Machine Learning with Python Practical Application

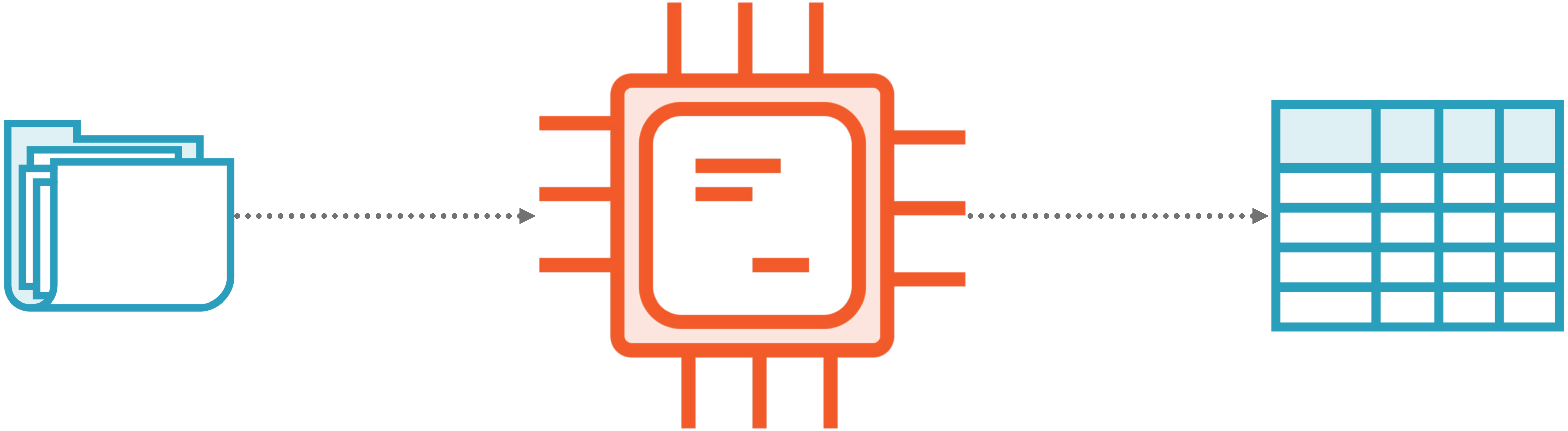
Getting Your Hands Dirty with Machine Learning

Xavier Morera

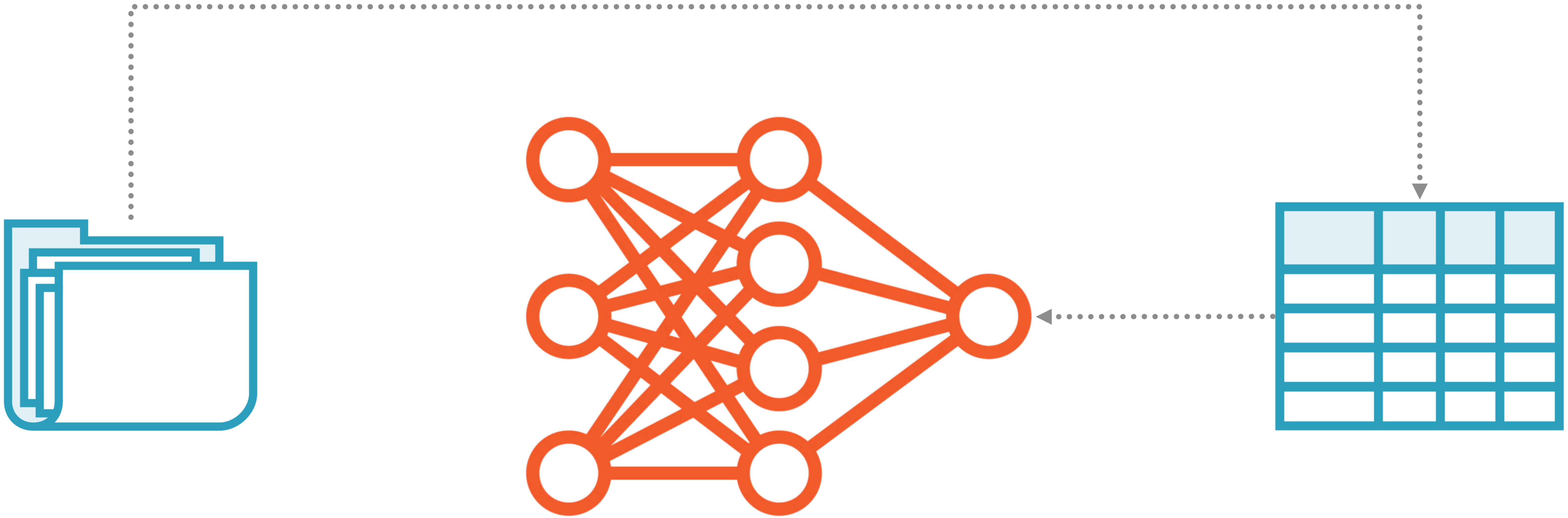
Helping developers understand search, Big Data, and AI/ML

@xmorera www.xaviermorera.com / www.bigdatainc.org

“Regular” Programming



Machine Learning



Getting (ML) Technical

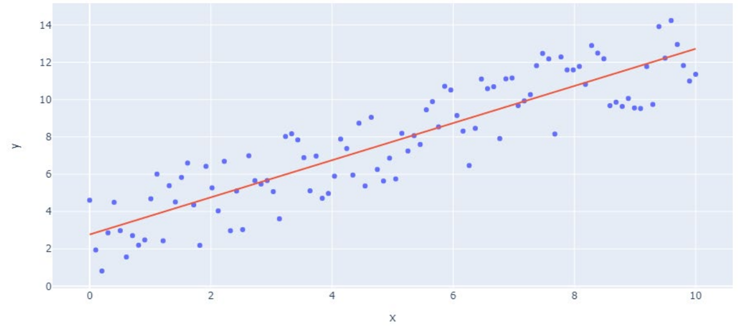
Machine learning as the study of computer algorithms that can improve automatically through experience using data

Tom M. Mitchell defines...

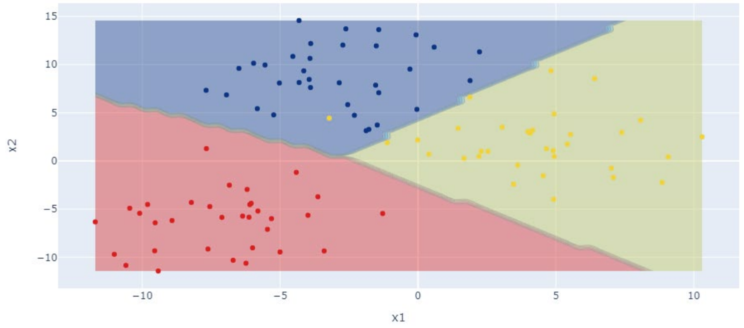
This course focuses on how to pick which Machine Learning algorithm can help you solve your problem

Learning Styles and Algorithm Types

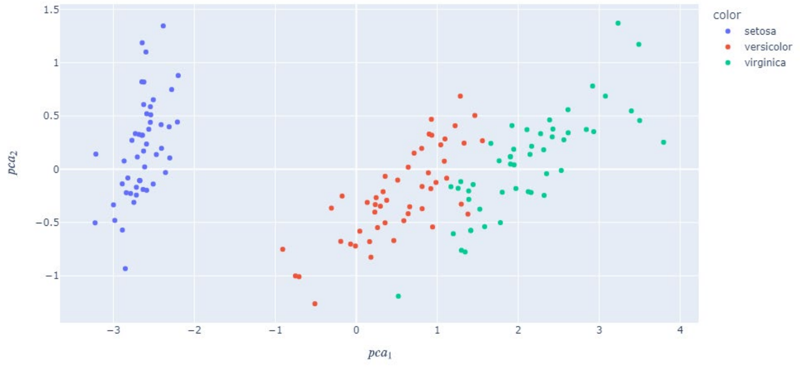
Machine Learning with Python
Practical Application



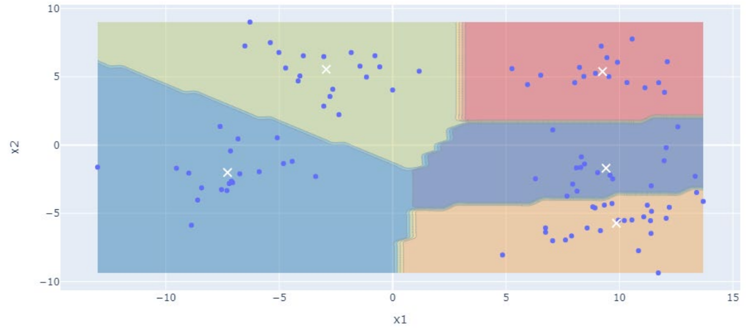
Regression



Classification



Dimensionality Reduction



Clustering

Platforms and Libraries

Formula

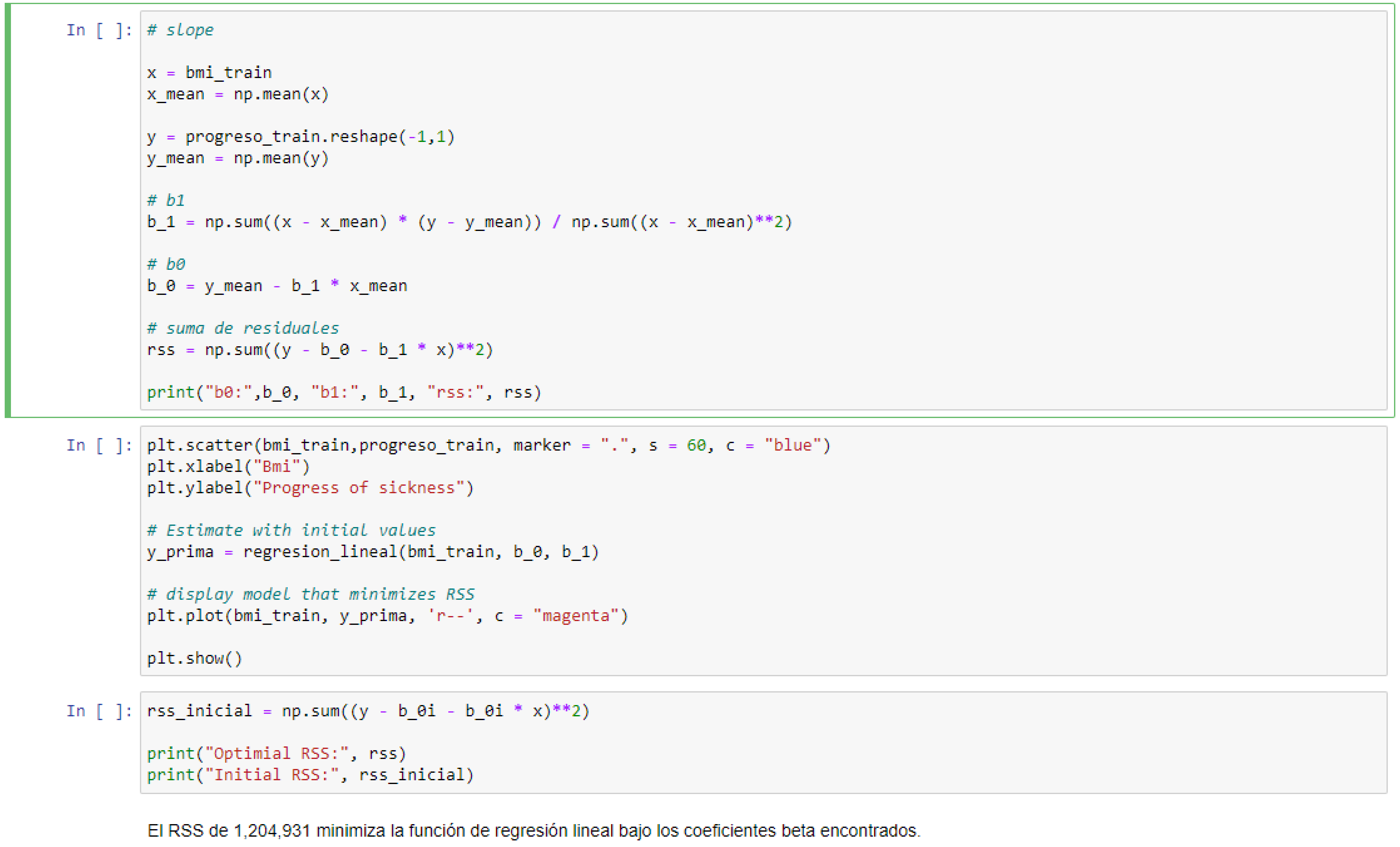
$$y = \beta_0 + \beta_1 x$$

$$\min_{\beta_0,\, \beta_1} Q(\beta_0, \beta_1), \quad \text{for } Q(\beta_0, \beta_1) = \sum_{i=1}^{n} \hat{\varepsilon}_i^{\,2} = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

$$\beta_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2} \qquad \beta_0 = \bar{y} - \beta_1 \bar{x}$$

A Lot of Math and Programming

- $\beta_0 = \bar{y} - \beta_1 \bar{x}$



In [ ]:

```
# slope

x = bmi_train
x_mean = np.mean(x)

y = progreso_train.reshape(-1,1)
y_mean = np.mean(y)

# b1
b_1 = np.sum((x - x_mean) * (y - y_mean)) / np.sum((x - x_mean)**2)

# b0
b_0 = y_mean - b_1 * x_mean

# suma de residuales
rss = np.sum((y - b_0 - b_1 * x)**2)

print("b0:",b_0, "b1:", b_1, "rss:", rss)
```

In [ ]:

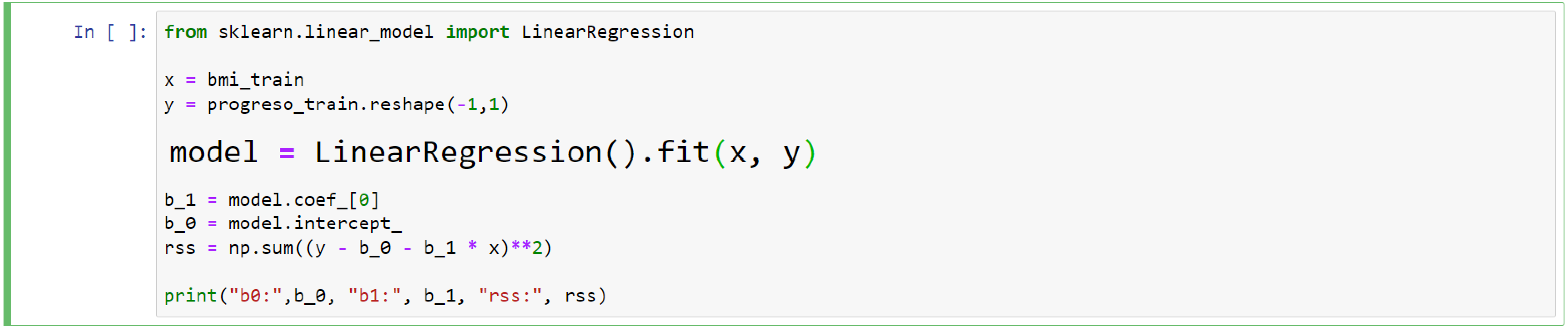
```
plt.scatter(bmi_train,progreso_train, marker = ".", s = 60, c = "blue")
plt.xlabel("Bmi")
plt.ylabel("Progress of sickness")

# Estimate with initial values
y_prima = regresion_lineal(bmi_train, b_0, b_1)

# display model that minimizes RSS
plt.plot(bmi_train, y_prima, 'r--', c = "magenta")

plt.show()
```

In [ ]:

```
rss_inicial = np.sum((y - b_0i - b_0i * x)**2)

print("Optimial RSS:", rss)
print("Initial RSS:", rss_inicial)
```

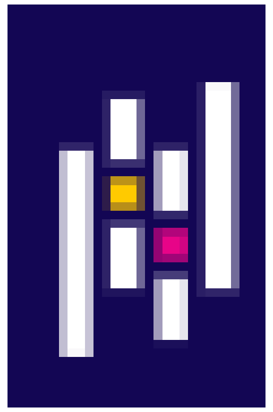
El RSS de 1,204,931 minimiza la función de regresión lineal bajo los coeficientes beta encontrados.

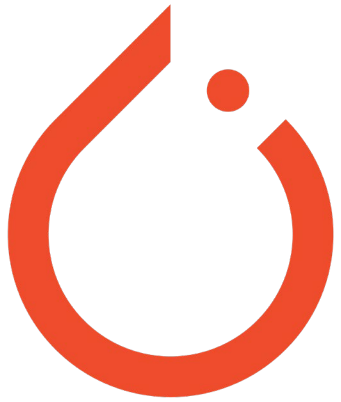
Or Use a Method



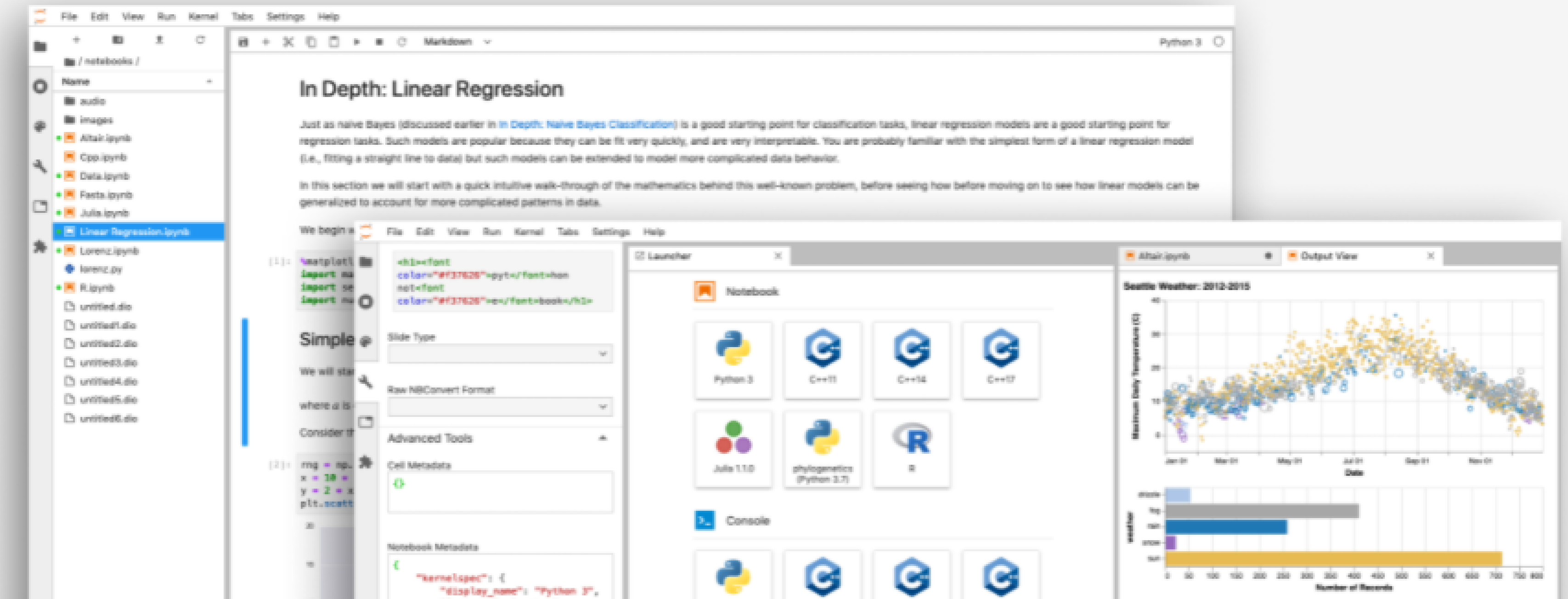
```
from sklearn.linear_model import LinearRegression

x = bmi_train
y = progreso_train.reshape(-1,1)

model = LinearRegression().fit(x, y)

b_1 = model.coef_[0]
b_0 = model.intercept_
rss = np.sum((y - b_0 - b_1 * x)**2)

print("b0:",b_0, "b1:", b_1, "rss:", rss)
```

Libraries and Frameworks





Project Jupyter exists to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages.

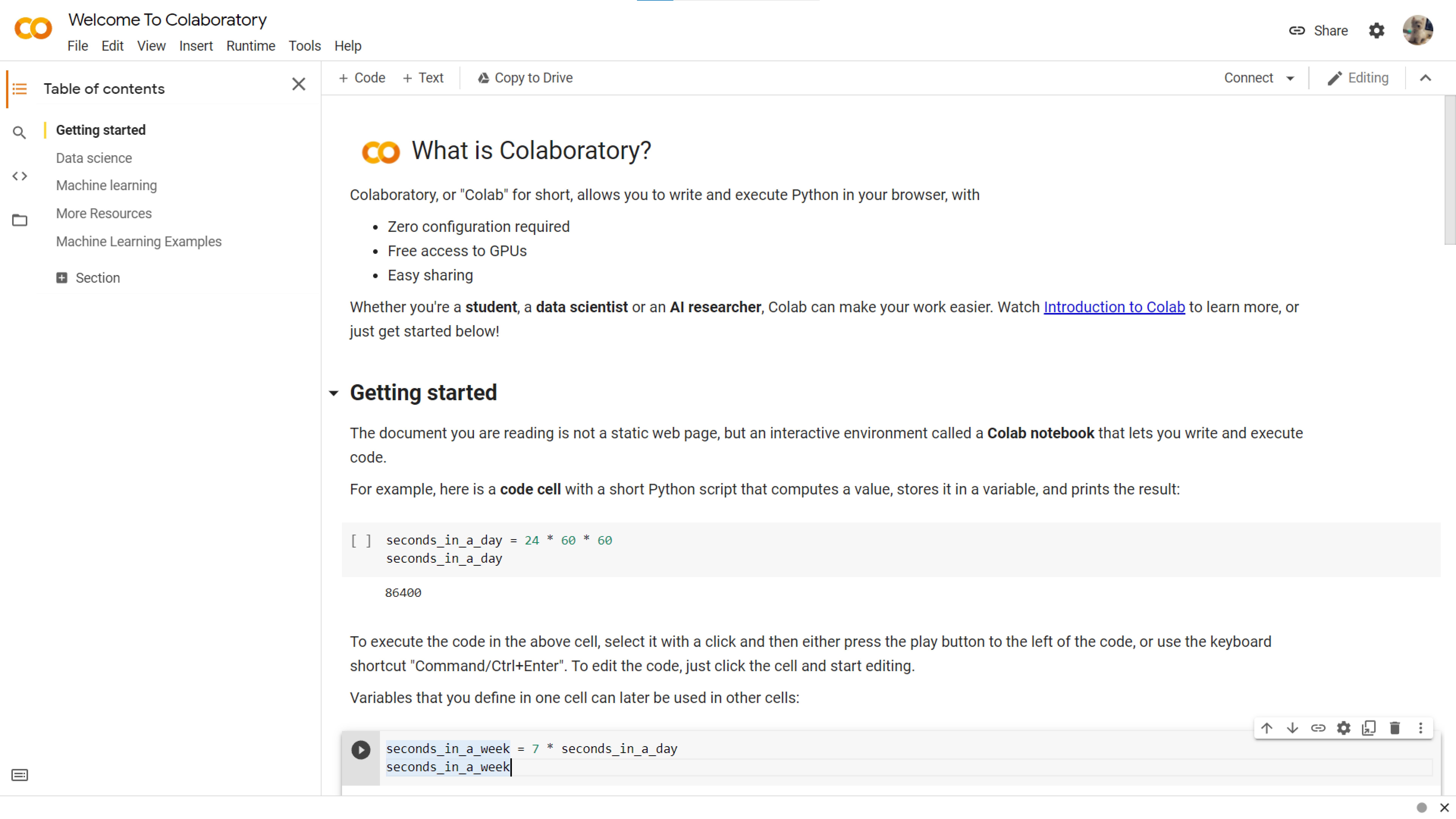

File Edit View Run Kernel Tabs Settings Help
In Depth: Linear Regression
Launcher
Notebook
Console
Seattle Weather: 2012-2015

JupyterLab: Jupyter’s Next-Generation Notebook Interface

JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning. JupyterLab is extensible and modular: write plugins that add new components and integrate with existing ones.

Try it in your browser

Install JupyterLab



What is Colaboratory?

Colaboratory, or "Colab" for short, allows you to write and execute Python in your browser, with

- Zero configuration required
- Free access to GPUs
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier. Watch Introduction to Colab to learn more, or just get started below!

Getting started

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ]  seconds_in_a_day = 24 * 60 * 60
     seconds_in_a_day
```

```
86400
```

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter". To edit the code, just click the cell and start editing.

Variables that you define in one cell can later be used in other cells:

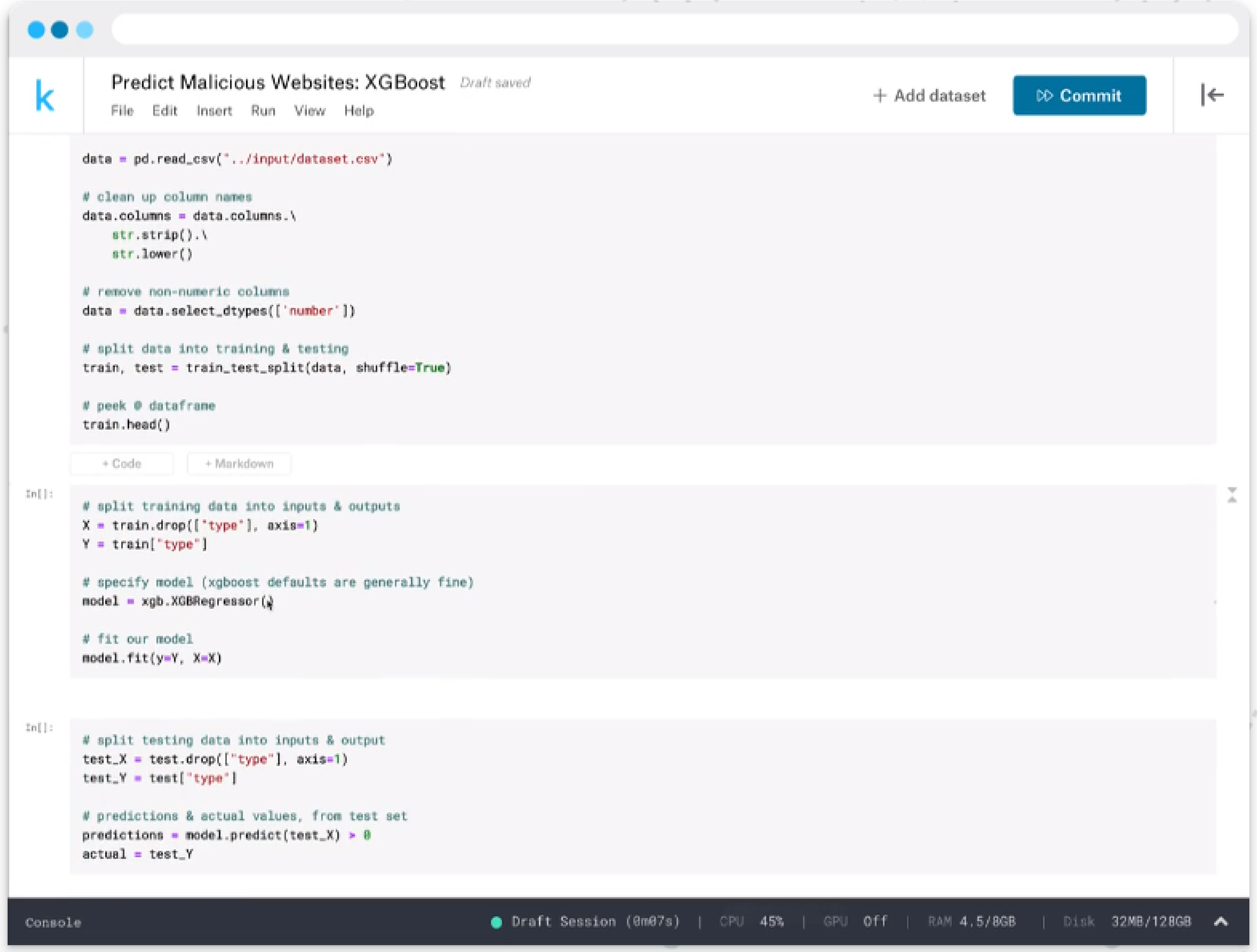
```
seconds_in_a_week = 7 * seconds_in_a_day
seconds_in_a_week
```

Start with more than a blinking cursor

Kaggle offers a no-setup, customizable, Jupyter Notebooks environment. Access free GPUs and a huge repository of community published data & code.

REGISTER WITH GOOGLE

Register with Email


k
Predict Malicious Websites: XGBoost
Draft saved
File
Edit
Insert
Run
View
Help
+ Add dataset
Commit
+ Code
+ Markdown
In[ ]:
In[ ]:
Console
Draft Session (0m07s) | CPU 45% | GPU Off | RAM 4.5/8GB | Disk 32MB/128GB

```
data = pd.read_csv('../input/dataset.csv')

# clean up column names
data.columns = data.columns.\
    str.strip().\
    str.lower()

# remove non-numeric columns
data = data.select_dtypes(['number'])

# split data into training & testing
train, test = train_test_split(data, shuffle=True)

# peek @ dataframe
train.head()
```

```
# split training data into inputs & outputs
X = train.drop(['type'], axis=1)
Y = train['type']

# specify model (xgboost defaults are generally fine)
model = xgb.XGBRegressor()

# fit our model
model.fit(y=Y, X=X)
```

```
# split testing data into inputs & output
test_X = test.drop(["type"], axis=1)
test_Y = test['type']

# predictions & actual values, from test set
predictions = model.predict(test_X) > 0
actual = test_Y
```

Inside Kaggle you'll find all the code & data you need to do your data science work. Use over 50,000 public datasets and 400,000 public notebooks to conquer any analysis in no time.

Maintained by Kaggle

Which platform and libraries you use depends on your particular needs and limitations or possibilities

You should focus on delivering your best results with whatever tools are available.
The objective is not to work hard but to work smart.

Something I tell everyone I can

Takeaway

This course

- Does not teach you Machine Learning
- Helps you pick the right algorithm
- For your kind of problem

Focus

- Regression
- Classification
- Dimensionality reduction
- Clustering