Infiniti Machine Learning Predictive Analytics

Overview

This document describes the Machine Learning and Predictive Analytics processes and architecture that the Infiniti team utilized as part of our KMT project.

Our data scientists ran the Machine Learning/Predictive Analytics as a separate sprint within the overall KMT prototype project.

Knowledge Management Tools (KMT) have ability to store extensive user interaction data. This data can be utilized to learn user behavior. We implemented a Machine Learning (ML) model that demonstrates the use of such data.

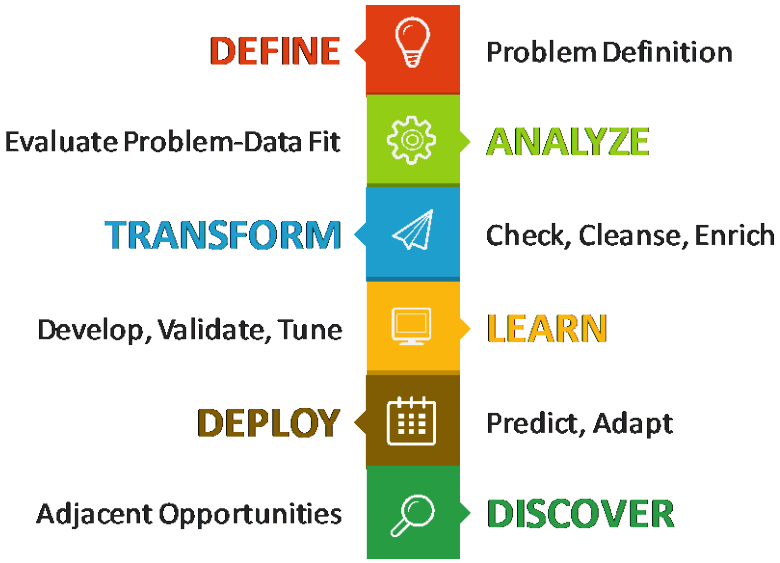
Machine Learning/Predictive Analytics Project Objective

The objective of the project is to utilize the valuable user interaction data in the KMT to improve user experience with the KMT. By utilizing past usage of knowledge articles by various users within the organization, it is possible to generate customized recommendations for each user as they log in to the portal. One example of this is the development of a recommender engine. The core idea behind the recommender engine is to identify patterns in the past knowledge article consumption of a given user and others similar to him/her to recommend new articles that might be of interest.

Machine Learning/Predictive Analytics Approach

At Infiniti we follow a rigorous, disciplined six-step process to solving predictive analytics problems. It is critical to have a disciplined approach to ensure clear business value is generated at the end of the project. The rest of this section will walk through each of the sections, its key goals and what we did for the KMT Machine Learning prototype. We also outline other typical tasks that are done in each phase, including ones that were not directly applicable to the prototype.

This project will involve the following tasks. The steps outlined below follow the general structure of our teams' approach to predictive analytics projects. The process has internal feedback loops not explicitly highlighted.



Define

Kick-off, Scope, Strategy: The goal of this step is to clearly identify the problem statement. Studying data can be an open-ended exercise that can lead to many possible inferences and outcomes. Not all of them generate business value to an organization. Laying out clear goals is critical to ensure a well guided machine learning effort.

For this KMT prototype effort our intention was to focus on adding tangible value to the KMT through machine learning. We had several ideas to utilize machine learning in this context. We settled on solving one of the most pervasive problems that users interacting with knowledge sources tend to encounter: That is getting the right information in front of them in the easiest, quickest possible means. We decided to build a recommender system engine to recommend new unseen knowledge articles that the user might be interested in.

Analyze

There are many ways to approach this problem. We settled on building a model based on user's knowledge article viewing habits. We picked this since it was easy and expedient to obtain this information from within the KMT's data infrastructure. There are many other sources that we could use to augment and improve the current model. We will highlight those in the "Discover" Section of our ML process.

Data Extraction: The data required for this model was user's knowledge article viewing habits. It is essentially a log that tracks articles viewed by each user over time. Since the database was set up as a prototype we didn't have real user interaction data so we created a dataset that mimics various user, articles they have viewed and the frequency of use for each article.

Phase Note: In typical machine learning projects that don't involve generated data, we go beyond identification of data sources in this phase. We study the data through exploratory data analysis. We understand the data quality, gaps in data, study statistical indicators of input and output variables and so on. One critical outcome of this exercise is the conclusion of whether

there is a problem-data fit. If the data sources don't have the underlying indicators of success, we head back to the define phase to discuss alternative problem definitions (or) look at ways to augment/replace the existing data sources.

Transform

Data Preparation: The generated user-article viewing frequency data is treated as an implicit rating of the article by the user. If the user viewed an article greater then say 10 times, we inferred that the article is of high interest to the user and graded it a 5 (highest rating on the scale). In this manner, we normalized the viewing data on a 0.5-5 rating scale. Articles that the user has not viewed as left unpopulated. Through this approach we created a matrix of users and articles with ratings for each valid combination of user-article. This was the input to the machine learning model.

Phase Note: In some machine learning projects, there are other aspects that we address in this phase. Feature Engineering is a key aspect of this phase. The term encompasses transformations such as deriving new useful features from available data, enriching data sources through external contextual information, reducing dimensionality of data where relevant, and so on. In cases where we try to implement a classifier, we study the balance (or lack thereof) of classified samples in data. If the number of data points for a particular class is very small in comparison to the total data set, we will apply compensatory techniques to enable the model to yield stronger results. In the end, the data will need to be converted (typically) to a numerical basis before feeding the data for training a model

Learn

In this phase, we train various machine learning models, evaluate the model against standard metrics and pick a model (or) an ensemble of models as our final model. We then continue to fine-tune the model for performance. Below are some details for the recommender system model.

Define Error Metrics: Define metrics utilized to evaluate model outputs. We will use a combination of the following error metrics to support our model performance analysis.

1. Root Mean Squared Error (RMSE)
2. Mean Absolute Error (MAE)

Evaluate Multiple Models: Run a variety of initial models against the dataset. We built a Collaborative Filtering (CB) based recommender system model for this prototype. Various core algorithms can be used in CB based approach. The following algorithms tend to work well for our recommender system scenario given the dataset:

1. K-Nearest Neighbors (KNN)
2. Singular Vector Decomposition (SVD)
3. Non-negative Matric Factorization (NMF)

Given that we were dealing with engineered data in the prototype, we didn't pursue aggressive tuning of the model. We identified the KNN model as performing well for this dataset.

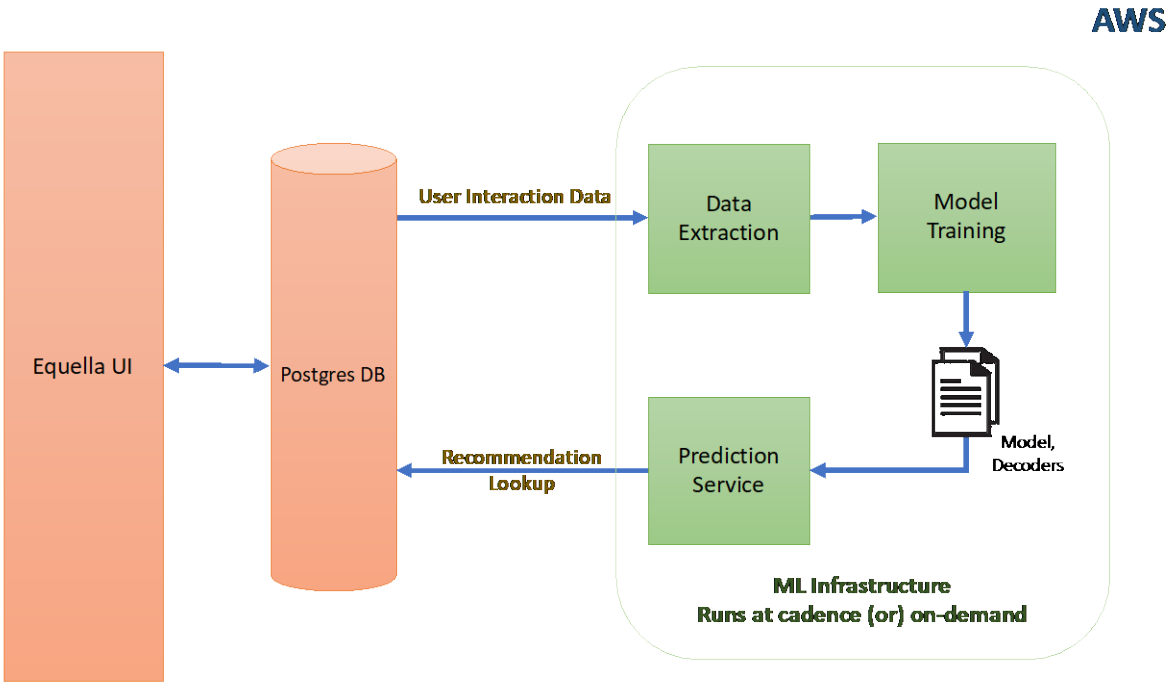
Other Aspects: There are a few other aspects to consider when choosing a model. We perform a cost-benefit analysis based on client inputs to settle on a model of choice.

1. Model Complexity: The more complex the model, the greater the time/effort to tune the model. They also require more work maintenance of the models in production environments.
2. Intuitiveness: Some models like Artificial Neural Networks, Support Vector Machines tend to be less explainable in terms of predictive outcomes than others like Linear Regression, Nearest Neighbors and some tree-based models. Depending on the domain, visibility might be a significant factor (Eg: Security, Financial Modeling)
3. Compute: Some models require a lot more computational power than others. Factors such as cost, ability to spin quick models would result in models that are computationally efficient.

Phase Notes: In typical machine learning projects, we will continue this phase through the study of learning/validation curves. This gives us a good understanding of how well the data (and more of it) reflects in the model performance. We fine-tune the parameters to generalize predictions for data that the model hasn't seen yet. It helps understand limitations both in data and the model. We also study ensembling (or combining) different models together as a potential final solution.

Deploy

In this phase, we integrate the solution into the client's infrastructure of choice. Our prototype was integrated into the Infiniti Knowledge Management solution deployed in AWS. Outlined below is a high-level architectural view of how the implementation looks in production.



Discover

We would like to highlight that there are several ways to improve the performance of this model. We plan to address these in upcoming sprints to further refine the prototype.

The current model utilizes only a subset of usable rich data in the Postgres database. Outlined below are data elements that can improve the predictive capabilities of the model

1. **Beyond Articles Viewed:** Integrating measure like timestamps, duration of viewing for each document, will further help the model. Articles viewed a long time back should have a lesser effect on recommendations than recently view articles. Articles viewed for a long time indicate greater value than those that were viewed only briefly.
2. **User Profiles:** Comprehending the profile of users, their interests, roles will allow Machine Learning models to generate a user-profile.
3. **Article Profiles:** Comprehending the profile of articles, including key tags, type of document, title and its content, contributors will allow us to create a detailed Article Profile. We will deploy Natural Language Processing (NLP) techniques to study the internals of documents to extract key themes and topics. This will further aid Machine Learning models to refine recommendations.

With data from #2, #3, we can deploy a class of models called Content-Boosted Collaborative Filtering models (or) Hybrid Recommender systems with this data.

Machine Learning and Predictive Analytic Tools

The following table summarizes the tools utilized in the project.

Vector	Tools
Infrastructure	AWS: This include EC2-instances, S3 Storage to support extract from Postgres, ML back-bone.
Language	Python 2.7
Packages (Python)	**Scikit-learn, Pandas:** These are robust open-source Python packages for data science and machine learning. **Surprise:** An open-source recommender system building package. These packages are actively developed and are built and optimized for speed and efficiency. Using open-source solutions allows us to do rapid prototyping and development while keeping costs low.