Introdução à Ciência dos Dados
Ciência da Computação – UFV/Florestal

Aula Prática 08
Regressão Linear

Objetivo: extrair correlação entre atributos de dados reais por meio de regressão
Pré-requisitos: linguagem de programação Python, Linux, estatística
Meta: ao final da prática, o aluno será capaz de preparar dados e extrair correlações interessantes a partir da aplicação de técnicas de regressão linear

Roteiro

Caso ainda não esteja instalado, instalar módulo Scikit-learn:

Vamos utilizar como exemplo um conjunto de dados que contém várias características e os preços de imóveis da cidade de Boston. O objetivo da regressão é gerar um modelo de regressão que permita prever o preço de um imóvel com base em suas características.

- Importar as bibliotecas a serem utilizadas

```
import numpy as np
import pandas as pd
import scipy.stats as stats
import matplotlib.pyplot as plt
import sklearn
import seaborn as sns
```

- O conjunto de dados está disponível diretamente na biblioteca scikit-learn

```
# Bouston Housing data set: disponível no scikitlearn
from sklearn.datasets import load_boston
boston = load_boston()
```

```
# o objeto boston é um dicionário
print boston.keys()
print boston.data.shape
print boston.feature_names
print boston.DESCR
```

```
['data', 'feature_names', 'DESCR', 'target']
(506, 13)
['CRIM' 'ZN' 'INDUS' 'CHAS' 'NOX' 'RM' 'AGE' 'DIS' 'RAD' 'TAX' 'PTRATIO'
 'B' 'LSTAT']
Boston House Prices dataset
===========================

Notes
------
Data Set Characteristics:

    :Number of Instances: 506

    :Number of Attributes: 13 numeric/categorical predictive

    :Median Value (attribute 14) is usually the target

    :Attribute Information (in order):
        - 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
        - MEDV     Median value of owner-occupied homes in $1000's
```

- É preciso converter os dados para o formato de DataFrame do Pandas

```
# converter os dados de Boston para um DataFrame do pandas
df = pd.DataFrame(boston.data, columns=boston.feature_names)
df.head()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33

```
# boston.target contém os preços dos imóveis, que são os valores
# a serem previstos.
# Colocar o preço como uma coluna adicional do data frame
df['PRICE'] = boston.target
df.head()
```

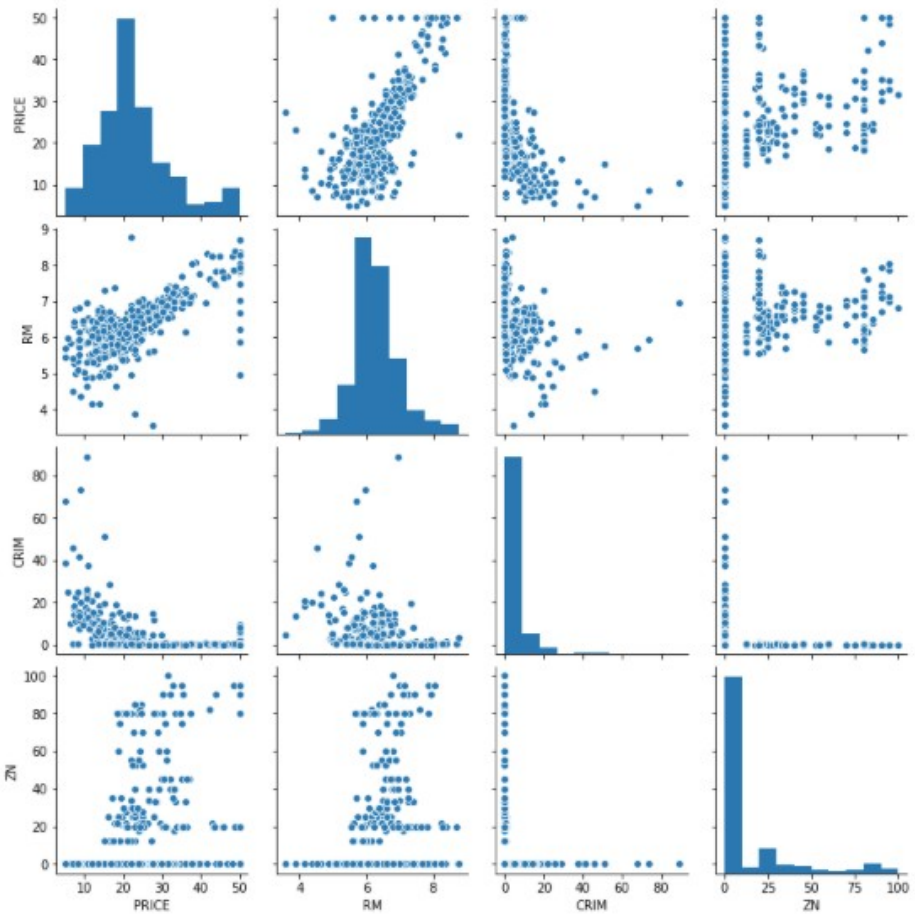
	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2

- Os preços dos imóveis estão no campo *target* do conjunto original. Então, vamos colocar os preços como uma nova coluna (lembre, o preço é a variável dependente que queremos estimar utilizando regressão)

- Primeiro, vamos avaliar a relação entre algumas variáveis e o preço do imóvel

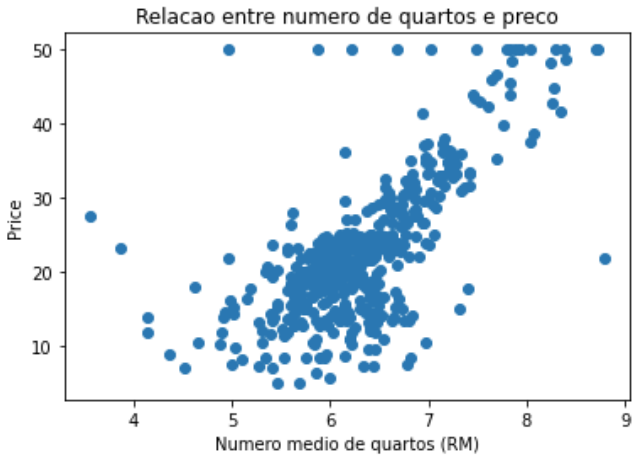
```
# Antes de se criar um modelo de regressão, é importante avaliar as correlações
# entre todas as possíveis variáveis
sns.pairplot(df[['PRICE','RM','CRIM','ZN']])
```

```
<seaborn.axisgrid.PairGrid at 0x7fa128039cc0>
```



- Vamos olhar individualmente para o RM (Número de Quartos)

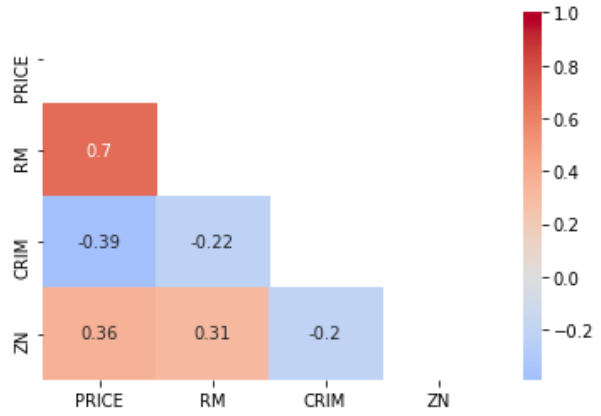
```
# Verificar a relação entre o número de quartos (RM) e o preço
plt.scatter(df.RM, df.PRICE)
plt.xlabel("Numero medio de quartos (RM)")
plt.ylabel("Price")
plt.title("Relacao entre numero de quartos e preco")
plt.show()
```



- Vamos visualizar a correlação par-a-par entre as variáveis, usando o coeficiente de Pearson em uma matriz de correlação

```
# Uma medida importante para verificar a relação entre as variáveis
# é o coeficiente de pearson
cols=['PRICE','RM','CRIM','ZN']
Var_Corr = df[cols].corr()
matrix = np.triu(df[cols].corr())
sns.heatmap(Var_Corr, xticklabels=Var_Corr.columns, yticklabels=Var_Corr.columns,\
            annot=True,center= 0, cmap= 'coolwarm',mask=matrix)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa1343fac18>
```



- Usar regressão linear para calcular o coeficiente da reta representando a relação entre RM e PRICE (regressão simples, com uma única variável independente)

```
# Criar um modelo de regressão utilizando RM (independente) e PRICE (dependente)
from sklearn.linear_model import LinearRegression
X = df.RM
Y = df.PRICE
lm = LinearRegression()
lm.fit(X.values.reshape(-1,1),Y)
print('Coeficiente estimado: ', lm.coef_)
print('R2 (score): ', lm.score(X.values.reshape(-1,1),Y))
```

```
('Coeficiente estimado: ', array([ 9.10210898]))
('R2 (score): ', 0.48352545599133429)
```

- Ou seja, a cada aumento de uma unidade em RM (número de quartos), o preço aumenta em 9.10 o preço. O R2 (coeficiente de determinação) é de 0.48. Ou seja, 52% da variação do preço não é explicada pelo RM.

- Agora vamos fazer a regressão multi-variada. Para isso, vamos separar os atributos independentes (X) e o atributo dependente preço (Y), e criar uma instância da classe LinearRegression

```
# Criar o modelo de regressão utilizando como Y (variável dependente) o preço,
# e X (variáveis independentes) como todos os outros atributos.
# Utilizar a calsse LinearRegression da biblioteca sklearn
from sklearn.linear_model import LinearRegression
X = df.drop('PRICE',axis = 1)
Y = df.PRICE
lm = LinearRegression()
lm.fit(X,Y)
print('R2 (score): ', lm.score(X.values,Y))
```

```
R2 (score):  0.7406426641094095
```

- Mostrar os coeficientes estimados para todas as variáveis independentes

```
# Mostrar os coeficientes da regressão.
print("Intercept %.3f " % lm.intercept_)

coeff_df = pd.DataFrame(lm.coef_, X.columns, columns=['Coefficient'])
coeff_df
```

```
Intercept 36.459
```

	Coefficient
CRIM	-0.108011
ZN	0.046420
INDUS	0.020559
CHAS	2.686734
NOX	-17.766611
RM	3.809865
AGE	0.000692
DIS	-1.475567
RAD	0.306049
TAX	-0.012335
PTRATIO	-0.952747
B	0.009312
LSTAT	-0.524758

Equação: PRICE = 36.49 + CRIM * -0.10 + ZN * 0.04 + …. + LSTAT * -0.52

- Vamos prever o preço de um imóvel. Como o parâmetro para *predict* é um DataFrame, devemos converter o primeiro elemento de X em um objeto 2D.

```
1 # Prever preço de um imóvel em que não se sabe o preço, mas se conhece
2 # as outras características. Vamos prever o preço do primeiro imóvel
3 lm.predict(X.loc[1].values.reshape(1,-1))
```

```
array([ 25.0298606])
```

- Vamos utilizar outra biblioteca para obter mais resultados estatísticos

```
# Podemos utilizar uma outra biblioteca com mais detalhes estatísticos
import statsmodels.api as sm

X = df.drop('PRICE',axis = 1)
y = df.PRICE
X = sm.add_constant(X)
results = sm.OLS(y,X).fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  PRICE   R-squared:                       0.741
Model:                            OLS   Adj. R-squared:                  0.734
Method:                 Least Squares   F-statistic:                     108.1
Date:                Mon, 05 Oct 2020   Prob (F-statistic):          6.72e-135
Time:                        22:26:36   Log-Likelihood:                -1498.8
No. Observations:                 506   AIC:                             3026.
Df Residuals:                     492   BIC:                             3085.
Df Model:                          13
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         36.4595      5.103      7.144      0.000      26.432      46.487
CRIM          -0.1080      0.033     -3.287      0.001      -0.173      -0.043
ZN             0.0464      0.014      3.382      0.001       0.019       0.073
INDUS          0.0206      0.061      0.334      0.738      -0.100       0.141
CHAS           2.6867      0.862      3.118      0.002       0.994       4.380
NOX          -17.7666      3.820     -4.651      0.000     -25.272     -10.262
RM             3.8099      0.418      9.116      0.000       2.989       4.631
AGE            0.0007      0.013      0.052      0.958      -0.025       0.027
DIS           -1.4756      0.199     -7.398      0.000      -1.867      -1.084
RAD            0.3060      0.066      4.613      0.000       0.176       0.436
TAX           -0.0123      0.004     -3.280      0.001      -0.020      -0.005
PTRATIO       -0.9527      0.131     -7.283      0.000      -1.210      -0.696
B              0.0093      0.003      3.467      0.001       0.004       0.015
LSTAT         -0.5248      0.051    -10.347      0.000      -0.624      -0.425
==============================================================================
Omnibus:                      178.041   Durbin-Watson:                   1.078
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              783.126
Skew:                           1.521   Prob(JB):                    8.84e-171
Kurtosis:                       8.281   Cond. No.                     1.51e+04
==============================================================================
```

- Podemos ver que “INDUS” e “AGE” apresentaram um valor-p alto, o o intervalo de confiança inclui o zero. Então, podemos remover essas duas variáveis, e vamos ver que o resultado não muda.

```python
# AGE e INDUS estão com p-valor muito alto, e o intervalo de confiança inclui o zero.
# Então, vamos remover essas variáveis para ver os resultados
X = df.drop('PRICE',axis = 1)
X = X.drop('AGE',axis = 1)
X = X.drop('INDUS',axis = 1)
y = df.PRICE
X = sm.add_constant(X)
results = sm.OLS(y,X).fit()
print(results.summary())
# Manteve o mesmo R^2
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  PRICE   R-squared:                       0.741
Model:                            OLS   Adj. R-squared:                  0.735
Method:                 Least Squares   F-statistic:                     128.2
Date:                Mon, 05 Oct 2020   Prob (F-statistic):          5.54e-137
Time:                        22:28:36   Log-Likelihood:                -1498.9
No. Observations:                 506   AIC:                             3022.
Df Residuals:                     494   BIC:                             3072.
Df Model:                          11
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         36.3411      5.067      7.171      0.000      26.385      46.298
CRIM          -0.1084      0.033     -3.307      0.001      -0.173      -0.044
ZN             0.0458      0.014      3.390      0.001       0.019       0.072
CHAS           2.7187      0.854      3.183      0.002       1.040       4.397
NOX          -17.3760      3.535     -4.915      0.000     -24.322     -10.430
RM             3.8016      0.406      9.356      0.000       3.003       4.600
DIS           -1.4927      0.186     -8.037      0.000      -1.858      -1.128
RAD            0.2996      0.063      4.726      0.000       0.175       0.424
TAX           -0.0118      0.003     -3.493      0.001      -0.018      -0.005
PTRATIO       -0.9465      0.129     -7.334      0.000      -1.200      -0.693
B              0.0093      0.003      3.475      0.001       0.004       0.015
LSTAT         -0.5226      0.047    -11.019      0.000      -0.616      -0.429
==============================================================================
Omnibus:                      178.430   Durbin-Watson:                   1.078
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              787.785
Skew:                           1.523   Prob(JB):                    8.60e-172
Kurtosis:                       8.300   Cond. No.                     1.47e+04
==============================================================================
```

Atividades

1. Utilizar o conjunto de dados weatherHistory.csv, que contém o histórico de variáveis relacionadas ao clima da cidade de Szede, Hungria, entre 2006 e 2016.

a) Primeiramente, faça uma análise exploratória dos dados, entendendo as variáveis e suas características. Crie gráficos par-a-par das variáveis para tentar identificar possíveis relações lineares entre elas;

b) Qual a equação linear e o coeficiente de determinação que representa a temperatura em termos da umidade?

c) Qual a equação linear e o coeficiente de determinação que representa a temperatura em termos da umidade e velocidade do vento?

d) Qual a equação linear e o coeficiente de determinação que representa a temperatura aparente em termos da temperatura, da velocidade do vento e da umidade?

e) Quais variáveis do dataset você indica para serem utilizadas para prever a umidade? Explique sua resposta.