# Lab 9

In this laboratory, we will apply some classification algorithms to the Amazon fine-foods dataset we have been exploring so far, exploiting SparkSQL to create the input DataFrame of the machine learning algorithms.

Your goal is to produce a classification model able to label reviews as “useful” or “useless” (i.e., you will produce a binary classification model). For each review on Amazon.com, users are able to vote the helpfulness of such review, with a thumb up/down. Our dataset provides this information through two columns:

- Column HelpfulnessDenominator: the number of users that have voted the content of a review (helpfulness denominator)
- Column HelpfulnessNumerator: the number of users that have thumbed up (helpfulness numerator).

The helpfulness index of a review is given by the ratio of the two (HelpfulnessNumerator/ HelpfulnessDenominator), and it can be computed only for the reviews that have been voted at least one time (HelpfulnessDenominator>0). For this task, a review belongs to the “useful” class if its helpfulness index is above 90% (0.9). Otherwise, the review is labeled as “useless”.
We are interested in training a classification model that can predict the value of the class label (useful or useless).

There are different ways to evaluate the quality of the built classification model. Here, we choose one of the simplest: we divide the labeled input dataset (i.e., the reviews for which the label is known) in two splits, we train the dataset on the first one and then we test it on the second part, computing the average precision of the result.

For your ease, you are provided of a template (lab9_template.ipynb) to fill, which covers already the evaluation part (split of the dataset and testing phase, print of the quality metrics).
Your task is to:

1. Read and preprocess the dataset an store it into a DataFrame;
2. Create a Pipeline that builds a classification model that can predict the value of the class label attribute (i.e., it predicts if a review is useful or useless).

# Step 1 Preprocessing

The Amazon fine-foods dataset contains a large set of reviews (/data/students/bigdata-01QYD/Lab9/Reviews.csv).

Since not all the reviews have been rated, you must filter out the reviews that have never been rated (i.e., those having 0 as helpfulness denominator) before creating the DataFrame that you will use to train your classification model.

The next step consists in generating a DataFrame characterized by two fields: label (it is the target attribute of the classification problem) and features (it is a vector of doubles containing the values of the predictive attributes that are used to infer the value of the target attribute). Each record of the data frame is a labeled record and the DataFrame can be used as input of a classification algorithm to infer a classification model.
To this end, you have to manipulate the data to obtain the correct label column and the features column by means of a set of Transformers and/or Estimators (e.g, SQLTransformer, VectorAssembler).

- In this task, we set the value of label to 1.0 for representing the class label “useful” and to 0.0 for representing the class label “useless”.
- For the (predictive) features, we initially choose as only feature the length of the field “Text” (i.e., the vector features contains one single double value for each record that is set to the length of “Text”).

# Step 2 Training step

Now create a Pipeline for generating the classification model. Remember that one of the steps of the pipeline will be a classification algorithm (e.g., a logistic regression algorithm or a Decision Tree), and that you will probably need other pre-processing steps to prepare the content of Column features.

When you have created the pipeline, run and take note of the final prediction quality of the built model displayed on the stdout by the evaluation code provided at the end of the template file.

Create two versions of your application. One version based on the Decision tree algorithm and another one based on the Logistic regression algorithm. Finally, compare the quality of the two generated models.

# Step 3 Adding features

Now it is your turn to improve the classifier you have implemented so far. Try to generate 3-4 other features (to be included in the features vector) based on the original review features, and re-run the pipeline. Generate the features that you think could be useful for predicting the value of the class label (useful/useless). What is the quality of your classification model?

# Step 4 Use the content of the review

Try to create a classification model based on the content of the field “Text” (i.e., on the words appearing in Text).
Is the quality of the generated model better than the previous ones?