aula02_resolucao_exercicio

February 27, 2021

1 Aula 02- Resolução dos Exercícios

1.1 Novas perguntas do CEO para vocês

1. Crie uma nova coluna chamada: “house_age”
   - Se o valor da coluna “date” for maior que 2014-01-01 => ‘new_house’
   - Se o valor da coluna “date” for menor que 2014-01-01 => ‘old_house’
2. Crie uma nova coluna chamada: “dormitory_type”
   - Se o valor da coluna “bedrooms” for igual à 1 => ‘studio’
   - Se o valor da coluna “bedrooms” for igual a 2 => ‘apartament’
   - Se o valor da coluna “bedrooms” for maior que 2 => ‘house’
3. Crie uma nova coluna chamada: “condition_type”
   - Se o valor da coluna “condition” for menor ou igual à 2 => ‘bad’
   - Se o valor da coluna “condition” for igual à 3 ou 4 => ‘regular’
   - Se o valor da coluna “condition” for igual à 5 => ‘good’
4. Modifique o TIPO a Coluna “condition” para STRING
5. Delete as colunas: “sqft_living15” e “sqft_lot15”
6. Modifique o TIPO a Coluna “yr_build” para DATE
7. Modifique o TIPO a Coluna “yr_renovated” para DATE
8. Qual a data mais antiga de construção de um imóvel?
9. Qual a data mais antiga de renovação de um imóvel?
10. Quantos imóveis tem 2 andares?
11. Quantos imóveis estão com a condição igual a “regular” ?
12. Quantos imóveis estão com a condição igual a “bad”e possuem “vista para água” ?
13. Quantos imóveis estão com a condição igual a “good” e são “new_house”?
14. Qual o valor do imóvel mais caro do tipo “studio” ?
15. Quantos imóveis do tipo “apartment” foram reformados em 2015 ?
16. Qual o maior número de quartos que um imóveis do tipo “house” possui ?
17. Quantos imóveis “new_house” foram reformados no ano de 2014?

18. Selecione as colunas: “id”, “date”, “price”, “floors”, “zipcode” pelo método:
    - Direto pelo nome das colunas.
    - Pelos Índices.
    - Pelos Índices das linhas e o nome das colunas
    - Índices Booleanos
19. Salve um arquivo .csv com somente as colunas do item 10 ao 17.
20. Modifique a cor dos pontos no mapa de “pink” para “verde-escuro”

2 Resolução

2.1 Import Libraries

[4]:
```
import numpy  as np
import pandas as pd
import seaborn as sns

from matplotlib import pyplot as plt
import plotly.express as px
```

[2]:
```
# Supress Scientific Notation
np.set_printoptions(suppress=True)
pd.set_option('display.float_format', '{:.2f}'.format)
```

2.2 Loading Data

[3]:
```
# loading data into memory
data = pd.read_csv( '../kc_house_data.csv' )

# Garantir que o formato date é um datetime
data['date'] = pd.to_datetime( data['date'], format='%Y-%m-%d' )
```

2.3 1. Crie uma nova coluna chamada: “house_age”

```
- Se o valor da coluna "date" for maior que 2014-01-01 => ‘new_house’
- Se o valor da coluna "date" for menor que 2014-01-01 => ‘old_house’
```

[5]:
```
# Estrategia
# 1. Criar uma coluna nova, preenchida com NA
# 2. Substituir o valor NA, conforma a condicional

data['house_age'] = 'NA'
data.loc[data['date'] > '2014-01-01', 'house_age'] = 'new_house'
data.loc[data['date'] < '2014-01-01', 'house_age'] = 'old_house'
```

2.4 2. Crie uma nova coluna chamada: “dormitory_type”

- Se o valor da coluna "bedrooms" for igual à 1 => ‘studio’
- Se o valor da coluna "bedrooms" for igual a 2 => ‘apartament’
- Se o valor da coluna "bedrooms" for maior que 2 => ‘house’

[6]:
```
# Estrategia:
# 1. Criar uma coluna nova, preenchida com NA
# 2. Percorrer todas as linhas do conjunto de dados. Para cada linha, comparar␣
 ↪a coluna "bedrooms"
# 3. De acordo com a comparação, substituir o NA pelo dado valor
#
data['dormitory_type'] = 'NA'
for i in range( len( data ) ):
    if data.loc[i, 'bedrooms'] == 1:
        data.loc[i, 'dormitory_type'] = 'studio'

    elif data.loc[i, 'bedrooms'] == 2:
        data.loc[i, 'dormitory_type'] = 'apartment'

    elif data.loc[i, 'bedrooms'] > 2:
        data.loc[i, 'dormitory_type'] = 'house'
```

2.5 3. Crie uma nova coluna chamada: “condition_type”

- Se o valor da coluna "condition" for menor ou igual à 2 => ‘bad’
- Se o valor da coluna "condition" for igual à 3 ou 4 => ‘regular’
- Se o valor da coluna "condition" for igual à 5 => ‘good’

[7]:
```
# Estrategia:
# 1. Usar a função apply junto com a lambda para ter acesso a cada linha.
# 2. Em cada linha, comparar a coluna "condition" com a condição dada.
# 3. Aplicar a condição
#
data['condition'] = data['condition'].astype( int )
data['conditional_type'] = data['condition'].apply( lambda x: 'bad' if x <= 2␣
 ↪else 'regular' if (x == 3) | (x == 4) else 'good' )
```

2.6 4. Modifique o TIPO a Coluna “condition” para STRING

[8]:
```
data[['id', 'condition']].dtypes
```

[8]:
```
id           int64
condition    int64
dtype: object
```

[9]:
```
data['condition'] = data['condition'].astype( str )
```

[10]:
```
data[['id', 'condition']].dtypes
```

[10]:
```
id            int64
condition    object
dtype: object
```

2.7 5. Delete as colunas: “sqft_living15” e “sqft_lot15”

[11]:
```
# deletar as colunas
data = data.drop( ['sqft_living15', 'sqft_lot15'], axis=1 )
```

2.8 6. Modifique o TIPO a Coluna “yr_build” para DATE

[12]:
```
data['yr_built'] = pd.to_datetime( data['yr_built'], format='%Y' )
```

2.9 7. Modifique o TIPO a Coluna “yr_renovated” para DATE

[101]:
```
data['yr_renovated'] = data['yr_renovated'].apply( lambda x: pd.to_datetime( x,␣
↪format='%Y') if x > 0 else x )
```

2.10 8. Qual a data mais antiga de construção de um imóvel?

[13]:
```
min_date = data['yr_built'].min().year
print( 'The oldest house is {} years old'.format( min_date ) )
```

```
The oldest house is 1900 years old
```

2.11 9. Qual a data mais antiga de renovação de um imóvel?

[14]:
```
# Estrategia
# 1. Filtrar todas as linhas diferentes de ZERO ( imóveis que nao foram␣
↪reformados )
# 2. Encontrar a data mínima
df = data[data['yr_renovated'] > 0]['yr_renovated']

print( 'The oldest renovated house is {} years old'.format( df.min() ) )
```

```
The oldest renovated house is 1934 years old
```

2.12 10. Quantos imóveis tem 2 andares?

[15]:
```
# Estrategia:
# 1. Filtrar todoso os imóveis com 2 andares
# 2. Contar o número de linhas
#
houses = data[data['floors'] == 2].shape[0]

print( 'The number of houses with 2 floors is: {}'.format( houses ) )
```

```
The number of houses with 2 floors is: 8241
```

2.13 11. Quantos imóveis estão com a condição igual a “regular” ?

[16]:
```
# Estrategia
# 1. Filtrar todos os apartamentos com "condition_type" igual a "regular"
# 2. Contar o número de imóveis sob essa condição
#
houses = data[data['conditional_type'] == 'regular'].shape[0]

print( 'Number of Houses in "regular" condition is: {}'.format( houses ) )
```

```
Number of Houses in "regular" condition is: 19710
```

2.14 12. Quantos imóveis estão com a condição igual a “bad”e possuem “vista para água” ?

[17]:
```
# Estrategia:
# 1. Filtrar as colunas "conditional_type" igual a "bad" e "waterfront" igual a␣
 ↪1
# 2. Contar o número de linhas
houses = data[(data['conditional_type'] == 'bad') & (data['waterfront'] == 1)].
 ↪shape[0]
print( "Number of Houses with water view and bad condition: {}".format( houses␣
 ↪) )
```

```
Number of Houses with water view and bad condition: 2
```

2.15 13. Quantos imóveis estão com a condição igual a “good” e são “new_house”?

[18]:
```
# Estrategia:
# 1. Filtrar as colunas "conditional_type" igual a "good" e "house_age" equals␣
 ↪to "new_house"
# 2. Contar o número de linhas
houses = data[(data['conditional_type'] == 'good') & (data['house_age'] ==␣
 ↪"new_house")].shape[0]
print( "Number of new house with good conditional type is: {}".format( houses )␣
 ↪)
```

```
Number of new house with good conditional type is: 1701
```

2.16 14. Qual o valor do imóvel mais caro do tipo “studio” ?

[19]:
```
# Estrategia:
# 1. Filtrar as colunas "dormitory_type" igual a "studio"
# 2. Encontrar o máximo valor da coluna "price"
max_studio_price = data[data['dormitory_type'] == 'studio']['price'].max()
print( "Most expensive studio house: ${}".format( max_studio_price ) )
```

```
Most expensive studio house: $1247000.0
```

2.17 15. Quantos imóveis do tipo “apartment” foram reformados em 2015 ?

[39]:
```
# Estrategia:
# 1. Filtrar as colunas "dormitory_type" igual a "apartament" e "yr_renovated"
 ↪equals to "2015-01-01"
# 2. Contar o número de linhas
houses = data[(data['dormitory_type'] == 'apartament') &
              (data['yr_renovated'] == 2015 )].shape[0]

print( "Number of reformed house in 2015: {}".format( houses ) )
```

```
Number of reformed house in 2015: 0
```

2.18 16. Qual o maior número de quartos que um imóveis do tipo “house” possui ?

[31]:
```
# Estrategia:
# 1. Filtrar as colunas "dormitory type" igual a "house"
# 2. Encontrar o maior valor da coluna bedrooms
b = data[data['dormitory_type'] == 'house']['bedrooms'].max()

print( "Max number of bedrooms from a house: {}".format( b ) )
```

```
Max number of bedrooms from a house: 33
```

2.19 17. Quantos imóveis “new_house” foram reformados no ano de 2014?

[37]:
```
# Estrategia:
# 1. Filtrar as colunas "dormitory_type" igual a "apartament" e "yr_renovated"
 ↪equals to "2015-01-01"
# 2. Contar o número de linhas
houses = data[(data['house_age'] == 'new_house') &
              (data['yr_renovated'] == 2014 )].shape[0]

print( "Number of reformed house in 2014: {}".format( houses ) )
```

```
Number of reformed house in 2014: 91
```

2.20 18. Selecione as colunas: “id”, “date”, “price”, “floors”, “zipcode” pelo método:

```
-- Direto pelo nome das colunas
-- Pelos índices
-- Pelos índices das linhas e o nome das colunas
-- Índices Booleanos
```

[41]:
```
data.columns
```

[41]:
```
Index(['id', 'date', 'price', 'bedrooms', 'bathrooms', 'sqft_living',
       'sqft_lot', 'floors', 'waterfront', 'view', 'condition', 'grade',
       'sqft_above', 'sqft_basement', 'yr_built', 'yr_renovated', 'zipcode',
       'lat', 'long', 'house_age', 'dormitory_type', 'conditional_type'],
      dtype='object')
```

[45]:
```
### 1. Nome das colunas
df1 = data[['id', 'date', 'price', 'floors', 'zipcode']]

### 2. Índices
df2 = data.iloc[:, [0, 1, 2, 7, 16]]

### 3. Índices das linhas e nome das colunas
df3 = data.loc[:, ['id', 'date', 'price', 'floors', 'zipcode']]
```

2.21 19. Salve um arquivo .csv com somente as colunas do item 10 ao 17.

[47]:
```
#data[['house_age', 'dormitory_type', 'conditional_type']].to_csv( 'exercicio18.
↪csv' )
```