MINERAÇÃO MASSIVA DE DADOS

Parte 4 – Prática de uso do Spark, gráficos e estatística

Marcial Porto Fernández
marcial@larces.uece.br

Mestrado Acadêmico em Ciência da Computação (MACC)
Universidade Estadual do Ceará (UECE)
Laboratório de Sistemas Digitais (LASID)


LET'S
CODE

Sumário

- Ambiente de Trabalho
- Introdução ao Python
- Explicação do programa exemplo

Sumário

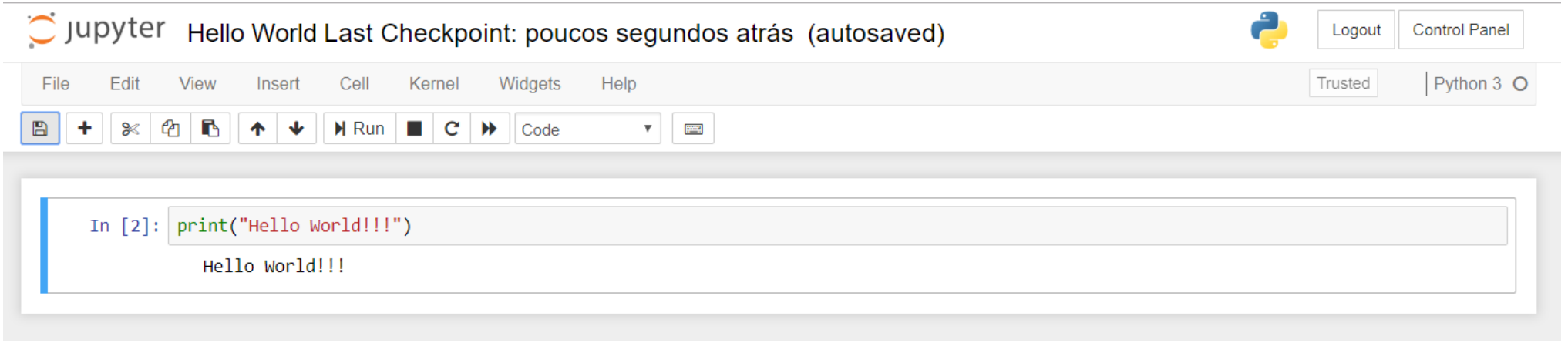
- Ambiente de Trabalho
- Introdução ao Python
- Explicação do programa exemplo

Ambiente de trabalho

- Apache Spark instalado em um servidor biprecessador e 64 GB de memória RAM
  - 1 TB em RAID para armazenar dados dos usuários
- Datasets instalados em servidor de armazenamento
  - Capacidade total 96 TB, sendo disponível ~40 TB
- Futuramente um cluster....
- Acesso via Jupyter Notebook: https://10.129.64.20:8000
  - Aceitar o certificado....
  - Login em conta criada no servidor
- Datasets podem ser vistos em http://10.129.64.20 e acessados pela aplicação como **/data/....**
- Disponibilização de Cheatsheets na página do curso

Jupyter Notebook

- Permite usar recursos do servidor de forma fácil



Sumário

- Ambiente de Trabalho
- Introdução ao Python
- Explicação do programa exemplo

Histórico

- Criada por Guido van Rossum em 1989, no Instituto Nacional de Matemática e Ciênciada Computação da Holanda (CWI)
- O nome é uma homenagem ao grupo humorístico britânico Monty Python
- Linguagem Open Source, com desenvolvimento comunitário
- É uma linguagem interpretada, não é necessário compilar o programa
- É uma linguagem de aprendizado fácil, com sintaxe clara e concisa
- É uma linguagem estruturada, orientada a objetos e funcional
- Duas versões incompatíveis: Python 2.7 e 3.7

Comentário, entrada e saída de dados

- Comentário: #
- Entrada e escrita de dados: input() e print()
- Continuação da linha: \

```
In [1]: # Comentário em uma linha
        # Comentário em outra linha

        print("Hello \
        World...")
```

```
Hello World...
```

Programa Hello World

Vamos fazer o primeiro programa que todo programador deve escrever quando aprende uma nova linguagem

```
In [1]: # Comentário em uma linha
        # Comentário em outra linha

        print("Hello World...")
```

```
In [*]: # Entrada de dados
        nome=input("Escreva seu nome: ")
        print('Olá',nome)

        Escreva seu nome: Marcial
```

```
In [2]: # Entrada de dados
        nome=input("Escreva seu nome: ")
        print('Olá',nome)

        Escreva seu nome: Marcial
        Olá Marcial
```

Variáveis e atribuições

- Variáveis não são declaradas
- Podem ser: caracter (char), string, booleano (boolean), números inteiros (int), números reais (float), números complexos (complex)
- Strings são identificados por “ ou ‘
- Variáveis tipadas, não pode somar tipos diferentes (string+inteiros). Mas pode somar tipos numéricos (inteiros+float)

```
In [3]: a="Hello"
        b='"World"'
        print(a+' '+b)

Hello "World"
```

```
In [4]: a=23
        b=56
        print(a+b)
        c=a+b
        print(c)

79
79
```

```
In [5]: a="Happy"
        b=2018
        print(a,b)

Happy 2018
```

Array e Listas

- Array é um vetor do mesmo tipo
- Um elemento é identificado como nome[posição]
- Lista pode ter vários tipos
- Uma lista pode acrescentar e retirar elementos:
  - append() e remove()

```
In [6]: array="abcde"
        print(array)
        print(array[1])
        print(array[3]+array[4])
```

```
abcde
b
de
```

```
In [7]: valor=[1,2,3,4,5]
        print(valor)
        print(valor[1])
        print(valor[2]+valor[3])
```

```
[1, 2, 3, 4, 5]
2
7
```

```
In [8]: lista=[1,'dois',3.0,0.4e2,[5,'seis']]
        print(lista)
        lista.append(7)
        print(lista)
        lista.remove(3.0)
        print(lista)
```

```
[1, 'dois', 3.0, 40.0, [5, 'seis']]
[1, 'dois', 3.0, 40.0, [5, 'seis'], 7]
[1, 'dois', 40.0, [5, 'seis'], 7]
```

Dicionário

- Estrutura muito usada em Python: Constituída por uma chave e um valor

```
>>> di = {'Julio': 'C', 'Jaime': 'Python', 'Ana': 'Ruby', 'Cláudia': 'Java'}
>>> di['Ana']
'Ruby'
>>> x = di['Julio']
>>> print(x)
C
```

```
>>> di = {'Julio': 'C', 'Jaime': 'Python', 'Ana': 'Ruby', 'Cláudia': 'Java', 'Mauro': 'PHP'}
>>> di['Jaime'] = 'PHP'
>>> di
{'Cláudia': 'Java', 'Jaime': 'PHP', 'Julio': 'C', 'Mauro': 'PHP', 'Ana': 'Ruby'}
>>>
```

```
>>> di = {'Claudia': 'Java', 'Jaime': 'Python', 'Julio': 'C', 'Mauro': 'PHP', 'Ana': 'Ruby'}
>>> del di['Mauro']
>>> di
{'Claudia': 'Java', 'Jaime': 'Python', 'Julio': 'C', 'Ana': 'Ruby'}
>>>
```

Blocos

- Endentação é importante! Use TAB

```
media = (nota1 + nota2)/2

print()
if media >= 7:
    print('{0} aprovado com media = {1:4.2f}'.format(nome, media))
elif media < 7 and media >= 3:
    print('{0} vai para prova final com media = {1:4.2f}'.format(nome, media))
else:
    print('{0} fiocu reprovado com media = {1:4.2f}'.format(nome, media))
print()
```

```
while contador <= 9:
    resultado = numero * contador
    print("{0} x {1} = {2}".format(numero, contador, resultado))
    contador += 1
print()
```

```
lista = ["p", "y", "t", "h", "o", "n"]
for item in lista:
    print item
```

Função

- Para realizar operações em vários pontos do programa

```
def spam():
eggs = 12
return eggs

print(spam())
```

```
def add_two(a, b):
    c = a + b
    return c
```

- Chamando funções externas

```
import math
print(math.sqrt(25))
```

```
from math import sqrt
```

```
from datetime import datetime
now = datetime.now()
print(now)
```

Sumário

- Ambiente de Trabalho
- Introdução ao Python
- Explicação do programa exemplo

Exemplo de Jupyter Notebook Spark

- Carrega o ambiente Spark
- Chama as bibliotecas necessárias

In [1]:
```
# Inicia ambiente Spark
import findspark
findspark.init()
```

In [2]:
```
# Carrega módulos
# Carrega módulo PySpark para permitir ao Python acessar funções do Spark
from pyspark.sql import SparkSession

# Importa biblioteca Pandas para Análise de Dados
import pandas as pd

# Importa biblioteca Numpy para Tratamento de Matrizes
import numpy as np

# Importa biblioteca Matplotlib para desenhar gráficos
import matplotlib.pyplot as plt

# Indica para mostrar os graficos inline no Notebook
%matplotlib inline
```

Exemplo de Jupyter Notebook Spark

- Cria seção Spark

In [3]:
```
# Cria Sessão Spark
sparkSession = SparkSession.builder \
       .master("local") \
       .appName("Estatistica e Gráficos") \
       .getOrCreate()
```

Fonte de dados

- Um arquivo com as informações organizadas em forma tabular
- Os dados devem ser de fácil entendimento e lidos por qualquer software.
- A fonte de dados NÃO pode ser alterada!!! Nunca escrevemos sobre a base antiga
  - Se for necessário criamos uma nova base de dados e salvamos com outro nome.
- O formato mais usado é o CSV
  - Outros formatos: JSON, XML, Excel
- Se a base de dados for muito grande pode ser dividida em vários arquivos

Arquivo CSV

- Formato de dataset muito usual.
- Lista de valores separados por vírgula
  - Leve, compacto e fácil leitura
- Primeira linha define o título do campo (coluna)
- Se o dataset for de figuras, existe um arquivo CSV com o nome dos arquivos e a categoria

```
1 Country,Year,Status,Life expectancy,Adult Mortality,infant deaths,Alcohol,percentage expenditure,Hepatitis
  B,Measles,BMI,under-five deaths,Polio,Total expenditure,Diphtheria,HIV/AIDS,GDP,Population,thinness 1-19
  years,thinness 5-9 years,Income composition of resources,Schooling
2 Afghanistan,2015,Developing,65,263,62,0.01,71.27962362,65,1154,19.1,83,6,8.16,65,0.1,584.25921,33736494,17.2,17.3,0.
  479,10.1
3 Afghanistan,2014,Developing,59.9,271,64,0.01,73.52358168,62,492,18.6,86,58,8.18,62,0.1,612.696514,327582,17.5,17.5,0
  .476,10
4 Afghanistan,2013,Developing,59.9,268,66,0.01,73.21924272,64,430,18.1,89,62,8.13,64,0.1,631.744976,31731688,17.7,17.7
  ,0.47,9.9
5 Afghanistan,2012,Developing,59.5,272,69,0.01,78.1842153,67,2787,17.6,93,67,8.52,67,0.1,669.959,3696958,17.9,18,0.463
  ,9.8
6 Afghanistan,2011,Developing,59.2,275,71,0.01,7.097108703,68,3013,17.2,97,68,7.87,68,0.1,63.537231,2978599,18.2,18.2,
  0.454,9.5
7 Afghanistan,2010,Developing,58.8,279,74,0.01,79.67936736,66,1989,16.7,102,66,9.2,66,0.1,553.32894,2883167,18.4,18.4,
  0.448,9.2
8 Afghanistan,2009,Developing,58.6,281,77,0.01,56.76221682,63,2861,16.2,106,63,9.42,63,0.1,445.8932979,284331,18.6,18.
  7,0.434,8.9
```

Carrega Arquivo de dados

- Carrega um arquivo CSV para um Dataframe.
- Dataframe é o formato preferível para usar no Spark.
  - O padrão da versão 1 é RDD e para a versão 2 é o DataFrame
  - O dataframe é armazenado em memória e nunca mais precisa ler o disco
  - Existe uma opção de Dataframe para leitura de stream quando o arquivo for muito grande.
  - No Spark após a leitura do dataset, passamos a referenciar tudo pelo nome da variável “df”.

In [4]:
```
# Le arquivo de dados CSV para Dataframe Spark
df = sparkSession.read.format("csv").options(sep=',',header='true',inferschema='true').\
     load("/data//biodata/LifeExpectancy_WHO/Life_Expectancy_Data.csv")

# A partir de agora trabalhanmos apenas com o dataframe 'df'
```

Carrega Arquivo de dados

- Exibe nome dos campos (coluna) da tabela.

```
#Exibe campos da tabela e os tipos de dados
df.dtypes
```

```
Out[4]: [('Country', 'string'),
 ('Year', 'int'),
 ('Status', 'string'),
 ('Life expectancy', 'double'),
 ('Adult Mortality', 'int'),
 ('infant deaths', 'int'),
 ('Alcohol', 'double'),
 ('percentage expenditure', 'double'),
 ('Hepatitis B', 'int'),
 ('Measles', 'int'),
 ('BMI', 'double'),
 ('under-five deaths', 'int'),
 ('Polio', 'int'),
 ('Total expenditure', 'double'),
 ('Diphtheria', 'int'),
 ('HIV/AIDS', 'double'),
 ('GDP', 'double'),
 ('Population', 'double'),
 ('thinness 1-19 years', 'double'),
 ('thinness 5-9 years', 'double'),
 ('Income composition of resources', 'double'),
 ('Schooling', 'double')]
```

Carrega Arquivo de dados

- Exibe conteúdo do Dataframe

```
In [5]: # Exibe primeiros 5 registros do Dataframe
        df.show(5)
```

```
+-----------+----+----------+---------------+--------------+-------------+-------+----------------------+----------+-------+
----+-----------------+-----+-----------------+----------+--------+----------+-----------+------------------+----------------
--+---------------------------------+---------+
|    Country|Year|    Status|Life expectancy|Adult Mortality|infant deaths|Alcohol|percentage expenditure|Hepatitis B|Measles|
BMI|under-five deaths|Polio|Total expenditure|Diphtheria|HIV/AIDS|       GDP| Population|thinness 1-19 years|thinness 5-9 year
s|Income composition of resources|Schooling|
+-----------+----+----------+---------------+--------------+-------------+-------+----------------------+----------+-------+
----+-----------------+-----+-----------------+----------+--------+----------+-----------+------------------+----------------
--+---------------------------------+---------+
|Afghanistan|2015|Developing|           65.0|           263|           62|   0.01|          71.27962362|        65|   1154|
19.1|               83|    6|             8.16|        65|     0.1| 584.25921|3.3736494E7|              17.2|               1
7.3|                            0.479|     10.1|
|Afghanistan|2014|Developing|           59.9|           271|           64|   0.01|          73.52358168|        62|    492|
18.6|               86|   58|             8.18|        62|     0.1|612.696514|   327582.0|              17.5|               1
7.5|                            0.476|     10.0|
|Afghanistan|2013|Developing|           59.9|           268|           66|   0.01|          73.21924272|        64|    430|
18.1|               89|   62|             8.13|        64|     0.1|631.744976|3.1731688E7|              17.7|               1
7.7|                             0.47|      9.9|
|Afghanistan|2012|Developing|           59.5|           272|           69|   0.01|           78.1842153|        67|   2787|
17.6|               93|   67|             8.52|        67|     0.1|   669.959|  3696958.0|              17.9|               1
8.0|                            0.463|      9.8|
```

Explicação dos campos da tabela

Campo	Explicação do valor do campo
Country	Country
Year	Year
Status	Developed or Developing status
Life expectancy	Life Expectancy in age
Adult Mortality	Adult Mortality Rates of both sexes (probability of dying between 15 and 60 years per 1000 population)
Infant deaths	Number of Infant Deaths per 1000 population
Alcohol	Alcohol, recorded per capita (15+) consumption (in litres of pure alcohol)
Percentage expenditure	Expenditure on health as a percentage of Gross Domestic Product per capita (%)
Hepatitis B	Hepatitis B (HepB) immunization coverage among 1-year-olds (%)
Measles	Measles - number of reported cases per 1000 population
BMI	Average Body Mass Index of entire population
under-five deaths	Number of under-five deaths per 1000 population
Polio	Polio (Pol3) immunization coverage among 1-year-olds (%)
Total expenditure	General government expenditure on health as a percentage of total government expenditure (%)
Diphtheria	Diphtheria tetanus toxoid and pertussis (DTP3) immunization coverage among 1-year-olds (%)
HIV/AIDS	Deaths per 1 000 live births HIV/AIDS (0-4 years)
GDP	Gross Domestic Product per capita (in USD)
Population	Population of the country
thinness 1-19 years	Prevalence of thinness among children and adolescents for Age 10 to 19 (%)
thinness 5-9 years	Prevalence of thinness among children for Age 5 to 9 (%)
Income composition of resources	Human Development Index in terms of income composition of resources (index ranging from 0 to 1)
Schooling	Number of years of Schooling (years)

Seleção de Dados Geração de Gráfico

- Para selecionar parte dos dados criamos um novo Dataframe

In [6]:
```
# Para gerar um gráfico precisamos extrair do dataframe os dados relevantes
# criamos um novo Dataframe gr apenas para gerar o gráfico
gr = df.select("Country","Year","Life expectancy").filter(df.Country == 'Brazil') \
    .drop("Country").sort("Year", ascending=True)

gr.show(10)
```

```
+----+---------------+
|Year|Life expectancy|
+----+---------------+
|2000|           75.0|
|2001|           71.0|
|2002|           71.4|
|2003|           71.8|
|2004|           72.0|
|2005|           72.7|
|2006|           73.0|
|2007|           73.3|
|2008|           73.4|
|2009|           73.6|
+----+---------------+
only showing top 10 rows
```

Seleção de Dados Geração de Gráfico

- O Matplotlib não entende Dataframe, precisamos converter para um vetor NumPy

In [7]:
```
# Converte Dataframe em array Numpy, definindo o eixo x e eixo y
gr_x = np.array(gr.select('Year').collect())
gr_y = np.array(gr.select('Life expectancy').collect())

print(gr_x)
print(gr_y)
```

```
[[2000]
 [2001]
 [2002]
 [2003]
 [2004]
 [2005]
 [2006]
 [2007]
```

Seleção de Dados Geração de Gráfico

- Desenha o gráfico

In [8]:
```
# Limpa gráfico anterior
plt.clf()

# Desenha gráfico com sinal + azul
plt.plot(gr_x,gr_y,'b+')

# Define nomes dos eixos
plt.xlabel('Year')
plt.ylabel('Life expectancy (years)')
plt.title(r'Life expectancy in Brazil')
plt.xlim(2000,2016)
plt.grid(color='g', linestyle=':', linewidth=.3)
plt.savefig('./life_expect_BR.png', dpi=300)
plt.show()
```

Cálculo de estatística

- Seleciona Dados

In [9]:
```
# Cria um novo Dataframe apenas com os dados relevantes (Pais, expetativa no ano de 2013)
es = df.select("Country","Year","Life expectancy").filter(df.Year == '2013')
es.show(10)
```

```
+------------------+----+---------------+
|           Country|Year|Life expectancy|
+------------------+----+---------------+
|       Afghanistan|2013|           59.9|
|           Albania|2013|           77.2|
|           Algeria|2013|           75.3|
|            Angola|2013|           51.1|
|Antigua and Barbuda|2013|           76.1|
|         Argentina|2013|           76.0|
|           Armenia|2013|           74.4|
|         Australia|2013|           82.5|
|           Austria|2013|           81.1|
|        Azerbaijan|2013|           72.2|
+------------------+----+---------------+
only showing top 10 rows
```

Cálculo de estatística

- Calcula estatística
- Analisa dados no Panda

In [10]:
```
# Calculos estatisticos
summary = es.describe("Life expectancy")
summary.show()
```

```
+-------+-----------------+
|summary|  Life expectancy|
+-------+-----------------+
|  count|              183|
|   mean|71.23606557377052|
| stddev|8.413771405879869|
|    min|             49.9|
|    max|             87.0|
+-------+-----------------+
```

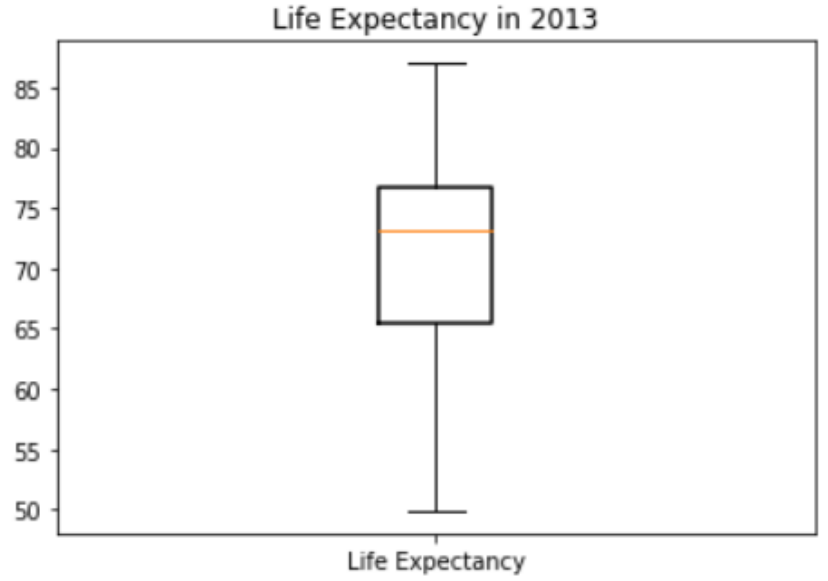
In [11]:
```
# Envia dados para o Panda
pandaDf = summary.toPandas()
min_string = pandaDf[pandaDf['summary'] == 'min']['Life expectancy'].iloc[0]
max_string = pandaDf[pandaDf['summary'] == 'max']['Life expectancy'].iloc[0]
mean_string = pandaDf[pandaDf['summary'] == 'mean']['Life expectancy'].iloc[0]
medianAndQuantiles = es.stat.approxQuantile("Life expectancy",[0.25,0.5,0.75],0.0)
(q1,median,q3) = (medianAndQuantiles[0], medianAndQuantiles[1], medianAndQuantiles[2])
```

Cálculo de estatística

- Desenha Gráfico

```
item["q3"] = q3
item["whislo"] = float(min_string)
item["whishi"] = float(max_string)
item["fliers"] = []
stats = [item]

fig, axes = plt.subplots(1, 1)
axes.bxp(stats)
axes.set_title('Life Expectancy in 2013')

plt.savefig('./life_expect_stat_2013.png', dpi=300)
plt.show()
```

```
<Figure size 432x288 with 0 Axes>
```

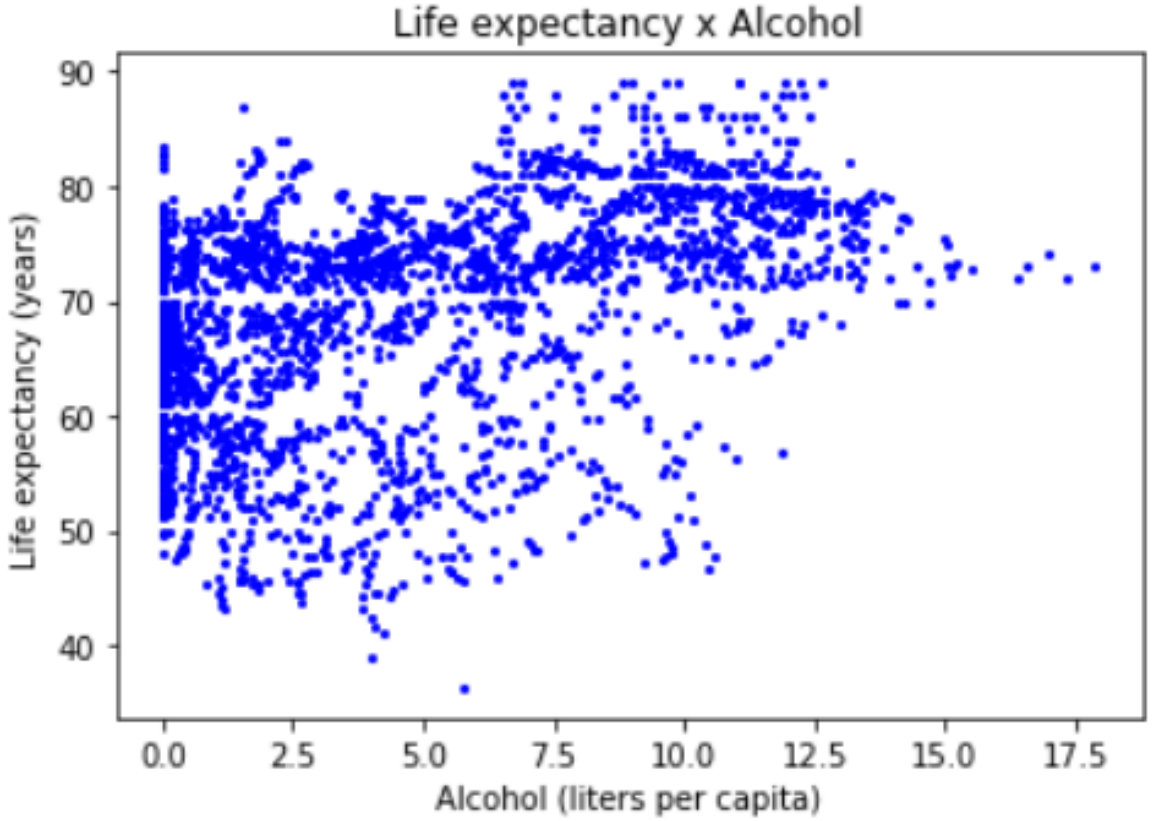


Covariância e Correlação

- Cálculo de covariância e Correlação com o NumPy

In [13]:
```
cov_df = df.select('Life expectancy', 'Alcohol')
#cov_df.show(10)

cov_x = np.array(cov_df.select('Alcohol').collect())
cov_y = np.array(cov_df.select('Life expectancy').collect())

# Limpa gráfico anterior
plt.clf()

# Desenha gráfico com sinal + azul
plt.plot(cov_x,cov_y,'bo',markersize=2)

# Define nomes dos eixos
plt.xlabel('Alcohol (liters per capita)')
plt.ylabel('Life expectancy (years)')
plt.title(r'Life expectancy x Alcohol')
plt.savefig('./life_expect_alcohol_cor.png', dpi=300)
plt.show()

print("Covariance and Correlation Life expectancy x Alcohol")
print("Covariance = %5.2f" % df.stat.cov('Life expectancy', 'Alcohol'))
print("Correlation = %1.2f" % df.stat.corr('Life expectancy', 'Alcohol'))
```

Covariância e Correlação




```
Covariance and Correlation Life expectancy x Alcohol
Covariance = 15.15
Correlation = 0.36
```

Covariância e Correlação

In [14]:
```
cov_df = df.select('Life expectancy', 'Schooling')
#cov_df.show(10)

cov_x = np.array(cov_df.select('Schooling').collect())
cov_y = np.array(cov_df.select('Life expectancy').collect())

# Limpa gráfico anterior
plt.clf()

# Desenha gráfico com sinal + azul
plt.plot(cov_x,cov_y,'bo',markersize=2)

# Define nomes dos eixos
plt.xlabel('Schooling (years of schooll)')
plt.ylabel('Life expectancy (years)')
plt.title(r'Life expectancy x Schooling')
plt.savefig('./life_expect_schooling_cor.png', dpi=300)
plt.show()

print("Covariance and Correlation Life expectancy x Schooling")
print("Covariance = %5.2f" % df.stat.cov('Life expectancy', 'Schooling'))
print("Correlation = %1.2f" % df.stat.corr('Life expectancy', 'Schooling'))
```

Covariância e Correlação


```
Covariance and Correlation Life expectancy x Schooling
Covariance = 24.74
Correlation = 0.56
```

Encerra Spark

- Encerra a sessão Spark
- Encerra o processo, fecha os arquivos abertos e libera a memória.

```
In [15]: sparkSession.stop()
```

OBRIGADO !

marcial@larces.uece.br
http://marcial.larces.uece.br