atividadeRegressao

June 13, 2021

[79]:
```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

[4]:
```
from sklearn.datasets import load_boston
```

[5]:
```
data = load_boston(return_X_y=False)
```

[15]:
```
df = pd.DataFrame(
    data=data['data'],
    columns=data['feature_names'])
```

[90]:
```
df.describe()
```

[90]:

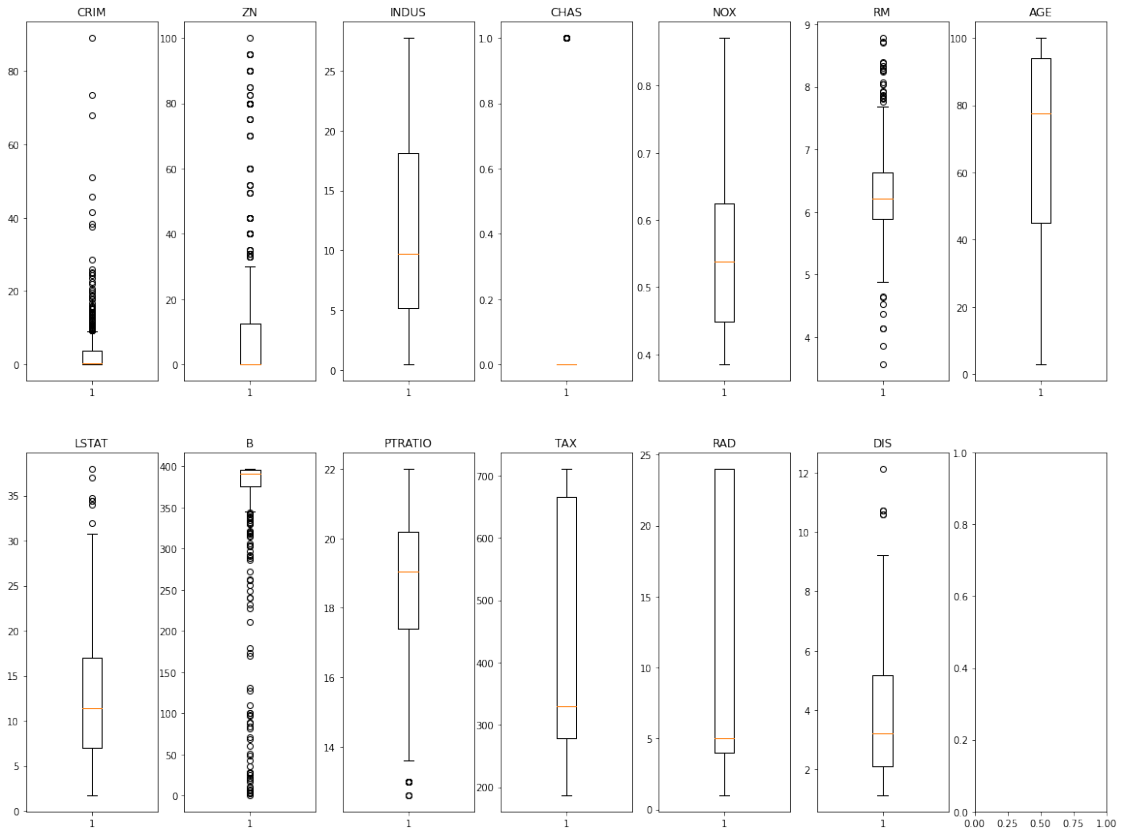
	CRIM	ZN	INDUS	CHAS	NOX	RM	\
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	
75%	3.677083	12.500000	18.100000	0.000000	0.624000	6.623500	
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	

	AGE	DIS	RAD	TAX	PTRATIO	B	\
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	
mean	68.574901	3.795043	9.549407	408.237154	18.455534	356.674032	
std	28.148861	2.105710	8.707259	168.537116	2.164946	91.294864	
min	2.900000	1.129600	1.000000	187.000000	12.600000	0.320000	
25%	45.025000	2.100175	4.000000	279.000000	17.400000	375.377500	
50%	77.500000	3.207450	5.000000	330.000000	19.050000	391.440000	
75%	94.075000	5.188425	24.000000	666.000000	20.200000	396.225000	
max	100.000000	12.126500	24.000000	711.000000	22.000000	396.900000	

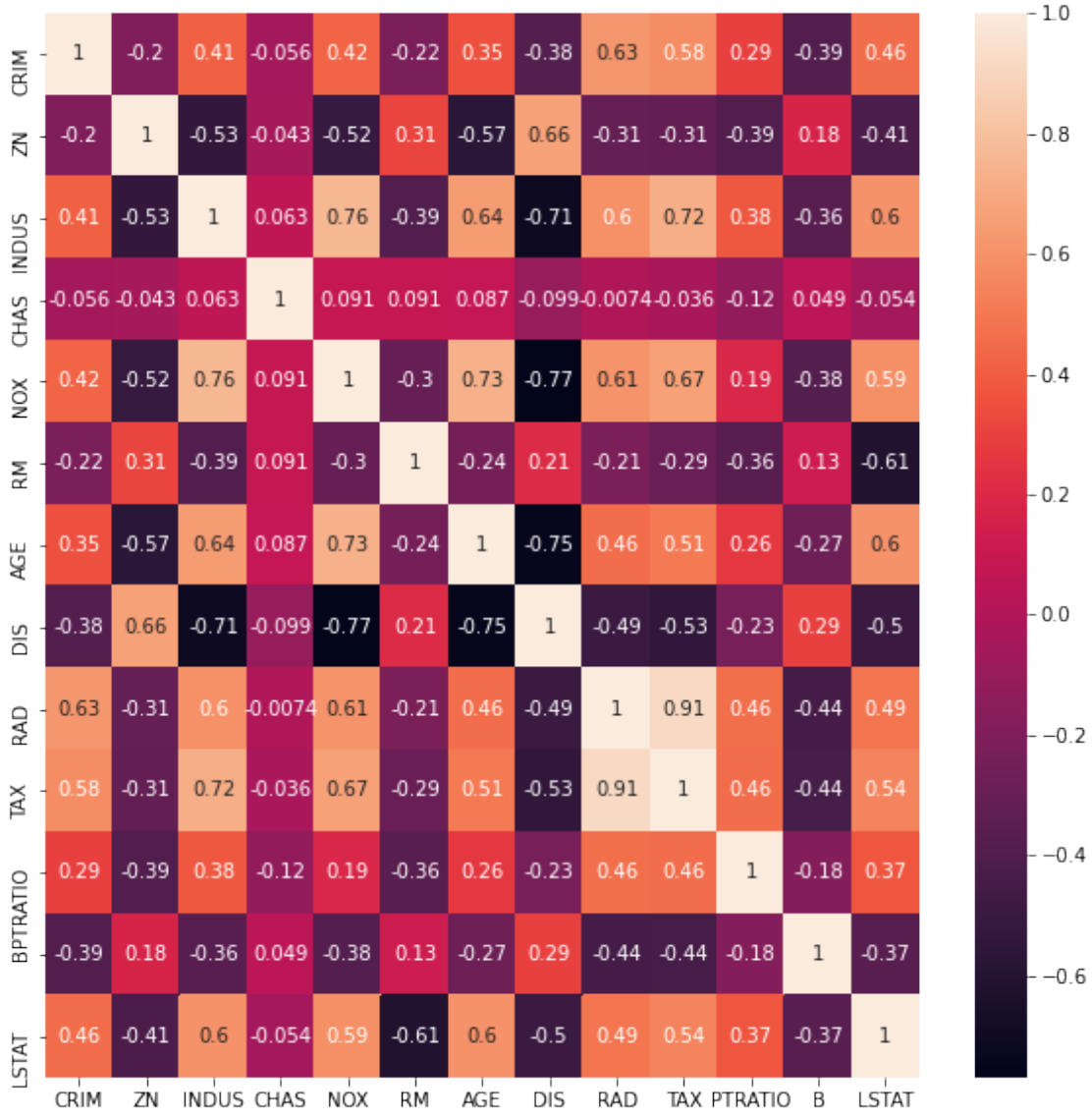
	LSTAT

```
count  506.000000
mean    12.653063
std      7.141062
min      1.730000
25%      6.950000
50%     11.360000
75%     16.955000
max     37.970000
```

[100]:
```
plt.rcParams["figure.figsize"] = (20,15)
fig, axs = plt.subplots(2,7)
for i in range(data['data'].shape[1]):
    if i < 7:
        axs[0,i].boxplot([data['data'][:,i]])
        axs[0,i].set_title(data['feature_names'][i])
    else:
        i_ = 5-i
        axs[1,i_].boxplot([data['data'][:,i]])
        axs[1,i_].set_title(data['feature_names'][i])
plt.show()
```



[16]:
```
import seaborn as sns
plt.rcParams["figure.figsize"] = (10,10)
sns.heatmap(df.corr(), annot=True)
plt.show()
```



[52]:
```
descriptions = {}
for feature in data['feature_names']:
    pos = data['DESCR'].find(f'- {feature}')
    end = data['DESCR'][pos:].find('\n')
    print(data['DESCR'][pos:pos+end])
    descriptions[feature] = data['DESCR']
```

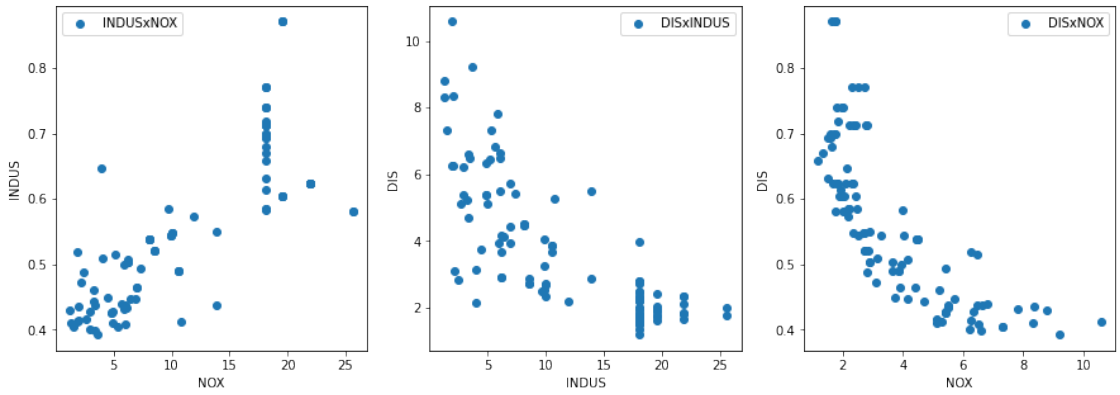
- CRIM   per capita crime rate by town
- ZN   proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS   proportion of non-retail business acres per town
- CHAS   Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX   nitric oxides concentration (parts per 10 million)
- RM   average number of rooms per dwelling
- AGE   proportion of owner-occupied units built prior to 1940
- DIS   weighted distances to five Boston employment centres
- RAD   index of accessibility to radial highways
- TAX   full-value property-tax rate per $10,000
- PTRATIO pupil-teacher ratio by town
- B   1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- LSTAT   % lower status of the population

[58]:
```
correlations = []
for index, row in df.corr().iterrows():
    for corr in row.items():
        if (corr[0], index) in correlations: continue
        if corr[1] > 0.7 and corr[0] != index:
            print(f'Correlação POSITIVA forte entre {index} e {corr[0]}:␣
 ↪{corr[1]}')
            correlations.append((index,corr[0]))
        elif corr[1] < -0.7:
            print(f'Correlação NEGATIVA forte entre {index} e {corr[0]}:␣
 ↪{corr[1]}')
            correlations.append((index,corr[0]))
```

```
Correlação POSITIVA forte entre INDUS e NOX: 0.7636514469209145
Correlação NEGATIVA forte entre INDUS e DIS: -0.7080269887427683
Correlação POSITIVA forte entre INDUS e TAX: 0.7207601799515441
Correlação POSITIVA forte entre NOX e AGE: 0.7314701037859579
Correlação NEGATIVA forte entre NOX e DIS: -0.7692301132258261
Correlação NEGATIVA forte entre AGE e DIS: -0.7478805408686316
Correlação POSITIVA forte entre RAD e TAX: 0.9102281885331822
```

[141]:
```
plt.rcParams["figure.figsize"] = (15,5)
fig, axs = plt.subplots(1,3)
axs[0].scatter(x_test['INDUS'],x_test['NOX'], label='INDUSxNOX')
axs[0].legend()
axs[0].set_ylabel('INDUS')
axs[0].set_xlabel('NOX')
axs[1].scatter(x_test['INDUS'],x_test['DIS'], label='DISxINDUS')
axs[1].legend()
axs[1].set_ylabel('DIS')
axs[1].set_xlabel('INDUS')
axs[2].scatter(x_test['DIS'],x_test['NOX'], label='DISxNOX')
axs[2].legend()
```

```
axs[2].set_ylabel('DIS')
axs[2].set_xlabel('NOX')
plt.show()
```



[63]:
```
import scipy.stats as stats
fvalue, pvalue = stats.
 ↪f_oneway(df['CRIM'],df['ZN'],df['INDUS'],df['CHAS'],df['NOX'],
                                df['RM'],df['AGE'],df['DIS'],df['RAD'],df['TAX'],
                                df['PTRATIO'],df['B'],df['LSTAT'])
print(fvalue, pvalue)
```

```
3369.851016356049 0.0
```

Temos um p-valor significante (<0.05), logo podemos rejeitar a hipótese nuila de que não há diferença entre as médias, ou seja, há uma diferença significativa.

[82]:
```
linear_regression_model = LinearRegression()
```

[83]:
```
x_train, x_test, y_train, y_test = train_test_split(df.iloc[:, :], pd.
 ↪DataFrame(data['target']),
                                                    test_size=0.2,␣
 ↪random_state=3)
linear_regression_model.fit(x_train, y_train)
```

[83]:
```
LinearRegression()
```

[84]:
```
print(f'Coeficientes do modelo:\n{linear_regression_model.coef_}')
```

```
Coeficientes do modelo:
[[-1.23897571e-01  4.81822924e-02 -4.74497796e-02  3.36938950e+00
  -1.56635488e+01  3.59419367e+00 -9.33206067e-03 -1.47089101e+00
   3.05053544e-01 -1.08397039e-02 -9.08791339e-01  1.00352939e-02
  -4.77714677e-01]]
```

[85]:
```
from sklearn.metrics import mean_squared_error, r2_score
y_pred = linear_regression_model.predict(x_test)
print(f'Erro quadrático médio das predições: {mean_squared_error(y_test,␣
 ↪y_pred)}')
print(f'Coeficiente de determinação: {r2_score(y_test, y_pred)}')
```

```
Erro quadrático médio das predições: 16.943073013833807
Coeficiente de determinação: 0.7952617563243856
```