Machine Learning

AN INTRODUCTION TO NEURAL NETWORKS

CASCON 2017

Speakers

Di Xiao
Di.Xiao@ibm.com

Mike Andrea
andrea@ca.ibm.com

Serjik G. Dikaleh
serjikgd@ca.ibm.com

Dharmesh Mistry
dmistry@ca.ibm.com

Chris Felix
chrisfel@ca.ibm.com

Workshop Outline

Part 1

- Speaker Introductions (5 min)
- Machine Learning Overview Lecture (5 min)
- Hands-on Lab and exploration (40 min)
- IBM Data Science Experience Introduction (10 min)

Part2

- Neural Networks Introduction (10 min)
- Hands-on Lab and exploration (50)

Machine Learning Intro

Basic Premise

We define the ***target*** of the exercise, which could be classifying some input, identifying some hidden patterns, or maybe a prediction

This in turn clarifies what ***data representation*** (choices of features and values) we need and the ***training data*** required

This also helps determine the ***learner*** to be applied based on the target

Neural Networks as the learner, in theory, can be proven to *approximate* any function to any degree.

Approaches to Machine Learning

Supervised Learning

- ***Classify*** data based on learning from labeled data
- Can be limited (e.g. yes/no or happy/sad/angry) or a continuous range (***regression***, like age)

Unsupervised Learning

- No training with labeled data, the algorithm decides the outcome, for example using clustering
- Data subsets should have high similarity in some classes, and low similarity among other subsets
- For example K-means is a clustering algorithm

Reinforcement Learning

- Works based on a feedback loop, where the results are measured by some score or advantage
- Common application in games, maximize the reward like getting higher score or losing less pieces

Data & Overfitting

Data is organized by ***features***

- Attributes that impact the decisions and final goal of the algorithm (collect, ingest, cleanse, slice)

Training Data

- Labeled data to train the model

Test Data

- Test the model while training

Validation Data – Hold-out set

- Validate the accuracy of the model

To avoid ***Overfitting***, use tighter loops of train & test and then validate at the end

Algorithms based on implementation

Regression Algorithm

- Linear Regression

Classical Classifiers

- K Nearest Neighbor
- Decision trees
- Naïve Bayes
- Support Vector Machine

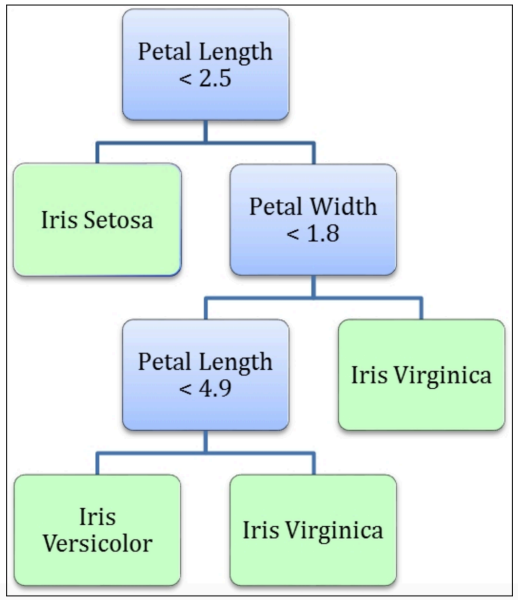
Clustering

- K-means

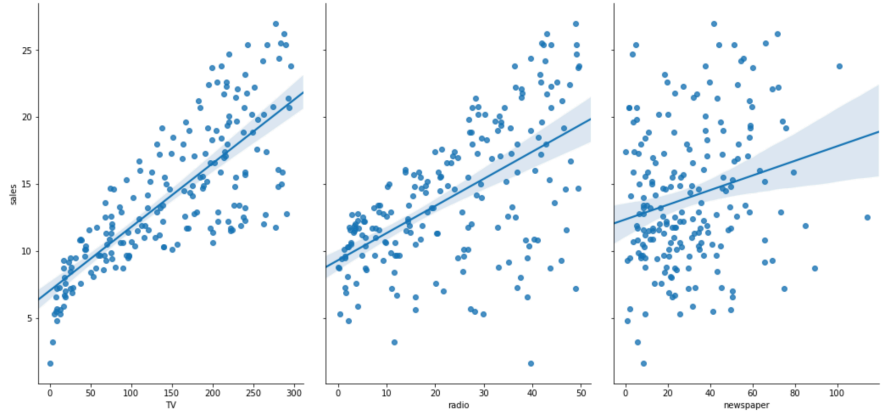
Reinforcement learning

- Cross-entropy method

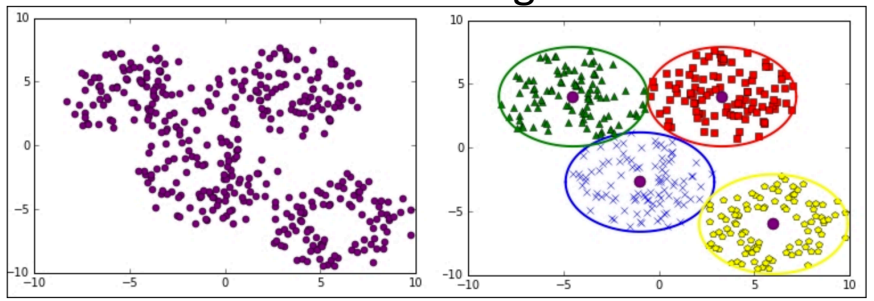
Decision Tree



Linear Regression



Clustering



DSX Overview

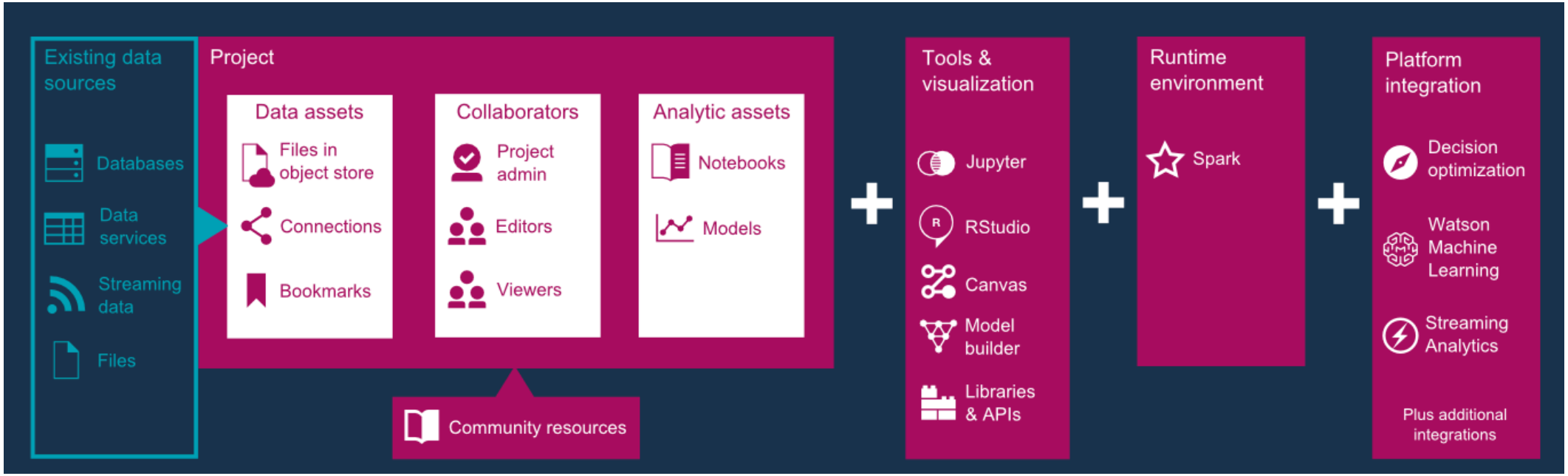
What is DSX?

IBM Data Science Experience (DSX) allows data scientists to analyze data using RStudio, Jupyter, and Python in a configured, collaborative environment that includes IBM value-adds, such as managed Spark. Data Science Experience offers all this functionality, including:

- a single workspace for your tools
- a searchable, growing community
- shareable, collaborative projects
- supporting content, where you need it

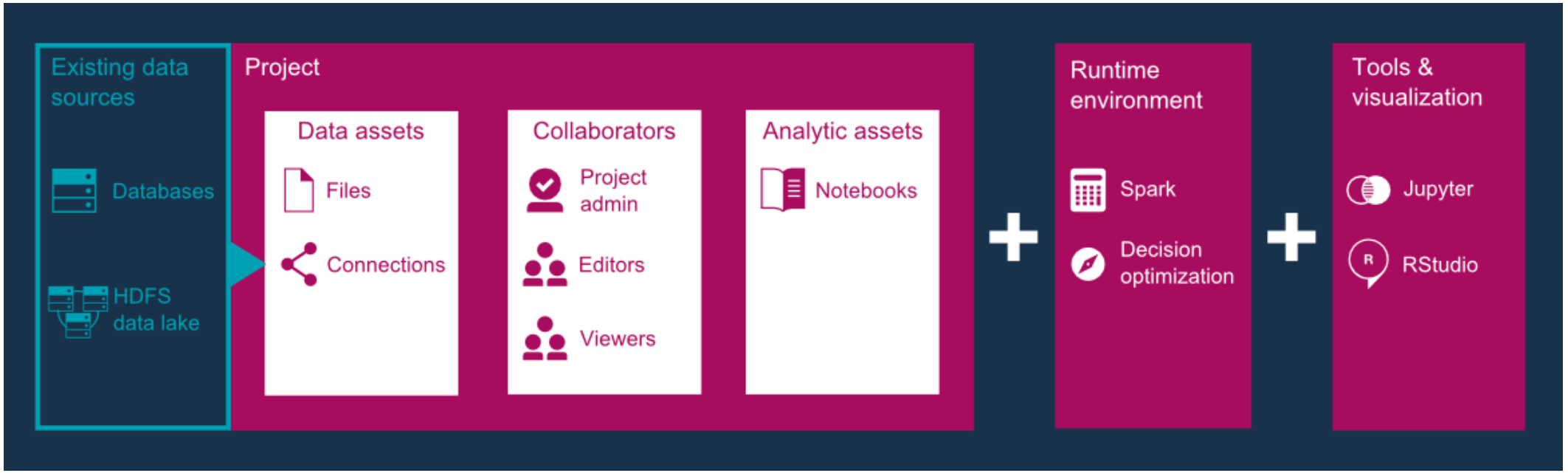
IBM Data Science Experience

DSX Cloud



Source: https://dataplatform.ibm.com/docs/content/getting- started/architecture.html?context=analytics&linkInPage=true

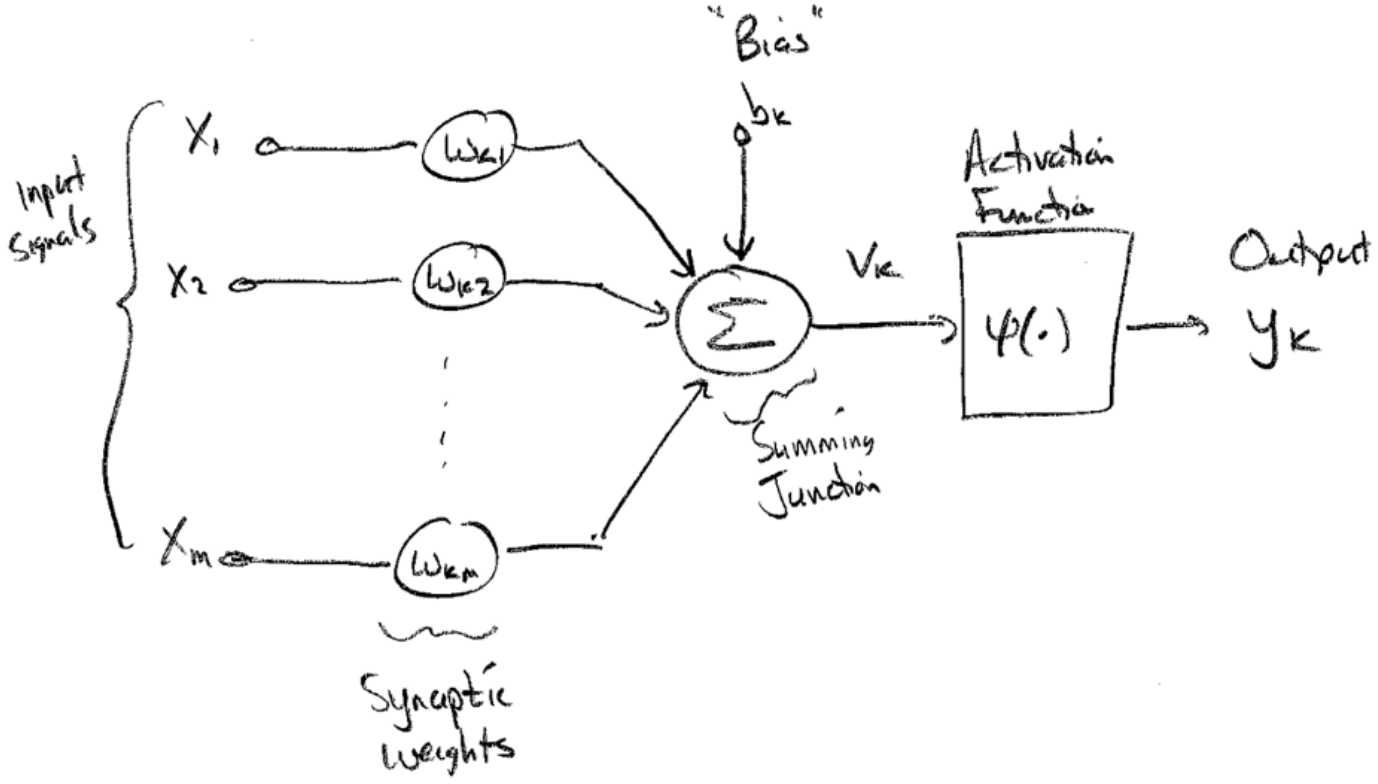
DSX Local



Source: https://dataplatform.ibm.com/docs/content/local/overview.html?context=analytics&linkInPage=true

Neural Networks Intro

Neural Networks

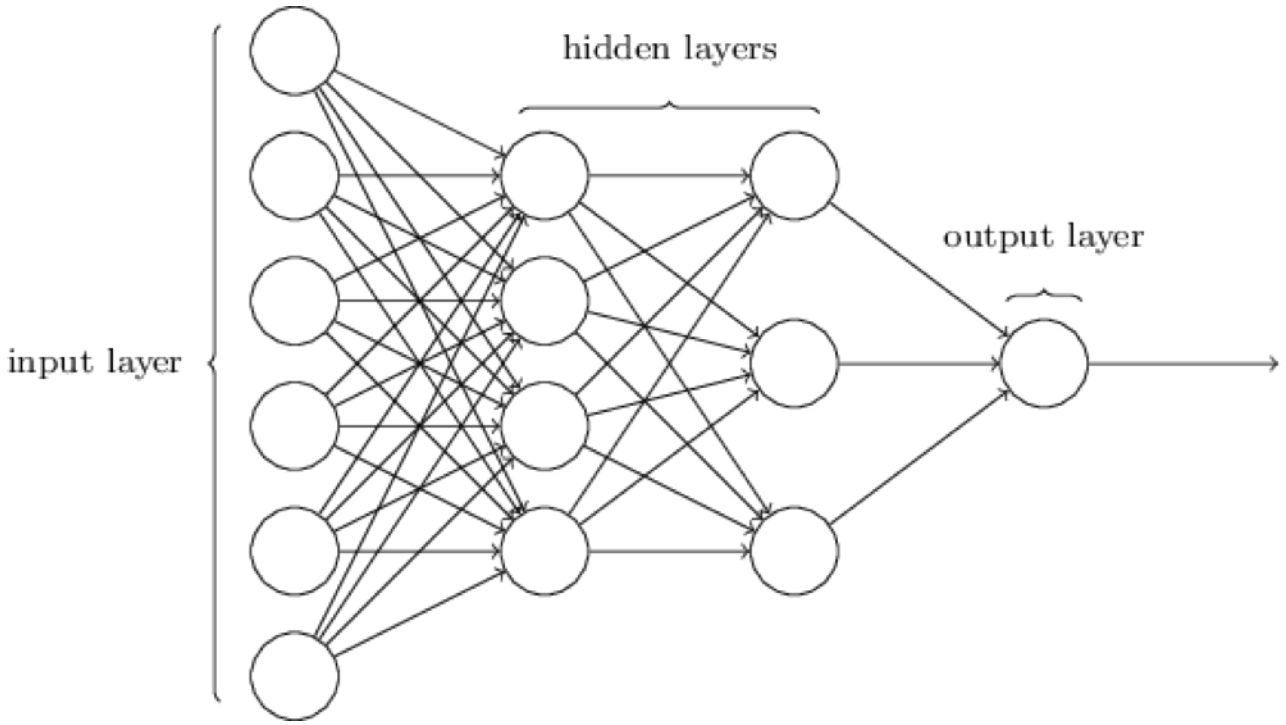


Frank Rosenblatt invented the first neural network called ***perceptrons***

A simple representation of a neuron with its input (a feed-forward one-layer network)

It can only learn ***linearly separable*** patterns

Deep Learning



More layers are added between the input and output layers, called ***hidden layers***

The ***Universal Approximation Theorem*** tells us that any function can be approximated to an arbitrarily close degree by a neural network with enough neurons and hidden layers

Training the neural network

Stochastic Gradient Descent

- An common optimization algorithm to solve minimization problems
- In the neural network context, this function to be minimized is depends entirely on the data that we train on (how different our solution is from the labeled answer)

Backpropagation

- Trickles back “error corrections” from the gradient calculations,
- The predominant algorithm for training neural networks
- Because we train step by step, we run through the data set multiple times (epochs) to reduce the cost function

Open Source Packages

Open source packages that implement most of the popular and classic ML algorithms

Base package examples:

- NumPy, SciPy, Matplotlib

Scikit-learn (http://scikit-learn.org)

- Classifiers, regressors and support vector machine (SVM) algorithms, nearest neighbors, random forest, linear regression, k-means, decision trees and neural networks

TensorFlow (https://www.tensorflow.org/)

Theano (https://github.com/Theano/Theano)

Keras (https://keras.io/)

References

Python Deep Learning

- by Peter Roelants; Valentino Zocca; Gianmario Spacagna; Daniel Slater (Published by Packt Publishing, 2017)

Fundamentals of Deep Learning

- by Nikhil Buduma

Machine Learning Tutorial

- https://www.youtube.com/watch?v=elojMnjn4kk&list=PL5-da3qGB5ICeMbQuqbbCOQWcS6OYBr5A

Data Science

- https://datascience.ibm.com
- Data Science Experience Documentation
- Data Science Courses on Cognitive Class
- Chronic Kidney Disease Data Set