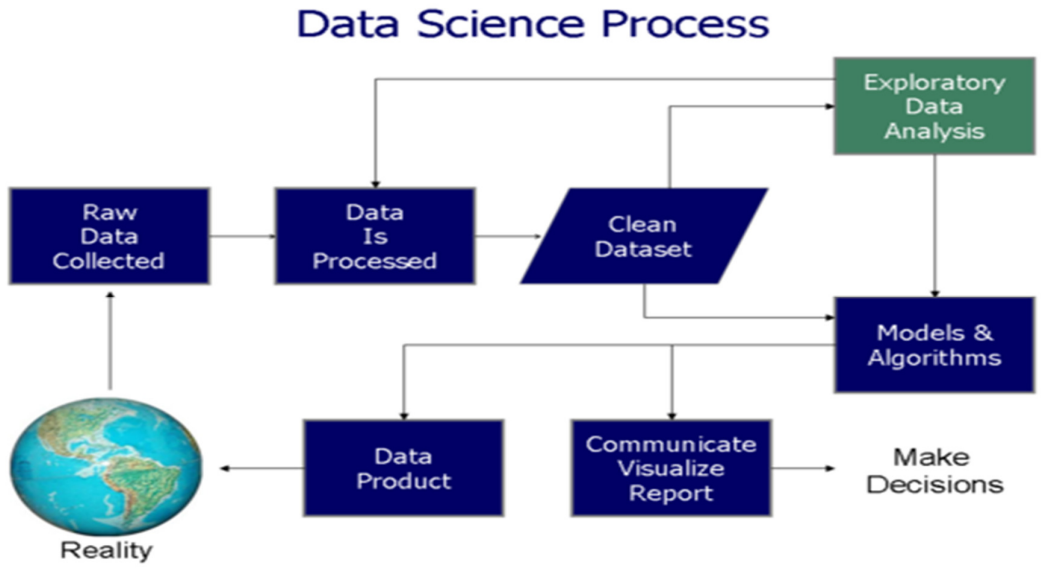
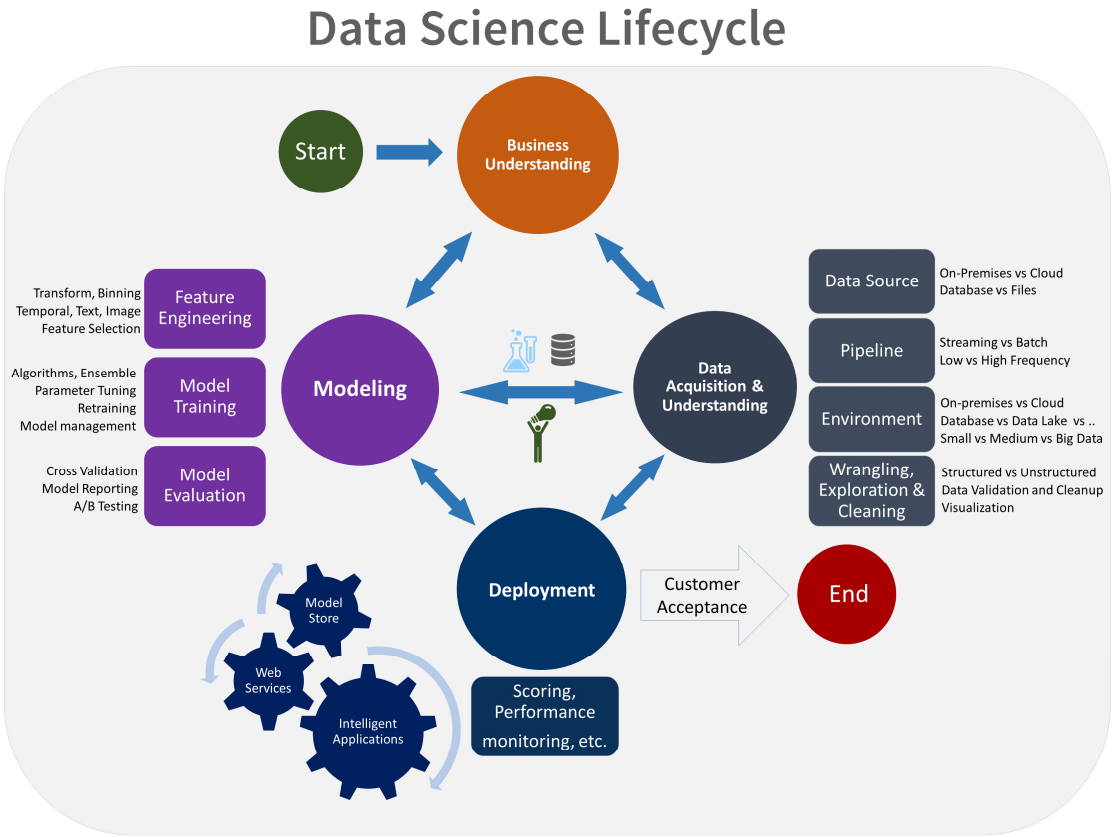
IBM Coursera Advanced Data Science Capstone-Prediction of Defaulting on Credit Card Payment-Madhan Bose

Architectural Decisions Document





1 Architectural Components Overview

1.1 Data Source

1.1.1 Definition

Understanding data is one of the most important part when designing any machine learning algorithm. In the notebook, I will use a data set which is available at the Center for Machine Learning and Intelligent Systems, Bren School of Information and Computer Science, University of California, Irvine:
https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients
and which is available in the URL:
https://www.kaggle.com/lucabasa/credit-card-default-a-very-pedagogical-notebook/data

1.1.2 Technology choice

- IBM Watson Studio Jupyter Notebooks, scikit-learn, pandas, matplotlib
- IBM Watson Studio Jupyter Notebooks, Tensor Flow with Keras

1.2 Data Integration

1.2.1 Technology Choice

- IBM Watson Studio Jupyter Notebooks, scikit-learn, pandas, matplotlib
- IBM Watson Studio Jupyter Notebooks, Tensor Flow with Keras

1.2.1.1.1 Tensor Flow and Keras

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.[5] It is used for both research and production at Google.

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet also is the author of the XCeption deep neural network model.

1.2.1.2 Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

1.2.1.3 IBM Watson Studio

IBM Watson Studio provides tools for data scientists, application developers and subject matter experts to collaboratively and easily work with data to build and train models at scale. It gives you the flexibility to build models where your data resides and deploy anywhere in a hybrid environment so you can operationalize data science faster.

1.3 Data Repository

1.3.1 Architectural Decision Guidelines

There exists an extremely huge set of technologies for persisting data. IBM Cloud Object Store is used for the data storage in the project. The most important questions to be asked are:

- How does is the impact of storage cost?
- Which data types must be supported?
- How good must point queries (on fixed or dynamic dimensions) be supported?
- How good must range queries (on fixed or dynamic dimensions) be supported?
- How good must full table scans be supported?
- What skills are required?
- What's the requirement for fault tolerance and backup?
- What are the constant and peak ingestion rates?
- What's the amount of storage needed?
- How does the growth pattern look like?
- What are the retention policies?

1.3.2 Technology choice

- IBM Watson Studio jupyter notebooks, scikit-learn, pandas, matplotlib
- IBM Watson Studio jupyter notebooks, Tensor Flow and Keras
- IBM Cloud object storage

1.4 Discovery and Exploration

1.4.1 Definition

This component allows for visualization and creation of metrics of data.
In various process models, data visualization and exploration are one of the first steps. Similar tasks are also applied in traditional data warehousing and business intelligence. So, for choosing a technology, the following questions should be kept in mind:

- What type of visualizations are needed?
- Are interactive visualizations needed?
- Are coding skills available / required?
- What metrics can be calculated on the data?

1.4.2 Technology Choice

- IBM Watson Studio Jupyter Notebooks, scikit-learn, pandas, matplotlib
- IBM Watson Studio Jupyter Notebooks, Tensor Flow with Keras

1.4.3 Justification

1.4.3.1 Jupyter, python, scikit-learn, pandas, matplotlib

The components mentioned above are all open source and supported in the IBM Cloud. Some of them have overlapping features, some of them have complementary features. This will be made clear by answering the architectural questions

- What type of visualizations are needed? Matplotlib supports the widest range of possible visualizations including bar charts, run charts, histograms, box-plots and scatter plots.
- Are interactive visualizations needed? Whereas matplotlib creates static plots, pixiedust supports interactive ones
- Are coding skills available / required? Whereas matplotlib needs coding skills,. For computing metrics, some code is necessary in Python
- What metrics can be calculated on the data? Using scikit-learn and pandas, all state-of-the-art metrics are supported
- Do metrics and visualization need to be shared with business stakeholders? Watson Studio supports sharing of jupyter notebooks, also using a fine-grained user and access management system

1.5 Actionable Insights

1.5.1 Technology Choice

There exists an abundance of open and closed source technologies (IBM Watson Studio, Jupyter, Tensor Flow and Keras…). Here, the most relevant are introduced. Although it holds for other sections as well, decisions made in this section are very prone to change due to the iterative nature of this process model. Therefore, changing or combining multiple technologies is no problem, although decisions let to those changes should be explained and documented.

1.5.2 Justification

1.5.2.1 Python, pandas and scikit-learn

Python is a much cleaner programming language than R and easier to learn therefore. Pandas is the python equivalent to R dataframes supporting relational access to data. Finally, scikit-learn nicely groups all necessary machine learning algorithms together. It's supported in the IBM Cloud via IBM Watson Studio as well.

- What are the available skills regarding programming languages? Python skills are very widely available since python is a clean and easy to learn programming language.
- What is the cost of skills regarding programming languages? Because of python's properties mentioned above, cost of python programming skills is very low
- What are the available skills regarding frameworks? Pandas and scikit-learn are very clean and easy to learn frameworks, therefore skills are widely available.
- What is the cost of skills regarding frameworks? Because of the properties mentioned above, cost of skills is very low.
- Is model interchange required? All scikit-learn models can be (de)serialized. PMML is supported via 3rdparty libraries.
- Is parallel or GPU based training or scoring required? Neither GPU nor scale-out is supported, although scale-up capabilities can be added individually to make use of multiple cores.
- Do algorithms need to be tweaked or new algorithms to be developed? Scikit-learn algorithms are very cleanly implemented. They all stick to the pipelines API making reuse and interchange easy. Linear algebra is handled throughout with the NumPy library. So, tweaking and adding algorithms is straightforward