**Case Study – Assignment Group Prediction from Service Now case data (draft)**

**Problem Statement:**

Automate the routing of new Service Now incidents to the correct Assignment Group (AG) based on the details available at the time of incident creation.

**Benefits**

* Reduce or eliminate delays in incident resolution
* Reduce or eliminate incident re-assignment effort for support groups.
* Reduce workload on Tier 1 support team who route incidents to Assignment groups

**Features and Target variable**

* The target label for the prediction was the AG name.
* The features used were
  + Incident requester profile and location
  + Incident short and detailed descriptions
  + Incident classification
* Incident data from 1 Jan 2018 to 15 May 2019 was used to create the model and validate it. The training data was from 1 Jan 2018 to 31 March 2019 and the validation data was from 1 April 2019 till 15 May 2019.
* Only Closed (Resolved) incidents were used to train and validate.

**Exploratory Data Analysis**

* Data volumes
  + The total number of incidents was around 529,000 out of which around 421,000 were Closed.
  + The number AGs were 297
* Observations about the data (TBD)

**Approach**

* Model
  + The model category is multi-class single label classification.
  + The classes are heavily unbalanced with 40 out of 297 classes comprising over 80% data.
* Training
  + Due to the unbalanced nature of the labels, the following 4 scenarios were used and the scenario which provided the best accuracy was used
    - 40 of the most frequently used AGs which comprise around 80% of the incidents. To this set a subset of the AGs most frequently used by the Finance group was added because the predictions of the AGs used by Finance was deemed important.
    - 100 of the most frequently used AGs which comprise around 90% of the incidents
    - All AGs
    - 5% of the least frequently used AGs were combined into a group called ‘Others’
  + The scenario where all AGs were used to train, provided the best accuracy.
* Validation
  + The validation set contained AGs which were not present in the training set. This is due to AGs being added to the application in April and May 2019.
* Algorithms used
  + RandomForest Classifier from scikit-learn
  + XGBoost Classifier from scikit-learn
  + LightGBM Classifier from scikit-learn
  + DecisionTree Classifier from scikit-learn
  + SVM from scikit-learn
  + Logistic Regression from scikit-learn
  + Tensorflow 2.0

**Prediction Reporting and Analysis**

* The following outputs were recorded for each training scenario and algorithm mentioned above
  + Model parameters like training and validation set details, model hyper parameters, AGs considered for training and validation and run date.
  + Probabilities for all labels, the label with the highest probability and the label with the second highest probability for each validation sample.
  + Confusion matrix in a flattened format for easy analysis in Tableau
  + Metrics like accuracy, precision, recall and f1 score overall as well as for each class.
* The metrics where compared across all the scenarios and algorithms to understand the effectiveness of each approach.
* (Conclusions from the report)

**Challenges**

* The classes being heavily unbalanced impacted accuracy of predictions. A large number of classes has less than 5 samples in the training set. The accuracy of classes under-represented were low and brought down the overall accuracy.
* There were labels in the validation set which were not preset in the training set. This impacted accuracy because labels unseen in the training were predicted wrongly.
* Due to the large amount of text in the Description features, the TF IDF created from the Descriptions had over 900,000 words. This caused performance and memory issues.
* The training set had both text features as well as category features. These needed to be processed differently. The text features were converted to a TF IDF and the other categorical features were converted to One-Hot encoded columns. These were then concatenated after converting the TF IDF to dense matrix. Category codes could not be used for the categorical features because the large numerical values in the coded columns would have introduced bias in the training set.