Mobile App Login Failure Analysis & Machine Learning Prediction

Donald Gennetten

May, 2017

# Introduction

With increasing speed to market pressures, technology software delivery teams are continually prioritizing feature development against technical debt. This results in challenges detecting, measuring and resolving issues impacting end users.

With more than 125 million opportunities to fail each month, a 1% failure rate would equate to over 1.24 million negative customer experiences. This significantly increases customer dissatisfaction which may even reduce market share.

The goals of this project were dependent on using statistical methods and machine learning to identify and predict problems that are not known, or are otherwise difficult for the business to identify.

* **GOAL 1**: Identify Login failures impacting customers using the iOS servicing app. Business/Product Ownerswill have improved visibility of issues allowing them to refine their delivery roadmap and drive prioritization of technical debt and other fixes which impact end users.
* **GOAL 2**: Identify correlations with available device, iOS and App data in order to offer machine learning prediction on statistically significant predictors. Platform/Technology/DevOps teams will be able to identity production support, capacity and infrastructure needs.

# Data Set

Data was extracted from APIs, login activity logs, and publicly available device manufacturer lists.

Customer login activity was extracted from real-time data streams, aggregated in hourly counts (using Splunk) then exported into daily CSV files for processing in R (**see Fig. 1 – Important Categorical Fields**).

Device Manufacturer data was collected from widely available online lists and wrangled into a clean lookup table (**see Fig. 2 – Example of Raw Device Manufacturer List Fields**).

All files were then wrangled, joined into a single data frame and used for analysis and prediction in R/Rstudio.

The following R Libraries were installed and used.

* dplyr, tidyr, data.table for data wrangling and transformation
* Ggplot2, scales for visualization
* caTools, ROCR and effects for Machine Learning

**Data Limitations & Collection Plan:**

* Steps were taken to ensure sensitive and proprietary data was not included in raw data files. Impact on the final output for this project was minimal.
* Login attempts resulting in fatal device level failures, and/or where there was no connection to the API, will not be reflected in the data. These failures will therefore not be reflected in final results.
* Login volumes were aggregated at the hour due to projected total raw file size(>600Mb) and machine memory(<6GB) limitations. This is believed to have low impacting on the final output as weighting was applied where appropriate for distribution by volume and Logistic Regression.

**Fig. 1 - Important Categorical Fields:**

NOTE: This is a subset of the total data and represents the top categorical values used in analysis and prediction.

|  |  |  |
| --- | --- | --- |
| Field Name | Sample Values | Definition |
| APP\_VERSION | 8.28.1, 9.16.0, 9.15.0 | Code version for the installed mobile application |
| AUTH\_METHOD | Password, Finger Print, Pattern | Method used by the user to authenticate |
| CHANNEL\_\_TYPE | MOBILE, WEB | Channel used by the customer during Login. Always expected to be “MOBILE”. |
| DEVICE\_OPERATING\_SYSTEM | iOS, iPhone OS | Operating system installed on the mobile device |
| DEVICE\_OPERATING\_SYSTEM\_VERSION | 10.2.1, 6.0.1, 9.3 | Operating system version installed on the mobile device |
| APP\_TYPE | iPhone, iPad | App type installed on the device |
| RESULT\_DISPOSITION | SUCCESS, POLICY, DEFECT | General business result from a login attempt. SUCCESS = Successful login, DEFECT = Failed login due to technical issue, POLICY = Failed login due to business rule (Ex: Invalid Password) |
| DEVICE\_MODEL | iPhone5,3, iPhone8,1 | Unique device model identifier. Used as lookup to get friendly product names |

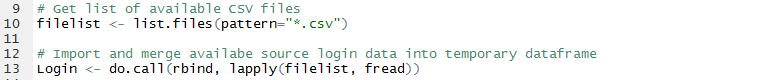
**Fig. 2 – Example of Raw Device Manufacturer List Fields:**

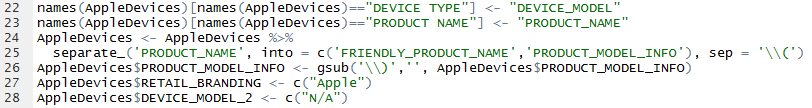
|  |  |
| --- | --- |
| DEVICE\_TYPE | PRODUCT\_NAME |
| iPhone7,1 | iPhone 6 Plus |
| iPhone7,2 | iPhone 6 |
| iPhone8,1 | iPhone 6s |
| iPhone8,2 | iPhone 6s Plus |
| iPhone8,4 | iPhone SE |
| iPhone9,1 | iPhone 7 (A1660/A1779/A1780) |
| iPhone9,2 | iPhone 7 Plus (A1661/A1785/A1786) |
| iPhone9,3 | iPhone 7 (A1778) |
| iPhone9,4 | iPhone 7 Plus (A1784) |

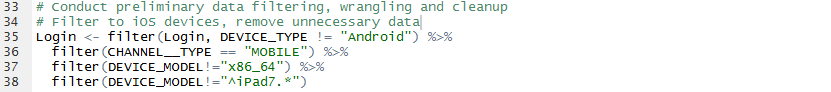
# Data Wrangling Approach

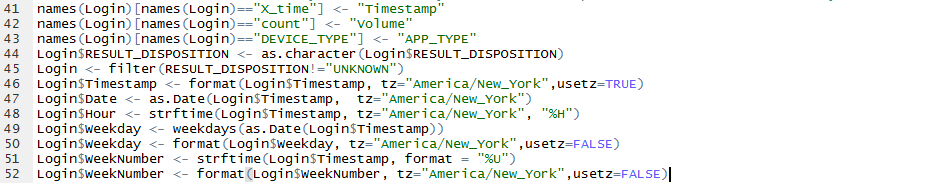
The following approach and high level steps were taken to ensure the final data frame was clean, complete and “junk” or unnecessary data was removed.

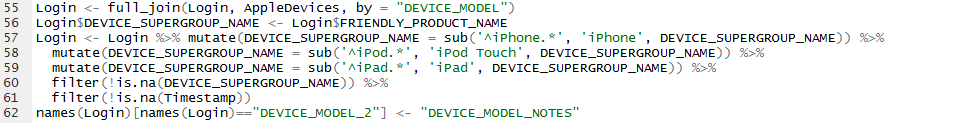
1. Daily raw login activity CSV files were combined into a single data frame using rbind and lapply. (**see Fig. 3 – Login Data Import & Bind**).
2. The Device Manufacturer list was cleaned and PRODUCT\_NAME was separated in order to convert DEVICE\_TYPE into the more widely recognized friendly marketing name. For example, iPhone9,4 maps to iPhone 7 Plus (**see Fig. 4 –Device Manufacturer Wrangling Code**).
3. The login activity data frame was then wrangled to remove outliers, out of scope data or any non critical values that were not necessary to include. (**see Fig. 5 – Remove Unnecessary Data**).
4. Corrected column names for accuracy and consistency then formatted timestamp and created Date, Hour, Weekday and Week Number for filtering and evaluation as predictors in Machine Learning algorithm (**see Fig. 6 – Column Name & Timestamp Changes**).
5. Merged Device Manufacturer and Login Activity into single data frame and completed final wrangling (**see Fig. 6 – Join & Final Wrangling**).
6. Collected identified “junk” data and outliers in the event they became a relevant or significant in analyst, which they did not (**see Fig. 7 – Data Quality Checks**).

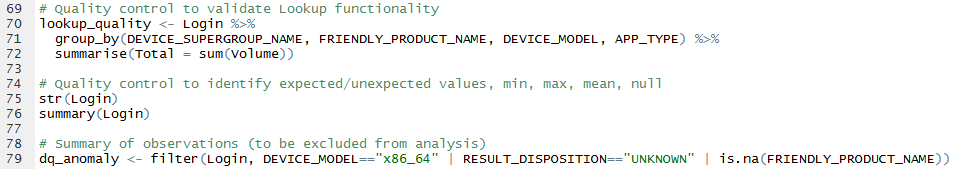
**Fig. 3 – Login Data Import & Bind:**

**Fig. 4 – Device Manufacturer Wrangling Code**:

**see Fig. 5 – Remove Unnecessary Data:**

**Fig. 6 – Column Name & Timestamp Changes:**

**Fig. 6 – Join & Final Wrangling:**

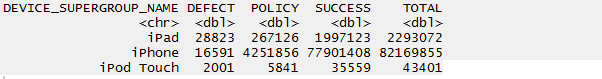
**Fig. 7 – Data Quality Checks**:

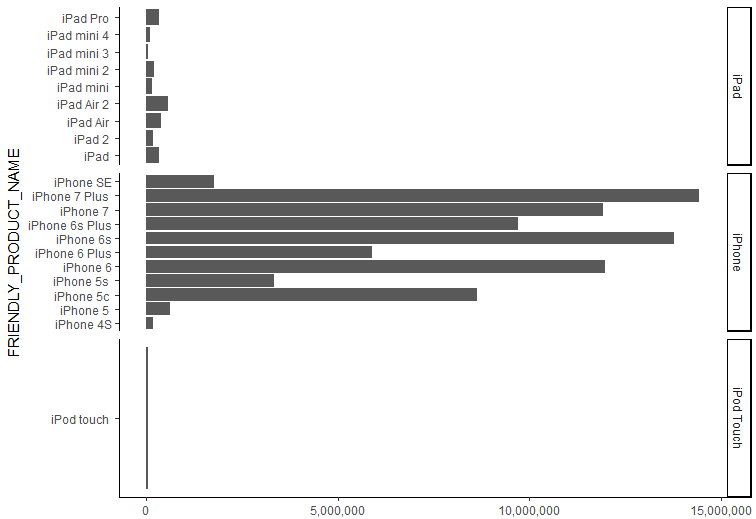
# Code Walkthrough of Analysis & Top Observations

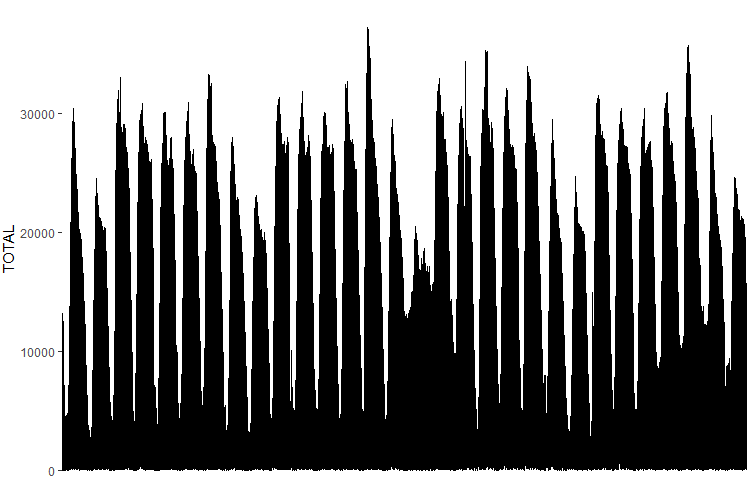
The cleaned data represents ~82MM iOS login attempts during the month of April, 2017. A top down approach was taken in this analysis by first I checking Phone, iPad and iPod volumes, failure rates and policy rates. This was followed by drilling into more granular device type and comparing failure/policy rates with various data dimensions to identify any patterns, trends or general observations.

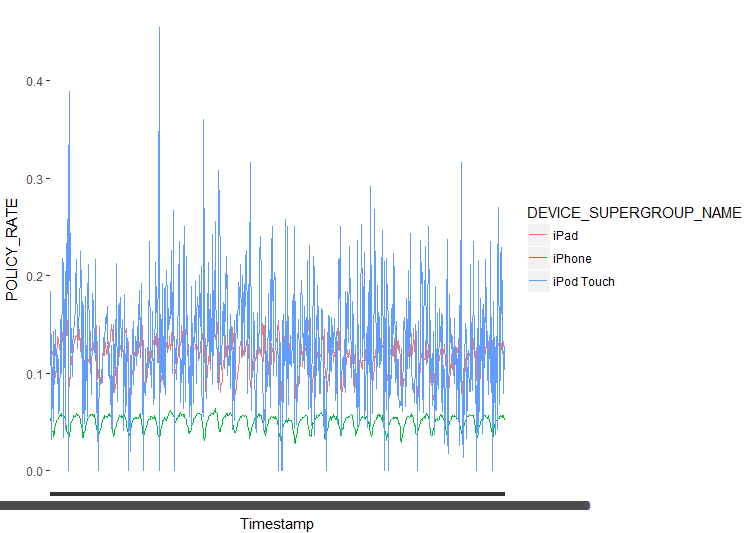
The following summarizes the key observations and code seen in **MobileLogin\_Analysis\_Apple.R**.

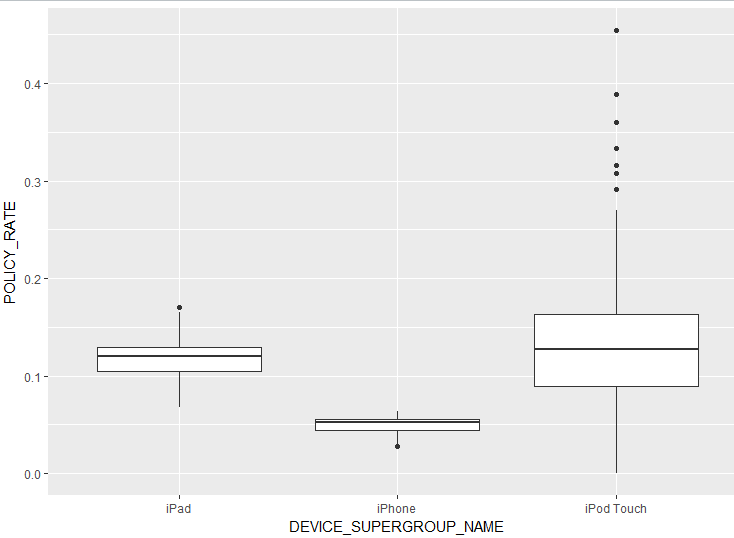
**LINES 84 – 181: TOP LEVEL LOGIN VOLUME, POLICY/FAILURE RATE**

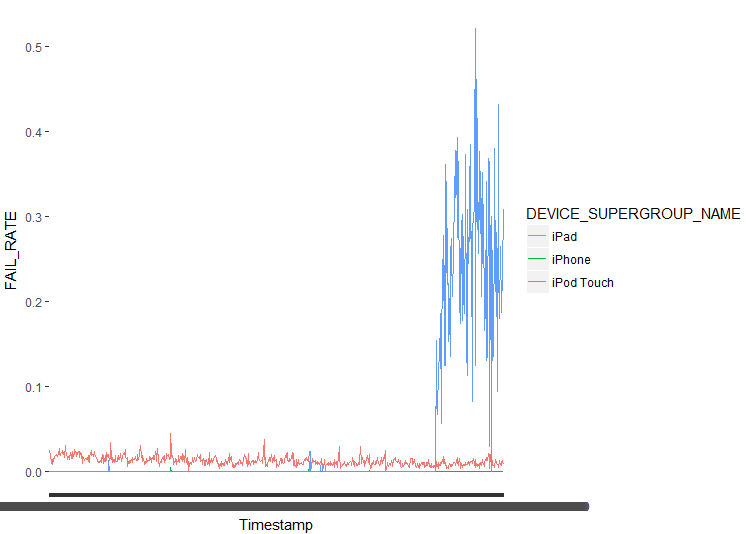
@ LIne 86: At ~82.2MM, iPhone represents over >97% of all iOS logins

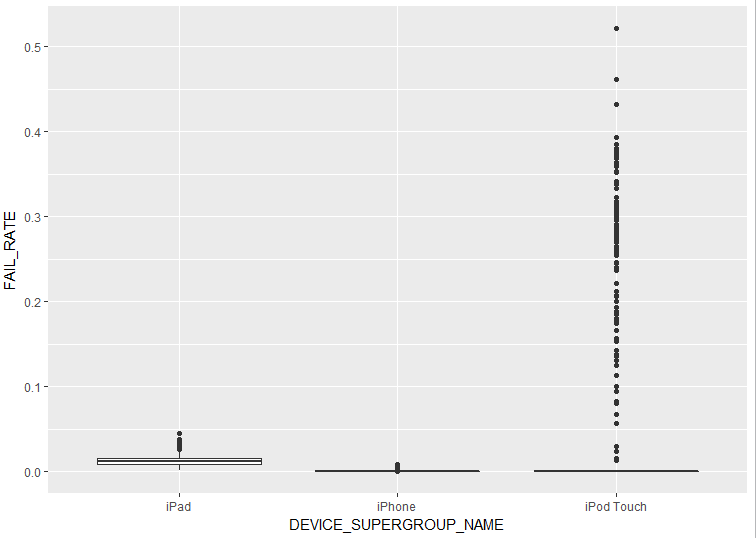
@ Line 100: There is a large number of iPhone 5 users

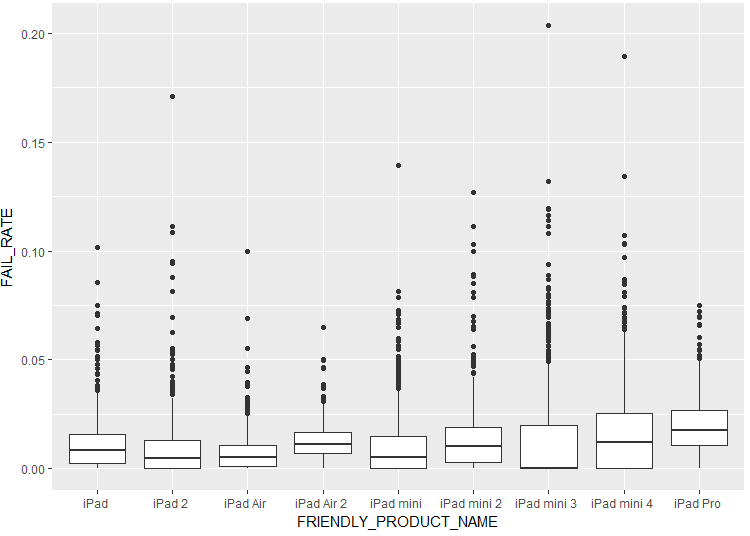
@ LIne 118: Total hourly login volumes over time appear normal; Decreased volume mid-month attributed to a holiday weekend (Easter Sunday)

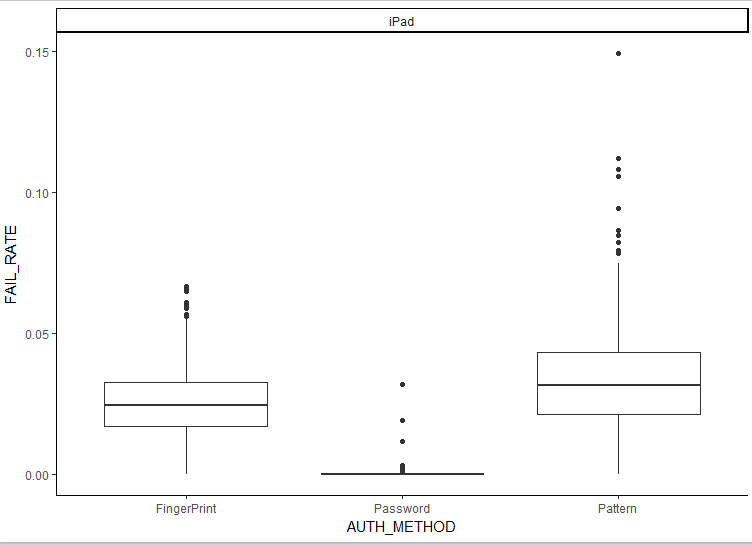
@ Line 156: Hourly policy volumes by device super group don’t appear to have any significant step changes up/down thought the month; iPad and iPod appear to be significantly higher than iPhone

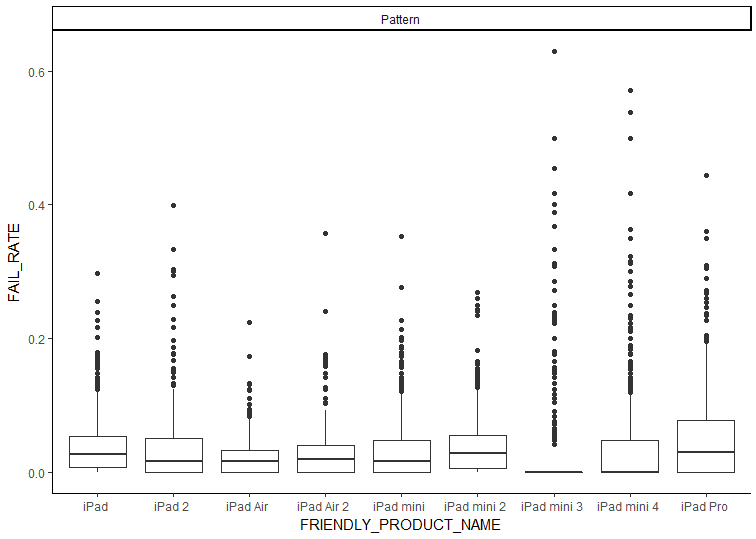
@ Line 162: iPad and iPod hourly policy rates are centered at nearly double the rate of iPhone

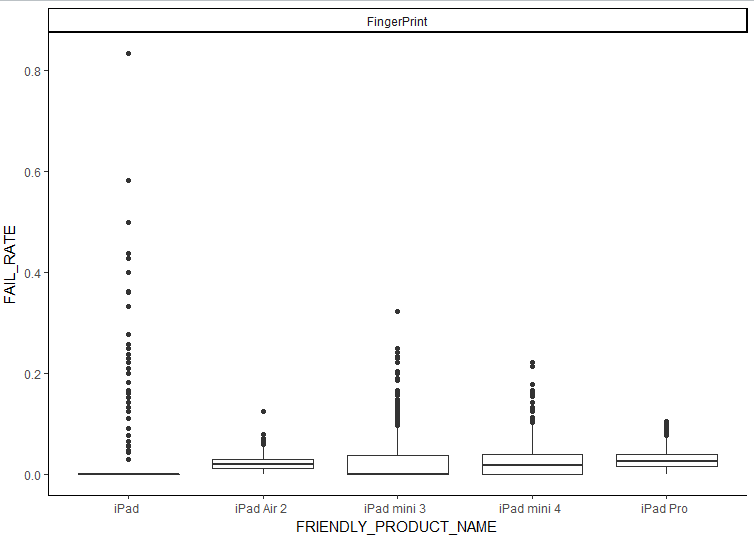
@ Line 168: iPod failure rates appear to have spiked during the last week of the month. 

@ Line 174: iPad and iPod failure rate are difficult to read due to scale, but iPod variation is clear due as seen

@ Line 238: iPad hourly failure rates appear relatively consistent across different iPad types. Presumably the elevated failure rate among iPad is attributed to a defect in the iPad app code. 

Line 275: Further investigation appears to show the elevated failures are specific to login attempts using Touch-ID and Pattern, not User ID and Password

Lines 319 & 321: Digging deeper to see hourly failure rates by Authentication Method and iPad type may offer helpful clues. 



# Algorithm Training and Testing

Algorithm training, testing and output is documented in the slide deck located [here](https://github.com/Sigmium/capstone/blob/master/Capstone_Final_Slides.pdf).

# Results & Proposed Next Steps

This model has been developed and tested to provide the following results:

* An overall accuracy rate of 88% for any login result.
* A true positive rate (login failure predicted as failure) of 56%.
* A false positive rate (login success or policy predicted as failure) of 11%.
* Increasing the true positive rate by 2% is possible but false positive rates will increase by 1.5%

**NEXT STEPS:**

* + Conduct risk assessment of large iPhone 5c population using old iOS versions.
  + Evaluate iPad app codebase to determine root cause of disproportionately high failure rates.
  + Improve prediction capabilities thru Feature Engineering production incident and user level data.
  + Predict fatal app crash errors thru measuring device mix proportions to determine what is missing.
  + Improve true positive rates by excluding failures associated with special cause incidents then re-run Logistic Model training.
  + Expand analysis to logins from Android devices.