**Capstone Project**

**Documentation**

# Process overview

The following diagram shows the overall end-to-end process for defining, designing, and delivering the Capstone project.



# Problem statement

### What is the problem being investigated?

Possible Me & Co. is a start-up in the mobile application development space. The product owner had no time to read the customer reviews concerning the mobile application. Because most of the people working at the start-up were busy freelancers, the owner proposed that the data scientist create a micro-application to help automate the process of evaluating customer sentiment on a product, to prioritize tasks for the IT team. I, the data scientist has been tasked with resolving this issue for both the short and long term.

### Why is this problem valuable to address?

* This will enable the team to have a ground truth to evaluate whether or not customers are happy with the product without having to read through every single review on every product in the portfolio.
* Addressing this problem with big data tools allows the process to be integrated into the workflow pipeline, allowing teams to immediately see the sentiments of customer reviews in real time.
* Having sentiment analysis as part of the data pipeline enables the team to get a visualization metric that helps visualize the effect of feature deliverables quickly. This means a faster mean time to error (MTE) than before.

### What is the current state?

* The manager has a barrage of customer reviews for 20 different mobile applications he is managing at once.
* The communications between staff at Possible Me are not as productive as other start-ups. Outdated workflow technologies are still used. Even though team members have proposed that the manager use better project management software and methodologies, the response is unconvincing.
* As a result, once an application feature is up and launched as a beta test, the next iteration of Kanban meetings is only decided once the blooming project hype has already died down. Hence most of the users go to other competitors for better substitute products.
* Customer complaints are not attended to as too much attention is focussed on applications that are already doing well. As a result, prioritization of fixing declining software relies on understanding customer sentiment of an app at a glance.

# Industry/ domain

### What is the industry/ domain?

The domain being discussed is the software development domain, more specifically mobile application development for Android and iOS. Applications can be made for mobile or for the web. Usually, a start-up like Possible Me will want to do both.

### What is the current state of this industry?

The mobile application market was valued at 154.05 million USD and is forecast to grow at a 11.5% compound annual growth rate (CAGR) from 2020 to 2027. The growth is primarily due to the rise in Internet usage by developing countries such as China, India and Brazil.

Also, with new technologies like 5G, ecommerce, IoT, machine learning, AR and VR, gamers and shoppers a like have been purchasing more premium applications from the App Store or Google Play Store.

While the surge in the ability of technology to interact with the consumer is a good thing for sales, this also means that Cost of Goods Sold (COGS) tends to increase. Even while sales may increase in parallel with innovation, IT teams have become more pressured to speed up cycle time delivery to keep up with ever changing demands. Therefore, IT teams are now adopting Agile approaches to speed up software delivery.

### What is the overall industry value-chain?

Ever since the backlash of the 1980s towards heavyweight processes to software delivery started to gain steam, a new movement called the Agile Manifesto began to take root. Frustration was looming over Waterfall methodologies that utilized a single phase Software Development Life Cycle (SDLC) from Planning to Maintenance without revisiting the planning phase. So, when the project was finished, the product became obsolete, and the competition moved on.

Modern mobile development uses one of these Agile methodologies. These include scrum, Kanban, extreme programming (XP), DevOps and others. As such the Agile value-chain is still similar as the Waterfall model, going through the 7 stages: Planning, Requirements Analysis, Design, Development, Testing, Implementation and Integration, Operations and Maintenance. However, Agile is an interactive approach.

For example, in Scrum, the IT team keeps a backlog of ideal corrections to the list of functional requirements. The product owner must then manage the scope creep at the next meeting or stand up. This process is repeated constantly in “time-boxes” and as such the value-chain in software development tends to be iterative and fast.

### What are the key concepts in the industry?

* SDLC: Software Development Life Cycle
* The Agile methodology: popular reference to the Manifesto of Agile Software Development
* Concurrency: This is how process requests are interleaved. For example, in an Online Transactional Processing (OLTP) system, transactions tend to be isolated if there is little tolerance for error. However, Online Analytical Processing (OLAP) systems tend to allow more interleaving trading off security for efficiency.
* Containerization: This helps us to build scalable applications that are operating system agnostic. This means that we can launch an app in multiple environments as the code is not dependent on any infrastructure. This is great for scalability as multiple code versions of a same product need not be replicated.
* Responsive Web Design: With the rise of modern web development frameworks like Angular and React, responsive web design is now more important than ever to capture market share. Techniques in this space can be invaluable as it provides the product competitive advantage.
* Processing: In cloud mobile computing, we often talk about batch processing and stream processing. Stream processing is good for real time use cases, such as image recognition of invoices giving instant feedback to the user. Batch processing is a lot cheaper in most cases, as it processes its workload in batches. Often batch processing is more efficient as it does not need a lot of resource overhead and can be scheduled as when needed.
* Workflow: When software people talk of workflow, they are often referring to CI/CD processes. This can involve software like Jira, GitLab, Bamboo, TeamCity, circleci, Jenkins etc. CI stands for Continuous Integration and CD for Continuous Deployment/ Delivery. In a nutshell, these services automate the integration and deployment of code, such that the iterative cycles of code changes and updates happen seamlessly through coded protocols that make the process easier and repeatable without much communication overhead between the teams.

# Stakeholders

### Who are the stakeholders?

|  |  |
| --- | --- |
| **Stakeholders** | **Concerns** |
| Customers | *Problem:* frustrated that their complaints in the Google Play App Store are not being responded to.  *Expectations:* thinks the problem should be fixed in a week |
| Manager | *Problem:* flustered with multiple project deadlines. Every time he wants to report budget figures to the financial manager, he is always put on hold as his team take a long time to decide.  *Expectations:* To get budget expectations early to the financial manager and expects there to be no scope creep before the reporting deadline |
| Product Owner | *Problem:* ignores scope creep of projects because of time constraints, even if some scope creep may improve the product and raise revenues for the period.  *Expectations:* Would like more time to work on scope creep if only there was not so much lag time on feature prioritization |
| Requirements Analysts | *Problem:* Knows what requirements are important and which are not and believes that the IT team and managers just like talking about non-functional requirements and never do anything to help team productivity nor increase ROI  *Expectations:* They want the budget allocations to be entrusted to the prioritization of critical feature prioritizations so that they can prove that they know a good feature when they see one and can get it approved by product owner |
| UI/UX Designers | *Problem:* Believes that the applications are not keeping up with current technologies and thus are falling behind.  *Expectations:* With the new data solution, they want to be able to prove to the product owner and hence the manager that the lack of investment in new technologies is what is holding them back |
| Web developers | *Problem:* Believes that it is futile to escalate issues to product owner even though not all responsive design checkpoints are met  *Expectations:* Want an automated framework where approval for features can be based on justifying scope creep via customer theme/ topic extraction although currently team is more involved in sentiment analysis. |
| IT Project Manager | *Problem:* Currently has no integrated communication platform. Uses Google Meet and Trello to sort projects, prioritization and requirements lists.  *Expectations:* Want there to be better approvals for better enterprise software for Project Management without resistance to vendor lock in. |
| Financial Manager | *Problem:* Does not think that the current investments in cloud technology for the current structural changes in SDLC into an Agile framework is worth the investment.  *Expectation:* Expects a baseline proof of concept can help accelerate a use case for initiating iterative planning and design as the norm. I.e. there should be a return on investment in the next redevelopment. |
| Data Engineers | *Problem:* Although the models can be compiled for predicting new customer complaint sentiments, there needs to be a pipeline to scrape new customer data into local server for experimenting with sentiment analysis  *Expectation:* |

# Business question

### What is the main business question?

* Question: How to increase delivery cycle speeds by prioritizing applications with negative sentiments from customers?
  + Answering this question in a data-driven way will help the IT team focus on high-priority applications. This means that applications that are already doing well can be focussed on later.
  + Also, it would be further helpful if the sentiment themes/ topics can be extracted and analysed. This will help the team know what they should be working on in the future.
  + Thus, value added to the company would be:
    - Increase in the ROI for the next short life cycle on software release/ deployment.
    - Increase in the speed to recovery for CD processes (needs benchmarking)
  + The analysis of sentiments of customers will aid this process of discovering the appropriate solution.

### What is the required accuracy?

* Since the proposed data solution would likely be a machine learning algorithm that tries to predict the sentiment of customers, we want the solution to be able to be reproducible and updatable throughout its lifetime.
* It is also important that the data pipeline can bring in fresh data every time the code is run. This can be in the form of a webhook or API call.
* A baseline accuracy of 80-85% (Paul Barba, 2019) would be reasonable for sentiment analysis, although surpassing that baseline would be ideal.
* ­­For this use case, false negatives would be more important than false positives. In other words, we are interested in precision rather than recall. This is because our problem is focuses on the predictive value of the model, instead of information retrieval.

# Data question

### What is the data question that needs to be answered?

* Question: What are the models and features give us the best accuracy and ROC scores when predicting sentiment of customer reviews on the Google Play Store?

## What is the data required?

* To answer the question, we need to frame the problem as a supervised machine learning problem that requires labelled training and test data.
* Training data
  + We require customer reviews for the training data preferably in the form of text.
  + These customer reviews should be analysed one at a time for better training accuracy.
  + The target labels should be for sentiment. This would either be a binary value or a polarity score. Preference for binary values as a ROC curve can be generated for better visuals.

# Data

### Where was the data sourced?

* For the main Exploratory Data Analysis(EDA) and predictive modelling:
  + Kaggle: <https://www.kaggle.com/ashraf1997/anz-synthesised-transaction-dataset>
* For testing the pipeline and scalability of the model:
  + Google Play: <https://play.google.com/store>
  + Utilizing a scraper: <https://pypi.org/project/google-play-scraper/>

### What is the volume and attributes of the data?

* Volume
  + Rows: 64295
  + Columns: 5

### How reliable is the data?

* This is a solid dataset as it has few nulls and missing values.

### What is the quality of the raw data?

* Volume
  + The data is very dense but not sparse. It only has 5 features, 2 of which are completely useless for this exercise. It’s density is the real strength of the dataset
* Variety
  + In terms of variety there are not a lot of types of data. However, there is variety in terms of the comments. The comments are from a large sample of each app and are translated from many languages.

### How was this data generated?

* The data from Kaggle was generated via the Google Play API, and was likely manually coded for its sentiment polarity and subjectivity
* The dataset contains 1074 Google Play Mobile Apps with Reviews ranging from 30 -320 per app. There is relative diversity in the groupings of Apps as they are arranged alphabetically without bias toward a particular market or genre.
* The sentiment labels have 3 categories, negative, neutral, and positive labelled in a set {-1, 0, 1}.
* There is also information on sentiment polarity and subjectivity that is valued on a continuous scale.

### Is this data available on an ongoing basis?

* Not in the Kaggle dataset. Thankfully, there is a Google Play scraper that Python uses which is able to scrape analysis data of mobile applications.
* Using the scraper, we are able to get new customer comments and manually label them to be fed into our model to test for new accuracies periodically.
* For CI/CD, this would mean that the code will need to scrape new data periodically for the interim where feature planning is a requirement.
* Scraping new data does not necessarily have to be customer sentiments, it can also be a visualization of how well the app is doing relative to other applications in the same space, and what improvements it should make.

# Data science process

## Data analysis

### What data pipeline was to wrangle the raw data?

The pipeline used to wrangle the Kaggle data was written in Jupyter notebooks.

* What are the highlights of the Exploratory Data Analysis (EDA)?
* Particular words in sentiment categories stand out more than others. A particularly good visualisation of these words can be done on a word cloud.
* Is the pipeline reusable? (for example, to process future data?)
* The Google Play Python scraper is technically reusable but the comments have to be manually labelled. There may be a way to use OCR to label these data but that would be a challenge.
* The pipeline would be reusable at a low level of granularity if the data being scraped is only information about app data. The pipeline can then clean data to be utilized for relative app analysis to elicit some kind of feature demand from theme/ topic selection.
* What are the intermediary data structures used (if any)?
* The analysis should also take into consideration what kinds of data are common to the analysis. In order to do customer segmentation, we need to know how funds affect the different categories of disability. Hence the data from our nested database that includes all the traditional categories being recorded, such as budget allocations which include:
  + Support Class
  + Age band
  + Disability Group
  + State
  + Service District Number
  + Report Date
  + Number of downloads
  + Number of installs

## Modelling

* What are the main features used?
* Did you find any interesting interactions between features?
* Is there a subset of features that would get a significant portion of your final performance? Which features?
* How did you select features?
* What feature engineering techniques used?
* What are the models used?
* How long does it take to train your model?
* What are the tools used? (cloud platform, for example)
* What are the model performance metrics?
* Which model was selected?

## Outcomes

* What are the main findings and conclusions of the data science process?

## Implementation

* What are the considerations for implementing the model in production?

# Data answer

* Was the data question answered satisfactorily?
* What is the confidence level in the data answer?

# Business answer

* Was the business question answered satisfactorily?
* What is the confidence level in the business answer?

# Response to stakeholders

* What are the overall message and recommendations to the stakeholders?

# End-to-end solution

* What is the overall end-to-end solution to use the model developed in the project?

# References

* <https://www.grandviewresearch.com/industry-analysis/mobile-application-market>
* <https://agilemanifesto.org/>
* <https://www.investopedia.com/terms/v/valuechain.asp>
* <https://techgeek365.com/5-software-development-concepts-to-improve-your-programming/>
* <https://www.lexalytics.com/lexablog/sentiment-accuracy-baseline-testing>
* <https://www.stitchdata.com/resources/what-is-data-pipeline/>