

Machine Learning for .NET developers

with ML.NET

THE (MOST DIRECT) WAY TO MACHINE LEARNING



Dino Esposito

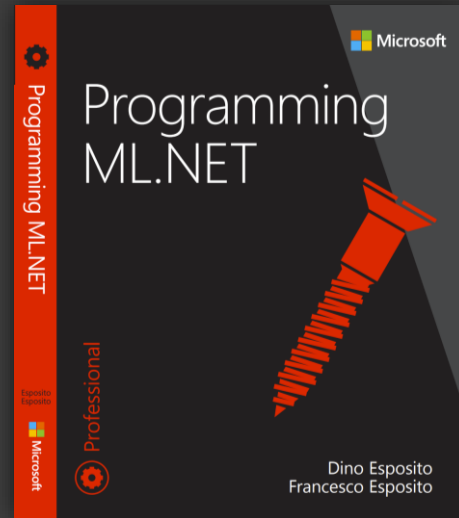
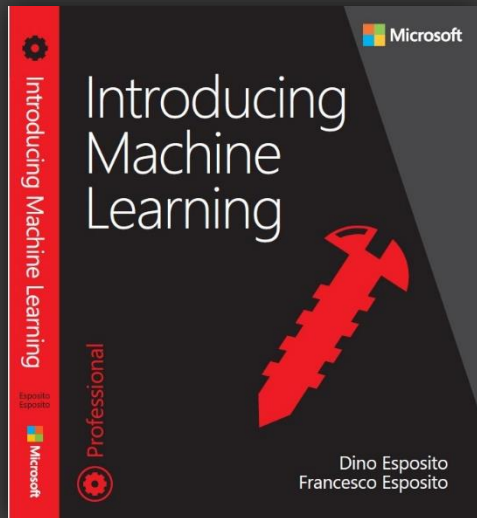
CTO CRIONET.COM

@despos

<https://www.linkedin.com/in/dinoesposito>

<https://youbiquitous.net>





Sport-tech

Scoring

Public API

Operations

TV Production

Data Mining

Healthcare

Support for Digital Therapy

Support for Research Projects

Hospital Ward Operations

Hospital Customer Care

Overview



Planning of ML.NET Projects

Tour of Predefined Tasks

End-to-end Solutions

Interoperability and real-world problems

Planning of ML.NET Projects



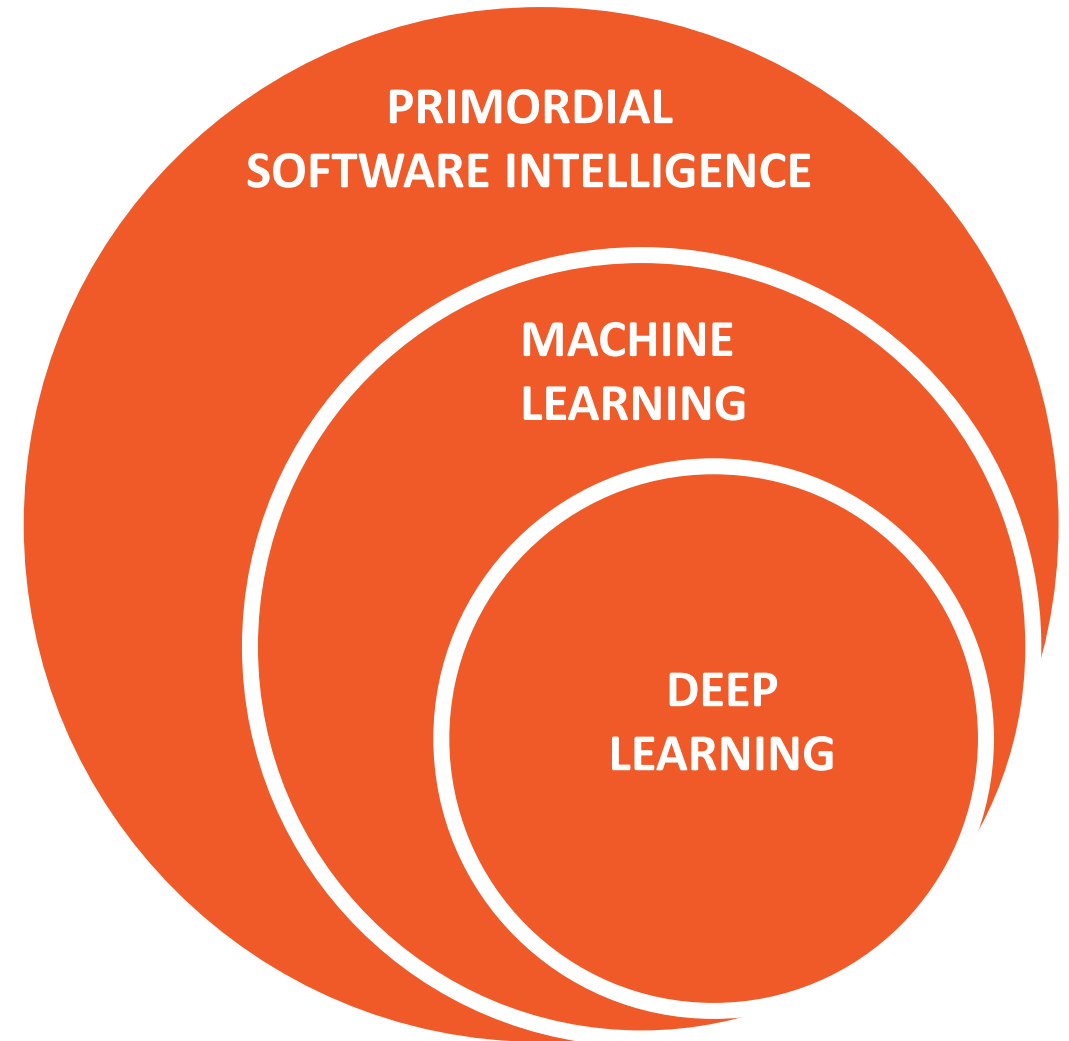
The name **machine learning** was coined in 1959 by Arthur Samuel

A computer program is said to learn with respect to some tasks if its measurable performance improves over time with experience

Supervised learning is when the learning algorithm builds a model from data that contains both inputs and expected outputs

Unsupervised learning is when the learning algorithm builds a model from data which contains only inputs but no clues about outputs

Deep learning is a subset of machine learning in which a network of layers progressively extract information from raw input



Automated Learning

Learning by example

From a list of Q&A, the machine learns how to answer questions it has never seen before.

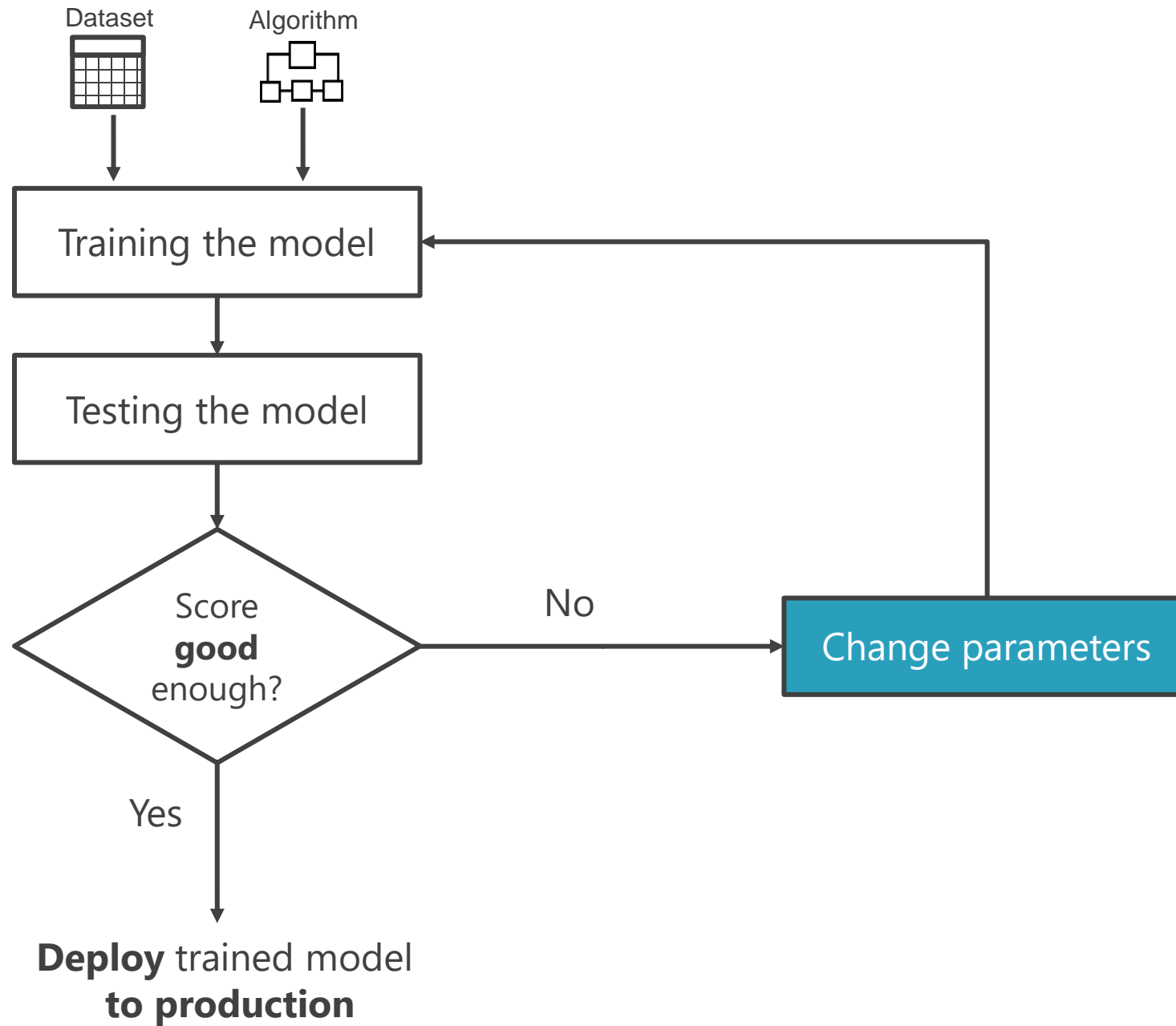
Supervised form of learning.

Learning by discovery

The machine is left alone to explore similarities and classify data items. It receives no guidance.

Unsupervised form of learning.

Machine Learning Workflow



Dataset



Collect raw business data

Possibly organize all in a data warehouse

Build a tailormade dataset for ML ops

Build a data processing pipeline for training/tests

Algorithm



Categorize the problem

- Numerical prediction
- Classification
- Clustering
- Time series prediction
- Other

Configure the selected algorithm

- Hyperparameters

Evaluation



Select the metrics ideal to evaluate the performance of the algorithm

- Metrics are problem-specific
- Multiple numbers to trade off

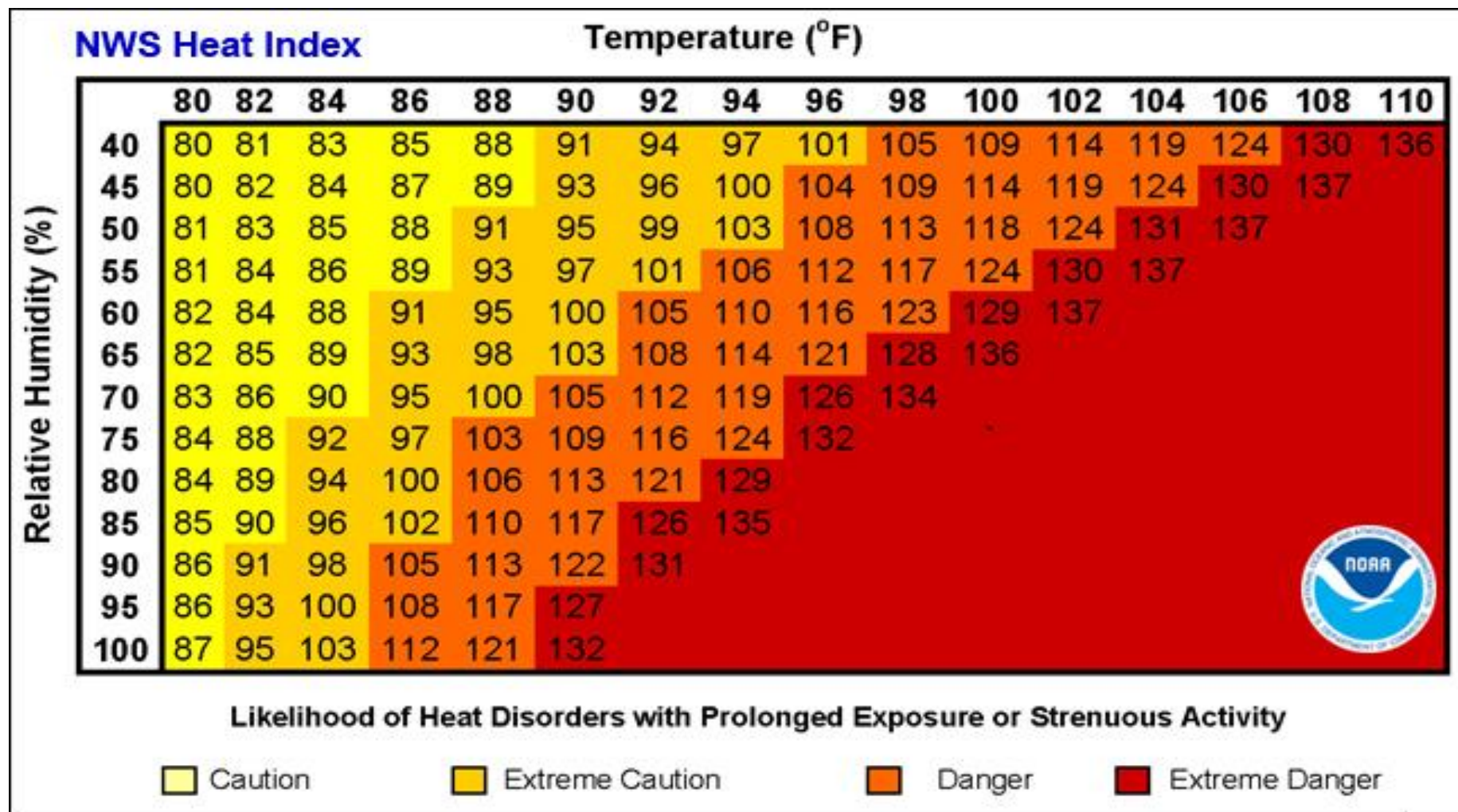
Change



Hyperparameter is a parameter external to the algorithm that influences the actual behavior

Columns in the dataset

Select a different algorithm



$$HeatIndex = m_1T + m_2H + m_3TH + m_4T^2 + m_5H^2 + m_6HT^2 + m_7TH^2 + m_8T^2H^2 + b$$

Coefficient	Value
m_1	2.04901523
m_2	10.14333127
m_3	-0.22475541
m_4	-0.00683783
m_5	0.05481717
m_6	0.00122874
m_7	0.00085282
m_8	-0.00000199
b	-42.379



Problems

Shallow Learning

Classification

Identifying the category an object belongs to

Regression

Predicting a continuous value (given relationships)

Clustering

Grouping objects in homogeneous classes

Time Series

Predicting the next value of a series

Computer Vision

Image and object recognition

Text Analysis

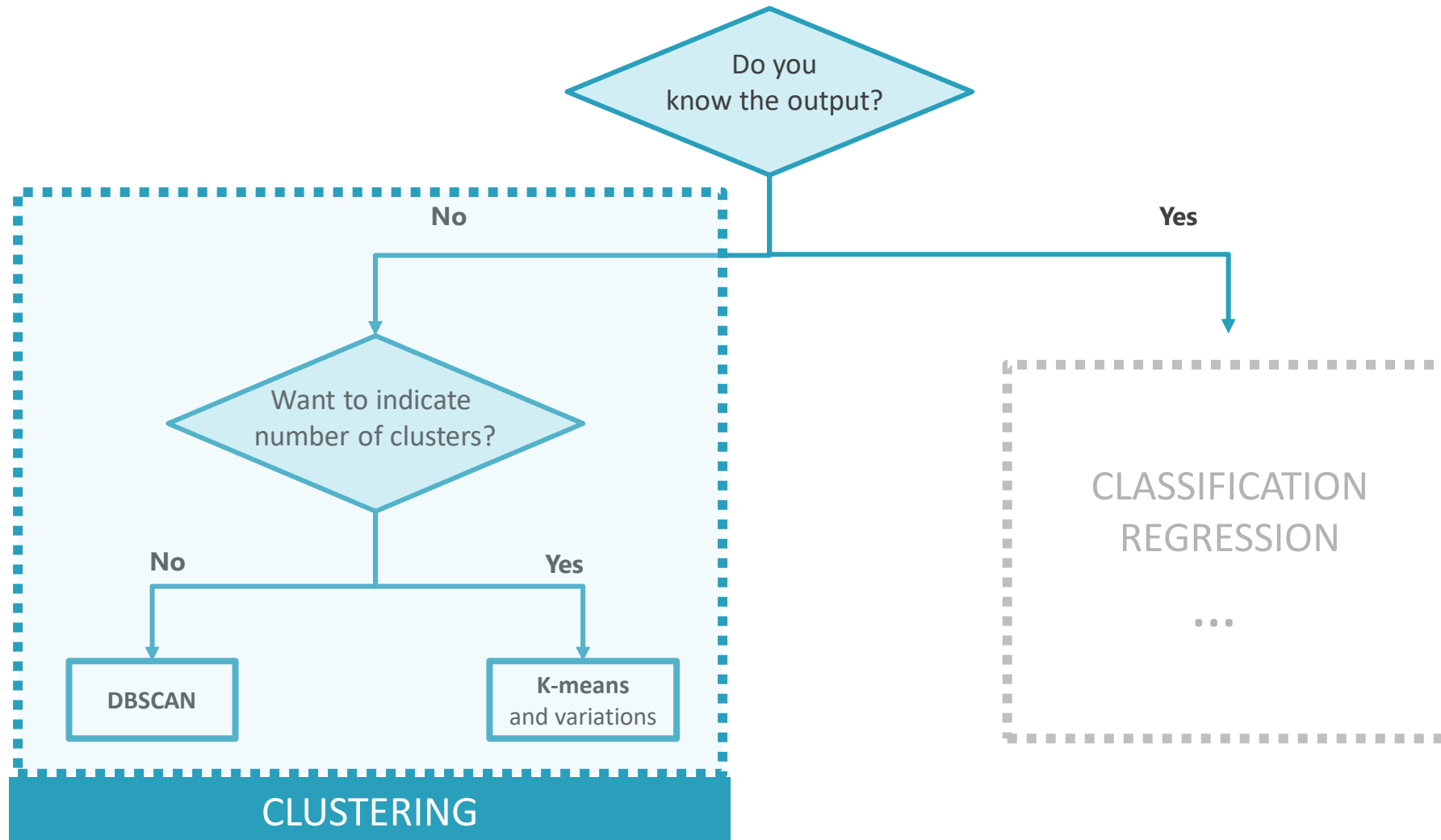
Speech/text recognition and understanding

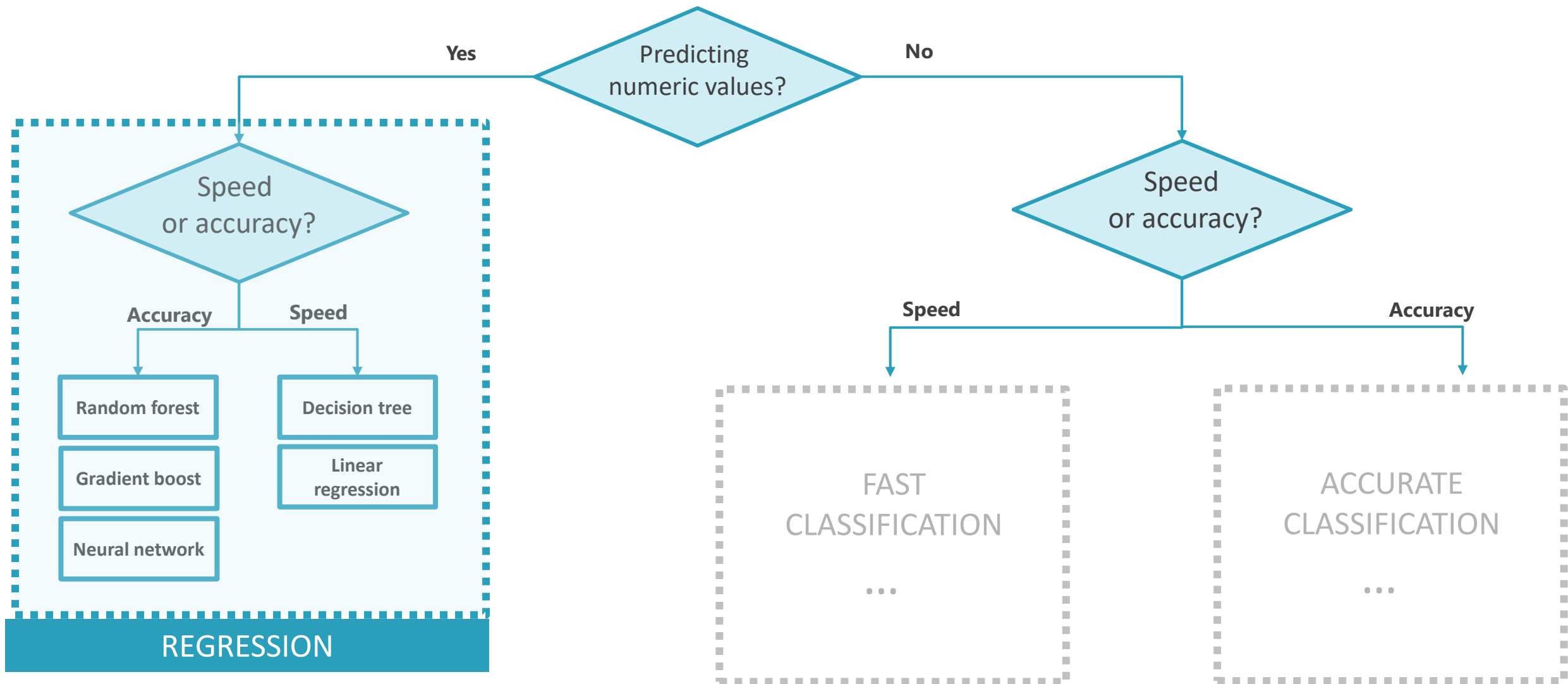
Deep Learning

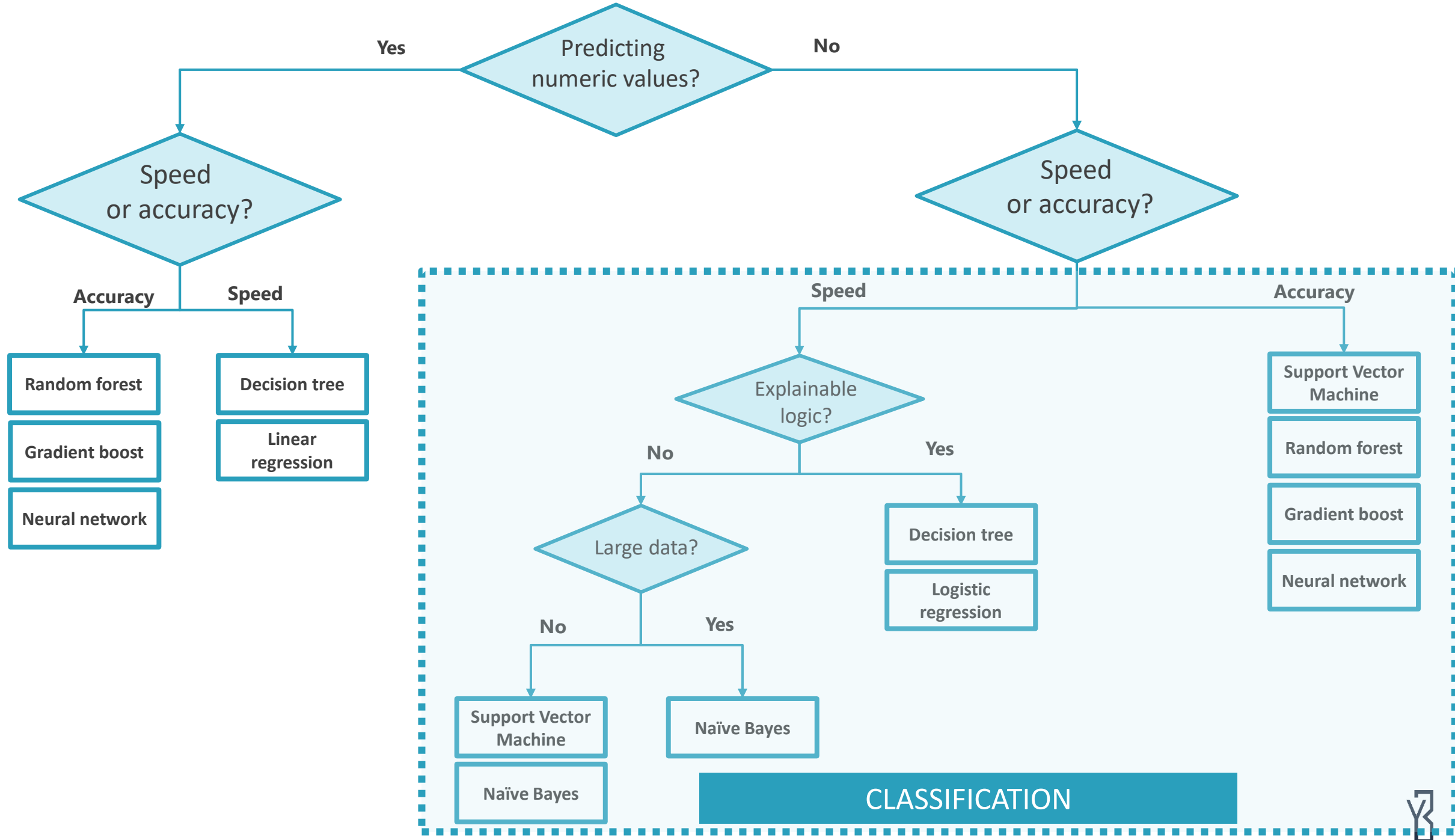


The Algorithm Cheat Sheet

for shallow learning







The Case for Neural Networks

Dealing with time-based data

Each data point is correlated to some of the previous points

When nothing else works

Target output to produce is particularly complex and made of multiple pieces

Doing creative work

Need to create a text or an image or recognize a text or an image



Top choice for machine
learning projects

Why?

Large ecosystem of libraries
and tools

Simplicity of the language

Plenty of learning and
support resources



Choosing Python for machine learning development is primarily a matter of convenience



Python Plus(ses)

Interactive nature of the language

- Quick test of ideas
- Easy to share code with peers
- Human readable
- No compile and build steps

Huge ecosystem of prebuilt libraries and tools

- NumPy
- Pandas
- Scikit-learn
- TensorFlow / Keras / PyTorch
- Deployment infrastructure



Why Looking Beyond Python?

Language issues

- **Interpreted language**
 - Slower than C/C++ and others
- **Not really multithreaded**
 - Implemented via a global mutex
 - Only one thread can hold it
 - Bypass mutex using C extensions!

Memory consumption

- **Unstoppable growth**
 - Fragmentation
- **Automatic string interning**
 - Lots of persistent small strings
 - Impossible to release

What if you load gigabytes of data in-memory for parsing and processing?



Bridging .NET and Machine Learning Communities



Free software machine learning library for the **C#** and **F#** programming languages.

Runs on top of .NET and .NET Core



Trade-offs

.NET Core **performance** getting close to C++

ML.NET will need to **grow** the ecosystem of algorithms and tools

Streaming and **thread**-safety

C# **bindings** to TensorFlow

TensorFlow training with GPU via .NET Core

A large orange circle containing the text "ML.NET" in white, sans-serif font.

ML.NET

A large orange circle containing the text "Python" in white, sans-serif font.

Python



The .NET Perspective

Use the same familiar C# and .NET environment

Performance and tools

No new language to learn

Database facilities

Threading and memory management

Some pure data science aspects hidden from view

In-process hosting of trained models

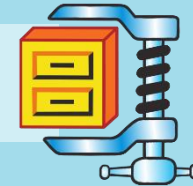
Steps of Machine Learning **with** ML.NET

ML.NET

TRAINING

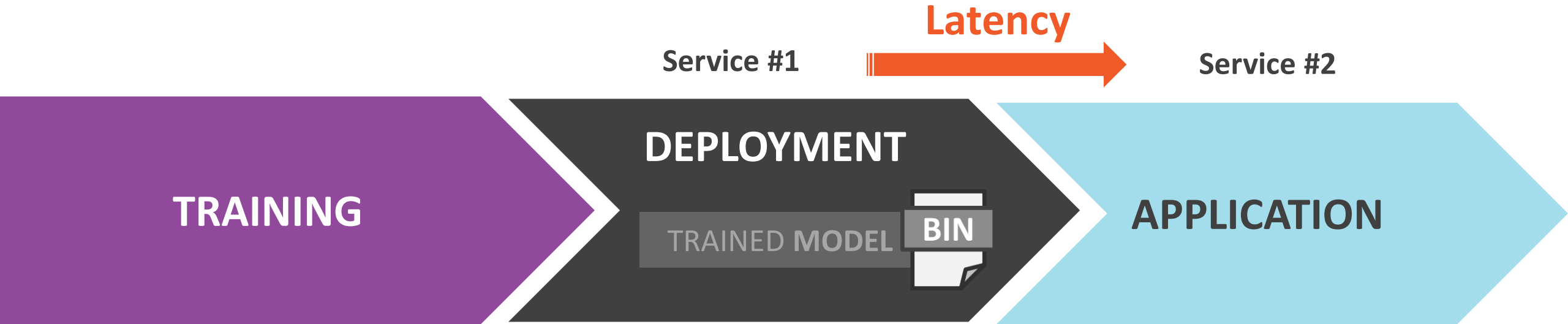
DEPLOYMENT

TRAINED MODEL

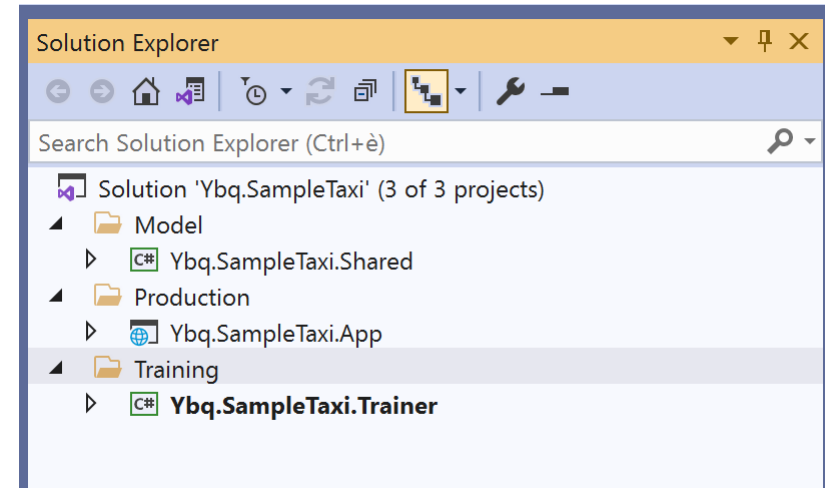


Microsoft Azure

Steps of Machine Learning **without** ML.NET



Demo



Sample ML.NET Project

- Trainer
- Hosting app
- Shared lib

Tour of the Predefined ML.NET Tasks



Pillars of ML.NET Infrastructure

Data Views

IDataView

Immutable object, guarantees access to data to transformers and estimators and provides rather advanced features such as data streaming and memory management. Not a container of data, works on demand and enables thread-safety and reduced memory access.

Transformers

ITransformer

Transforms source input (IDataView) data into new IDataView with a different output schema. Crucial for feature engineering to turn original data into a dataset suitable for training.

Estimators

IEstimator

Algorithms which can be fit on a dataset and produce a transformer which in turn takes a data view and produces a data view



Pipelines

A composition of transformers and estimators form a **pipeline**. The pipeline, or a chain of estimators, begins with a single transformer or estimator and others are appended using the method **Append**.

The pipeline is an **immutable** object. Whenever you append a new estimator, it doesn't append it to the current pipeline instance but it creates and returns a new pipeline object.



ML Task

Shallow learning algorithms only

An ML task groups common machine learning use cases under a common (and familiar) programming pattern.

To build a trained model in ML.NET:

1. First, choose **which of the available tasks** may work for your scenario.
2. Second, cherry-pick the best available algorithm to train the model.

Note that the notion of what is "best" is a moving target and can hardly be determined reasoning on paper without evidence of numbers and errors both in training and production.



ML Task

Shallow learning algorithms only

A task is a catalog with three main endpoints:

1. A list of training algorithms (property *Trainers*)
2. An evaluator to score results of training against the configured error function (method *Evaluate*)
3. A cross-validator tool (method *CrossValidate*)



REGRESSION

Aims at predicting the value of a data item. To some extent, it can be seen as a superclass of forecasting problems.

Algorithm	Method
FastForestRegressionTrainer	Based on the random forest method
FastTreeRegressionTrainer	Based on MART gradient boosting (an ensemble method)
FastTreeTweedieTrainer	Based on the Tweedie compound Poisson model
GamRegressionTrainer	Generalized Additive Model using shallow gradient boosted trees
LbfgsPoissonRegressionTrainer	Based on the Poisson regression method
SdcaRegressionTrainer	Based on the Stochastic Dual Coordinate Ascent method

Choosing the most appropriate method for the specific problem and available data is often a task well beyond the reach of a software person. This is where **data science** skills kick in



Key Steps in a ML Task

```
// Set up the trainer
var trainer = mlContext
    .Regression
    .Trainers
    .Sdca("Label", "Features", lossFunction: new SquaredLoss());

// Append the trainer to the (previously created) data processing pipeline
var trainingPipeline = dataProcessPipeline.Append(trainer);

// Train the model to fit the training dataset
var trainedModel = trainingPipeline.Fit(trainingDataView);
```



Validation Techniques

The quality of a machine learning model depends on how good it can be at working on data it has never seen before. The challenge here is that you can train the model on one sample dataset and sometimes not even a particularly large, well-balanced and highly representative one.

Cross-validation comes in two flavors

Holdout splits the source dataset in two segments: approximately two-third for training and the remaining part (approximately one-third) for testing

K-fold partitions the source dataset in K subsets and runs holdout on each taking the best. Each time one of the K is used for test and other K-1 for training

Iteration 1	Fold 1 TEST	Fold 2	Fold 3	Fold 4	Fold 5
Iteration 2	Fold 1	Fold 2 TEST	Fold 3	Fold 4	Fold 5
Iteration 3	Fold 1	Fold 2	Fold 3 TEST	Fold 4	Fold 5
Iteration 4	Fold 1	Fold 2	Fold 3	Fold 4 TEST	Fold 5
Iteration 5	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5 TEST

Training data

Testing data



Validation should ensure the model best-fits on the data it will be receiving. This requires the availability of enough data and structurally close to data in the real-world.

REGULARIZATION

Intervenes when the team is tempted to add more features to the model in the hope of achieving better results.

Regularization adds a penalty on each new column added to the dataset. Adding a penalty increases the error; so, in the end, it's a matter of adding just the features that bring an inherent value and reduce the error.

PERMUTATION FEATURE IMPORTANCE

Aims at identifying the most impactful features for a particular model and algorithm.

If you scramble the values of a column and still get similar score, then it means that the acted feature is not particularly important in the internal economy of the model. From this standpoint, a low importance feature can be removed without worries.



Normalization and Featurization

A few different types of operations are usable across the entire set of ML tasks.

Feature engineering methods are exposed out of the Transforms catalog object on the ML context.

Adding and removing columns

CopyColumns and
DropColumns

Normalization

Fits all values in the column
in a common interval,
typically 0-1

Binning

Actual values of a column
turned into a number of
reference values (bins).

Key-Value mapping

Maps string to a unique
integer value
CASH=1, CARD=2

One Hot Encoding

Maps distinct value of the
column to a number whose
binary representation has
just a single 1

Hashing

Condense a categorical value
to a number of a fixed size.
ML.NET can normalize
strings, numbers and dates



Missing Values

Any input dataset may have missing value here and there.
ML.NET natively supports only numerical values.

Default value of the type

0 for numbers, NULL for
strings

Mean value

Mean of all values in the
column

Custom

Ad hoc estimators or batch
processes run on the dataset



Large Datasets

- Running short of memory is fairly common in Python when the dataset is huge
 - ML.NET data views efficiently handle high-dimensional data
- Data views work in two ways
 - Load and enumerate data as a classic in-memory collection object
 - Stream data from the original data source via a cursoring mechanism conceptually similar to database cursors and ADO.NET data readers.
- Streaming data is just what most algorithms do during training—transparent to devs
- ML.NET training applications can easily handle huge data sets well beyond the gigabyte order of magnitude and up to 1 terabyte.



Metrics

No metrics is universally valid by itself but all indicators provide insights to an expert eye. It's not realistic to expect all numbers close to their ideal threshold

Squared Loss

The mean of the sum of the squares of the errors (difference between calculated and target values)

Root Squared Loss

The square root of Squared Loss value

Absolute Loss

The mean of the sum of absolute errors (difference between calculated and target values)

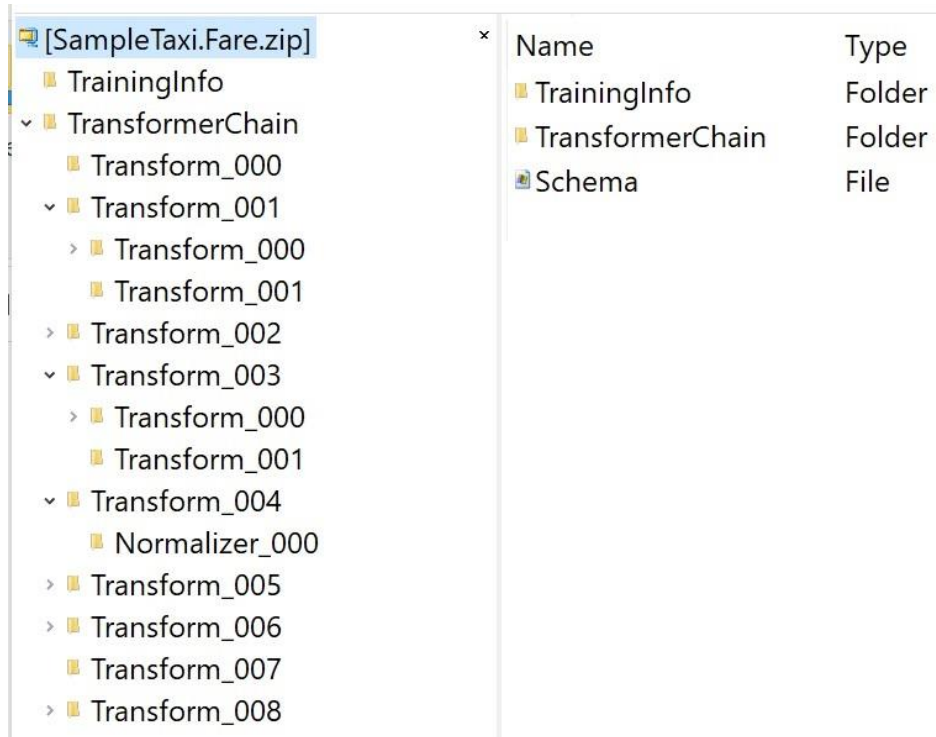
R2 Score

Ratio between the sum of squared errors (expected minus predicted) and the sum of expected minus the mean of expected values

EX: However, having R2 Score close to 1 is not sufficient to guarantee high quality but having it close to 0 is a clear indicator that something doesn't work.



The Trained Model



Name	Type
TrainingInfo	Folder
TransformerChain	Folder
Schema	File

- Saved model is a ZIP file
- Contains the list of necessary transformers and the schema of data
- Serialized in a proprietary way
- Loading of the model
 - In-process (much faster)
 - Embedded in a Web Service (or gRPC shell) to be consumable from other, non-.NET, platforms
- Exportable to ONNX

Invoking a Trained Model

```
// Get the trained model reference
var model = mlContext.Model.Load(path, out var inputSchema);
    .Regression
    .Trainers
    .Sdca("Label", "Features", lossFunction: new SquaredLoss());

// Set up the prediction engine
var predEngine = mlContext
    .Model
    .CreatePredictionEngine<TaxiTrip, FarePrediction>(model);

// Perform the prediction
var prediction = predEngine.Predict(trip);    // TaxiTrip type
```



Tips for Trained Models

- Make the trained model a singleton
 - Load it only once instead of once per request
- Method *CreatePredictionEngine* is relatively time consuming and calling it on every request may impact the overall performance
- Worse yet, the returned type *PredictionEngine* , is not thread-safe so the singleton shortcut is not an option here
- Built-in object pooling for ASP.NET Core clients



ML Devil's Advocate

REGRESSION

The task of predicting a continuous value, whether a quantity, a price or a temperature.

- ◀ **Linear regression** is great for quick and dirty predictions such as estimation of time and cost of a taxi ride that, all in all, would have a very limited impact on people and business
- ◀ Predicting price of houses is different story
 - Just giving a free quote of an apartment?
 - Long-term prediction about the variation of prices in a given area?

Linear regression is often used to demonstrate the need of a deeper, non-linear approach based on neural networks.

A cascading approach—a pipeline of models—may be necessary as in prediction of energy price where deep knowledge of the business dynamics is crucial.



CLASSIFICATION

The act of systematically arranging objects in homogeneous groups according to a number of established criteria. The number of groups (2 or more) gives the problem different connotations.

Each dataset item belongs to a class; the number of classes are fixed and known in advance. This sets a key different between **MultiClass** and MultiLabel and Clustering problems.

Algorithm	Method
LbfgsMaximumEntropy	Based on maximum entropy model, a generalization of logistic regression
NaiveBayes	Based on the Naïve Bayes probabilistic classifier model
OneVersusAll	Based on the One-Versus-All model
PairwiseCoupling	Based on the One-Versus-One model
SdcaMaximumEntropy	Based on a linear model that returns probabilities features belong to a class

Trainers

No metrics is universally valid by itself but all indicators provide insights to an expert eye. It's not realistic to expect all numbers close to their ideal threshold

Naïve Bayes

Probability of inclusion in a class; naïve because it assumes features are independent

One-Versus-All

Multiple instances of binary classification: y/n prediction for each class and takes the highest likelihood

One-Versus-One

Runs a bin-classifier for each pair of target classes and selects the class the wins the most matches

OVO triggers more bin-classifiers than OVA, but each classifier works on a smaller dataset that only comprises the rows having any of the two classes as the target value. OVA, instead, requires all of its fewer bin-classifiers to always work on the entire dataset.

The trade-off is given by the actual trainer used by the binary classifiers. A binary classifier such as SVM doesn't scale well with the number of rows. Therefore, if SVM is selected as the binary classifier then the OVO method is preferable over OVA regardless of the higher computational complexity.



Metrics

The SDCA algorithm combines several of the best properties and capabilities of logistic regression and SVM algorithms and in most cases represents a very good (first) fit for a multiclass problem

Confusion Matrix

Combines together predictions and labels on the rows and columns of a square matrix

		Classes		
		Green	Orange	Red
Predictions	Green	5	2	0
	Orange	3	3	2
	Red	0	1	11

The values in the columns (e.g., Green) indicate how many times elements in the class have been predicted as any of the values on the rows. Those numbers are used to calculate **Precision** and **Recall** for each class.

BINARY CLASSIFICATION

Any time there's a yes/no, binary answer to give, that's an instance of binary classification problem.

A real-world example is whether a given email should be classified as spam, a financial transaction flagged as suspicious (whatever that means in the given context).

Often a binary classification problem can be reduced to a simplified version of linear regression problem in which all values below or over a given threshold are mapped to one of the binary classes

Algorithm	Method
SdcaLogisticRegression	Based on the calibrated Stochastic Dual Coordinate Ascent (SDCA) method
LbfgsLogisticRegression	Based on the linear logistic regression strategy
LinearSvm	Based on a Support Vector Machine approach plus a special descent strategy
SgdCalibrated	Based on the Stochastic Gradient Descent (SGD) method
SgdNonCalibrated	Based on non-calibrated Stochastic Gradient Descent (SGD) method

Calibration

In classification, getting a direct (binary) answer on whether an object belong to a certain class may or may not be ideal. Sometimes it is more convenient to obtain the probability that the object belongs to any available class. Models with this characteristic are referred to as calibrated.

Calibration can be obtained through a number of methods

Isotonic

Uses a monotonic logic according to which objects with higher predicted scores are more likely to be positive

Naïve

Turns continuous values into categorical values by grouping all values in a range into a dedicated bin.

Platt

Based on the Platt's scaling method that applies logistic regression model to scores

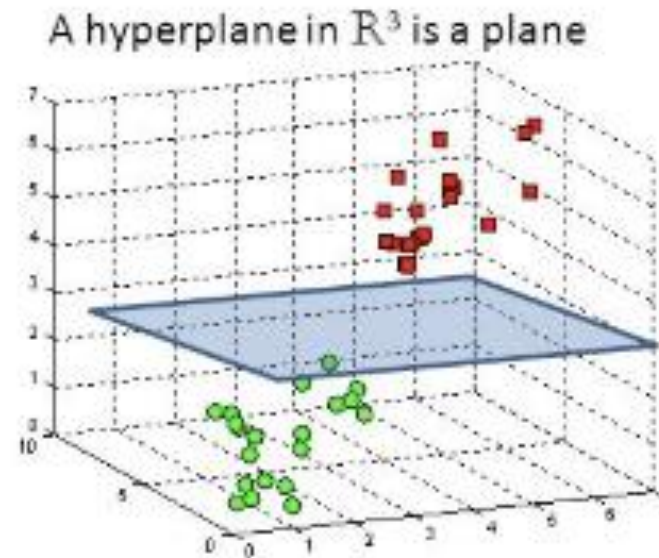
Support Vector Machine

Supervised algorithm with a proven success record on both **classification** and **regression** problems.

It shines at text classification, spam detection and sentiment analysis. It also performs well when used on images to recognize regular patterns such as handwritten notes or digits. SVM usually delivers accurate responses even when trained on relatively small datasets, as long as the data has limited overlapping.

SVM vs. Logistic regression

- Nearly the same performance
- Same accuracy on similar datasets
- Both unaffected by outliers
- Both are linear and can be trained well even on fairly large datasets



Logistic regression returns the likelihood that a data item falls in the default class.

SVM looks for the widest separating margin between data items that fall in each class

Metrics

- **Accuracy**

- Percentage of items classified correctly within the entire test set. Ideal value is close to 100%

- **Precision**

- Percentage of items in a class pos/neg that were effectively so. Ideal value is close to 100%.

- **Recall**

- Percentage of items in a class pos/neg versus the number of pos/neg in the dataset. Ideal value is close to 100%.

- **F1 Score**

- Harmonic mean of precision and recall. Ideal value is close to 100%.



ML Devil's Advocate

CLASSIFICATION

Refers to a modeling problem where given some sample data you predict the category it belongs to.

- ◀ There is no known theory that explains how to map algorithms and classification problems. Start with experiments and discover which algorithm and related configuration leads to the most acceptable performance for the problem at hand
- ◀ One thing is determining whether an email is spam; quite another is classifying an image or a video frame. The underlying data (and its internal intricacy) is quite different and makes for radically different approaches.
- ◀ A clear difference of learning power exists between shallow and deep algorithms
- ◀ Sentiment analysis is a delicate matter and a too sharp response may be even pointless. A neural network may be a savvier option than a classifier when the accuracy of the response is critical for the decision taken based on the response



CLUSTERING

Clustering algorithms do a kind of creative work as they autonomously decide how to split data rows in the specified number of groups. Rows fit into a given cluster based on the relationships the algorithm has detected with other data rows.

There are two classes of unsupervised algorithms: those that need to receive the number of clusters and those able to figure out the ideal number of groups. It is crucial to remark that returned clusters always form a partition meaning that the entire dataset is covered and each element belongs to exactly one cluster.

K-MEANS

Iteratively moves data points across K assigned clusters to ensure that all rows in each cluster fit uniformly around a center point.

Requires features expressed through continuous float values

K-MODES

Uses the Euclidean distance and supports categorical values. Also, it uses the mode rather than the mean. (The mean is the average of a set of values; the mode is the most common number in a set of values)

DBSCAN

Density-based groups together data points that lie in a neighborhood defined by a distance. The control that K exercises in K-Means, is exercised by the minimum number of points each cluster is required to have.

Initialization

- How to choose the initial centroids in K-Means?

KMeansPlusPlus

Default, refers to KM++ proposed in 2007 and considered the most reliable delivering a solution that is at most only $O(\log K)$ worse than the optimal solution

KMeansYingyan

2015, significant performance gain due to the initial clustering of centroids that makes unnecessary a good deal of subsequent distance computations. (From MS Research)

Just random

Initial centroids are selected randomly. This might lead to potentially bad approximations with respect to the optimal clustering



ML Devil's Advocate

CLUSTERING

Partitioning unknown data into close groups leaving the team to figure out why (business-wise) are those data points close.

- ◀ Unsupervised learning
- ◀ Not really producing a model, but simply a set of clusters.
- ◀ ML.NET requires you write ad hoc code to extract clusters in the manageable form of .NET collections
- ◀ From a business perspective, clustering is in a way closer to data preparation than it is a prediction for a specific problem
- ◀ It is mostly the means, and rarely the end, of a machine learning task. Although clustering may belong to any real-world machine learning pipeline, it is often only the **first step** of a longer workflow.



**ANOMALY
DETECTION**

FORECASTING

A time series is a sequence of values captured at successive, ideally equidistant points in time, a discrete (as opposed to continuous) collection of values.

The values of wind speed reported every thirty seconds by the anemometer installed on a specific wind turbine in a specific farm is a good example of a time series.

(Moving) Average

Just take the average of latest data points, possibly weighing more most recent N points.

Latest value (last season)

Just take the latest value (common in financial apps) or in case of seasonal data and large dataset the value the same time of last season (ex: year)

SSA

Single Spectrum Analysis algorithm encodes the time series into a list of trends and key change points and from there rebuilds a new algorithmic series of values.

Key to forecasting algorithms is decomposing the time series in chunks to identify **trends** and **seasons**



Trends and Seasons

- Trend

- Long-term, structural change of direction of the observed values. When it happens, values increase or decrease in a way that can be linear or nonlinear. The point in time when the change of direction begins is referred to as a **change point**.

- Cycles

- A business-related period that encompasses multiple trends, usually no longer than 2 years

- Season

- Periodic fluctuation of values such as sales spot higher in holiday season or electricity consumption low during the night hours

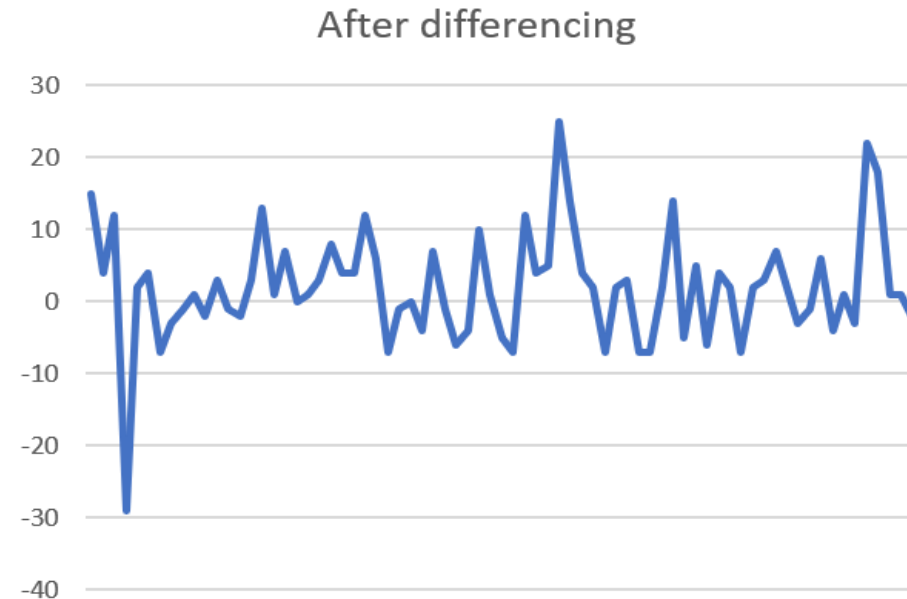
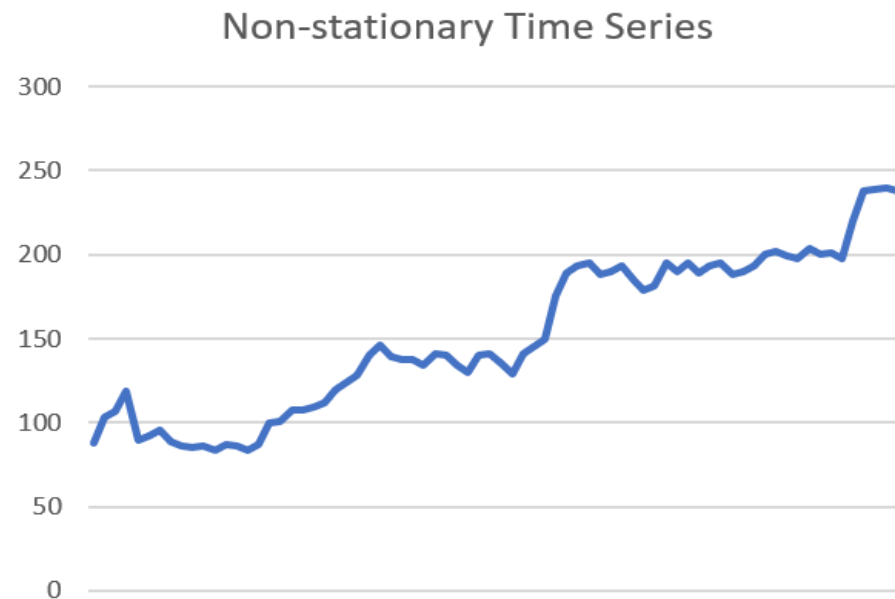
- Seasonality is a special type of cyclic behavior

- Seasons are periodic, repeatable and predictable, cycles are irregular and unpredictable in length
- A season is a cycle that repeats time after time



Stationarity

- Stationary time series are much easier to predict because values do not depend on the time they've been captured
- Time series with trends and cycles are non-stationary
- Dedicated algorithms (differencing) can turn non-stationary into stationary

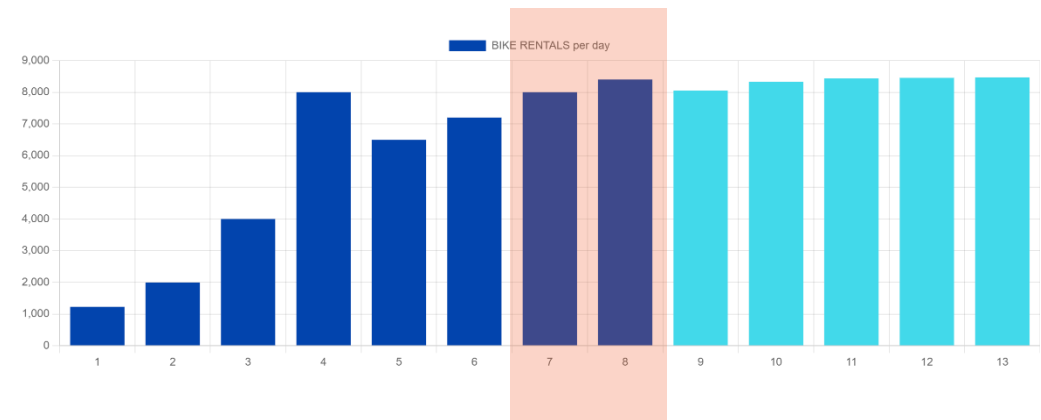
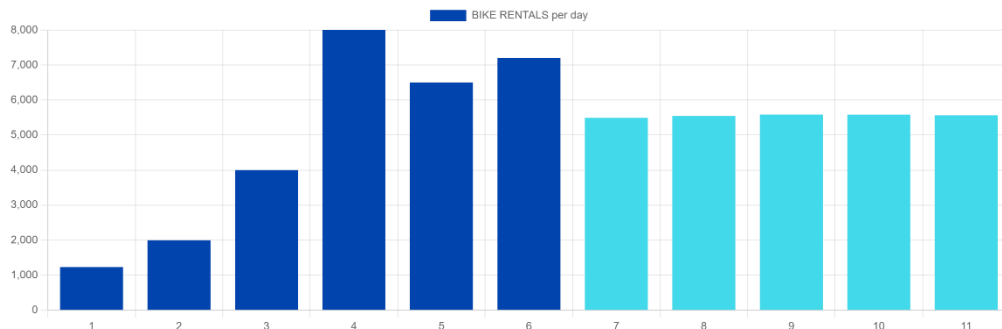


Differences
between 2
consecutive
points



Checkpoints

- By default, forecasts are always based on the time series used for training
 - Say your data end in 2019, no matter the day you ask any prediction will **always** be based on 2019
 - Forecasts, instead, make particularly sense in a continuous flow of data
- Must contribute new observations to the system, make the time series longer and get predictions based on latest entries
 - Enter checkpoints and discount factor (the $[0,1]$ weight of appended observations)
- In forecasting, checkpoint refers to updating the model with a new observation



ML Devil's Advocate

FORECASTING

Time series are representative of a continuous flow of data and for predictions incoming data points are relevant as much as input features.

- ◀ A forecasting model needs some sort of state to be kept and updated over time.
- ◀ Imagine asking '*Given these most recent values, and your knowledge of the business gained during training, what we could expect for the next N days?*'
- ◀ We should find a way to update the model so that it incorporates most recent observations in the internal state and is ready to use it for future predictions
- ◀ Multivariate SSA, forecasting on multiple columns
- ◀ Real-world forms of prediction require custom approach (EX: price and production forecasts)





RECOMMENDATION

A recommender system collects data from many users to guess the preferences of each. A recommender won't receive any direct input from the user. Recommendations are preferably sparse (fairly different from each other).

A ranking system measures the relevance of documents in an information retrieval system of any kind. A ranker receives feedback from the user (Google-style queries). Results are expected to be similar.

The Matrix Factorization Algorithm

1. Gets **a user and a movie** and looks for other users in the dataset that rated the same (different) movies the given user rated.
2. If any of the users with similar preferences rated the movie, then an average of the available ratings is taken.



The Matrix Factorization Algorithm

MF algorithms became popular in the early 2000s during the Netflix prize challenge about predicting how much someone would have enjoyed watching a given movie.

Prize was awarded to a combined team from various institutes of research globally known as BellKor's Pragmatic Chaos team.

The proposed algorithm is a **rather sophisticated variation** of a classic MF algorithm and uses an **Ensemble** approach to train multiple MF simultaneously (and then applies a sophisticated non-linear function to make the final decision).

	A	B	C
1	userId	movieId	rating
2	1	1	4
3	1	3	4
4	1	6	4
5	1	47	5
6	1	50	5
7	1	70	3
8	1	101	5
9	1	110	4
10	1	151	5
11	1	157	5
12	1	163	5
13	1	216	5
14	1	223	3
15	1	231	5

Data ETL-ed from some
production system

Issues of the Matrix Factorization Algorithm

1. Gets **a user and a movie** and looks for other users in the dataset that rated the same (different) movies the given user rated.
 2. If any of the users with similar preferences rated the movie, then an average of the available ratings is taken.
- What if there are **no ratings whatsoever** (say, new/inactive user and/or new/unpopular movie)?
 - Not much different from tossing a coin. Yet, it can be acceptable for a media platform or a web site. It's a mere suggestion in the end!
 - If you want to (try to) be as accurate as possible, an ad hoc algorithm must be arranged that likely connects together various pieces in cascade.
 - It's not just about training an algorithm but building, testing, and finally training, a learning machine.



Collaborative Filtering

- A technique that can filter out items that a user might like on the basis of reactions by similar users. It works by searching a large group of people and finding a smaller set of users with tastes similar to a particular use.
- Addresses limitations of basic recommenders
- The key difference between ranking and recommendation lies in the fact that ranking is global whereas recommendation is mostly personal.
 - **Ranking** tends to address ratings aggregated over large populations of users and produces a kind of general-purpose rating.
 - **Recommendation** tends to override the default ranking for the specifically recognized preferences and intents of each individual user.



KNN

- Unsupervised that applies a distance to the data points represented by the input (e.g., movie) and selects the top K points in the cluster where the input data item lies
- The effectiveness of a KNN method is strictly dependent on the effectiveness of the selected distance function.
 - Euclidean distance not optimal in recommenders with high degree of sparseness
 - Cosine distance may be preferable sometimes or something different as the actual quality of data may suggest
- The challenge is improving KNN in a way also scales well with very large datasets as those commonly involved with recommenders.



ML Devil's Advocate

RECOMMENDATION

The recommender estimates the rating the user would assign to an item (e.g., movie) and labels it as “you will like it” if the rating exceeds a fixed threshold.

- ◀ Matrix factorization is likely the most effective algorithm for collaborative filtering
- ◀ Building up a recommender system is not an easy task
- ◀ It may be so complex at times that often a rather naïve solution is reckoned more than acceptable. In the end, it's all about how much accurate you need to be in your specific context.
- ◀ The Netflix challenge of 2006



More ML Task

- Anomaly Detection
- ImageClassification



End-to-end Solutions with ML.NET



From incorrect (or inadequate or poorly relevant) data only stems
incorrect (or inadequate or poorly relevant) answers

Data in Machine Learning

- Large quantity of reliable data
- Reliability of data measured against qualitative and quantitative parameters
- Feature engineering
 - Computed and/or concatenated columns
 - Normalized columns
 - Filling in blanks
 - Removal of peaks



Data Integrity



Data Completeness

- Contains all attributes that are strictly required for human or machine comprehension
- You can miss some data (e.g., contact info) as long as rest of it is enough
- No gaps between what was supposed to be collected and what's actually been collected



Data Uniqueness

- No duplicates of it exist
- Depending upon the business, duplicates may or may not be an issue
- Poor data uniqueness is an issue if, for example, it could lead to skewed results and inaccuracies



Data Timeliness

- Distribution of data items within an acceptable timeframe
- Definition of an “acceptable timeframe” is context-specific
 - Refers to the duration of the time frame and the frequency of data points
 - Sometimes, a 10 minutes timeline is good; sometimes not
- Duration indicates the time interval for which data collection should occur to generate satisfactory results



Data Accuracy

- A data item is accurate if it correctly describes the state of the observed real-world item
- Business requirements set the specifications of what would be a valid range of values for any expected data item
- When inaccuracies are detected, some policies should be applied to minimize the impact on decisions
- Common practice is replacing out-of-range values with a default value or with the arithmetic mean of values detected in a realistic interval



Data Consistency

- A data item is consistent if its values are coherent with those of other data items and not in patent contradiction with some
- An example of data inconsistency is a negative value that appear in a data item when negative values are not likely in the range of expected values
- Actual definition of data consistency is highly influenced by business requirements



Harmonization Techniques

- Uniform representation of data
 - **EX:** if data contains monetary values, then you must ensure all values are based on the same scale and currency. Same for country names, dates, temp, measures
- Range normalization
 - Numeric values that fall in different ranges must be normalize to a common scale
 - Changes should not skew the ratio between numbers



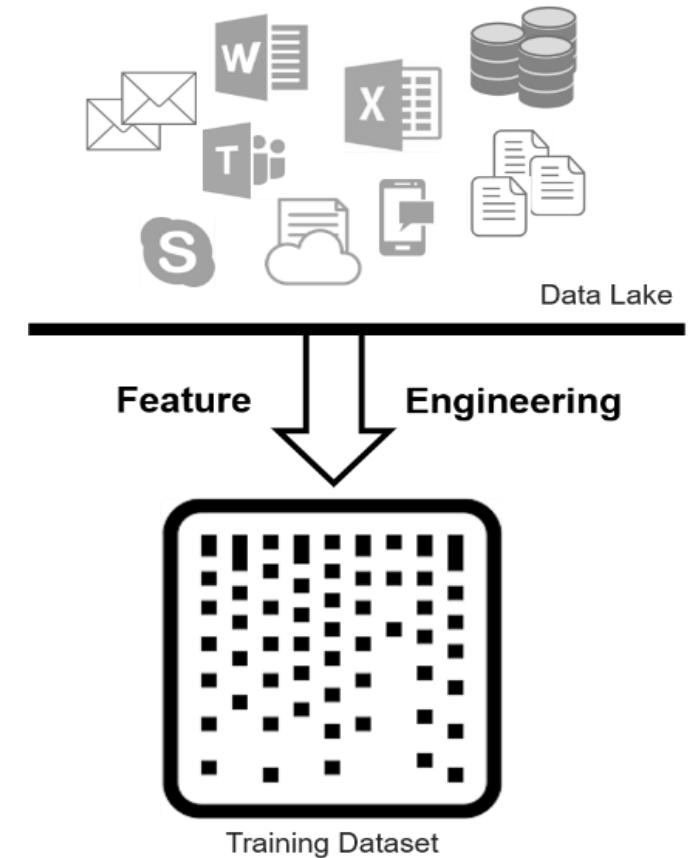
Harmonization Techniques

- Deduping
 - Merging data items that, although appearing distinct in the dataset, are in practice known (by domain experts) to refer to the same item
- Outlier removal
 - Values below and beyond some given thresholds are delicate and how you actually handle them depends on the scenario
 - Rich dataset → remove rows with at least one outlier value in some columns
 - Limited dataset → replace outlier values with mean of the column
 - If those outlier values are relevant business values then your dataset is imbalanced with too few relevant rows



Feature Engineering

- Feature generation
- Feature selection
- Feature extraction
- Generating a tabular representation of data that incorporates the content of the data source



Feature Generation

- Raw and unstructured data found in CSV files, relational and non-relational databases, textual documents and more is organized in a tabular format
- Domain knowledge suggests what could be a reasonable featurization of the raw data



Feature Selection

- Getting rid of any columns of data that is glaringly irrelevant for the purpose
- Data science provides a few techniques to evaluate the relevance of a feature algorithmically
 - If two features look particularly correlated, then the data scientist may decide that only one of the two features is needed and drop the other
 - If one feature looks poorly related to the output variable then it can be dropped
 - If the values of a feature change little and always in a short range then the feature can be dropped
- If after these changes the function still fails at predicting on live data, then probably a deeper refactoring of the dataset is necessary



Feature Extraction

- Starts from an initial set of data and builds derived columns to facilitate the learning steps
- If two (or more) features hold values but a plain combination of them is equally helpful, then a new computed feature may replace them all
- If one column takes categorical values, an option is reducing the number of options by grouping more values into a new category
- Dimensionality reduction to make it fast and effective
 - Occam's Razor 😊



OK... data is ready

- Training
- Testing
- Hosting



Anomaly Detection

Refers to the steps necessary to identify unusual occurrences of items in a possibly (very) large amount of data

On one hand

Anomaly Detection is the umbrella that covers a number of statistical techniques to spot anomalous values in a column of data

On the other hand

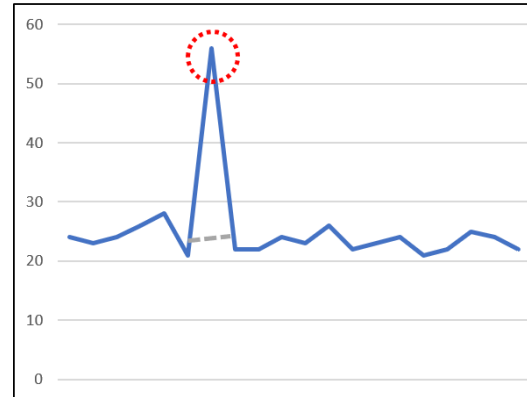
in the collective imaginary it refers to concrete business problems such as fraud detection, predictive maintenance, unusual activities like bots and cyber-attacks

Being aware of these two levels of abstractions is crucial to set the ground for effective solutions to effective business problems.

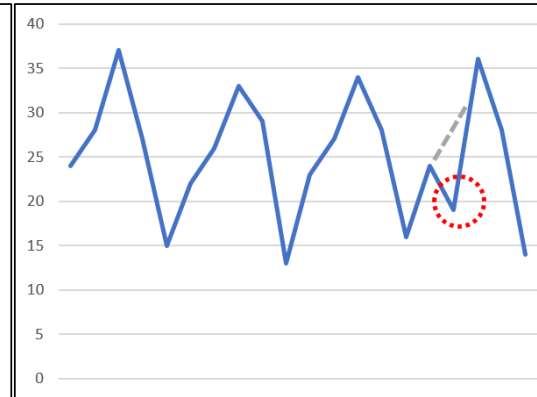


Anomalies (or outliers)

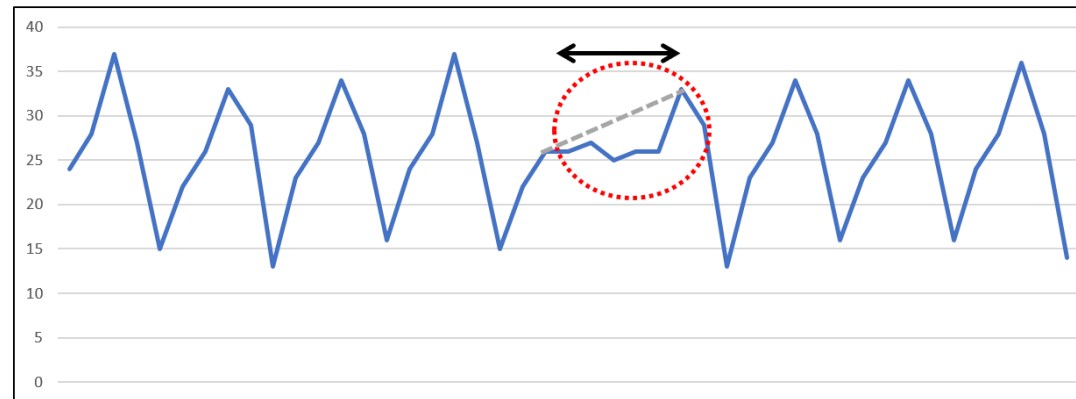
POINT OUTLIER



CONTEXTUAL OUTLIER



COLLECTIVE OUTLIER



Anomaly Detection in ML.NET

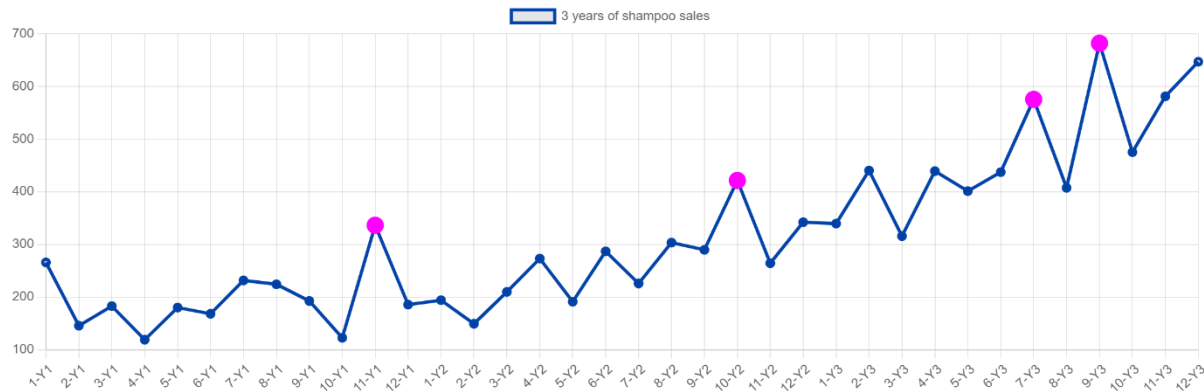
Spikes

(Point anomalies and occasional burst)

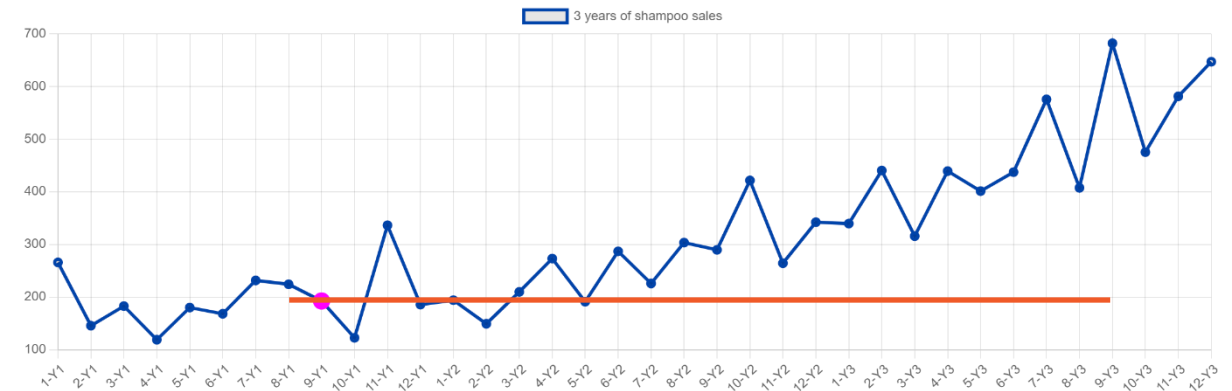
Change Points

(persistent change in the flow and beginning of a new trend of data)

Pass a time series and get a list of spikes or change points. No really trained models.



Spikes



Change points



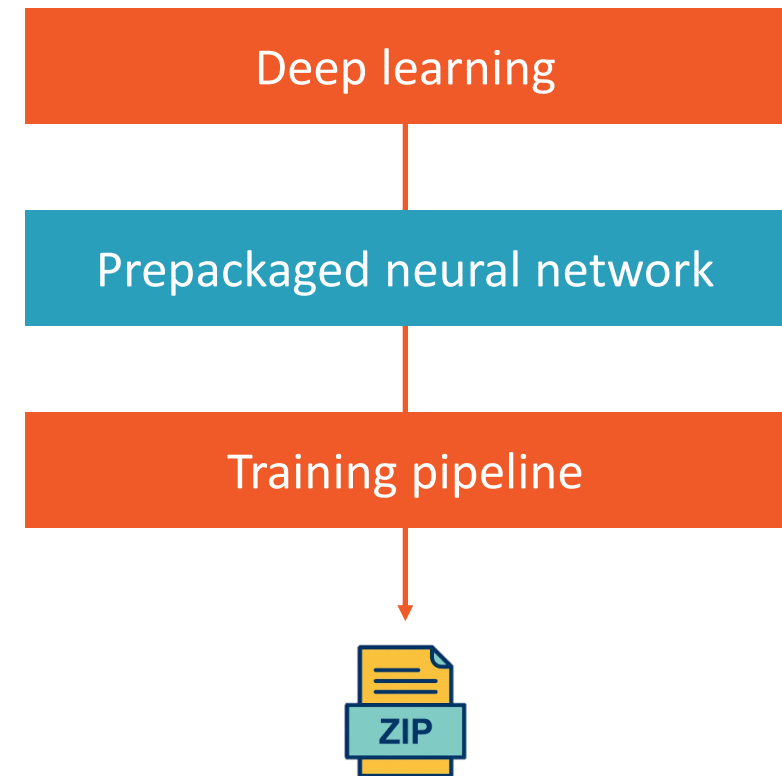
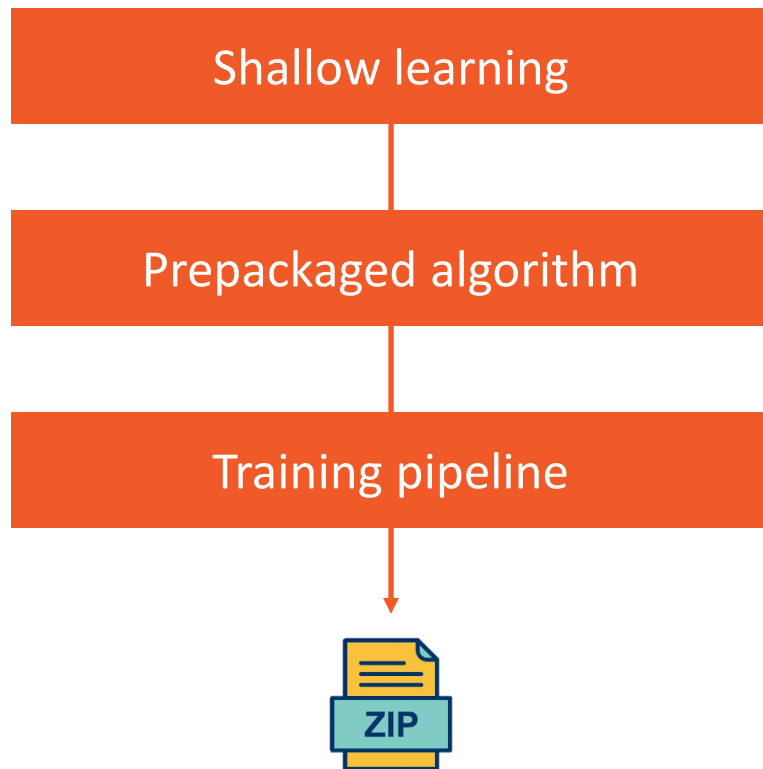
Demo



Detecting spikes and change points

Image Classification

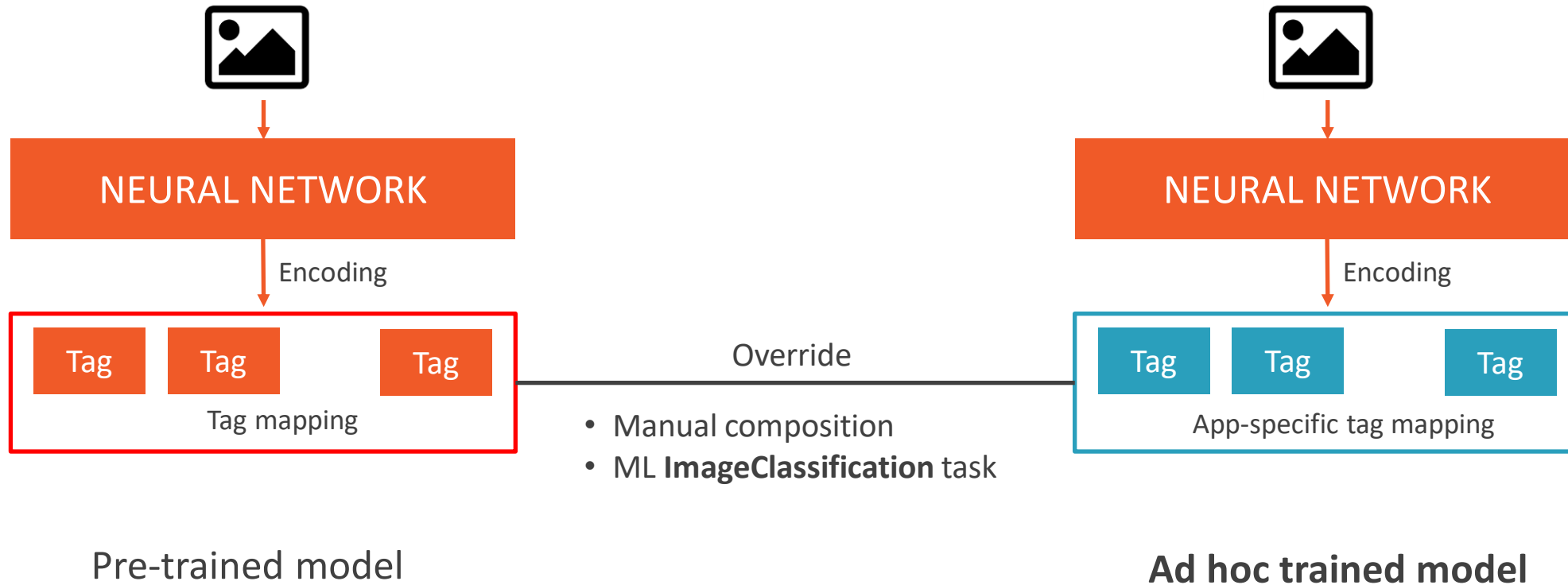
Realistically, no team can start from scratch. It's a relevant computational cost that no individual and not even the majority of teams can easily pay.



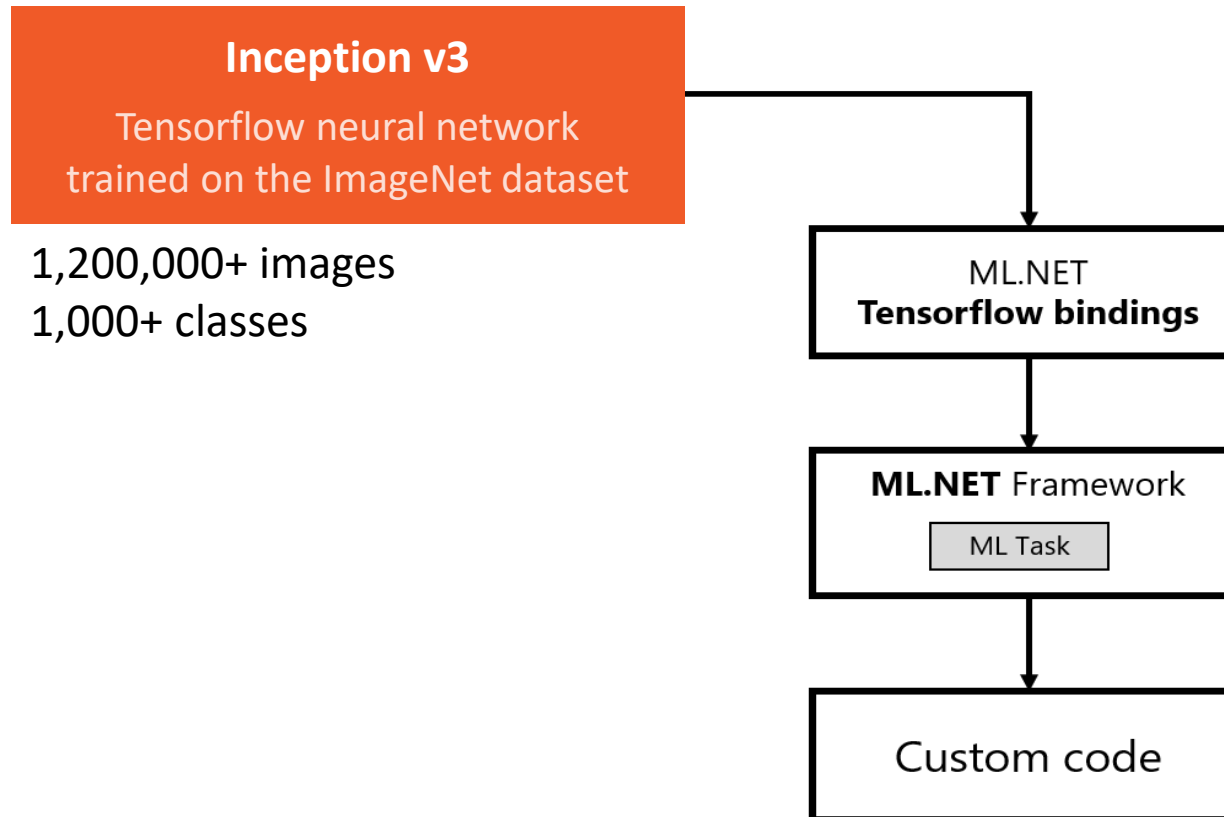
Transfer Learning

Pre-trained models retrained on purpose

Leverage an existing core neural network trained to do a number of image-related chores and then specialized to recognize and classify just the images of a specific application

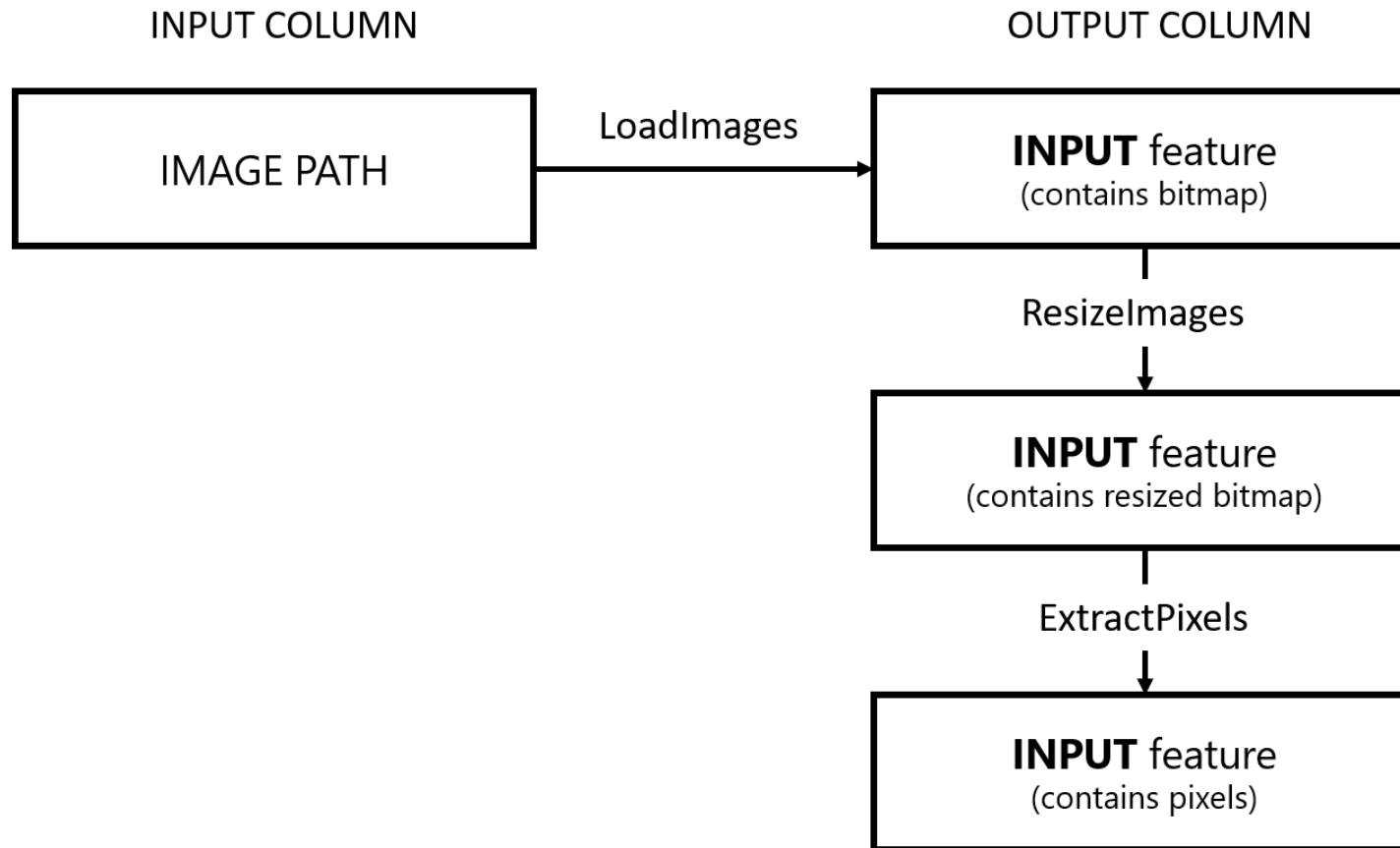


ML.NET Image Classification



Getting Ready for Inception

Mapping image classification to an instance of canonical multiclass classification problem



Ad hoc estimators to make
app-specific training images
available in a format that
Inception can understand

Start of the pipeline

```
// Categorical values (Label) to numerical (LabelKey)
var converter1 = mlContext.Transforms.Conversion.MapValueToKey("LabelKey", "Label");

var loading = mlContext.Transforms
    .LoadImages("input", _trainImagesRelativePath, "ImagePath");
var resizing = mlContext.Transforms
    .ResizeImages("input",
        InceptionSettings.ImageWidth,
        InceptionSettings.ImageHeight,
        "input");
var extracting = mlContext.Transforms
    .ExtractPixels("input",
        null,
        ImagePixelExtractingEstimator.ColorBits.Rgb,
        ImagePixelExtractingEstimator.ColorsOrder.ARGB,
        InceptionSettings.ChannelsLast,
        InceptionSettings.Mean);
```



The TF pipeline

```
var inceptionPipeline = mlContext
    .Model
    .LoadTensorFlowModel(tfModelPath)
    .ScoreTensorFlowModel(new[] { "softmax2_pre_activation" },
                           new[] { "input" },
                           true);

var trainer = mlContext
    .MulticlassClassification
    .Trainers
    .LbfgsMaximumEntropy("LabelKey", "softmax2_pre_activation");
```



Whole pipeline

```
// Numerical values (PredictedLabelValue) to categorical (PredictedLabel)
var converter2 = mlContext.Transforms
    .Conversion
    .MapKeyToValue("PredictedLabelValue", "PredictedLabel");

// Build up the whole pipeline
var trainingPipeline = converter1
    .Append(loading)
    .Append(resizing)
    .Append(extracting)
    .Append(inceptionPipeline)
    .Append(trainer)
    .Append(converter2);
```



Demo



Custom Image Classification



```
// Training
var pipeline = mlContext
    .MulticlassClassification
    .Trainers
    .ImageClassification(options)
    .Append(mlContext.Transforms.Conversion.MapKeyToValue("PredictedLabel"));

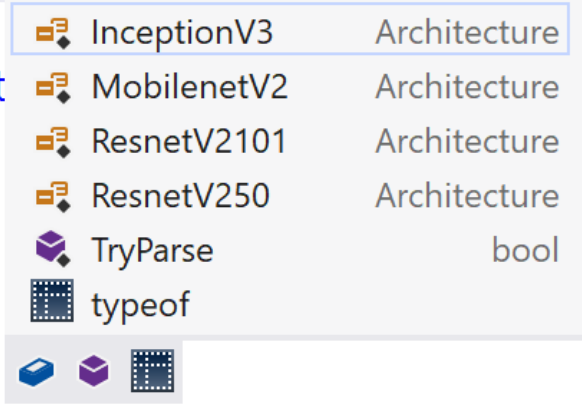
// Final model
var trainedModel = pipeline.Fit(trainSet);
```

Using the ImageClassification ML Task

- ❑ Same effect, more compact code
- ❑ The task encapsulates the composition of steps manually coded in the other scenario

Using the ImageClassification ML Task

```
var classifierOptions = new ImageClassificationTrainer.Options()
{
    FeatureColumnName = "Image",
    LabelColumnName = "LabelAsKey",
    Arch = ImageClassificationTrainer.Architecture.InceptionV3,
    ReuseTrainSetBottleneckCachedValues = true,
    ReuseValidationSetBottleneckCachedValues = true;
};
var pipeline = mlContext
    .MulticlassClassification
    .Trainers
    .ImageClassification(classifierOptions)
    .Append(mlContext.Transforms.Conversion.MapKeyToValue("PredictedLabel"));
```



InceptionV3	Architecture
MobilenetV2	Architecture
ResnetV2101	Architecture
ResnetV250	Architecture
TryParse	bool
typeof	

Computer Vision

- How the human brain recognizes objects is still largely unknown
 - MIT suggested inferotemporal neurons are dedicated to recognizing specific “images” (i.e., faces or objects)
 - Retina feeds frames to the visual cortex which transforms the input into coherent perceptions
 - Brain encodes the information in some way that at the end of some further neuronal computation chain produces the perception of the item seen
- Much like a neural network works!
 - Textures, shapes and color histograms offer a more stable representation of the information than raw pixel colors but the downside is that it just shifts the burden on feature engineering
 - Convolutional neural network boosted computer vision
- Re-training
 - Take a pretrained image neural network built for general object detection and add your own custom target classes and train for your own specific types of images
 - Still takes you hundreds (or thousands) of images to process but in a few hours of training, and most likely without expensive GPU activity, you surely get good results



Don't expect AI to clone humans.

Don't expect AI to be the implementation of
human dreams.

Expect AI to help humans.

Through software.



ARTIFICIAL INTELLIGENCE
is just software.

Smarter software.



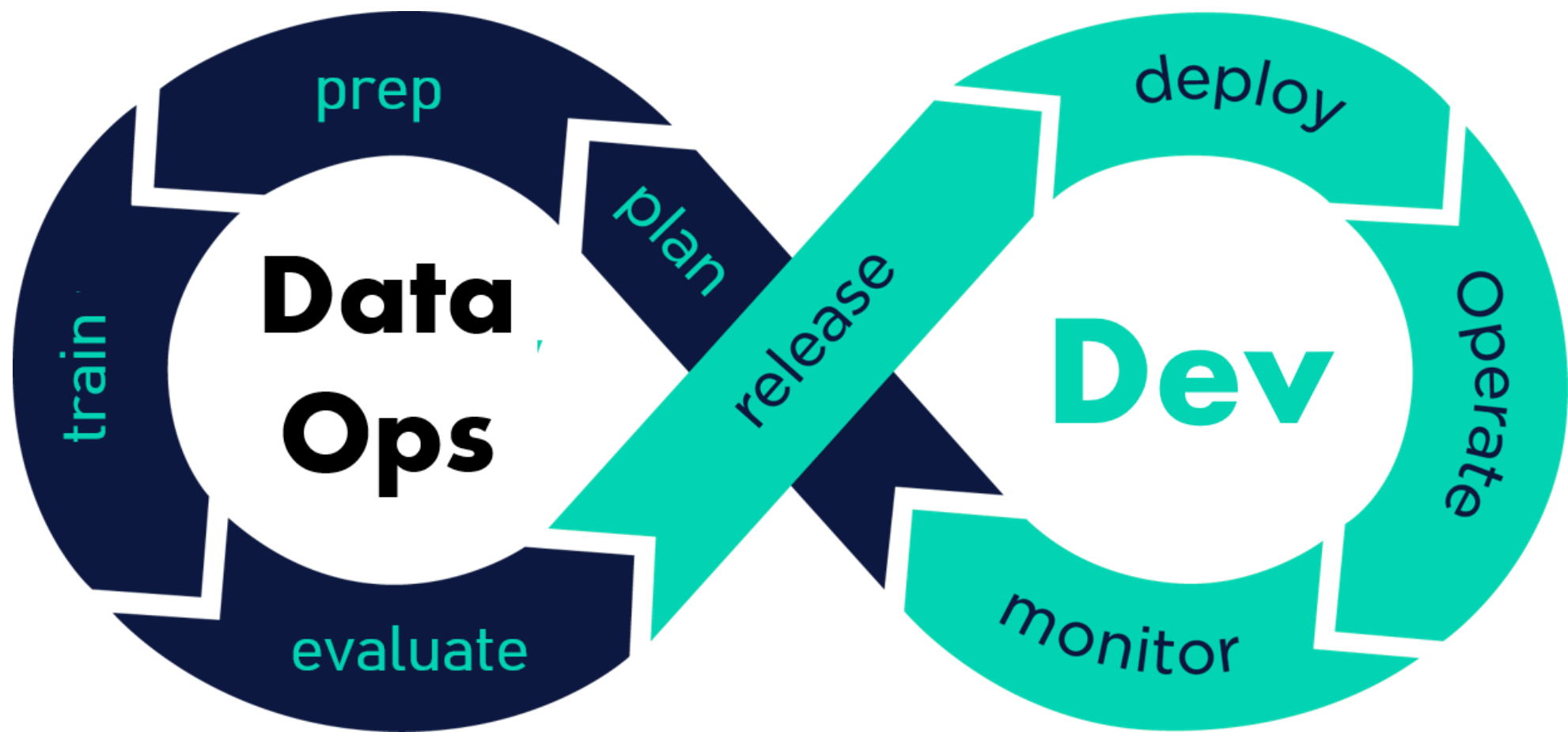
YOU CAN TOLERATE SOME LEVEL OF
INACCURATE RESPONSES

YOU'RE SOLVING A SPECIFIC PROBLEM; **IN A
SPECIFIC CONTEXT**

USE **ARTIFICIAL INTELLIGENCE** WHEN

YOU WANT TO ALLEVIATE REPETITIVE TASKS
AND KNOW HOW TO DO IT

YOU WANT TO ACCELERATE KNOWN TASKS
AND RESULTS



AI in the **user interface**

ML in the domain layer



POWER PRODUCTION FORECAST

Google is by far the largest producer of **renewable energy** among the major corporations. It is close to **2.5 GW**, 90% of which is from wind and the rest comes from solar panels.

1GW is enough for
700,000+ households in a
western country

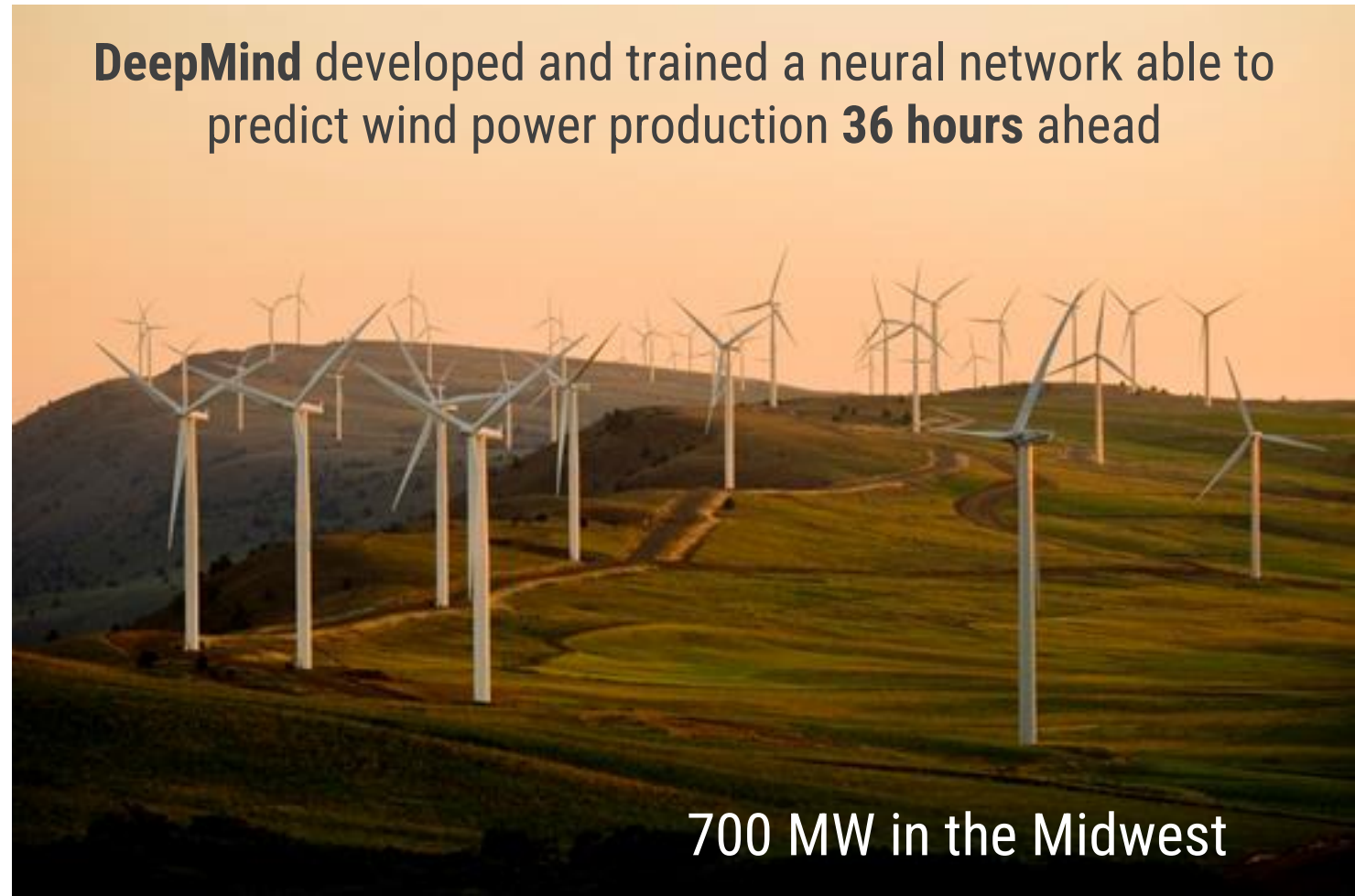
Wind forecasts

- Different level of precision in forecasts
- Wind is different at 80m rotor level
- Roughness, trees, buildings

Historical wind data

- In the region

DeepMind developed and trained a neural network able to predict wind power production **36 hours** ahead



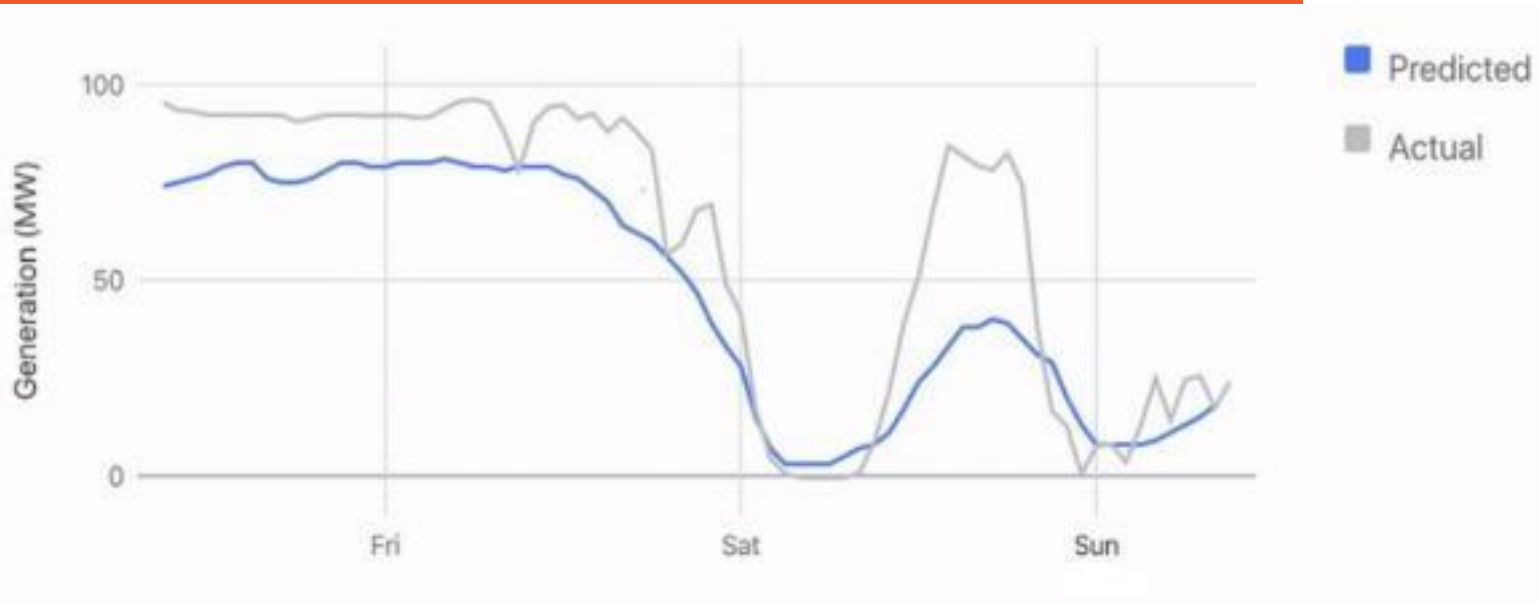
700 MW in the Midwest

Nearly constant **overproduction**



REPORTED CHART – one power plant, one day December 2018

POWER PRODUCTION FORECAST



Generically estimated
20% increased value
for MW/h

**Overproduction is worse
than underproduction**

Why Power Forecasting?

- Saves costs for direct marketers
 - Helps traders and power providers to estimate prices
 - Enables plant operators to schedule maintenance work
 - Timely identification of risks and faults
- Better rates resulting in the day-ahead market?
 - Better operational management of the power plant?

More realistically...
you need accurate wind history/forecasts and...

- 2+ years of 10-min time series data
- Historical data of power plant actual production
- Live production data (how much it is actually producing)
- Historical data of turbines availability (average of availability for each turbine) and, ideally, also maintenance records of turbines

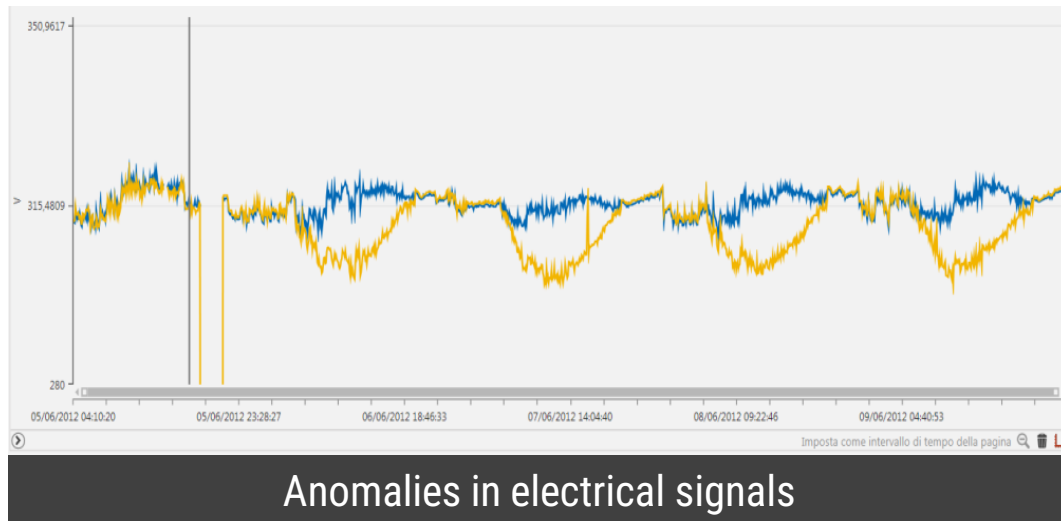
Not because there's wind all the turbines will produce regularly



ALARMS and TRENDS FORECAST



Complex UI for operators to figure out what could go wrong.



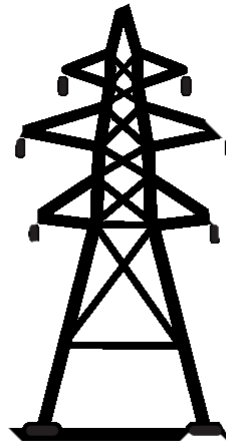
Silent check of data may trigger notifications. And the system can learn from mistakes.

BALANCING ENERGY OUTPUT



Conventional

nearly constant
output



Renewable

variable output



BALANCING ENERGY OUTPUT



Conventional

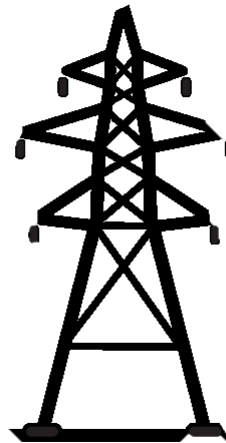
nearly constant
output



Renewable

AI system

contracted output



EMERGING SCENARIO



What data do we really need for accurate prediction?

How much are we going to invest and wait for results (and tolerate mistakes)?

Each turbine needs its own model.

A park-related model is not enough because many parameters change in a short geo distance: humidity, wind, air density and salinity, chemicals. They affect vibration and wear of components and corrosion of blades and rotors.

Granular data points.

10-min timeseries is probably too little, but it's a debated point. Different scientists might take it differently. For highly variable parameters 1-min precision would be great.

At least 2 years of data.

Records of past maintenance.

Most companies lack 2 years of such data or, at best, the data is not in a digital format or it can't be easily turned into structured data (i.e., PDF).

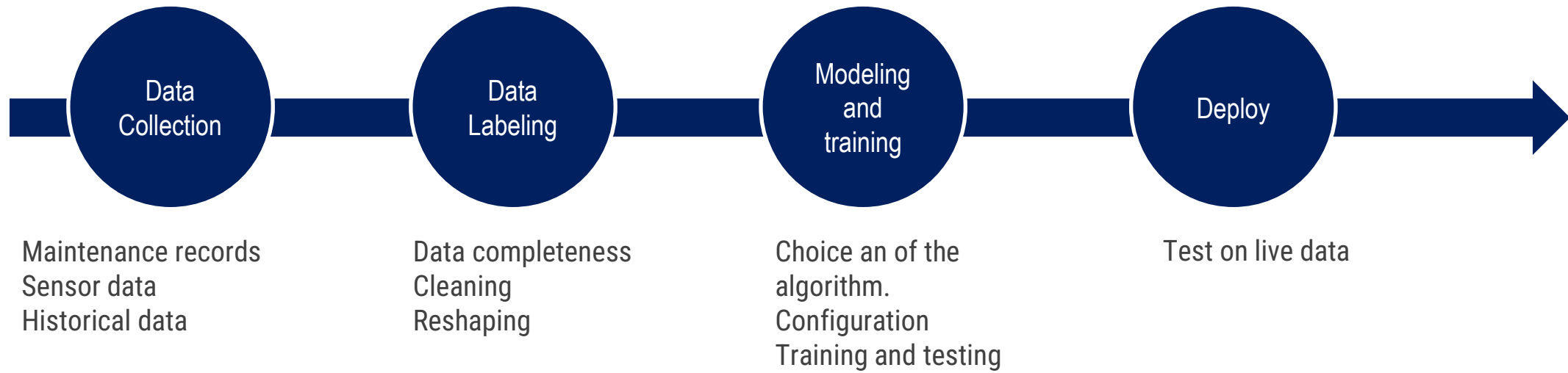
Question is, what was really wrong when the technician operated? What did he do?

The gearbox of a big 8MWh turbine can hardly be fixed (end-to-end) in less than 2 weeks. It means tens of thousands EUR of lost production plus the costs of repair.

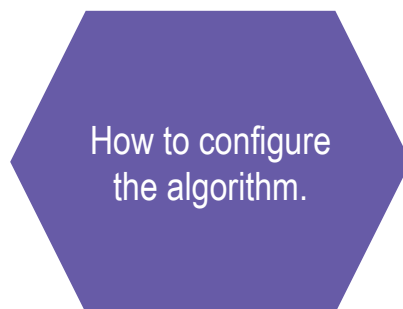


PREDICTIVE MAINTENANCE

SAMPLE APPROACH



WHAT COULD POSSIBLY GO WRONG?



What to look for exactly?
How long ahead?

- Operators balance risk vs cost and provide an optimal window for replacing or repairing an asset.
- Maximize the operating life of the asset, thereby optimizing CAPEX.
- Optimize the repair work, improving OPEX efficiency.



Just to cut a long story **(very)** short

STATISTICS

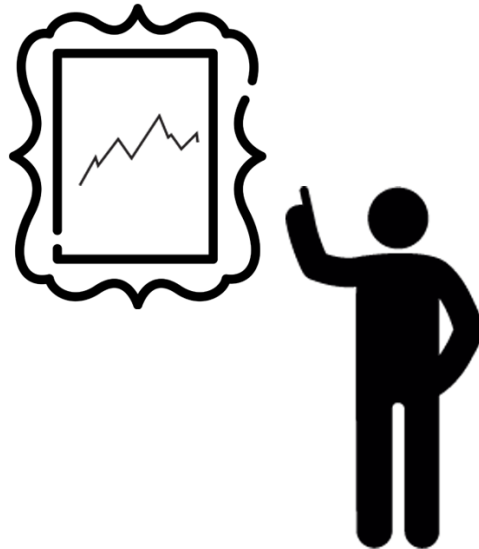


Just to cut a long story **(very)** short

STATISTICS



MACHINE LEARNING

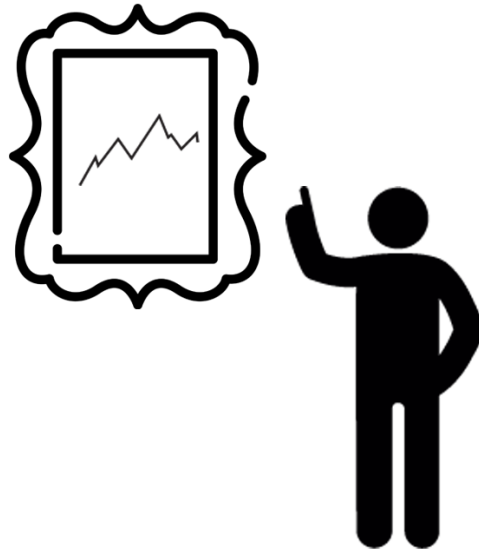


Just to cut a long story **(very)** short

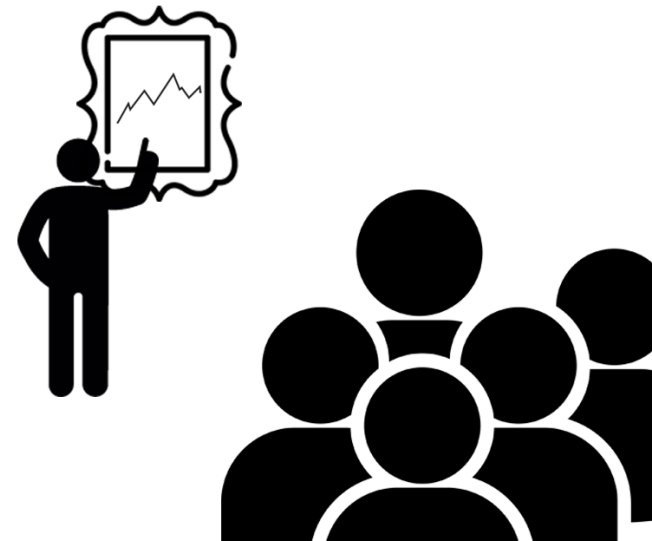
STATISTICS



MACHINE LEARNING



ARTIFICIAL
INTELLIGENCE



THOUGHTS?

