Dataset: Mushroom Data Set

Dataset estimate: 8124 records (4208 edible I.e. consumable and 3916 poisonous I.e. harmful); 22 columns (phenotypic traits codes, e.g. cap color, odor, veil type, habitat) and 1 label column (edible I.e. eatable or poisonous I.e. harmful).

Dataset portrayal: This informational index incorporates depictions of theoretical examples comparing to 23 types of gilled mushrooms in the Agaricus and Lepiota Family.

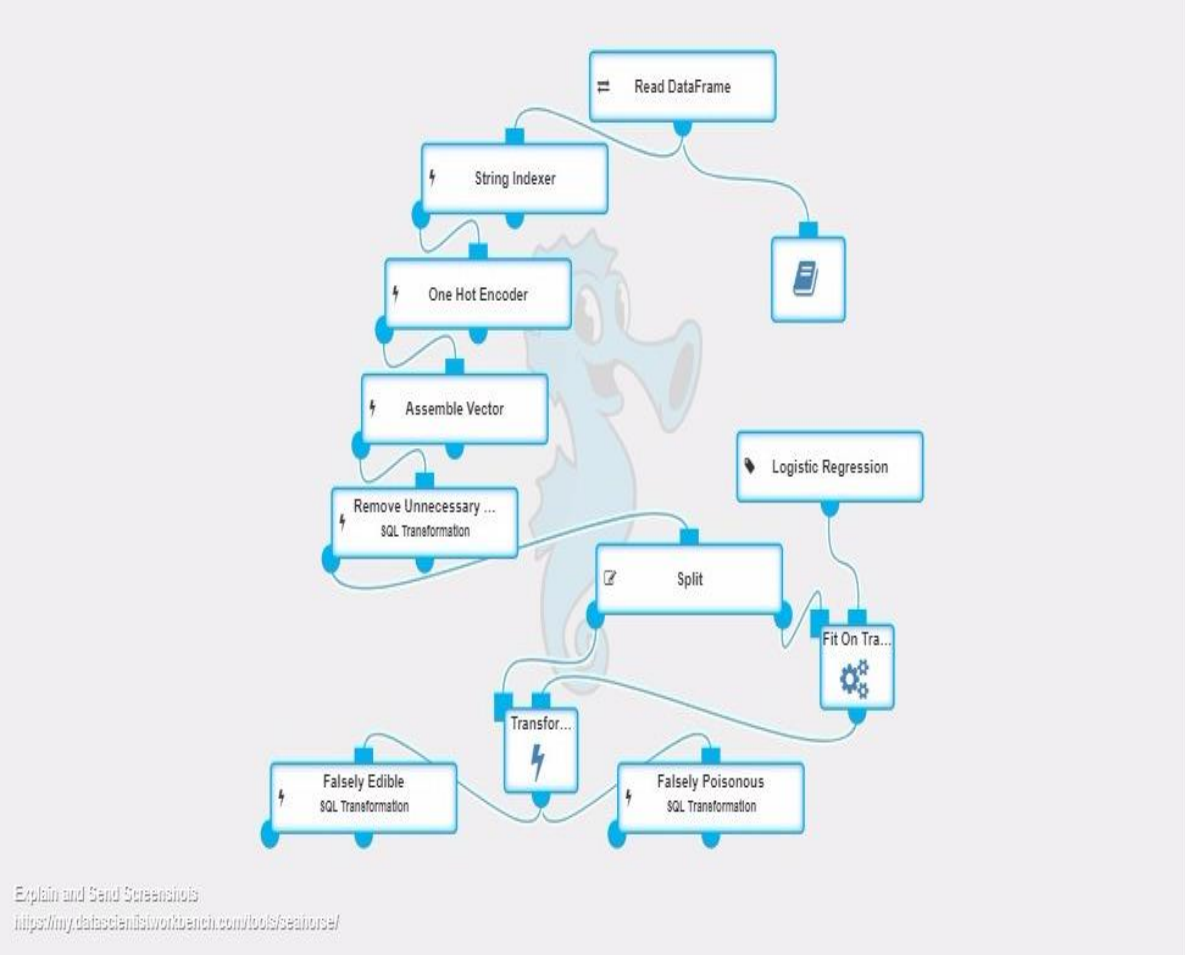
Business reason: Distinguishing toxic and palatable mushrooms. Grouping model made amidst this test could help data engineers and information specialists in characterizing mushroom examples as harmful or consumable.

Informational collection credits: Lichman, M. (2013). UCI Machine Learning Repository http://archive.ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science.

:

This is a total examination that produces a mushroom arrangement display. The work process record is incorporated into Seahorse, Here I will demonstrate to you one of the proper methodologies to make that test well ordered.

Here is how I have created the work flow:



Here is the step by step process of creating the work flow:

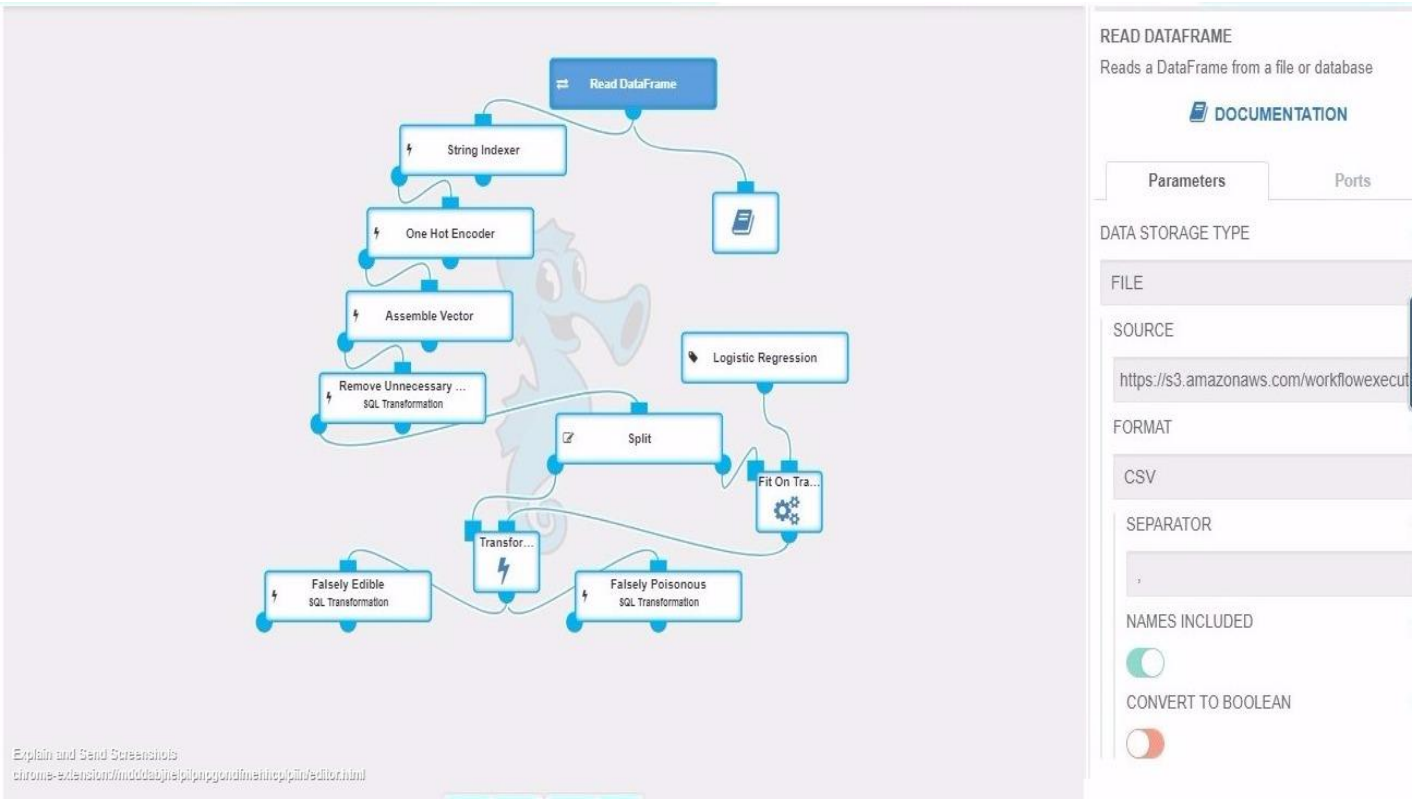
The data is formatted and processed in a way that it has a column name as the header.

The cleansed and processed data has 23-columns and is stored as a comma-separated CSV-file with column names in the first line.

To work with the dataset, it has to first be loaded into Seahorse. This is done by a Read DataFrame operation. Here I am placing the dataset on the canvas using drag-and-drop from the operations palette. To load the data, I have created a Data Source corresponding to the file.

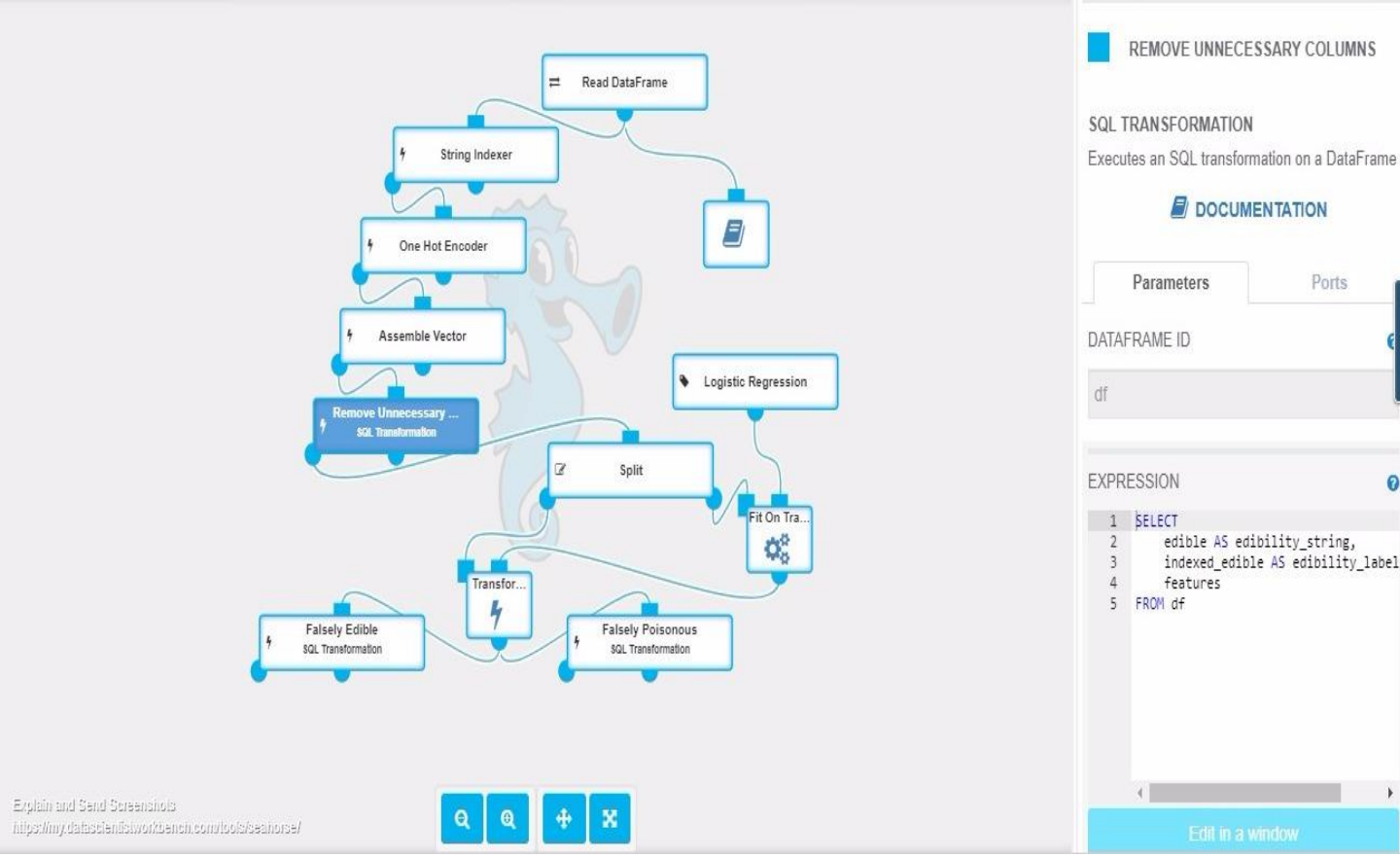
SOURCE:https://s3.amazonaws.com/workflowexecutor/examples/data/mushrooms.csv

Read DataFrame reads a DataFrame from a file as shown below



After preparing the Data Source for Read DataFrame, the operation is ready to be executed. I clicked the RUN button in the top Seahorse toolbar. Then I selected that operation before clicking RUN. When the execution ended, a report of the operation became available. I clicked on the operation output port to see its results.

The DataFrame is too wide (more than 20 columns) to permit seeing the information test in the report, so I have investigated here just the column type and names. Presently, I am able to open the Python Notebook by choosing it on the canvas and clicking at the "Open notebook" icon on the correct board.



In the newly opened window, I went ahead and entered: dataframe (). take (10) and clicked on the “run cell” button. Then I was able to closely investigate the data sample of the first 10 rows returned by the Read DataFrame operation.

In the first column I see names expressing to which class a particular example has a place (conceivable esteems: eatable or noxious). I have found that all sections have string esteems. To perform order, Seahorse needs a numeric name segment and a vector of numerics as highlights segment. I have to delineate esteems to numbers and get together highlights into a solitary vector segment

Data Transformation:

Here is a way how it can be done by combining multiple operations – String Indexerwith One Hot Encoder and Assemble Vector – as shown

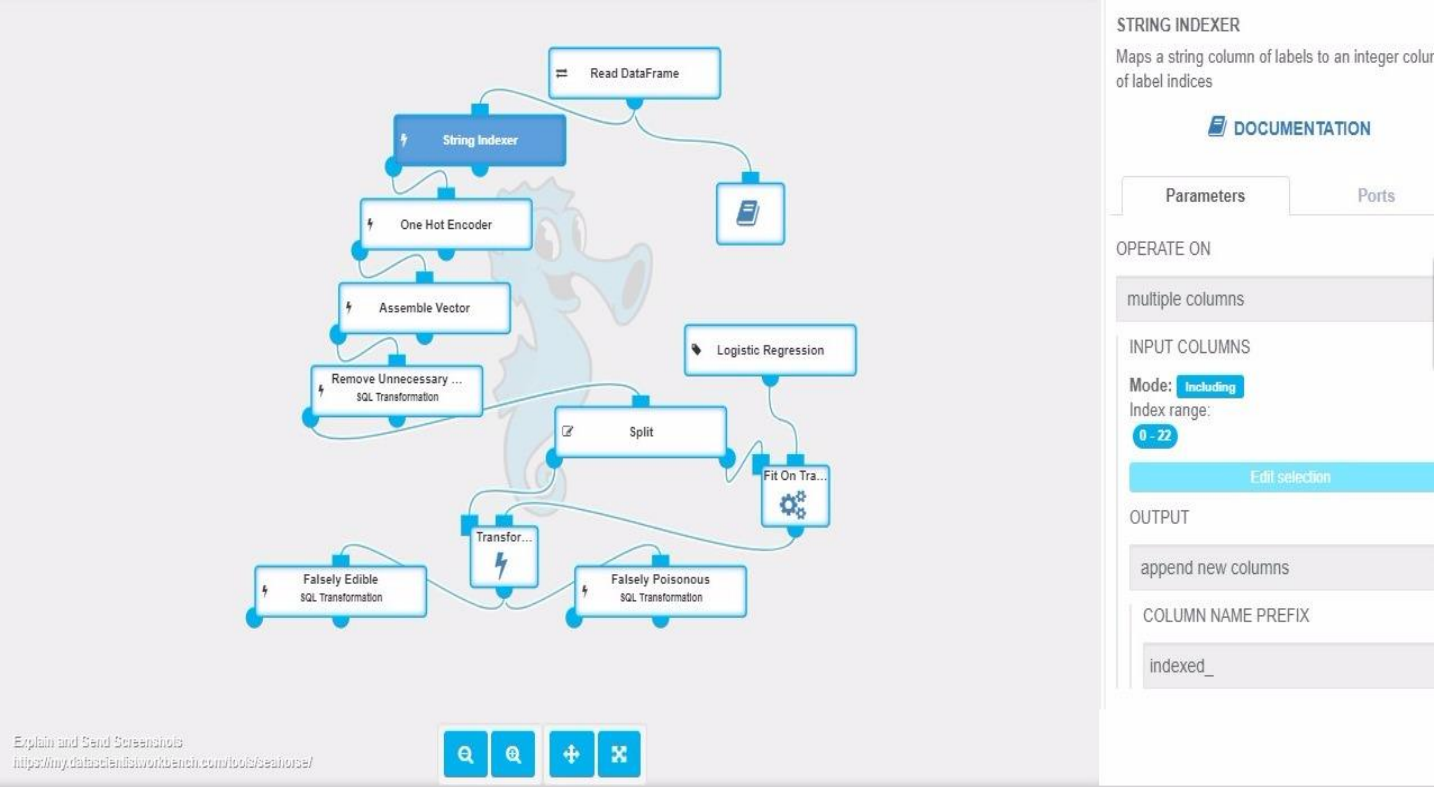
The String Indexer operation translates string values to numeric ordinal values. It needs to have its parameters modified, as follows:

OPERATE ON: multiple columns

INPUT COLUMNS: Including index range 0-22 (all columns)

OUTPUT: append new columns

COLUMN NAME PREFIX: indexed_

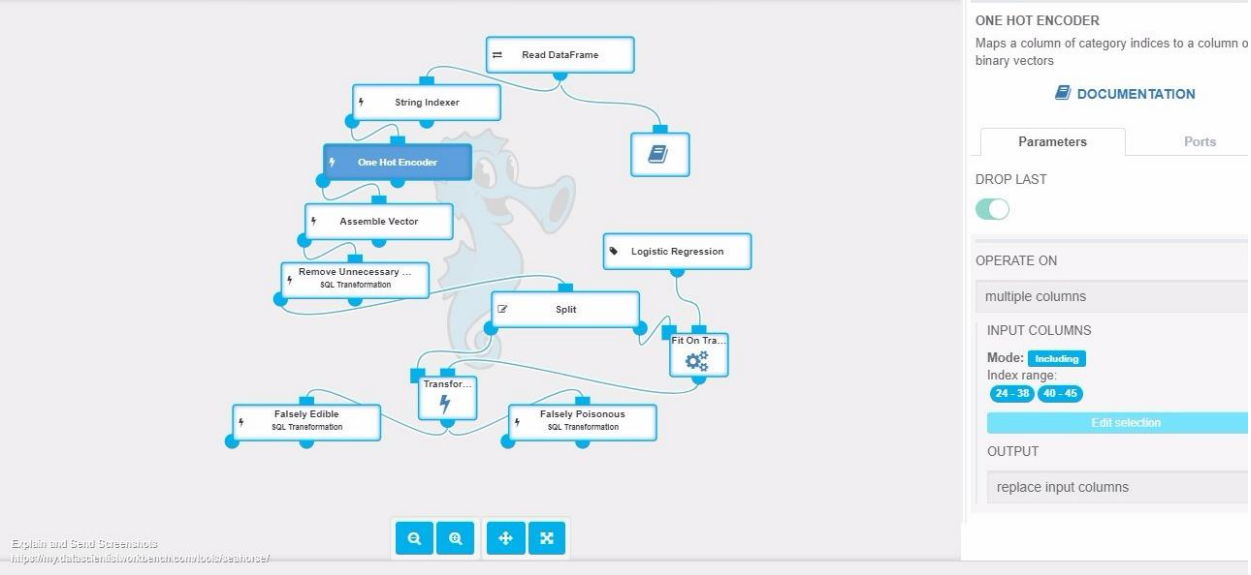


The One Hot Encoder operation translates ordinal values to vector having “1” only at position given by input numeric value. It needs to have its parameters modified, as follows:

OPERATE ON: multiple columns

INPUT COLUMNS: Including index range 24-38 and 40-45

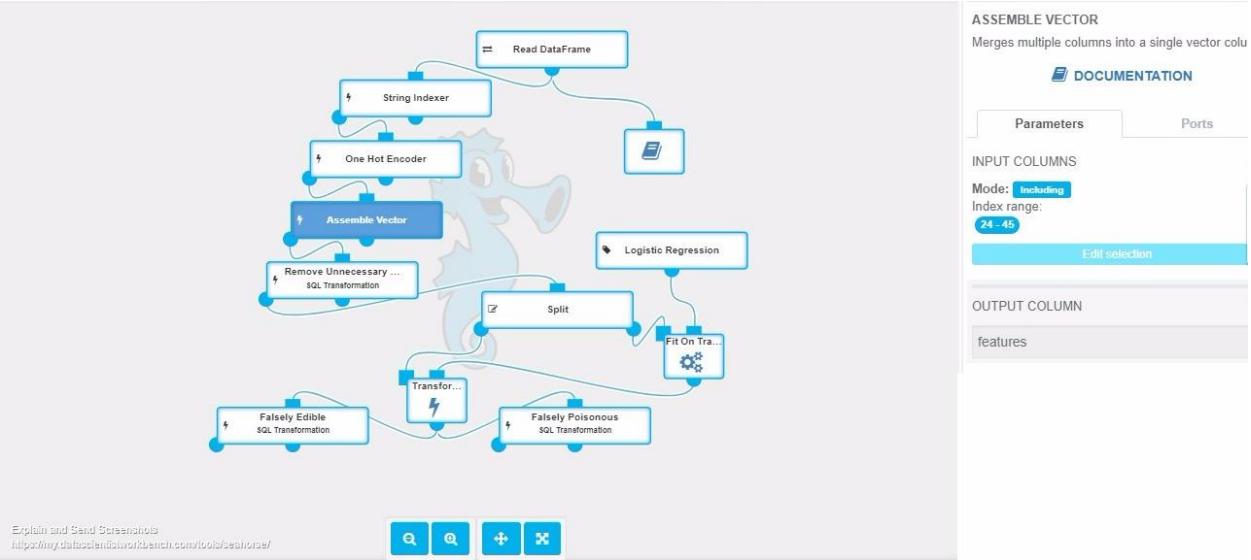
Here I have to excluded the column at index 39 (indexed_veil-type) because all mushroom specimens had partial veil. One Hot Encoder does not allow operating on columns with only one value (unless user wants to drop the last category using DROP LAST parameter).



Assemble Vector merges columns with numerics and vectors of numerics into a single vector of numerics. It needs to have its parameters modified, as follows:

INPUT COLUMNS: Including index range 24-45 (columns generated by the String Indexer, excluding generated column containing edibility label)

OUTPUT COLUMN: features



Removeing Unnecessary Columns

Here I will use the SQL Transformation operation to remove unnecessary columns from the dataset and it gives me more meaningful names to columns that are essential for this experiment. It will make the dataset reports smaller and facilitate exploring data.

Here SQL Transformation needs to have its parameter modified, which I have modified as follows:

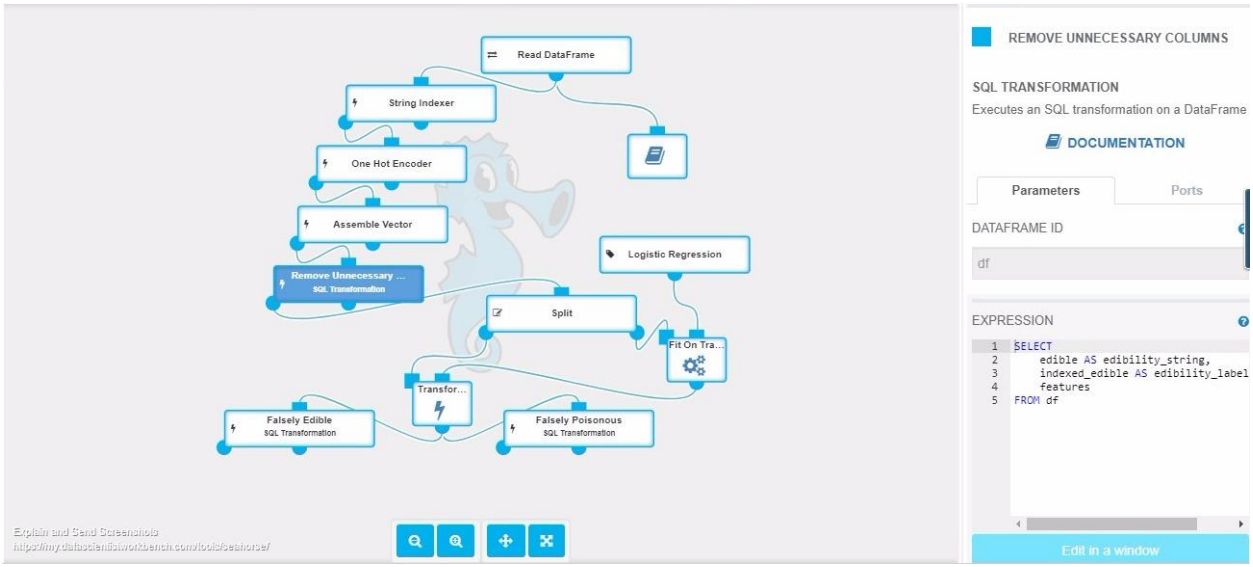
EXPRESSION:

SELECT edible AS edibility_string, indexed_edible AS edibility_label, featuresFROM df

Splitting into Training and Test Set

To perform a fair evaluation of this model, I need to split the input mushroom data into two parts: a testing dataset and a training dataset. That task can be accomplished by using a Split operation. To divide the dataset in ratio 1 to 3, what I will need to do is to modify its default parameters:

SPLIT RATIO: 0.75. This is nothing but the percentage of rows that should end up in the first output DataFrame – the training set.

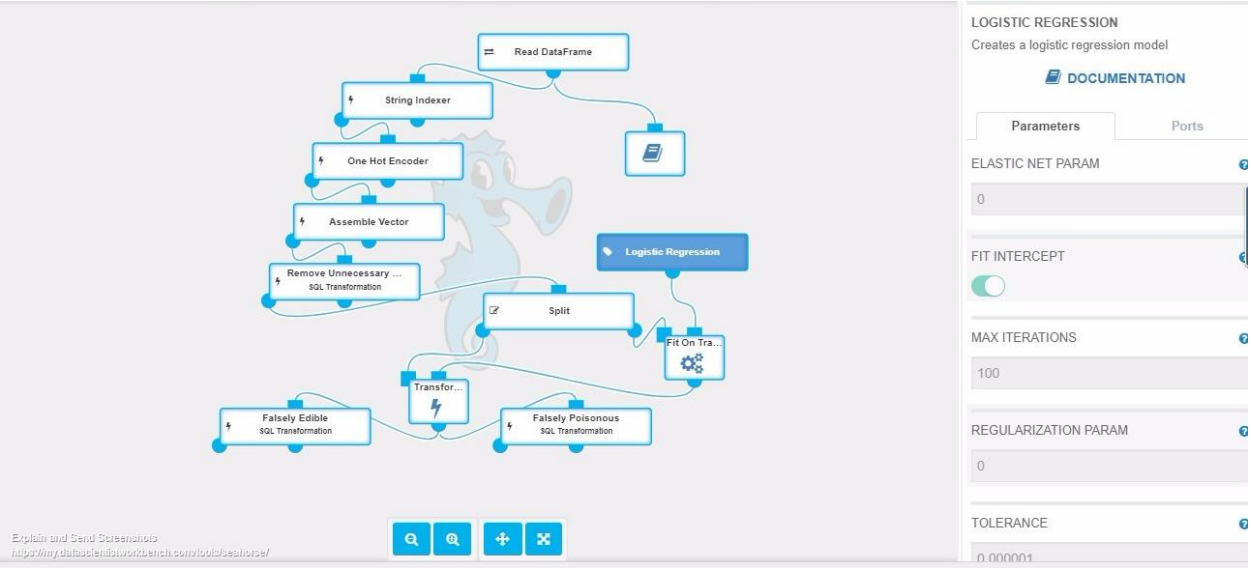


Model Training

To train a model, what I need to do is to use the Fit operation, which can be used to fit an Estimator. I have to use logistic regression classification, so I will put the Logistic Regression operation on the canvas and connect it to the Fit operation.

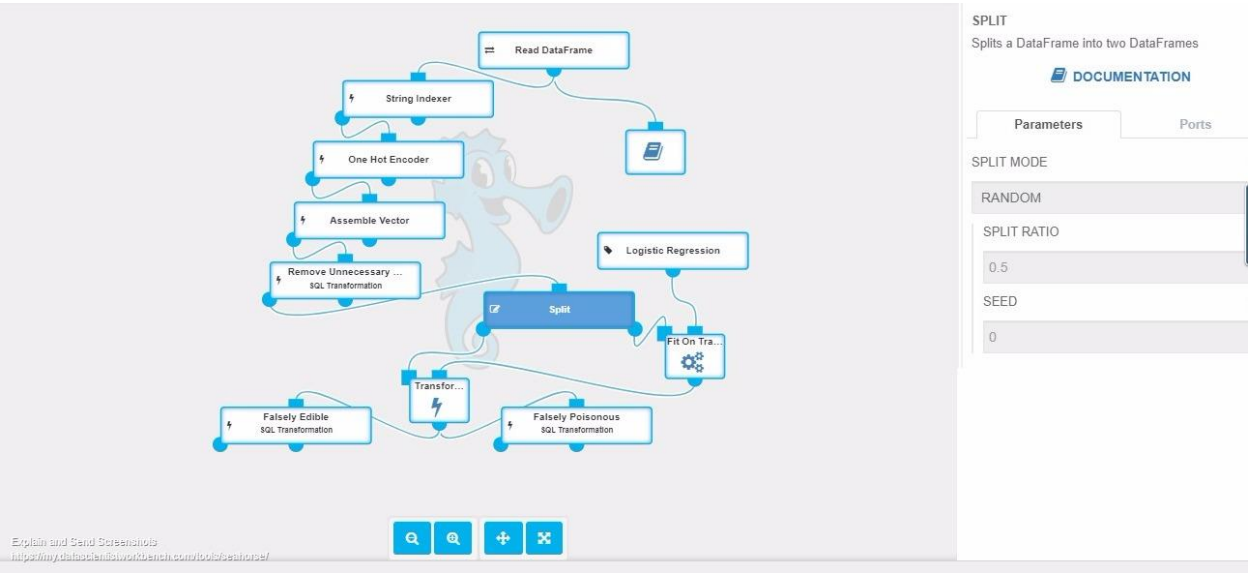
Here I will leave almost all default values of Logistic Regression parameters unchanged, I need only to change the label column to edibility_label.

LABEL COLUMN: edibility_label

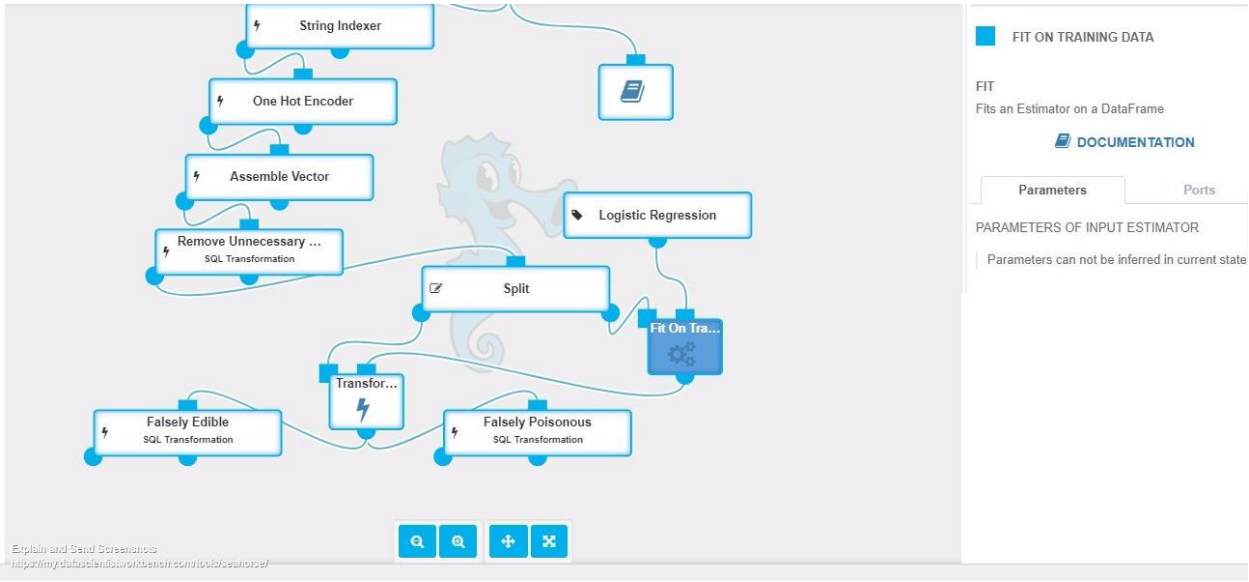


Verifying Model Effectiveness on Test Data

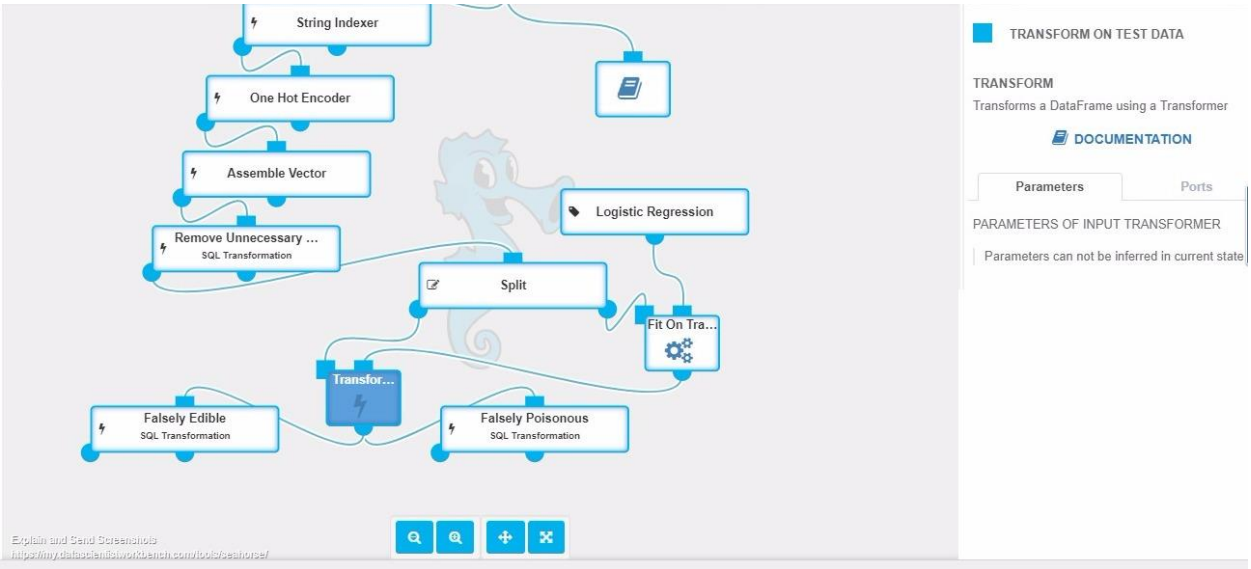
By using Split operation, I have generated a training dataset and a test dataset from the input data. Here I have trained the model on the training dataset. Now it is time to use the test dataset to verify the effectiveness of our classification model. To generate predictions using the trained model, I need to use a Transform operation. To assess effectiveness of the model I will count “Falsely Poisonous” (waste of edible mushrooms) and “Falsely Edible “(very dangerous) entries in the test dataset. To perform the calculations, I will use the SQL Transformation operation.



Here I am being thankful for being able to project only the necessary columns, the resulting dataset fits in the column number limit and I am able to view the data sample. I can notice that the test dataset has 2057 entries. :



I can also see that the String Indexer operation assigned 0 to “e” label (edible class) and 1 to “p” label (poisonous class). I can also notice that the prediction column has “almost the same” values as “edibility” column, so I can suspect that the model performs well. Let’s measure its performance



The SQL Transformation (Falsely Edible) needs to have its parameter modified, as follows:

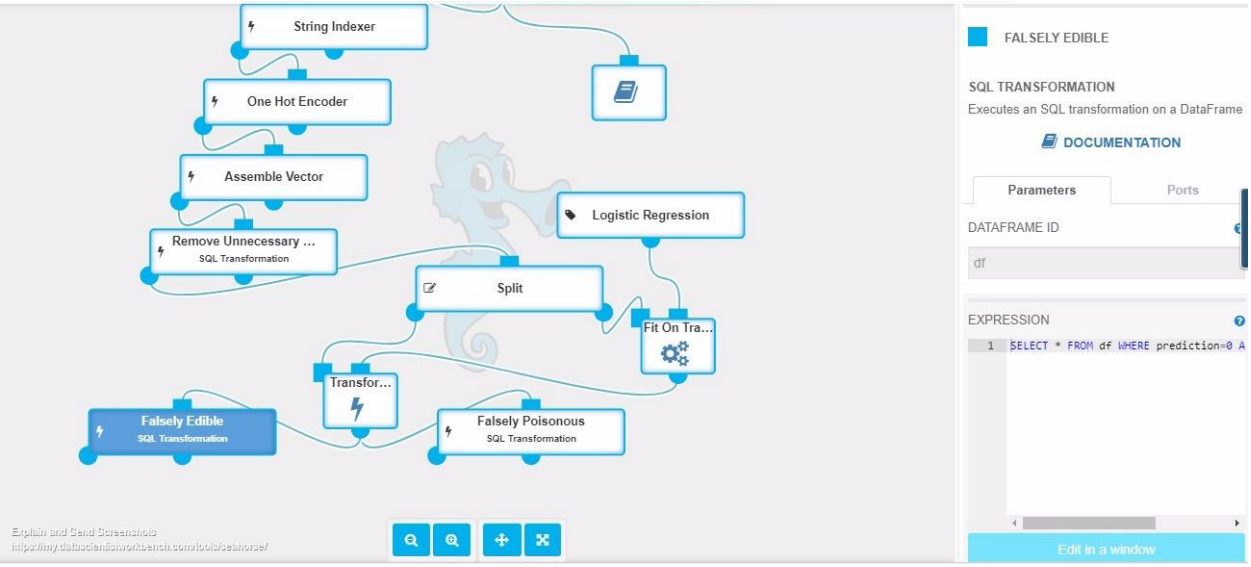
EXPRESSION:

SELECT * FROM df WHERE prediction=0 AND edibility_label=1

The SQL Transformation (Falsely Poisonous) needs to have its parameter modified, as follows:

EXPRESSION:

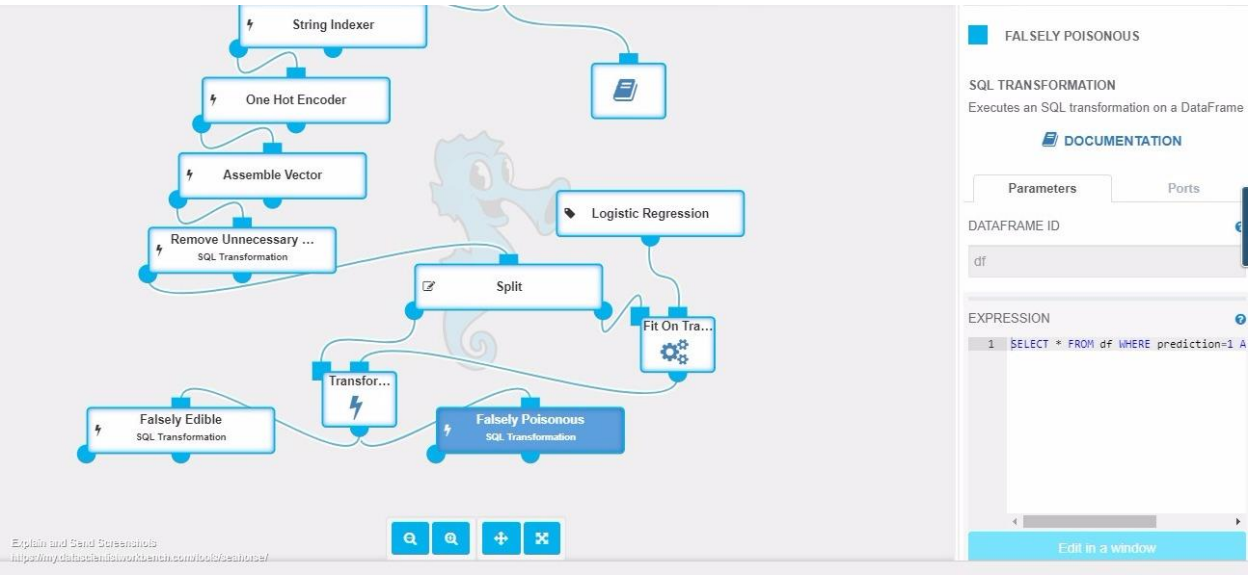
SELECT * FROM df WHERE prediction=1 AND edibility_label=0



Now I can explore reports in these two operations and compare the number of rows in each of them:

Falsely Poisonous: 0

Falsely Edible: 0



Conclusion

Here, it means that I have been able to train a surprisingly accurate prediction model for classifying mushrooms. I have to remember that some dangers remain such as:

- Mushroom picker examining the specimen can make a mistake during assessment of traits or entering data to the computer.
- Data sample might be too small to create a comprehensive model for predicting edibility of all mushrooms that users will want to classify.
- Two different mushroom species can have identical traits, while one of the species is edible and the second one is poisonous.

Due to those problems, I can use this model only to aid expertise on mushroom edibility. Experts would however have the last word over issues that are potentially dangerous for other people, like classifying poisonous mushrooms.