64 (2017.20170613.44572-8ubuntu0.1) ...
Selecting previously unselected package libtexlua52:amd64.
Preparing to unpack .../27-libtexlua52_2017.20170613.44572-8ubuntu0.1_amd64.deb
...
Unpacking libtexlua52:amd64 (2017.20170613.44572-8ubuntu0.1) ...
Selecting previously unselected package libtexluajit2:amd64.
Preparing to unpack
.../28-libtexluajit2_2017.20170613.44572-8ubuntu0.1_amd64.deb ...
Unpacking libtexluajit2:amd64 (2017.20170613.44572-8ubuntu0.1) ...
Selecting previously unselected package libzzip-0-13:amd64.
Preparing to unpack .../29-libzzip-0-13_0.13.62-3.1ubuntu0.18.04.1_amd64.deb ...
Unpacking libzzip-0-13:amd64 (0.13.62-3.1ubuntu0.18.04.1) ...
```

```
Selecting previously unselected package lmodern.
Preparing to unpack .../30-lmodern_2.004.5-3_all.deb ...
Unpacking lmodern (2.004.5-3) ...
Selecting previously unselected package preview-latex-style.
Preparing to unpack .../31-preview-latex-style_11.91-1ubuntu1_all.deb ...
Unpacking preview-latex-style (11.91-1ubuntu1) ...
Selecting previously unselected package t1utils.
Preparing to unpack .../32-t1utils_1.41-2_amd64.deb ...
Unpacking t1utils (1.41-2) ...
Selecting previously unselected package tex-gyre.
Preparing to unpack .../33-tex-gyre_20160520-1_all.deb ...
Unpacking tex-gyre (20160520-1) ...
Selecting previously unselected package texlive-binaries.
Preparing to unpack .../34-texlive-
binaries_2017.20170613.44572-8ubuntu0.1_amd64.deb ...
Unpacking texlive-binaries (2017.20170613.44572-8ubuntu0.1) ...
Selecting previously unselected package texlive-base.
Preparing to unpack .../35-texlive-base_2017.20180305-1_all.deb ...
Unpacking texlive-base (2017.20180305-1) ...
Selecting previously unselected package texlive-fonts-recommended.
Preparing to unpack .../36-texlive-fonts-recommended_2017.20180305-1_all.deb ...
Unpacking texlive-fonts-recommended (2017.20180305-1) ...
Selecting previously unselected package texlive-latex-base.
Preparing to unpack .../37-texlive-latex-base_2017.20180305-1_all.deb ...
Unpacking texlive-latex-base (2017.20180305-1) ...
Selecting previously unselected package texlive-latex-recommended.
Preparing to unpack .../38-texlive-latex-recommended_2017.20180305-1_all.deb ...
Unpacking texlive-latex-recommended (2017.20180305-1) ...
Selecting previously unselected package texlive.
Preparing to unpack .../39-texlive_2017.20180305-1_all.deb ...
Unpacking texlive (2017.20180305-1) ...
Selecting previously unselected package texlive-pictures.
Preparing to unpack .../40-texlive-pictures_2017.20180305-1_all.deb ...
Unpacking texlive-pictures (2017.20180305-1) ...
Selecting previously unselected package texlive-latex-extra.
Preparing to unpack .../41-texlive-latex-extra_2017.20180305-2_all.deb ...
Unpacking texlive-latex-extra (2017.20180305-2) ...
Selecting previously unselected package texlive-plain-generic.
Preparing to unpack .../42-texlive-plain-generic_2017.20180305-2_all.deb ...
Unpacking texlive-plain-generic (2017.20180305-2) ...
Selecting previously unselected package tipa.
Preparing to unpack .../43-tipa_2%3a1.3-20_all.deb ...
Unpacking tipa (2:1.3-20) ...
Selecting previously unselected package texlive-xetex.
Preparing to unpack .../44-texlive-xetex_2017.20180305-1_all.deb ...
Unpacking texlive-xetex (2017.20180305-1) ...
Setting up libgs9-common (9.26~dfsg+0-0ubuntu0.18.04.12) ...
Setting up libkpathsea6:amd64 (2017.20170613.44572-8ubuntu0.1) ...
```

```
Setting up libtexlua52:amd64 (2017.20170613.44572-8ubuntu0.1) ...
Setting up fonts-droid-fallback (1:6.0.1r16-1.1) ...
Setting up libsynctex1:amd64 (2017.20170613.44572-8ubuntu0.1) ...
Setting up libptexenc1:amd64 (2017.20170613.44572-8ubuntu0.1) ...
Setting up tex-common (6.09) ...
update-language: texlive-base not installed and configured, doing nothing!
Setting up poppler-data (0.4.8-2) ...
Setting up tex-gyre (20160520-1) ...
Setting up preview-latex-style (11.91-1ubuntu1) ...
Setting up fonts-texgyre (20160520-1) ...
Setting up fonts-noto-mono (20171026-2) ...
Setting up fonts-lato (2.0-2) ...
Setting up libcupsfilters1:amd64 (1.20.2-0ubuntu3.1) ...
Setting up libcupsimage2:amd64 (2.2.7-1ubuntu2.7) ...
Setting up libjbig2dec0:amd64 (0.13-6) ...
Setting up ruby-did-you-mean (1.2.0-2) ...
Setting up t1utils (1.41-2) ...
Setting up ruby-net-telnet (0.1.1-2) ...
Setting up libijs-0.35:amd64 (0.35-13) ...
Setting up rubygems-integration (1.11) ...
Setting up libpotrace0 (1.14-2) ...
Setting up ruby-minitest (5.10.3-1) ...
Setting up libzzip-0-13:amd64 (0.13.62-3.1ubuntu0.18.04.1) ...
Setting up libgs9:amd64 (9.26~dfsg+0-0ubuntu0.18.04.12) ...
Setting up libtexluajit2:amd64 (2017.20170613.44572-8ubuntu0.1) ...
Setting up fonts-lmodern (2.004.5-3) ...
Setting up ruby-power-assert (0.3.0-1) ...
Setting up texlive-binaries (2017.20170613.44572-8ubuntu0.1) ...
update-alternatives: using /usr/bin/xdvi-xaw to provide /usr/bin/xdvi.bin
(xdvi.bin) in auto mode
update-alternatives: using /usr/bin/bibtex.original to provide /usr/bin/bibtex
(bibtex) in auto mode
Setting up texlive-base (2017.20180305-1) ...
mktexlsr: Updating /var/lib/texmf/ls-R-TEXLIVEDIST...
mktexlsr: Updating /var/lib/texmf/ls-R-TEXMFMAIN...
mktexlsr: Updating /var/lib/texmf/ls-R...
mktexlsr: Done.
tl-paper: setting paper size for dvips to a4: /var/lib/texmf/dvips/config
/config-paper.ps
tl-paper: setting paper size for dvipdfmx to a4: /var/lib/texmf/dvipdfmx
/dvipdfmx-paper.cfg
tl-paper: setting paper size for xdvi to a4: /var/lib/texmf/xdvi/XDvi-paper
tl-paper: setting paper size for pdftex to a4:
/var/lib/texmf/tex/generic/config/pdftexconfig.tex
Setting up texlive-fonts-recommended (2017.20180305-1) ...
Setting up texlive-plain-generic (2017.20180305-2) ...
Setting up texlive-latex-base (2017.20180305-1) ...
Setting up lmodern (2.004.5-3) ...
```

```
Setting up texlive-latex-recommended (2017.20180305-1) ...
Setting up texlive-pictures (2017.20180305-1) ...
Setting up tipa (2:1.3-20) ...
Regenerating '/var/lib/texmf/fmtutil.cnf-DEBIAN'... done.
Regenerating '/var/lib/texmf/fmtutil.cnf-TEXLIVEDIST'... done.
update-fmtutil has updated the following file(s):
        /var/lib/texmf/fmtutil.cnf-DEBIAN
        /var/lib/texmf/fmtutil.cnf-TEXLIVEDIST
If you want to activate the changes in the above file(s),
you should run fmtutil-sys or fmtutil.
Setting up texlive (2017.20180305-1) ...
Setting up texlive-latex-extra (2017.20180305-2) ...
Setting up texlive-xetex (2017.20180305-1) ...
Setting up ruby2.5 (2.5.1-1ubuntu1.6) ...
Setting up ruby (1:2.5.1) ...
Setting up ruby-test-unit (3.2.5-1) ...
Setting up rake (12.3.1-1) ...
Setting up libruby2.5:amd64 (2.5.1-1ubuntu1.6) ...
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for fontconfig (2.12.6-0ubuntu2) ...
Processing triggers for tex-common (6.09) ...
Running updmap-sys. This may take some time... done.
Running mktexlsr /var/lib/texmf ... done.
Building format(s) --all.
        This may take some time... done.
Collecting pypandoc
  Downloading https://files.pythonhosted.org/packages/71/81/00184643e5a10a456b41
18fc12c96780823adb8ed974eb2289f29703b29b/pypandoc-1.4.tar.gz
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-
packages (from pypandoc) (42.0.1)
Requirement already satisfied: pip>=8.1.0 in /usr/local/lib/python3.6/dist-
packages (from pypandoc) (19.3.1)
Requirement already satisfied: wheel>=0.25.0 in /usr/local/lib/python3.6/dist-
packages (from pypandoc) (0.33.6)
Building wheels for collected packages: pypandoc
  Building wheel for pypandoc (setup.py) ... done
  Created wheel for pypandoc: filename=pypandoc-1.4-cp36-none-any.whl size=16716
sha256=fe8c8fc29646171464f4e5b7acf9ad9cae1fd91f0071d04bd12ab31322aa56a1
  Stored in directory: /root/.cache/pip/wheels/3e/55/4f/59e0fa0914f3db52e87c0642
c5fb986871dfbbf253026e639f
Successfully built pypandoc
Installing collected packages: pypandoc
Successfully installed pypandoc-1.4
```

[0]:
```
# Mount your google drive to get access to your ipynb files
from google.colab import drive
drive.mount('/content/drive')
```

[0]:
```
!jupyter nbconvert --to PDF "drive/My Drive/Colab Notebooks/latestDataAnalisi.
↪ipynb"
```