Cloudera Data Science Workbench

Data Visualization

Date published: 2020-02-28
Date modified:

CLOUDERA

Legal Notice

Contents

Data Visualization

Each language on Cloudera Data Science Workbench has a visualization system that you can use to create plots, including rich HTML visualizations.

Simple Plots

You can create a simple plot by running a console in your favorite language.

Procedure

To create a simple plot, run a console in your favorite language and paste in the following code sample:

R

```
# A standard R plot
plot(rnorm(1000))
# A ggplot2 plot
library("ggplot2")
qplot(hp, mpg, data=mtcars, color=am,
facets=gear~cyl, size=I(3),
xlab="Horsepower", ylab="Miles per Gallon")
```

Python

```
import matplotlib.pyplot as plt
import random
plt.plot([random.normalvariate(0,1) for i in xrange(1,1000)])
```

Cloudera Data Science Workbench processes each line of code individually (unlike notebooks that process code per-cell). This means if your plot requires multiple commands, you will see incomplete plots in the workbench as each line is processed.

To get around this behavior, wrap all your plotting commands in one Python function. Cloudera Data Science Workbench will then process the function as a whole, and not as individual lines. You should then see your plots as expected.

Saved Images

You can display images using one of the commands provided in this topic.

Procedure

You can display images, using a command in the following format:

R

```
library("cdsw")

download.file("https://upload.wikimedia.org/wikipedia/commons/2/29/Minard.
png", "/cdn/Minard.png")
image("Minard.png")
```

Python

```
import urllib
```

```
from IPython.display import Image
urllib.urlretrieve("http://upload.wikimedia.org/wikipedia/commons/2/29/Minar
d.png", "Minard.png")

Image(filename="Minard.png")
```

HTML Visualizations

Your code can generate and display HTML.

Procedure

To create an HTML widget, paste in the following:

R

```
library("cdsw")
html('<svg><circle cx="50" cy="50" r="50" fill="red" /></svg>')
```

Python

```
from IPython.display import HTML
HTML('<svg><circle cx="50" cy="50" r="50" fill="red" /></svg>')
```

Scala

Cloudera Data Science Workbench allows you to build visualization libraries for Scala using jvm-repr. The following example demonstrates how to register a custom HTML representation with the "text/html" mimetype in Cloudera Data Science Workbench. This output will render as HTML in your workbench session.

```
//HTML representation
case class HTML(html: String)
//Register a displayer to render html
Displayers.register(classOf[HTML],
  new Displayer[HTML] {
    override def display(html: HTML): java.util.Map[String, String] = {
      Map(
        "text/html" -> html.html
      ).asJava
    }
  })

val helloHTML = HTML("<h1> <em> Hello World </em> </h1>")

display(helloHTML)
```

IFrame Visualizations

Cloudera Data Science Workbench versions 1.4.2 (and higher) added a new feature that allowed users to enable HTTP security headers for responses to Cloudera Data Science Workbench.

Note:

This setting is enabled by default. However, the X-Frame-Options header added as part of this feature blocks rendering of iFrames injected by third-party data visualization libraries.

To work around this issue, a site administrator can go to the Admin Security page and disable the Enable HTTP security headers property. Restart Cloudera Data Science Workbench for this change to take effect.

Most visualizations require more than basic HTML. Embedding HTML directly in your console also risks conflicts between different parts of your code. The most flexible way to embed a web resource is using an IFrame:

R

```
library("cdsw")
iframe(src="https://www.youtube.com/embed/8pHzROP1D-w", width="854px", heigh
t="510px")
```

Python

```
from IPython.display import HTML
HTML('<iframe width="854" height="510" src="https://www.youtube.com/embed/8
pHzROP1D-w"></iframe>')
```

You can generate HTML files within your console and display them in IFrames using the /cdn folder. The cdn folder persists and services static assets generated by your engine runs. For instance, you can embed a full HTML file with IFrames.

R

```
library("cdsw")
f <- file("/cdn/index.html")
html.content <- paste("<p>Here is a normal random variate:", rnorm(1), "</
p>")
writeLines(c(html.content), f)
close(f)
iframe("index.html")
```

Python

```
from IPython.display import HTML
import random

html_content  = "<p>Here is a normal random variate: %f </p>" % random.norma
lvariate(0,1)

file("/cdn/index.html", "w").write(html_content)
HTML("<iframe src=index.html>")
```

Cloudera Data Science Workbench uses this feature to support many rich plotting libraries such as htmlwidgets, Bokeh, and Plotly.

Grid Displays

Cloudera Data Science Workbench supports built-in grid displays of DataFrames across several languages.

Python

Using DataFrames with the pandas package requires per-session activation:

```
import pandas as pd
pd.DataFrame(data=[range(1,100)])
```

For PySpark DataFrames, use pandas and run df.toPandas() on a PySpark DataFrame. This will bring the DataFrame into local memory as a pandas DataFrame.

> **Note:**
>
> A Python project originally created with engine 1 will be running pandas version 0.19, and will not auto-upgrade to version 0.20 by simply selecting engine 2 in the project's Settings Engine page.
>
> The pandas data grid setting only exists starting in version 0.20.1. To upgrade, manually install version 0.20.1 at the session prompt.
>
> ```
> !pip install pandas==0.20.1
> ```

R

In R, DataFrames will display as grids by default. For example, to view the Iris data set, you would just use:

```
iris
```

Similar to PySpark, bringing Sparklyr data into local memory with as.data.frame will output a grid display.

```
sparkly_df %>% as.data.frame
```

Scala

Calling the display() function on an existing dataframe will trigger a collect, much like df.show().

```
val df = sc.parallelize(1 to 100).toDF()
display(df)
```

Documenting Your Analysis

Cloudera Data Science Workbench supports Markdown documentation of your code written in comments. This allows you to generate reports directly from valid Python and R code that runs anywhere, even outside Cloudera Data Science Workbench.

To add documentation to your analysis, create comments in Markdown format:

R

```
# Heading
# -------
#
# This documentation is **important.**
#
# Inline math: $e^ x$
#
# Display math: $$y = \Sigma x + \epsilon$$

print("Now the code!")
```

Python

```
# Heading
# -------
#
# This documentation is **important.**
#
# Inline math: $e^ x$
#
# Display math: $$y = \Sigma x + \epsilon$$

print("Now the code!")
```

Cloudera Data Visualization for ML

Cloudera Data Visualization enables you to explore data and communicate insights across the whole data lifecycle by using visual objects. The fast and easy self-service data visualization streamlines collaboration in data analytics through the common language of visuals.

Using this rich visualization layer enables you to accelerate advanced data analysis. The web-based, no-code, drag-and-drop user interface is highly intuitive and enables you to build customized visualizations on top of your datasets, build dashboards and applications, and publish them anywhere across the data lifecycle. This solution allows for customization and collaboration, and it provides you with a dynamic and data-driven insight into your business.

Cloudera Data Visualization is integrated with Cloudera Machine Learning (CML) in all form factors. You can use the same visualization tool for structured, unstructured/text, and ML analytics, which means deeper insights and more advanced dashboard applications. You can create native data visualizations to provide easy predictive insights for business users and accelerate production ML workflows from raw data to business impact.

For more information, see the *Cloudera Data Visualization documentation*.

Related Information

Cloudera Data Visualization in CDP Public Cloud

Cloudera Data Visualization in CDSW