"Without big data analytics, companies are blind and deaf, wandering out onto the Web like deer on a freeway" – Geoffrey Moore

Introduction

It's hard to believe the growth that has been seen and the technological advances that have occurred over the past one hundred years and what is even more daunting is the thought of the growth to come. This report will analyse the software engineering process that we see today, but what's interesting to start with is when did this phenomenon begin?

In 1940, the first digital computers began to appear and the instructions to operate them were wired into the machine. However, professionals quickly noticed how inefficient this was and thus abstraction quickly began to be used to deal with the complexity of computing. Another major step in abstraction was the introduction of computer programming languages in the early 1950s. The origins of "software engineering" are more unsure, however, it is most commonly said to have begun when it appeared in a list of services offered by companies in the June 1965 issue of 'Computers and Automation'.

Despite being a relatively new field of engineering, its growth is amongst the fastest fields globally today. The rapid development of this field caused a multiplication in the number of fields it covered. Today, software engineering spans across the design, development and maintenance of software. It can also be divided into a number of subdisciplines including software requirements, software testing and software quality, to name but a few.

As the scope of the job broadens controlling and measuring the software engineering process is becoming increasingly difficult. The evolvement of the software engineering industry alongside the measurement and analysis used in the industry will form the basis of this report, "Measuring Software Engineering".

Throughout this report I will focus on four key topics, while also occasionally going outside the scope of these topics to discuss other related material. The four topics are as follows:

- Ways in which the software engineering process can be measured and assessed in terms of measurable data
- An overview of the computational platforms available
- The algorithmic approaches available
- The ethics surrounding this kind of analytics

I will finish the report with a conclusion of my findings and some personal opinions on the topics.

Ways in which the software engineering process can be measured and assessed in terms of measurable data

Collecting Software Engineering Data

Collecting data about the software engineering process certainly isn't an easy process as organisations strive to find the balance between collecting data that will be useful and beneficial to the success and efficiency of the company while ensuring the measurement process doesn't burden nor take too much time from development teams. Organisations are constantly working and developing methods to make this process as efficient and insightful as possible.

One historically successful approach to this process was put forward by Basili and Weiss in 1984. Their data collection methodology is goal orientated. It starts with a set of goals to be satisfied, then uses these to generate a set of questions to be answered, and then proceeds step-by-step through the design and implementation of a data collection and validation mechanism.

The methodology is summarised in six simple steps and I believe that by following these steps organisations can help improve the measurement of their data and in turn advance their data analytics systems. The steps are as follows:

1. Establish the goal of the data collection
2. Develop a list of questions of interest
3. Establish data categories
4. Design and test data collection forms
5. Collect and validate data
6. Analyse data

Measurement process

There are no standard metrics that every software engineering process should measure, as different metrics have different value to different products and processes. As Basili and Weiss recommend, organisations should focus on the goals of the data collection when deciding what kind of metrics to use. I also recommend that organisations should use a balance of both hard and soft data metrics in order to achieve the most representative results.

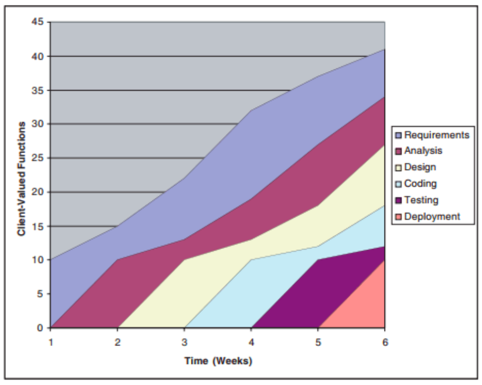
Hard Data Approach

I have evaluated a number of measures which help to review a software engineers code during the software engineering process. They are as follows:

1. Lead Time
2. Number of Commits
3. Code Churn
4. Technical Debt

Lead Time

Is the time between the initiation and completion of a process. It quantifies how long it takes between a software engineers ideas to be developed and delivered as software. Lead time can have a serious impact of the quality of data in some cases. It is a way of tracking how responsive software developers are to their customers. If a lead time is short, an engineer is generally more responsive to the needs of their customer.



Number of commits

This refers to the amount of time a software engineer contributes to their code. It is important to note that the size and frequency of commits does not correlate with the work needed to achieve a goal, i.e. more commits does not always suggest better progress nor productivity. Primarily, commits are an indication of activity. Thus, this metric is used to indicate which engineers are the most active and this can be important for project managers to analyse when comparing the productivity of certain team members, along with influencing decisions such as promotions.

Code Churn

This represents the number of lines of code that were modified, added or deleted in a specific time frame. Code churn tells you the rate at which your code evolves and is an excellent visualization tool of the development process. Churn gives especially important insights into the quality of the code. Research has shown that there is a strong connection between high volumes of code churn and number of defects discovered while testing. Therefore, keeping an eye on churn is very important as noticing spikes in churn can help diagnose problems early. When churn starts to spike, it can be an indicator that something has gone wrong in the development process, for example, if the same line of code has been changed multiple times it may suggest that this certain area is proving problematic. Although, closer to release dates churn may spike as developers are repeatedly polishing features so it is ready for release.

Technical Debt

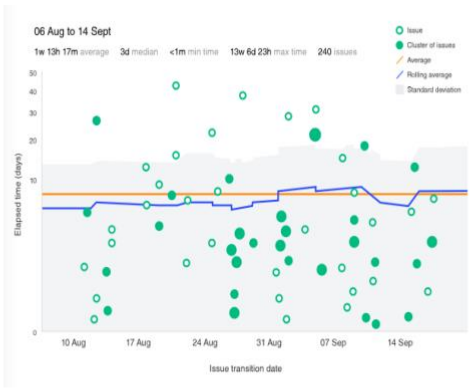
Can be also known as design debt or code debt and reflects the implied cost of additional rework caused by choosing an easy solution now instead of using a better approach that would take longer. If a technical debt is not 'repaid' it can accumulate 'interest' making it much more difficult to implement changes later on. For this reason it is a very important metric to keep track of. Technical debt is something that all software engineers strive to avoid.

Soft Data Approach

Above I discussed in detail a number of metrics which help to assess a software engineers data. However, one must not forget there is far more to the software engineering process than just writing code. Attributes such as ability to work on a team, peer interaction and ability to clearly express and present your opinions are equally important aspects of the software engineering process which cannot be explicitly measured by hard data alone. Giving promotions or making recommendations based on hard metric facts alone is too harsh and simply not representative of a software engineers work in my opinion. This is why I recommend that a certain level of human judgement needs to be included into the assessment of the process.

Agile process metrics

These metrics focus on how agile teams of software engineers make decisions and plans. These metrics do not describe the data but they can be used to improve the software development process. Agile metrics focus on measuring a team's progress and setbacks throughout the development cycle. Agile metrics help teams better understand their development process, making releasing software easier and quicker. Extreme Programming (XP) is a commonly used agile process model. XP values are communication, simplicity, feedback, courage and respect.



The ways in which the software engineering process can be measured and assessed is a huge field which is expanding day by day. The methods outlined above are only a few of thousands available which help to measure the software engineering process. The process must be measured consistently and accurately to ensure optimal success. I believe achieving a happy medium between hard and soft data is at the core of a successful strategy for assessing the software engineering process. Over-reliance on either type of data can lead to inefficiencies along with incorrect judgements.

Benefits of tracking and analysing software engineering metrics

Tracking and analysing the software engineering process will provide the business with the following benefits

- Increasing return on investment
- Identify areas/processes that can be improved
- Reduce costs
- Improve debugging performance

- Improve productivity
- Help teams track progress, set goals and measure performance

<u>Shortcomings when it comes to the measurement of the software engineering process</u>
Time and time again businesses incur issues with the measurement of the software engineering process as terms used to describe these metrics often have a number of definitions and many different ways to count or measure characteristics.

A good example that highlights the discrepancy in measurement is lines of code (LOC). LOC is a common measure in the software engineering process, however, there are two ways to count each line of code. One method is to count each line of code that ends with a return. However, some developers disagree with this method as it may count "dead code" or comments. To get around this, another method is to simply count each logical statement as a line of code. Therefore, a software package could have two very different counts depending on which LOC method is applied.

"Measuring software productivity by lines of code is like measuring progress on an airplane by how much it weighs." – Bill Gates

To counteract this type of discrepancy it is vital for organisations to stick with the same measurement method throughout the duration of a project.

Not only are there technical issues but there are also issues with the fact that statistics show that far too few companies have established software engineering measurement programmes. Whether this is due to resistance to high costs (estimated cost of 4% of development budget) is unsure. Additionally, software development teams often consider it more important to actually do the work than to measure it. This is obviously a misconception as measurement of data is imperative to success. Therefore, ensuring the measurement and collection process is easy is essential as otherwise, it will not be done.

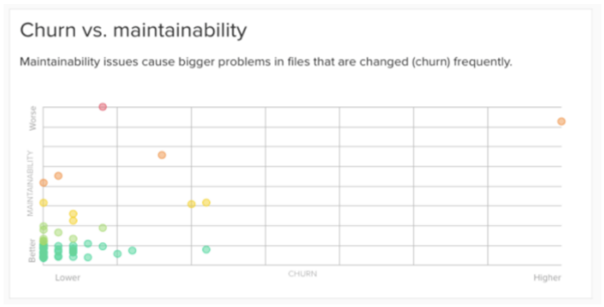
<u>**An overview of the computational platforms available**</u>

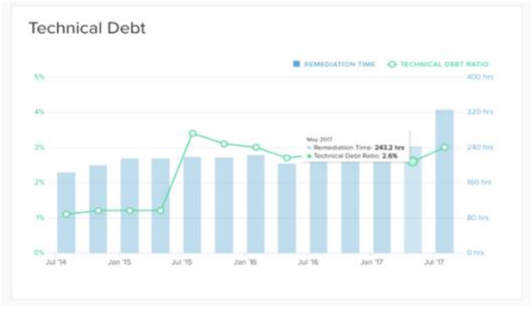
All of these measurements and processes are fantastic- but they mean nothing if there are not appropriate computational platforms available for them to be performed and assessed on. In an ideal world we would be able to collect and measure our information and produce insightful visualisations all in one go. However, unfortunately, in most cases raw data must be analysed and measured before conclusions are made. Below, I will review the computational platforms available today.

<u>Code Climate</u>
Code Climate is an online platform available to software engineers which efficiently measures and asses the data of a software engineer. Code Climate was set up in 2011 by Bryan Helmkamp and Noah Davis and now has over 20 employees and this number is expanding every year. Code Climate is used by over 100,000 projects and analyses over 2 billion lines of code every single day and it has been used by engineering teams in large global companies such as Salesforce and Intercom. Code Climate can work with a number of software languages such as Ruby, JavaScript, PHP and Python, to name a few!

The trade marked logo for the company is “Superpowers for engineering teams” and I believe it lives up to this expectation and is a very competent yet dynamic tool available for businesses. The platform focuses on two main aspects: velocity and quality. Their website states that velocity gives ‘data driven insights for your engineering process’ and quality focuses on an ‘automated code review and quality analytics’.

By choosing a package like Code Climate, organisations can visualize their data seamlessly and they are provided with meaningful insights. As aforementioned in the previous section of the report, code churn and technical debt are very useful tools in measuring software engineering and especially during the software development process. Code climate analyse the churn in your data and churn metrics will be left in repositories. They will help flag high churn, low quality files which need attention. Code Climate also provide insights into technical debt which identifies which files are frequently adjusted and changed yet have no increased value and have inadequate coverage. As technical debt is becoming of greater concern in the modern climate this is a fantastic facility that will not only save businesses time, but money too. Below are some examples of visualisations which are displayed on the ‘trends’ page of each user’s repository.





What is most striking about Code Climate is how effectively and easily it links to Git Hub. It runs and analyses code every time a new commit is pushed. This connection is both time efficient and convenient for the user.

<u>Codebeat</u>
Codebeat is another online platform which helps measure the software engineering process and assess results, so that the process is as productive as possible. Codebeat’s website states that they “gather the results of code analysis into a single, real-time report that gives all project stakeholders the information required to improve code quality”. Codebeat is interrogated with slack, GitHub, BitBucket and HipChat and it works on an extensive list of software languages including Go, Ruby, Java and Python.
Codebeat pride themselves on helping “developers write clean code”. They do this in four steps which I have outlined below:

1. Connect your Git Repository
   - This will allow Codebeat to track every change in your repositories
2. Check your full stack
   - Codebeat provide an automated code review and help users highlight issues which can be quickly and easily fixed along with prioritizing issues.
3. Merge with confidence
   - Codebeat help companies learn and visualize the impact of their changes, without leaving tools that are part of their workflow.
4. Manage your teams
   - Codebeat provides various team-management tools for example it can move people between projects within seconds.

No doubt, there are a number of other platforms available however, from my research, Code Climate and Codebeat seemed to be two of the market leaders. Other platforms include Atlassian, SonarQube, Codacy and Hackystat. These platforms all help organisations analyse and measure the software engineering process. The metrics they use and the visualizations and tools they provide may vary, however, their core purposes are all the same.

Analytics as a Service

Analytics as a Service (AaaS) is a new phenomenon which is generating a great deal of attention in the news. AaaS started because businesses wanted to get out of data centres and move towards cloud services as technology advanced. AaaS is defined as "the provision of analytics software and operations through web-delivered technologies. These types of solutions offer businesses an alternative to developing internal hardware setups just to perform business analytics".

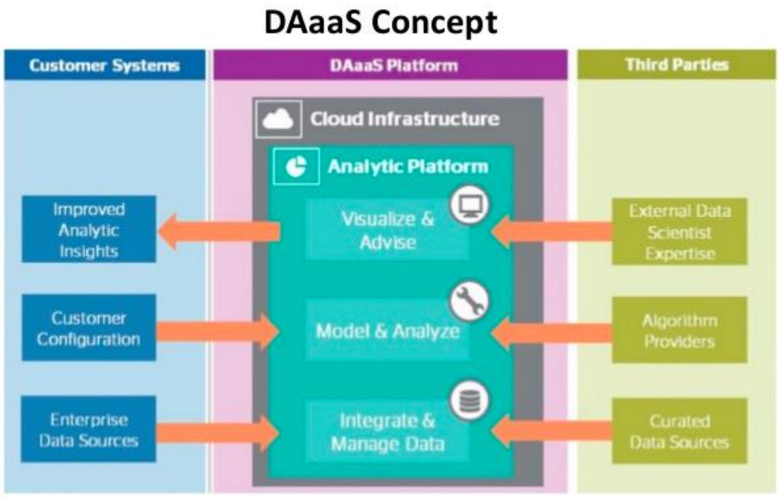
Analytics as a service has breaded success because setting up an analytics process in a business is an extremely intensive process which costs the firm both time and money, but by using AaaS businesses can avoid these new business costs. Another component that aided in the success of AaaS is its ability to combine data the company has already collected with outsourced components from the web. This equips organisations with powerful tools and empowers knowledge workers by granting them personalized access to centrally managed information sets. In an information-driven economy, companies who empower their staff with instant access to information will have a clear advantage.

How does it all work?

1. Analysts go to the web portal to request a personalised data sandbox with the data they need from the master data warehouse

2. Once access is granted, these analysts can utilize any of the visualisations or tools to investigate the data they now have access to

3. Upon completion they release access from their sandbox back to IT



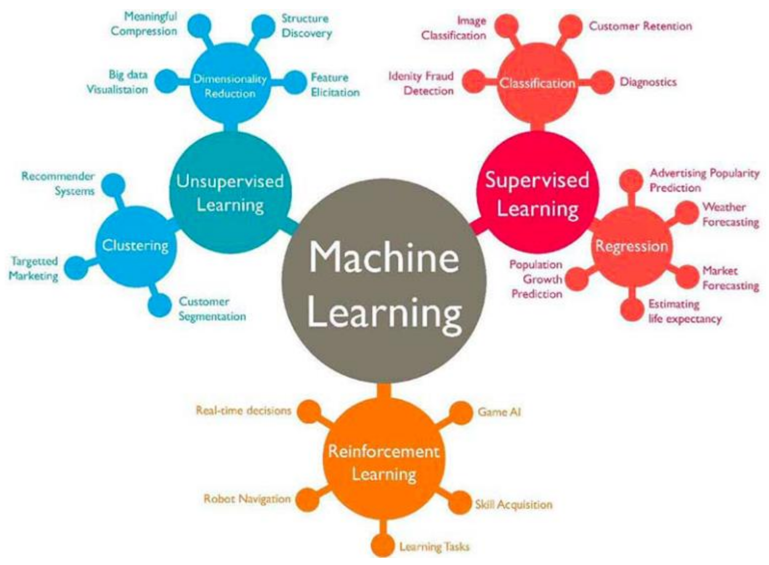
<u>The algorithmic approaches available</u>

There are numerous algorithmic approaches available which help measure and assess the software engineering process. When considering which algorithm to use, a company must consider, which ones are relevant and useful to us? There are infinite ways of approaching these problem, some of which are more suited than others.

Machine Learning

As Big Data is one of the hottest trends in the tech industry at the moment, machine learning is becoming an incredibly powerful and popular tool to make predictions or calculate suggestions based on large amounts of data. An example of where machine learning is used today is in Netflix's algorithms that make movie suggestions based on movies you have watched before.

What is important to begin is dividing machine learning up into three categories, supervised learning, unsupervised learning and reinforcement learning.



<u>Supervised Machine Learning</u>
In short supervised learning is when you have input values and output values and you use an algorithm to learn the mapping function from input to output. It is best practiced where a property (label) is available for a certain dataset (training set) but this label is missing for certain pieces of data and needs to be predicted. Supervised learning can often be related to a teacher supervising a lesson. The algorithm works by iteratively making predictions based on the training data and once finished it is corrected by the 'teacher' who knows the answer. The algorithm ceases when it achieves an acceptable level of performance. Supervised machine learning problems can be further split into classification and regression problems. Some examples of these problems include linear discriminant analysis, logistic regression, KNN and decision tree analysis.

There is often a bias towards using classification techniques because most analytical problems involve making a decision. Therefore I have focused on the k-nearest neighbours algorithm below.

<u>K-nearest neighbours (KNN)</u>
KNN is also a commonly used algorithm because it has a low calculation time and it is easy to interpret. In the algorithm, the analyst first has to classify the data or use data which is already classified. Next, the computer takes part of the data and assigns it as a test set. It analyses the data and variables in this tests set. Then, to classify the unknown point the algorithm looks at the k closest points of the known origin to the point of unknown origin. The point is then classified as belonging to the group which contains the most of these k points.

However, analysts should heed caution when using k-nearest neighbours as it is a distribution free method of assigning group membership, i.e. it makes no assumption on the spread of the data within each class. The consequence of this is that class assignment is fixed with no measurement of uncertainty concerning any particular assignment.

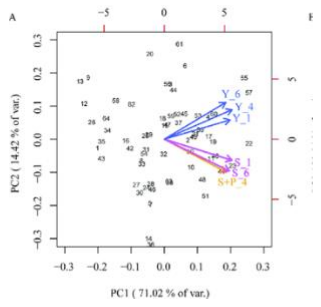
An example of this algorithm in practice would be classifying the developers based on their productivity levels. If a new developer joined the team, his data would be analysed and the algorithm would look at the k closest points of the known origin to this new developer point. The new developer would then be classified as belonging to the group which contains the most of these k points.

<u>Unsupervised Machine Learning</u>

In comparison, unsupervised learning is where you only have the input data and no corresponding output data. Unsupervised algorithms are used to solve very different problems, it is used in cases where the task at hand is to discover the inferred relationships in a given unlabelled dataset (e.g. they are not already allocated in clusters). Unsupervised machine learning problems can be further split into clustering and association. Examples of unsupervised algorithms include k-means clustering, independent component analysis and principle component analysis.

Principle Component Analysis (PCA)

PCA is a dimension reduction technique. It is useful as with numerous variables it is often difficult to comprehend or visualize inherent associations. PCA is a method for re-expressing the data so as to reveal its internal structure and explain its variation through the use of a few linear combinations of the original variables. The benefit of PCA is that it can summarise data through far fewer variables and with fewer variables, visualizations and results are much more meaningful and impactful.



The principle components are found through the linear combination of coefficients that represent the greatest proportion of variance in the data. This is found through the Eigen-decomposition of the covariance matrix of the data. The result of this decomposition is a matrix of eigenvectors and corresponding eigenvalues.

For example in a software engineering context, a manager could get a break down of the most important variables that are affecting the productivity of an engineer’s code, rather than having to read through every single possible metric. The most important and impactful variables will be highlighted by this method, reducing the amount of time needed to analyse every single component of the process.

Reinforcement Learning

Reinforcement learning refers to goal-orientated algorithms. Software agents determine complex objectives (goals) within a specific context and the algorithms learn how to attain these goals. The algorithm is called a reinforcement algorithm because they are penalized when they make wrong decisions and rewarded when they make correct decisions. It has been said that “under the right conditions they achieve superhuman performance”. As this is an automated learning scheme it implies that there is little need for a human with knowledge of the specific application and simply requires someone who is familiar with reinforcement learning. This means that much less time will be spent designing a solution, which increases both time and cost efficiencies. Some common reinforcement learning algorithms include Q-learning, SARSA and Deep Q-Networks.

The ethics surrounding this kind of analytics

Software engineering is the invisible force behind many of the applications and devices that power our day-to-day lives. Those who work in analytics are at the coal face of the digital transformation and opportunities for achieving social good within the field is huge. But how can we ensure that the data obtained from the measurement and analysis of these forces is done in an ethical way and is done for social good? Each day a new story breaks on yet another data breach at yet another company. So how can we stop this from happening? The ethics behind data analysis is a question which lies in a grey area and there has been a lot of concern and media attention surrounding the ethics behind this kind of analytics in recent times. While organisations understandably need to analyse their data in order to help them improve things such as efficiency, productivity and to increase profits, they are walking a fine line between ethical and unethical behaviours and in my opinion in too many cases what they are doing is unethical.

Studying and comparing the practices of two different companies who utilise data analytics in their companies is a nice way to see how these ethical issues are being handled at a corporate level. It is important to consider the reasons behind the data analysis and then the impact of this analysis on both the client and customer. By looking at these two aspects it can often help us to decipher whether the activity is ethical or not

Pitt Ohio, a $700 million freight company decided to introduce a data analysis and measurement system of their historical data. The analysis involved using the historical data to predict and calculate numerous pieces of future information such as freight weight and driving distance. The results from this analysis allowed the company to predict their delivery times at a ninety nine percent accuracy rate. This has benefited the company as it reports revenue increases of an estimate $110,000 per year through receiving repeat orders and through the reduced the risk of losing unsatisfied customers. Yet, this analysis has also allowed the company to improve their efficiency for the benefit of customers. Customers can now receive more in depth information about their deliveries and also have a much more accurate estimate delivery time. This is a case of data measurement where the benefits are being reaped by both customers and clients and, in my opinion, it is not intrusive of the clients data.

A more unethical prediction model was devised by Andrew Pole for Target in the US. It analysed the products women bought at Target and based on this model predicted whether a women could be pregnant or not based on what she had been buying. For example, a 23 year old woman purchasing cocoa-butter lotion, a purse large enough to also be a diaper bag, zinc and magnesium supplements and a blue rug is said to have an 87% of being pregnant under this analysis. They used this analysis to single out high potential clients with advertisements and coupons for baby clothes, cribs etc. in order to encourage the expecting mothers to buy childcare supplies in Target.

Although I can appreciate that analysing customer data and using results to target people with certain products is a very effective tool for boosting business and profits, in some cases like this one, I strongly believe this analysis is crossing the line and is extremely intrusive.

This intrusive activity was highlighted when a father of a young girl went to his local target in Minneapolis to speak to the manager and complain about the coupons his daughter was receiving in the post. As a result of this the teenage expecting mother had to responded to her father's complaint explaining to him that in fact she was pregnant and the coupons were not misleading. Although I understand the coupons were not misleading as the father had once suspected, I find it horrific that a young girl would have to explain her pregnancy to her family based on post received that was based on results from data analysis and a pregnancy prediction model. You have to step back and ask yourself – is it ethical that a computer knows about the pregnancy of a young girl before her own father does?

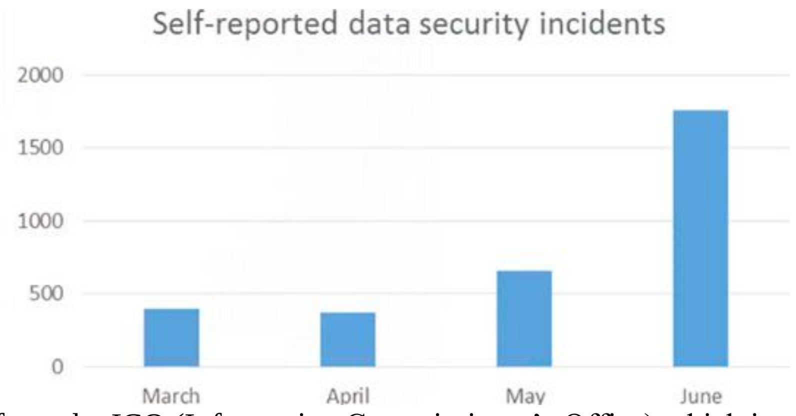
In both corporate examples above, personal information about both customers has been gathered and analysed. What is alarming is the difference in what was done with this analysis. Pitt Ohio used the data to help improve the efficiency and predictability of its delivery schedule. Conversely, Target predicted highly confidential medical information and used it to encourage customers to buy more at their store. Of course, in most cases, the motives behind data collection and analytics are noble, but unfortunately this is not always the case and what is most disappointing is that while unethical these practices are often legal.

Obviously this is something which is extremely difficult to manage and control as it really is a grey area but we need to find some way to improve it. In my opinion legislation is the first way to start combatting this problem. Without law enforcement I worry that data protection issues could spiral out of control. However, a step in the right direction was the introduction of GDPR in May 2018.

The letters ringing in peoples ears and flooding peoples emails in 2018 have been GDPR. The General Data Protection Regulation (GDPR) is a European Union regulation that governs consumer's private information. It came into full force in May 2018 and it has already changed how business all over the globe handle privacy. GDPR came into effect after a number of data protection scandals broke into the world news including those of massive global companies such as Facebook.

GDPR protects customers in a number of ways. To begin, it covers an extremely broad jurisdiction – it applies to all companies that process personal data of EU citizens. Data subjects now must give consent for their information to be analysed and used (hence the flooding of inbox's in 2018 with consent forms). A daunting prospect for firms is the strong penalties that can be enforced on firms in cases of data breach. These fines can be up to 20 million euros.

A topic of mixed opinion is the number of data security incidents have risen astronomically since the introduction of GDPR. Yet although some may see this rise as a negative, I feel one must view it as a positive. In my view, the new regulations are far more stringent and hence the any figures before May 25th 2018 are not representative of the number of firms fairly and securely using data. However, as organisations begin to adapt to this new law I hope we will see a drop off in the number of data breaches after the initial settling period.



Based on figures from the ICO (Information Commissioner's Office) which is an independent UK authority which was set up to uphold information rights in the public interest

<u>Overall Conclusion</u>

In conclusion, the software engineering process is a complicated one which brings with it many questions. With regards to the ways the software engineering process can be measured and assessed I believe striking a balance between the use of hard and soft data is the key to success. By analysing both interpersonal skills along with hard facts like lead times and code churn, one can get a more rounded picture of how effectively a software engineer is performing.

I feel the missing piece between the raw data and drawing conclusions is a computational platform which allows us to process data, and make it legible. As for these computational platforms that are available, the list is endless. It is about finding a platform that is best suited to the practices of a firm. Code Climate and Codebeat have a huge online presence. However, they are not short of competitors as there is an array of platforms and resources online.

When it comes to the algorithmic approaches for the software engineering process using machine learning algorithms is often the key to success and certainly a trend nowadays. After deciding which approach is appropriate (supervised/unsupervised/reinforced learning) a company can begin to pick the algorithms which best suit their needs.

From an ethical standpoint it is obvious that in some cases data analytics can majorly improve the efficiency of a company while benefiting both the company and customer. Yet, in other cases we see an absolute intrusion of the data subjects privacy which is only of benefit to the company. I hope that through increased legislation, these unethical activities will begin to be reduced.

Data analysis is not going anywhere any day soon, in fact, it is growing enormously every day. More and more analytics is becoming a force behind our day to day lives. Analysing and assessing the process in the coming years will become more important as the market becomes more competitive and efficiencies become more important. Also, with the introduction of stricter data privacy laws such as GDPR, companies will have to become more adaptive to ensure they comply with the new requirements. Despite all of, this no matter how much time and money we invest into assessing the process, all of these questions will never be fully answered. Therefore, human intuition and creativity are still vital contributors to high performance.

Bibliography

I consulted the following websites and books to build my report:

1. https://martechtoday.com/guide/gdpr-the-general-data-protection-regulation
2. http://www.analyticshero.com/2012/10/25/31-essential-quotes-on-analytics-and-data/
3. http://www.dtic.mil/dtic/tr/fulltext/u2/a131332.pdf
4. https://about.gitlab.com/2016/03/08/commits-do-not-equal-productivity/
5. https://insights.sei.cmu.edu/sei_blog/2014/09/agile-metrics-seven-categories.html
6. https://blog.gitprime.com/why-code-churn-matters/
7. https://docs.codeclimate.com/docs/trends#section-technical-debt
8. https://codeclimate.com
9. https://blog.drinkbird.com/code-churn
10. https://www.techopedia.com/definition/29893/analytics-as-a-service-aaas
11. https://www.forbes.com/sites/oracle/2014/09/26/the-road-to-analytics-as-a-service/#7e9255cf3622
12. https://martechtoday.com/guide/gdpr-the-general-data-protection-regulation
13. https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/
14. https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/
15. https://www.kdnuggets.com/2016/08/10-algorithms-machine-learning-engineers.html/2
16. https://skymind.ai/wiki/deep-reinforcement-learning
17. https://www.nytimes.com/2012/02/19/magazine/shoppinghabits.html?_r=1&hp=&pagewanted=all
18. https://www.cio.com/article/3221621/analytics/6-data-analytics-success-stories-an-inside-look.html
19. http://engineering.kapost.com/2015/08/you-can-and-should-measure-software-engineering-performance/
20. https://www.tutorialride.com/software-engineering/agile-process-in-software-engineering.htm