# **A-Fixer Uilities**

**#Program to demonstrate fixer utilities**

**#Removing leading or lagging spaces from a data entry**

print("#Removing leading or lagging spaces from a data entry")

baddata="  Datascience with too many spaces is bad   "

print('>',baddata,'<')

cleandata=baddata.strip()

print(">",cleandata,"<")

**#Removing non-printable characters from the dataentry**

print("\n#Removing non-printable characters from the data entry")

import string

printable=set(string.printable)

baddata="Data\x00science with too many funny\x01 characters is \x10bad!!!"

cleandata=''.join(filter(lambda x:x in string.printable,baddata))

print("Baddata:",baddata)

print("Cleandata:",cleandata)

print('\n#Reformatting date entry to match specific formatting criteria')

print('#Convert YYYY/MM/DD TO DD Month YYYY')

import datetime

baddate = datetime.date(2019,10,31)

baddata = format(baddate,'%Y-%m-%d')

gooddate = datetime.datetime.strptime(baddata,'%Y-%m-%d')

gooddata = format(gooddate,'%d %B %Y')

print('BadData: ',baddata)

print('GoodData: ',gooddata)

**OutPut:**

#Removing leading or lagging spaces from a data entry

> Datascience with too many spaces is bad <

> Datascience with too many spaces is bad <

#Removing non-printable characters from the data entry

Baddata: Data�science with too many funny characters is bad!!!

Cleandata: Datascience with too many funny characters is bad!!!

#Reformatting date entry to match specific formatting criteria

#Convert YYYY/MM/DD TO DD Month YYYY

BadData: 2019-10-31

GoodData: 31 October 2019

### **B-Data Binning**

import numpy as np  
import matplotlib.mlab as mlab  
import matplotlib.pyplot as plt

import scipy.stats as stats

np.random.seed(0)

**# example data**

mu = 90 # mean of distribution

sigma = 25 # standard deviation of distribution

x = mu + sigma \* np.random.randn(5000)

num\_bins = 25

fig, ax = plt.subplots()

**# the histogram of the data**

n, bins, patches = ax.hist(x, num\_bins, density=1)

**# add a 'best fit' line**

y = stats.norm.pdf(bins, mu, sigma)

**# mlab.normpdf(bins, mu, sigma)**

ax.plot(bins, y, '--')

ax.set\_xlabel('Example Data')

ax.set\_ylabel('Probability density')

sTitle=r'Histogram ' + str(len(x)) + ' entries into ' + str(num\_bins)

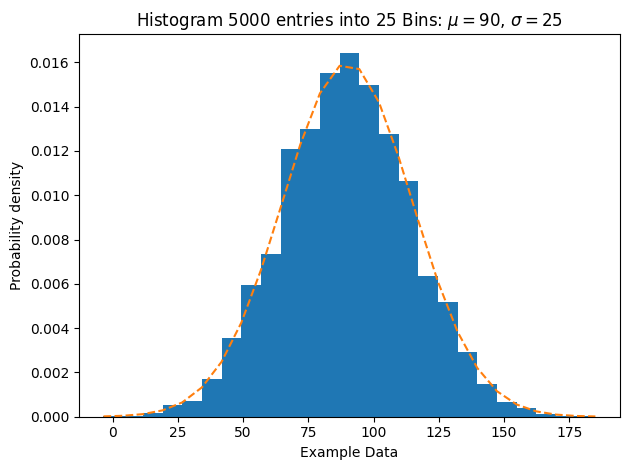
+ ' Bins: $\mu=' + str(mu) + '$, $\sigma=' +str(sigma) + '$'

ax.set\_title(sTitle)

fig.tight\_layout()

sPathFig = ’ /content/Histogram.png'

fig.savefig(sPathFig)

plt.show()

## **C-Averaging Of Data**

import pandas as pd

InputFileName='IP\_DATA\_CORE.csv'

OutputFileName='Retrieve\_Router\_Location.csv'

Base='C:/VKHCG'

print('Working Base :',Base, ' using ')

sFileName=Base + '/01-Vermeulen/00-RawData/' + InputFileName

print('Loading :',sFileName)

IP\_DATA\_ALL=pd.read\_csv(sFileName,header=0,low\_memory=False,usecols=['Country','Place Name','Latitude','Longitude'], encoding="latin-1")

IP\_DATA\_ALL.rename(columns={'Place Name': 'Place\_Name'}, inplace=True)

AllData=IP\_DATA\_ALL[['Country', 'Place\_Name','Latitude']]

print(AllData)

MeanData=AllData.groupby(['Country', 'Place\_Name'])['Latitude'].mean()

print(MeanData)

**OutPut:**

Country Place\_Name Latitude

0 BW Gaborone -24.6464

1 BW Gaborone -24.6464

2 BW Gaborone -24.6464

3 BW Gaborone -24.6464

4 BW Gaborone -24.6464

.. ... ... ...

194 DZ Algiers 36.7631

195 DZ Algiers 36.7631

196 DZ Algiers 36.7631

197 DZ Algiers 36.7631

198 DZ Algiers 36.7631

[199 rows x 3 columns]

------------(MEAN)--------------

Country Place\_Name

BW Gaborone -24.6464

DZ Algiers 36.7631

GH Accra 5.5500

Kumasi 6.6833

Takoradi 4.8833

Tema 5.6167

MZ Maputo -25.9653

NE Niamey 13.5167

Name: Latitude, dtype: float64

#### **D-Outliers Detection**

import pandas as pd

InputFileName='IP\_DATA\_CORE.csv'

OutputFileName='Retrieve\_Router\_Location.csv'

Base='C:/VKHCG'

print('Working Base :',Base)

sFileName=Base + '/01-Vermeulen/00-RawData/' + InputFileName

print('Loading :',sFileName)

IP\_DATA\_ALL=pd.read\_csv(sFileName,header=0,low\_memory=False,usecols=['Country','Place Name','Latitude','Longitude'], encoding="latin-1")

IP\_DATA\_ALL.rename(columns={'Place Name': 'Place\_Name'}, inplace=True)

LondonData=IP\_DATA\_ALL.loc[IP\_DATA\_ALL['Place\_Name']=='London']

AllData=LondonData[['Country', 'Place\_Name','Latitude']]

print('All Data\n', AllData)

MeanData=AllData.groupby(['Country', 'Place\_Name'])['Latitude'].mean()

StdData=AllData.groupby(['Country', 'Place\_Name'])['Latitude'].std()

print('Outliers')

UpperBound=float(MeanData+StdData)

print('Higher than ', UpperBound)

OutliersHigher=AllData[AllData.Latitude>UpperBound]

print(OutliersHigher)

LowerBound=float(MeanData-StdData)

print('Lower than ', LowerBound)

OutliersLower=AllData[AllData.Latitude<LowerBound]

print(OutliersLower)

print('Not Outliers')

OutliersNot=AllData[(AllData.Latitude>=LowerBound) & (AllData.Latitude<=UpperBound)]

print(OutliersNot)

**Output:**

All Data

Empty DataFrame

Columns: [Country, Place\_Name, Latitude]

Index: []

Outliers

Higher than Series([], Name: Latitude, dtype: float64)