Lab: Watson Explorer for Watson Studio Local (WEX for WSL)

*October, 2018*

*Author: Keith Doan keith.doan@au1.ibm.com*

**Table of contents**

Contents

[Overview 1](#_Toc526795408)

[Required software, access, and files 1](#_Toc526795409)

[Part 1: Introduction 1](#_Toc526795410)

[Scenario Overview 2](#_Toc526795411)

[Business Use Case: Customer Service at a Food Retail 3](#_Toc526795412)

[Part 2: Create WEX Collection for Feature Engineering 3](#_Toc526795413)

[Create a new WEX Collection from Project Dataset 3](#_Toc526795414)

[Verify the new WEX collection in SPSS Modeler 8](#_Toc526795415)

[Verify the new WEX collection in Jupyter Notebook 9](#_Toc526795416)

[Part 3: Build Document Classification Model 11](#_Toc526795417)

[Build Document Classification Model in Modeler Flow 11](#_Toc526795418)

[Build Document Classification Model in Notebook 11](#_Toc526795419)

[Part 4: Model Management and Deployment 12](#_Toc526795420)

[Part 5: Text Feature Selection in WEX Collection 12](#_Toc526795421)

[Conclusion from the Analysis 17](#_Toc526795422)

# Overview

In this lab you will learn how to extend analysis to unstructured data within Watson Studio Local. You will learn how use text as an input and create SPSS prediction models visually. You also learn how to create and share notebooks that contain live code, equations, visualizations and narrative text along with WEX static scores (correlation, trends, etc.).

You can learn more about WEX for Watson Studio Local in the official product documentation: https://content-dsxlocal.mybluemix.net/docs/content/local/wex.html

# Required software, access, and files

* To complete this lab, you will need access to a Watson Studio Local cluster.
* You will also need to download and unzip this GitHub repository: <https://github.com/KeithDoan/DSX_Local_Workshop_12>

# Part 1: Introduction

IBM Watson Explorer for Watson Studio Local (WEX for WSL) allows you to analyze text data in Watson Studio Local (WSL). Its natural language processing capability can extract key information from text data as if it were structured data. Therefore, you can use text data along with the other structured data for better business decision.

With WEX for WSL, you can do the following tasks.

* Ingest data to a collection from WSL data sets.
* Enable natural language processing (NLP) capabilities to annotate part-of-speech, named entities, and sentiment expressions.
* Customize a dictionary to extract key information from text based on your interest.
* Understand data in a collection using a visual content mining tool.
* Use text analysis capabilities from Data Science Experience Local tools such as Notebook and SPSS Modeler.

## Scenario Overview

1. **Develop WEX collection for Feature Engineering**
   1. Create Watson Explorer collection asset
   2. Load data from WS Local dataset (CSV)
   3. Configure NLP processing
   4. Save the collection
2. **Develop predictive model in SPSS Modeler Flow**
   1. Create a new SPSS Modeler Flow
   2. Insert data node from WS Local dataset (CSV or connection)
   3. Insert WEX node and set the collection created at step 1.
   4. Insert Classification node
   5. Insert export node to export the classification result as Watson Studio Local dataset
   6. Train the model
   7. Save the flow
3. **Develop predictive model in Jupyter Notebook**
   1. Create a new Jupyter Notebook
   2. Load data from Watson Studio Local dataset
   3. Use WEX to extract features from the unstructured text
   4. Use a Classifier, Train & Save the model
4. **Use the classification in business process**
   1. Deploy the classification model as REST API or batch job

## Business Use Case: Customer Service at a Food Retail

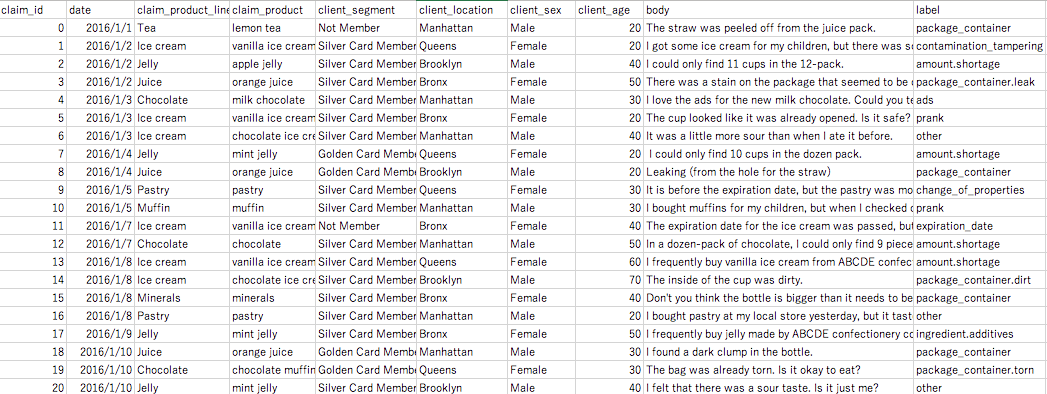
Laura, customer service in a food retail company gets bunch of daily customer reports for their products She looks into each report and transfer to appropriate department.

For example, the report about liquid leaks need to be transferred to packaging department so that this department can perform further investigation and take actions.

Because late report transfer causes late actions, timely and correct transfer is important.

Ryan, data scientist in the company, think he can improve the process by adopting automated classification with machine-learning using Watson Studio Local.

The daily customer report looks like this:



# Part 2: Create WEX Collection for Feature Engineering

## Create a new WEX Collection from Project Dataset

1. Log in to a **Watson Studio Local cluster**.
2. Select the **Datasets** to add dataset “sample\_voc.csv” and “small\_sample\_voc.csv” under folder “wex\_lab/data”
3. Select the **Watson Explorer Collections** and add a new Watson Explorer Collection called “RetailStoreVoCAnalysis” as per figure below:

*Note: It would be recommended to give a meaningful, descriptive name for the collection and ensure the name does not contain spaces.*

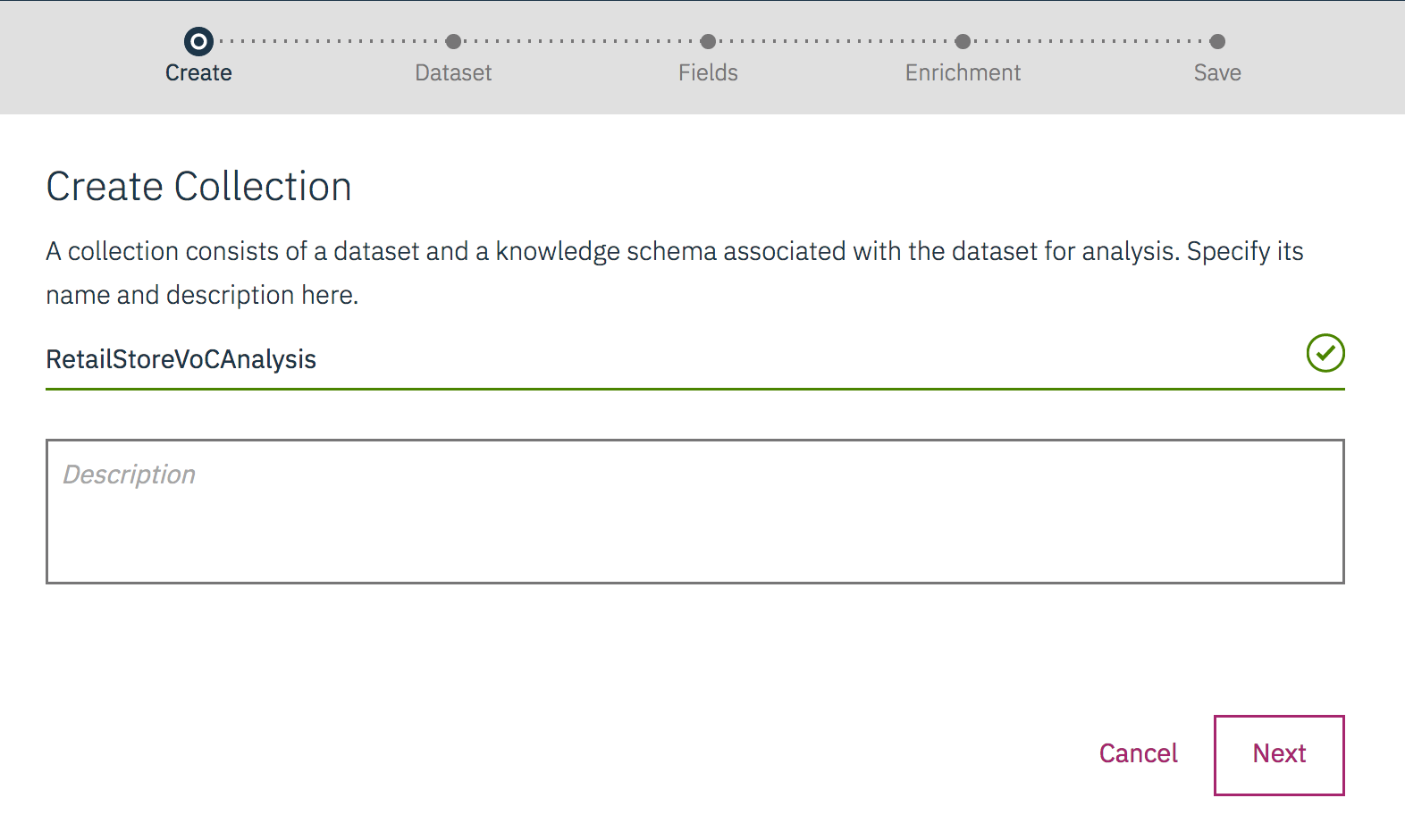


Figure 1 Create Watson Explorer Collection

1. Next, to import the dataset “sample\_voc.csv” (from step 2), as per figure below:

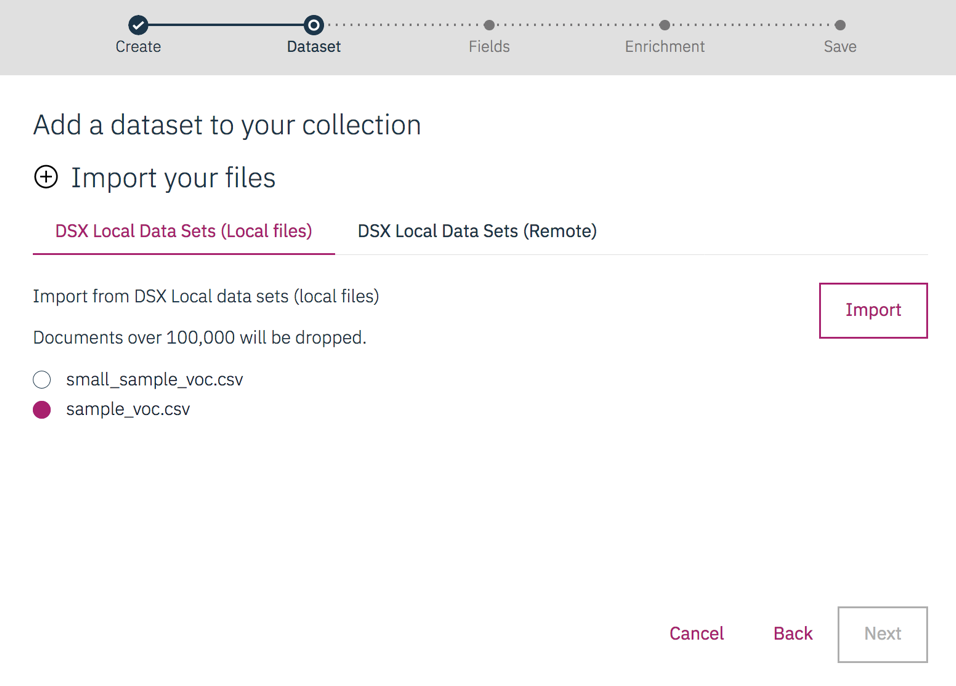
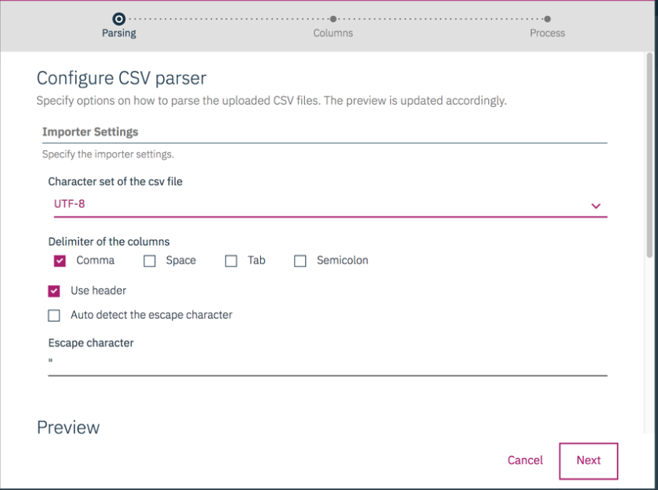
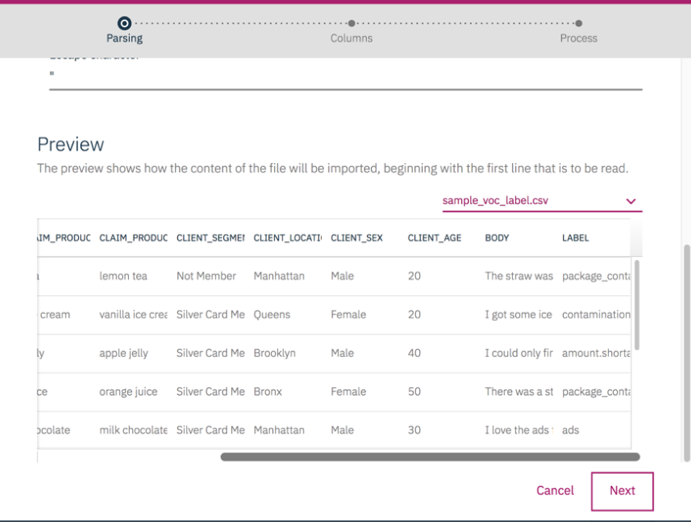


Figure 2 Import file into the Watson Explore Collection

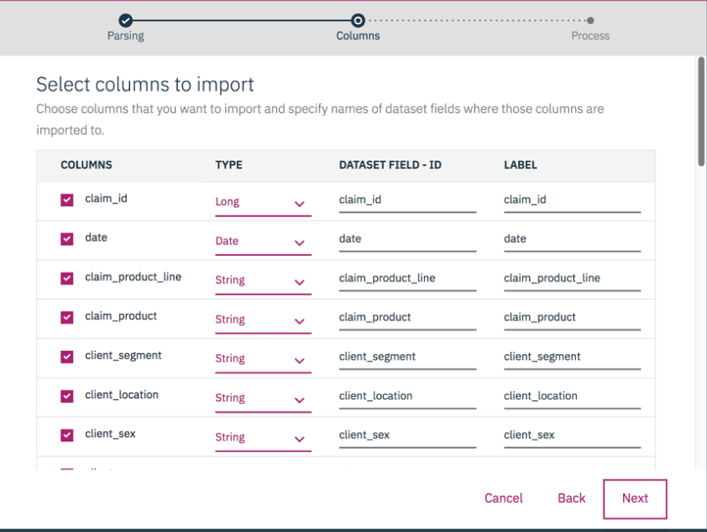
1. Proceed the next three pages, leaving all field settings with their default values, as per figure below.
   1. Configure CSV Parser

**

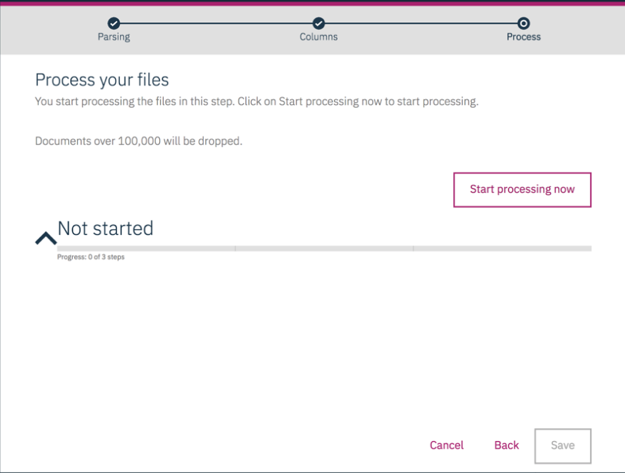
Scroll down to preview the file data and click **Next**:

**

* 1. Select Columns to import and click **Next:**

**

* 1. Select **Start processing now** and once it’s done, select **Save.**

**

1. Configure the collection field with “body for Body field and click **Next**, as shown in the figure below.

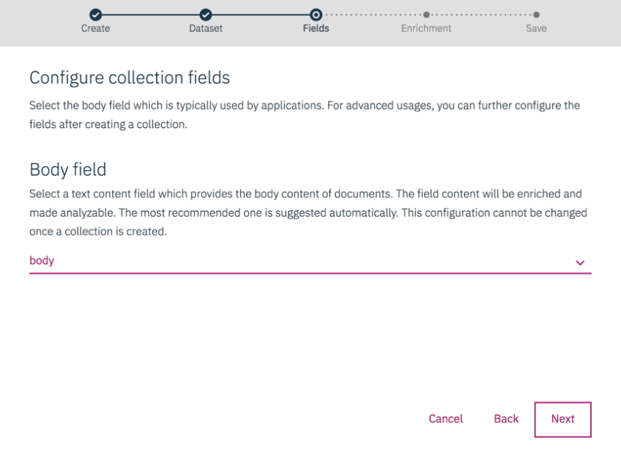
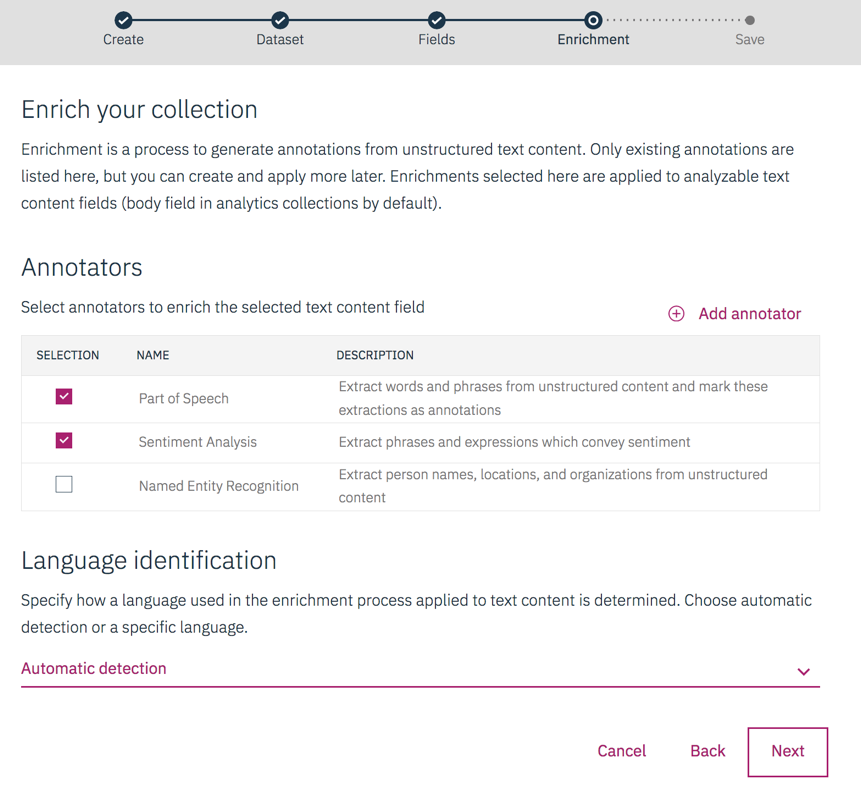
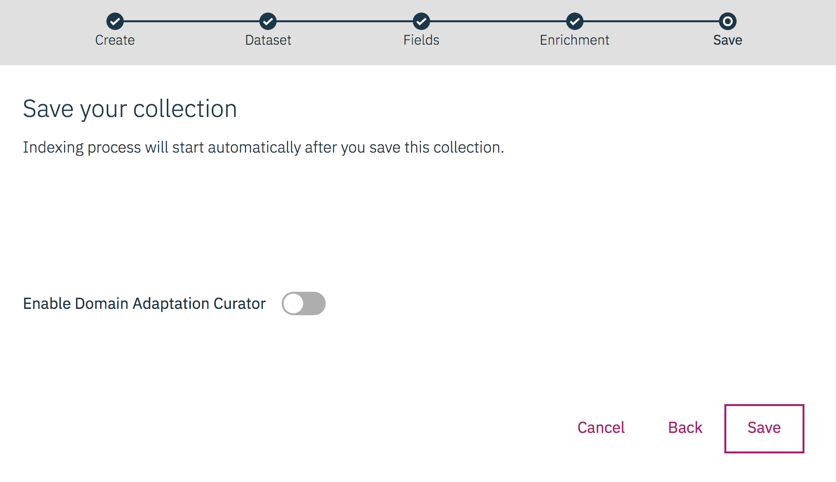


Figure 3 Configure Collection Fields

1. On the enrichments page you can define which enrichments you want to evaluate. For this lab, select "**Parts of Speech**" and "**Sentiment**" and click **Next**, as per figure below.



1. Select **Save**.



It takes about several minutes to index the new Collection.

Once the collection is indexed and ready for the analysis to follow, the page should look like this:

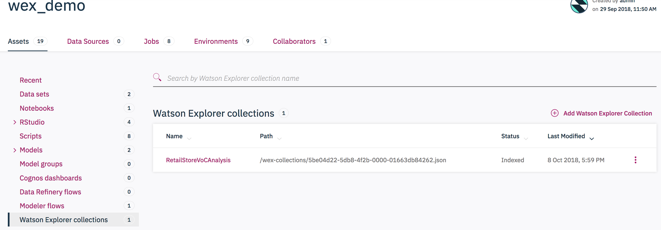
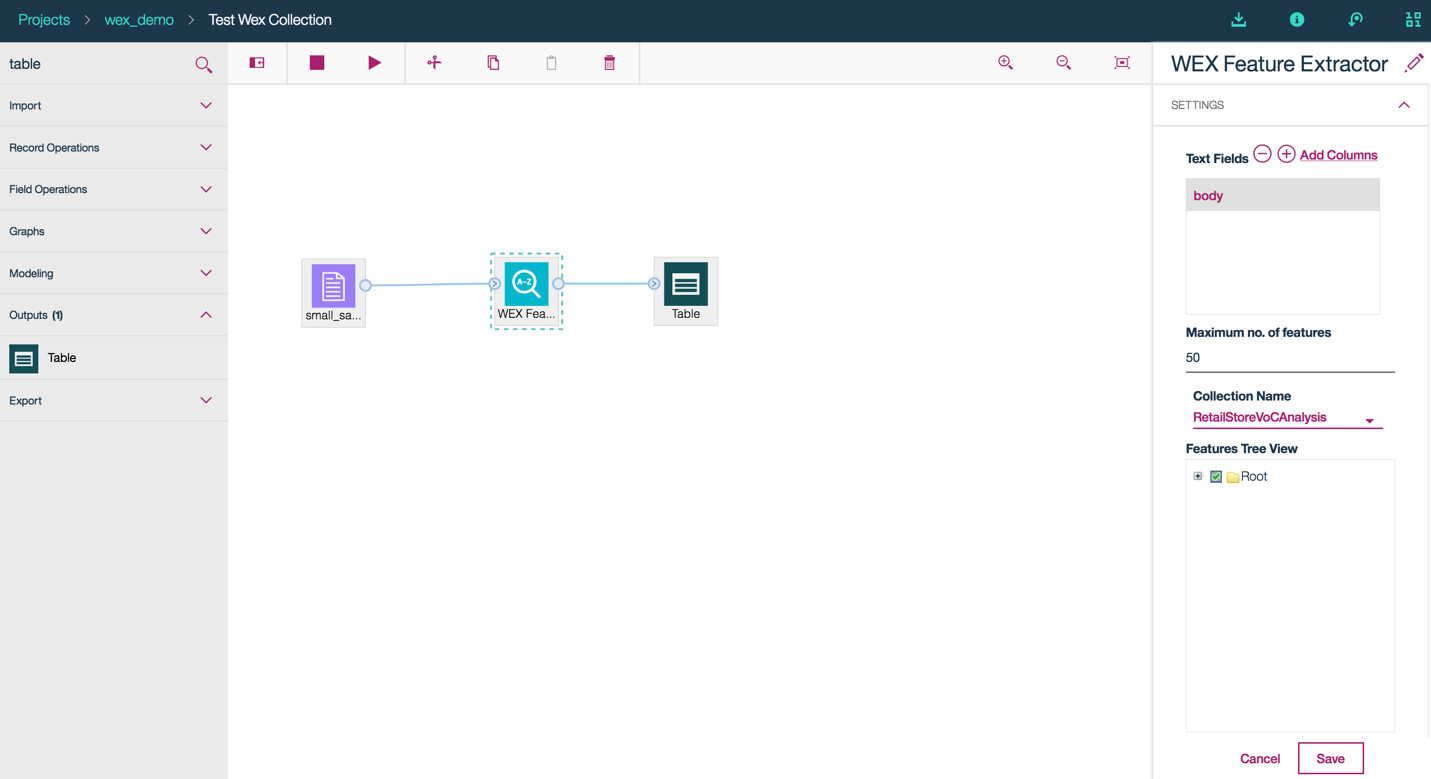


Figure 4 The Indexed Watson Explorer Collection for Analysis

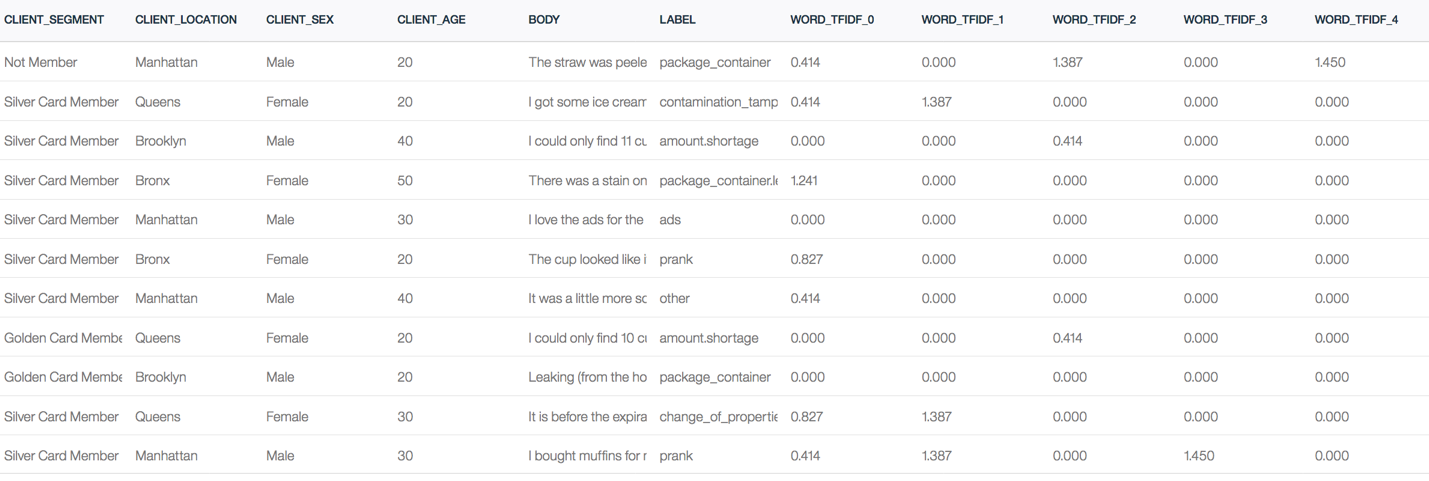
## Verify the new WEX collection in SPSS Modeler

1. Switch to Modeler Flow.
2. Select to add a Modeler Flow to create a new flow called “Test WEX Collection”
3. Once the Modeler empty canvas open, Click Find and Add Data (Finad and add data) in the toolbar, and then select the dataset “small\_sample\_voc.csv” to drag into the canvas.
4. Go to the search node and type “WEX” to find the “WEX Feature Extractor”.
5. Drag the WEX Feature Extractor and connect it from the data source node.
6. Open the WEX Feature Extractor **setting** and select the following:
   1. “Add Columns” to select the text field “**body**”
   2. Collection Name as “RetailStoreVoCAnalysis”
   3. Part of Speech from the Feature Tree View and select **Save**.
   4. Add the Table node to connect it from the WEX node.

The screen would now look like this:



1. Run the table node to see several additional fields with the names staring with “WORD\_TFIDF\_n”, as a result of WEX extraction, as per screenshot.

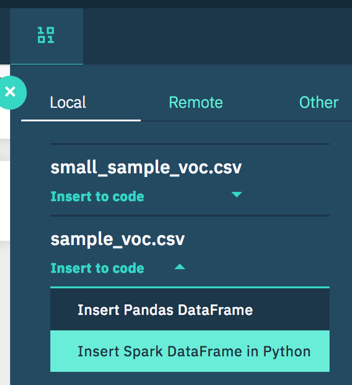


This shows the new WEX collection is ready for text feature extraction in Modeler Flow for analysis.

1. Close the Modeler Flow.

## Verify the new WEX collection in Jupyter Notebook

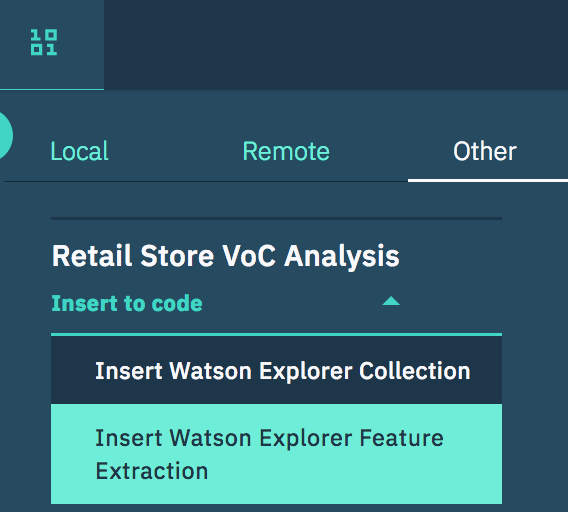
1. Create a new Python 3.5 notebook called “Test WEX Collection”.
2. Click Find and Add Data (Finad and add data) in the toolbar, and then select the Local “sample\_voc.csv” and “Insert Spark DataFrame in Python” into a new cell.



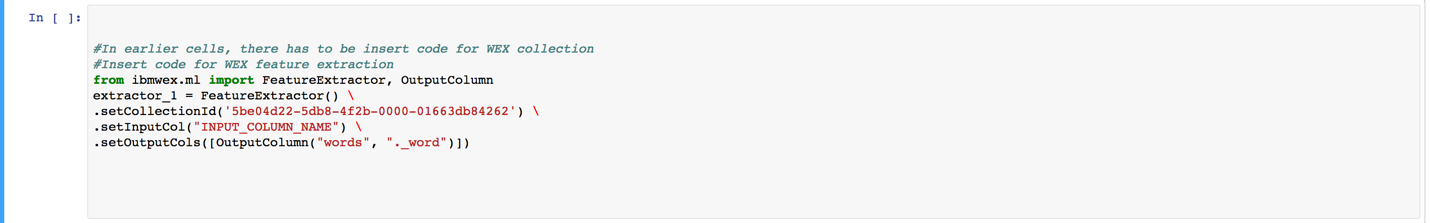
The resulted code is as shown below:



1. Then select Other and select “RetailStoreVoCAnalysis” and “Insert Watson Explorer Feature Extraction” into a new cell to access the collection as shown:

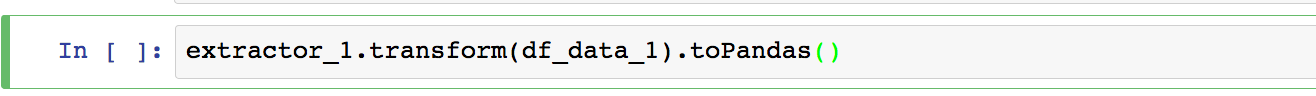


The resulted code is as shown below:



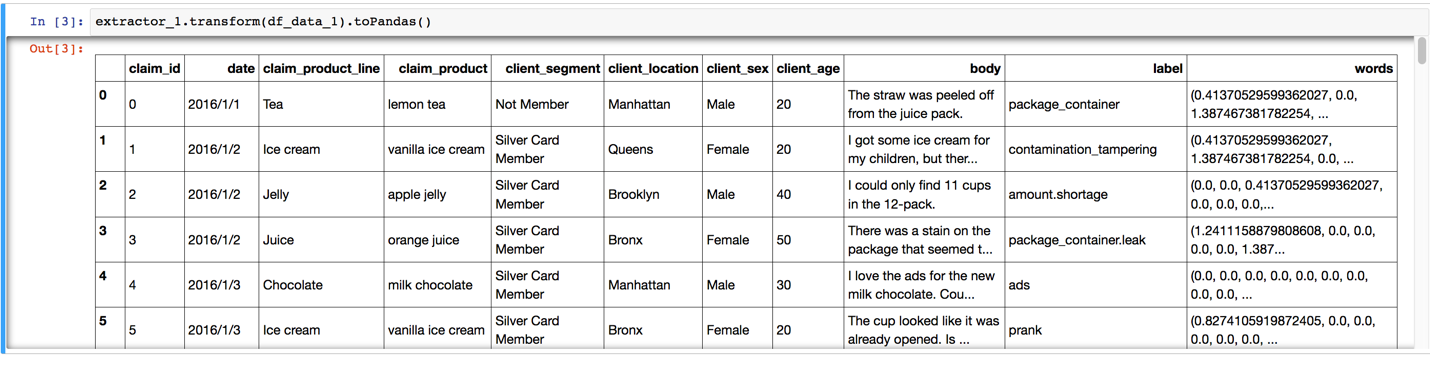
Update the line with .setInputCol(“INPUT\_COLUMN\_NAME”) as .setInputCol(“body”)

1. Insert a new cell and code as per screenshot below:



Note: Since every time you do the step 2 or step 3, the index n in the default variable names: “df\_data\_**n**” and “extractor\_**n**” will increment, so please ensure that you have the right index **n** in the step 4.

1. Run All to get the following output for the step 4:

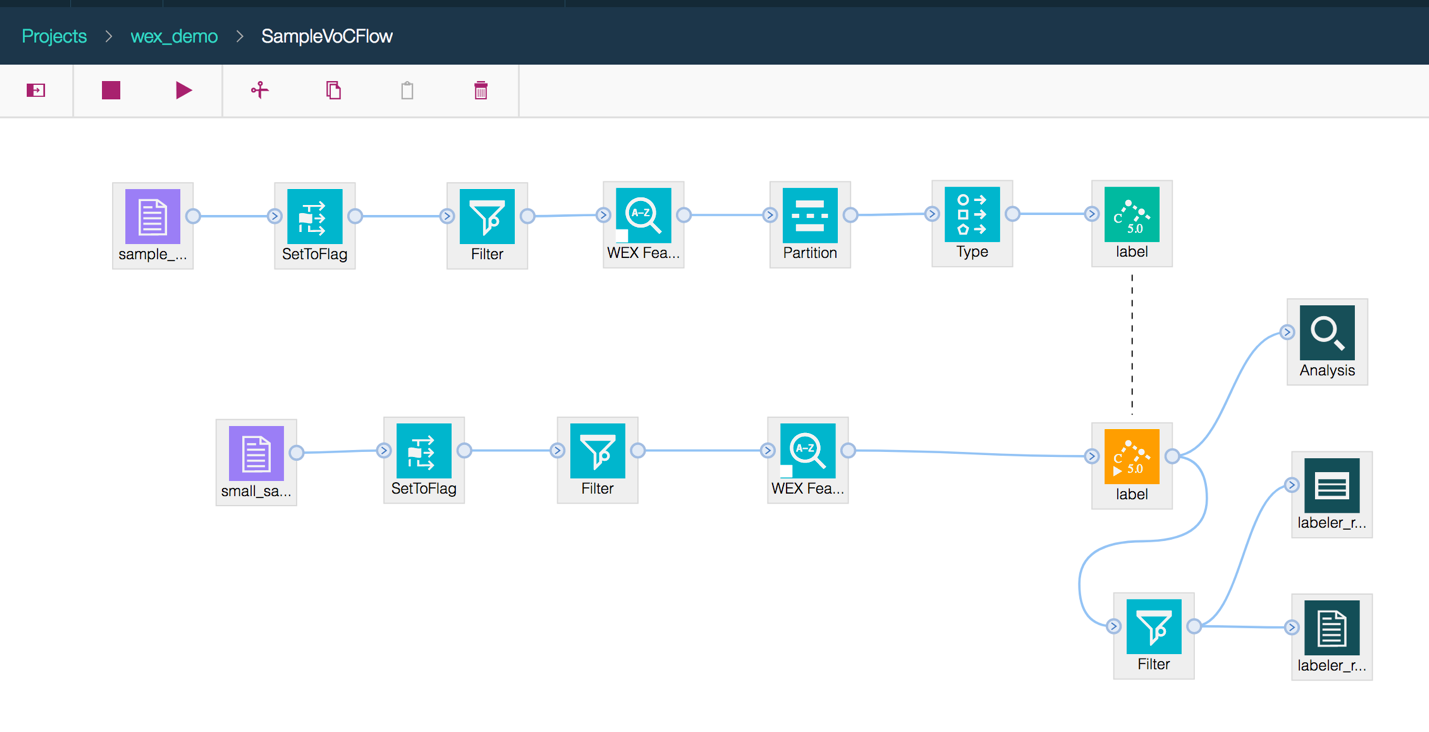


This shows the new WEX collection is ready for text feature extraction in notebook for analysis.

# Part 3: Build Document Classification Model

## Build Document Classification Model in Modeler Flow

1. Import the existing Modeler Flow from the downloaded file “SampleVoCFlow.str” under the folder “wex\_lab/spss”
2. The imported flow would look like this:

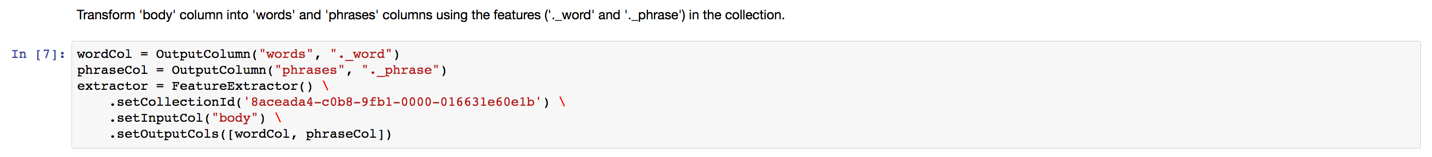


1. Replace the two data source nodes with the datasets in the project: “sample\_voc.csv” and “small\_sample\_voc.csv”
2. Examine the flow and execute to see the resulted table node and the analysis node
3. Save the model nugget “label” as “DocumentClassificationModel\_Flow”.

So you have created a new model using Modeler Flow.

## Build Document Classification Model in Notebook

1. Import the existing Python Notebook from the downloaded file “SampleVoCNotebook” under the folder “wex\_lab/notebook”
2. Examine the notebook
3. Replace the collection id with your collection id in the following code:



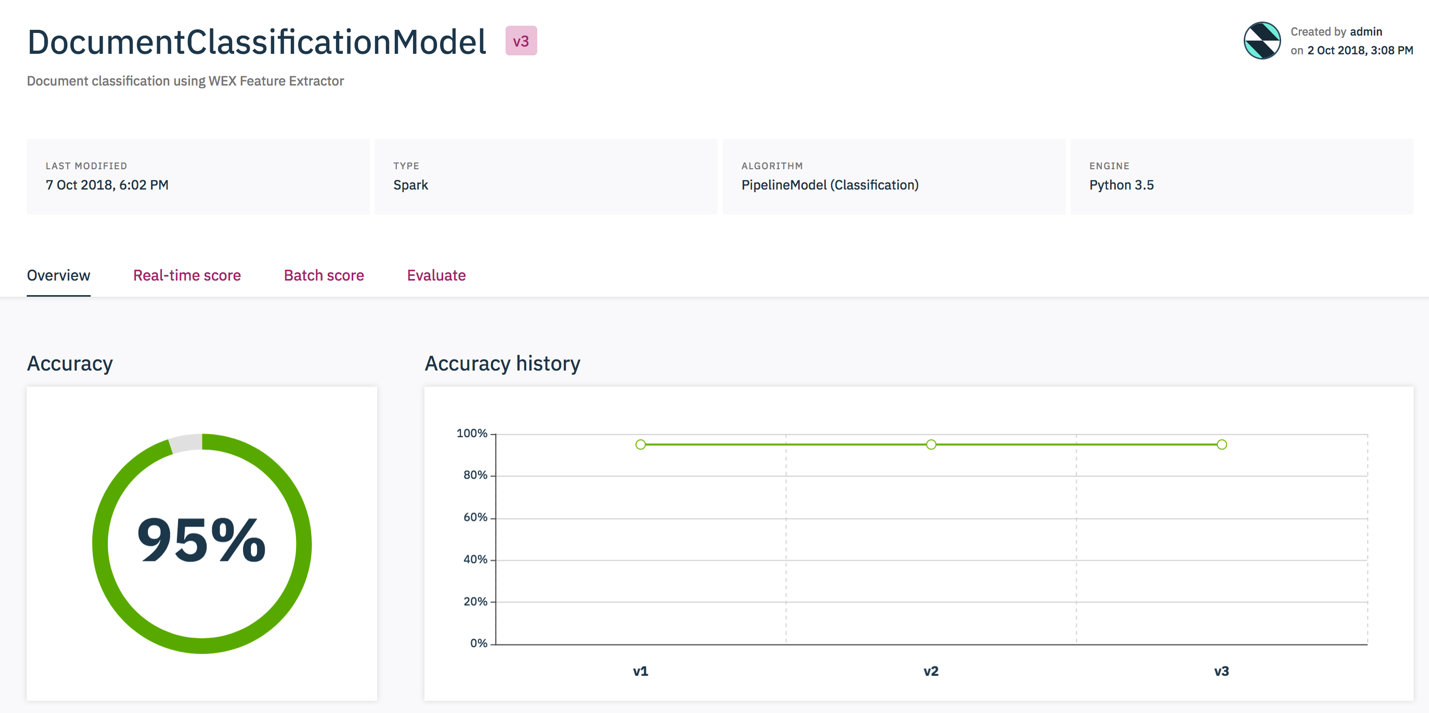
1. Run All cells to create the new DocumentClassificationModel as per following code:



So you have created a new model using Notebook.

# Part 4: Model Management and Deployment

1. Switch to Models section, you will see two models created from Modeler Flow and Notebook in Part 3.
2. Select to open the DocumentClassificationModel:

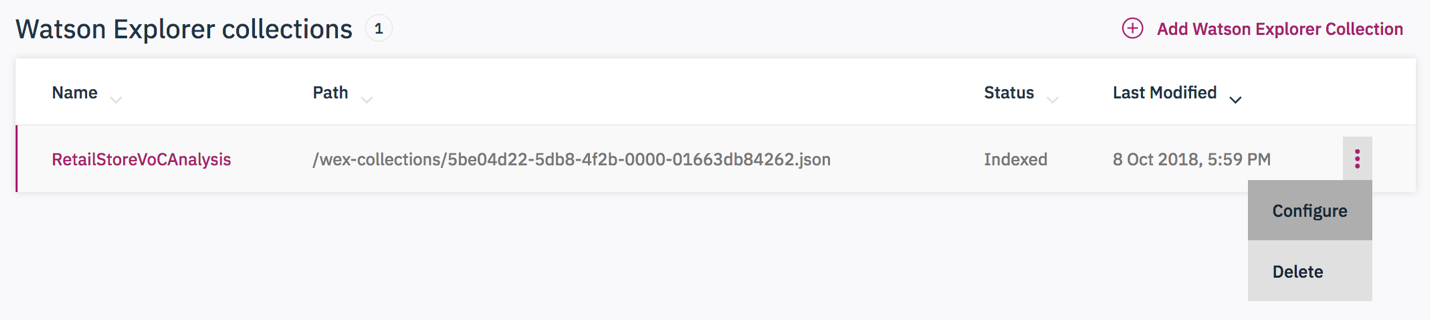


Next, you schedule to evaluate the model and deploy it through project releases.

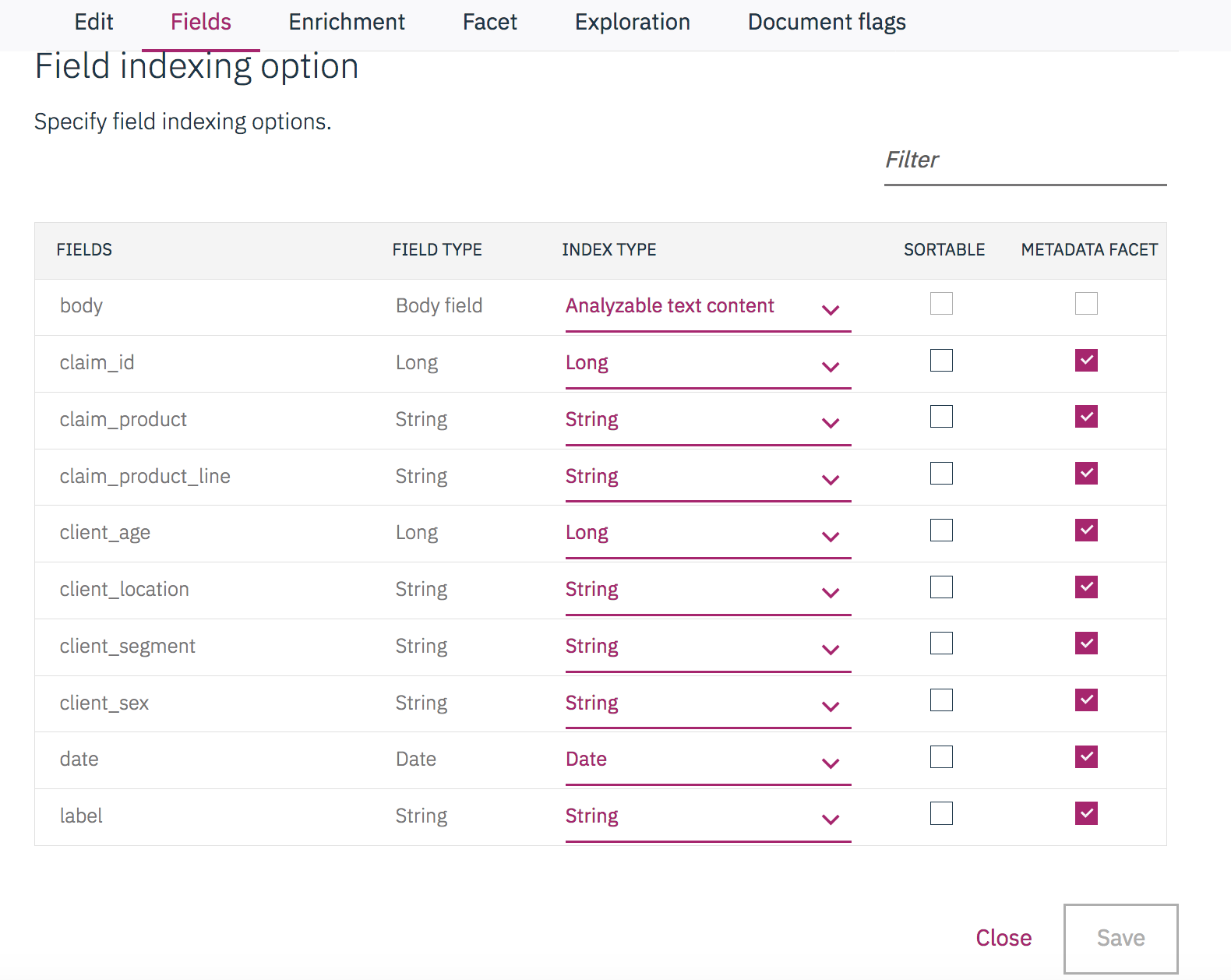
# Part 5: Text Feature Selection in WEX Collection

Since you wish to perform feature selection for predicting the label, you need to include label as the Metadata Facet in the WEX Collection for the correlation analysis between label and other features.

1. Switch to Watson Explorer collections, select to configure the indexed Collection “RetailStoreVoCAnalysis” as per screenshot:



1. Select Fields tab in the Configuration and select **Metadata Facet** checkbox to include Label as String and select **Save**.



The Collection is then re-indexed to include the change for a few minutes.

1. Select to open the indexed Collection to examine the extracted elements:

On the right hand side of the screen there is a dropdown, make sure **Metadata Facets** is selected.

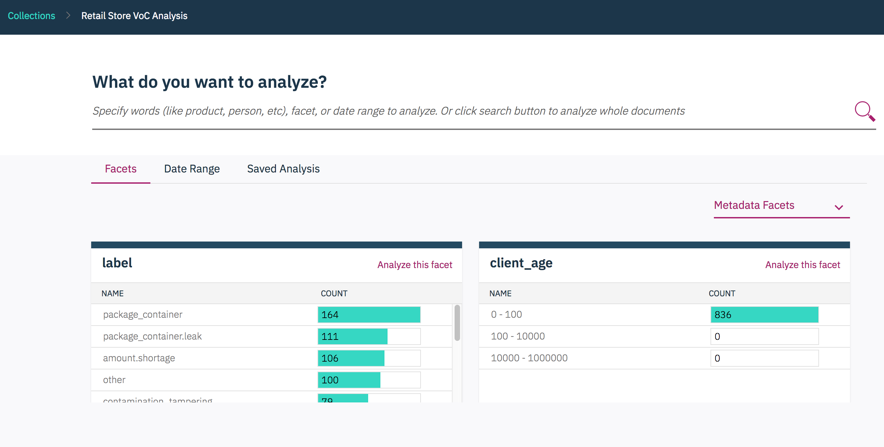


Figure 5 Analyse data with WEX Collection

Go the **label** facet and select “**Analyze this facet”** then click on the magnifying glass to analyse the collection using the selected facet; the page looks like this.

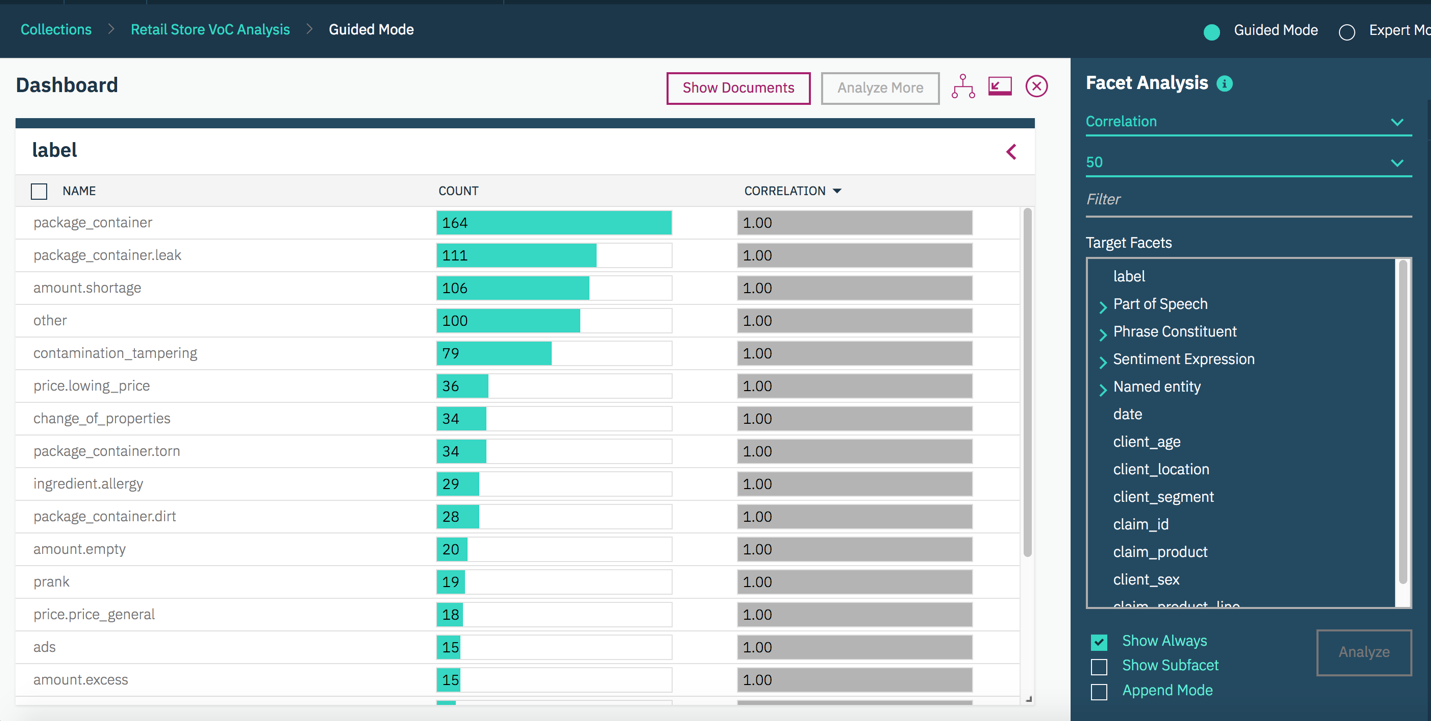


Figure 6 Facet Analysis

1. Have a look at the correlation between the Part of Speech facet and the label facet: you are looking for those that have a high correlation. You do this as follows:

* Switch from "Correlation" to "Pairs" at the top of the Facet Analysis panel
* Select "Parts of Speech" as the row (left side icon) and "label" as the column (right side icon) from the Custom Facet Panel. This displays a matrix (2D Map) of the correlations, as shown like figure below:

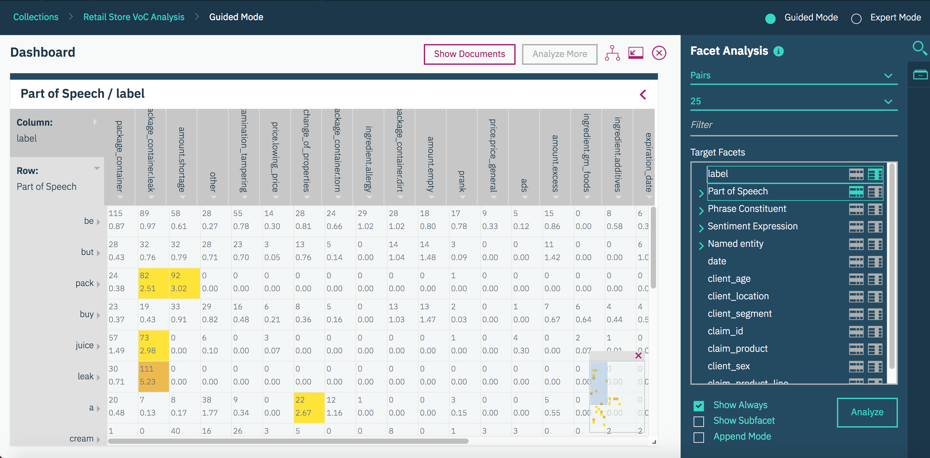
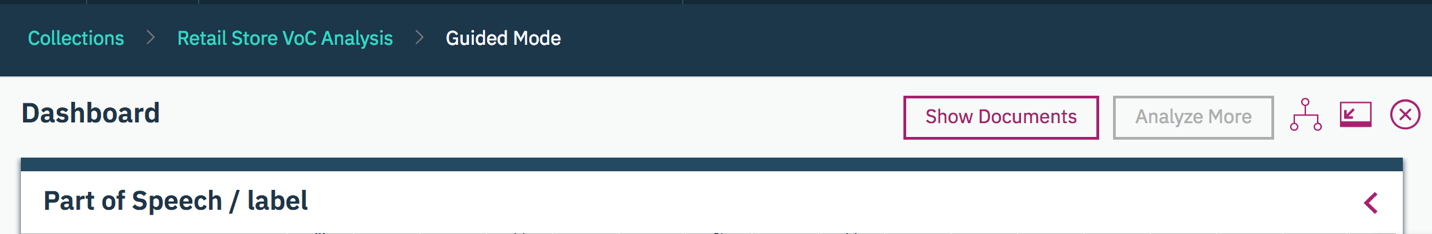
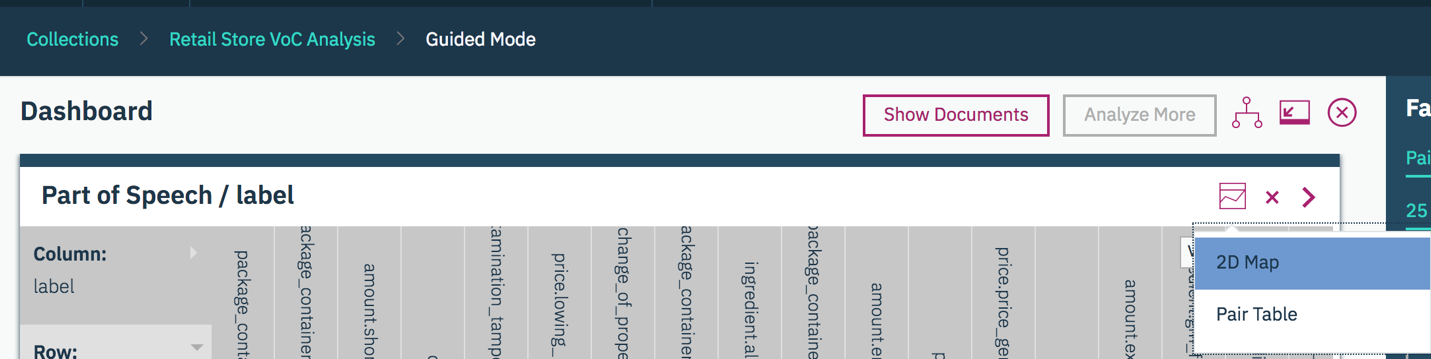


Figure 7 The matrix (2D) of Part of Speech and label Correlations

* Change the visualisation by selecting options for the display

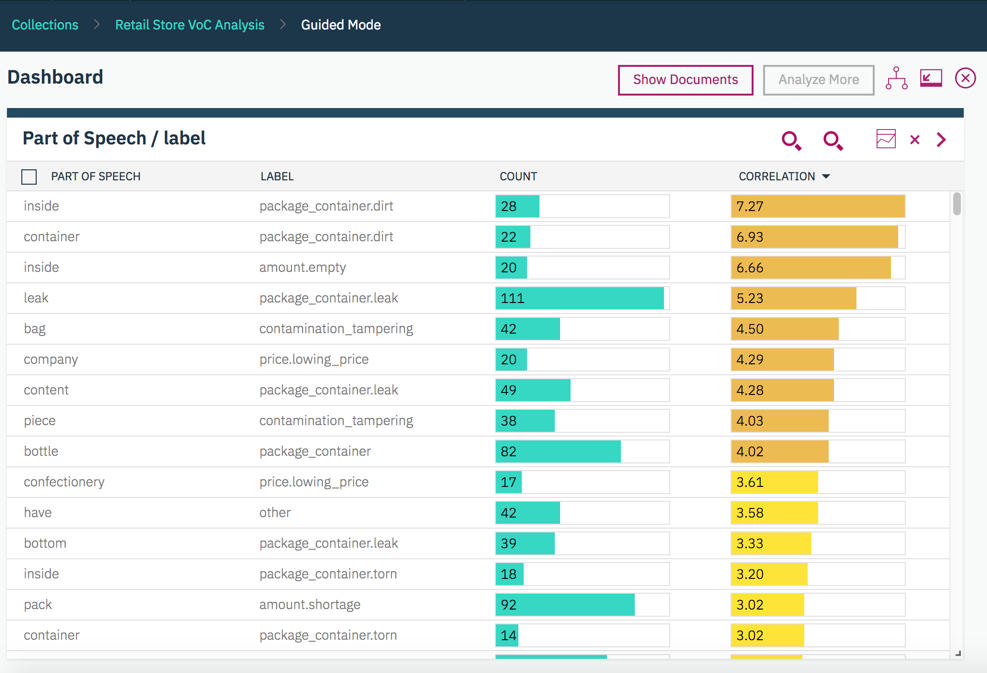


* Select the "Pair Table" option.



Based on the resulted Pair Table visualisation below, you can see here that certain facets pairs (Parts of Speech compared to Label) have a relatively high correlation.

Anything over 1 tells you that the document content relates well to the label and is likely to be a good source for training a classifier. Anything below 1 tells you that the words in the documents may not be helpful to predict labels. However, even in such cases, the multi-label classifier may still work effectively if documents have “helpful” metadata or extracted entity, concepts and relationships.



1. Now you will check if the document metadata and the labels have high correlation values (again using a pair- wise comparison).

Select different document metadata facets (left hand icon in the Facet Analysis Panel) to see which show metadata have high correlation.

If a document metadata and label pair has high correlation (> 1), the document metadata may be helpful to predict labels. If a document metadata and label pair has low correlation (< 1), the document metadata may not be helpful to predict labels.

In this example, the various metadata facets do show some high correlation values, so it is worth keeping the document metadata as training input.

For example, the correlations between Claim Product and Label are shown in this screen shot.

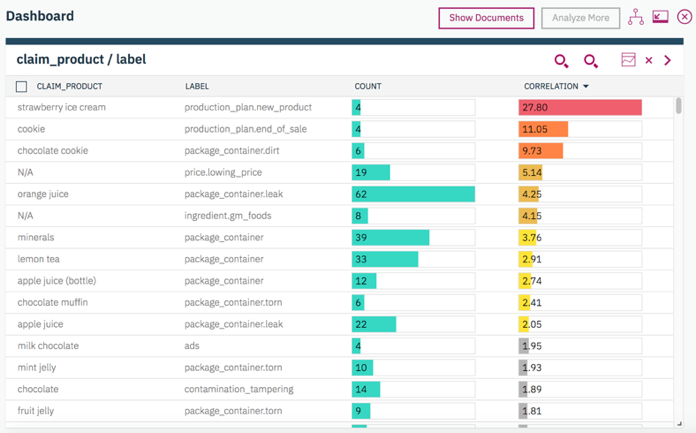
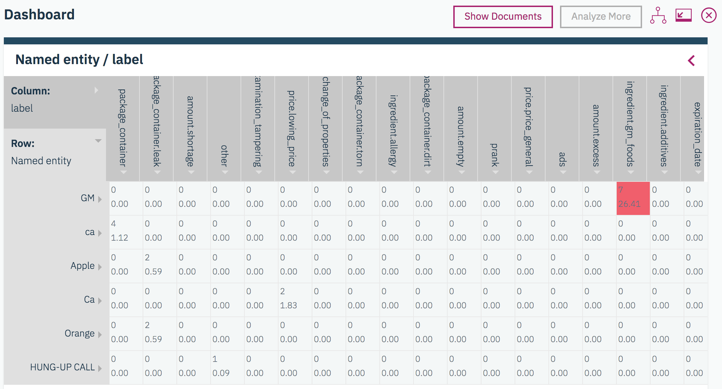


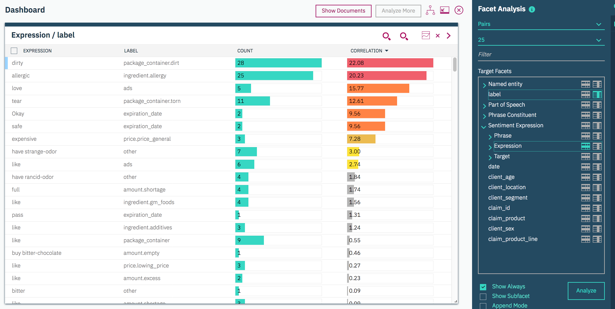
Figure 8 Pairwise Correlations of Claim Product and label

1. Finally, you will check if concepts or relationships in the text (extracted by annotators) and the labels have high correlation values (again using a pair-wise comparison). In this lab you have two annotators to look at: Named Entity and Sentiment.

Named Entity shows a high correlation for only one value (GM), so this is probably not a good enrichment option for training the Classification model.



Sentiment shows a high correlation for several Sentiment Expressions and Sentiment Targets. Thus it may be useful to include the Sentiment annotator when training The Classifier.



As a general rule, classification results will be poor if no Part of Speech or metadata values have high correlation with the labels. If this happens you should re-examine the training data and consider developing a new set, or consider developing some custom annotators to extract entities (concepts and relationships) from the text, that could be used to train a Classifier. Fortunately, in this dataset this is not the case.

## Conclusion from the Analysis

There is likely enough data to train an effective document classification model. The following elements can be used in the training process: Document content (that is the parts of speech enrichment), Document metadata and Sentiment Expression.