Data Science Workshop

Classification Analysis

Toulon, February 2019

Emmanuel Génard – *genard@fr.ibm.com*
Data Scientist & Cloud Developer Advocate Europe,
IBM Business Solution Center Nice, France

ISEN
ALL IS DIGITAL!
MÉDITERRANÉE

Objectives

- What is Classification?
- What are the types of Classification techniques?
- Why do we use Classification?

- K-Nearest Neighbors
- Decision Trees

What is classification?

Classification is the task of ***learning a target function*** **f** that maps attribute set **x** to one of the predefined class labels **y**

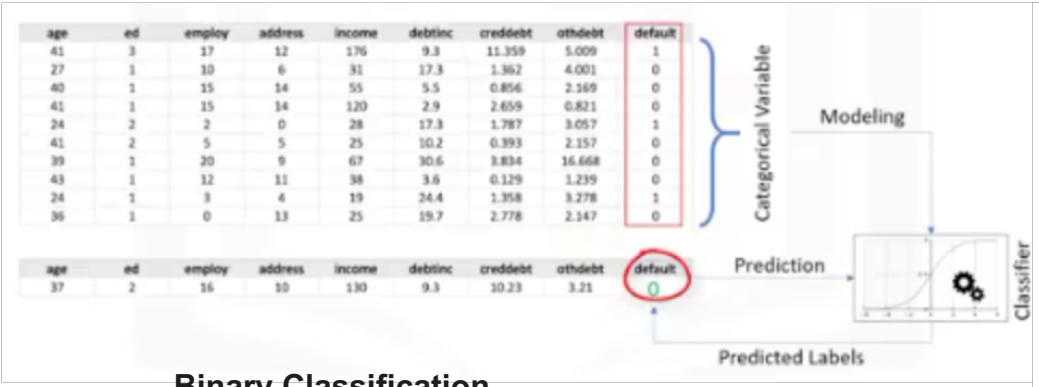
	binary	categorical	continuous	class
Tid	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Figure 4.6. Training set for predicting borrowers who will default on loan payments.

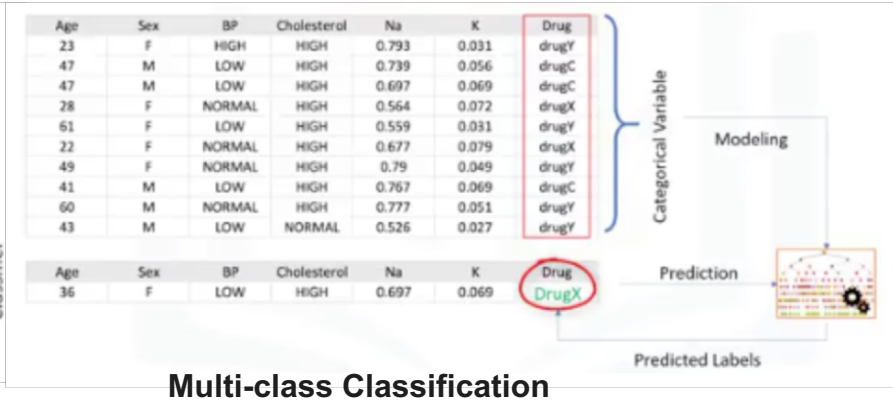
Why Classification?

The target function **f** is known as a ***classification model***

- **Descriptive modeling:** ***Explanatory tool*** to distinguish between objects of different classes (e.g., description of who can pay back his loan)
- **Predictive modeling:** Predict a class of a previously ***unseen*** record



Binary Classification



Multi-class Classification

Typical Classification Applications

- **Credit Approval**
- **Target Marketing**
  - Predict whether or not a customer responds to a particular advertising campaign
- **Medical Diagnosis**
- **Treatment Effectiveness analysis**
- *For email filtering, speech recognition, handwriting recognition, bio-metric identification, document classification, and much more.*

General Approach of Classification

- **Training set** consists of records with **known class labels**
- Training set is used to **build a classification model**
- The classification model is applied to the **test set** that consists of records with **unknown labels**

Evaluation of classification models

Counts of test records that are **correctly** (or incorrectly) **predicted** by the classification model

- **Confusion matrix**

		Predicted Class	
		Class = 1	Class = 0
Actual Class	Class = 1	f_{11}	f_{10}
	Class = 0	f_{01}	f_{00}

$$\text{Accuracy} = \frac{\#\text{ correct predictions}}{\text{total }\#\text{ of predictions}} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

$$\text{Error rate} = \frac{\#\text{ wrong predictions}}{\text{total }\#\text{ of predictions}} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

Types of Classification

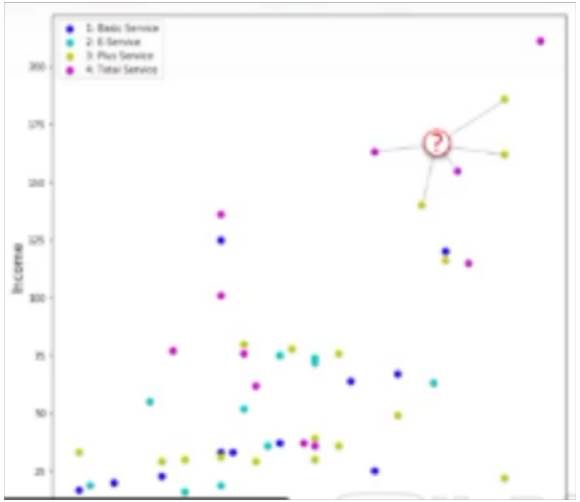
- **Decision Trees**
  - ID3, C4.5, C5.0, CHAID
- **Naïve Bayes**
- **Linear Discriminant Analysis**
- **K-Nearest Neighbor**
- **Logistic Regression**
- **Neural Networks**
- **Support Vector Machines (SVM)**

K-Nearest Neighbor Algorithm

In k-nearest neighbors, data points that are near each other are said to be “**neighbors**". K-nearest neighbors is based on this paradigm:

“Similar cases with the same class labels are near each other”

Thus, the distance between two cases is a measure of their **dissimilarity**.



K-Nearest Neighbors Algorithm

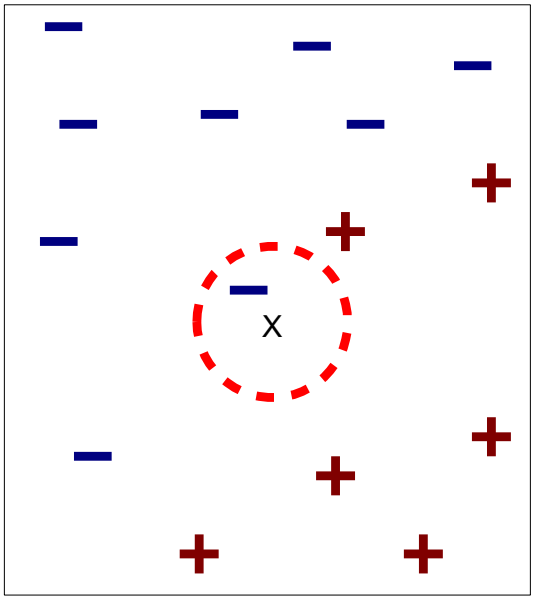
1. Pick a value for k
2. Given a data record **x** find its **k** closest points
   - Closeness: Euclidean, Hamming, Jaccard distance...
3. Predict the response of the unknown data point using the most popular response value from the K nearest neighbors.
   - Majority vote, weight the vote according to distance, probabilistic voting

There are two parts in the algorithm that might be confusing:

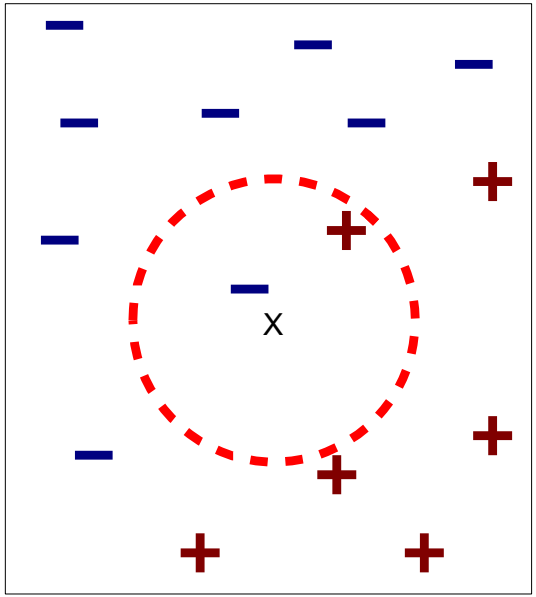
1. How to select the correct K value
2. How to compute the similarity between cases.

A low value of K causes a highly complex model as well, which might result in over-fitting of the model. It means the prediction process is not generalized enough to be used for out-of-sample cases.

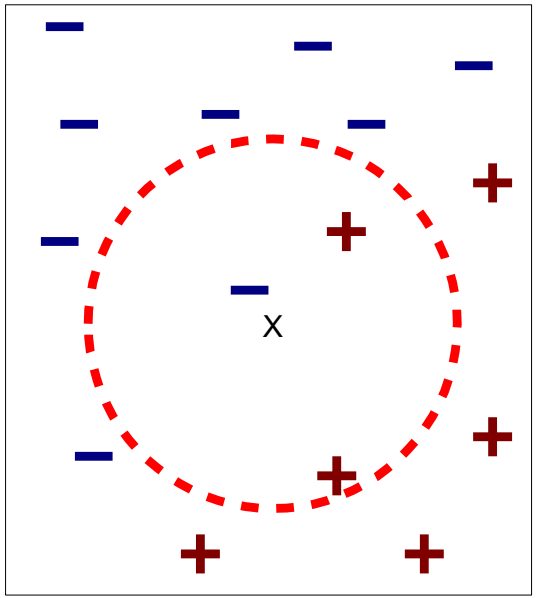
K-Nearest Neighbors Algorithm



(a) 1-nearest neighbor



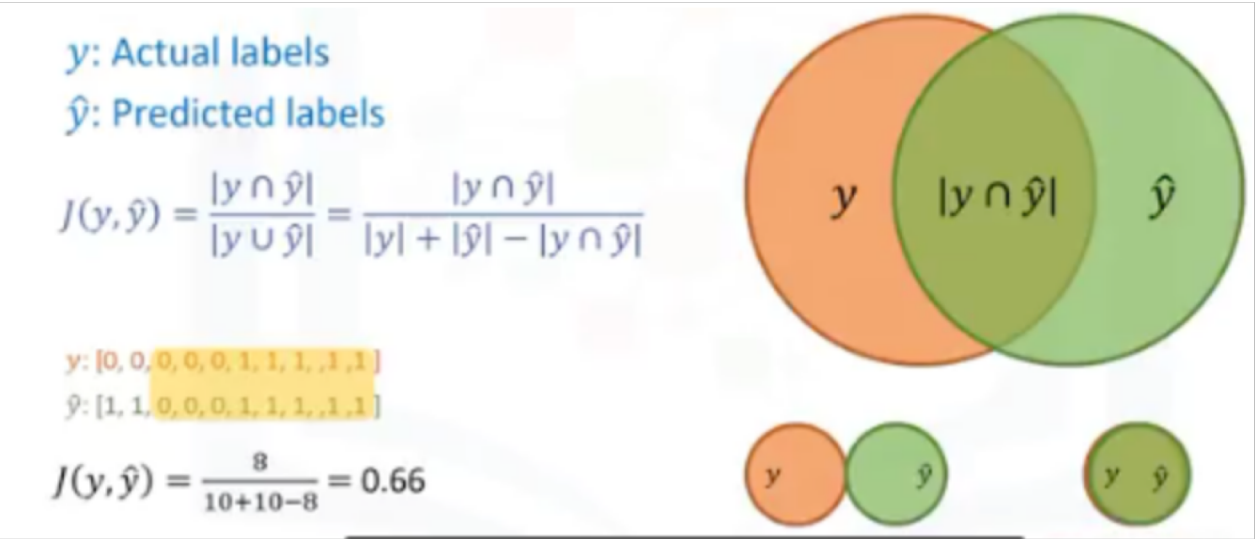
(b) 2-nearest neighbor



(c) 3-nearest neighbor

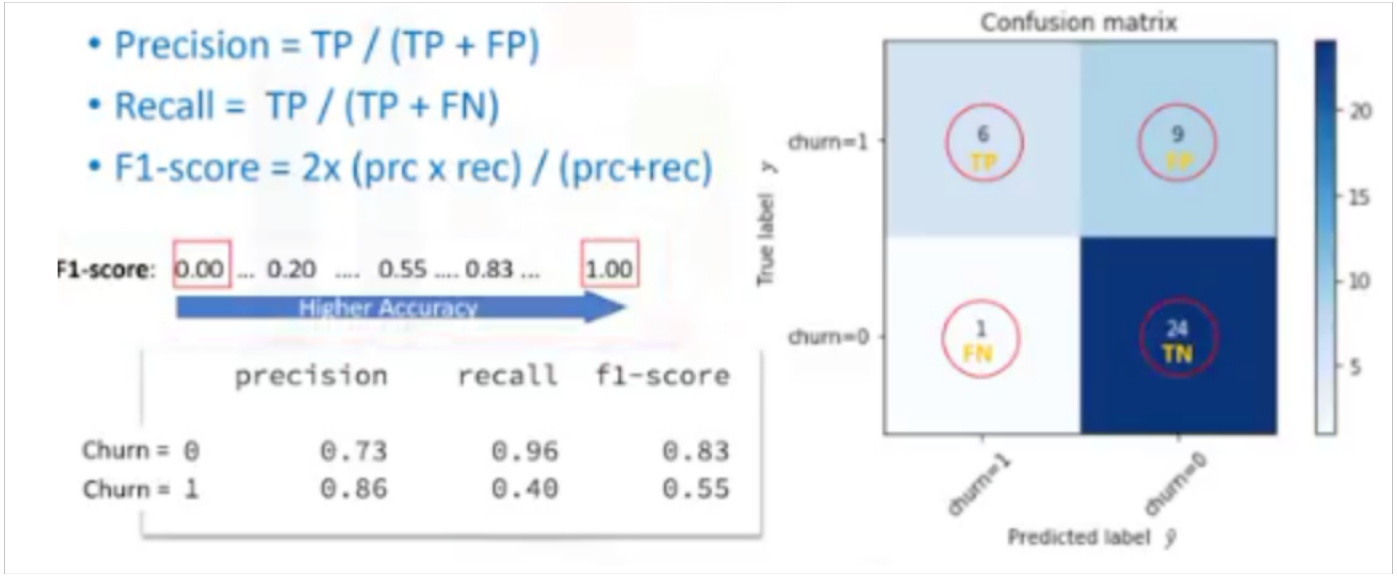
Evaluation Metrics in Classification

- **The Jaccard Index**
  - **Y** represents the true labels of churn dataset
  - $\hat{\mathbf{Y}}$ represents predicted values by the classifier



Evaluation Metrics in Classification

- **F1-Score**
  - Looking at the **Confusion Matrix**



Decision Trees

- A flow-chart-like tree structure
- Internal node denotes a test on an attribute
- Each Branch represents an outcome of the test
- Leaf nodes represent class labels or class distribution

Decision tree generation consists of two phases

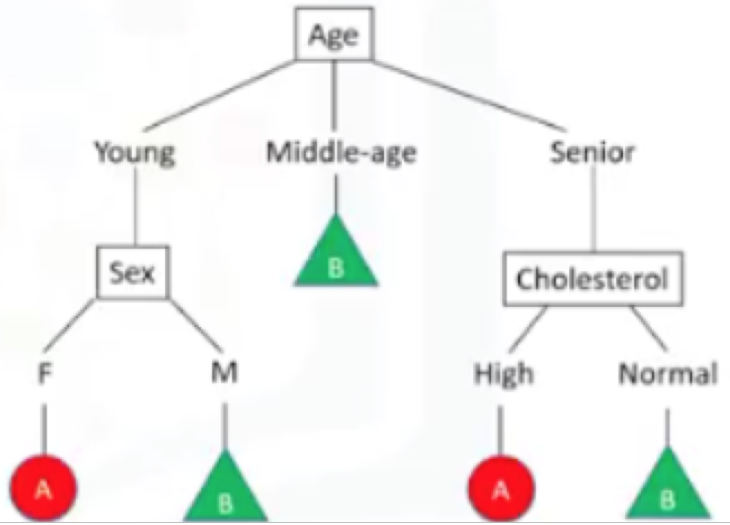
- **Tree construction**
  - At start, all the training examples are at the root
  - Partition examples recursively based on selected attributes
- **Tree pruning**
  - Identify and remove branches that reflect noise or outliers

Use of decision tree: Classifying an unknown sample

- Test the attribute values of the sample against the decision tree

Decision Tree Algorithm

1. Choose an attribute from your dataset.
2. Calculate the significance of attribute in splitting of data.
3. Split data based on the value of the best attribute.
4. Go to step 1.



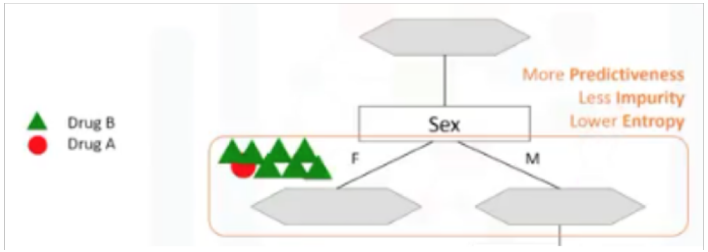
Decision trees are built using **recursive partitioning** to classify the data
The algorithm chooses the **most predictive feature** to split the data.

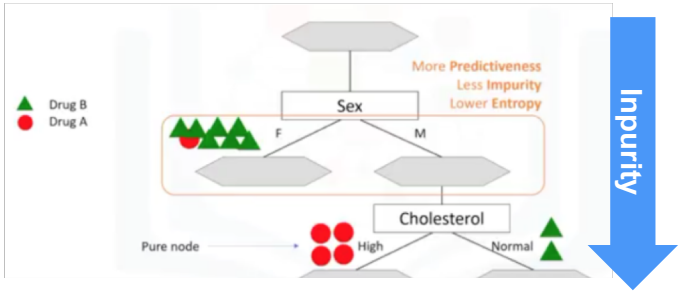
How predictiveness is done?

What is important in making a decision tree, **is to determine "which attribute is the best, or more predictive, to split data based on the feature."**

"**predictiveness**" is based on decrease in "impurity" of nodes.

- We're looking for the best feature to decrease the "impurity" of patients in the leaves, after splitting them up based on that feature.
- A node in the tree is considered "pure" if, in 100% of the cases, the nodes fall into a specific category of the target field.





What is Entropy? What is Information gain?

- **Entropy** is the amount of information disorder, or the amount of **randomness** in the data.
  - The entropy in the node depends on how much random data is in that node and is calculated for each node
  - In decision trees, we're looking for trees that have the smallest entropy in their nodes.
  - The entropy is used to calculate the homogeneity of the samples in that node.

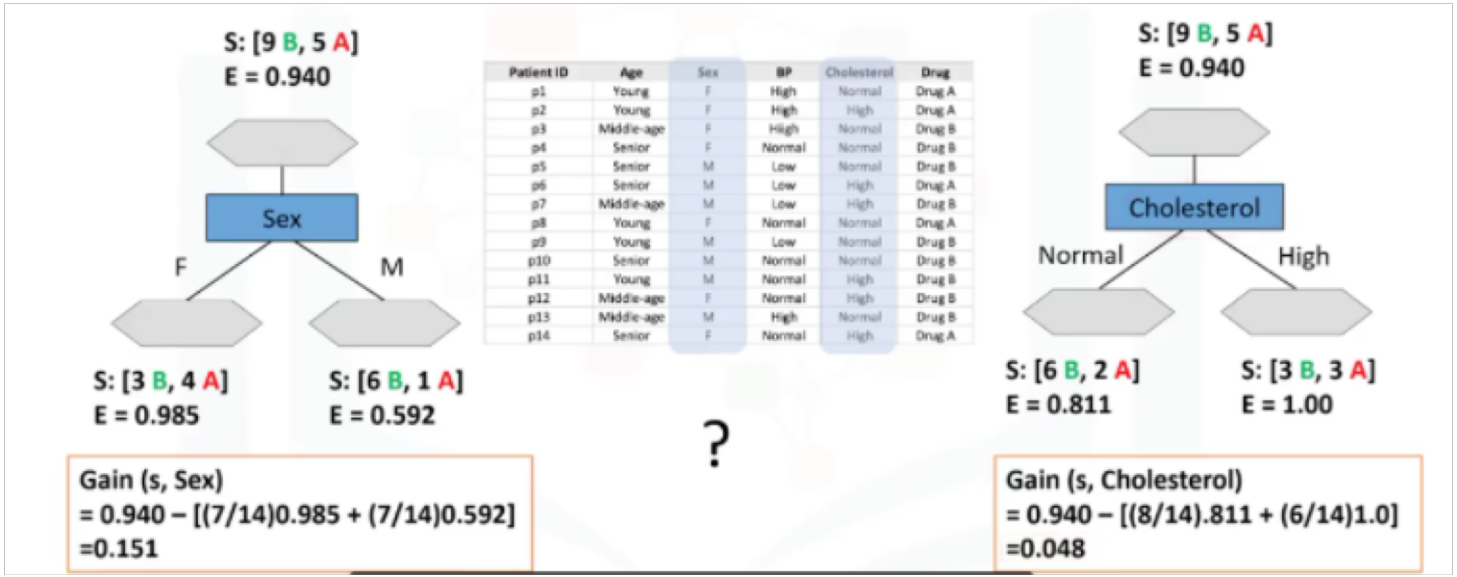
$$\text{Entropy} = - p(A)\log(p(A)) - p(B)\log(p(B))$$

- **Information gain** is the information that can increase the level of certainty after splitting.

Information Gain = (Entropy before split) – (weighted entropy after split)

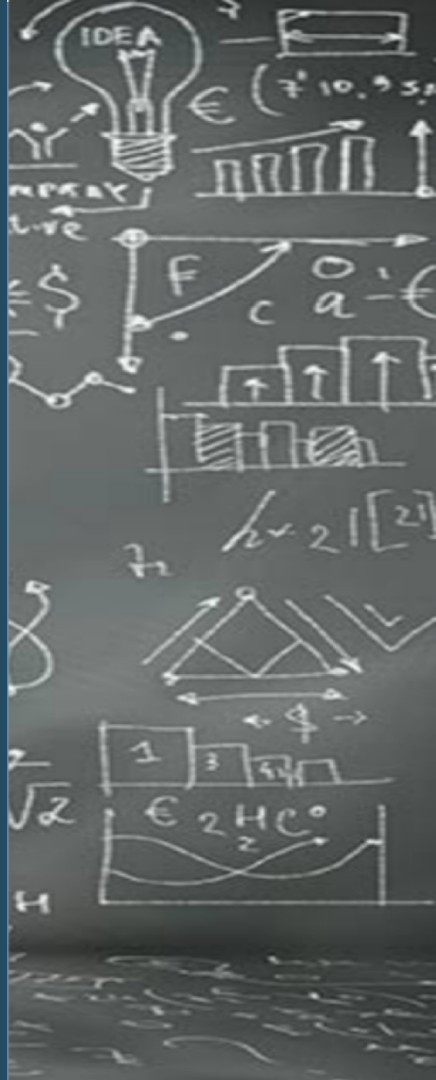
Finally

- **So constructing** a decision tree is all about finding attributes that return the highest information gain



Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	High	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A

Thank You



Useful Links & Resources

External

Getting Started:
Service Homepage
Feature Requests / Suggestions

Case Studies:
OmniEarth
Aerialtronics
BlueChasm
iTrend

Tutorials & Best Practices:
Training models with Watson Studio
Getting started with Watson + Core ML
Stacking Multiple Custom Models
Create a Calorie Counting App
Watson Visual Recognition & Twilio
Best Practices for Custom Models

Code Patterns:
Classify vehicle damage
Analyze industrial equipment for defects
Create an Android calorie-counter app

External continued

Books:
Redbook: Building Cognitive Application using IBM Watson Services vol3 – Watson Visual Recognition

Blogs:
IBM Watson on Medium

Internal

Slack Channel: #ibmvisual-recognition
Service Roadmap
IBMer key limit increase request form
ZACS portal
Digital Sales Play
Content Request & Feedback Form