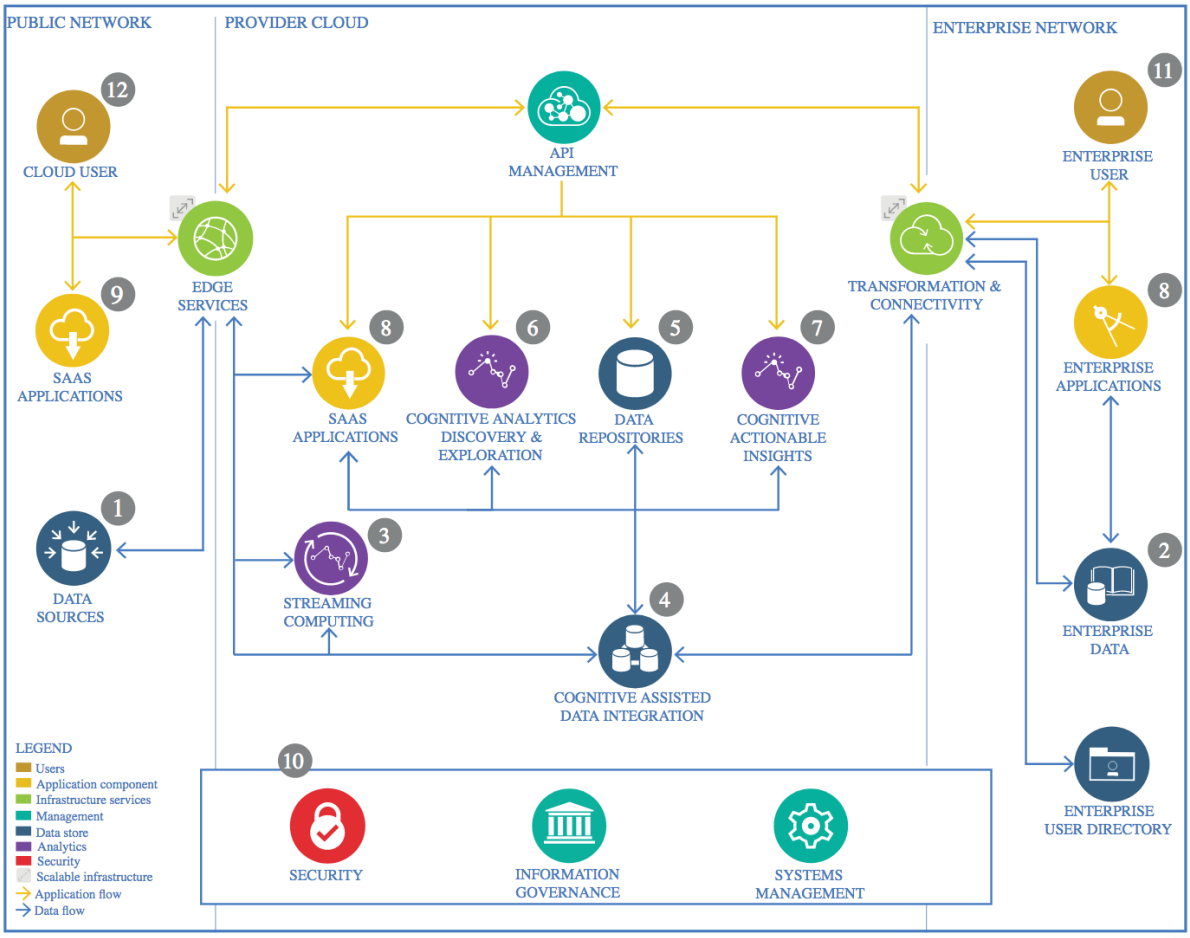
The Lightweight IBM Cloud Garage Method for Data Science

Architectural Decisions Document

1 Architectural Components Overview



IBM Data and Analytics Reference Architecture. Source: IBM Corporation

1.1 Data Source

1.1.1 Technology Choice

The data source is kaggle where the kaggle API is used to fetch the data. The original file is in csv format. Jupyter notebook is the preferred technology for development. Python pandas and numpy is used to load the data as dataframe and perform data quality checks. Python matplotlib and seaborn are used for visualizations.

1.1.2 Justification

Jupyter notebook helps the dynamic development and debugging. Given the original file csv format and the size of the file, pandas is the preferred technology. Pandas dataframe is a structured way to load and analyze the data. Numpy is used also for fast array calculations. Matplotlib and seaborn are great tools for pandas dataframes visualizations.

1.2 Enterprise Data

1.2.1 Technology Choice

Not applicable

1.2.2 Justification

Enterprise data is not applicable. The data source is kaggle and the dataset is continuously updated. It would make sense to use object storage on the IBM cloud but for the project needs that would be an overhead.

1.3 Streaming analytics

1.3.1 Technology Choice

The file format is csv with a size of approximately 2MB and therefore for development batch processing is the preferred technology. For production though, real time analytics capabilities would increase the value of the data product, where tweets are classified in real time.

1.3.2 Justification

Batch processing is the preferred technology because of the file format (CSV) and size (2MB). Online processing and partitioning would be better for larger datasets and other formats.

1.4 Data Integration

1.4.1 Technology Choice

There are many options to consider when we process the tokens from a corpus of text. These are some of the questions you might want to ask: Which stop words do I include? Which stemmer/lemmatizer is best? Which n-grams do I include? Do I filter based on frequency min and max? There are many ways to process tokens (words, dates, emojis etc). I used NLTK python package where tokens are modified via stemming or lemmatization.

1.4.2 Justification

NLTK python package is often used to pre-process text data before the tokens are vectorized.

1.5 Data Repository

1.5.1 Technology Choice

ETL stores the data in npz format after NLTK processing in the local directory.

1.5.2 Justification

Having the dataset in a clean format speeds up the process. The dataset in this format could be directly used for feature engineering and modeling. Once the original data source is updated a new version of the file is needed.

1.6 Discovery and Exploration

1.6.1 Technology Choice

For the data transformation, scikit-learn pipeline is used where the first step is CountVectorizer followed by TfidfTransformer. For visualization, TSNE model from scikit-learn was used.

1.6.2 Justification

A bag-of-words model is a representation of text. A document or sentence is represented as numeric counts of the individual words, without considering grammar and punctuation. Even the word order is ignored unless you expand your feature matrix with n-grams. We can calculate various measures to characterize the text. The most common type matrix derived from the bag-of-words is representation term frequency (TF), which is the number of times a token appears in the text. Another useful matrix is the term frequency-inverse document frequency (tf-idf) matrix. Scikit-learn python package provides models to implement the text processing as mentioned above. To deal with high dimensional data, dimensionality reduction models from scikit-learn are provided.

1.7 Actionable Insights

1.7.1 Technology Choice

For the modelling, different algorithms are explored. Machine learning and statistical models from package scikit-learn in python are used. For deep learning modelling keras package and tensorflow are used.

1.7.2 Justification

Scikit-learn provides with classes in python to model machine learning applications. In combination with the scikit-learn pipeline class, they allow fast implementations, model comparison and optimal tuning of the models. Keras is the state-of-the-art package for deep learning development. The keras API allows fast development of deep learning applications. For further tuning and customization tensorflow package in python is the best solution.

1.8 Applications / Data Products

1.8.1 Technology Choice

For deployment the python flask package is used. The final data product is a Flask APP which runs on a localhost server.

1.8.2 Justification

The Flask APP creates an API to the model. The model on the localhost server can make predictions given new data in the right format. This technology allows to several users to run the model given that they have access to the localhost server.

1.9 Security, Information Governance and Systems Management

1.9.1 Technology Choice

Not applicable

1.9.2 Justification

All technologies and data used for the development of the model are open source and maintained by the data science community.