MBA em Ciência de Dados

Técnicas Avançadas de Captura e Tratamento de Dados

Módulo III - Aquisição e Transformação de Dados

Exercícios - com soluções

Moacir Antonelli Ponti

CeMEAI - ICMC/USP São Carlos

Recomenda-se fortemente que os exercícios sejam feitos sem consultar as respostas antecipadamente.

Utilize as bibliotecas e carrege os dados conforme descrito abaixo

In [4]:
```
# carregando as bibliotecas necessárias
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Exercício 1)

Tomando como base o conteúdo das seções 1.3.1, 1.3.2 e 1.3.3 do Capítulo "Introduction to Data", de Open Statistics. Considere os seguintes cenários:

I - Analisando dados de educação, amostramos 30 escolas de todo o Brasil e observamos que, dessas 30 escolas, 10% não submeteram o resultado Pisa para Escolas, sendo que todas essas estão no estado de São Paulo. Assim, concluímos que existe uma possível relação entre não submissão e o estado de São Paulo. II - Desejamos estudar a percepção da facilidade de usar um aplicativo desenvolvido por nossa empresa para uso no segmento de atividades físicas/esportes. Para isso montamos um questionário e selecionamos 20 pessoas da própria empresa, que não trabalham com desenvolvimento, para avaliar a sua facilidade de uso.

Podemos considerar que I e II representam:

(a) I - evidência confiável e conclusão correta; II - dados com amostragem de conveniência
(b) I - evidência anedotal e conclusão incorreta; II - dados com viés de seleção
(c) I - evidência confiável e conclusão provavelmente correta; II - dados com amostragem representativa
(d) I - evidência anedotal e conclusão incorreta; II - dados com amostragem baseada em agrupamento

Resposta: *I - não é possível confiar em poucos dados, mesmo que sejam 10% dos dados amostrados, representam uma evidência "anedotal" e tirar conclusões seria incorreto; II - ao selecionar pessoas da própria empresa temos um viés de seleção conhecido por amostra de conveniência, sendo a amostra não representativa do que gostaríamos de estudar (usuários gerais).*

Exercício 2)

Gostaríamos de obter e analisar dados disponíveis publicamente a partir de um repositório existente na Internet. Esses dados são referentes a indivíduos que contrataram serviços. Qual a primeira investigação ou procedimento a realizar ao obter esses dados?

(a) conhecer o espaço amostral dos dados e observar questões éticas como a privacidade dos respondentes
(b) realizar uma análise exploratória antes de qualquer análise
(c) auditar os dados, procurando por inconsistências como outliers
(d) inferir/treinar modelos diretamente a partir dos dados e medir sua acurácia

Resposta: *conhecer os dados, seu espaço amostral e verificar questões éticas é mais importante como prioridade do que as demais, pois podemos estar realizando uma análise inválida do ponto de vista dos dados que obtivemos, ou quebrando a ética ou lei de uso desses dados.*

Exercício 3)

Uma empresa deseja entender melhor o potencial de mercado para um novo produto em um certo público alvo. Qual das alternativas abaixo representa a melhor forma de proceder após decidir que a coleta dos dados é necessária?

(a) permitir que os usuários enviem suas opiniões por meio de áudio ou vídeo, para depois coletar os dados a partir desse material
(b) implementar um questionário via aplicativo rapidamente em uma rede social popular e testá-lo massivamente, para verificar se os dados são consistentes com o que é esperado
(c) segmentar o público alvo e pedir ao menos 1 de cada segmento qual a chance, de 1 a 5 do indivíduo comprar esse produto
(d) especificar detalhes dos dados a serem coletados, planejar como obter uma amostra representativa do público alvo

Resposta: *ainda que as outras opções sejam possíveis, é sempre melhor especificar bem os detalhes dos dados a serem coletados, e planejar para que o público alvo seja atingido de maneira uniforme, para não ter viés de seleção.*

Exercício 4)

Acesse o portal : http://catalogo.governoaberto.sp.gov.br/ (http://catalogo.governoaberto.sp.gov.br/)

Procure por duas fontes de dados e verifique o formato em que estão disponíveis

I - "Quantidade de alunos por tipo de ensino da rede estadual - 01/2019" (Secretaria da Educação - Sede)
II - "Pesquisa de Caracterização Socioeconômica do Usuário e seus Hábitos de Viagem - 2018" (Companhia do Metropolitano de São Paulo - Metrô)

Esses dados estão disponíveis no tipo:

(a) I e II são arquivos simples em dados estruturados
(b) I e II são dados estruturados disponíveis em sistema gerenciador de bancos de dados
(c) I são dados estruturados em arquivo simples, II dados não estruturados em arquivo binário
(d) I dados estruturados em arquivo binário, II são dados estruturados em arquivo texto

Resposta: *I está disponível em arquivo CSV com dados estruturados descritos em um "dicionário de dados", enquanto II apenas em formato PDF, binário, e de maneira não estruturada.*

Exercício 5)

Baixe os dados relativo ao item I do exercício anterior, e remova as colunas 21 em diante, mantendo as colunas de 0 a 20.

Essas colunas restantes possuem significado de acordo com o "dicionário de dados" disponível ao visualizar o recurso dos dados. Sendo rotuladas da seguinte forma:

- CDREDE
- DE
- CODMUN
- MUN
- CATEG
- COD_ESC
- TIPOESC
- CODVINC
- NOMESC
- ENDESC
- NUMESC
- BAIESC
- EMAIL
- FONE 1
- ZONA
- ED_INFANTIL
- CLASSES ESPECIAIS
- SALA DE RECURSO
- ANOS INICIAIS
- ANOS FINAIS
- ENSINO MEDIO

Quantas linhas/exemplos existem nessa base de dados e qual é o tipo das variáveis NOMESC e ENSINO MEDIO, respectivamente?

(a) 5366, object, int64
(b) 17366, category, int8
(c) 21, object, int64
(d) 5366, category, float64

Resposta: *ver codigo abaixo*

In [5]:
```
# carregando dados
dc = pd.read_csv("./dados/VW_ALUNOS_POR_ESCOLA_20190517_0.csv", sep=';', header=None)
ncols = dc.shape[1]
dc.drop(dc.columns[range(21,ncols)], axis=1, inplace=True)
dc.head()
ncols = dc.shape[1]
print(ncols)
dc.columns = ['CDREDE', 'DE', 'CODMUN', 'MUN', 'CATEG', 'COD_ESC', 'TIPOESC', 'CODVINC', 'NOMESC', 'ENDESC', 'NUMESC', 'BAIESC', 'EMAIL', 'FONE1', 'ZONA', 'ED_INFANTIL', 'CLASSES ESPECIAIS', 'SALA DE RECURSO', 'ANOS INICIAIS', 'ANOS FINAIS', 'ENSINO MEDIO']
print("linhas: ", dc.shape[0])
```

```
21
linhas:  5366
```

In [7]:
```
# tipos
for var in dc:
    print("%s (%s)" % (var, dc[var].dtype.name))
```

```
CDREDE (int64)
DE (object)
CODMUN (int64)
MUN (object)
CATEG (int64)
COD_ESC (int64)
TIPOESC (int64)
CODVINC (int64)
NOMESC (object)
ENDESC (object)
NUMESC (object)
BAIESC (object)
EMAIL (object)
FONE1 (float64)
ZONA (int64)
ED_INFANTIL (int64)
CLASSES ESPECIAIS (int64)
SALA DE RECURSO (int64)
ANOS INICIAIS (int64)
ANOS FINAIS (int64)
ENSINO MEDIO (int64)
```

Exercício 6)

Visualize os dados únicos e o histograma da variável SALA DE RECURSO.

Realize a discretização da variável utilizando o método do intervalo considerando os seguintes intervalos e rotulos

0 - '0'
1 a 4 - '1 a 4'
5 a 9 - '5 a 9'
10 ou mais - '10+'

Use o método cut() lembrando que os intervalos são definidos de forma que:

$$[a, b, c, d]$$

resulta em 3 intervalos:
(a, b] - entre a e b, não inclui a
(b, c] - entre b e c, não inclui b
(c, d] - entre c e d, não inclui c

Adicione essa nova variável na base, com o nome 'SALA_DE_RECURSO_D'

Responda, quantas linhas recaem em cada um dos 4 intervalos, respectivamente 0; 1 a 4; 5 a 9; e 10+?

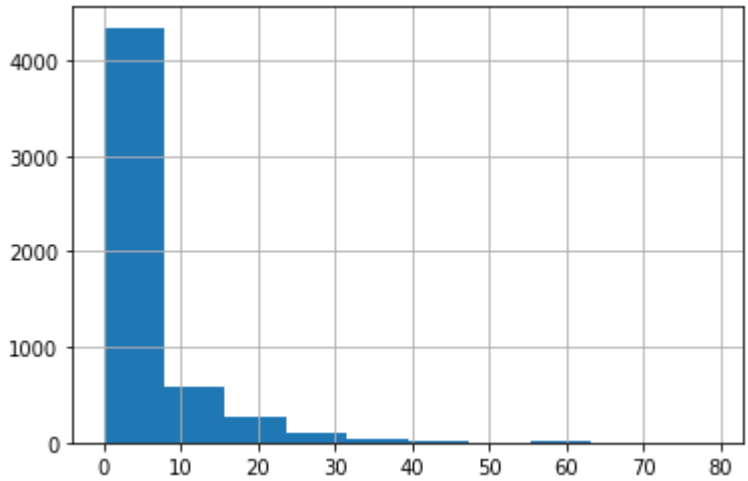
(a) 93, 415, 446, 772
(b) 3636, 413, 470, 847
(c) 772, 446, 415, 93
(d) 3619, 411, 469, 846

Resposta: *ver codigo abaixo*

In [8]:
```
atts = 'SALA DE RECURSO'
print(dc[atts].unique())
dc[atts].hist()
```

```
[ 0 26 11 21 10  6  7 68 28 24 20 15  8 19 25 17 32 14  1 1
8  2 31  4 16
 13  5  9  3 33 36 22 12 29 57 34 27 45 35 42 37 43 23 49 3
0 71 60 44 48
 38 39 79 59 40 41 77 67 46 64 61 65]
```

Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x7f44315b7280>



In [9]:
```
#labels = ['0', '1 a 4', '5 a 9', '10+']
maxsala = np.max(dc[atts].unique())

# definindo intervalos
interv = [-1, 0, 4, 9, maxsala]
# realizar discretizacao e armazenar
salas_rec_d =  pd.cut(dc[atts], bins=interv)#, labels=label
s)
# inserir nova coluna
dc.insert(18, 'SALA_DE_RECURSO_D', salas_rec_d)

# exibir o tipo da coluna
print(dc['SALA_DE_RECURSO_D'].value_counts())
dc[atts].value_counts()
```

```
(-1, 0]    3636
(9, 79]     847
(4, 9]      470
(0, 4]      413
Name: SALA_DE_RECURSO_D, dtype: int64
```

Out[9]:

0	3636
2	111
3	106
6	104
4	103
5	97
7	93
1	93
8	90
9	86
11	78
10	75
14	74
12	64
15	59
13	56
17	52
18	48
16	44
19	40
21	28
20	24
25	19
23	16
26	16
28	15
22	14
24	14
33	11
29	10
30	10
27	10
31	8
35	7
37	7
32	5
40	5
34	4
36	3
48	3
60	3
39	3
57	2
67	2
45	2
38	2
41	1
77	1
59	1
43	1
46	1
42	1
65	1
68	1
61	1

Exercício 7)

Vamos normalizar 3 variáveis: ANOS INICIAIS, ANOS FINAIS e ENSINO MEDIO

A normalização utilizada será diferente para cada uma delas. Utilizaremos

- min-max para ANOS INICIAIS, com a=0, b=1
- norma L-∞ para ANOS FINAIS
- z-score para ENSINO MEDIO

Para isso, codifique funções que recebam uma coluna por parâmetro e retornem um atributo já normalizado

Depois, aplique as funções e crie novas variáveis com os atributos normalizados: INICIAIS_n, FINAIS_n, MEDIO_n.

Após normalização, quais os valores de média e mediana de cada um deles, considerando arrendondamento para 2 casas decimais?

(a) INICIAIS_n: 0.00, 0.00; FINAIS_n: 0.06, 0.00; MEDIO_n: 0.00, 0.17.
(b) INICIAIS_n: 0.07, 0.00; FINAIS_n: 0.16, 0.15; MEDIO_n: 0.00, -0.17.
(c) INICIAIS_n: 0.07, 0.00; FINAIS_n: 0.15, 0.16; MEDIO_n: 0.00, -0.17.
(d) INICIAIS_n: 0.00, 0.00; FINAIS_n: 0.00, 0.16; MEDIO_n: 0.00, 0.17.

Resposta: *ver codigo abaixo*

In [10]:
```
def norm_0_1(att):
    # computa minimo e maximo
    var_min = att.min()
    var_max = att.max()

    # computa normalizacao
    att_norm = (att - var_min) / (var_max-var_min)
    return att_norm

def norm_linf(att):
    # encontra máximo
    norminf = att.max()
    # computa normalizacao
    att_norm = att / norminf
    return att_norm

def norm_zscore(att):
    # computa média e desvio padrao
    var_mean = att.mean()
    var_sigm = att.std()

    # computa normalizacao
    return ((att - var_mean) / var_sigm)
```

In [11]:
```
# INICIAIS_n,  FINAIS_n, MEDIO_n
dc['INICIAIS_n'] = norm_0_1(dc['ANOS INICIAIS'] )
dc['FINAIS_n'] = norm_linf(dc['ANOS FINAIS'] )
dc['MEDIO_n'] = norm_zscore(dc['ENSINO MEDIO'] )
```

In [12]:
```
np.round(dc[['INICIAIS_n',  'FINAIS_n', 'MEDIO_n']].describ
e(),2)
```

Out[12]:

	INICIAIS_n	FINAIS_n	MEDIO_n
count	5366.00	5366.00	5366.00
mean	0.07	0.16	-0.00
std	0.15	0.14	1.00
min	0.00	0.00	-0.94
25%	0.00	0.00	-0.94
50%	0.00	0.15	-0.17
75%	0.07	0.25	0.52
max	1.00	1.00	6.65

In [15]:
```
ds = dc.sample(5, random_state=0)
ds
```

Out[15]:

	CDREDE	DE	CODMUN	MUN	CATEG	COD_ESC	TIPOESC
379	10207	LESTE 1	100	SAO PAULO	1	36997	8
3637	20409	MOGI MIRIM	662	SERRA NEGRA	1	921579	8
4849	20801	ANDRADINA	259	CASTILHO	1	407215	8
2272	20102	SANTOS	335	GUARUJA	2	523604	8
3905	20417	CAMPINAS LESTE	244	CAMPINAS	1	18119	8

5 rows × 25 columns

Exercício 8)

Utilizando as variáveis normalizadas no exercício anterior, compute distâncias entre a escola de COD_ESC (atribuno na coluna de índice 5) cujo valor é 24648

e todas as escolas cujo código da rede (CDREDE) seja 20510, excluindo a de COD_ESC = 24648

Utilize a distancia Euclidiana.

Compare usando vetor de atributos formado por 'INICIAIS_n', 'FINAIS_n' e 'MEDIO_n'.

Observação: deve-se ter cuidado ao usar normalizações distintas como feito nesse exercício para comparar atributos, em particular considerando que o z-score produz valores negativos. Considere esse procedimento apenas a título de exercício com diferentes tipos de normalização e, na dúvida, utilize normalização uniforme entre os atributos.

Qual escola foi a mais próxima (NOMESC) e a respectiva distância (arredondada para 2 casas decimais)?

(a) MANOEL MARTINS, distância 1.6.
(b) EDDA CARDOZO DE SOUZA MARCUSSI, distância 0.18
(c) EDDA CARDOZO DE SOUZA MARCUSSI, distância 105.19
(d) MANOEL MARTINS, distância 0.18

Resposta: *ver codigo abaixo*

In [20]:
```
# implementando a funcao
def dEuclidean(a, b):
    return np.sqrt(np.sum((a - b) ** 2))
```

In [21]:
```
# escolas
rede20510 = dc.loc[~(dc['COD_ESC']==24648) & (dc['CDREDE']==20510), :].copy().reset_index()
size = rede20510.shape[0]

attrs = ['INICIAIS_n',  'FINAIS_n', 'MEDIO_n']

esc24740 = dc.loc[(dc['COD_ESC']==24648), :].copy().reset_index()
a = np.array(esc24740.loc[0,attrs])

attrs_cat = attrs + ['MUN', 'NOMESC']
print(esc24740[attrs_cat])
print(a)

dists = np.empty(size)
for i in range(size):
    b = np.array(rede20510[attrs].loc[i,:])
    dists[i] = dEuclidean(a,b)

# argmin retorna a posicao com o valor mínimo
print(rede20510.loc[np.argmin(dists),attrs_cat])
print("dist = %.2f" % np.round(np.min(dists),2))
# rede20510[attrs_cat]
```

```
   INICIAIS_n  FINAIS_n   MEDIO_n             MUN
NOMESC
0         0.0   0.26209  0.107901  SALES OLIVEIRA  GETULIO
LIMA CAPITAO
[0.0 0.262089552238806 0.10790137317108682]
INICIAIS_n                                    0
FINAIS_n                                0.204776
MEDIO_n                                 0.280619
MUN                        SAO JOAQUIM DA BARRA
NOMESC          EDDA CARDOZO DE SOUZA MARCUSSI
Name: 18, dtype: object
dist = 0.18
```

Exercício 9)

Utilize os atributos 'ENSINO MEDIO', 'ANOS INICIAIS', 'ANOS FINAIS'. Vamos transformá-los por meio da função logaritmica. Para isso:

1. Faça uma cópia da base de dados, e atribua nulo (nan) a todos os valores iguais a zero nesses atributos.
2. Transforme esses atributos utilizando a operação logarítmica e os adicione à base de dados
3. Exiba a matriz de correlação de Pearson entre os atributos originais e os transformados.
4. Mostre o scatterplot entre 'ENSINO MEDIO' e 'ANOS FINAIS', e compare com log(ENSINO MÉDIO) e log(ANOS FINAIS)

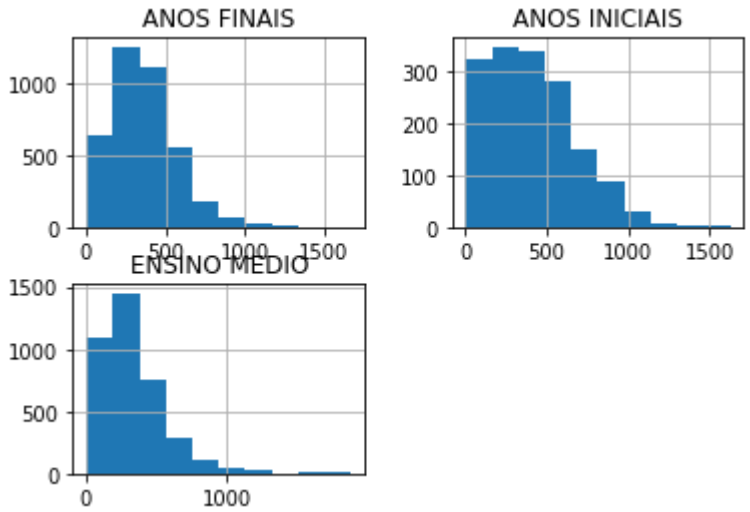
Qual é a correlação, arredondada para duas casas decimais, entre o atributo transformado: log(ENSINO MÉDIO) e os outros dois atributos: log(ANOS INICIAIS) e log(ANOS FINAIS), respectivamente?

(a) 0.45 e 0.61
(b) 0.74 e 0.78
(c) -1 e 1
(d) 0.45 e 0.78

Resposta: *ver codigo abaixo*

In [22]:
```
dc2 = dc.copy()

dc2.loc[dc2['ENSINO MEDIO']==0,'ENSINO MEDIO'] = np.nan
dc2.loc[dc2['ANOS INICIAIS']==0,'ANOS INICIAIS'] = np.nan
dc2.loc[dc2['ANOS FINAIS']==0,'ANOS FINAIS'] = np.nan


atts = ['ENSINO MEDIO', 'ANOS INICIAIS', 'ANOS FINAIS']
dc2[atts].hist()
```

Out[22]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x
7f4431473d60>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x
7f4430d83850>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x
7f4430dadd00>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x
7f4430d631c0>]],
      dtype=object)
```

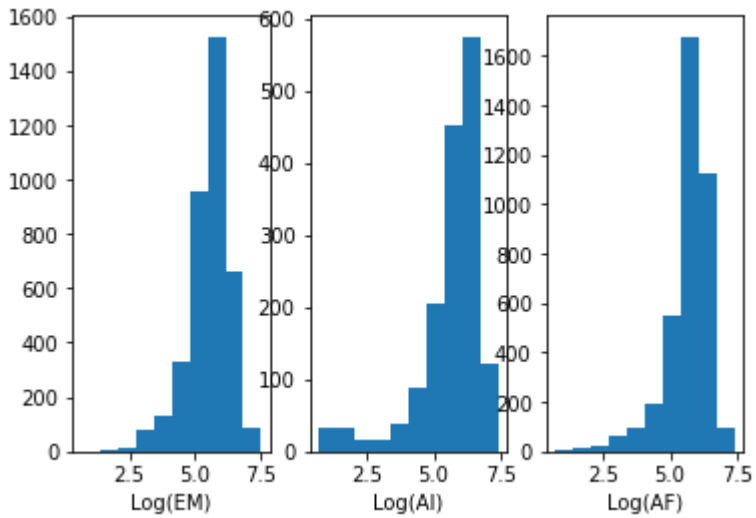


In [23]:
```
# vamos adicionar essas novas variaveis a base
x_E = np.log(np.array(dc2['ENSINO MEDIO'])+1)
x_I = np.log(np.array(dc2['ANOS INICIAIS'])+1)
x_F = np.log(np.array(dc2['ANOS FINAIS'])+1)
x_A = x_E+x_I+x_F
```

In [24]:
```
plt.subplot(131)
h = plt.hist(x_E)
plt.xlabel('Log(EM)')
plt.subplot(132)
h = plt.hist(x_I)
plt.xlabel('Log(AI)')
plt.subplot(133)
h = plt.hist(x_F)
plt.xlabel('Log(AF)')
```

```
/home/maponti/.virtualenvs/mba_ds/lib/python3.8/site-packag
es/numpy/lib/histograms.py:839: RuntimeWarning: invalid val
ue encountered in greater_equal
  keep = (tmp_a >= first_edge)
/home/maponti/.virtualenvs/mba_ds/lib/python3.8/site-packag
es/numpy/lib/histograms.py:840: RuntimeWarning: invalid val
ue encountered in less_equal
  keep &= (tmp_a <= last_edge)
```

Out[24]:
```
Text(0.5, 0, 'Log(AF)')
```



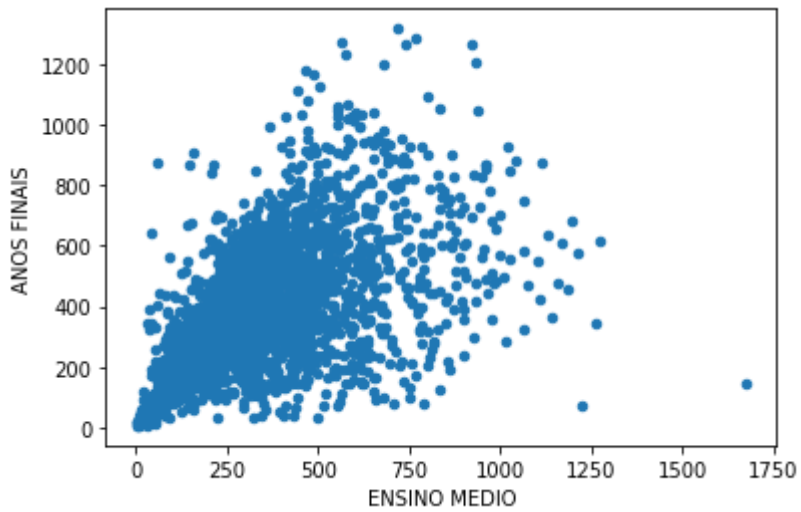
In [25]:
```
dc2['Log_EM'] = x_E
dc2['Log_AI'] = x_I
dc2['Log_AF'] = x_F

#dc2.loc[dc2['Log_EM']==0,'Log_EM'] = np.nan
#dc2.loc[dc2['Log_AI']==0,'Log_AI'] = np.nan
#dc2.loc[dc2['Log_AF']==0,'Log_AF'] = np.nan

atts_t = atts + ['Log_EM', 'Log_AI', 'Log_AF']
np.round(dc2[atts_t].corr("pearson"),2)
```

Out[25]:

	ENSINO MEDIO	ANOS INICIAIS	ANOS FINAIS	Log_EM	Log_AI	Log_AF
ENSINO MEDIO	1.00	0.46	0.61	0.85	0.51	0.57
ANOS INICIAIS	0.46	1.00	0.61	0.57	0.79	0.64
ANOS FINAIS	0.61	0.61	1.00	0.66	0.62	0.85
Log_EM	0.85	0.57	0.66	1.00	0.74	0.78
Log_AI	0.51	0.79	0.62	0.74	1.00	0.81
Log_AF	0.57	0.64	0.85	0.78	0.81	1.00

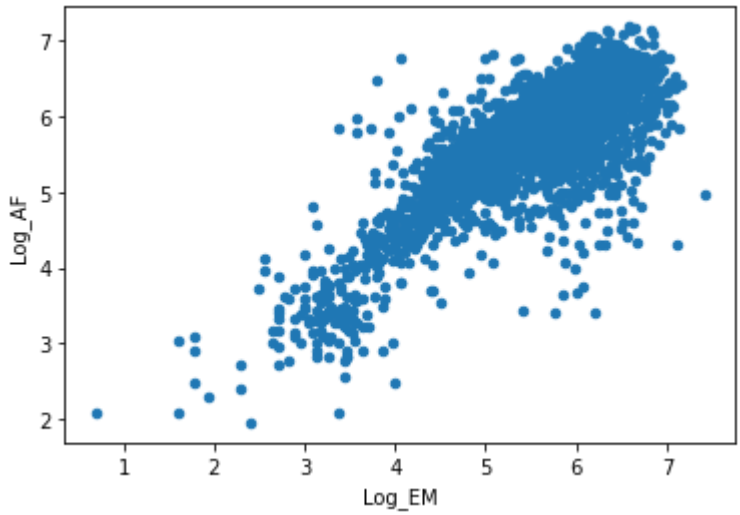
In [26]:
```
dc2.plot.scatter(x='ENSINO MEDIO', y='ANOS FINAIS')
```

Out[26]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4430b51df0>
```



In [27]:
```
dc2.plot.scatter(x='Log_EM', y='Log_AF')
```

Out[27]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f443151a9d0>
```



Exercício 10)

Codifique as variáveis categóricas 'SALA_DE_RECURSO_D' (categórica ordinal) e 'DE' (categórica nominal).

Para a primeira, use números inteiros sequenciais, iniciado por 0 para codificar a variável segundo sua ordenação, sendo que os dois últimos (os dos valores maiores na ordenação) devem ser mapeados para um único código. Gere um novo atributo 'SALA_DE_RECURSO_D_ord'.

Para a segunda, use números inteiros sequenciais, iniciados por 0 para codificar a variável em ordem alfabética, e gere um novo atributo 'DE_cod'

A seguir, use a função `value_counts()` para mostrar a quantidade de cada código e responda abaixo quais os códigos numéricos (após a codificação) que possuem a maior contagem, i.e. são mais frequentes na base de dados, para cada variável nova:

(a) DE_cod: 59 e 66; SALA_DE_RECURSO_D_ord: 0 e 3
(b) DE_cod: 66 e 55; SALA_DE_RECURSO_D_ord: 1 e 3
(c) DE_cod: 66 e 82; SALA_DE_RECURSO_D_ord: 0 e 3
(d) DE_cod: 66 e 82; SALA_DE_RECURSO_D_ord: 0 e 2

Resposta: *ver codigo abaixo*

In [31]:
```
cats_sala = np.sort(dc['SALA_DE_RECURSO_D'].unique())
num_sala = np.arange(cats_sala.shape[0])
# repete o penúltimo código
num_sala[-1] = num_sala[-2]
map_sala = dict(zip(cats_sala, num_sala))

dc['DE_cod'] = dc['DE'].astype("category").cat.codes
dc['SALA_DE_RECURSO_D_ord'] = dc['SALA_DE_RECURSO_D'].map(m
ap_sala)
print("DE_cod:")
dc['DE_cod'].value_counts()
```

```
DE_cod:
```

Out[31]:
```
66    110
82    110
55    105
50    104
15     99
     ...
10     24
31     21
73     20
59     18
62     15
Name: DE_cod, Length: 91, dtype: int64
```

In [30]:
```
print("SALA_DE_RECURSO_D_ord:")
dc['SALA_DE_RECURSO_D_ord'].value_counts()
```

```
SALA_DE_RECURSO_D_ord:
```

Out[30]:
```
0    3636
2    1317
1     413
Name: SALA_DE_RECURSO_D_ord, dtype: int64
```