

UFRN



Open Data Day  
Natal, Brazil

01 a 03 de março de 2018

Minicourse

# Machine Learning Fundamentals in Python - Platform & First Model



Speaker:  
Prof. Ivanovitch Silva ([ivan@imd.ufrn.br](mailto:ivan@imd.ufrn.br))



# ANACONDA®

Modern open source analytics platform  
powered by Python



<https://www.continuum.io/downloads>

# Why Anaconda?

## ANACONDA.

### ANACONDA PARTNERS

... and many more!

### ANACONDA ENTERPRISE

Enterprise-Ready Data Science Platform

Collaboration

Reproducibility

Deployment

Security

Scalability

Governance

### ANACONDA DISTRIBUTION

The Most trusted Python Distribution for Data Science

Jupyter

Spyder

NumPy

SciPy

Numba

pandas

DATA

DASK

Bokeh

HoloViews

DataShader

matplotlib

Cython

H<sub>2</sub>O.ai

TensorFlow

CONDA

... and many more!

### ANACONDA SUPPORT

World-Class  
Support for  
Deployments of  
Open Source

### ANACONDA CONSULTING & TRAINING

Solve Enterprise  
Data Science  
Challenges with  
the Creators of  
Anaconda

### DATA SOURCES

Flat Files

SQL

NoSQL

Distributed  
SQL Engines

Spark  
Hadoop

Cloud-Based  
Storage



Home



Environments



Projects (beta)



Learning



Community

Documentation

Developer Blog

Feedback



Applications on

base (root)

Channels



jupyterlab

0.31.5

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Launch



notebook

5.4.0

Web-based, interactive computing notebook environment to write and run human-readable docs while describing the data analysis.

Launch



qtconsole

4.3.1

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Launch

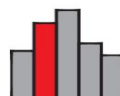


spyder

3.2.6

Scientific PYTHON Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Launch



glueviz

0.12.0

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Install



orange3

3.4.1

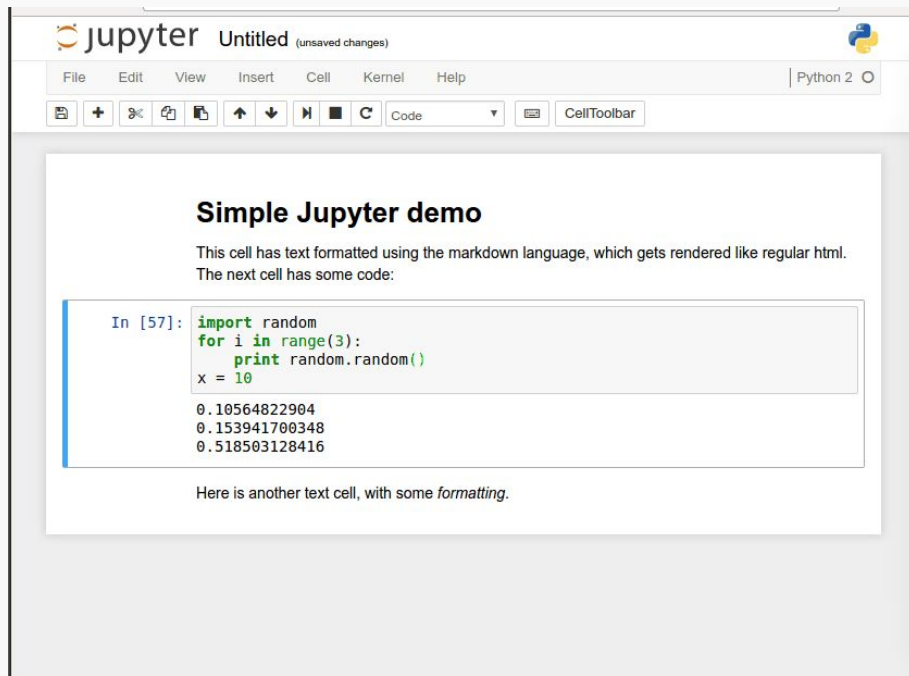
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

Install

# What is a Jupyter Notebook?

---

Mix of code and rich elements (text, figures, links, equations, etc)



Aside from JULia, PYThon and R (JUPYTER) notebook technology also supports many other languages.

<https://github.com/jupyter/jupyter/wiki/Jupyter-kernels>

# Setup a data science environment on Cloud

---

<https://www.datacamp.com/community/tutorials/google-cloud-data-science>

<https://www.datacamp.com/community/tutorials/deep-learning-jupyter-aws>



Google Cloud Platform

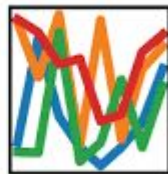






pandas

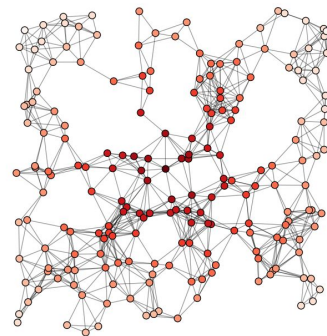
$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



NetworkX  
PyGSP



Folium



Seaborn



bokeh



Beautiful Soap



## EVOLUTION



Line plot  
Area plot  
Stacked area plot  
Parallel plot  
Streamchart

## MAPS



Map  
Choropleth map  
Connection map  
Bubble map

## FLOW



Chord diagram  
Network chart  
Sankey diagram

## Other



Animation  
Cheat sheet  
Data Art  
Color  
3D  
Bad chart



THE PYTHON  
GRAPH GALLERY

<https://python-graph-gallery.com/>

## DISTRIBUTION



VIOLIN  
DENSITY  
BOXPLOT  
HISTOGRAM

## CORRELATION



Scatterplot  
Connected Scatter plot  
Bubble plot  
Heatmap  
2D density plot  
Correlogram

## RANKING



Barplot  
Boxplot  
parallel plot  
Lollipop plot  
Wordcloud  
Spider

## PART OF A WHOLE

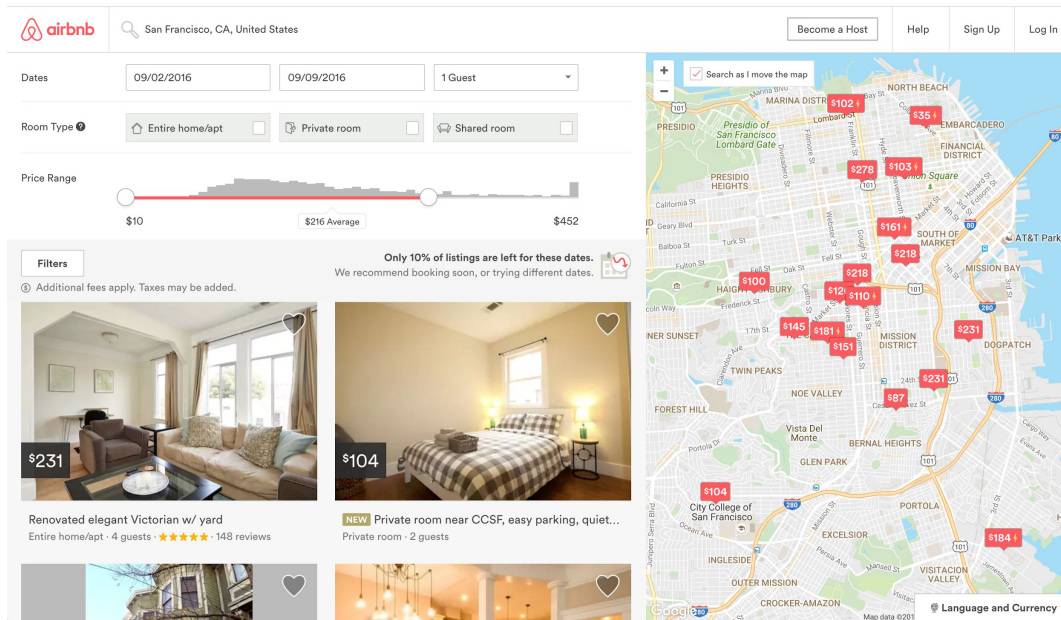


Stacked barplot  
Tree plot  
Venn diagram  
Doughnut plot  
Pie plot  
Tree diagram



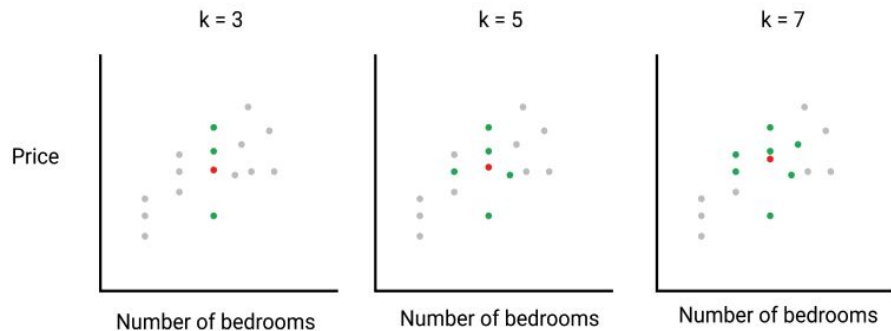
# Problem definition

One challenge that hosts looking to rent their living space face is determining the optimal nightly rent price



- **host\_response\_rate**: the response rate of the host
- **host\_acceptance\_rate**: number of requests to the host that convert to rentals
- **host\_listings\_count**: number of other listings the host has
- **latitude**: latitude dimension of the geographic coordinates
- **longitude**: longitude part of the coordinates
- **city**: the city the living space resides
- **zipcode**: the zip code the living space resides
- **state**: the state the living space resides
- **accommodates**: the number of guests the rental can accommodate
- **room\_type**: the type of living space (Private room, Shared room or Entire home/apt)
- **bedrooms**: number of bedrooms included in the rental
- **bathrooms**: number of bathrooms included in the rental
- **beds**: number of beds included in the rental
- **price**: nightly price for the rental
- **cleaning\_fee**: additional fee used for cleaning the living space after the guest leaves
- **security\_deposit**: refundable security deposit, in case of damages
- **minimum\_nights**: minimum number of nights a guest can stay for the rental
- **maximum\_nightss**: maximum number of nights a guest can stay for the rental
- **number\_of\_reviews**: number of reviews that previous guests have left

Select the number of similar listings, **k**, you want to compare with.



For each listing, calculate how similar it is to our unpriced listing.



For this example, we'll use 3 for our **k** value.

## K-Nearest Neighbors

Rank each listing by the similarity metric and select the first **k** listings.

dataset (ordered by similarity)		
bedrooms	price	similarity
1	160	0
1	60	0
1	95	0
1	50	0
3	350	2

our unpriced listing	
bedrooms	price
1	?

Calculate the mean list price for the **k** similar listings and use as our list price.

dataset (ordered by similarity)		mean price
bedrooms	price	105
1	160	
1	60	
1	95	

# Euclidean distance

$$d = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \dots + (q_n - p_n)^2}$$

index	host_listings_count	accommodates	bedrooms	bathrooms	beds
0	26	4	1	1	2
1	1	6	3	3	3

$$(q_1 - p_1) + (q_2 - p_2) + \dots + (q_n - p_n) \quad \text{differences} \quad (26 - 1) + (4 - 6) + (1 - 3) + (1 - 3) + (2 - 3)$$

$$(q_1 - p_1)^2 + (q_2 - p_2)^2 + \dots + (q_n - p_n)^2 \quad \text{squared differences} \quad (25)^2 + (-2)^2 + (-2)^2 + (-2)^2 + (-1)^2$$

$$\sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \dots + (q_n - p_n)^2}$$

Euclidean distance	= $\sqrt{625 + 4 + 4 + 4 + 1}$
	= $\sqrt{638}$
	= 25.258661

# Euclidean distance - example

Univariate case

$$d = \sqrt{(q_1 - p_1)^2}$$

$$d = |q_1 - p_1|$$

	accommodates
our listing	8

	index	accommodates	distance
dc_listings	0	4	$(4 - 8)^2$
	1	6	$(6 - 8)^2$
	2	1	$(1 - 8)^2$
	3	2	$(2 - 8)^2$

# Randomizing and sorting

---

```
dc_listings[dc_listings["distance"] == 0]["accommodates"]
```

```
26      3  
34      3  
36      3  
40      3  
44      3  
45      3  
48      3  
65      3  
66      3  
71      3  
75      3  
86      3
```

...





# Cleaning & preparing data

---

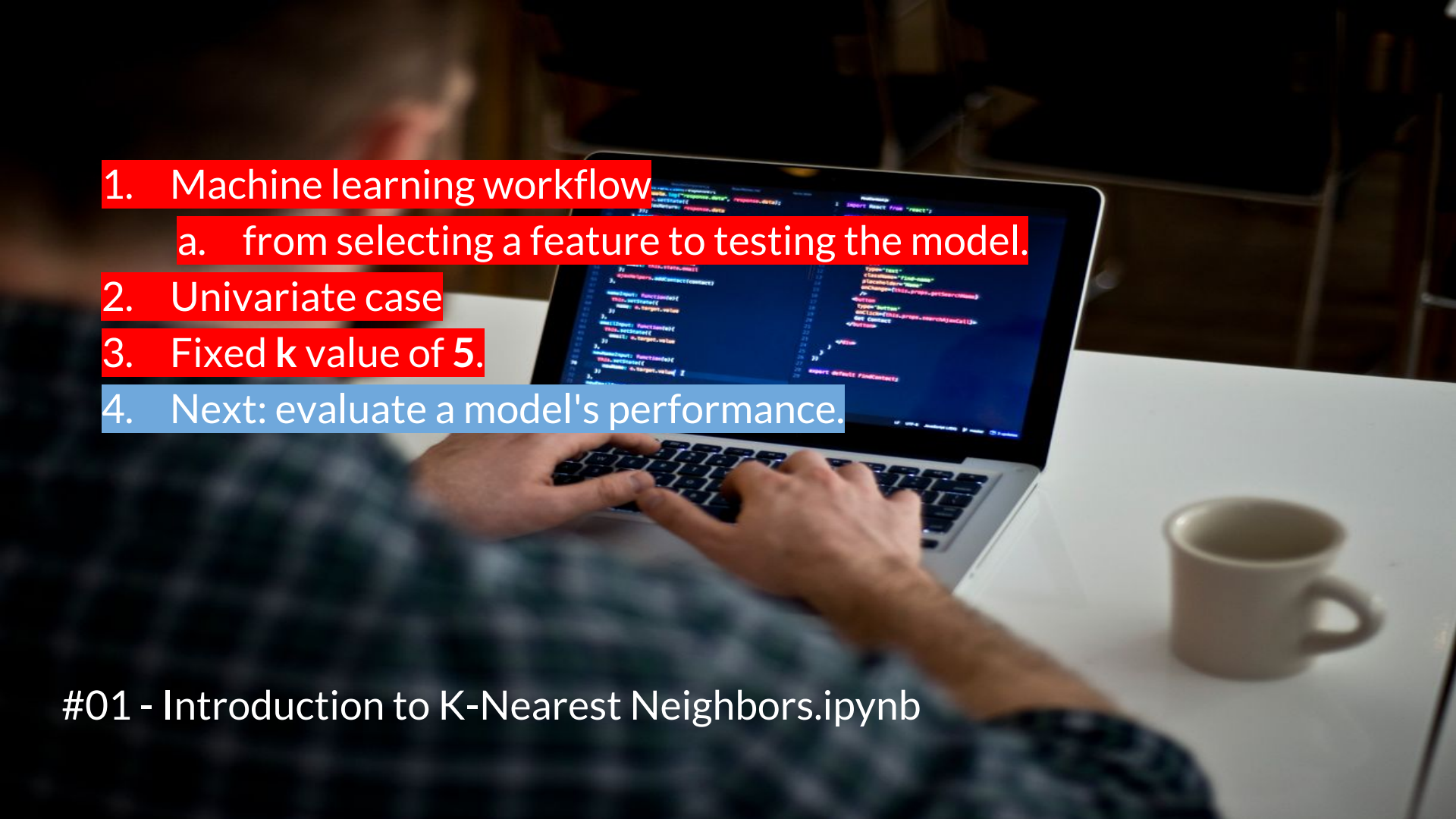
```
# Brought along the changes we made to the `dc_listings` Dataframe.  
dc_listings = pd.read_csv('dc_airbnb.csv')  
stripped_commas = dc_listings['price'].str.replace(',', '')  
stripped_dollars = stripped_commas.str.replace('$', '')  
dc_listings['price'] = stripped_dollars.astype('float')  
dc_listings = dc_listings.loc[np.random.permutation(len(dc_listings))]
```

**Made your first prediction!!!!**

# Function to make predictions

---

```
def predict_price(new_listing):  
    temp_df = dc_listings  
    temp_df['distance'] = temp_df['accommodates'].apply(lambda x: np.abs(x - new_listing))  
    temp_df = temp_df.sort_values('distance')  
    nearest_neighbors = temp_df.iloc[0:5]['price']  
    predicted_price = nearest_neighbors.mean()  
    return(predicted_price)
```

- 
- A person is sitting at a desk, typing on a laptop. The laptop screen displays a Jupyter Notebook with Python code. The code includes imports for pandas, sklearn, and numpy, followed by data loading and preprocessing steps. The person's hands are visible on the keyboard. A white mug is on the desk to the right of the laptop. The background is slightly blurred, showing a bookshelf.
1. Machine learning workflow
    - a. from selecting a feature to testing the model.
  2. Univariate case
  3. Fixed  $k$  value of 5.
  4. Next: evaluate a model's performance.

#01 - Introduction to K-Nearest Neighbors.ipynb