DRAPER&DASH Meetup

Using Supervised ML Methods to augment intelligent dashboards in Healthcare

Gary Hutson – Head of Solutions and AI / ML
Joseph Frost – Director of Solutions and Architecture

About D&D

Draper & Dash Healthcare (D&D) is a London based Venture Capitalist (VC) backed healthcare AI and Machine learning predictive data and analytics company. We leverage data from the public sector and private healthcare companies both nationally and globally. We have been providing organisations with solutions that improve quality, safety, outcomes, efficiencies and opportunities.

Over the years we have worked with over 70 healthcare providers in the UK, US and Australia, helping them to drive cost improvements, form new strategic partnerships, improving health outcomes and more.

The Solution

D&D's solutions give hospitals and healthcare facilities the analytical insight needed to improve patient care and flow throughout the organisation. D&D customers can ensure processes run as efficiently as possible whilst highlighting further improvements and cost-saving opportunities.

Draper & Dash is a provider of **operational patient flow, predictive analytics and insights software**. Our platform is an intrinsic part of how hospitals and healthcare providers run their organisations on a daily basis and is not an optional add-on but rather a fundamental patient flow tool.

We’ll be the presenters this evening

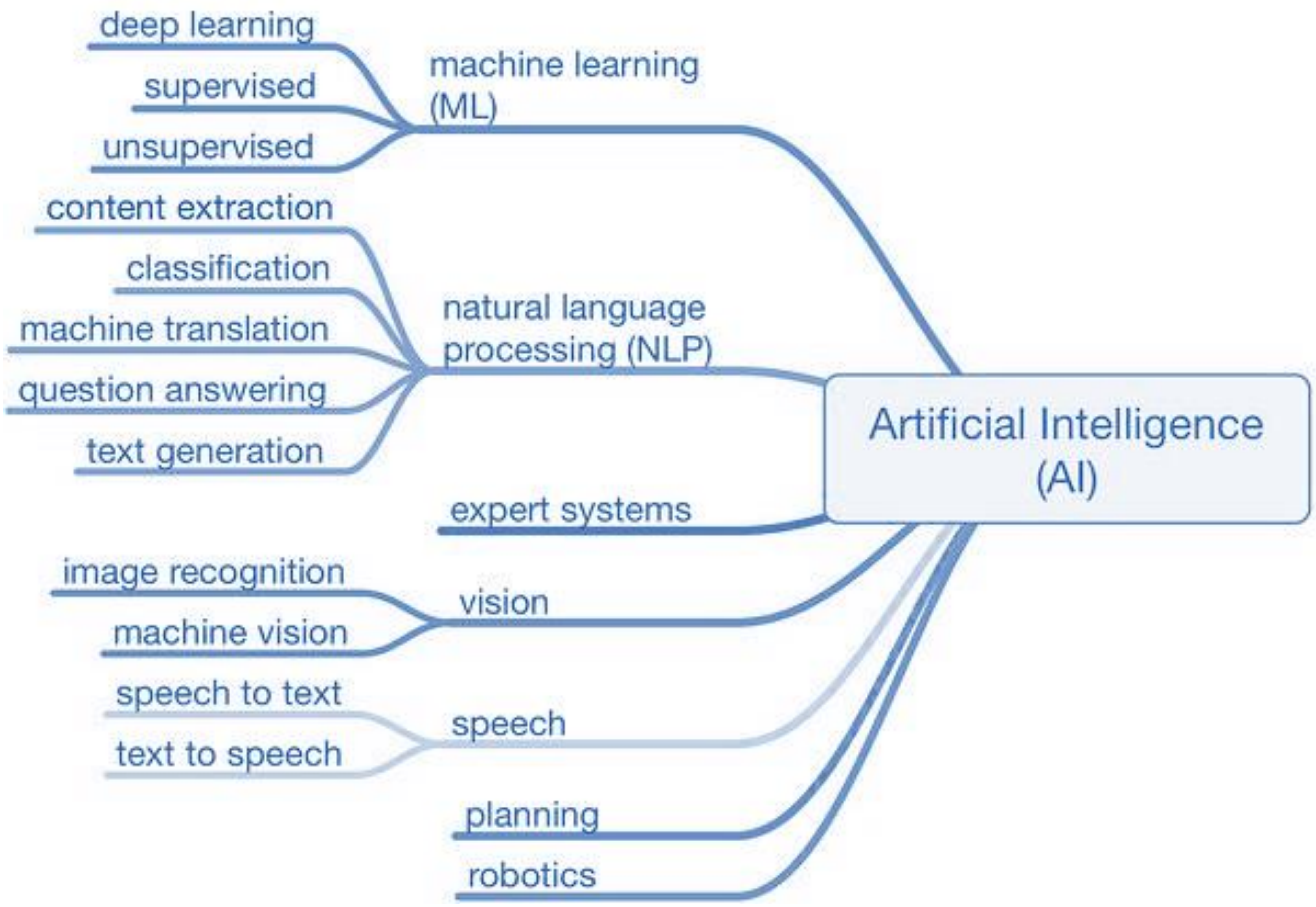
Joseph Frost
Director of Solutions

Gary Hutson
Head of Solutions & AI

How far along are the NHS in terms of data science and ML?

- Aim for the NHS to be the world leader in artificial intelligence and machine learning within 5 years (https://www.england.nhs.uk/2019/06/nhs-aims-to-be-a-world-leader-in-ai-and-machine-learning-within-5-years/).
- Lots of great work happening locally and statistical research published in BMJ, EMJ, etc. However, there is no central function in the NHS to drive this presently.
- Use cases of advances and leading the way regarding AI and ML:
- Addenbrooke hospital in Cambridge is using Microsoft's InnerEye system to automatically process scans for patients with prostate cancer
- Google's DeepMind firm has been working with Moorfield's Eye Hospital NHS Foundation Trust since 2016 to help clinicians improve the way serious eye conditions are diagnosed and treated. See: https://deepmind.com/applied/deepmind-health/working-partners/health-research-tomorrow/moorfields-eye-hospital-nhs-foundation-trust/
- The rise of programming and data science communities for programming in R. The NHS has its own NHS-R Community (https://nhsrcommunity.com/).

Where D&D currently sits on the AI branch



[Visit our website to find out more about our solutions]()

Problem context / background

Most dashboards in the NHS rely on retrospective reporting – of the 'what has happened' state. However, many of the internal monitoring tools do not take into account what is going to happen. This is our selling point, as our products:

- Have custom built visuals and drillable options
- Smart and fit for purpose visualisations
- Predictive integration – from univariate forecasting to predict demand and supervised ML to allow our dashboards to become intelligent dashboards – i.e. the predictions improve with the number of retrospective observations we can observe
- Gap in the market for augmented dashboarding in the NHS

Visit our website to find out more about our solutions

Aim and objectives

The aim being:

- to augment the historic descriptive analytics with prospective (forward looking) solutions.
- In addition, to use our in house ML pipeline support the training, testing and production of our supervised ML models.

The objectives:

- to allow services and healthcare partners to have advanced warning of prospective system blockages / capacity shortfalls, etc.
- Provision of useful tools that can be replicated across the healthcare landscape
- Great value decision making tools to aid with effective and smarter working

Visit our website to find out more about our solutions

Stranded Dashboard in focus

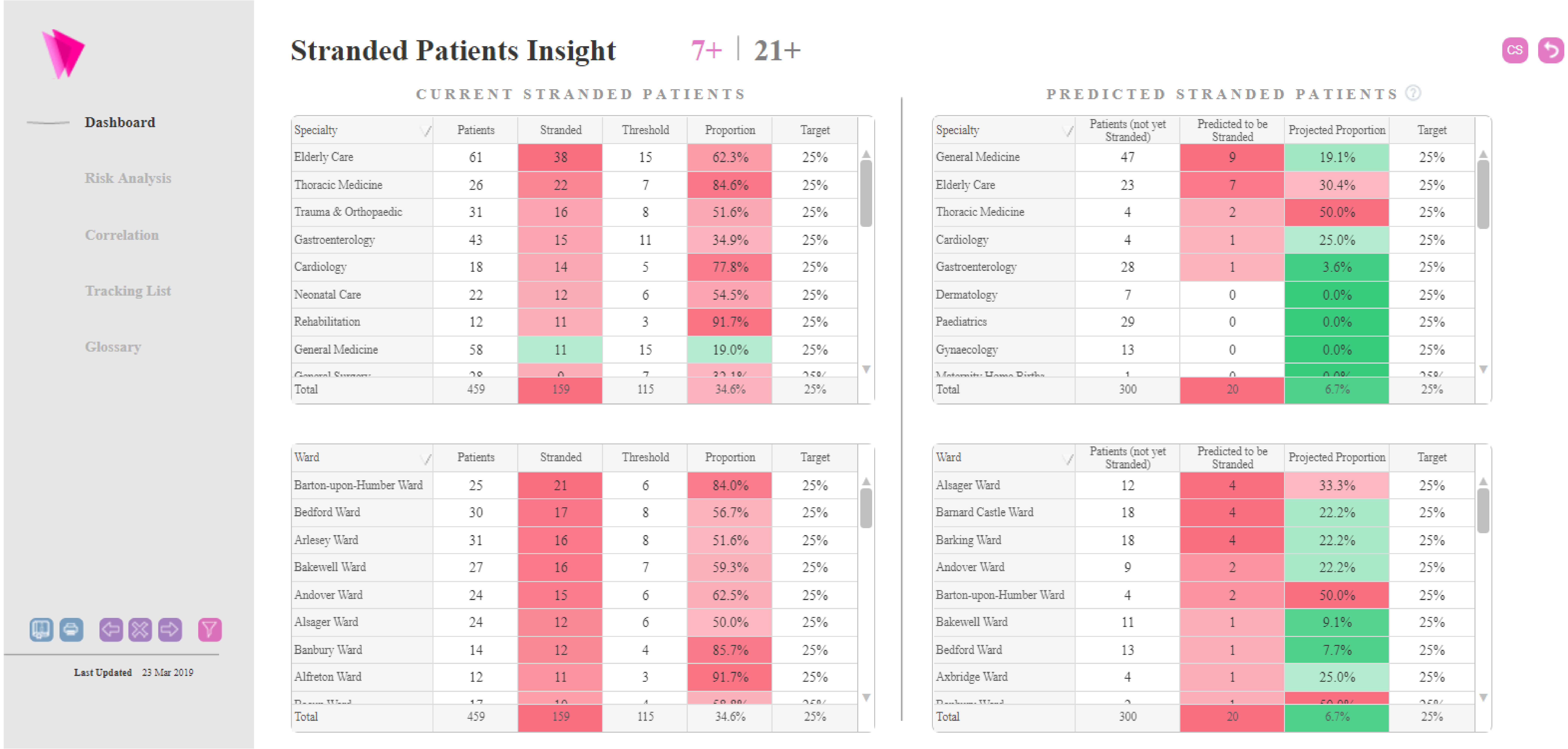
Dashboard 'in focus'

The example dashboard we are working with is the stranded application. This application allows healthcare services to detect the likelihood of a patient being stranded, dependent on a host of feature variable. The app includes:

- Dashboard – containing the current stranded patients by specialty and ward and the predicted stranded patients for the same breakdown
- Risk analysis – the driver here is our Machine Learning approach and the algorithms we have deployed
- Correlation plot – this is a correlation plot of the specific insights
- Tracking list – patient level – this gives an overview of patients with relevant data items appended to the patient, alongside things like their time (length of stay in the department) and their dominant factors. These rely on D&Ds own importance factor algorithm.

[Visit our website to find out more about our solutions]

Dashboard view



CURRENT STRANDED PATIENTS

Specialty	Patients	Stranded	Threshold	Proportion	Target
Elderly Care	61	38	15	62.3%	25%
Thoracic Medicine	26	22	7	84.6%	25%
Trauma & Orthopaedic	31	16	8	51.6%	25%
Gastroenterology	43	15	11	34.9%	25%
Cardiology	18	14	5	77.8%	25%
Neonatal Care	22	12	6	54.5%	25%
Rehabilitation	12	11	3	91.7%	25%
General Medicine	58	11	15	19.0%	25%
Total	459	159	115	34.6%	25%

Ward	Patients	Stranded	Threshold	Proportion	Target
Barton-upon-Humber Ward	25	21	6	84.0%	25%
Bedford Ward	30	17	8	56.7%	25%
Arlesey Ward	31	16	8	51.6%	25%
Bakewell Ward	27	16	7	59.3%	25%
Andover Ward	24	15	6	62.5%	25%
Alsager Ward	24	12	6	50.0%	25%
Banbury Ward	14	12	4	85.7%	25%
Alfreton Ward	12	11	3	91.7%	25%
Total	459	159	115	34.6%	25%

PREDICTED STRANDED PATIENTS

Specialty	Patients (not yet Stranded)	Predicted to be Stranded	Projected Proportion	Target
General Medicine	47	9	19.1%	25%
Elderly Care	23	7	30.4%	25%
Thoracic Medicine	4	2	50.0%	25%
Cardiology	4	1	25.0%	25%
Gastroenterology	28	1	3.6%	25%
Dermatology	7	0	0.0%	25%
Paediatrics	29	0	0.0%	25%
Gynaecology	13	0	0.0%	25%
Total	300	20	6.7%	25%

Ward	Patients (not yet Stranded)	Predicted to be Stranded	Projected Proportion	Target
Alsager Ward	12	4	33.3%	25%
Barnard Castle Ward	18	4	22.2%	25%
Barking Ward	18	4	22.2%	25%
Andover Ward	9	2	22.2%	25%
Barton-upon-Humber Ward	4	2	50.0%	25%
Bakewell Ward	11	1	9.1%	25%
Bedford Ward	13	1	7.7%	25%
Axbridge Ward	4	1	25.0%	25%
Total	300	20	6.7%	25%

[Visit our website to find out more about our solutions]

Insights view

Dashboard

Risk Analysis

Correlation

Tracking List

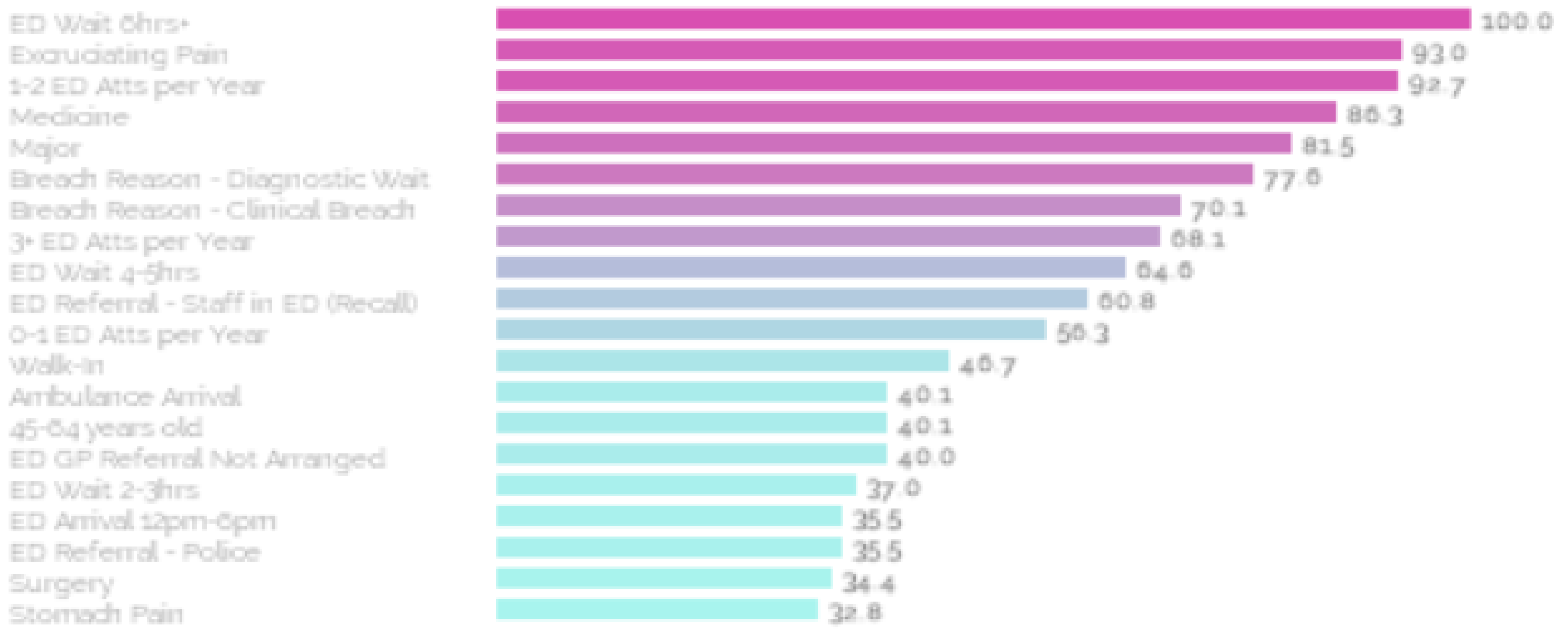
Glossary

Last Updated 23 Mar 2019

Stranded Patients Insight 7+ | 21+

CS

20 Most Important Variables that drive Stranded Patients Prediction



The machine learning algorithm takes various contributing factors towards a patient becoming stranded and weights their impact on the overall position.

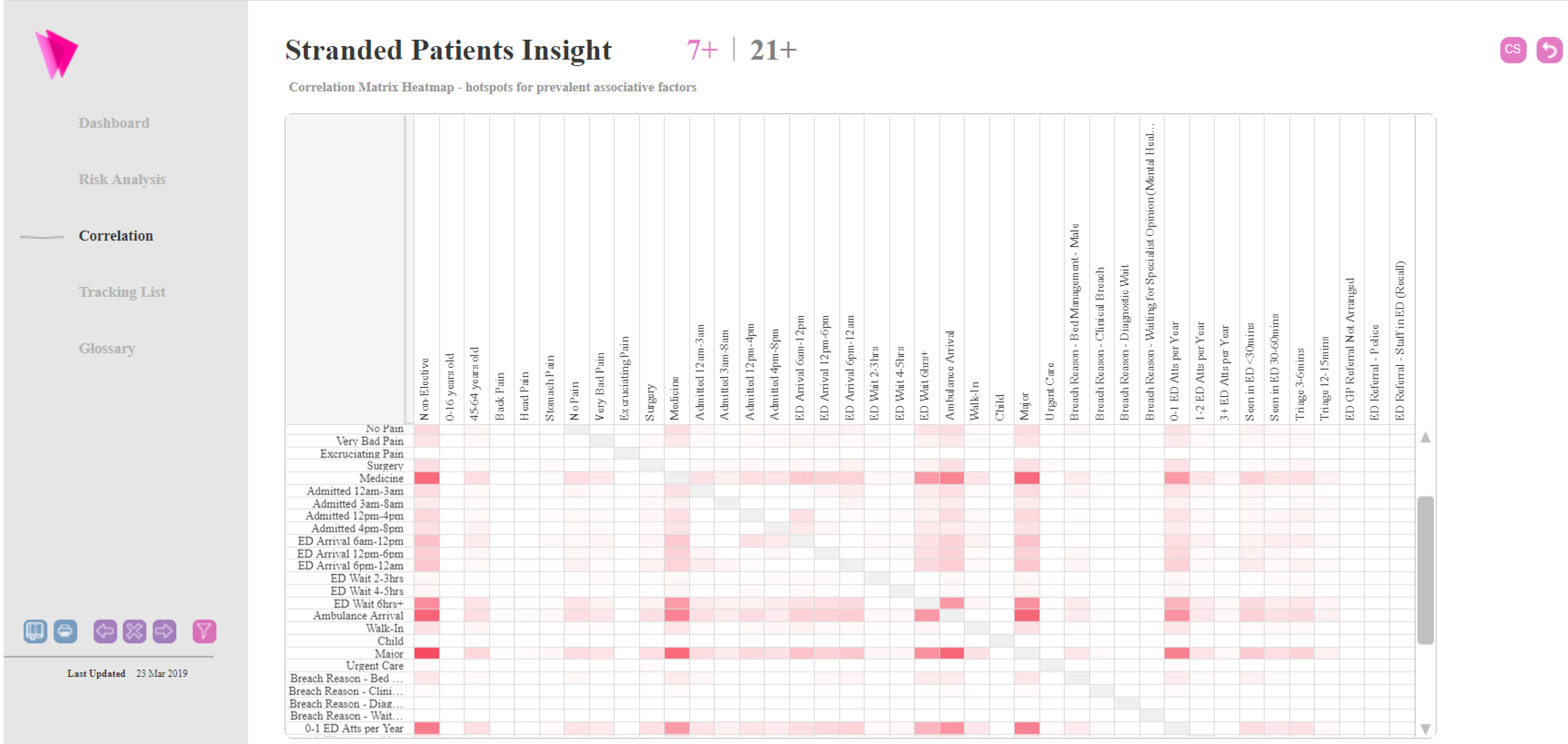
The following risk analysis aggregates these figures up from patient level to Specialty or Ward in an effort to discover where certain factors are more prominent so that investigations into why can occur. Natural associations may apply but where concerning figures are highlighted, resource and attention can be diverted in order to address what could be a pertinent process issue increasing the risk of stranded patients.

Current Prominence of Stranded Factors by Specialty | **Ward** | Consultant

Ward	Non-Elective	Major	Ambulance Arrival	Medicine	0-1 ED Atts per Year	ED Wait 6hrs+	ED Arrival 6am-12pm	ED Arrival 6pm-12am	Seen in ED <30mins	ED Arrival 12pm-6pm	Triage 3-6mins	45-64 years old
Alsager Ward	66.7	66.7	54.2	58.3	37.5	41.7	37.5	16.7	16.7	8.3	25.0	4.2
Alton Ward	25.0	25.0	25.0	25.0	-	25.0	25.0	-	-	-	25.0	-
Andover Ward	66.7	66.7	62.5	62.5	45.8	54.2	12.5	16.7	29.2	29.2	8.3	8.3
Arlesey Ward	45.2	41.9	45.2	12.9	41.9	29.0	3.2	16.1	6.5	16.1	3.2	6.5
Axbridge Ward	55.6	55.6	44.4	33.3	55.6	33.3	-	22.2	-	33.3	-	22.2
Axminster Ward	100.0	100.0	-	100.0	100.0	-	-	-	-	-	-	-
Bacup Ward	47.1	47.1	41.2	5.9	35.3	17.6	29.4	17.6	5.9	-	23.5	23.5
Bakewell Ward	59.3	55.6	59.3	51.9	40.7	37.0	14.8	25.9	18.5	14.8	7.4	7.4
Bampton Ward	50.0	50.0	-	-	50.0	-	-	50.0	-	-	50.0	-
Banbury Ward	85.7	85.7	71.4	85.7	64.3	57.1	28.6	14.3	21.4	35.7	7.1	35.7
Barking Ward	22.2	22.2	22.2	22.2	16.7	11.1	5.6	5.6	-	11.1	11.1	-
Barnard Castle Ward	26.3	26.3	26.3	26.3	10.5	21.1	15.8	10.5	10.5	-	15.8	-
Barnstaple Ward	100.0	100.0	-	100.0	100.0	-	-	-	-	100.0	-	-
Barton-upon-Humber Ward	91.7	91.7	75.0	91.7	58.3	54.2	41.7	25.0	45.8	16.7	25.0	25.0
Batley Ward	100.0	100.0	100.0	-	100.0	-	100.0	-	-	-	-	100.0
Bedford Ward	56.7	56.7	33.3	56.7	40.0	36.7	20.0	20.0	13.3	13.3	13.3	23.3

Visit our website to find out more about our solutions

Correlation view



[Visit our website to find out more about our solutions](http://www.draperanddash.com)

Demonstration of our o
solutions

Our steps to designing ML model

Hurdles encountered along the way

- Capacity
- Understanding that training some of our models can take over 6 hours – so doing this more than once can waste valuable time – therefore the importance of domain knowledge and getting the data inputs right is crucial
- Technology for our data scientists – we need good kit – training models occupies system memory quickly with large datasets
- Domain knowledge – data scientists working with domain experts is key. It upskills both. However, capacity is a premium and this is not always possible
- Scope evolves through our projects – so models must change with the scope
- Demand – our customers so far opt for dashboards over predictive analytics and machine learning. Therefore, we need to create demand by showing them what is possible
- Culture of healthcare – reactionary centric, as opposed to prospective and future looking. We can change this be the advocation of intelligent dashboards augmented by predictive analytics

Visit our website to find out more about our solutions

Supervised ML algorithms – focus on transferability and generalisability

Our algorithms are transferrable to other NHS providers, as they all share some common design principles:

- We use a minimal data set – all the features we engineer are available across all NHS trusts
- The techniques and methods used are generalisable and can we replicated with R
- Domain experts are the key here – knowing what features to include in your models upfront, without statistical elimination is key. This is our strength in healthcare, as we employ ex NHS analysts, developers and data scientists to work on augmenting our dashboards
- Granularity is important to design in – so our algorithms are built at a macro level allowing the filters that are applied, say specialty (the function treating the patient e.g. cardiology, oncology, etc.)

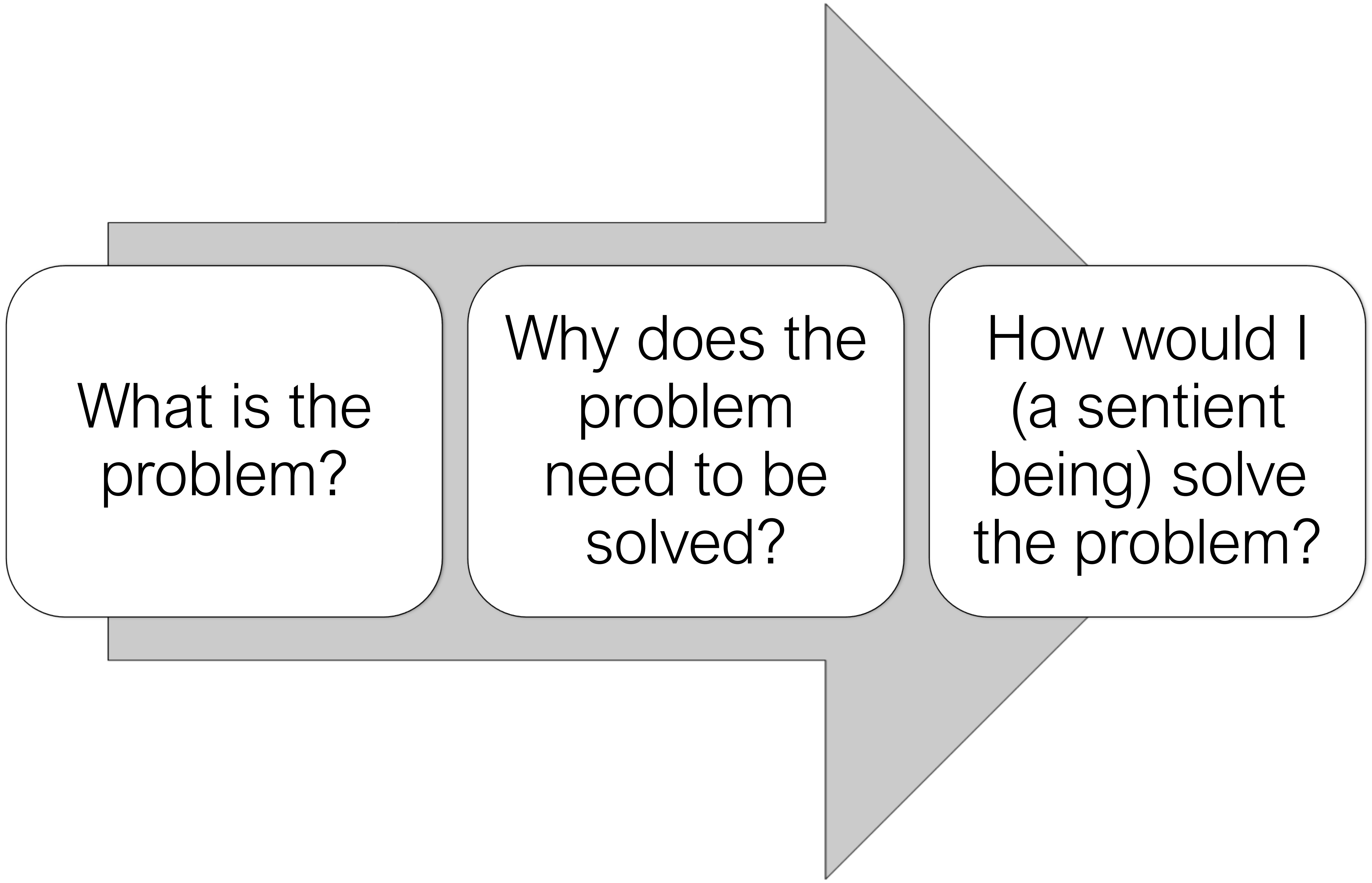
Visit our website to find out more about our solutions

Supervised ML algorithms – development process

1. What is the problem you are trying to solve with ML
2. Think about how the problem, how this could be an issue to other healthcare providers and will the problem be the same in other areas
3. Agree on the approach – whether this be supervised (regression / classification), unsupervised or more deep methods such as neural nets
4. Think about how to evaluate the model, how the data is going to be partitioned (cross validation, hold out, etc.), for classification understand and adjust for class imbalance
5. Start with the data, feature encode, clean and prepare the data and work closely with the BI development team to make sure the underlying datasets do not change, or the problem domain has not shifted
6. Choose the right algorithm for the task, use multiple algorithms and benchmark the accuracy (or other performance metrics) against, or evolve this over time. Usually, our data science team knows what models to use for the purpose – this comes with experience
7. Train the model
8. Test the model – regression (RMSE) and classification (confusion matrices) by making predictions using model coefficients / rules
9. Improve the model
10. Deploy the model

[Visit our website to find out more about our solutions](www.draperanddash.com)

Supervised ML algorithms – first phase – problem definition



What is the problem?

- Informal description – design it from a friend or colleagues viewpoint. For example "I need a solution that will identify stranded patients".
- Look for similar problems

Why does the problem need to be solved?

Examine:

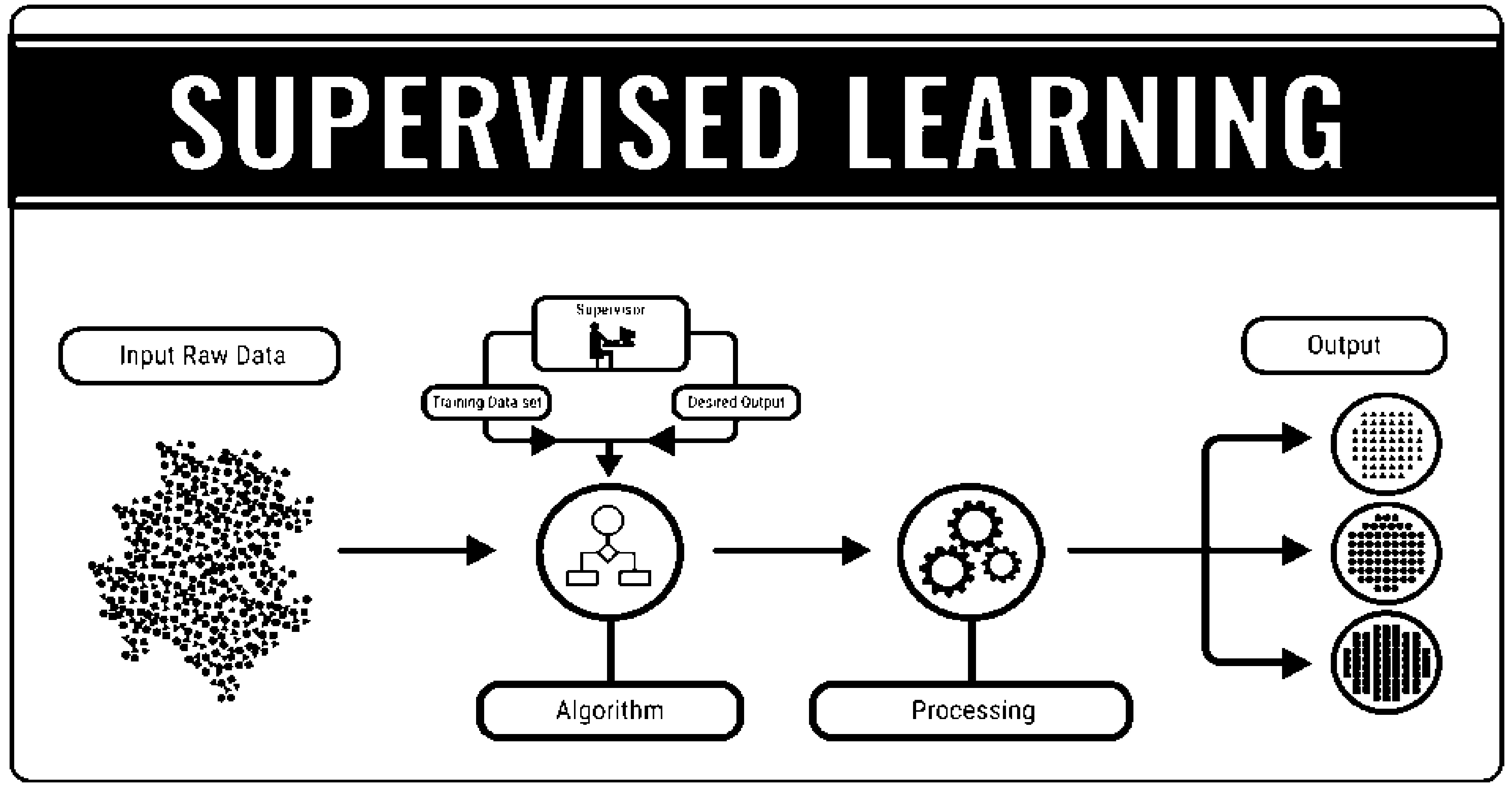
- Motivation – what is the motivation for solving the problem
- Solution benefits – what will be the espoused benefits of the solution
- Use – how will this be used in practice

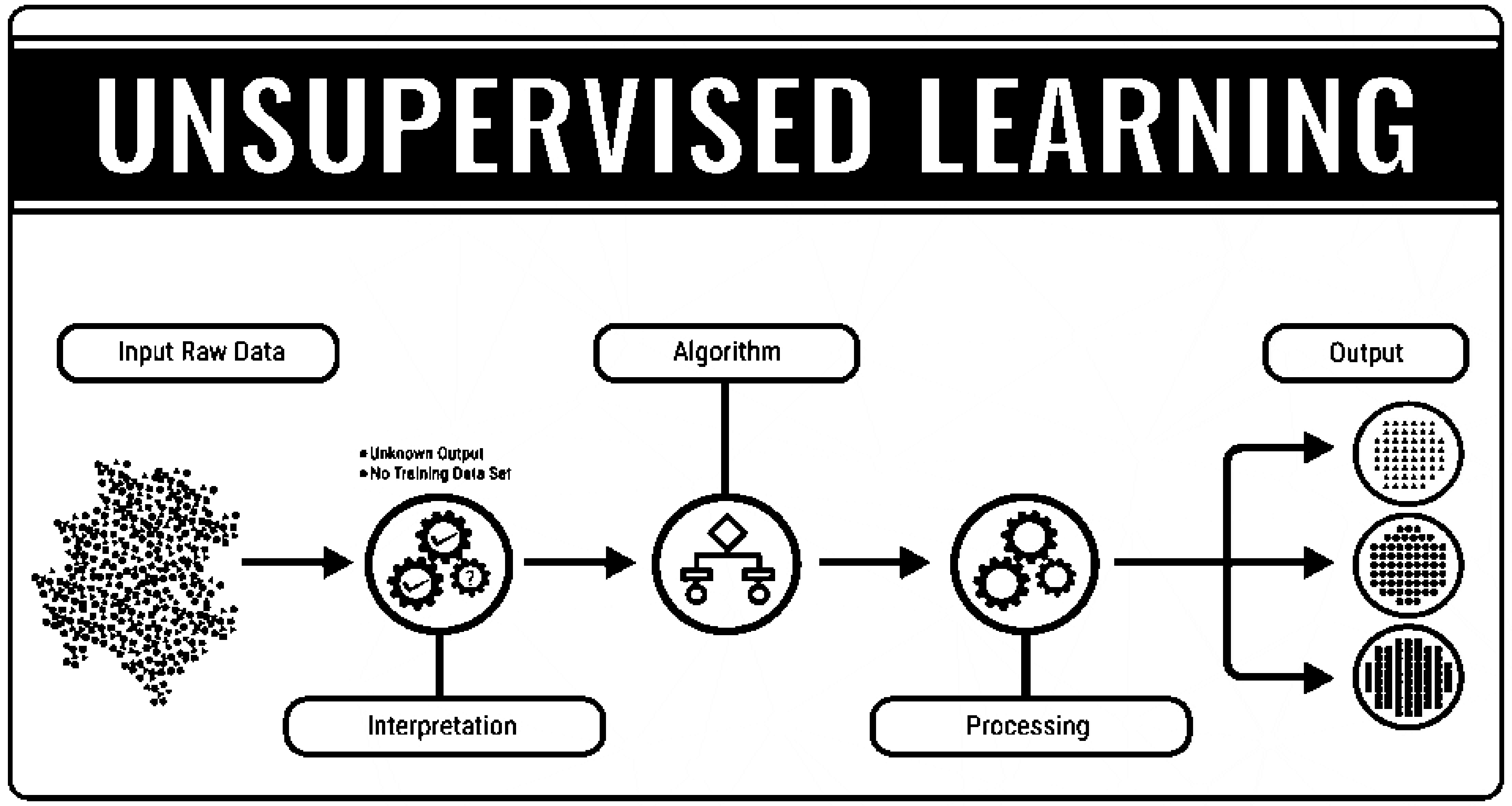
How would you (sentient being) solve the problem?

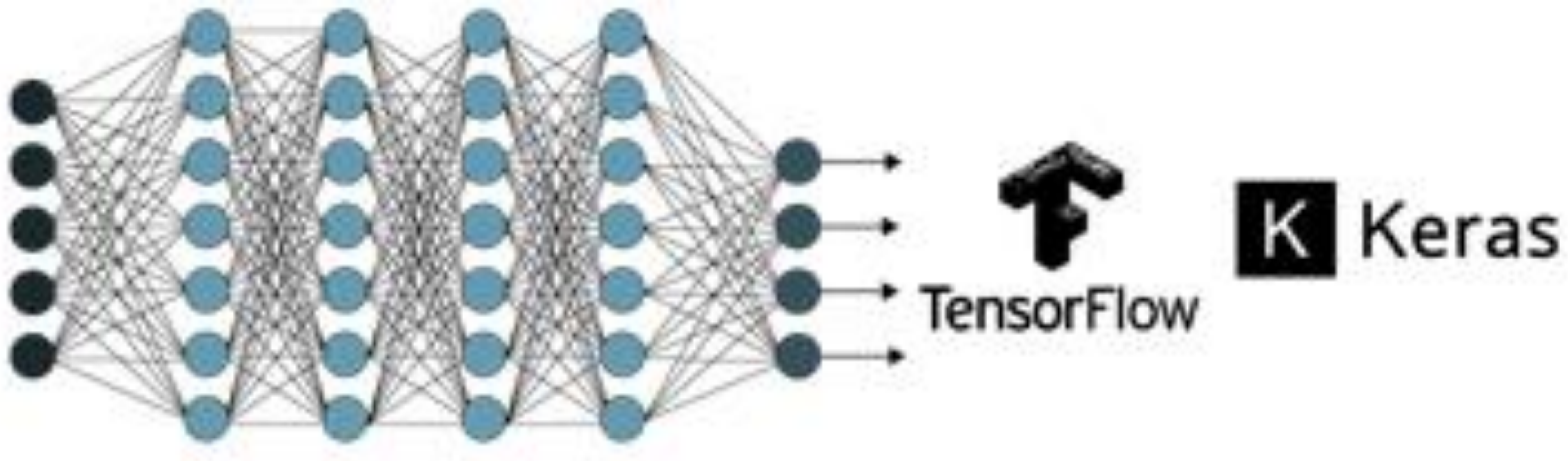
- If I were to do this manually – how would it work?

[Visit our website to find out more about our solutions](http://www.draperanddash.com)

Supervised ML algorithms – second phase – approach / method





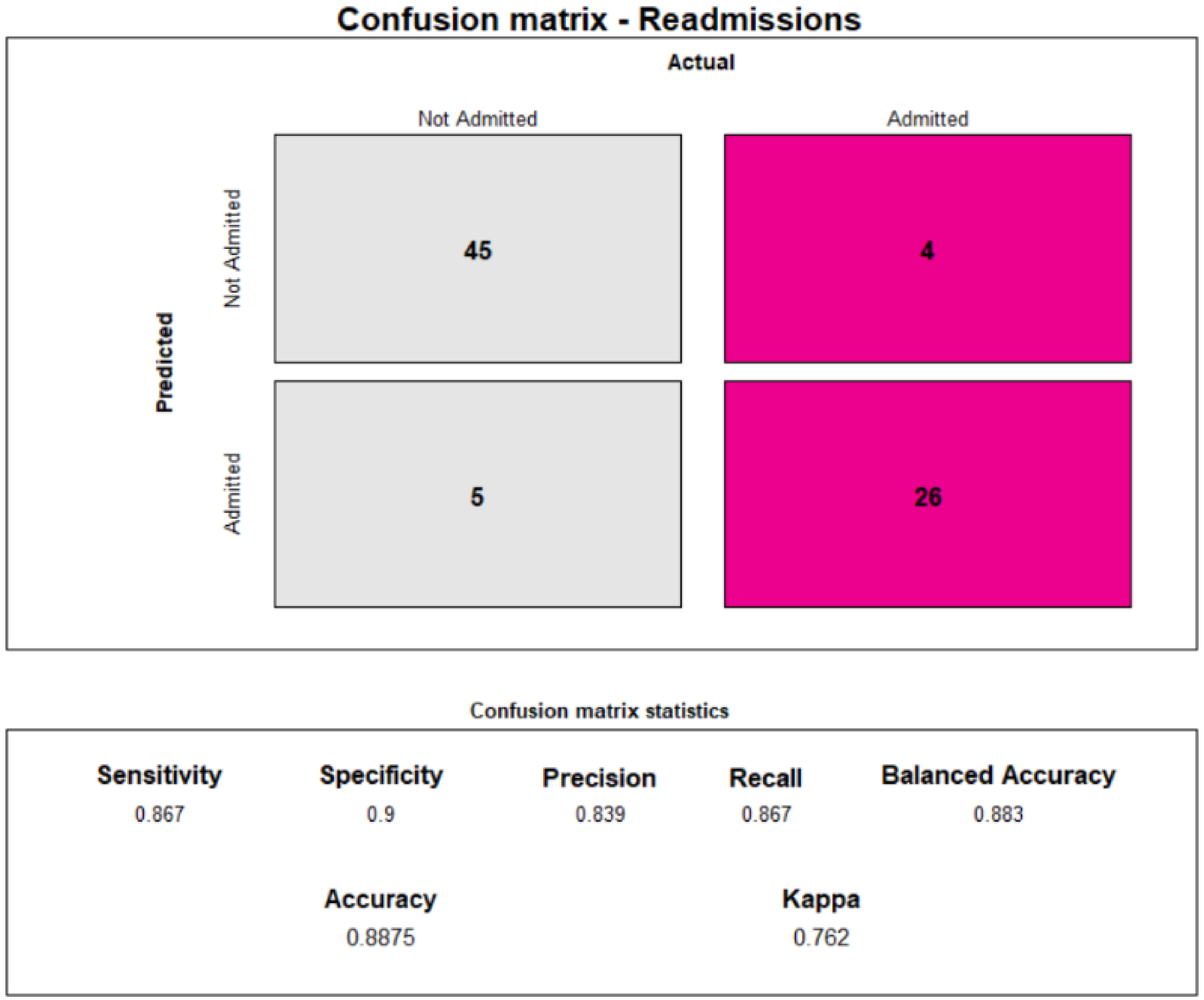


[Visit our website to find out more about our solutions](http://www.draperanddash.com)

Supervised ML algorithms – third phase – plan for evaluation

Evaluation

- Confusion matrices – these are our go to for evaluating our classification models. We have developed a nice confusion matrix function in R: https://www.draperanddash.com/machinelearning/2019/07/confusion-matrices-evaluating-your-classification-models/
- Packaged up and ready to use. The above link gives you the information needed to evaluate a binary classification problem. A multiclass version is currently in development.
- Regression – I tend to like to compare Root Mean Squared Error and AIC to benchmark the performance of my regression models

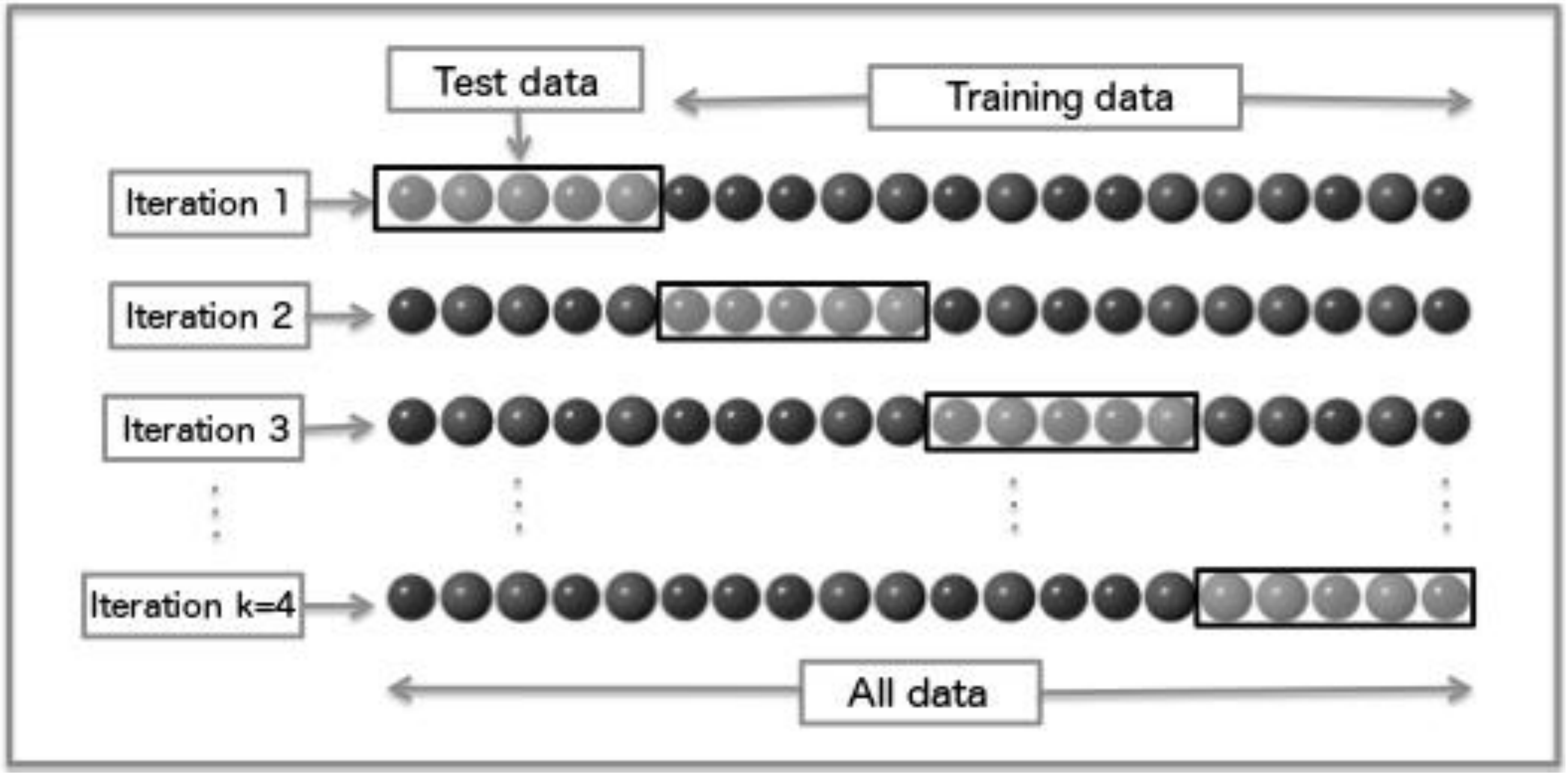


[Visit our website to find out more about our solutions](https://www.draperanddash.com)

Supervised ML algorithms – third phase – plan for evaluation (cont.)

Data partitioning

- Training and test set splits
- Training, test and validation set splits
- K fold cross partitioning (K=10 is our default) then resample to compare accuracy of subsamples



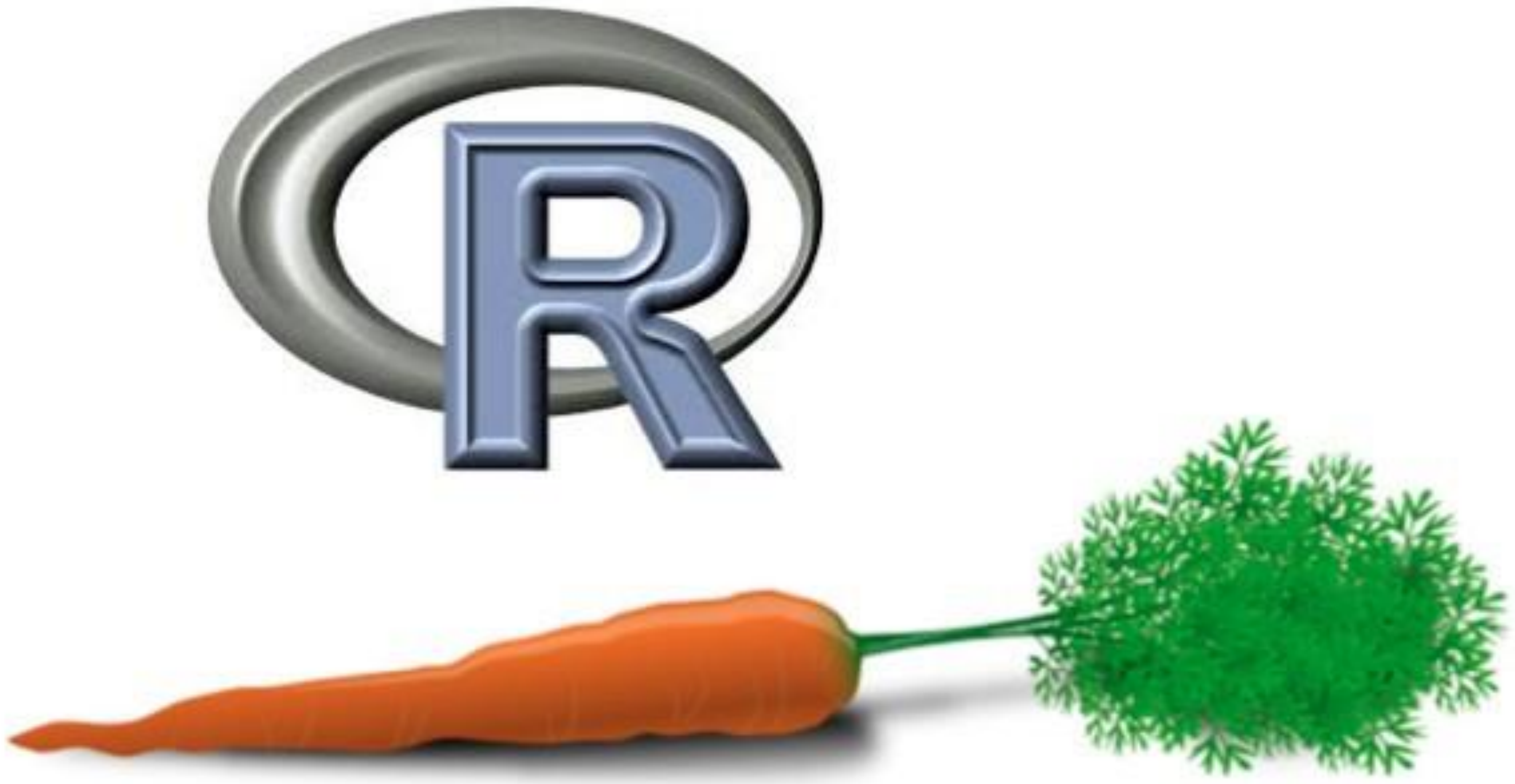
Class imbalance

- Use SMOTE to under or over sample – R has an implementation here: https://www.rdocumentation.org/packages/DMwR/versions/0.4.1/topics/SMOTE

Supervised ML algorithms – fourth and sixth phase– choosing algorithm(s)

Choosing algorithm

- Decide whether you want a probabilistic approach, a classification approach or a regression approach to your ML model
- For classification, our go to is the Caret package. See this list for the available packages in caret (classification and regression training) package: https://rdrr.io/cran/caret/man/models.html



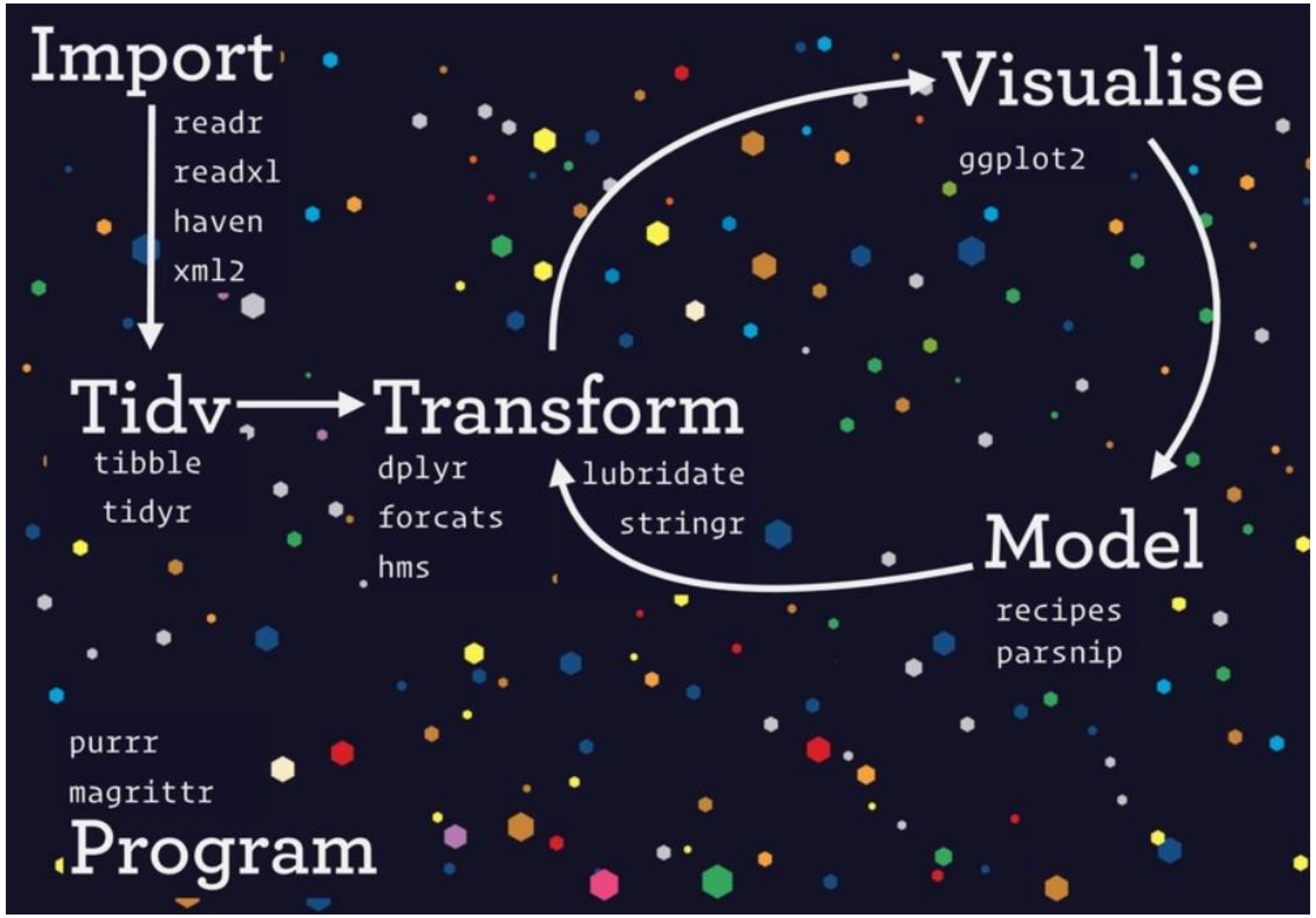
Why CARET?

- The main reason is that it provides a consistent framework for training and evaluating machine learning models in R. We have created a template function which allows any caret supported model to be passed and it returns all the key information as a list of outputs. If this template was not available, then this would become more difficult.

Visit our website to find out more about our solutions

Supervised ML algorithms – fifth phase – getting the data in the right shape

- **Data Import** – R has numerous packages for data import. These are some of the most popular packages: readr (comma separated data), readxl (import Excel). Haven (read outputs from other stats packages such as SPSS), httr (read elements from website – normally html tables), rvest (web scraping), xml2 (xml file structure reading) and RODBC (for reading in database data)
- **Data Cleaning** – for data cleaning and transformation: tidyr (tidys data and allows multiple row and column wise operations), dplyr (all things data transformation – joining data, summarising data, renaming and adding columns, etc.), forcats (package for working with factors), hms (time formatting package), lubridate (date and time manipulation package) and stringr (string matching and formatting)
- **One hot encoding** – to retain control – we tend to do this with dplyr using the case_when() syntax with the creation of new columns (mutate)



[Visit our website to find out more about our solutions](www.draperanddash.com)

Supervised ML algorithms – seventh to eighth phase– train, predict and evaluate model

Train model

- Train the goal variable, or variables, with the relevant ML algorithm of choice. For the stranded – we used a Naïve Bayes and a regression tree to produce the probability estimates.

Predict

- Use held back data (test / validation) data to make predictions – these could be classifications or numerical predictions (regression results)

Evaluate the model

- We previous saw that we have created a custom confusion matrix for evaluation of the models. Please refer to this link, as it provides you with all the narrative around the performance benchmarks as well.
- ROC curves can also be used, for classification, see for https://cran.r-project.org/web/packages/pROC/pROC.pdf

Supervised ML algorithms – ninth phase – improve the model

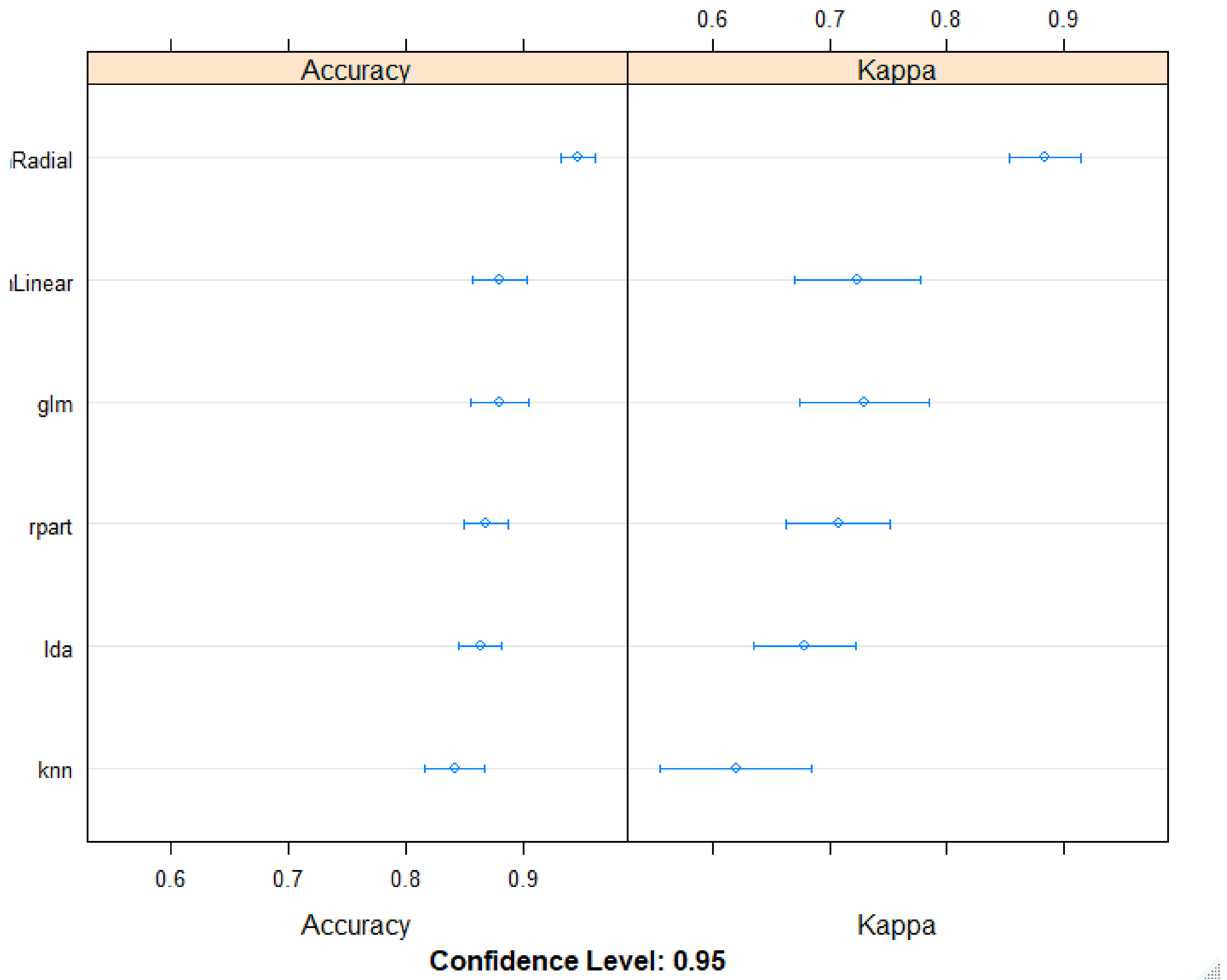
- Explore bagging algorithms in R – these are ensemble methods such as bag of trees and random forests – which use a voting mechanism and data sampling logic – to improve accuracy and pattern detection. However, can be computationally expensive with larger datasets
- Explore boosting models – gradient boosting machines (GBM) have a great implementation in R: https://datascienceplus.com/gradient-boosting-in-r/
- Stacking algorithms – I use these a lot, as it allows you to stack a family of algorithms together and these models are then used to improve the results of the predictions (see implementation on next slide). For classification the default implementation is majority voting (see:
- Look for obvious issues – overfitting, underfitting, dependence in factors, etc.
- Partition your data differently to see if this improves the accuracy output
- Add more data
- Scaling and feature transformation
- Feature selection and deselection
- Algorithm tuning – adjust model parameters – for random forests this could be the number of trees grown, etc. Iterate through to hyperparameter tune in code
- Work with the domain knowledge experts to find out more about the data you are trying to model

Supervised ML algorithm Improvement - stacking implementation in R

#Generate list of algorithms and train the multiple models with multiple resamples using the **caretList** function

```
seed <- 123
train_ctl <- trainControl(method="repeatedcv", number=10, repeats=3,
                          savePredictions = TRUE, classProbs=TRUE)
algo_list <- c('lda', 'rpart', 'glm', 'knn', 'svmRadial', 'svmLinear')
set.seed(seed)
class_name <- "Class"
models <- caretList(as.formula(paste(class_name, "~ .")),
                    data=dataset, trControl=train_ctl, methodList=algo_list)
models
results <- resamples(models)
summary(results)
dotplot(results)
```

```
Call:
summary.resamples(object = results)

Models: lda, rpart, glm, knn, svmRadial, svmLinear
Number of resamples: 30

Accuracy
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
lda       0.7647059 0.8333333 0.8611111 0.8632524 0.8857143 0.9714286    0
rpart     0.7647059 0.8333333 0.8840336 0.8689418 0.9117647 0.9444444    0
glm       0.6764706 0.8382353 0.8987395 0.8793931 0.9166667 1.0000000    0
knn       0.6944444 0.7969771 0.8452381 0.8414970 0.8857143 0.9722222    0
svmRadial 0.8571429 0.9166667 0.9428571 0.9467507 0.9722222 1.0000000    0
svmLinear 0.7058824 0.8539916 0.8888889 0.8802661 0.9160714 0.9722222    0

Kappa
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
lda       0.4594595 0.6129032 0.6715328 0.6790697 0.7371753 0.9378330    0
rpart     0.4594595 0.6294222 0.7448583 0.7077766 0.7970611 0.8795987    0
glm       0.3048327 0.6243361 0.7843134 0.7295437 0.8163265 1.0000000    0
knn       0.1885246 0.5116055 0.6280481 0.6202616 0.7380667 0.9387755    0
svmRadial 0.6679317 0.8200499 0.8747793 0.8836278 0.9387755 1.0000000    0
svmLinear 0.3307087 0.6505741 0.7419355 0.7236712 0.8092292 0.9387755    0
```



[Visit our website to find out more about our solutions](http://www.draperanddash.com)

Supervised ML algorithm Improvement – stacking implementation in R

```
#Use higher order model to improve the results by majority voting for classification

stack_ctl <- trainControl(method="repeatedcv", number=10, repeats=3,
                          savePredictions=TRUE, classProbs=TRUE)

set.seed(seed)
rand_forest.stack <- caretStack(models, method="rf",
metric="Accuracy", trControl=stack_ctl)
print(rand_forest.stack)
```

Original results

```
Call:
summary.resamples(object = results)

Models: lda, rpart, glm, knn, svmRadial, svmLinear
Number of resamples: 30

Accuracy
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
lda       0.7647059 0.8333333 0.8611111 0.8632524 0.8857143 0.9714286    0
rpart     0.7647059 0.8333333 0.8840336 0.8689418 0.9117647 0.9444444    0
glm       0.6764706 0.8382353 0.8987395 0.8793931 0.9166667 1.0000000    0
knn       0.6944444 0.7969771 0.8452381 0.8414970 0.8857143 0.9722222    0
svmRadial 0.8571429 0.9166667 0.9428571 0.9467507 0.9722222 1.0000000    0
svmLinear 0.7058824 0.8539916 0.8888889 0.8802661 0.9160714 0.9722222    0

Kappa
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
lda       0.4594595 0.6129032 0.6715328 0.6790697 0.7371753 0.9378330    0
rpart     0.4594595 0.6294222 0.7448583 0.7077766 0.7970611 0.8795987    0
glm       0.3048327 0.6243361 0.7843134 0.7295437 0.8163265 1.0000000    0
knn       0.1885246 0.5116055 0.6280481 0.6202616 0.7380667 0.9387755    0
svmRadial 0.6679317 0.8200499 0.8747793 0.8836278 0.9387755 1.0000000    0
svmLinear 0.3307087 0.6505741 0.7419355 0.7236712 0.8092292 0.9387755    0
```

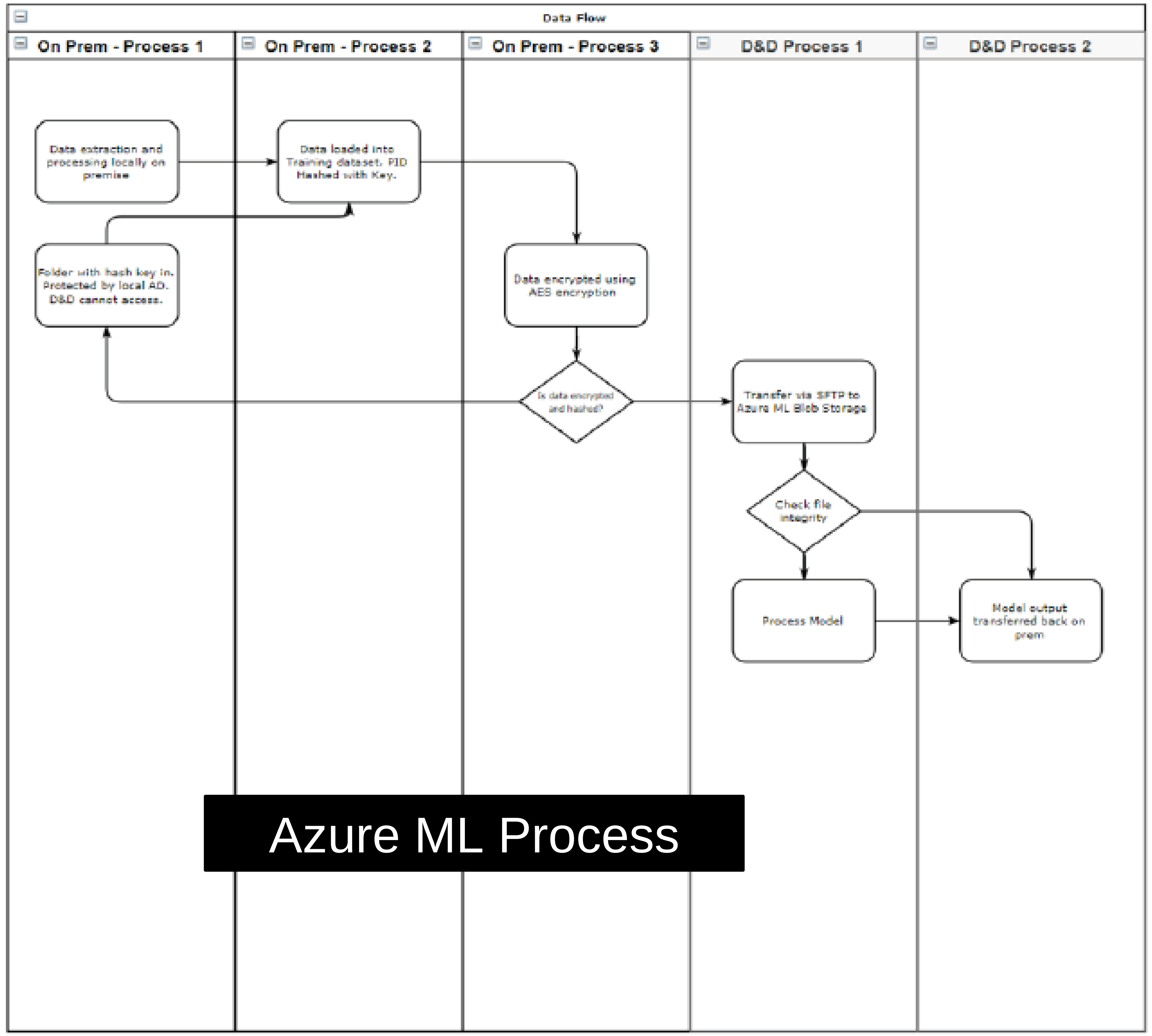
Post results using Ensemble

```
A rf ensemble of 2 base models: lda, rpart, glm, knn, svmRadial, svmLinear

Ensemble results:
Random Forest

1053 samples
   6 predictor
   2 classes: 'bad', 'good'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 947, 947, 948, 948, 947, 949, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
  2     0.9595113  0.9111485
  4     0.9582384  0.9083602
  6     0.9550635  0.9015299

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 2.
```

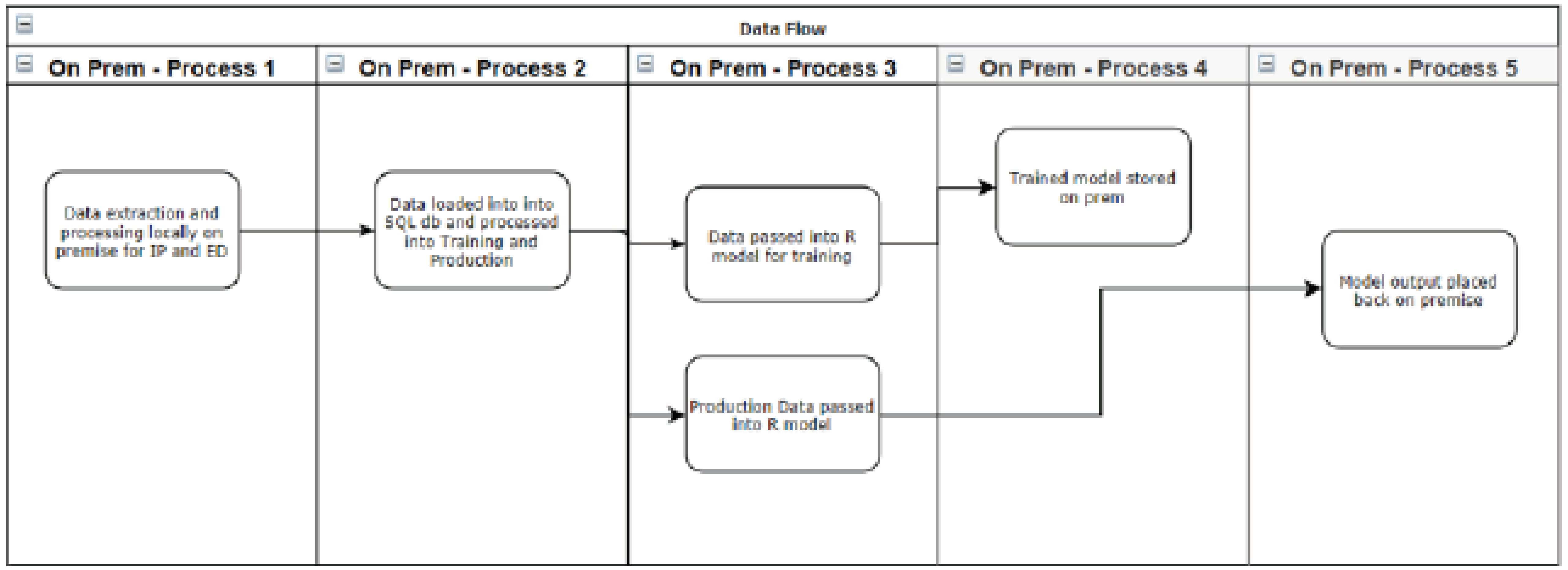
[Visit our website to find out more about our solutions](www.draperanddash.com)

Supervised ML algorithms – tenth phase – deploy the model

- We at D&D have written a blog on how to complete this process: https://www.draperanddash.com/machinelearning/2019/08/deploying-a-trained-supervised-ml-model/
- The majority of our solutions are deployed directly on client sites:



Azure ML Process



Client-Side Process

Thank you

Draper & Dash (D&D)

Contact: gary@draperanddash.com