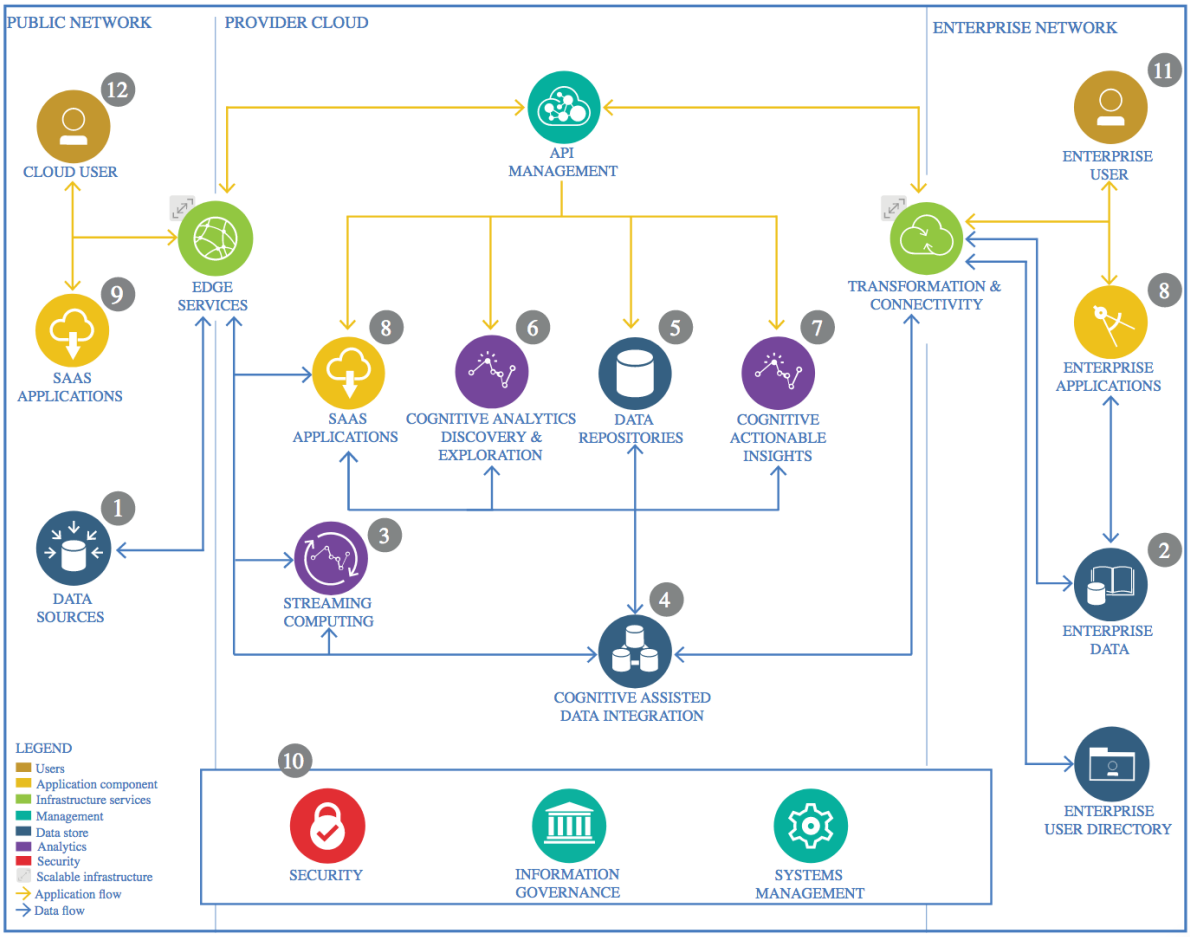
The Lightweight IBM Cloud Garage Method for Data Science

Architectural Decisions Document Template

1 Architectural Components Overview



IBM Data and Analytics Reference Architecture. Source: IBM Corporation

1.1 Data Source

1.1.1 Technology Choice

Please describe what technology you have defined here. Please justify below, why. In case this component is not needed justify below.

Data source used for the project is called Women's E-Commerce Clothing Reviews. It is an external data and downloadable here from Kaggle as a CSV file.

1.1.2 Justification

Please justify your technology choices here.

Publicly accessible and available.
Data contains both qualitative and quantitative which I was interested to make use of by combining as one single textual input feature after having read this post.

1.2 Enterprise Data

Not applicable because data source is public and downloadable from Kaggle.

1.2.1 Technology Choice

Please describe what technology you have defined here. Please justify below, why. In case this component is not needed justify below.

Not applicable.

1.2.2 Justification

Please justify your technology choices here.

Not applicable.

1.3 Streaming analytics

1.3.1 Technology Choice

Please describe what technology you have defined here. Please justify below, why. In case this component is not needed justify below.

Not applicable because no real-time analytics is needed for the use case.

1.3.2 Justification

Please justify your technology choices here.

Not applicable.

1.4 Data Integration

1.4.1 Technology Choice

Please describe what technology you have defined here. Please justify below, why. In case this component is not needed justify below.

- EDA by Python using Google Colaboratory (Plan: Free of charge) with RAM: 12.68GB and Disk: 107.72GB, Python 3 Google Compute Engine Backend.
- Model training by Python in Google Cloud Platform Workbench Vertex AI JuptyerLab 3 Notebook on machine type 8 vCPUs, 30GB RAM

1.4.2 Justification

Please justify your technology choices here.

Runtime crashes during model training in Google Colab due to resource limitation hence Google Cloud Platform Workbench Vertex AI was explored to continue.

Both are Google products and therefore less likely to be incompatible after some setups to the environment.

1.5 Data Repository

Source data download and stored at GitHub repository here as well as the training and test sets generated during EDA.

1.5.1 Technology Choice

Please describe what technology you have defined here. Please justify below, why. In case this component is not needed justify below.

Post-EDA, Spark dataframe of training and test sets were saved as parquet files in Google Colaboratory (Plan: Free of charge) with RAM: 12.68GB and Disk: 107.72GB, Python 3 Google Compute Engine Backend in order to load back in Google Cloud Platform Workbench Vertex AI JuptyerLab 3 Notebook as a Spark dataframe for model training on machine type 8 vCPUs, 30GB RAM.

1.5.2 Justification

Please justify your technology choices here.

Resource limitation of Google Colab free plan.

1.6 Discovery and Exploration

The following findings can be obtained from EDA.iynb located in GitHub repository here.

Due to resource limitation, a random sample with replacement (n=6000) of the original data(n=23486) was taken with similar target proportions ensured (≈82% vs ≈18%). This proportions indicate a very high class imbalance where there is about five times recommended clothing reviews more than not recommended reviews.

Two duplicates were identified and thus removed. Missing string data was imputed with "unknown" to be able to combine with the rest of the numerical features as one single textual feature as input to classification algorithms which I was interested to implement after having read this post.

According to the boxplots and summary statistics table, numerical features i.e. Age, Rating and Feedback Positive Count show similar distribution for both recommended and non-recommended groups. Whereas, it is extremely clear that the "Rating" is very much better for recommended group than non-recommended group. There does not seem to be any correlation among this numerical features with the highest absolute correlation being only 0.0669 (i.e. age and positive feedback count).

Most reviews come from the department "Top" (n = 2698), division "General" (n = 3564) and class "Dresses" (n = 1631).

A single textual feature was created as follows after having read this as an input for natural language processing to classification algorithms training.

> This item comes from{"Department Name} department and division is {"Division Name"}, and is classified under {"Class Name"}. There are {"Positive Feedback Count"} customers who found this review positive. I am {"Age"} years old. I rate this item {"Rating"} out of 5 stars.

After text preprocessing i.e. tokenization, normalization, stop word removal and stemming to calculate the term frequency-inverse document frequency, two classical classification algorithms of logistic regression and random forest were trained.

The word clouds of before and after text preprocessing show that there are both the positive and negative terms in both the groups making both groups not as easy to differentiate.

Class weights were estimated to handle the class imbalance in training set as follows.

$weight_i$ = (Total number of entries in training set)/[(number of distinct target group)*(Total number of entries in ith target group in training set)], where i = ith target group

```
# https://scikit-learn.org/stable/modules/generated/sklearn.utils.class_weight.compute_class_weight.html
# Estimate class weights for unbalanced datasets
weight0=train.count()/((train.select('Recommended IND').distinct().count())*(train.filter(train["Recommended IND"] == 0).count()))
weight1=train.count()/((train.select('Recommended IND').distinct().count())*(train.filter(train["Recommended IND"] == 1).count()))
```

1.6.1 Technology Choice

Please describe what technology you have defined here. Please justify below, why. In case this component is not needed justify below.

- Google Colab Jupyter Notebook for EDA.

- Google Cloud Platform Workbench Vertex AI JupyterLab 3 Notebook for coding machine learning training.
- gdrivefs 0.14.13: Mount to Google Drive
- spark-nlp 4.0.1: Spark session, text pre-processing and deep learning algorithm
- pyspark 3.3.0: Spark dataframe and pandas-on-spark dataframe manipulation, grid search, cross validation, classical machine learning algorithms e.g. logistic regression and AUC
- sklearn: Confusion matrix, classification report
- pandas 1.3.5, Numpy: Other non-Spark data manipulation e.g. dataframe, list, array, dictionary, tuple
- matplotlib: Pie chart, bar charts, boxplot for distribution, pattern and outlier
- seaborn: Heatmap
- word Cloud

1.6.2 Justification

Please justify your technology choices here.

Spark NLP is specialized in distributed processing in NLP with many pre-trained models available for use case in in NLP.
Pyspark allows distributed processing in dataframe manipulation and classical machine learning algorithms.

1.7 Actionable Insights

The data is heavily imbalanced. Hence, accuracy and area under the receiver operating curve were produced but not used for model evaluation as well as the training time taken.

After hyperparameter tuning by grid search and 3-fold cross validation (3 due to resource limitation), the simple logistic regression with numerical features is the best classifier model among the other three. Rating seems to be the best predictor as shown by the boxplots and summary statistics. On the other hand, the logistic regression and random forest by TF-IDF are both comparable in performance but random forest takes the longest time to train i.e. 2.5 hours. We have managed to get a very satisfactory result from the classical classification algorithm. Deep learning algorithm with transformer-based universal sentence encoder and default parameters is the worst. It fails to correctly predict any "non-recommended" reviews according to the confusion matrix.

The classical classification algorithm of logistic regression with TF-IDF is recommended out of the other three classifiers considering the training time ($\approx$ 398 seconds), evaluation metrics (precision = 0.98, recall = 0.91, F1-score = 0.94 and area under PR curve $\approx$ 0.99) where it makes use of both the numerical and textual data.

1.7.1 Technology Choice

Please describe what technology you have defined here. Please justify below, why. In case this component is not needed justify below.

- Google Colab Jupyter Notebook for EDA.

- Google Cloud Platform Workbench Vertex AI JupyterLab 3 Notebook for coding machine learning training.
- gdrivefs 0.14.13: Mount to Google Drive
- spark-nlp 4.0.1: Spark session, text pre-processing and deep learning algorithm
- pyspark 3.3.0: Spark dataframe and pandas-on-spark dataframe manipulation, grid search, cross validation, classical machine learning algorithms e.g. logistic regression and AUC
- sklearn: Confusion matrix, classification report
- pandas 1.3.5, Numpy: Other non-Spark data manipulation e.g. dataframe, list, array, dictionary, tuple
- seaborn: Heatmap

1.7.2 Justification

Please justify your technology choices here.

Runtime crashes during model training in Google Colab due to resource limitation hence Google Cloud Platform Workbench Vertex AI was explored to continue.

Both are Google products and therefore less likely to be incompatible after some setups e.g. gdrivefs done to the environment.

Other modules and libraries allow for jobs of both distributed and non-distributed processing interchangeably e.g. converting Spark dataframe to Pandas dataframe to use seaborn for visualization.

1.8 Applications / Data Products

GitHub project repository

https://github.com/wongkhoon/Coursera/tree/main/Advanced%20Data%20Science%20with%20IBM/Advanced%20Data%20Science%20Capstone

1.8.1 Technology Choice

Please describe what technology you have defined here. Please justify below, why. In case this component is not needed justify below.

- Google Cloud Platform Workbench Vertex AI Workbench JupyterLab 3 Notebook
- Google Colab Jupyter Notebook

1.8.2 Justification

Please justify your technology choices here.

Google products and therefore incompatibility is likely to be minimal after some setups to the environment.

1.9 Security, Information Governance and Systems Management

1.9.1 Technology Choice

Please describe what technology you have defined here. Please justify below, why. In case this component is not needed justify below.

Not applicable

1.9.2 Justification

Please justify your technology choices here.

Not applicable