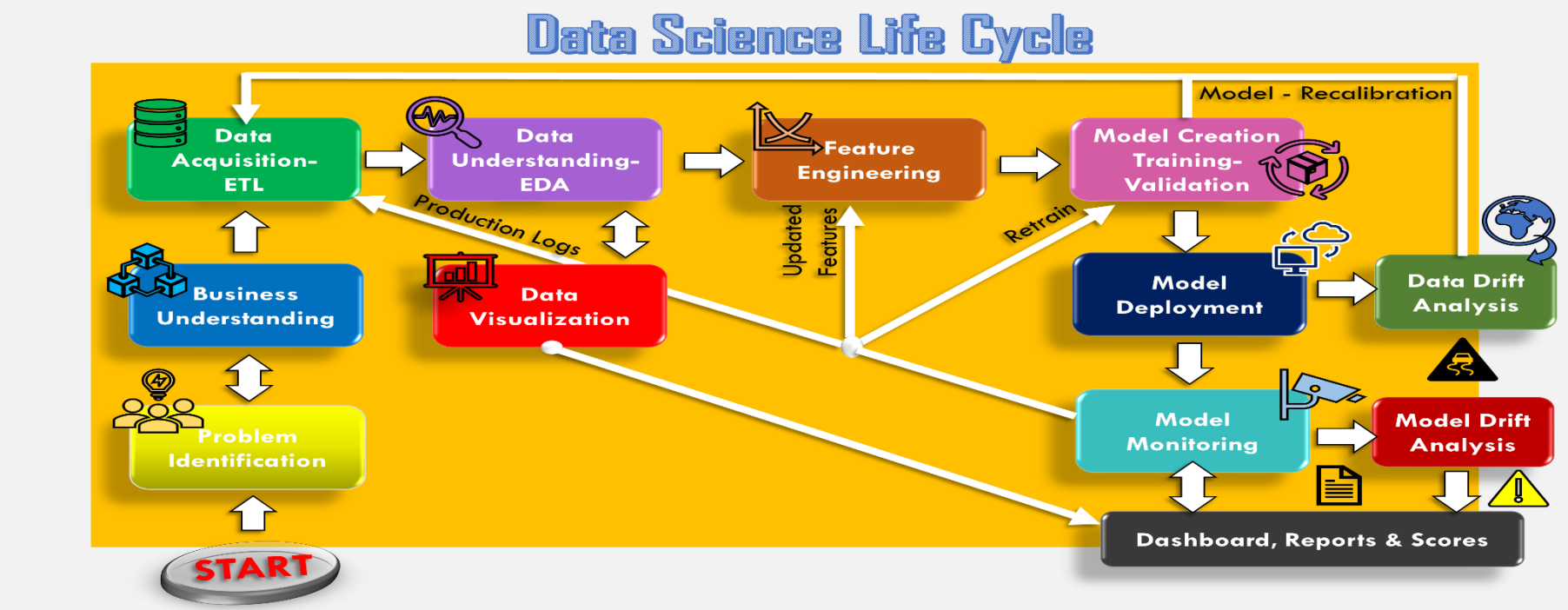
Complete Life Cycle of Data Science :

In order to make a Data Science life cycle successful, it is important to understand each section well and distinguish all the different parts. Specifically is very important to understand the difference between the ***Development stage*** versus the ***Deployment stage***, as they have different requirements that need to be satisfied as well the business Aspect.

Most Data Science projects have similar work-flow/ structure, that you can use to structure your projects. The lifecycle below outlines the major stages that a data science project typically goes through. It is never a linear process, though it is run iteratively multiple times to try to get to the best possible results, the one that can satisfy both the customer(s) and the Business.



Data Science life cycle (Image by Author)

The Horizontal line represents a typical machine learning lifecycle looks like starting from Data collection, to Feature engineering to Model creation: ***Model Development Stage***. The left-hand vertical line represents the initial stage of any kind of project: Problem identification and Business understanding, while the right-hand vertical line represents the whole aspects of the ***Model Deployment stage***. Now without further ado let's go and find out each stage.

Problem Identification:

Like any problem, the first thing to discuss is all business scenarios is Problem Identification, This is where we:

1. Clearly identify the root cause of a problem, by identifying the true, underlying problem.
2. Develop a detailed problem statement that includes the problem's effect on the targeted client /customer.

HOW DO YOU IDENTIFY THE PROBLEM?

We Identify the root cause of the problem *by collecting information and then talking with stakeholders and industry experts*. Followingly, *Merging existing research and information from your stakeholders* can offer some insight into the problem and possible high-level solution. Consider data sources that could help you more clearly define the problem. Starting by doing a **literature review**, and if necessary, surveys in the data science community.

Problem Identification Summary (Image by Author)

Business Understanding:

This where the Business problem is defined, understanding the business problem is key in coming up with a good model because it makes you understand the Business objective.

For instance, if you are credit card example you want to make sure I underwrite customers who could pay, or another one could be detecting fraudulent customers, so the intention can be to reduce risk and the same time increase revenue.

Now that we have a business understanding, we can define the success criteria of the ML project, the success criteria can be based on what we are currently doing. You need to see if the project is viable in longer-term and does it give you a business advantage to really to take it forward.

Now one thing very obvious, but still extremely important to denote is that the low barriers to entry internet marketplace where anyone has great information available have made the whole market extremely competitive in a business point of view. Every one of us is facing competition, therefore only by continually measuring and **tracking the right metrics and thinking about how to move the metrics to improve your performance we can increase our chances to come out on top.**

This is where we need to have the right metrics:

· **KPI (Key Performance Indicator)**

KPIs stands for **Key Performance Indicator:** Measure your performance against key business objectives. Businesses use this to measure their performance against objectives and the overall health of their business. It's crucial to agree upon the right set of KPIs for each business, and not to fall into the trap of chasing vanity metrics that look great but aren't meaningful.

· **SLA (Service Level Agreement)**

A **Service-Level Agreement (SLA)** defines the level of service expected by a customer from a supplier, laying out the metrics by which that service is measured. For instance, If you are creating a very complex pipeline and you have an SLA of 100 milliseconds to meet and you have to process like 2000 transactions per second. You have to make sure you will meet your SLA and KPI need, otherwise, the model is not gonna be to any benefit to you.

Misalignment in the metrics can have a negative impact on deal pricing, quality of service delivery, customer experience overall Business objective and reputation. The best possible way will be to monitor these KPI(s) and SLA(s) using, for instance, KPI Dashboard to monitor key results.

Again a very important distinction to remember is the difference between the algorithmic objective with the business objective. Most of the times they can be confused, **THE BUSINESS OBJECTIVE MUST BE SIMPLE AND CLEAR IN BUSINESS TERMS.**

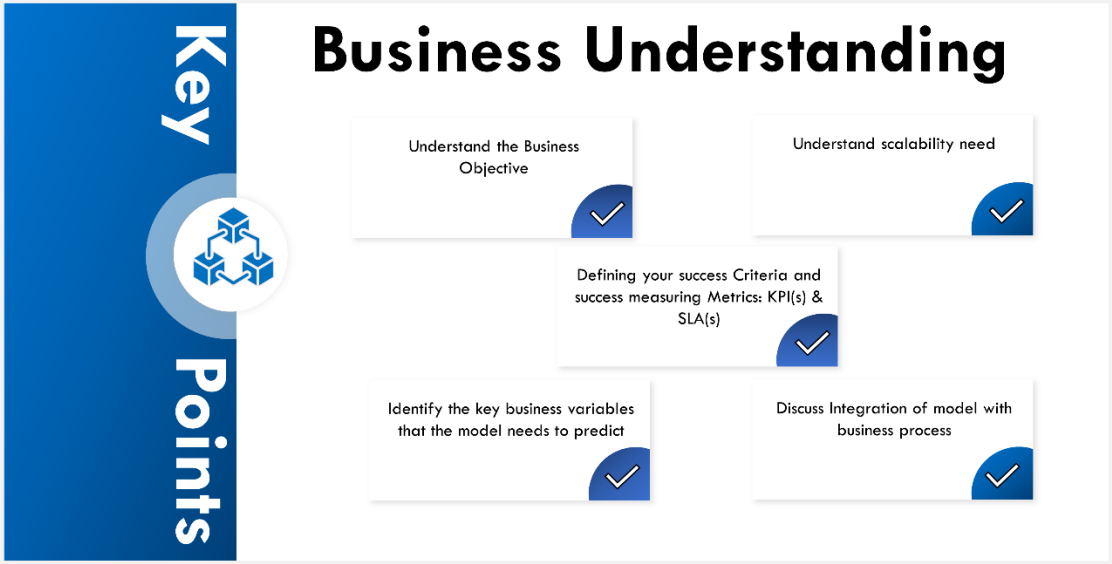
It shouldn't be:

" Reducing the mean squared error of the model ecc…"

Instead, it should be:

"Increase my revenue by 5%"

"Attracting more customers and reducing the number of customers churning by 10%"



Business Understanding Summary (Image by Author)

Data Acquisition

Once we understand the business and its objectives. We need to understand the data and all the different needs. First, we go and collect the data. The latter can be already ordered or centralized or we may have to go to the individual source system and collect the data that is required for our model. Determining whether there is sufficient data to move forward with the next modelling steps. Again, this process is often iterative. This is the Data Acquisition

phase. this is generally done through an ETL pipeline, ETL stands for Extract, Transform and Load.

ETL

Here this the step where the data is pulled processed and loaded into a data warehouse, this is generally done through an ETL pipeline, ETL stands for Extract, Transform and Load.

ETL is a 3 steps process:

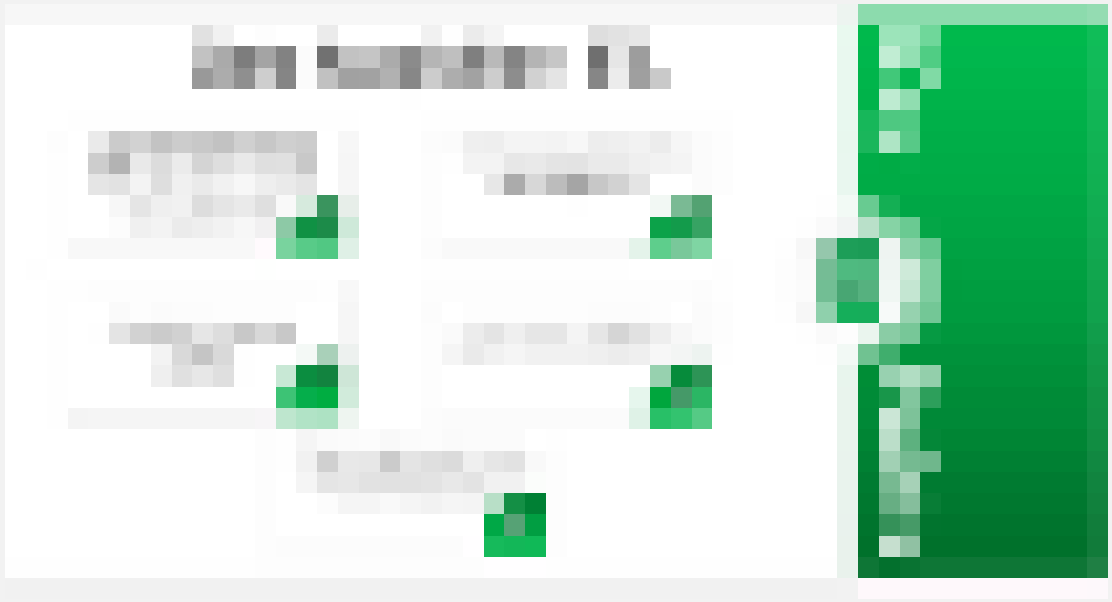
- ***Extracting*** data from single or multiple Data Sources.
- ***Transforming*** data as per business logic. Transformation is in itself a two steps process- data cleansing and data manipulation.
- ***Loading*** the previously transformed data into the target data source or data warehouse.

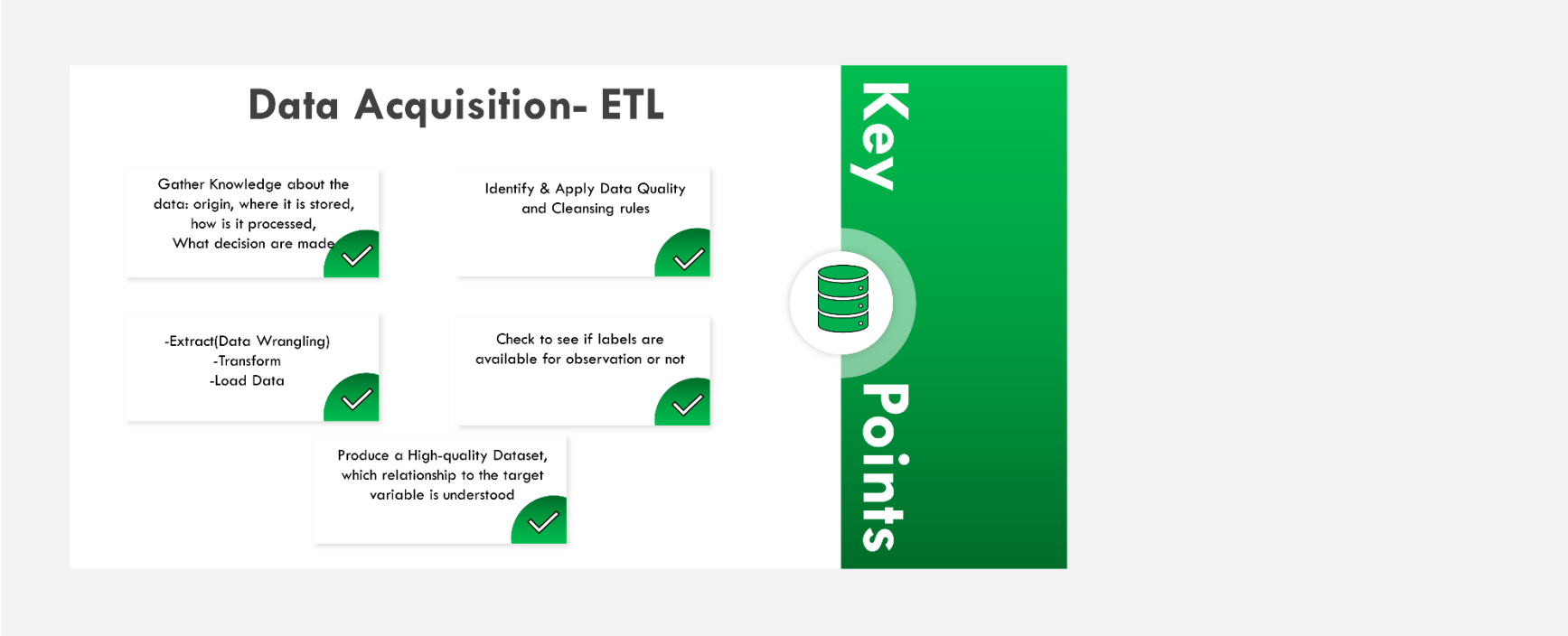
In the Transformation part of the data, we might have to deal with different issues, One for instance is the **Imputation of missing values**. Frequently data contains missing values

or null values which lead to lower the potential of the model. So we try to impute the missing values.

1. For **continuous values**, we fill in the null values using the mean, mode or the median depending on the need.
2. For **categorical values**, we use the most frequently occurred categorical value.

Moreover, the data is almost never in the right format, therefore a pre-processing step is always a must, so identifying these pre-processing rules for instance: Date-Time data.





Data Acquisition and ETL summary, (Image by Author)

EDA (EXPLORATORY DATA ANALYSIS)

The Exploratory data analysis part is the part where we understand everything about the data, the patterns etc... This is done by adopting **Statistics and Data Visualization**. Here we study information such as:

- How are the different variables related to one another and what their distributions look like?
- What is the correlation between the variables, Looking for evidence of how well connected the data is to the target variable?

STATISTICS

1. We use ***Descriptive Statistics*** to get a bigger picture of the data, this describes the data based on its properties.

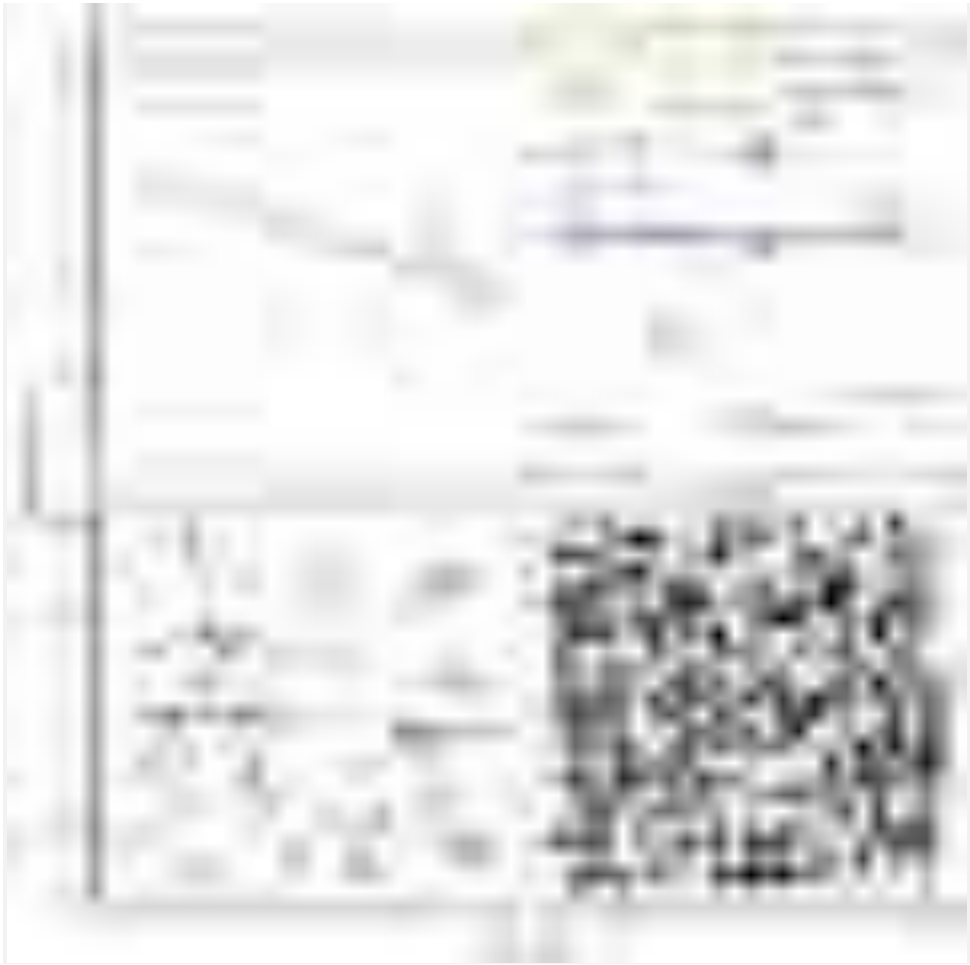
The four major types of descriptive statistics are:

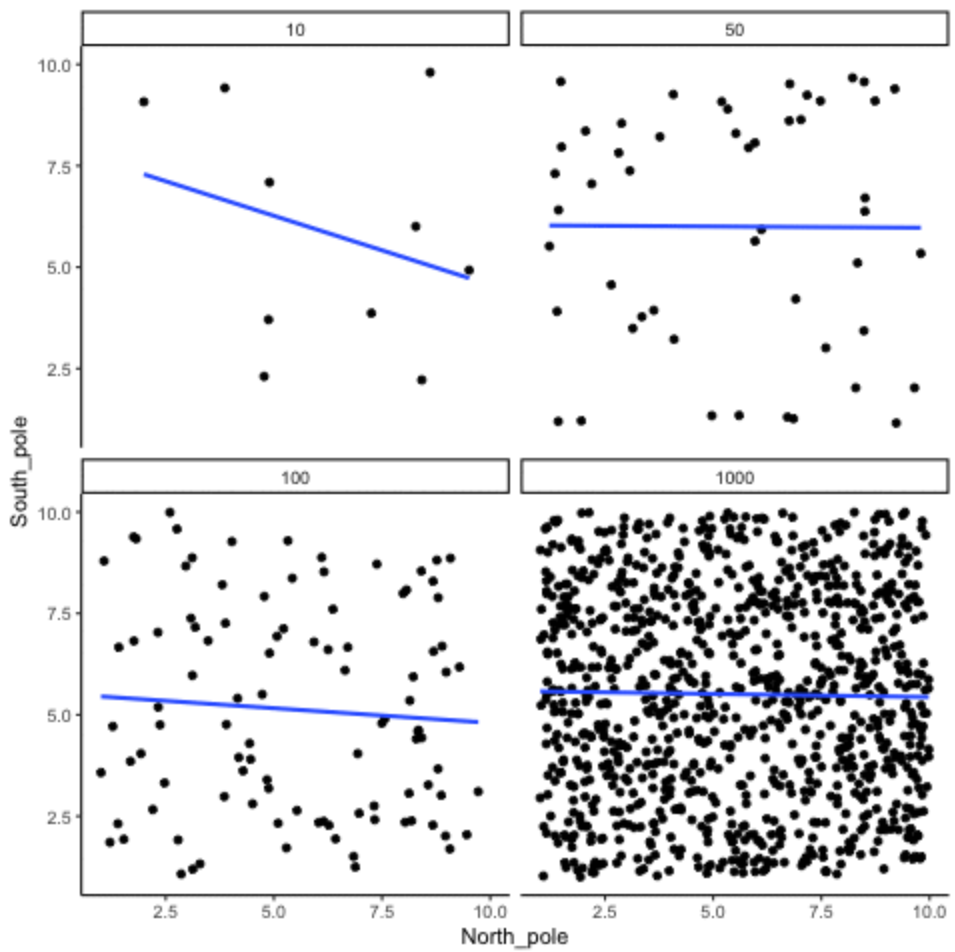
- *Measures of Frequency*
- *Measures of Central Tendency*
- *Measures of Dispersion or Variation*
- *Measures of Position*

2. We use ***Inferential Statistics*** to validate distributions across datasets and model outputs, the most used methods are :

- *The estimation of the parameter(s)*
- *Testing of statistical hypotheses*.

3. Finally, we adopt ***Univariate and Multivariate Analysis*** to identify correlation and trends between the dependent and independent feature(s):





Source: https://crumplab.github.io/statistics/gifs.html

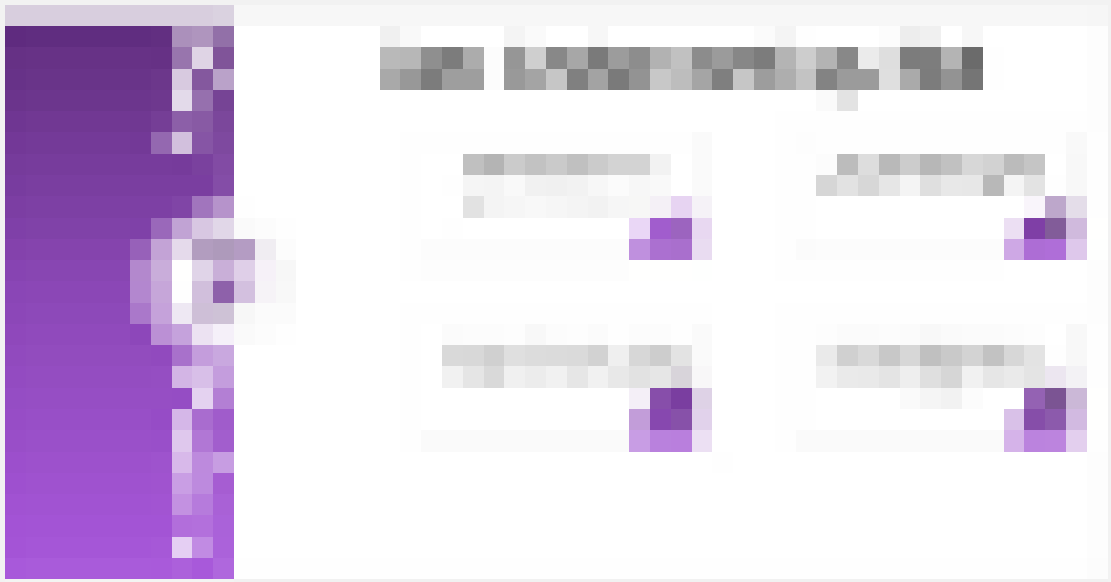
DATA VISUALISATION

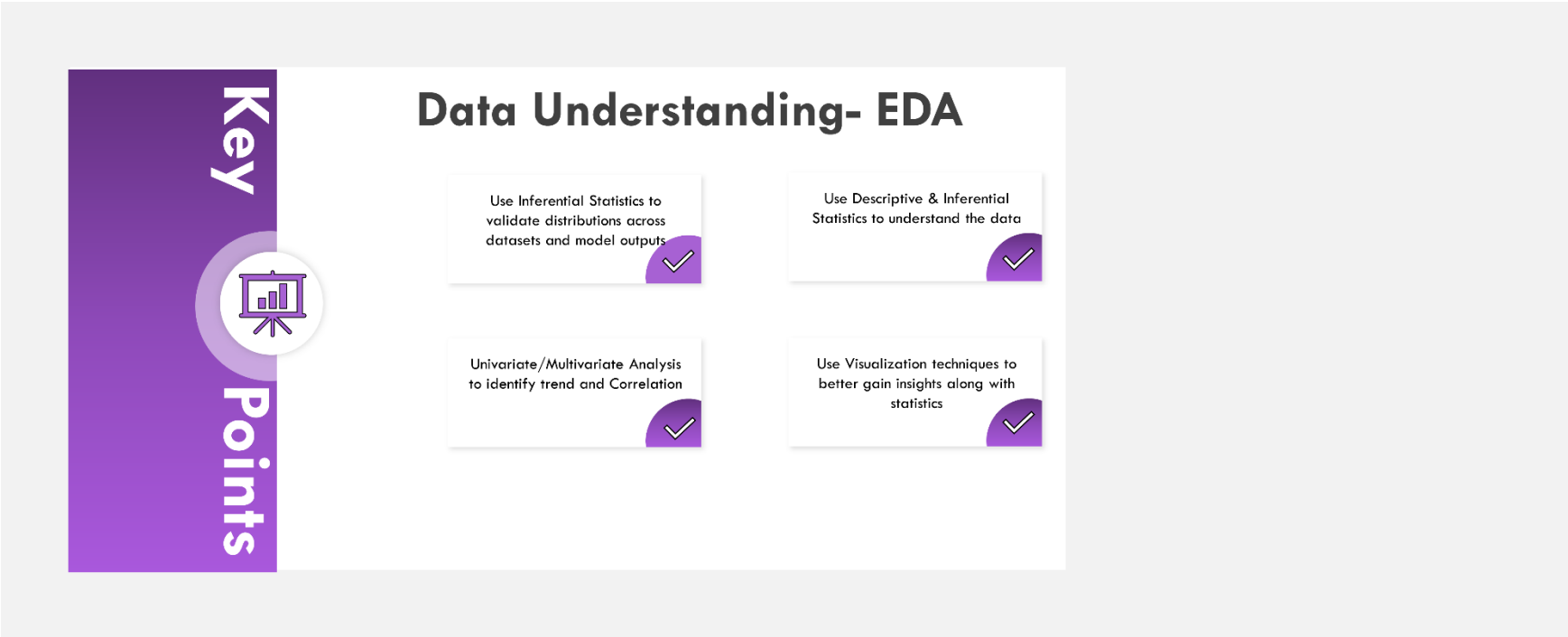
Source: [Bootslabs]

Data visualisation plays a big role in a data science life cycle, it is used in every single part from data understanding to deployment, to communicating business insight to stakeholders.

Data visualisation is able to generate visible insights in all terms, just by plotting a simple pattern on a graph, this makes the patterns and trends identification so much easier rather than just looking at thousands of rows on a spreadsheet and just using statistics.

Even when information is pulled from statistics, insights from data without visualization are more difficult to communicate without good and appropriate visualization, an example of different types of visualization figure above.





EDA Summary,(Image by Author)

Feature Engineering:

Feature engineering is the process of using domain knowledge to extract **features** from raw data via data mining techniques. These **features** can be used to strengthen the performance of machine learning models. **Feature engineering** can be considered as applied machine learning itself.

> “Machine learning consists of only 20 to 30 % of an entire Data Science life cycle”

The most meaningful techniques of feature engineering are used to transform data into a form where a model can understand better and dealing with data and pattern anomalies:

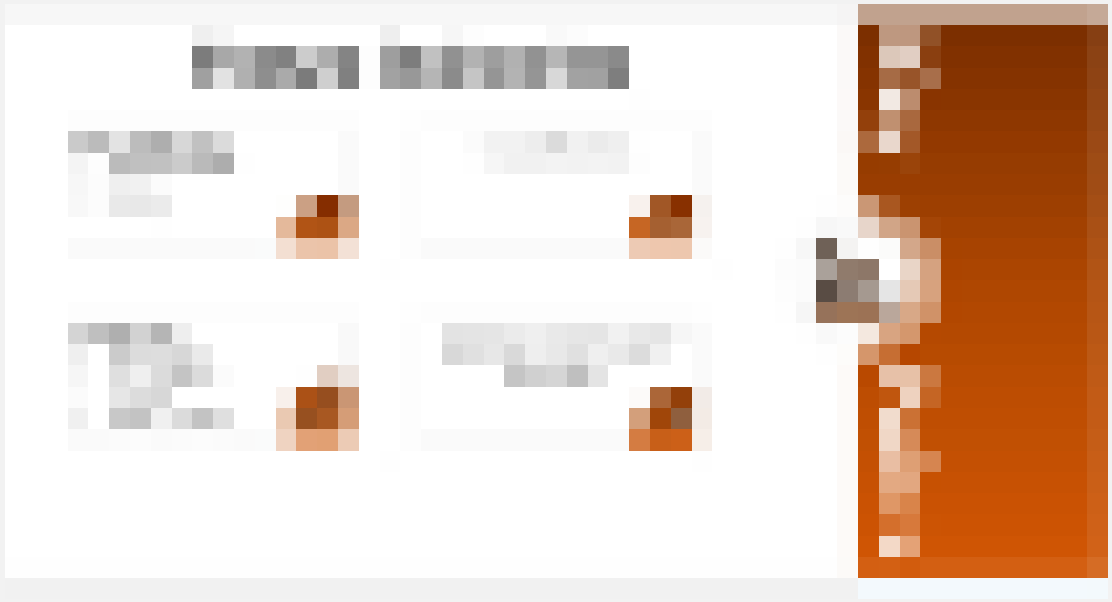
- **Data Transformation techniques:** *Log transform, Square-root transformation, Categorical encoding, Power functions and Scaling*.
- **Dealing with:** S*kewed data, Bias mitigation, Binning, Outlier detection* the latter identifies data points or observations that deviate from a dataset’s normal behaviour. Outliers data can indicate critical incidents, such as a technical glitch, or potential hazards therefore this need to be treated accordingly otherwise it could mislead our final model(s).

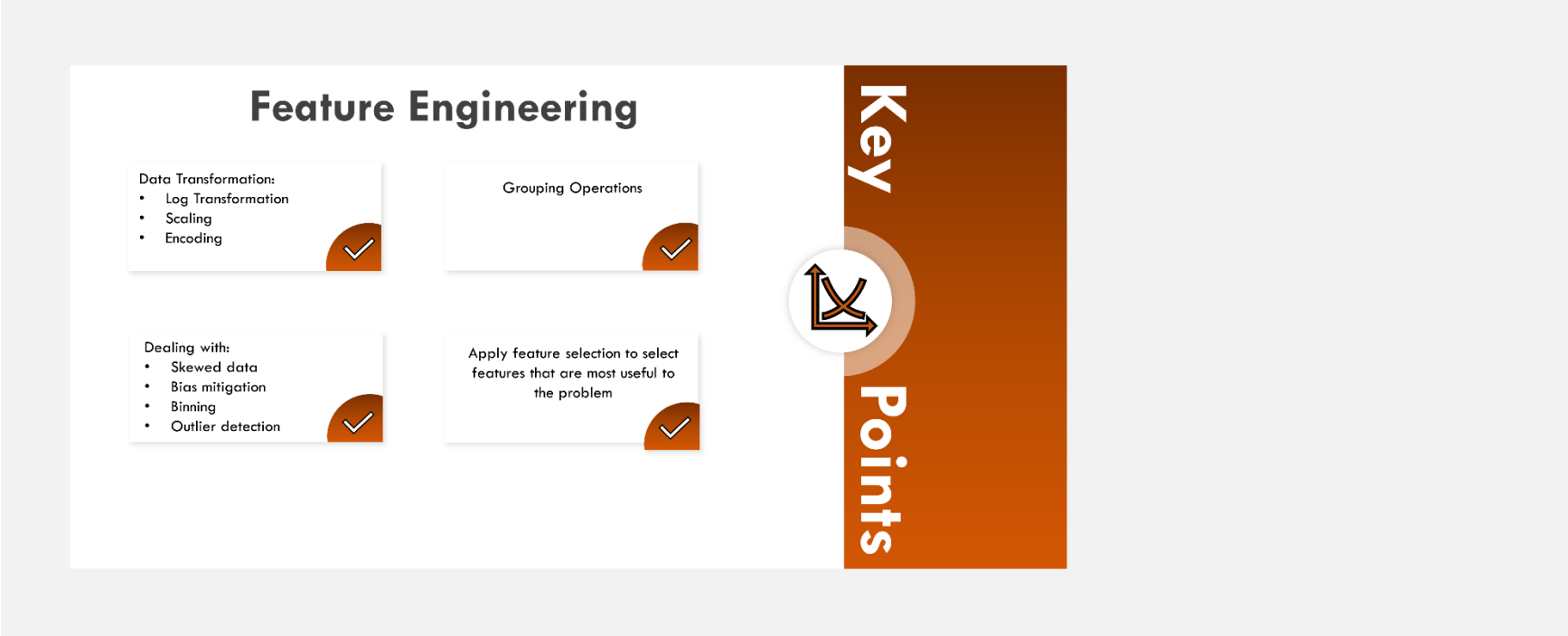
After transforming the data to the right format and dealing with potential data hazard, most of the time especially with a high dimensional dataset, we end up with many features, we can not feed all the features to the machine learning model, that is not how it works, that would overfit the model hugely. Instead, we have to choose the right number of features.

This is partially done in the prior step, in the Data Analysis or EDA: when studying the correlation of the variables, however, there is an appropriate way of doing it and this is called **Feature Selection.**

These are three main methods:

- ***Filter based method:*** *w*e specify some metric and based on that filter features, for instance: C*orrelation, Chi-square, ANOVA*.
- ***Wrapper-based method:*** we consider the selection of a set of features as a search problem, for instance: *Recursive Feature Elimination, Forward/Backward Selection*.
- ***Embedded method:*** we use algorithms that have built-in feature selection methods, for instance, *LASSO(L1) and RIDGE(L2) regression*.

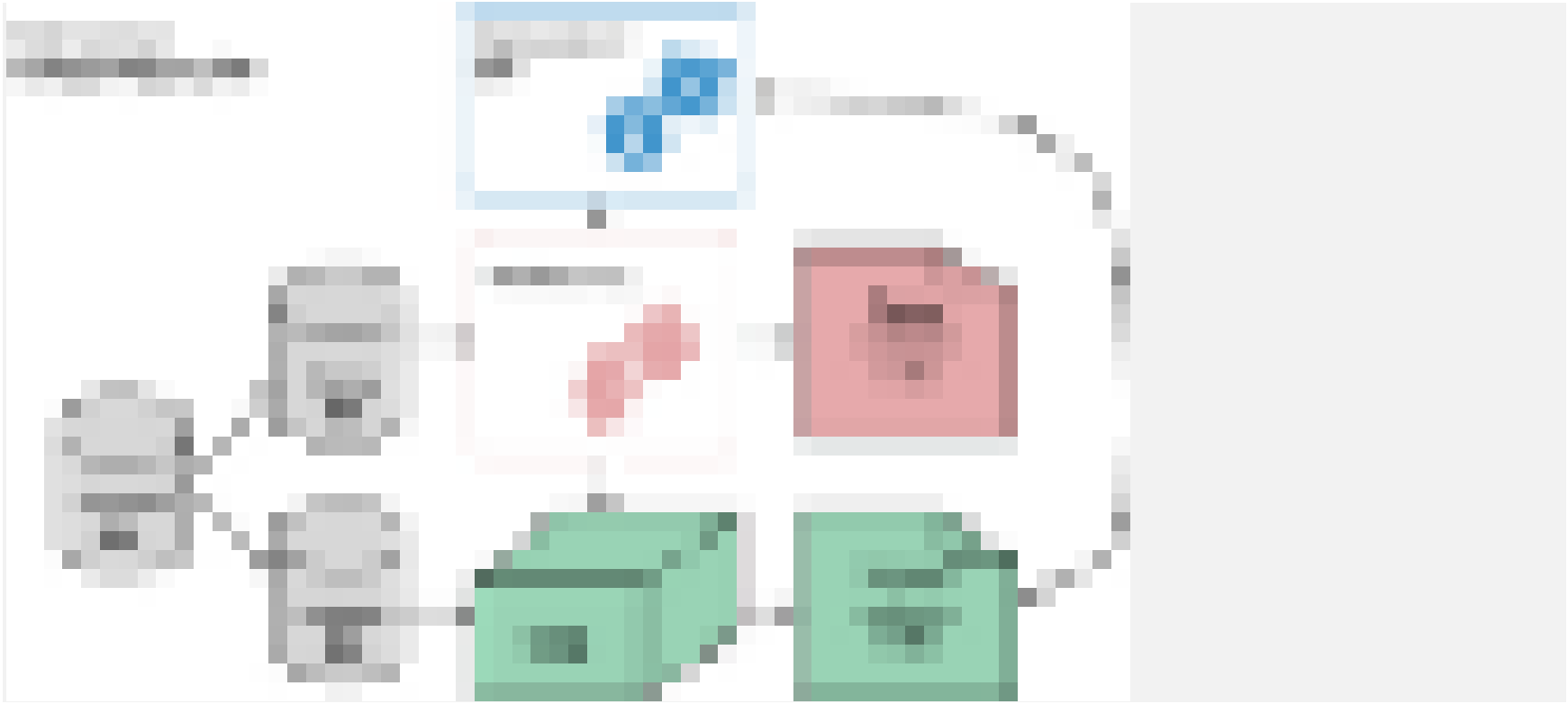


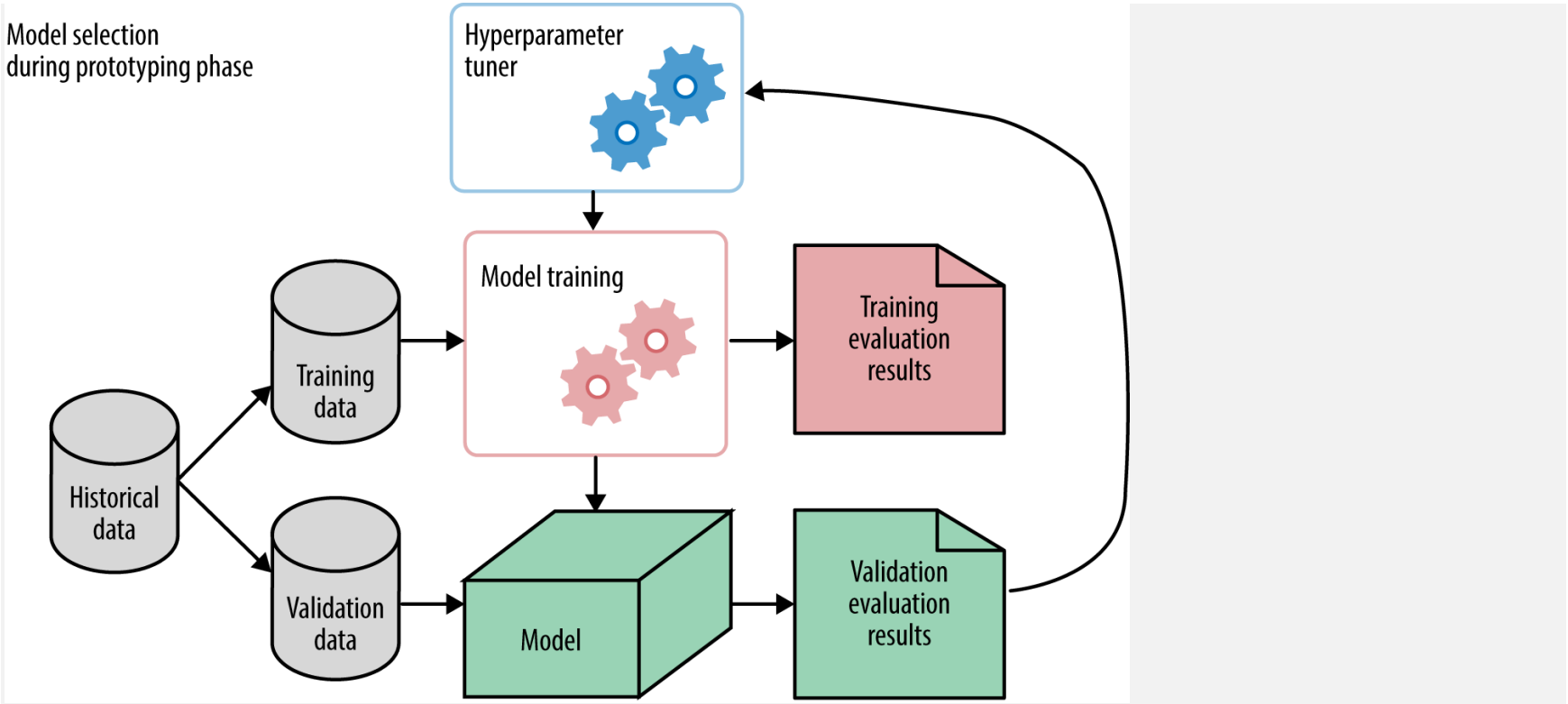


Feature Engineering Summary, (Image by Author)

Model Training

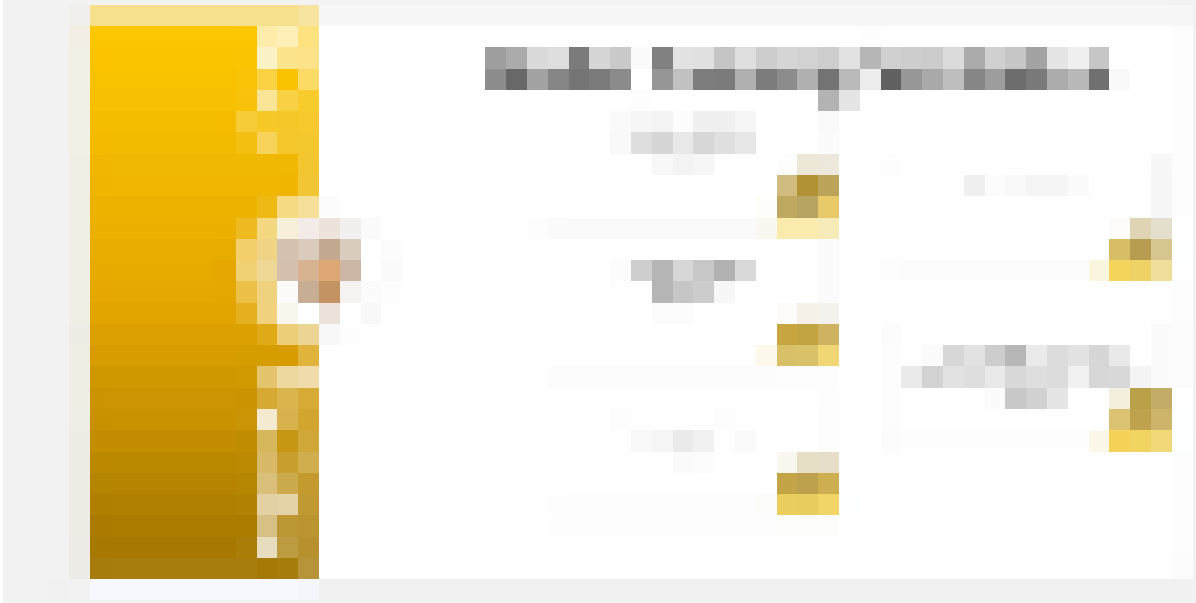
Once the features are ready to be modelled First, we create a **Baseline model**, then we do the Model training, then we keep increasing the complexity of the model and increasing testing with various algorithms and see how this particular dataset respond to this algorithm then you do **Hyperparameter Tuning** to set the right hyperparameter of the model, then we apply techniques to prevent overfitting such as **Cross-validation**.

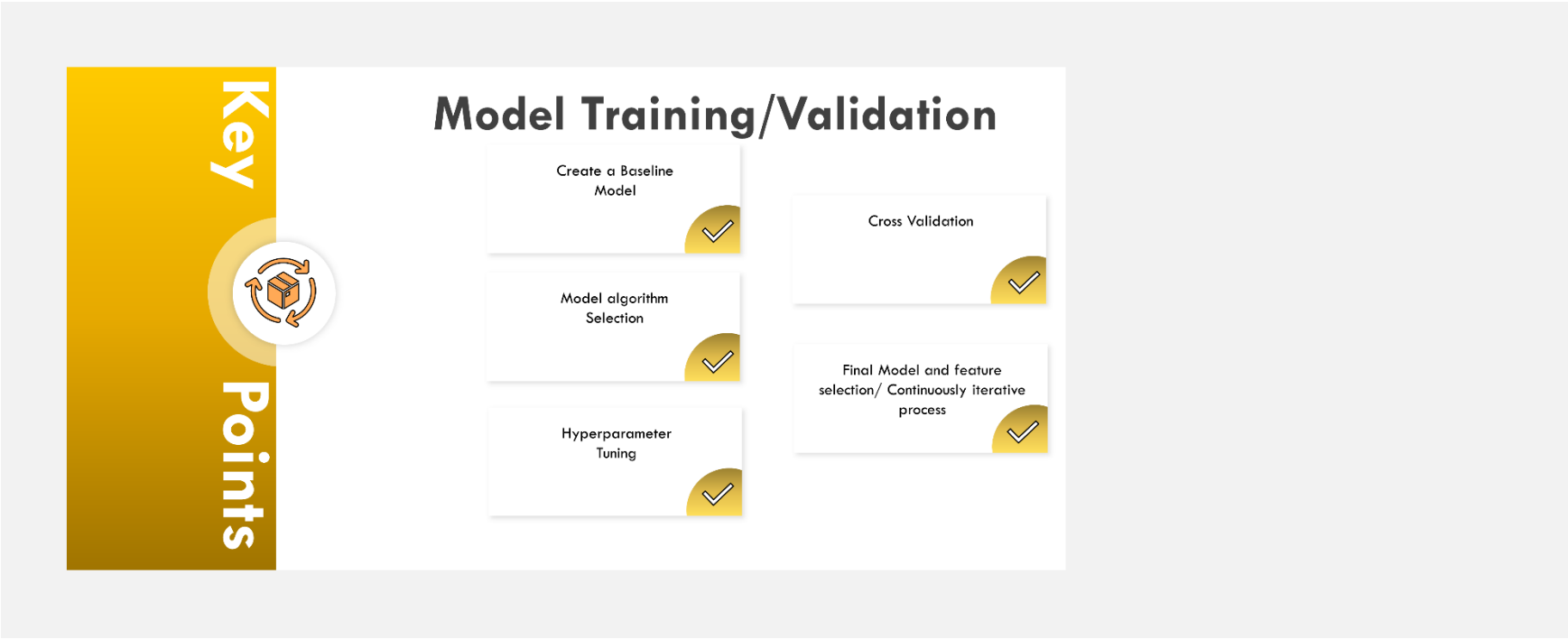




Source: Oreilly by Alice Zheng

We need to evaluate the performance of the Trained model, on a dataset, this happens on a test set because the model might overfit, not generalising well so we have to make sure we remove any bias in our model, that is why this part is very important.





Model Training Summary (Image by Author)

Model Deployment

Once we have trained the model, model deployment is not straight forward, we need to think of it way ahead.

Model Development is hard but Model Deployment is even Harder — Srivatsan Srinivasan

Prior to deployment, we have to make sure we test the following:

- Whether the model deployed matches the expectations of the original model. By using a test input set identified during the development stage which produces validated results.
- Make sure the data is clean so it can be processed by downstream machine learning components so we can enforce expectations and transactions between the process thanks to the consistency and quality of the data.
- Whether the model being deployed is robust enough to take in a different range of inputs by testing extremes of those outputs.

By doing constant scheduled checks and running tests prior to deploying the model to production, we minimize and prevent the risk of the model from crashing and, in effect, affecting its downstream consumers.

DIFFERENT WAYS TO PUT THE MODEL INTO PRODUCTION

Choosing how to deploy predictive models into production is quite a complex problem, there are different way to handle the lifecycle management of the predictive models, different formats and ways to stores them, multiple ways to deploy them and very vast technical landscape to pick from.

There are generally different ways to both train and server models into production:

- **TRAIN**: One-off, Batch and Real-time
- **SERVE:** Batch, Real-time (Database Trigger, Pub/Sub, web-service, inApp)

ONE-OFF TRAINING

Machine learning models don't necessarily need to be continuously trained in order to be productionized, it can be just trained and pushed into production until its performance deteriorates enough, then they are refreshed and retrained.

This type of model deployment is preferred by Data-Scientists who do prototyping and doing machine learning, the typical model formats are:

- Pickle
- ONNX
- PMML
- POJO and MOJO

BATCH TRAINING

Batch training permits to have a constantly updated version of the model based on the latest train.

REAL-TIME TRAINING

When deploying this type of models in real-time, there needs to be enough support and monitoring as the model can be sensitive to new data and noise as well as the model

performance need to be constantly monitored, this type of training is possible with 'Online machine learning models' specifically:

- K-means (through mini-batch)
- Linear and Logistic Regression (through Stochastic Gradient Descent)
- Naive Bayes classifier

BATCH vs REAL-TIME PREDICTION

In terms of prediction/ serve, when looking at whether to set up a batch or real-time prediction, it is very important to understand the key differences.

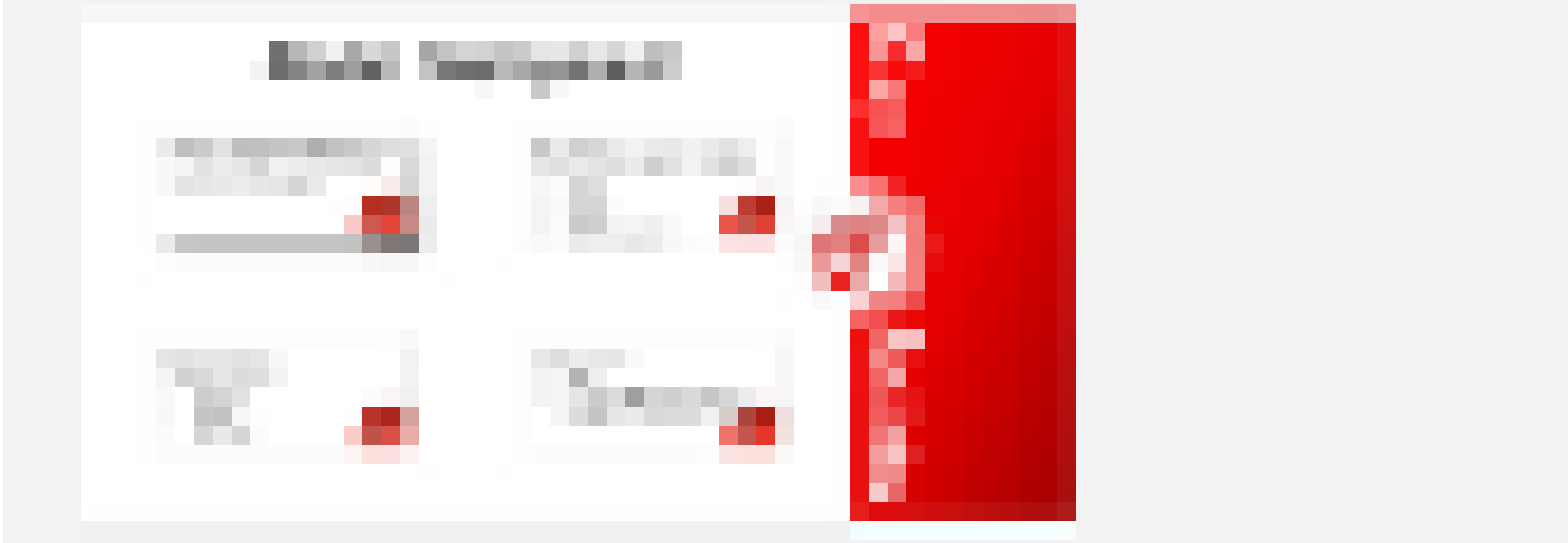
Example of Batch prediction:

The batch jobs are typically generated on some recurring schedule (e.g. hourly, daily). These predictions are then stored in a database and can be made available to developers or end-users. Batch inference may sometimes take advantage of big data technologies such as Spark to generate predictions. This allows data scientists and machine learning

engineers to take advantage of scalable compute resources to generate many predictions at once.

Example of Real-time prediction or Online Inference:

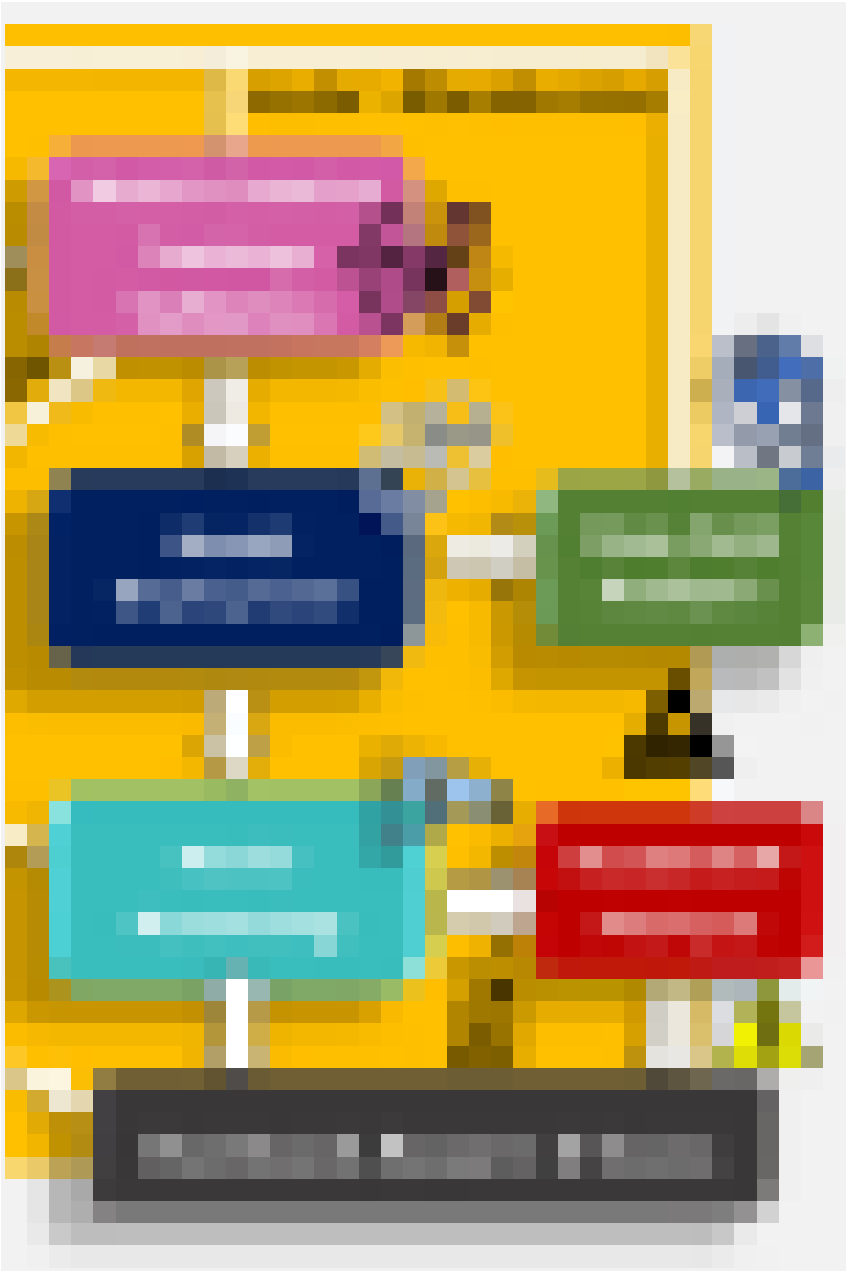
UberEats estimated-time-to-delivery is generated whenever a user orders food through the service. It would not be possible to generate a batch of these estimates and then serve them to users. Imagine not knowing how long it will take to receive your order until after *the food arrives. Other examples of applications that can benefit from online inference are augmented reality, virtual reality, human-computer interfaces, self-driving cars, and any consumer-facing apps that allow users to query models in real-time.*

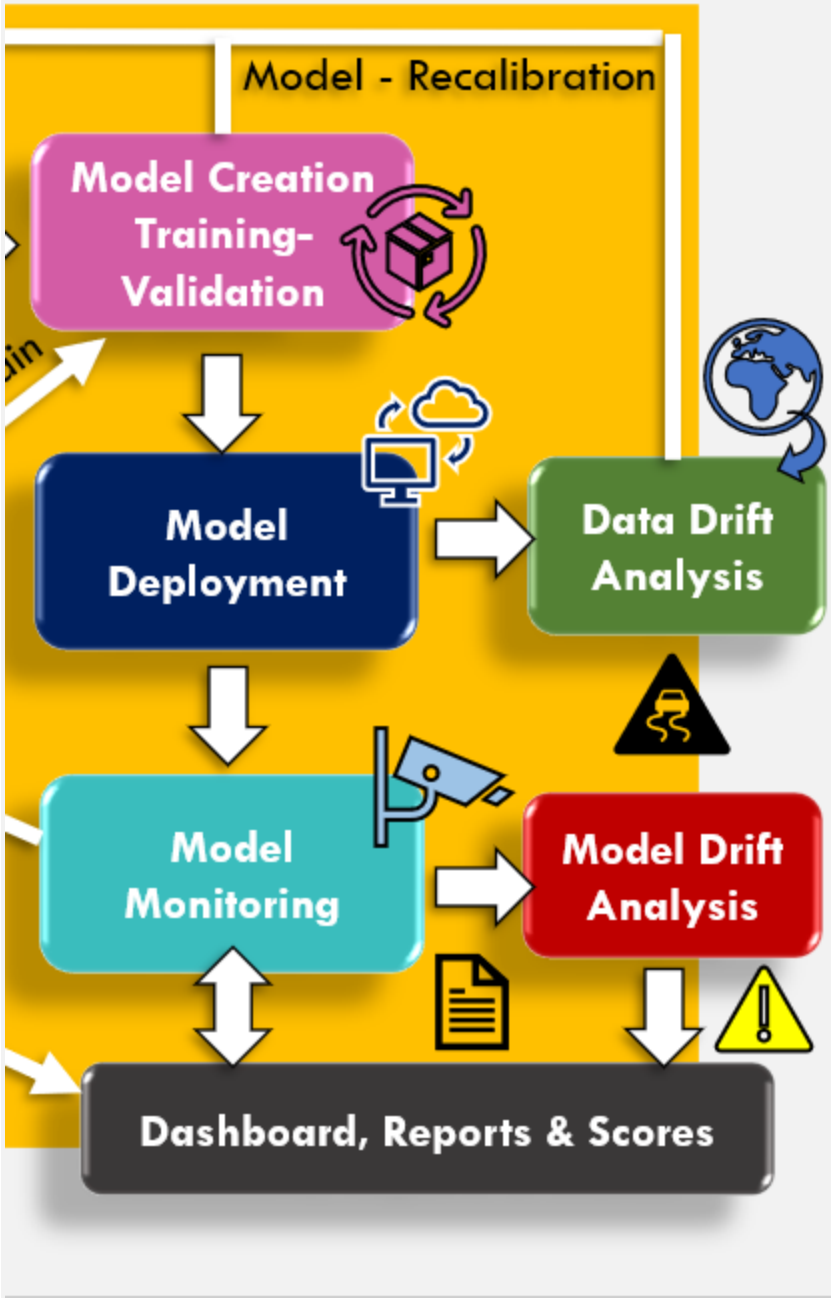


Model Deployment summary (Image by Author)

Model Monitoring

Once the model is deployed in some extreme cases the model can be non-performant, it may be just throwing out garbage score, this will affect the business objective and so on, therefore we need to continuously monitor the model, often this part is ignored by beginners.




Model - Recalibration
Model Creation
Training-
Validation
Model
Deployment
Data Drift
Analysis
Model
Monitoring
Model Drift
Analysis
Dashboard, Reports & Scores

(Image by Author)

Here comes in to play two important components:

Data Drift Analysis: we compare the incoming data distribution that is coming from the real world with the one that is scored, if we notice some difference between those, we need to select a threshold. However, if the difference is huge, the incoming data is drifted. In this case, we need to go back and analyse the difference.

Model Drift Analysis: we compare the difference between the score of the real-world model distribution against the scored model distribution, so here this is fed into a system which continuously reports and give output to the business user. Again The Business would not understand accuracy, recall, precision as previously stated in the business understanding section so the best metrics here are the KPI(s) which is normally found in the Dashboard: as visualization and reports, it would say for instance how much we customers we acquired, how much we made in dollars and how this mean in revenue streams and so on.

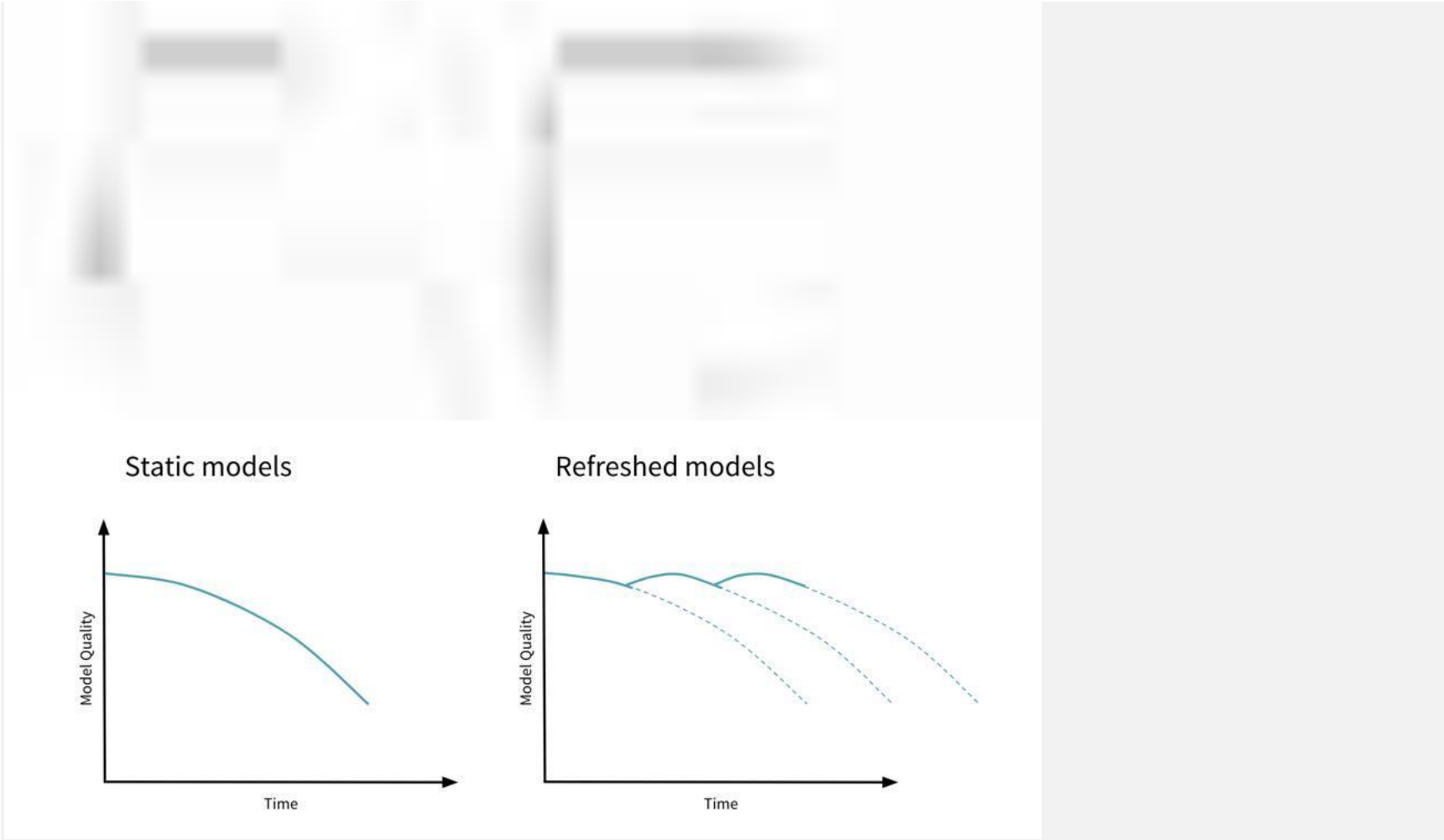
In terms of Data Drift, the causes could be :

- Upstream process changes, such as a sensor being replaced that changes the units of measurement, a broken sensor always reading 0.
- A Natural drift in the data, such as mean temperature changing with the seasons.
- Changes in the relation between features.

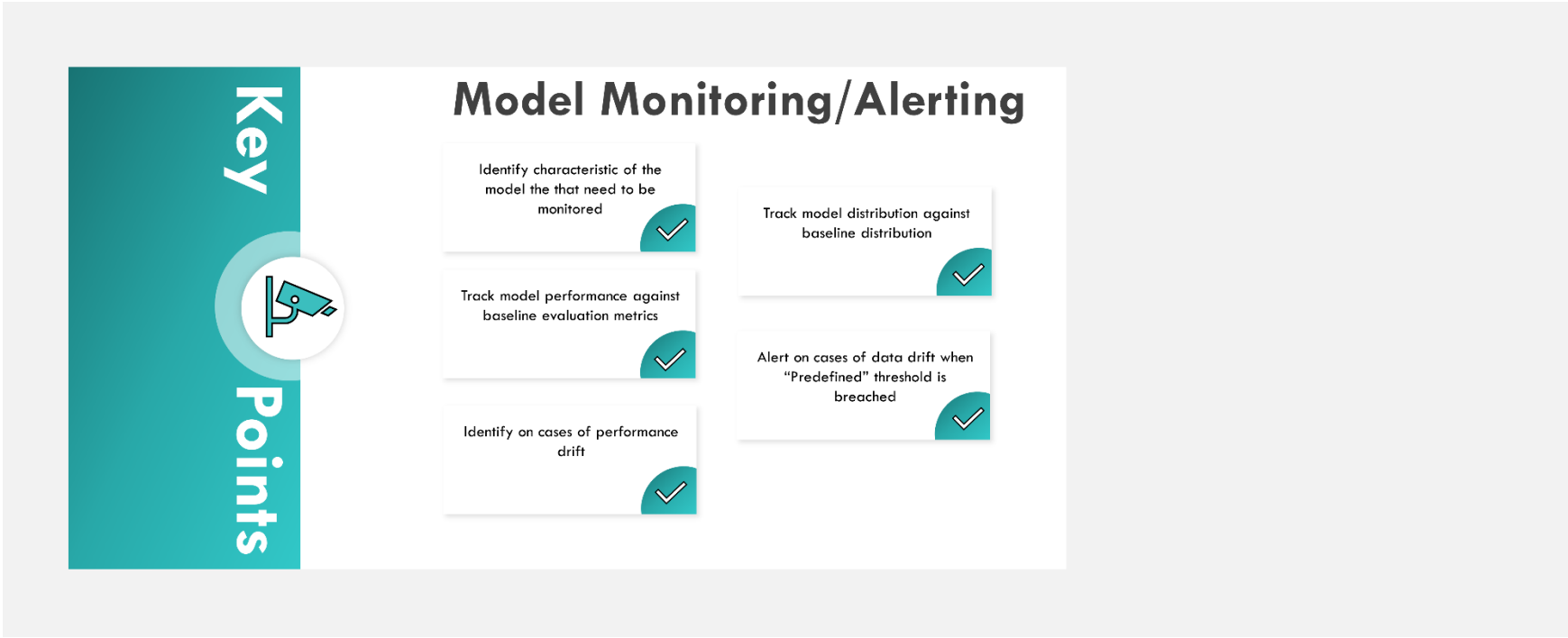
How to intervene?

- Clean up erroneous labels
- Retrain / Calibrate Model
- Detect and correct upstream data problems early

In cases where there is concept or data drift and your machine learning model(s) become stale as shown in the graph below on the left, we can see well on the right by refreshing the model over time we can prevent Model Staleness over time.



Source: [Databricks]



Model Monitoring/Alerting Summary (Image by Author)

Conclusion

This is a suggestion for the flow of Data Science Projects and the overall organization structure and its' interactions. Obviously, there are many variations on this cycle- for the sake of complexity, I did not go in-depth with some aspects of Model building and Evaluation because largely those are discussed in a frequent manner in all Data science and ML Platforms, so I preferred to stick with the other parts. To wrap up:

When adopting make sure you pick the best for your problem, understanding specific use cases and choosing the right metrics starting from the business perspective: KPIs and SLAs, this will help come to the right approach from development to deploying ML models to production. Machine learning is not the solution for everything, if you could solve a problem in a simpler manner and not adopting Ml, you should opt for that, the easier the solution the better. Hope you find this post informative! Thank you for reading!

What's your thought? Anything I missed or I could improve on? Please feel free to comment below. I welcome any feedback.

REFERENCES

Overview of Different Approaches to Deploying Machine Learning Models in Production - KDnuggets
There are different approaches to putting models into productions, with benefits that can vary dependent on the…
www.kdnuggets.com

What is a KPI? Definition, Best-Practices, and Examples
Measure your performance against key business objectives. A Key Performance Indicator is a measurable value that…
www.klipfolio.com

Models: From the Lab to the Factory - Silicon Valley Data Science
In our industry, much focus is placed on developing analytical models to answer key business questions and predict…

www.svds.com

Batch Inference vs Online Inference - ML in Production

You've spent the last few weeks training a new machine learning model. After working with the product team to define…

mlinproduction.com