Eduardo Bonet
3˚ Meetup Data Science Floripa

Python ~~(vs)~~ e R

Um Exemplo Prático

Bio

Bonet

Engenharia de Controle e Automação

"Mestrando" em Ciência da Computação

Full Stack, Mobile, Data Science

github.com/ebonet

Objetivos

Python ~~(vs)~~ **E** R : Não é uma competição

Resolver um problema simples com as duas linguagens.

Analisar algumas diferenças na hora do uso.

Código disponível em: https://github.com/ebonet/pythonandr

O Problema

Criar um modelo para preço de aluguel mensal em Floripa

Fonte de dados: http://api.vivareal.com/#!/listings

Etapas:

- Aquisição
- Processamento e Exploração
- Criação de modelo
- Visualização

Setup

Python 2.7.10

- IDE: PyCharm
- Libs: SciPy, NumPY, Pandas, statsmodels

R 3.1.13

- IDE: RStudio
- Libs: Rmisc, rjson, ggplot2, png

Aquisição (prepare.py)

- Dados em JSON, query paginada

```python
def perform_paged_request(page):
    url = "http://api.vivareal.com/api/1.0/locations/listings"
    params = {
        "apiKey": "183d98b9-fc81-4ef1-b841-7432c610b36e",
        "exactLocation": False,
        "currency": "BRL",
        "business": "RENTA",
        "listingType": "APART",
        "listingUse": "RESIDENCIAL",
        "rankingId":0,
        "locationIds": "BR>Santa Catarina>NULL>Florianopolis",
        "maxResults": 40,
        "page": page
    }

    response = requests.get(url, params=params)
    return json.loads(response.content)
```

```python
def load_data(out, max  = 100):

    current_page = 1

    data = perform_paged_request(current_page)
    print data
    listings = []

    while current_page < max and data and data["listings"]:
        print current_page
        listings = listings+data["listings"]
        current_page += 1
        data = perform_paged_request(current_page)


    json.dump({"rentals": listings}, open(out, "w"))


load_data("rentals.json")
```

Aquisição (prepare.R)

```
 8  buildUrl <- function(page) paste(c("http://api.vivareal.com/api/1.0/locations/listings?",
 9        "apiKey=183d98b9-fc81-4ef1-b841-7432c610b36e&exactLocation=FALSE",
10        "&currency=BRL&business=RENTA&listingType=APART&listingUse=RESIDENCIAL",
11        "&rankingId=0&locationIds=BR%3ESanta%20Catarina%3ENULL%3EFlorianopolis",
12        "&maxResults=40&page=", page), collapse="")
13
14  loadData <- function(max=100) {
15
16    all <- c()
17
18    for (i in 0:max) {
19      d <- fromJSON(file=buildUrl(i))
20
21      if(length(d$listings) ==0 )
22        break
23
24      all <- c(all, d$listings)
25    }
26
27    all
28  }
29
30  data <- loadData()
```

Conversão para CSV (prepare.py)

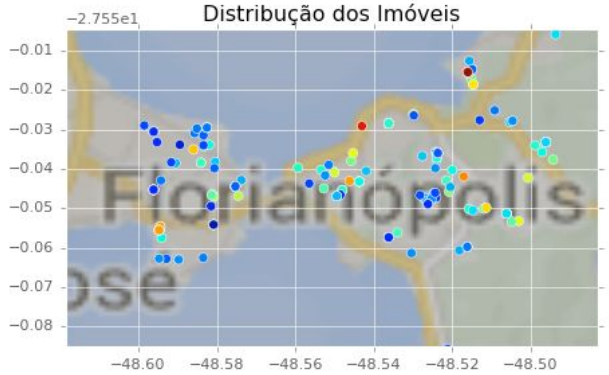
```
46  def toCsv(inputF, outputF):
47
48      rentals = json.load(inputF, encoding="utf-8")["rentals"]
49      keys = ["propertyId", "rentPrice", "area", "bathrooms", "rooms",
50              "garages", "latitude", "longitude", "address", "suites",
51              "rentPeriodId", "condominiumPrice", "iptu"]
52
53      writer = csv.DictWriter(outputF, encoding="utf-8", fieldnames=keys, quoting=csv.QUOTE_ALL)
54      writer.writeheader()
55      writer.writerows([{k:rental[k] for k in keys} for rental in rentals ])
```

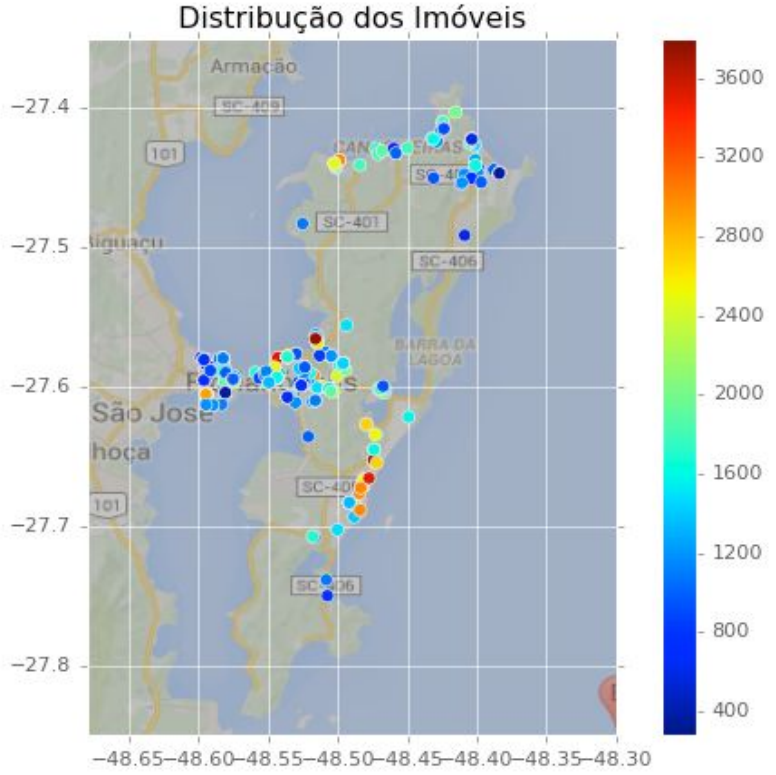
Conversão para CSV (prepare.r)

```r
json2df <- function(data) {
  keys <- c("propertyId", "rentPrice", "area", "bathrooms", "rooms",
            "garages", "latitude", "longitude", "address", "suites",
            "rentPeriodId", "condominiumPrice", "iptu")

  # Black Magic to transform the list into a data frame
  k<- lapply(data, function(d){as.character.default(d[keys])})
  d <- as.data.frame(do.call(rbind, lapply(k, rbind)))
  d[d=="NULL"] <- NA
  names(d) <- keys
  d
}

dt <- json2df(data)
write.csv(dt, file="rentals.csv", row.names=F)

```

Exploração (visualization.py)

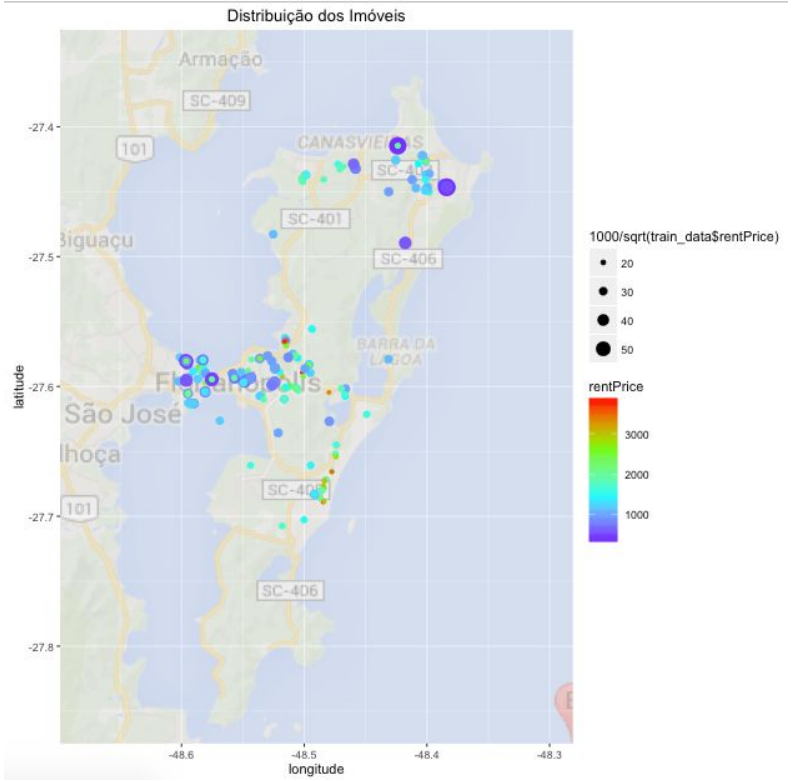
```
33  plt.figure();
34  im = matplotlib.image.imread('flnp2.png');
35  im[:,:,-1] = 0.7;
36  plt.imshow(im, extent = [-48.68, -48.3, -27.85, -27.35]);
37  plt.scatter(train_data.longitude,
38              train_data.latitude,
39              c = train_data.rentPrice);
40
41  plt.colorbar()
42  plt.axis([-48.68, -48.3, -27.85, -27.35]);
43  plt.title(u"Distribuição dos Imóveis");
44
```





Exploração (visualization.R)

```
56  img <- readPNG("flnp2.png")
57  img[,,4] <- 0.3
58  ggplot(train_data, aes(x=longitude, y=latitude, color = rentPrice) ) +
59    scale_colour_gradientn(
60      colours=rev(rainbow(4)), guide = "colourbar") +
61    annotation_custom(
62      rasterGrob(img, width=unit(1,"npc"), height=unit(1,"npc")),
63                 -Inf, Inf, -Inf, Inf) +
64    coord_cartesian(xlim = c(-48.68, -48.3), ylim = c(-27.85, -27.35)) +
65    scale_radius(range=c(1,6)) +
66    geom_point(aes(size = 1000/ sqrt(train_data$rentPrice)))+
67    ggtitle("Distribuição dos Imóveis")
```



Modelo

Restrições

- -49 < longitude < -48
- -28 < latitude < -27
- Aluguel Mensal
- Aluguel < R$ 4000,00
- Número de quartos > 0
- Número de banheiros > 0

Variáveis

- latitude
- longitude
- Número de quartos
- Número de banheiros
- Número de vagas
- Area

Criação do modelo (predict.py)

```
# Load libraries
import pandas as pd, numpy as np, statsmodels.formula.api as sm

# Read data
data = pd.read_csv("rentals.csv")

# Clean data
data =data[(data.rentPeriodId=="MON") & (data.latitude.between(-28, -27)) & (data.longitude.between(-49, -48))
           & (data.bathrooms > 0) & (data.rooms > 0) & (data.rentPrice <4000) ]

data = data[[u'propertyId', u'area', u'rentPrice', u'latitude', u'longitude',u'garages',u'rooms', u'bathrooms']]
data.describe()

# Explore using pandas
# BoxPlot: train_data.rentPrice.plot('box')
# Histogram: train_data.rentPrice.plot('histogram')

# Split data
np.random.seed(42)
rds = np.random.random(len(data))
train_data, test_data = data[rds < 0.7], data[rds >= 0.7]

# Simple prediction

result = sm.ols(formula="rentPrice ~ rooms + bathrooms + latitude + longitude + garages + area", data = train_data).fit()
result.summary()
```

Criação do modelo (predict.r)

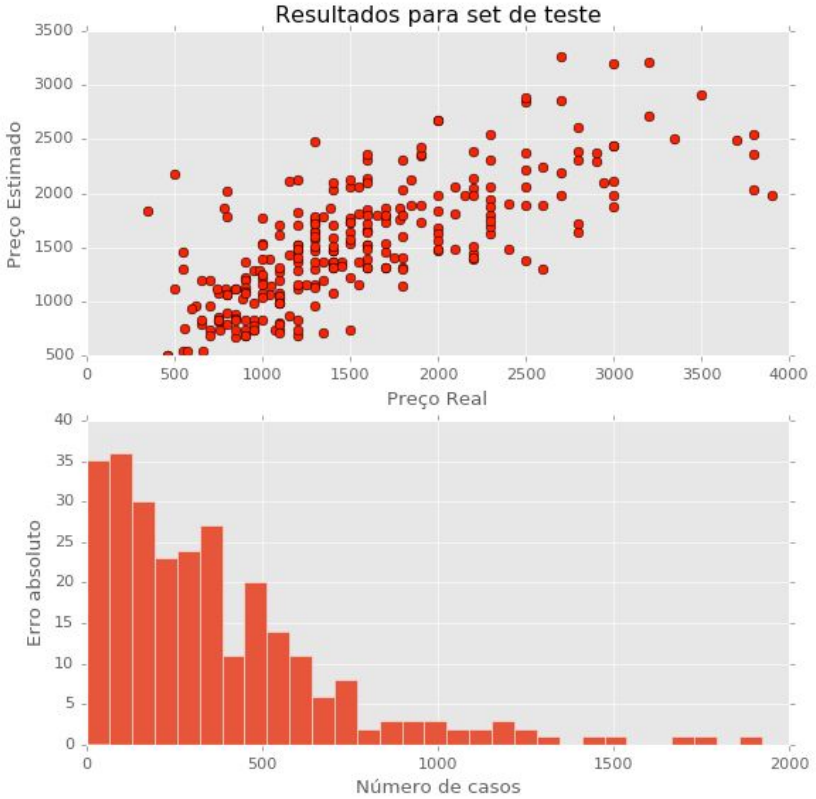
```r
data <- read.csv("rentals.csv")

data <- data[(data$rentPeriodId=="MON") & data$latitude> -28 & data$latitude < -27 & data$longitude > -49
             & data$longitude < -48 & data$bathrooms > 0 & data$rooms > 0 & data$rentPrice < 4000 &
               !is.na(data$area),  ]

train_indexes <- runif(nrow(data)) < 0.7

train_data <- data[train_indexes,]
l <- lm(data = train_data, formula = rentPrice ~  rooms + bathrooms + latitude + longitude + garages + area)

test_data <- data[!train_indexes,]

test_data <- data[!train_indexes,]
predicted <- predict(l, test_data)

source('visualization.R')
```

Resultados (predict.py)

```
26  p = result.predict(test_data)
27  # Plot results
28  plt.figure()
29  plt.subplot(211)
30  plt.plot(test_data.rentPrice, p, 'ro')
31  plt.xlabel(u"Preço Real")
32  plt.ylabel(u"Preço Estimado")
33  plt.title(u"Resultados para set de teste")
34
35  plt.subplot(212)
36  plt.hist(np.abs(test_data.rentPrice- p), bins=30 )
37  plt.xlabel(u"Número de casos")
38  plt.ylabel(u"Erro absoluto")
```

OLS Regression Results

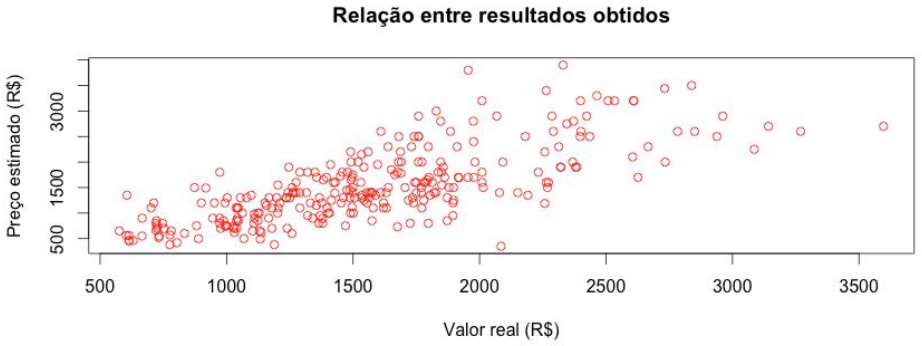
Dep. Variable:	rentPrice	R-squared:	0.608
Model:	OLS	Adj. R-squared:	0.605
Method:	Least Squares	F-statistic:	202.3
Date:	Tue, 12 Apr 2016	Prob (F-statistic):	5.24e-130
Time:	00:54:20	Log-Likelihood:	-4939.2
No. Observations:	658	AIC:	9890.
Df Residuals:	652	BIC:	9917.
Df Model:	5		
Covariance Type:	nonrobust		

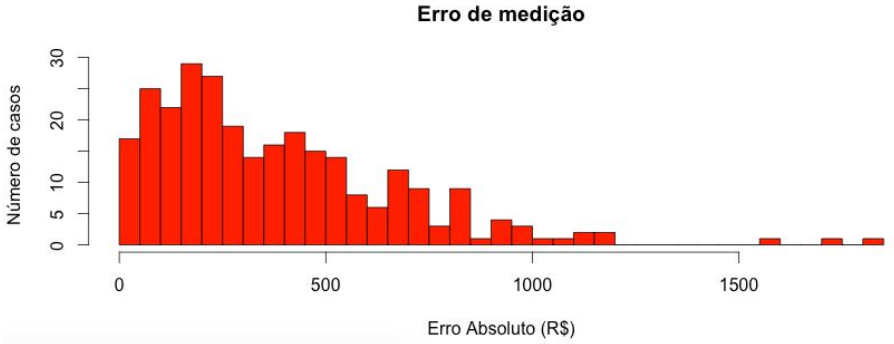


Resultados (predict.R)

```
Residual standard error: 476.8 on 549 degrees of freedom
Multiple R-squared:  0.5536,    Adjusted R-squared:  0.5495
F-statistic: 136.2 on 5 and 549 DF,  p-value: < 2.2e-16
```

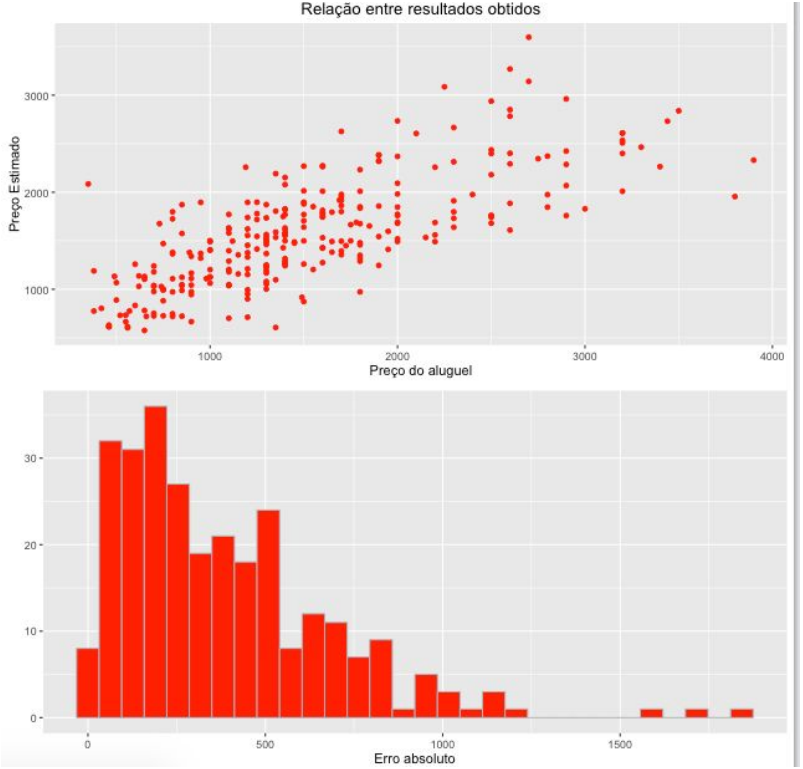
```r
# plot using pure R
par(mfrow=c(2,1))
par(new = F)
plot(test_data$rentPrice ~ predicted,
     xlab = "Valor real (R$)",
     ylab = "Preço estimado (R$)",
     col = 'red',
     main = "Relação entre resultados obtidos")

hist(abs(test_data$rentPrice - predicted),
     breaks = 30,
     c='red',
     xlab = "Erro Absoluto (R$)",
     ylab = "Número de casos",
     main="Erro de medição")
```





Resultados (R + ggplot2)

```
27  library(ggplot2)
28  library(multiplot)
29  p1 <- ggplot(test_data, aes(x=rentPrice, y=predicted)) +
30    xlab("Preço do aluguel") +
31    ylab("Preço Estimado") +
32    geom_point(colour='red') +
33    ggtitle("Relação entre resultados obtidos")
34
35  p2 <- ggplot(test_data, aes(abs(rentPrice- predicted)) ) +
36    geom_histogram(fill='red', colour='grey') +
37    xlab("Erro absoluto") +
38    ylab(" ")
39
40  multiplot(p1, p2, cols=1)
```



O que não discutimos?

- Performance
- Uso no Mercado
- Comunidade
- Dificuldade de aprendizado

Próximos Passos

- Jupyter / JupiteR

Conclusões

Python e R são ferramentas excelentes

para momentos específicos de cada projeto.

Não são excludentes.

Recursos

Código: github.com/ebonet/pythonandr

Exemplos de Matplotlib: http://matplotlib.org/examples/

Exemplos de ggplot2: http://www.cookbook-r.com/Graphs/

Curso online de R: https://www.datacamp.com/courses

Obrigado!

Dúvidas?