Data Refinery for DSX Local

*August, 2018*

*Author: nmanchev@uk.ibm.com*

**Table of contents**

Contents

[Overview 1](#_Toc521860143)

[Pre-requisites, access, and files 1](#_Toc521860144)

[The Problem 1](#_Toc521860145)

[Part 1: Set up a Project 2](#_Toc521860146)

[Part 2: Exploring the data and defining transformations 5](#_Toc521860147)

[Part 3: Modifying the data set 9](#_Toc521860148)

[Part 4: Running the data transformation script 19](#_Toc521860149)

# Overview

This Lab exercise will guide you on how to use the Data Refinery add-on to wrangle data sets in DSX Local. You will learn how to:

* Load data
* See various statistics and visually explore the data
* Apply transformation
* Export the flow to an R script and run it to produce a transformed version of the original data

# Pre-requisites, access, and files

* Working knowledge of DSX Local.
* Access to a DSX Local cluster with the DODS add-on installed.
* Download this lab file:

<https://raw.githubusercontent.com/nmanchev/DSXBPEnablement/master/DataRefinery/cardata.csv>

The data in this file is a modified version of the Auto MPG Data Set, freely available from the UCI Machine Learning Repository.

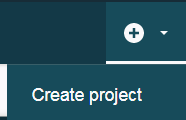
# The Problem

You have received a data extract containing data for vehicles from different auto manufacturers. The data set is an CSV format. The task is to explore onboard and explore the data in Data Science Experience Local, and to select features that would be useful for training machine learning models.

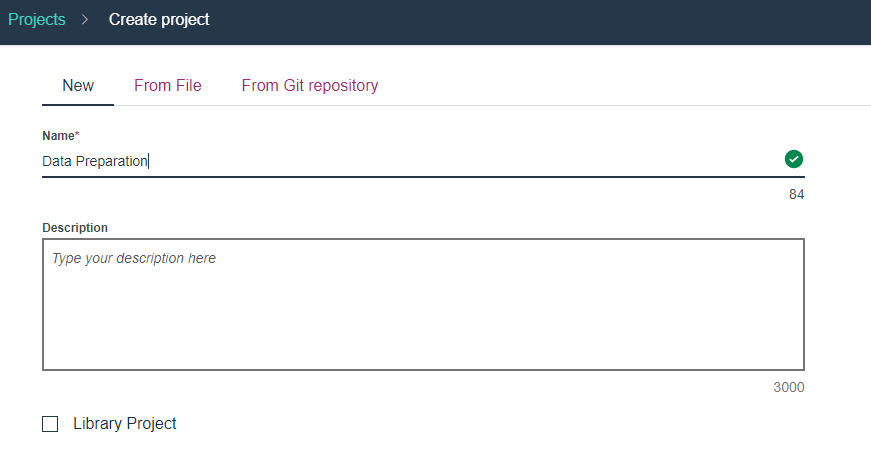
# Part 1: Set up a Project

In this section we will set up a DSX Local Project. The Project is a high-level container where all assets are stored or referenced.

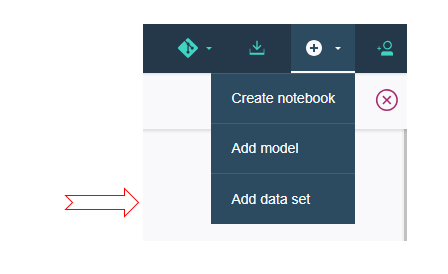
1. Login DSX Local



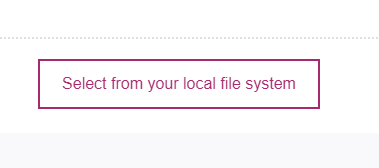
1. Once on the DSX landing page, click on the little plus (+) sign on the top right of the screen and select *Create Project*. 🡪
2. Enter a project name (i.e. “Data Preparation”) and a brief description. Click *Create*.



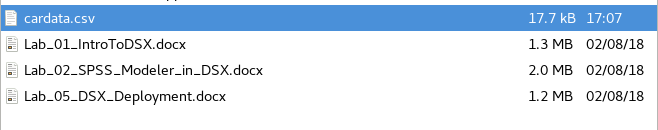
1. On the Project Dashboard click on the Plus sign and select "Add data set"



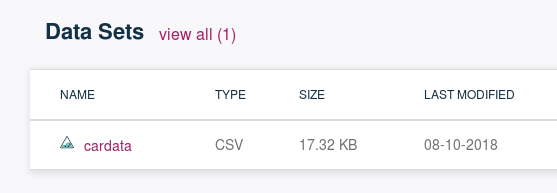
1. Click the "Select from your local file system" button.



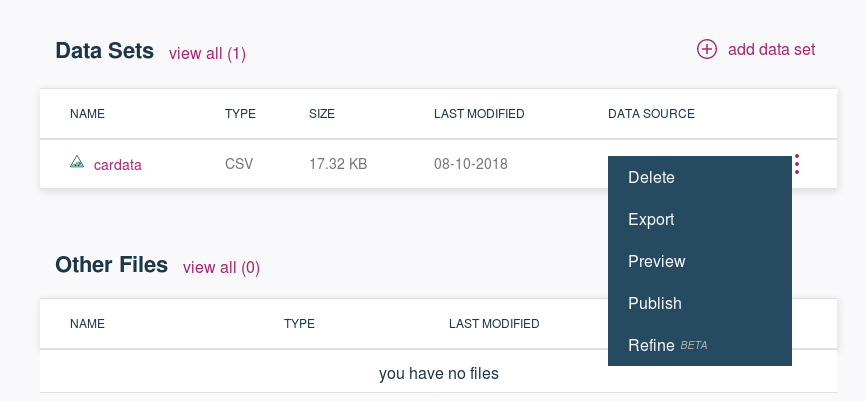
1. Navigate to where you have downloaded the cardata.csv file, select it and click "Open".



The file gets uploaded to your project and once the operation completes you should be able to see it in the Data Sets section.



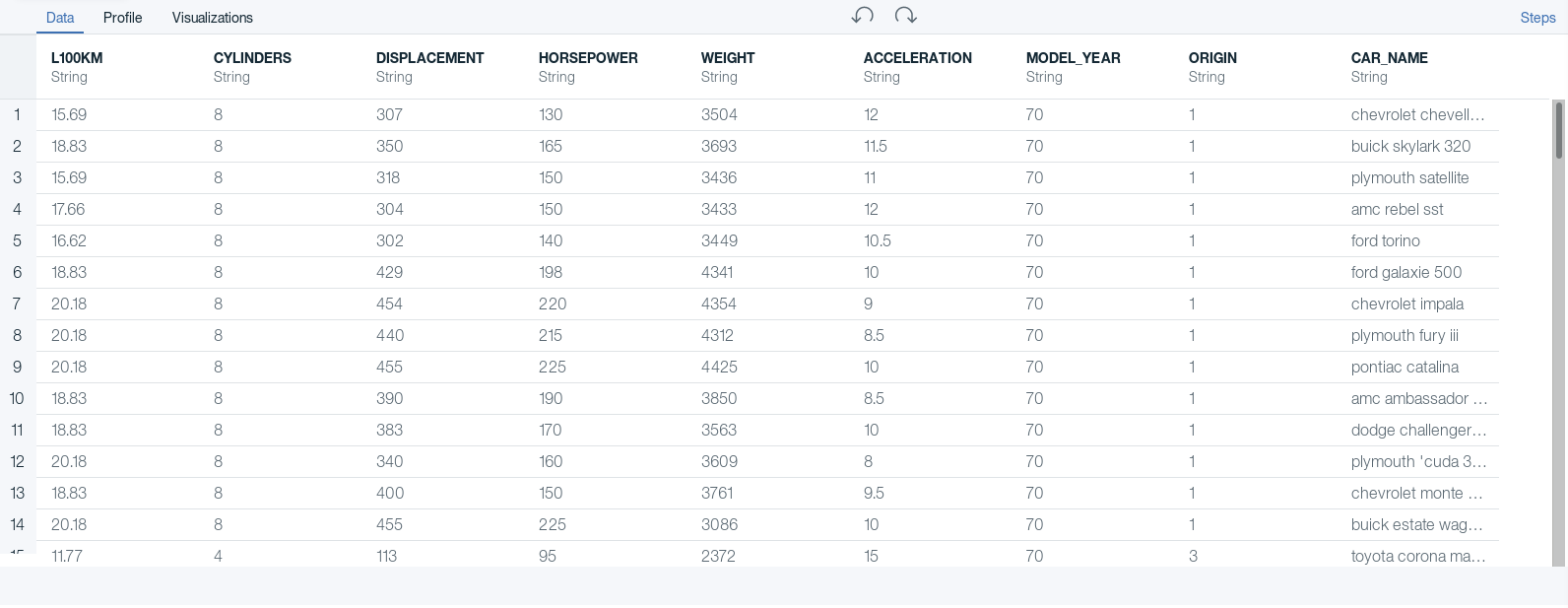
1. Click the  button next to the file name and select the Refine option.



This starts Data Refinery and automatically opens the select file for inspection.

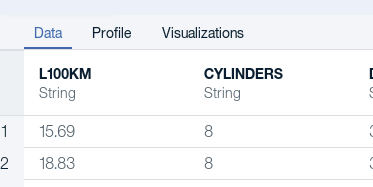
# Part 2: Exploring the data and defining transformations

Once in Data Refinery, you should see a sample from the cardata.csv file.

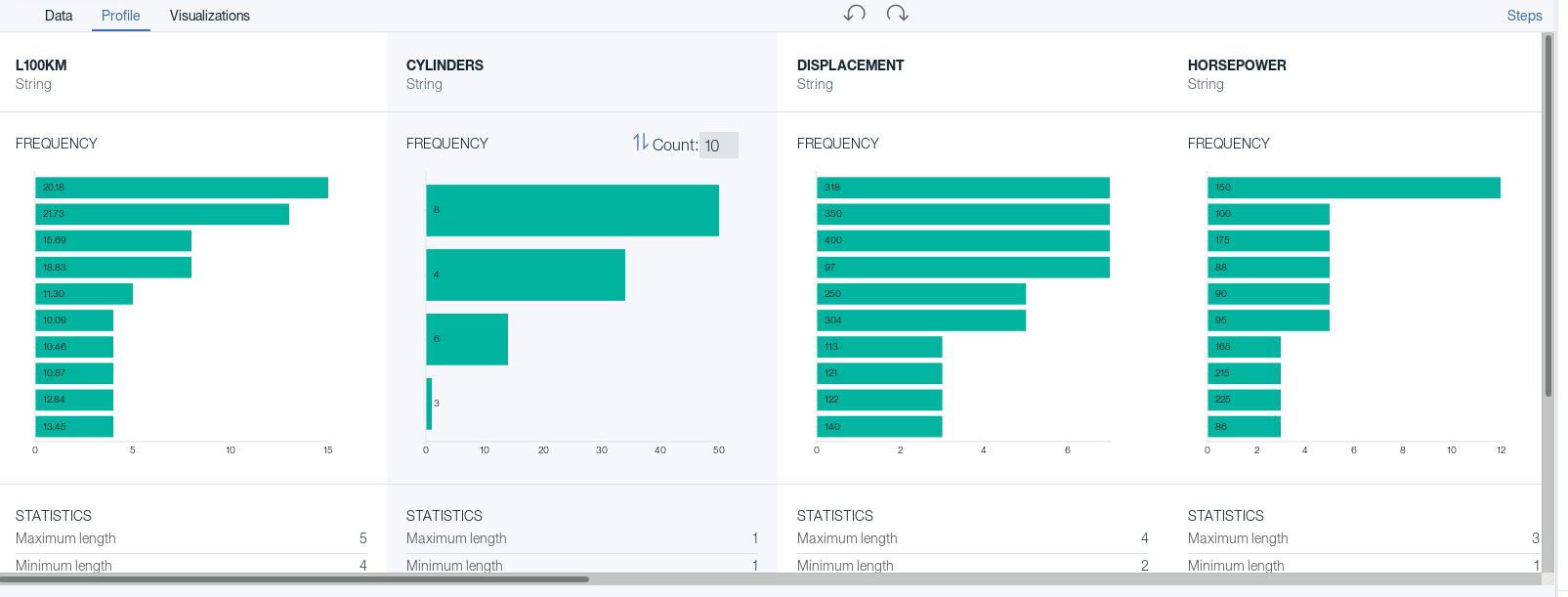


As you can see, every record (line in the file) is an entry for one specific vehicle. For each vehicle we have various attributes like fuel consumption (in litres per 100 km), number of cylinders, displacement, horsepower, weight etc.

1. Click on the Profile link above the data.



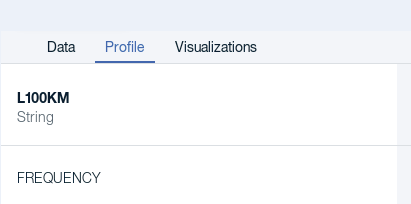
This opens the Profile page that shows you various statistics for each attribute in the data set.



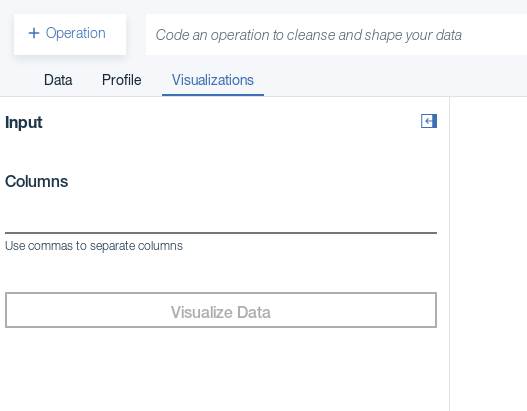
Feel free to explore around by looking at the data distribution for each attribute, and at the accompanying statistics.

Note, for example, that most the vehicles in the data set have 8 cylinder engines. Why do you think that is? Can the Year attribute tell you something about it? Can you infer something about the Origin attribute?

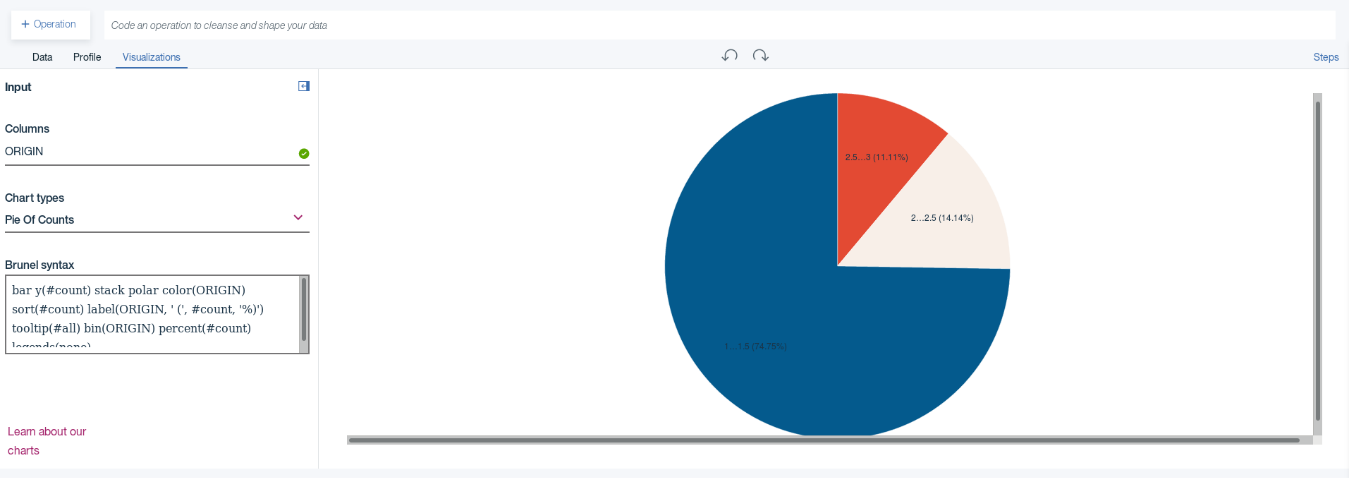
1. Click on the Visualizations link.



This opens the Visualizations tab, where you can define and look at various plots based on the data set. The pane on the left hand side allows you to pick various attributes, and based on their number and type, Data Refinery will present you with a list of different char types that you can select from.



1. Select ORIGIN as the single column for this chart, press the Visualize Data button, then set the type to Pie of Counts.



As you can see, most of the cars are from manufacturers based in region 1 (this is US, in case you're wondering.)

**Hint:** Note the Brunel syntax text area in the Input pane? Data Refinery automatically generates the Brunel commands that will recreate the plot. Brunel is an easy to use visualization library that sits on top of low-level Python visualization frameworks. You can copy and save this syntax and use it to recreate plots in Python notebooks if needed.

1. Now change the columns to L100KM and WEIGHT, and set the chart type to Scatterplot. Your screen should now look similar to this

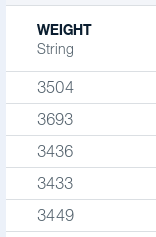


As you can see, there is certainly a connection between the weight of the car and the fuel consumption. Maybe we explore this later to train a predictive model that gives us the expected consumption based on the weight of the vehicle.

There is, however, a problem. The consumption is given as litres per 100 km, while the weight is expressed in pounds. We'd like to have everything consistent, so let's see how we can introduce modifications to the data.

# Part 3: Modifying the data set

1. Inspect the WEIGHT attribute. Notice, that by default its type is set to String. We have to change this if we want to apply calculations to the column.

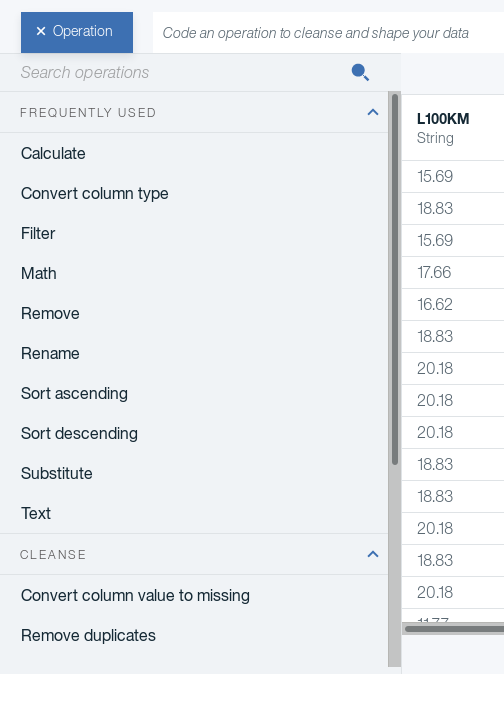


1. Click the  button next to WEIGHT and select Convert Column -> Integer, as shown below.

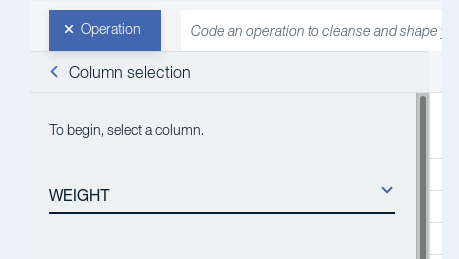


Notice that its type changes to Integer.

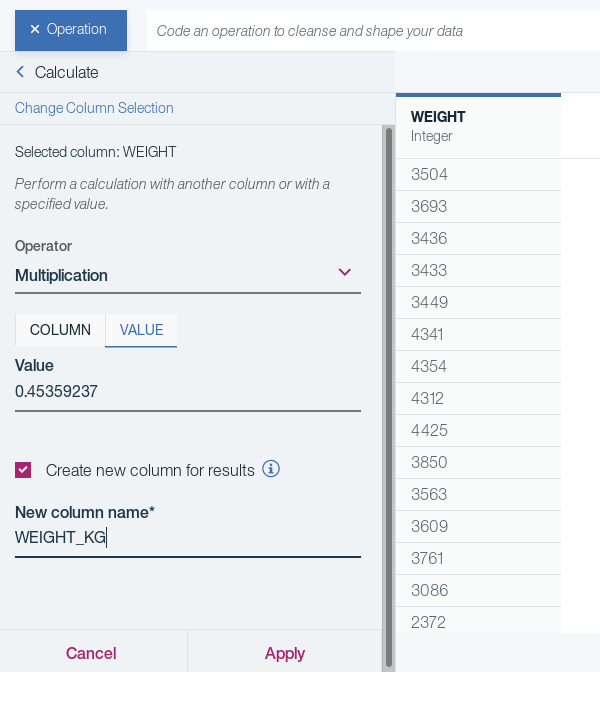
1. Click the  button to open the Operations pane on the left hand side. Then chose Calculate.



1. Select the WEIGHT attribute and click Next.



1. Set the operator to Multiplication as shown below.

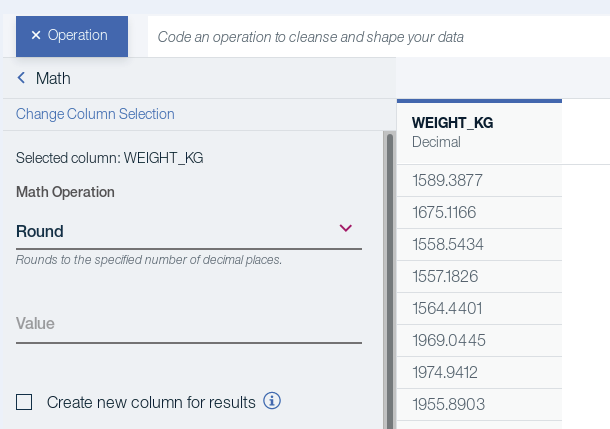


Switch from column to VALUE and enter 0.45359237. Multiplying the weight by this constant will convert it from pounds to kilograms. Select the "Create new column for results" options and name the new column WEIGHT\_KG. Click Apply.

You will notice that the data set now contains a new attribute WEIGHT\_KG:

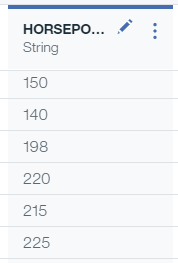


1. As a final step, let's round the values to make them a bit more presentable. Add a new operation (using the  button), elect the WEIGHT\_KG column, and set the operation to Round. Leave the Value field empty.

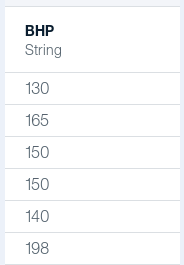


After clicking Apply you will see that the values for WEIGHT\_KG are now rounded to the nearest integer.

1. As a next step, let's pretend that the HORSEPOWER column is mislabeled and it contains data for the break horsepower instead. Hover the mouse next to the column name and click the  button when it appears. Change the name of the column to BHP.

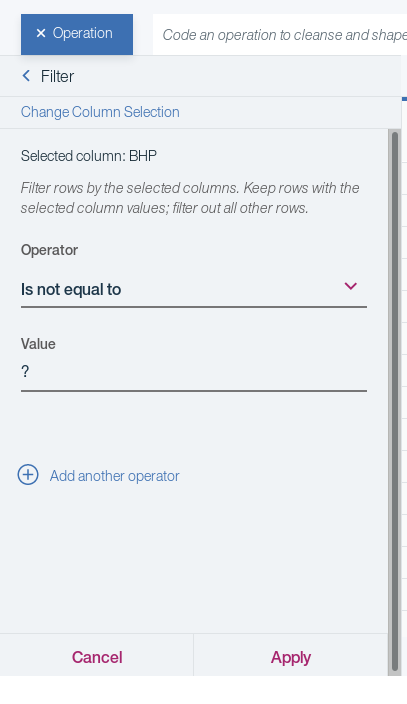


The renamed column should now look like this:



1. If you inspect the values for BHP more closely, you will note that there are some missing values in the dataset denoted by the ? character. Let's get rid of those. Add a new operation (use the  button).

From the list of operations chose Filter and select the BHP column.



Set the operator to "Is not equal to" and the Value field to ?. Click Apply.

Inspect the values in the data set again and notice that the records containing missing values have been removed.

1. As a final step, let's only keep the L100KM, BHP, MODEL\_YEAR, and WEIGHT\_KG columns and remove everything else. Select  next to the name of the DISPLACEMENT column and click Remove.

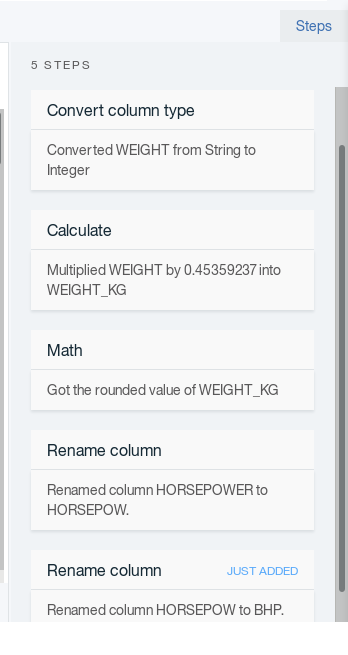


Repeat this step for all other columns except L100KM, BHP, MODEL\_YEAR, and WEIGHT\_KG.

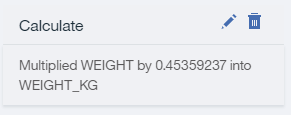
1. Your dataset should now look like similar to this.



Click on the  button in the top right corner above the dataset. This will open the Steps pane, revealing that Data Refinery keeps all defined operations as a sequential flow.

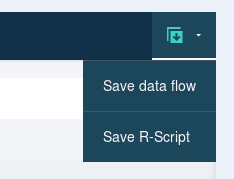


At any given time you can revisit an operation, modify it or completely remove it from the flow. Hover your mouse next to the title of any operation to reveal the Edit/Delete buttons.

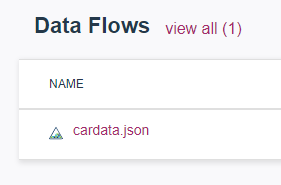


1. Assuming we are happy with the flow, let's store it as it is.

Click the  button in the top right hand corner of the screen. This reveals the save options.

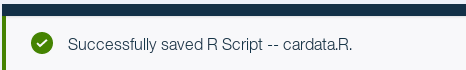


1. Click the Save data flow option. This will store the flow as a JSON document that appears in your Data Flows section of the project.

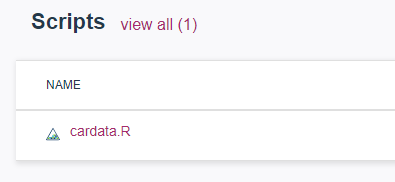


You can click on this file at any given time, and it will open Data Refinery, loading all of the steps you have defined so far.

1. The other option presented by the  button is "Save R-Script". This option converts your flow into an R script that you can later execute, modify with custom R code, or incorporate into larger scripts. Select this option and make sure you receive a confirmation that the flow was successfully stored.



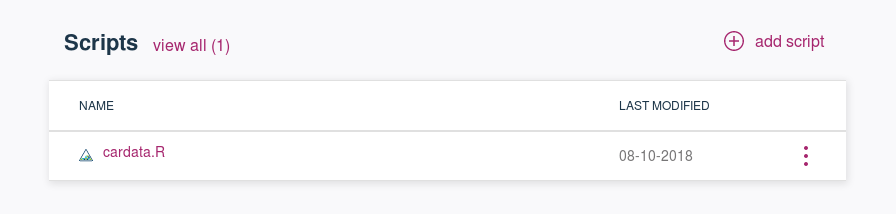
You should be able to find the cardata.R script in the Scripts section of your project.



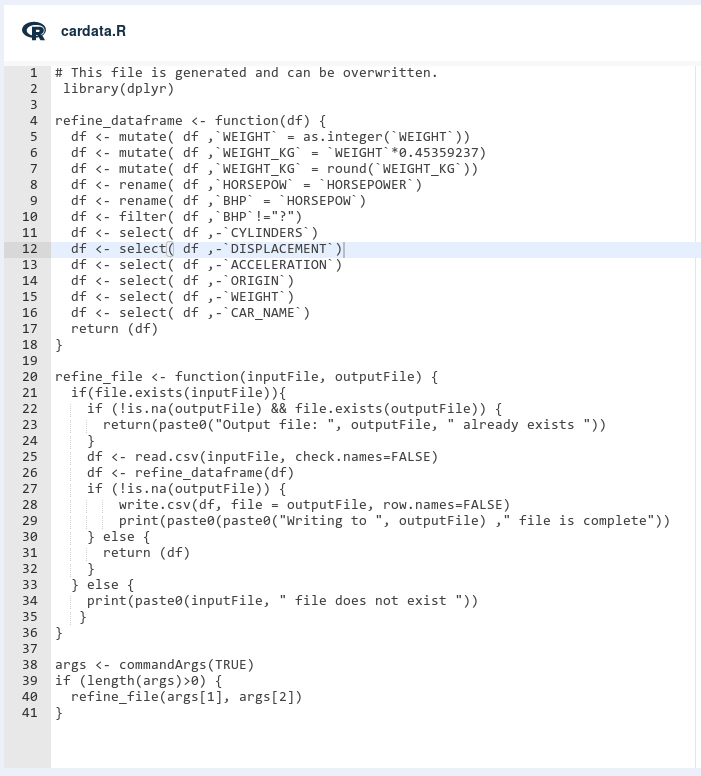
# Part 4: Running the data transformation script

Let's now use the flow to process the original dataset and create a copy of the data with the modification we've just specified.

1. Go to the Scripts section of your project and click on the name of your R script (cardata.R)



This opens a script editor that reveals the generated R code.

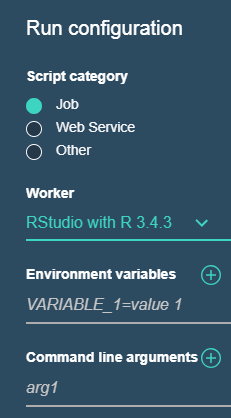


**Hint:** Note that you can edit the script by hand and introduce additional modifications that can be more or less complex.

1. Note that the script expects two arguments that will tell it what the source and destination file names are. We can specify this in the run configuration of the script. Click the Run configuration button as shown below.



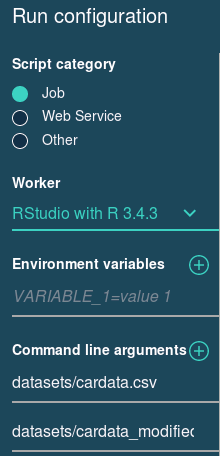
This opens the Run configuration pane.



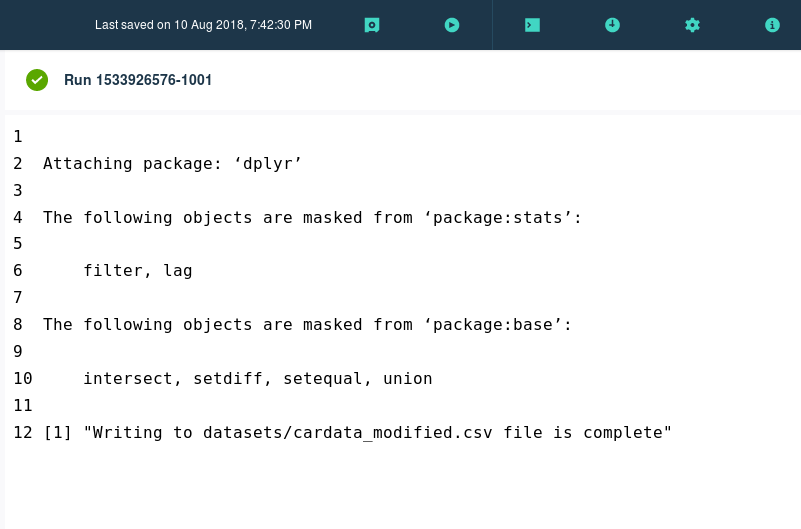
The options above allow you to set the run category (e.g. Job / Web Service), select worker to run the script, set environment variables, and pass command line arguments.

1. Set the first argument to datasets/cardata.csv. Click the  button to add a second argument and set its value to datasets/cardata\_modified.csv.

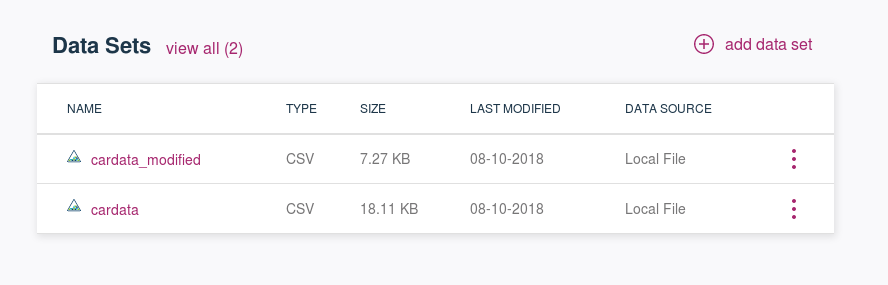
Your configuration should now look similar to this



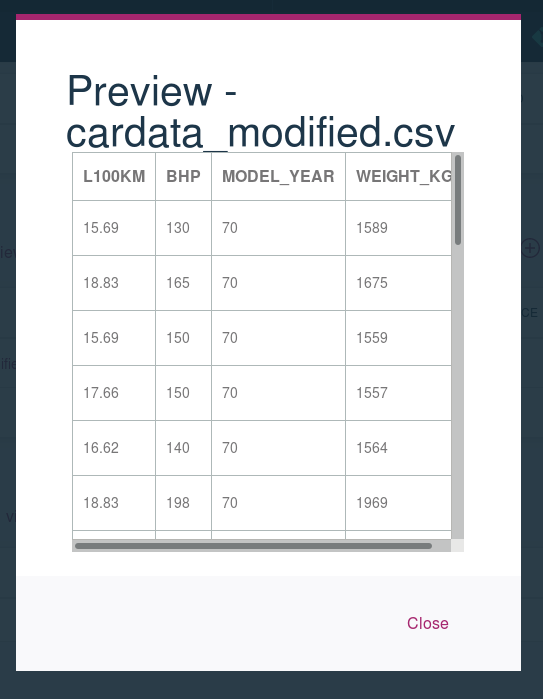
1. Now click the Run button () and look as the output produced by DSX as your script is being executed.



1. Once the execution completes, get back to your Assets project page and inspect the Data Sets section. You should now see the newly created CSV file alongside the original data set.



1. Click on cardata\_modified and confirm that it contains the modified version of the data. You can now feed this data to Model Builder or any other DSX component that assists in training machine learning models.



**CONGRATULATIONS!**

**You have successfully completed the Data Refinery Lab.**