



# Is deployment the elephant in the room?

**John McConnell**

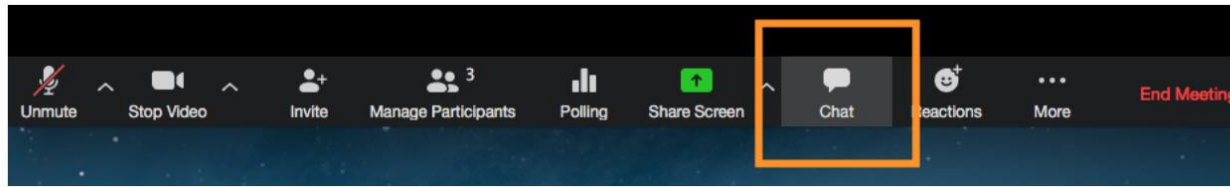
**January 2025**

[www.sv-europe.com](http://www.sv-europe.com)

A SELECT INTERNATIONAL COMPANY

# FAQ's

- Is this session being recorded? Yes
- Can I get a copy of the slides? Yes, we'll email links to download materials after the session has ended.
- Can we arrange a re-run for colleagues? Yes, just ask us.
- How can I ask questions? All lines are muted so please use the chat panel
  - If we run out of time we will follow up with you.



# Is there a problem?

---

- **WHAT OTHERS ARE SAYING**

“43% of Data Scientists say that 80%, or more, of their built models fail to deploy”

- Eric Siegel – [KD Nuggets January 2024](#)

According to [Gartner](#) analyst Nick Heudecker, over 85% of data science projects fail.

- Ryohei Fujimaki, [DataNami 2020](#)

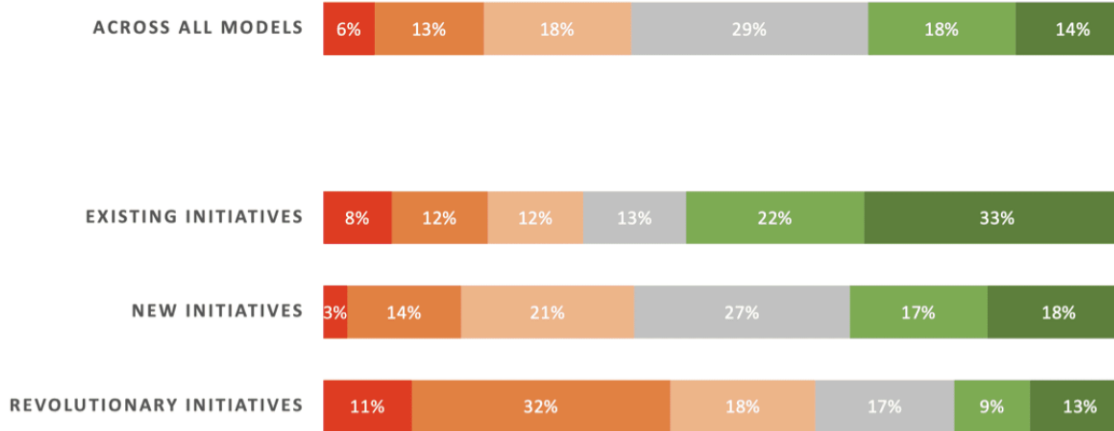
- **OUR OWN EXPERIENCE**

We don't always succeed but have learned from our experiences

We see more use of platforms and pipelines and more joined-up teamwork across functions

## PROPORTION OF MODELS THAT ARE DEPLOYED

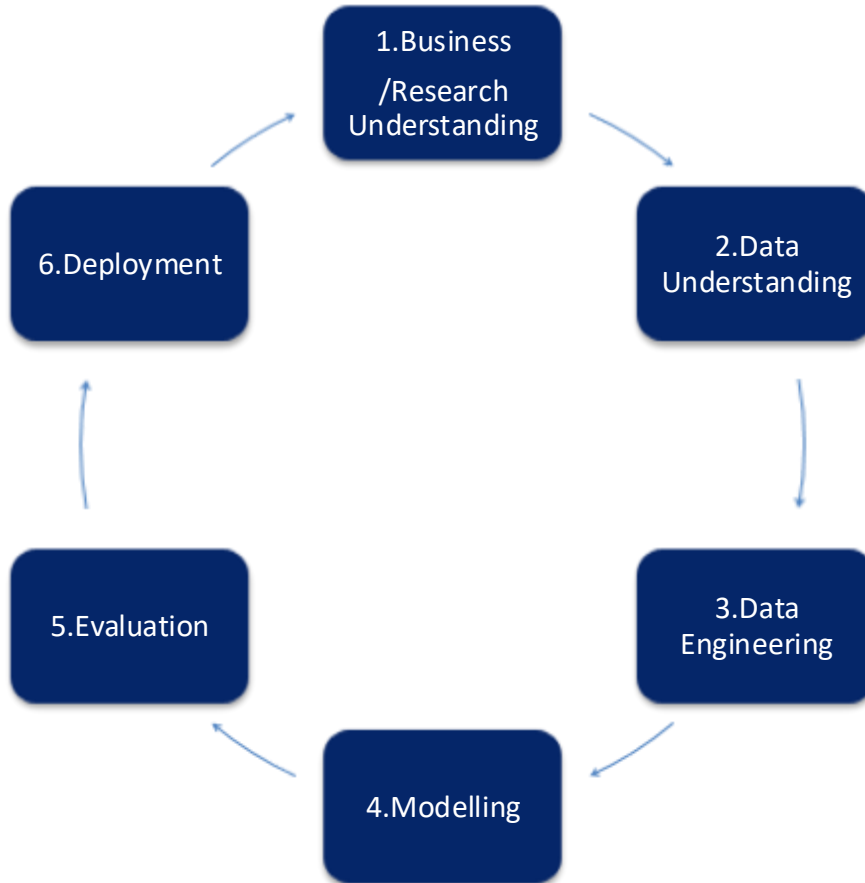
■ 0% ■ 1% -20% ■ 21% -40% ■ 41% - 60% ■ 61% -80% ■ 81% - 100%



### Key:

- Existing initiatives:** Models developed to update/refresh an existing model that's already been successfully deployed
- New initiatives:** Models developed to enhance an existing process for which no model was already deployed
- Revolutionary initiatives:** Models developed to enable a new process or capability

# The CRISP DM/DS process model

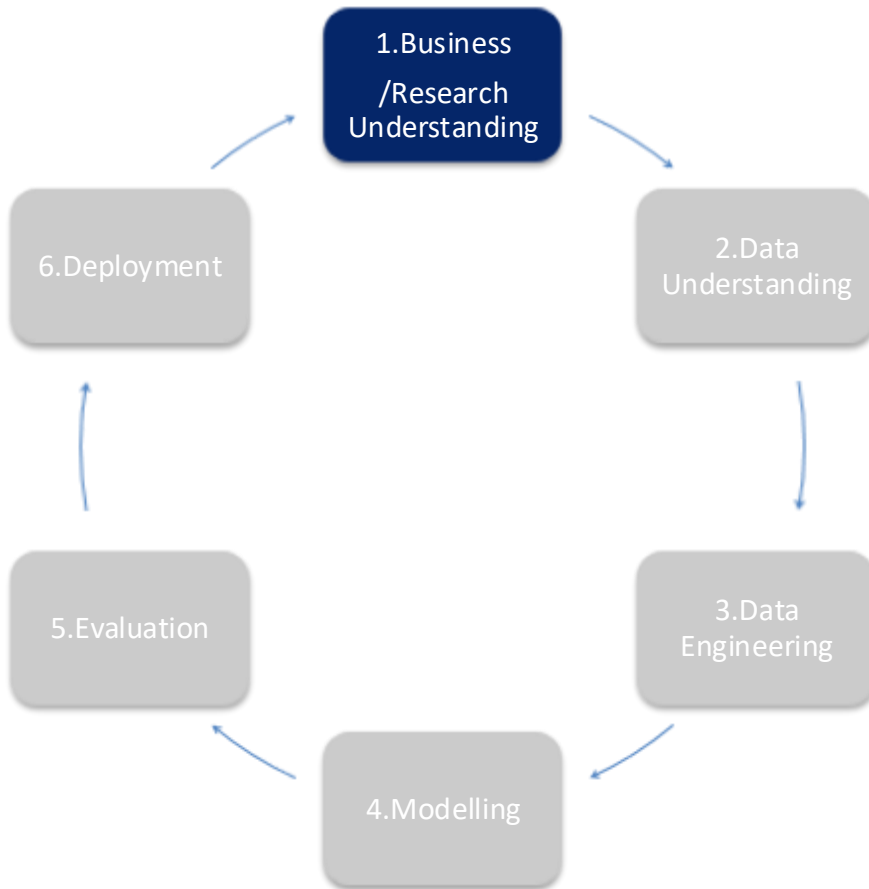


The Cross Industry Standard Process for Data Mining and Data Science.

For New Initiatives we look at the whole cycle

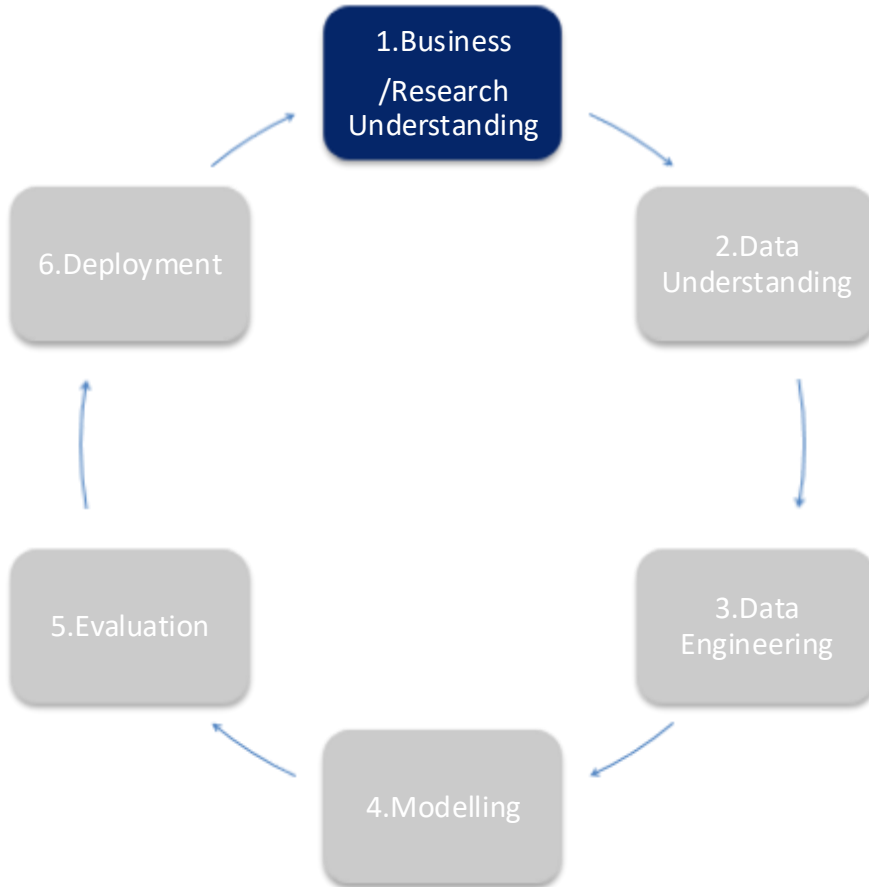
At the outset **Deployment** might seem somewhat far off ... something we can think about if/when we have arrived at a successful model

# 1. Business/Research Understanding



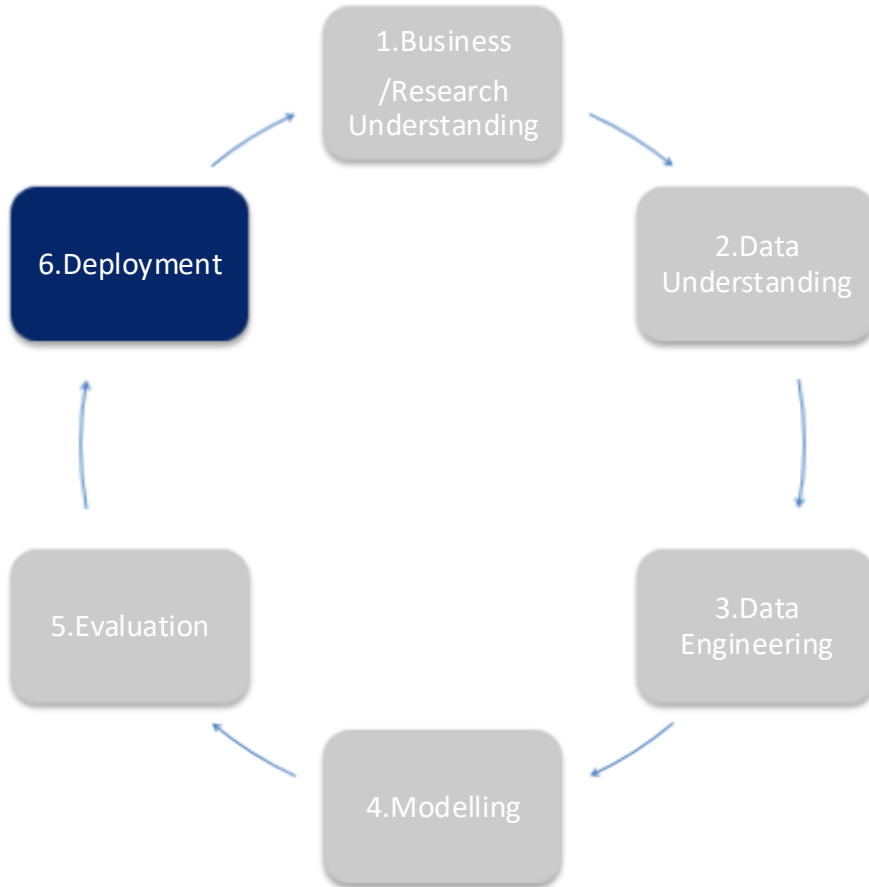
- We start with business/research conversations.
- Agree the objectives
- Understand the context
- Set success criteria
- Be sure to talk about “Deployment”

# 1. Agreeing Success Criteria



- An important step in the project where all stakeholders should align.
- For example, in a customer retention scenario, let's say we can only operationally intervene with 2,000 customer at a time
- So we might agree that we want our model to accurately identify 70% of customers who are likely to leave within the top 2,000 most "at risk" (of leaving)
  - If our churn rate is 20% that is likely to be a strong model

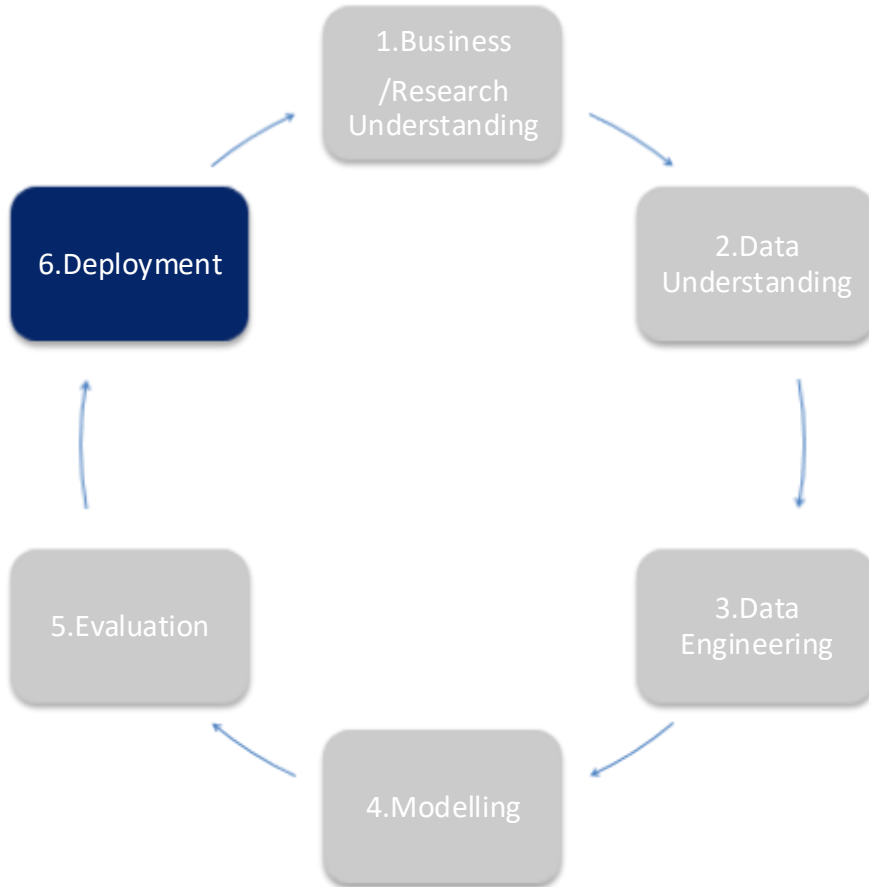
# 6. Deployment



- The whole point of the exercise
- Can be about deploying, often strategic, insights that come from simpler (often statistical) models and statistical analyses
- More often about deploying models (and often the data engineering that feeds them)

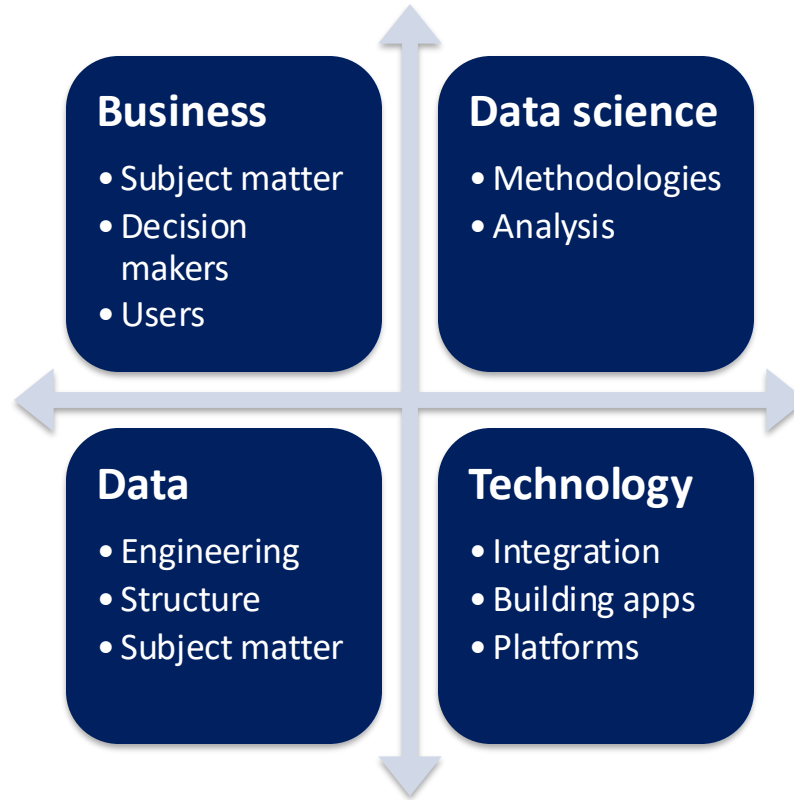


# 6. Deployment



- Increasingly needs to be Real-Time
- Frequently needs to be scheduled
  - Typically processing batches
- Often integrated into operational systems
- Behind apps including
  - What-if? simulators
  - Decision Management tools

# People (/ Roles)



# **BLOCKERS TO DEPLOYMENT SUCCESS**

# The Data

- We don't have enough (when we assess in Data Understanding)
- The quality is too low
- It is too difficult to access, clean or prepare
  - The “effort to insight” ratio is too high
- What was accessible to us when we built our model is not available when we deploy



# The Model(s)

- The model does not meet our accuracy target
- OR
- It is essentially flawed
    - This could be data related, or issues in the data prep
  - We may not realise it is flawed until it is deployed and it does not deliver the results we expected



# Stakeholder alignment (people, politics)

- May block us during the Evaluation
- Often happens as/after we deploy
- Decision making stakeholders may not “buy” the model
  - Especially when we start a new initiative they often want to know how the model is working
- End user stakeholders may not know how to use the model
- They may not want the model
  - It can be seen as a threat
  - They may not “buy it”
  - The use of the model may be perceived to be adding to their workload





# Technical alignment

- This most often hits right at the point of deployment
- Simply put, we can't move our model into production for a technical reason e.g.
  - The data sources we need for the model are not reachable or reachable in time
  - The target platform cannot run the model (or we don't have one)
  - IT governance e.g. concerns about open source



# Monitoring



- If we did our **Evaluation** job properly then the deployed model should correspond to what we saw in evaluation
  - Other factors may intervene
- Ongoing evaluation (“monitoring”) still needs to happen if models are to be used over time



# Monitoring



- “Model drift”, where deployed models become less accurate over time is expected
- When our monitored model accuracy falls below an agreed threshold we need to “refresh” the model
- This process should not take as long as the original model build (initiation)
- If refreshing the model in this route does not improve the accuracy sufficiently, we may need to go back into the data steps

# DEPLOYMENT OPTIONS

**Option 1 – Recode the model into something else**

# Simple models can be (hand) coded

```
Call:
lm(formula = CONSUME ~ PRICE + INC + TEMP + PRICEINCI, data = datavar)
```

```
Residuals:
    Min       1Q   Median       3Q      Max
-0.0575279 -0.0163589 -0.0008483  0.0168662  0.0718922
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.1570203   0.2324673   0.675   0.5058
PRICE       -0.1636906   0.7438870  -0.220   0.8277
INC          0.0012301   0.0012133   1.014   0.3208
TEMP         0.0028231   0.0004171   6.769 5.31e-07 ***
PRICEINCI   -0.2786003   0.1344397  -2.072   0.0491 *
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

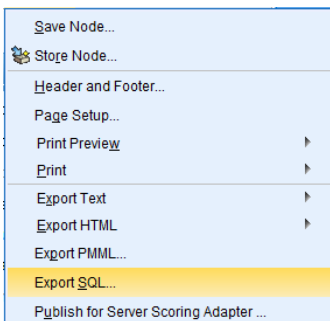
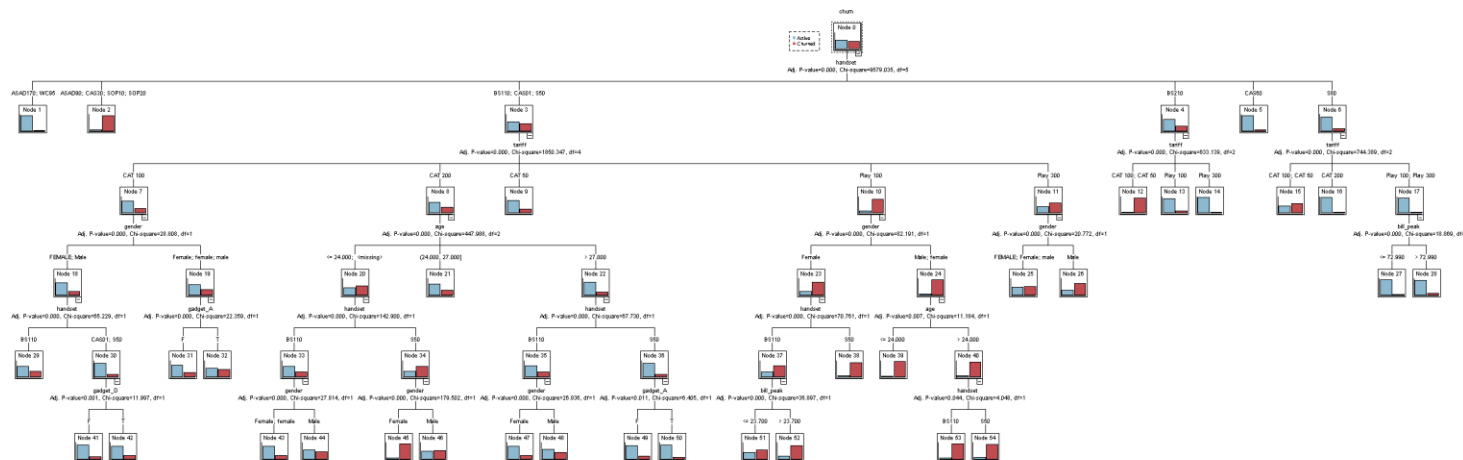
```
Residual standard error: 0.03094 on 24 degrees of freedom
Multiple R-squared:  0.7411, Adjusted R-squared:  0.698
F-statistic: 17.18 on 4 and 24 DF,  p-value: 8.968e-07
```

- Linear Regression models are a good example
- They fit easily into Excel formulae or into other code e.g. we've sometimes deployed models in SAP this way

**Consume = 0.1570203 -0.1636906\*PRICE + 0.0012301\*INC + 0.0028231\*TEMP -0.278600\*PRICEINCI**

**Option 2 – Export the model into a code format that can be plugged into other software**

# Exporting SQL is one option (sometimes)



- Some tools support this. E.g. Python and R support it for some model types
- Caution around the **differences in SQL** between databases e.g. Oracle and IBM/DB2
- Some tools provide **target-specific SQL** which can sometimes be published into the database

# PMML is the standard export/import format



[www.dmg.org](http://www.dmg.org)

**Predictive  
Model  
Markup  
Language**

## PMML 4.4.1 - General Structure

PMML uses XML to represent mining models. The structure of the models is described by an XML Schema. One or more mining models can be contained in a PMML document. A PMML document is an XML document with a root element of type PMML. The general structure of a PMML document is:

```
<?xml version="1.0"?>
<PMML version="4.4"
  xmlns="https://www.dmg.org/PMML-4_4"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <Header copyright="Example.com"/>
  <DataDictionary> ... </DataDictionary>

  ... a model ...

</PMML>
```

# Standards-based, Open-source Middleware for Predictive Analytics Applications

Convert your fitted **Scikit-Learn**, **R** or **Apache Spark** models and pipelines into the standardized **Predictive Model Markup Language** (PMML) representation, and make quick and dependable predictions in your **Java/JVM** application.

**Standardization enables automation**, which in turn enables higher efficiency and higher quality business processes.

GET STARTED NOW!

<https://openscoring.io/>



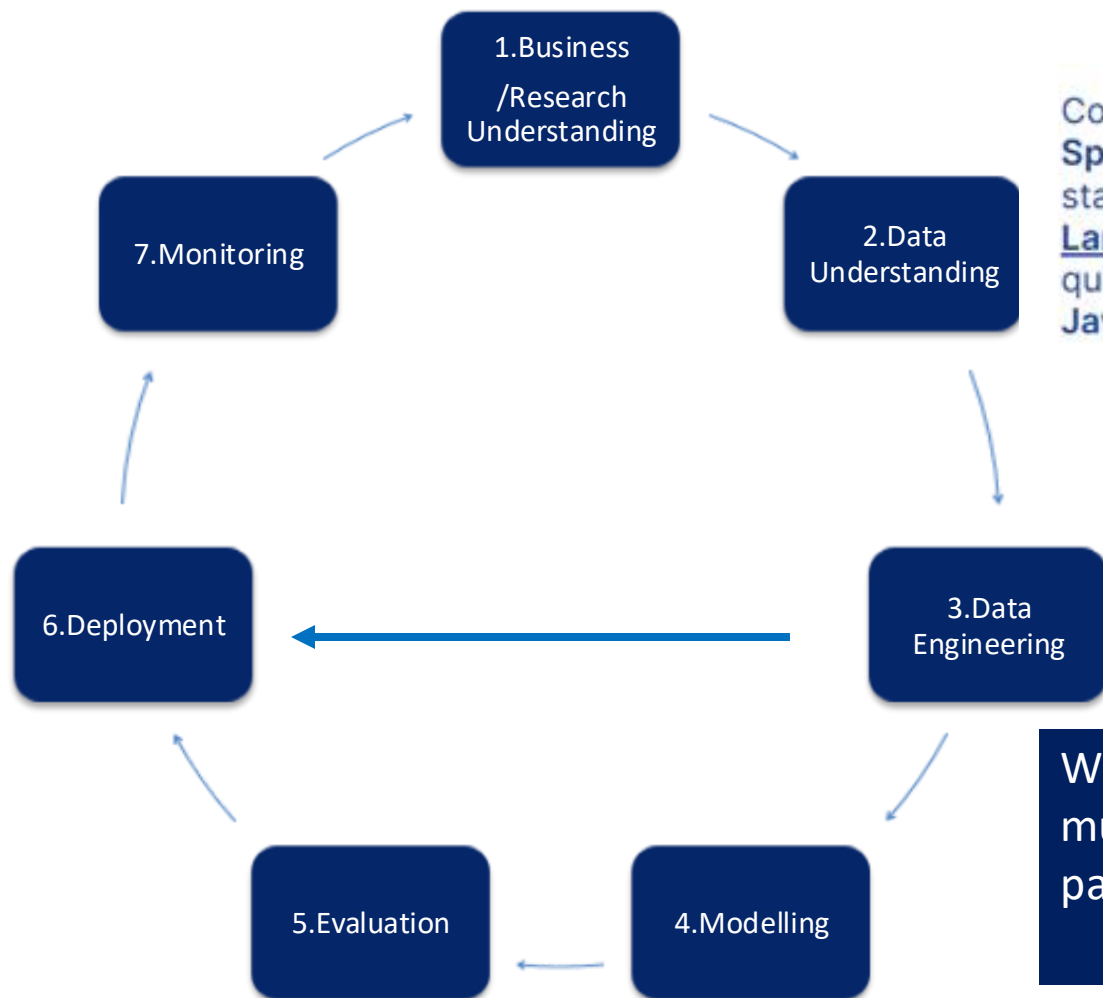
Villu Russmann and his team have developed tools to make PMML work from the most commonly used advanced Python and Spark algorithms/models



# A Use Case

- In a recent project, we worked to develop a model to forecast operational customer satisfaction for a UK bank
- We tested multiple algorithms, including **ARIMAS** and various **Regressions**
- **XGBoost** was the most accurate
- The bank needed to deploy the model in a forecasting (What-if?) simulator in Excel
- In the end, we were not able to install any additional open-source engines ... for technical reasons
- Hence, the currently deployed model uses a combination of **Factor Analysis and Linear Regression**
  - – The accuracy was still acceptable, but we had to compromise it
- The OpenScoring team is working on a prototype to deploy XGBoost models into MS Excel as "JavaScript add-ins."

# Data preparation is also part of a Deployment pipeline



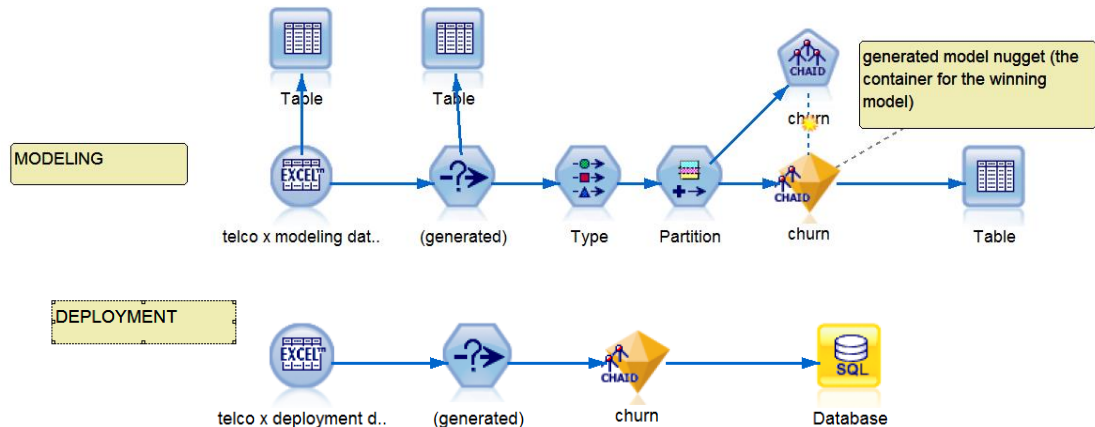
Convert your fitted **Scikit-Learn, R or Apache Spark models and pipelines** into the standardized **Predictive Model Markup Language** (PMML) representation, and make quick and dependable predictions in your **Java/JVM** application.

What happens in **Data Engineering** must also happen in **Deployment** as part of the broader pipeline

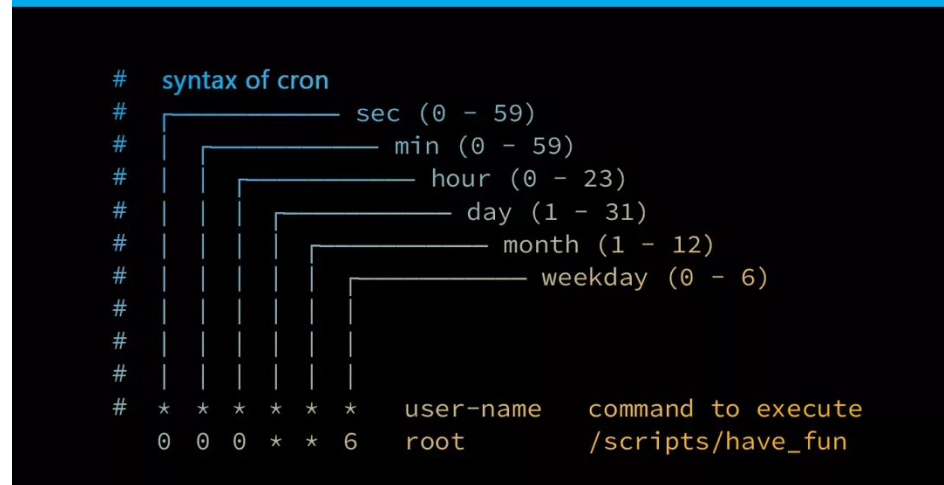
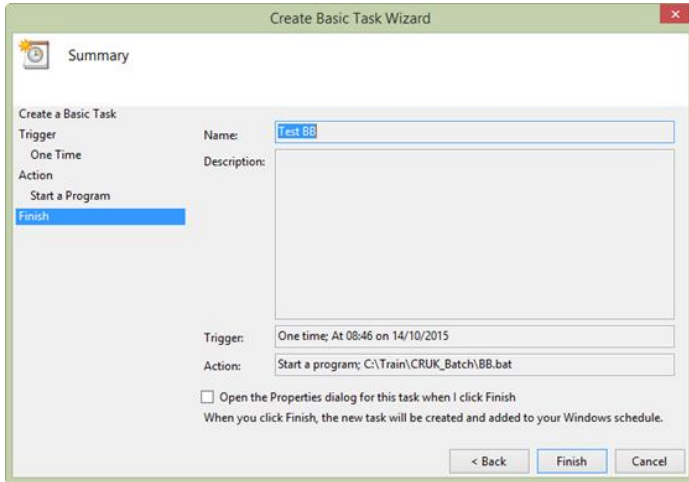
**Option 3 – Using the same engine to build  
(train) and deploy**

# The basic idea

- A. Start with the modelling “pipeline/job”
  - You typically need to adapt it to create a deployment/scoring version
    - Usually a simplified version
- B. Install the software on a deployment machine; R, Python, SAS, SPSS, WPS, etc.
  - Or use the same machine
- C. Periodically run new data through the job and send the scores/predictions to an operational target



# Which means we can usually schedule easily



As long as the analytical tool/engine allows **batch execution** (usually via the command line) we can use standard scheduling tools in the OS e.g. Windows Task Scheduler or CRON to run at a given time/periodically

Real-time would require more engineering

## **Option 4 – Use a Data Science/ML platform**

# Platform essentials

## Support for low/no-code

- Visual interfaces
- May optionally generate code

## And code

- Scripts
- Notebooks

## Support collaboration

- Roles
- Within functions
- Between functions

## Auto ML

- Algorithm selection
- Algorithm settings optimisation
- Feature selection

## Ease of deployment

- Scheduled (batches)
- Real-time

## Post deployment life cycle management

- Monitoring
- Reporting/Alerting
- Model refreshing
- Model retirement

## MLOps support

- Enables integration and scaling of deployments across operational environments e.g. Digital, CRM, Finance, Sales, Contact centre, etc.

# Platform nice-to-haves

## Recommendations

- Auto ML +
- Data engineering
  - Feature extraction/creation

## Advanced stuff

- Simulation
- Optimisation
- Deep learning

## SDKs and APIs

- Support more programmatic code-based model development
  - C, Java, etc.
- API into platform

## Model interpretability

- Tools to help understand models better
  - Many algorithms product opaque models

## Support for GenAI

- Access to LLMs
  - E.g. for RAG
- Foundation models

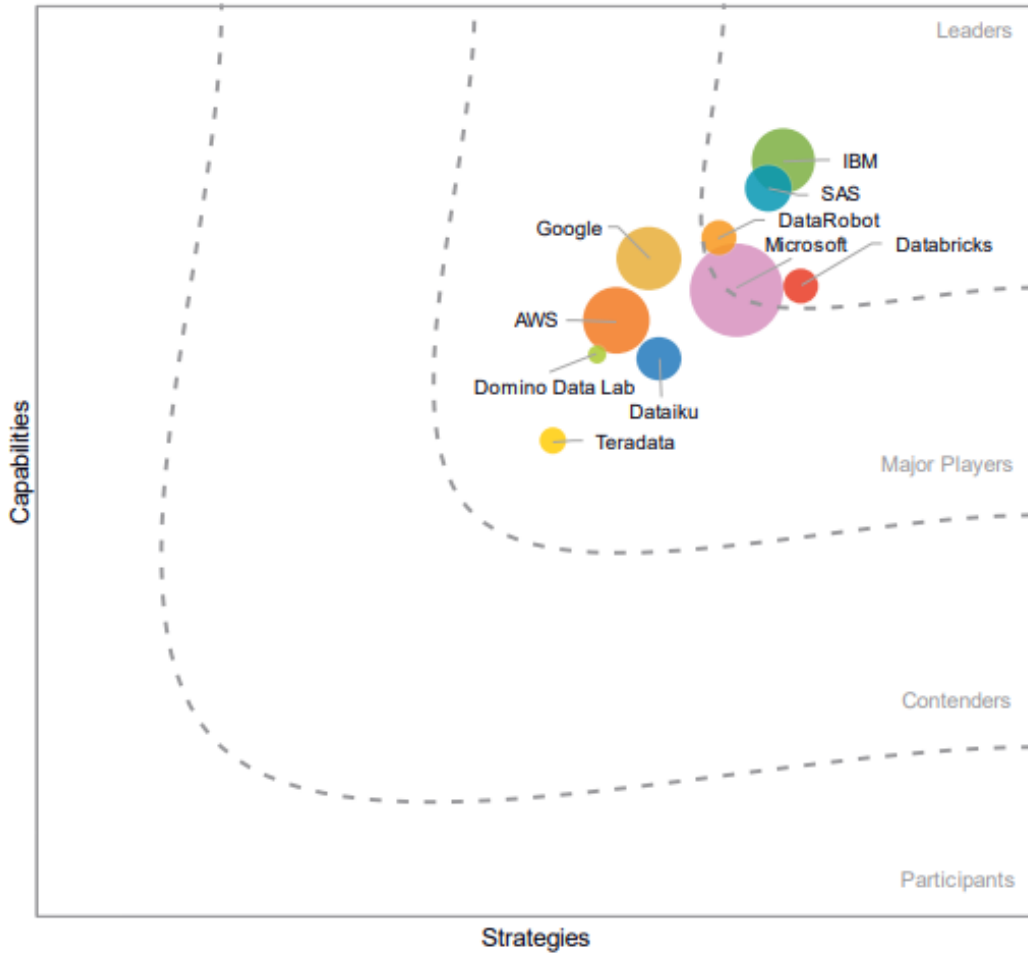




## Magic Quadrant for Data Science and Machine Learning Platforms

[Gartner MQ 2024](#)

- DS/ML Specialists
  - Dataiku, DataRobot, DataBricks, SAS, Altair
- Broader Cloud Platforms
  - Microsoft, Amazon, Google, IBM, Alibaba
- Open source alternatives exist beyond Anaconda e.g. Airflow and MLOps



# IDC MarketScape Worldwide Machine Learning Operations Platforms Vendor Assessment

[IDC Marketplace 2024](#)

# Canvass for Low/No-code

## Model accuracy results

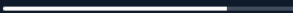
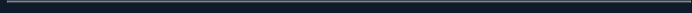
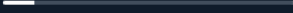
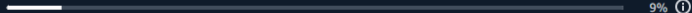
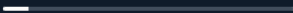

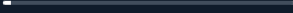
Current model accuracy

95.673%

Great job!

Your model's prediction score is higher than the target of 80%!

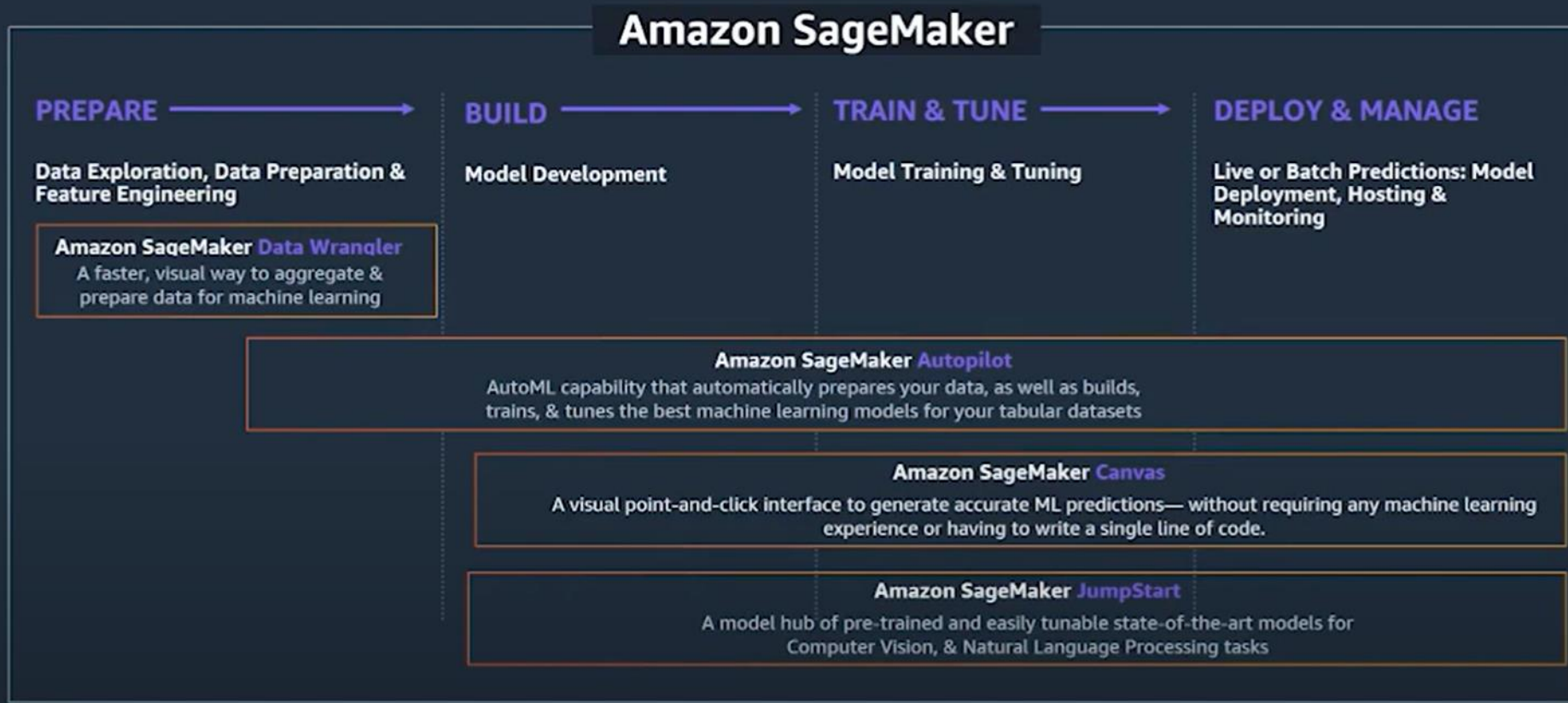
Modify the feature values to see how they affect the On-time prediction in real time.

| Feature       | Feature importance                                                                    | Value | On time prediction                                                                                |
|---------------|---------------------------------------------------------------------------------------|-------|---------------------------------------------------------------------------------------------------|
| Total items   |  77% | 200   |  On time       |
| U.S. ZIP code |  11% | 15203 |  Delayed 9% ⓘ  |
| Temperature   |  9%  | 0°C   |  On time 91% ⓘ |
| Humidity      |  3%  | 20%   |                                                                                                   |

[Back](#)


[Complete the demo](#)

# Low-Code Machine Learning on AWS












AWS Sagemaker support Low-Code ML through to deployment and monitoring

# Notebooks etc. for Code

 jupyter Smarty\_Bot Last Checkpoint: a few seconds ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

       Code   nbdiff

```
In [5]: import streamlit as st
from gtts import gTTS
from PyPDF2 import PdfReader
#from langchain.embeddings import GPT4AllEmbeddings
from langchain_community.embeddings import GPT4AllEmbeddings

In [6]: #from langchain.vectorstores import Chroma
from langchain_community.vectorstores import Chroma
#from langchain.embeddings import HuggingFaceInstructEmbeddings
from langchain_community.embeddings import HuggingFaceInstructEmbeddings

In [ ]: #from langchain.LLms import HuggingFaceHub
from langchain_community.llms import HuggingFaceHub
from langchain.memory import ConversationBufferMemory, ConversationBufferWindowMemory
from langchain.chains import ConversationalRetrievalChain, LLMChain
from langchain.text_splitter import RecursiveCharacterTextSplitter
from langchain_core.prompts import PromptTemplate
#from langchain.memory.chat_message_histories import StreamlitChatMessageHistory
from langchain_community.chat_message_histories import StreamlitChatMessageHistory
#from langchain.retrievers.multi_query import MultiQueryRetriever
import chromadb
from langchain_core.prompts import ChatPromptTemplate
from langchain.retrievers.multi_query import MultiQueryRetriever
import sagemaker, boto3, json
from sagemaker.session import Session

In [8]: from langchain_community.embeddings import HuggingFaceEmbeddings
from langchain_core.prompts import ChatPromptTemplate
from langchain_community.llms import Bedrock
from langchain_community.embeddings import BedrockEmbeddings

In [9]: sagemaker_session = Session()
aws_role = sagemaker_session.get_caller_identity_arn()
aws_region = boto3.Session().region_name
sess = sagemaker.Session()
client = boto3.client('sagemaker')
Bedrockclient = boto3.client("bedrock-runtime")
import torch
device = "cuda" if torch.cuda.is_available() else "cpu"
print(f"Using device: {device}")

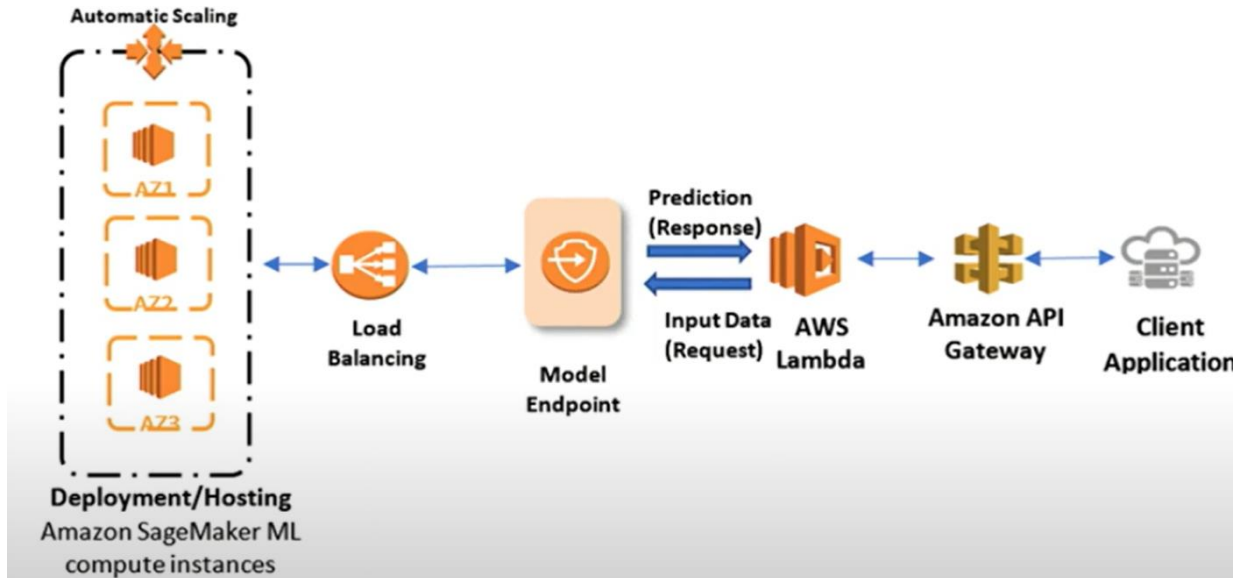
Using device: cpu
```

# Real-time deployment from code

## Hosting / Inference

Once the training is done, we can deploy the trained model as an Amazon SageMaker real-time hosted endpoint. This will allow us to make predictions (or inference) from the model. Note that we don't have to host on the same type of instance that we used to train. Because instance endpoints will be up and running for long, it's advisable to choose a cheaper instance for inference.

```
In [23]: text_classifier = bt_model.deploy(initial_instance_count = 1, instance_type = 'ml.m4.xlarge')
-----!
```



# Premised platforms are also available

The screenshot displays the IBM Deployment Manager interface for a job titled "\*Multi-Step Scoring Job". The main workspace shows a workflow diagram with the following steps and components:

- ETL Completed**: The starting point of the job, represented by an envelope icon.
- Data Preparation**: A step with a checkmark icon, leading to a **Data Preparation Clean-up** task (gear icon).
- Scoring**: A step with a checkmark icon, leading to a **Scoring Clean-up** task (gear icon).
- Export to SPSS**: A step with a checkmark icon, leading to an **Export to SPSS Clean-up** task (gear icon).
- Export to SPSS**: This step branches into two paths:
  - Notify Analysts**: A step with a checkmark icon, leading to an **Executive Summaries Report** (document icon).
  - Performance Report.sps**: A step with a checkmark icon, leading to a **Performance Report.sps** file (document icon).

The left sidebar shows a tree view of the content repository, including folders like "Content Repository", "Jobs", "Data Mining", "Python", "Reporting", "SAS", "Statistics", "Survey", "Various", "Output", "Q", "Question Repository", "Reports", "Samples", "SAS", "Scenarios", "Scripts", "Statistics", "Streams", "test", "Submitted Jobs", "Enterprise View", and "Resource Definitions".

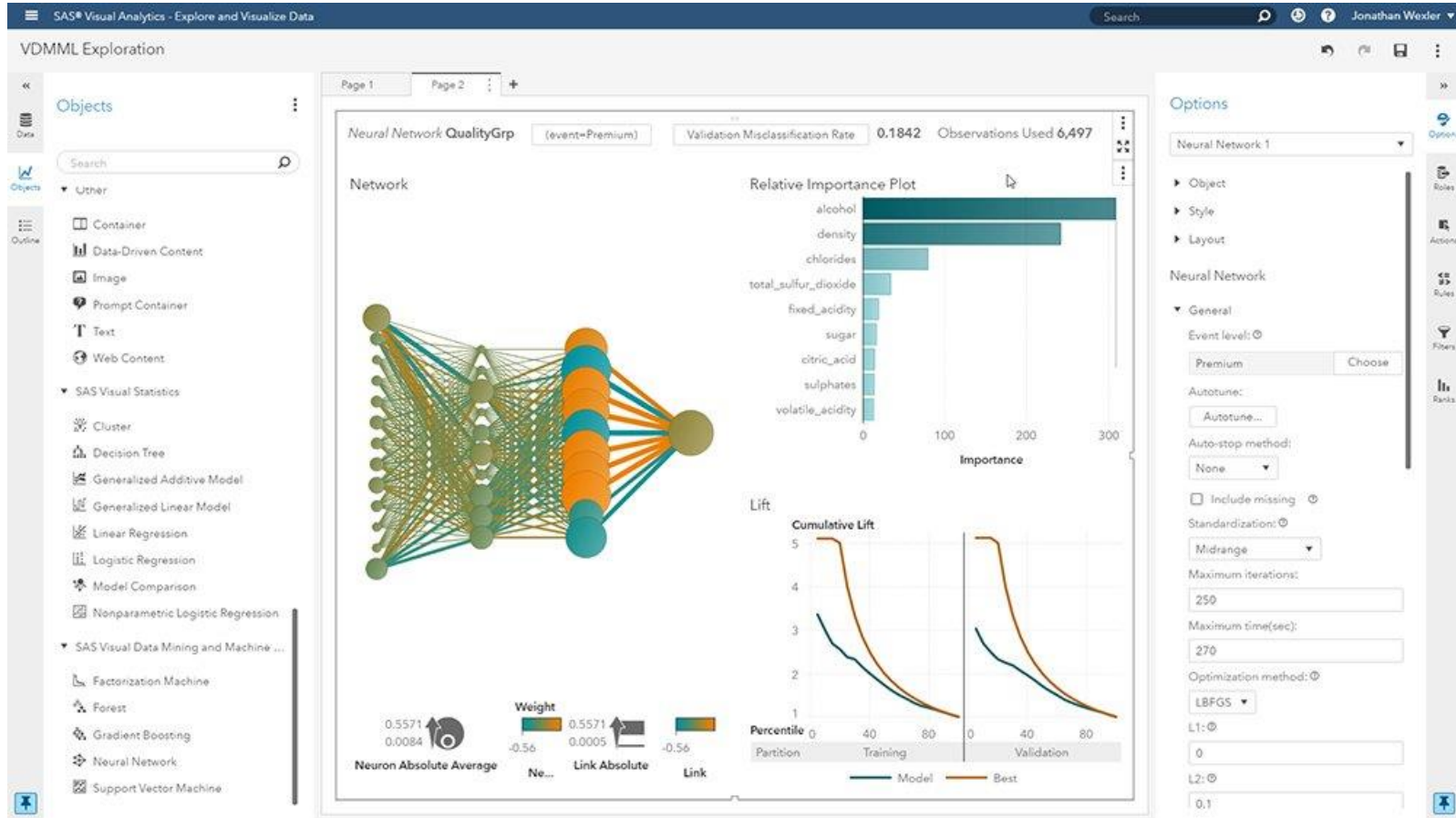
The bottom panel shows the "Notifications Overview" section with the following information:

- Recipients For Job Success Notifications: 9
- Recipients For Job Failure Notifications: 3

The bottom status bar includes tabs for Job History, Job Schedule, Locked Objects, Model Management, Predictors, Scoring, Search Results, and Server Status. The status is "Writable" and "34M/50M".

IBM/SPSS Collaboration & Deployment Services

# Premised platforms are also available



**SAS Viya** is cloud native but it can be installed on-prem using containers



| Blocker                                                                                         | Mitigation                                                                                                                                                                                                                                       |
|-------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Technical                                                                                       | Involve technical colleagues early<br>Figure out the technical path to get there<br>Automate that                                                                                                                                                |
| Data                                                                                            | Improving data quality is a broad project often leading to a Data Lake/Warehouse to support Data Science<br>Tactically assess quality and look to clean what we can and only use data of<br>Assess if this will be sufficient to hit our targets |
| Stakeholder buy-in                                                                              | To get alignment we need to agree what a good outcome is<br>Agree the criteria in a less technical way e.g. Lift, gain, profit                                                                                                                   |
| End user buy-in                                                                                 | Engage early. Get feedback.<br>Make sure there are benefits e.g. productivity and communicate them                                                                                                                                               |
| Model does not work in training. It isn't accurate enough or other success criteria are not hit | Need to clearly agree what "work" means with stake holders. Is any model better than no model?<br>This is the "art+science" of building ML models which can be solved with better data, more feature engineering, algorithmic choices and tuning |
| Model works in training but not in deployment                                                   | Add a QA process<br>Peer reviews                                                                                                                                                                                                                 |

# In summary

- Deployment planning is key
  - Make sure we have a plan to deploy assuming for model success
  - Make sure we set expectations ... we don't always get to where we want
- Get communication/alignment with stakeholders early and often ... don't forget end user stakeholders who may be a broad group who become involved post deployment
- Involve technical colleagues early too
- Depending on where you are in your deployment journey think about the next level of automation
  - The more automated the less work ... the less likely errors
- Platforms are definitely not for all use cases/contexts
  - Depends on scale and availability
  - Platforms/automation do not eliminate blockers on their own ... but they help
- Did we say plan deployment from the beginning?

# Smart Vision Service/Product Offerings

---

- **CONSULTING SERVICES**

Providing expert guidance and strategic advice to help organizations achieve their analytical goals

- **IMPLEMENTATION SERVICES**

Seamless integration and deployment of software, systems, and technologies to ensure successful adoption and utilization.

- **TRAINING, SUPPORT & ENABLEMENT**

Comprehensive training programs and ongoing support to empower clients and ensure they maximize the value of our services. Enabling our customers to become self sufficient through collaborative projects

- **SOFTWARE LICENSING AND SUPPORT**

- **MANAGED SERVICES**

Outsourced data, analytics and business process management solutions that optimize operations, reduce costs, and enhance efficiency.

- **CUSTOM DEVELOPMENT**

Bespoke software and application development to address unique business/analytical requirements, increase automation and provide competitive advantages.



Contact us:

+44 (0)207 786 3568

[info@sv-europe.com](mailto:info@sv-europe.com)

Twitter: @sveurope



[Follow us on LinkedIn](#)



[Sign up for our Newsletter](#)

Thank you