Technical Approach

The Dataset

We worked with a dataset that consisted of the following columns:

- Transaction ID
- Item
- Date/Time
- DayPart
- DayType

The dataset contained over 20,000 rows with over 9,000 transactions. The data was initially provided by kaggle as a csv file. We parsed this data using the pandas python library, converting the csv file into a dataframe. From this point we could preprocess the data in order to filter out any erroneous or unnecessary entries.

Preprocessing

Given the litany of results we would like to produce with this dataset, it made little sense to preprocess the data the same way for each step of the technical approach. For calculating Support, Confidence, and lift little preprocessing was needed. A quick audit of the dataset revealed that it was very clean initially, requiring little preprocessing.

In order to run the FP-Growth algorithm, we had to create a dictionary of transactions each containing all items in that transaction. Given that the original dataset simply had items bought correlated with a transaction ID, we had to iterate over the dataset and add each transaction ID to a dictionary, adding any items found in that transaction ID to the dictionary as well. After this is done, we have the dictionary required to run the FP-Growth algorithm.

Support & Confidence

Support is a key metric in finding what items are most popular in a given dataset. The motivation behind using such a metric was to find what items should be stocked to what amounts on any given day. Further, finding the support of each item during a given season, day of the week, or part of a day, gives further insight into what items should be stocked to what amounts in a given specific case. Calculating support is simple, just count the number of entries that match the criteria you are looking for. We once again used python to calculate both the support and confidence.

Confidence is another key metric in finding what items may be bought together in a dataset. The motivation behind using such a metric is to find what items can be offered as packages together to increase sales. Calculating confidence is similarly simple to calculating support, simply create a dataset consisting of all items that fit the antecedent. Then count the proportion of rows which contain the consequent item.

FP-Growth

We then used FP-Growth to find rule associations in our dataset. In order to run the FP-Growth algorithm, we used the mlxtend python library. However, for the technical approach a

summary of the algorithm is the following. FP-Growth involves two main parts: the construction of the FP-tree and the mining of frequent itemsets. Constructing the FP-tree involves building an FP-tree, consisting of nodes each of which represents an item. The paths from the root to the leaf nodes represent transactions. The primary goal of this data structure is to compactly store the items in the dataset and their relationships. The algorithm scans the dataset only once to construct the FP-tree. Transaction reduction is then performed by merging transactions with identical itemsets, which reduces the memory required for the FP-tree. After creating the FP-Tree, the FP-tree is mined for association rules. The tree is then recursively traversed, finding the most associated items using conditional pattern bases.

The motivation behind using FP-Growth, and, more broadly the use of association rules, is to find items that are highly correlated to offer them as sales together. This is similar to the confidence metric, however this approach is much more targeted.

Lift

Lift measures the predictive power of a rule association generated by FP-Growth by comparing the observed support of the combined items in the rule with the expected support if those items were independent of each other. For our purposes, lift helps in determining the strength and direction of association between items. While rule associations between the most popular items and every other item will be prominently shown as results of the FP-Growth algorithm, these results are not particularly interesting. Since we already know that coffee is the most popular item at a cafe, it is important to find items that have strong correlations, even if they are less popular themselves, in order to find the best targeted advertisements and deals.

Naive Bayes

Decision Tree Classifiers