Scaling Machine Learning at Salesforce

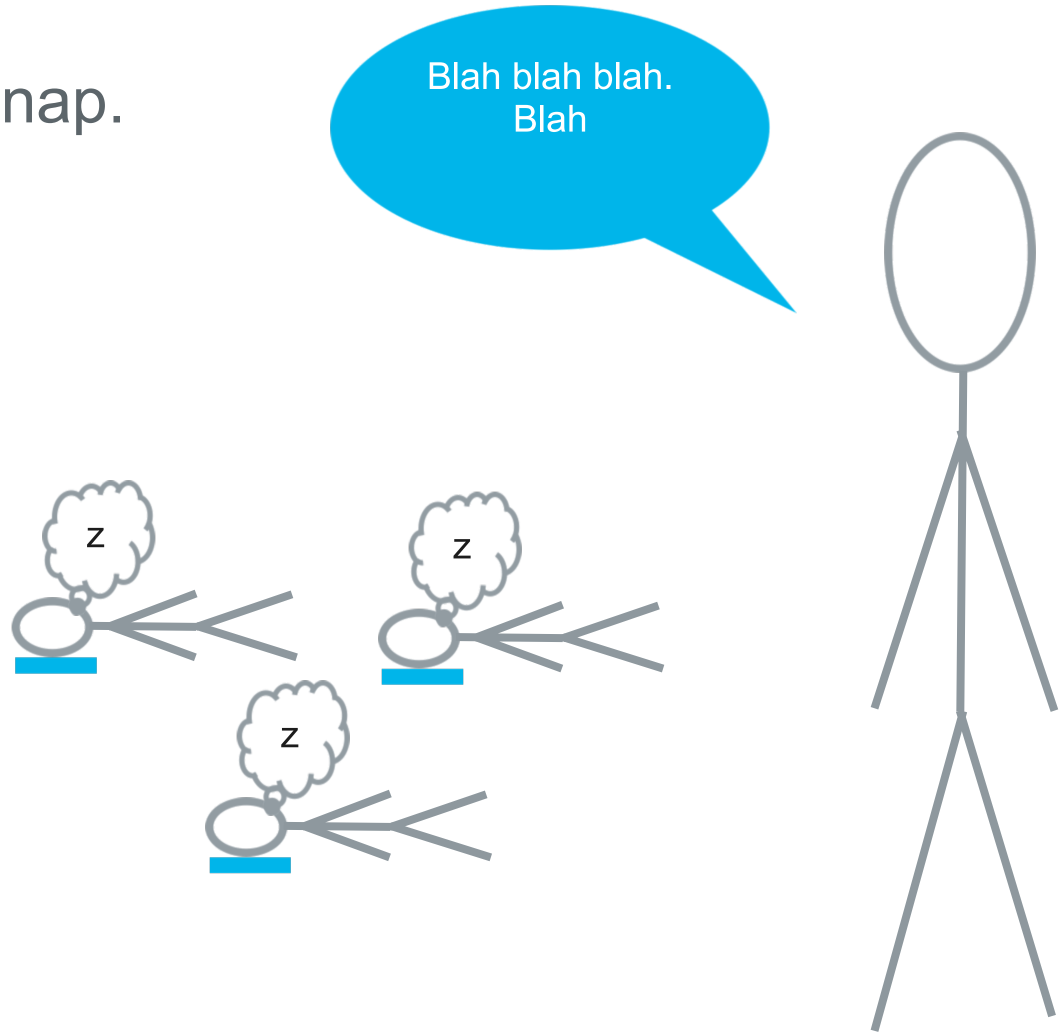
Leah McGuire, PhD
Lead Member of Technical Staff

salesforce

What I am going to talk about:

In case you are curious… or want to take a nap.

- The Salesforce use case – helping companies make better use of their data
- One model per company – scaling model building
- Our machine learning platform – how to build many different types of models with one one model per company
- The importance of monitoring in automation



The magical panacea that is machine learning…

Or not.

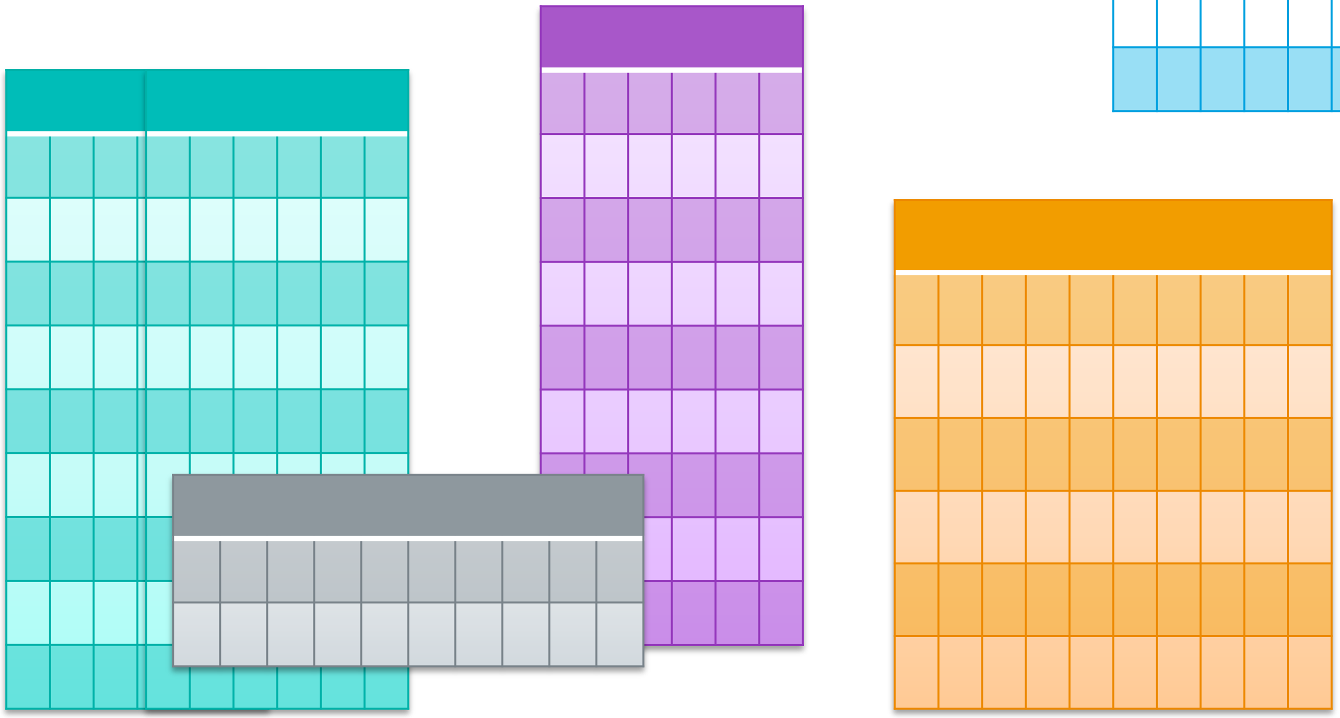
- Definition: Machine Learning

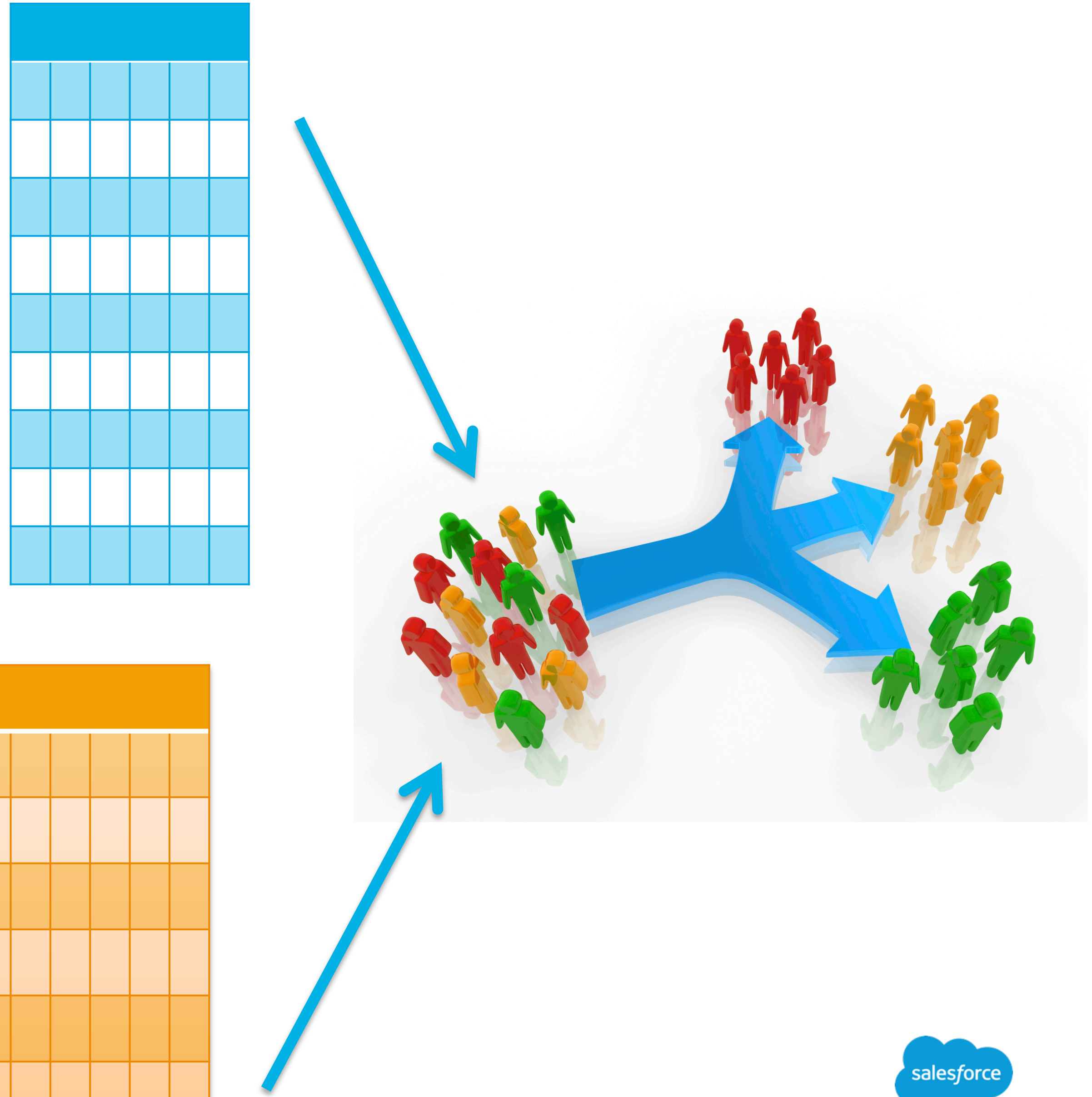
"Machine learning algorithms can figure out how to perform important tasks by ***generalizing from examples****. This is often feasible and cost-effective where manual programing is not. As more data becomes available, more ambitious problems can be tackled. As a result, machine learning is widely used in computer science and other fields. However, developing successful machine learning applications requires a substantial amount of* ***'black art'*** *that is hard to find in text books" – Pedro Domingos, U of Washington, A Few Useful Things to Know about Machine Learning.*

- ML is **not** magic, just statistics – generalizing examples
- But what is this 'black art'?
  - You cant just throw algorithm at your raw data and expect good results
  - Different types of problems require different algorithms
  - Data needs to be: 1) cleaned so that 'bad' data is removed 2) manipulated so that the most predictive features are available 3) put into the correct format

The Salesforce use case

- We store data for other companies – all kinds of data (sales, marketing, operations, etc.)
- They want this data to be "smart"
- We need to provide machine learning on top of the data stored in our systems



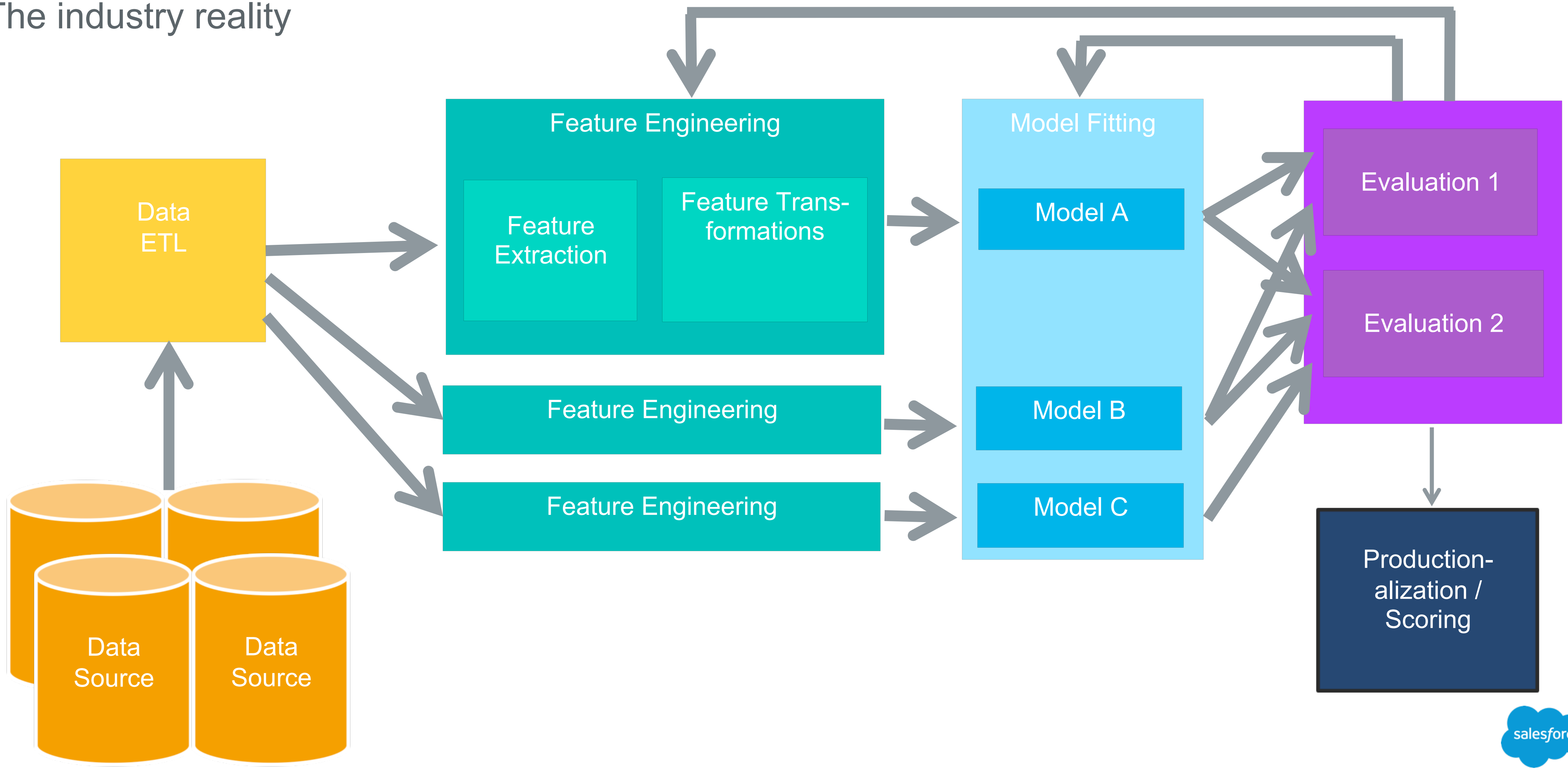


The Salesforce use case

- The key difference from most ML use cases – building a model for a single use case means building hundreds or thousands of models
- Each companies data is treated separately!
- We know what type of information is each table and column, but companies use the fields differently and have different properties

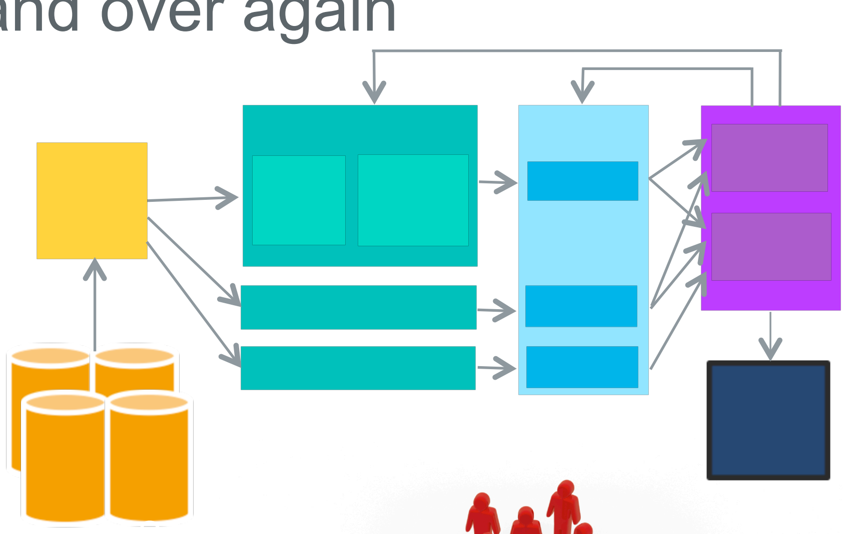
Building a machine learning model

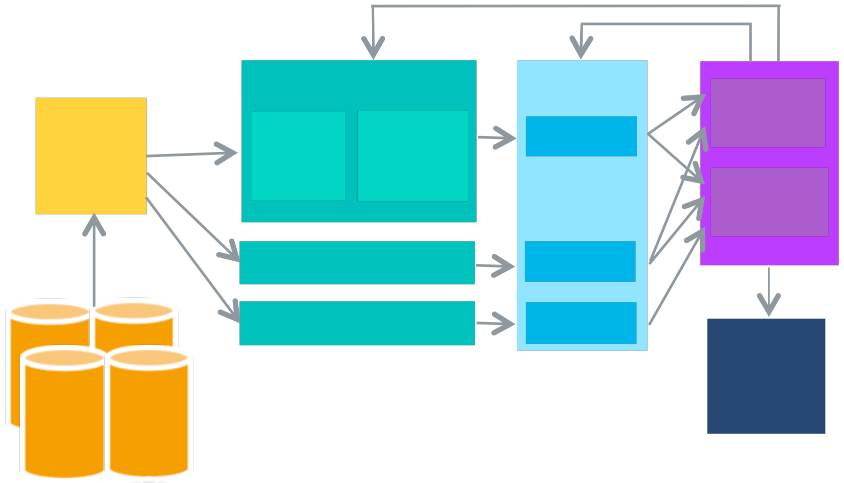
The industry reality



Building a machine learning model

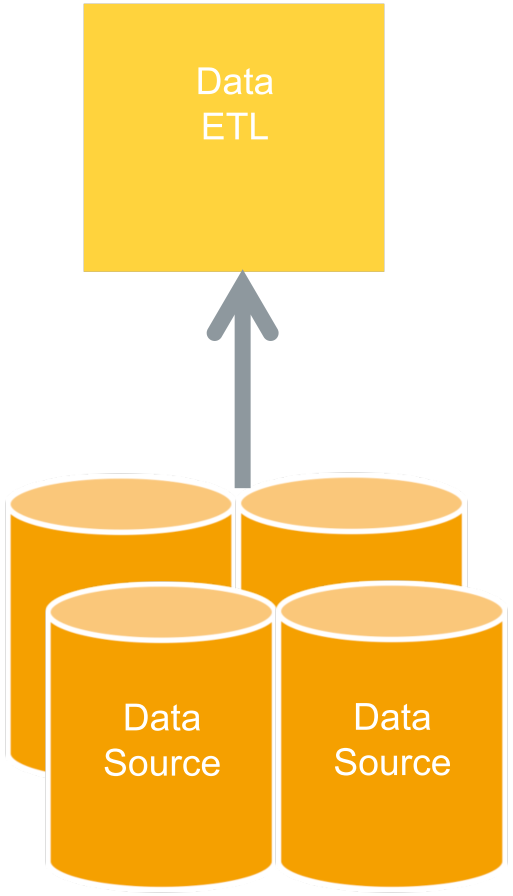
Over and over again





Building a machine learning model

How do we scale this?



- Need to have the data extraction and processing happen automatically, seamlessly, and with as much information as possible about the data (STRONGLY TYPED DATA)
- Need to manage model updates to be sure nothing goes wrong with model retraining
- Need to score in a timely manner so that the information is useful
- All this alone could be several talks, but I am going to talk about the middle…How do you build all these models?

Building a machine learning model

How do we scale this?

- Most of the time goes into data manipulation (80-95% depending on who you talk to)
- So this as automated as possible for first pass
- Modeling wrapped in standard interface so can switch models easily

LOTS of people have build ML frameworks.

Spark ML

WEKA The University of Waikato

ML base

scikit learn

Scalding

pandas

$y_{it} = \beta' x_{it} +$

mahout

R Studio

KNIME

KeystoneML

PredictionIO

cascading

salesforce

LOTS of people have build ML frameworks.

Spark ML

WEKA
The University of Waikato

ML base

scikit learn

Scalding

pandas

$y_{it} = \beta' x_{it} +$

mahout

R Studio

KNIME

KeystoneML

PredictionIO

cascading

salesforce

What can we use and what do we need that isn't there?

Lets not reinvent the wheel here.

- Heavily influenced by Spark ML, Keystone ML, Prediction IO
  - Everything is build on Spark
  - Modular reusable pieces
  - Type safe
  - Take whatever pieces from these platforms we can and build them into our platform
- Automation of everything we can possibly automate
  - We need to deal with feature engineering in a smart way
  - We need to do model selection and hyperparameter tuning automatically (to some extent)
- Evaluation and metrics everywhere!
  - Measure EVERYTHING – and respond appropriately

The pieces of our ML platform

Workflow

Feature Extractor

Data Prep

Joining data sources

Time based aggregation

Conditional aggregation

Transformation Plan

Feature Engineering

Znorm,
Log transforms,
TFIDF, cosine
similarity, categorical
pivots

Model Selector

Model Selection

Sanity Checking

Rebalancing

Model Fitting

Recalibration

Scoring

Prediction

Load Model

Data Prep & Feature Transformation

Apply model

salesforce

Feature Extractors

The first part of making each step re-usable is to put things in a standard format

Feature Extractor

Data Prep

Joining data sources

Time based aggregation

Conditional aggregation

- Extractors function as an interface between the data and our framework
- They are generally defined one per data source – can have many per workflow
- Conversion from input to our data format is several stages
  - Data is read and a specific type of record is returned
  - Events are defined for that record type
  - Events are used to extract features for each row
  - Each type of feature is aggregated overtime or condition
  - Features are combined to give a single feature vector for each entity to be scored
- **All our data looks the same no matter where it came from**!

salesforce

Feature Transformers

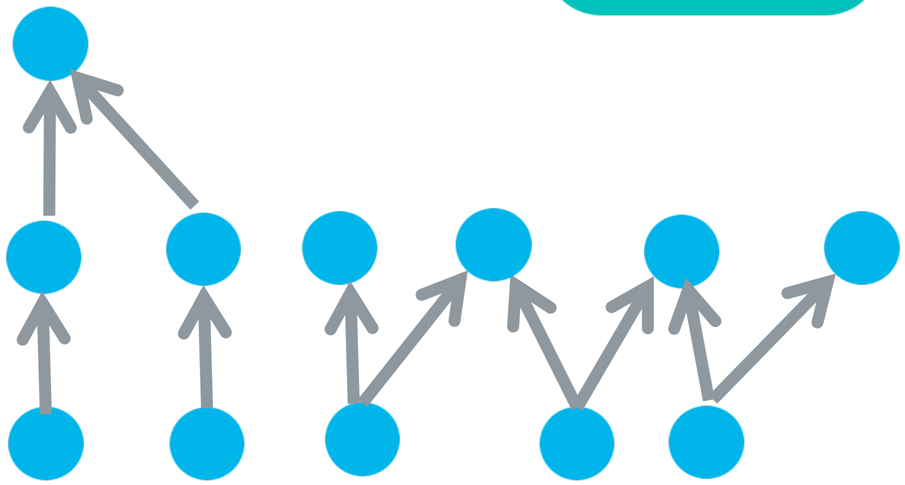
Feature engineering is a large part of building a good model

Transformation Plan

Feature Engineering

Znorm,
Log transforms,
TFIDF, cosine
similarity, categorical
pivots

- There are many types of transformations that we may want to perform
  - Mathematical – Log, Normalize, Cap …
  - Expansion – Pivot, Bin, TFIDF …
  - Reduction – Hash, Minimum Requirements …
  - Combination – Interaction, Similarity …
  - Time – Days Since, Weeks Since, Occurred on ..
  - Type specific – Valid phone number, email domain extraction
- Can capture these in two main types of transformers
  - Simple – takes a single row and produces a new value
  - Aggregate – needs to know about the entire column values (Twitter Algebird: prepare, reduce, present)
  - Can chain these together as efficiently as possible in a DAG



Feature Transformers

Transformation Plan

Feature Engineering

Znorm,
Log transforms,
TFIDF, cosine
similarity, categorical
pivots

What you write and what you get

- A sequence of transformations, generated by mapping over the features names that need that transformation

```
val loggedClicks = clicks.log()
val pivotedState = state.topKPivot(10)
val tfidfRespondedSubjects = respondedSubjects.tfidf()
val tfidfIgnoredSubjects = ignoredSubjects.tfidf()
val subjectSimilarity= tfidfRespondedSubjects.similarity(tfidfIgnoredSubjects)
```

- A brand new set of features that have been explicitly transformed (even if just with identity)

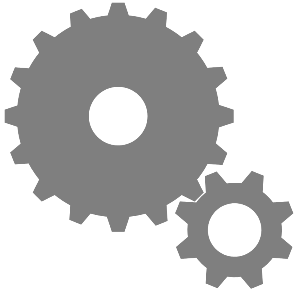
Key	Clicks	State	Opens	Subject
A	0	CA	0	Blah
B	5	NM	10	Boo
C	1	TX	2	Stuff

→

Key	Clicks-Log	State-CA	State-NM	Opens/ Send	Subject-Similarity
A	0.0	1	0	0.0	0.99
B	1.791759	0	1	0.5	0.01
C	0.693147	0	0	0.13	0.04

Model Selectors

Make a uniform interface for all machine learning models



Model Selector

Model Selection

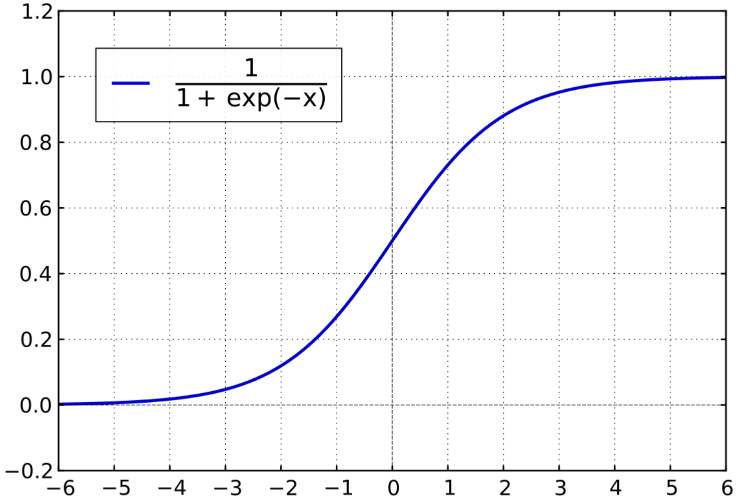
Sanity Checking

Rebalancing

Model Fitting

Recalibration

- Want to be able to switch models easily – One interface for all models
  - Need to get the data in the correct format for whatever library or model
  - Check your data before fitting the model (**Sanity Checker**) - Make sure there is no label leakage, make sure your features have the values / ranges you expect
  - Do resampling and rebalancing as needed
  - Fit the model or models and do hyperparameter tuning
  - Save model for later use
- What you get out: the model
  - Needs to score data
  - Provide info about the model performance
  - Load saved models



Scoring

- Use saved feature transformations and model to provide scores
- Reuses the model training workflow with different parameters
- Occurs as frequently as needed to provide customers useful scores
- Write the scores back out to whatever format needed to serve the customers

Scoring

Prediction

Load Model

Data Prep & Feature Transformation

Apply model

salesforce

The pieces of our ML platform

Workflow

Feature Extractor

Data Prep

Joining data sources

Time based aggregation

Conditional aggregation

Transformation Plan

Feature Engineering

Znorm, Log transforms, TFIDF, cosine similarity, categorical pivots

Model Selector

Model Selection

Sanity Checking

Rebalancing

Model Fitting

Recalibration

Scoring

Prediction

Load Model

Data Prep & Feature Transformation

Apply model

So great, we have a way to make lots of models!

Is it actually working?

- Have to make sure your models are worth shipping
  - Need many metrics of performance
  - If the model doesn't meet the criteria set it does not go out
- Need to make sure that model quality is consistent
  - Retrain models periodically and report quality to end users
  - If quality drops need to figure out why
- If the pipeline fails need to know why
  - New customers can break your assumptions about the data
  - Old customers can change the way they are using fields or have data issues

Or you know, failure…

Summary and next steps.

Ok so far but this isn't enough…

- At Salesforce we need to scale machine learning not only for the size of data but for the number of customers
- We can build a single machine learning pipeline which builds many models for a particular application
  - Each model is customized to the customers data
  - The platform automatically transforms data to deal with differences between companies
  - The platform automatically selects the best model and parameters from the set you define
- We can detect issues with the data and respond appropriately
- We WANT to be able to allow customers to tweak the models

http://learningradiology.com/misc/sitemap.htm

thank you

&

We are hiring ☺