Wifi Log Anomaly Detection

Author: Szuhui Wu

What Has Been Done

run_wifi.py is a processor that returns a cleaned, aggregated (sessionalized) wifi log dataframe given the from-date and to-date

assign_user_type.py is a processor that match the users in wifi log to data warehouse, by either utln or email, and assign user type and role.

k_mean.py is a library that processes data to fit a k-mean cluster model, including feature normalization, transformation, model fitting simulation, model creation and prediction. See run_kmeans.ipynb for examples

Data Cleaning

- Filters: ex) exclude server/WAN connection, exclude sessions starting before the time range
- Field selection: the current version focuses on well-understood timestamps and categorical data such as ssid, ap, mac address etc.
- Cleaning: ex) user name

Key Data Transformation

- Created sessions partitioned by macAddress, ap, association time
- Created aggregated variables such as session length, lag length (time between 2 sesions), count of distinct ssid/userName/deviceType
- Matched utln/email to Data Warehouse, get active role(s) and user type
- Created continuous timestamp for time-span overlap cases
- Created categorical variables such as campus, buildings, part_of_date (morning/afternoon/evening/night)

Data Explored

exploration_plot.ipynb

- Data from 5/1/2019 to 5/7/2019 (week-level data).
- 50% of the sessions are under 1-hour, and 80% 2 hours. But the max session length within the selected date range is almost 7 days (never disconnect) - very high variance
- 50% of the session lag time (time between two sessions from the same mac address) are around 15mins, and 80% 2 hours. But the max lag length within the selected date range is almost 7 days - also very high variance
- *Lag length is systematic such that larger values are striated along common values.*
- Categorical variables, aside from part_of_day, are highly skewed toward normative values.
- 99% of the sessions are using the Secure network, mostly used by current and ex members. Most sessions from other networks (Roam, Wireless, Guest...) are short, 0-5 minute sessions, a different distribution compared to Secure.

- 98% of the sessions are from active(current) members, followed by individuals with email, mostly using Roam. The session length distributions echo those by networks.
- 80% of the sessions are from the Medford campus, followed by Boston, Grafton, SMFA.
- 39% of the sessions start in the afternoon, followed by evening, morning, and night. There's no notable difference in terms of session length distribution

Challenges and Decisions Regarding These Data:

- In raw form, data contains no continuous ratio data (no numeric values with a true zero).
- Certain clusters of behavior are not outliers, but seem illogical. For example devices occupy duplicated rows, usernames change mid-session, the same mac address has different device types (iphone, workstation, apple watch...), or two sessions from the same device overlap, and that groups of sessions disconnected after 0.3 minutes at the same associationTime -- are they anomaly or to be ignored? In some cases these may not be anomalies, but may indicate abnormal network performance.
- It is difficult to know if something is an anomaly. This may be because I lack the expertise to make judgement calls regarding network behavior. An expert would be necessary to separate patterns from noise. For a continued effort a network subject matter expert should be enlisted as a resource. Decisions made here regarding the use of categorical data are based upon data exploration and judgement calls about usefulness of such variables in the model, however an expert will see paths I did not.
- Session length and lag length are the only continuous variables in these data. Remaining variables are restricted to categorical. This limits approaches. Outlier analysis on large datasets is inherently complex: most sensitive approaches are strongly affected by outliers and so have difficulty detecting outliers. Here I have constructed continuous variables from categorical data which showed the most potential in data exploration (see above), but doing so is a suboptimal strategy. Having multiple continuous dimensions would assist.

Analysis Techniques Tested

- A number of techniques were considered. Factors in my final strategy selection included robustness to outlier, ability to run the approach natively on PySpark, and ability to use the approach with only two continuous variables.
- Considered approaches included Local Outlier Factor (LOF), Symbolic Aggregate approXimation (SAX), wavelet analysis, nearest neighbor approaches, K-means Clustering (and variants, including combining with hidden markov chains), Euclidian distance approaches for cluster outlier detection, hierarchical clustering, unsupervised random forest approaches. Of these approaches, K-means clustering followed by mean standard deviation-based cluster analysis showed the most promising results given the current form of data and resources at hand.

K-Mean Clustering

kmean_cluster_session_day.ipynb

kmean_cluster_session_week.ipynb

kmean_cluster_session_week_normalized.ipynb

run_kmeans.ipynb

- K-means was used to provide a multi-variate and multi-dimensional search across all possible combinations of the variables.
- Both one-week data(5/1-5/7) and one-day data(5/6) were tested for model fitting. The day data seems to be superior for the purpose of anomaly detection because the session/gap length variance is much smaller.
- The included model saved in this repository is based on the 1-day data on 5/6/2019 with session length and many categorical variables. It is in no way a sophiscated model, but can provide a view on how to create one and test it on a new data set. See run_kmeans.ipynb.
- Categorical data was recoded into numerical indices through dummy coding and translated (vectorized) into several binary indicators using OneHotEncoderEstimator. A lot of time was spent experimenting with different ways to aggregate the data, epoch, and generate new dimensions.
- Sampled the data and ran simulation at different number of clusters (k). Two measures were used to evaluate the optimal k: a. sum of squared distances of points to their nearest center (this may not be effective as a long term strategy, as it is a depreciated method on PySpark) b. silhouette score using the squared Euclidean distance, a measure for the validation of the consistency within clusters. This is also used in evaluating the model later.
- Experimented different variables level of data aggregation selection: adding lag length, using only categorical variables, various feature scaling techniques etc. The included Notebooks show part of this process.
- The cluster that has few data points (or the fewest) is by definition composed of outliers, which should be evaluated to determine if they are anomalous. Further inspection as a group and individually by an expert would allow separation of anomalies from outliers. Finally, we would need to build a refined model to exclude categories of non-anomalous outliers from future analysis.
- For each cluster, checked distribution of distances of data points from cluster centers (using the Frobenius norm/Euclidean norm, which is the square root of the sum of the absolute squares of its elements). This allows the isolation of outlier data lost within non-outlier clusters. By mapping out in histograms, it shows some appear in subgroups that break the rule of "the further away from the centroide the less data points". Others appear as step functions in density, which may identify disperse outer groups.
- It is no surprise that the centroids are dominated by session length because it is the only continuous variable with sufficient variability to drive diverse patterns.

What's Next

- Outliers identified by the model need to be further inspected by a subject matter expert to determine if they are true anomalies. The judgement should be incorporated into refining the model.
- K-means provides a path forward which can be run on our Hadoop Cluster today. Determining the optimal number (k) of clusters requires iteratively building a model. Perhaps a way to do it is to leverage the power of the HDFS to run a year of data day by day, and examine batches of aggregated outliers with an expert, fine-tuning as we go.
- One interesting idea, which I just started to do, is to look at the "movement" of a device (from building to building). By calculating the distance and the lag time between sessions, we may find outliers that, for example, move at an unreasonable speed. This can be an implementation on top of the movement analysis presented before.
- Device type needs to be further grouped. It is found that a device can be indentified as different device types at different access points. By grouping the device type, we can flag, for example, a mobile device that contains two user information.
- Additional data scaling methods should be explored. Scaling data is usually necessary for

distance-based algorithm. However, popular methods such as min-max and mean-sd are both sensitive to outliers, it may increase or decrease the effectiveness of outlier detection.

- Several approaches that would require more extensive work to run on our cluster nonetheless show promise. Local Outlier Factor (LOF) would provide greater sensitivity and signal detection (fewer false positives). Libraries are available in Python, R, and Spark Scala.
- Symbolic Aggregate approXimation (SAX) is a computationally cheap approach with excellent motif detection. It takes the data as "ordered" time series data. But is implemented in Python, and would also require more truly continuous dimensions and further feature engineering to the data.
- Principal Component Analysis (PCA) was considered, and may be an excellent strategy with a much larger (years scale) dataset. As it is highly affected by outliers, using Robust PCA would be necessary. Another suggestion I came across is combining hidden markov chains with clustering - first clustering the data, and then using clusters as the states in a Markov Chain and building a Markov Chain. Also, there's a variation of k-means known as k-modes, which is suitable for categorical data.