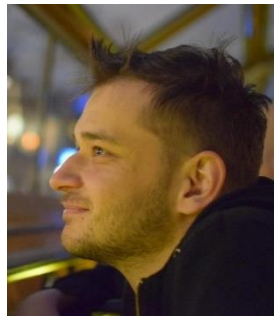
NLP

14th May 2019

Who Are We?

- Online Data Science – Dell.com websites
- All dell.com application (Commerce, B2B, B2C, Support, etc.)
- Data we deal with
  - UX/UI
  - Web application
  - Hardware and support



Ondrej Jariabka
Machine Learning Developer
ondrej.jariabka@dell.com

DSF Team Composition – 4 Core Pillars

- Team in 5 countries – Slovakia, USA, India, Brazil, Ireland
- Various backgrounds: Developers, Business Analysts, Marketing, Academia

Data Engineering

Data Architecture

Data Science and Machine Learning

DevStack

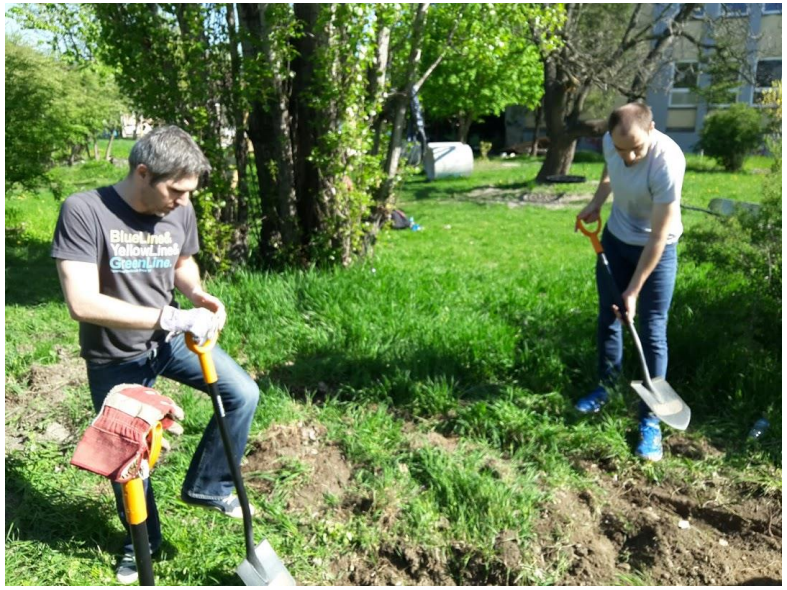
What We Do?

- **Offline analyses**
  - Customer Intent
  - Search Keyword Analysis
  - Anomaly detection on various server logs
  - Automated root cause analysis
  - Website feature launch assessment
  - Problem2solve

- **Online projects**
  - Smart Select (Better customer configuration)
  - SNP Recommender system (EOL/OOS products)
  - Discover Products (recommendation system on product variants)
  - Product Ranking (website building – how to sort and display products for customers)

Contributing To Community

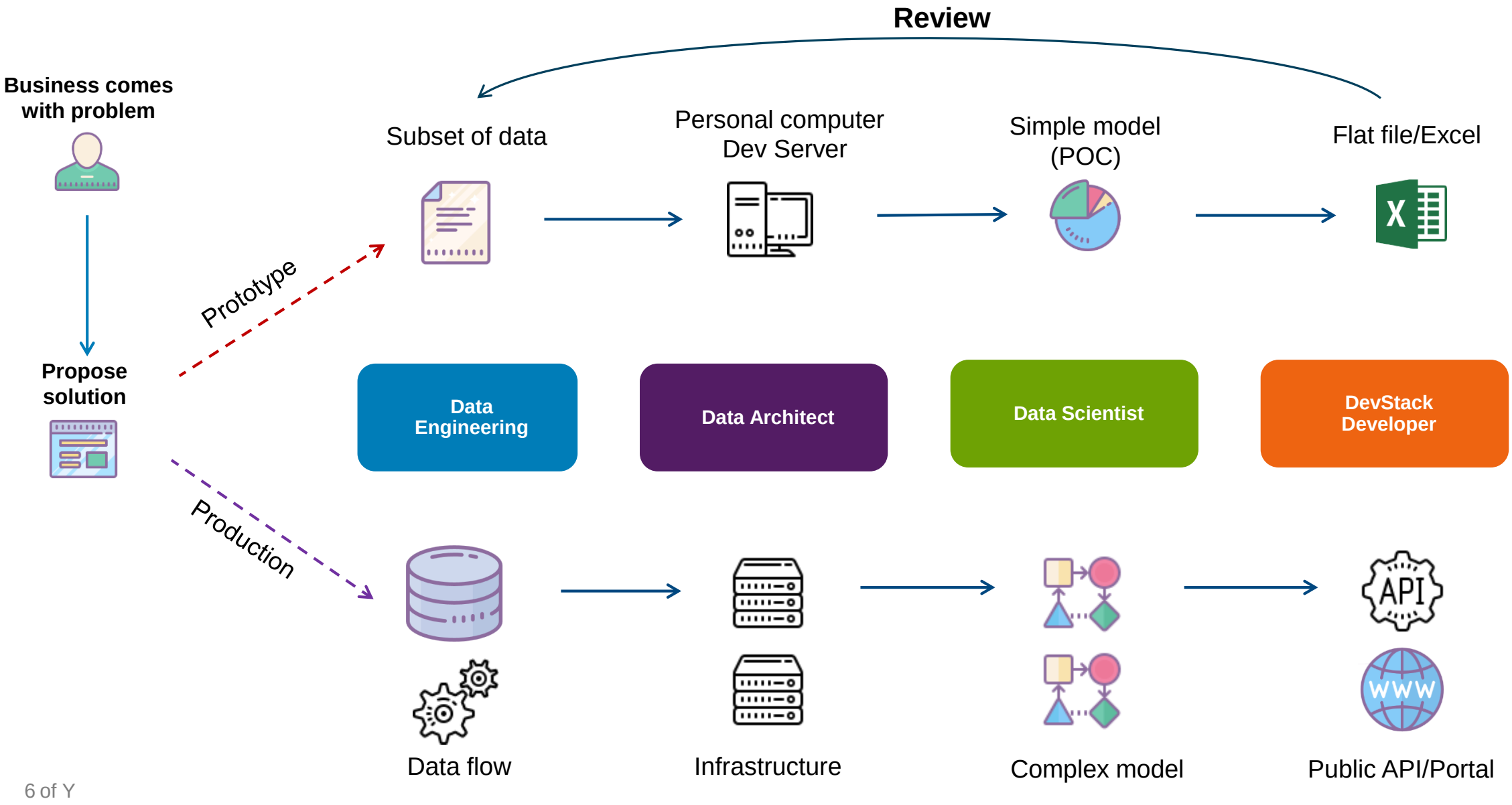
Nova Cvernovka #1



Nova Cvernovka #2

OpenSlava 2018

Prototype to Production Lifecycle



Roles and Responsibilities of Data Scientist

- Creates new solutions based on given business ask
- Automates Decision Making
- Data Cleaning / Feature Engineering / Data Prep
  - more data = more cleaning (80%-90%)
- Developing Machine Learning models
- Optimizing the processes to ensure scalability
- Monitoring developed models
  - Developing automated tests for machine learning models
  - Root Cause Analysis
- ML Ops

Warm up

End Of Life/Out Of Stock Recommender

Dell 27 Monitor: S2718NX

★★★★☆ 3.9 (9) Write a review Ask a question

☐ Add to Compare

Dell

Manufacturer Part FP7M5
Dell Part 210-ALIO

Experience every thrill in Dell HDR with this beautiful 27" monitor featuring AMD FreeSync™ technology & a virtually borderless InfinityEdge display.

Order Code 210-alio

No longer available

Similar products

Viewing 1-4 of 4

Dell 27 Monitor – S2719H

Price ~~$329.99~~ $249.99

Dell 27 Monitor: S2719HN

Price ~~$319.99~~ $199.99

Dell 27 Monitor: P2719H

Price ~~$359.99~~ $269.99

Dell 27 Ultrathin Monitor: S2719DM

Price ~~$499.99~~ $329.99

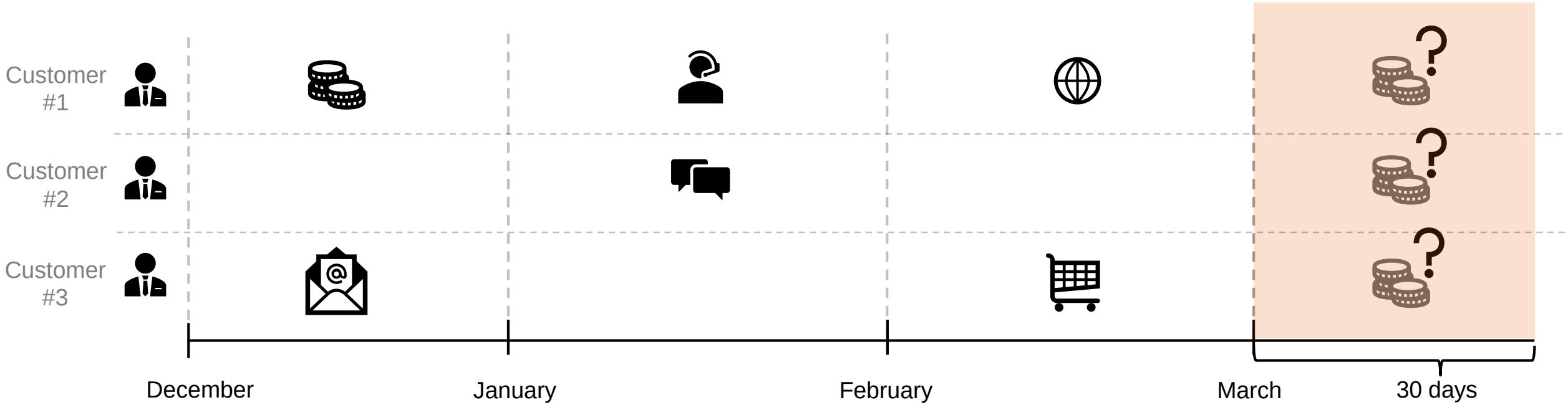
ML Template Project

Problem Statement

Predict customer likelihood to buy in next 30 days

(this is called a propensity model)

We know what customers did up to certain date and want to "guess how likely they are to buy

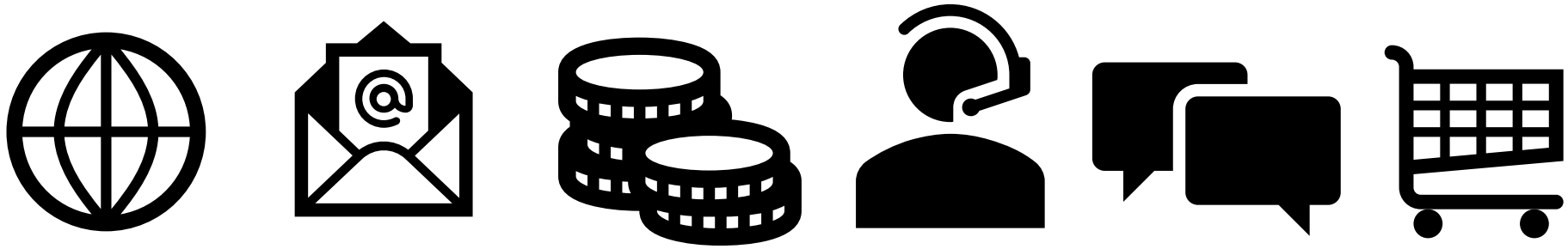


How our input data looks like

Large dataset of customers (300 GB)

- 90M+ customers, NM+ customers
- 700+ features (e.g. #products, #bought_products, #emails…)

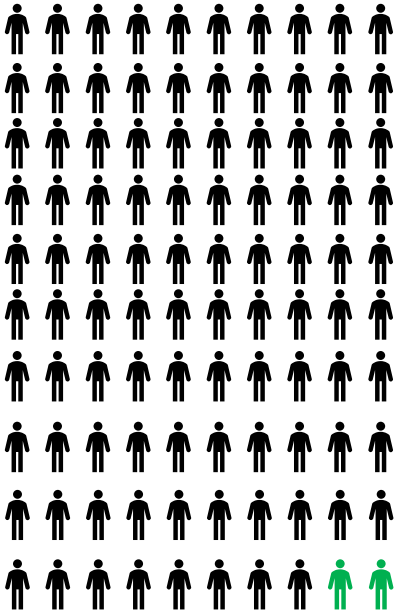
  Website, emails, firmographics data…



How our input data looks like

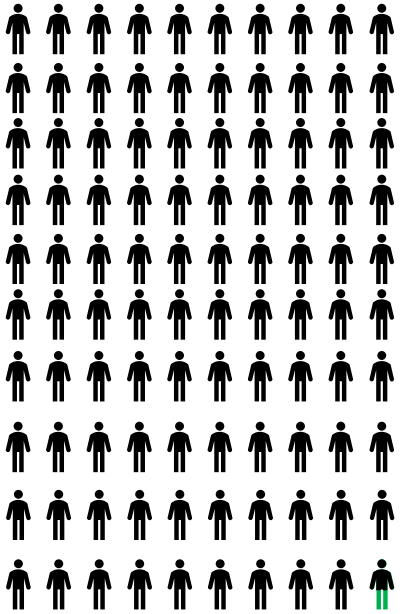
- Target variable highly skewed
  - Target – indicator variable if customer purchased in 30 days

~2% of customers buy on average



2.0%

In some segments <0.01% purchase

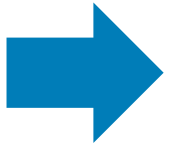


0.01%

Why big target skewness is an issue & accuracy not a best measure ….

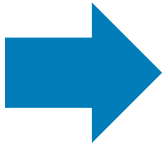
- Easy to achieve high accuracy just by predicting majority class
  (easy to ignore/overlook buyers as there only few)
- Goal is ranking customers by likelihood → accuracy does not measure this

```
 8
 9
10 def model():
11     return 0
```



90,000,000

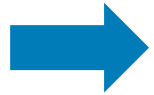
TP 0	FP 18,000,000
FN 0	TN 72,000,000



$$\frac{TP+TN}{TP+FP+TN+FN}$$

$$\frac{0+72000000}{0+18000000+72000000+0}$$

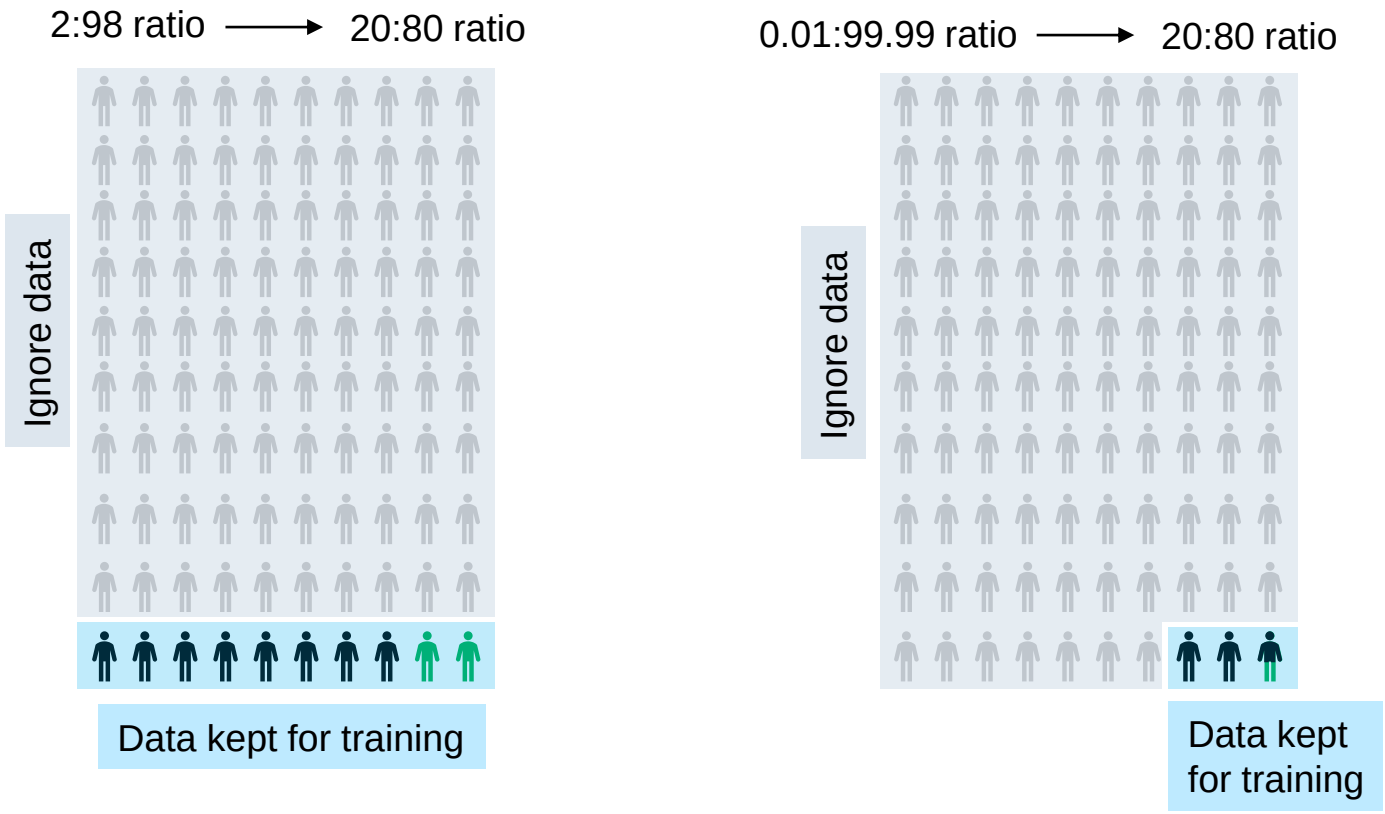
accuracy



98% accuracy

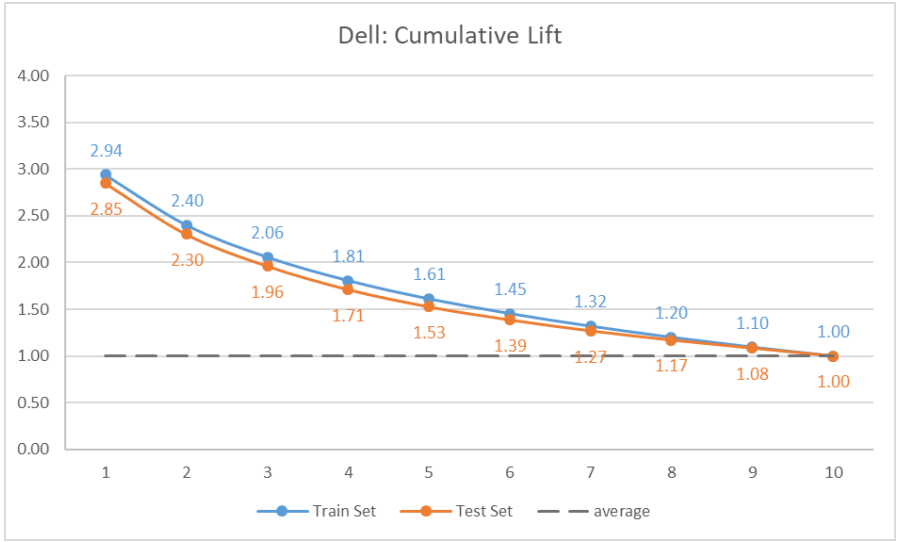
Solution to Skewness problem

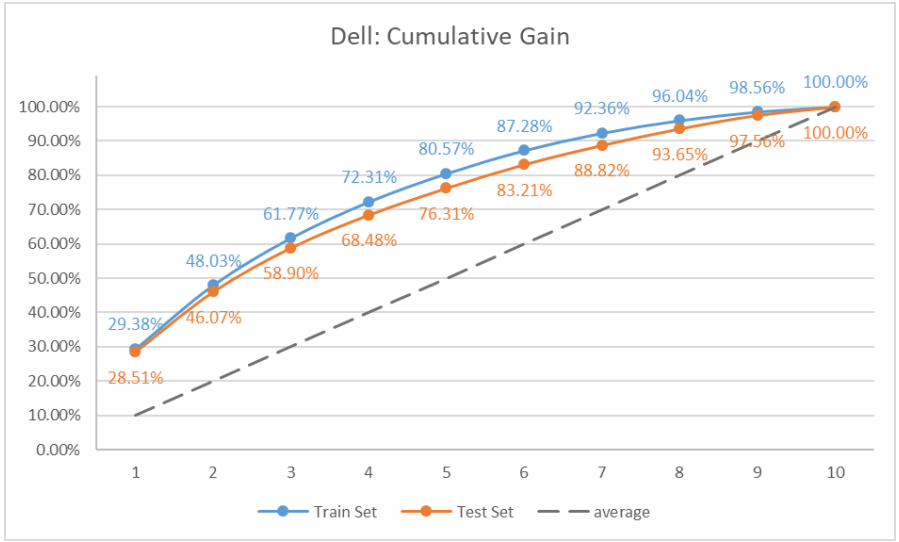
- To balance the problem we under-sample the non-buyers
  - E.g. to have 20% buyers vs. 80% non-buyers



Solution to Metric problem

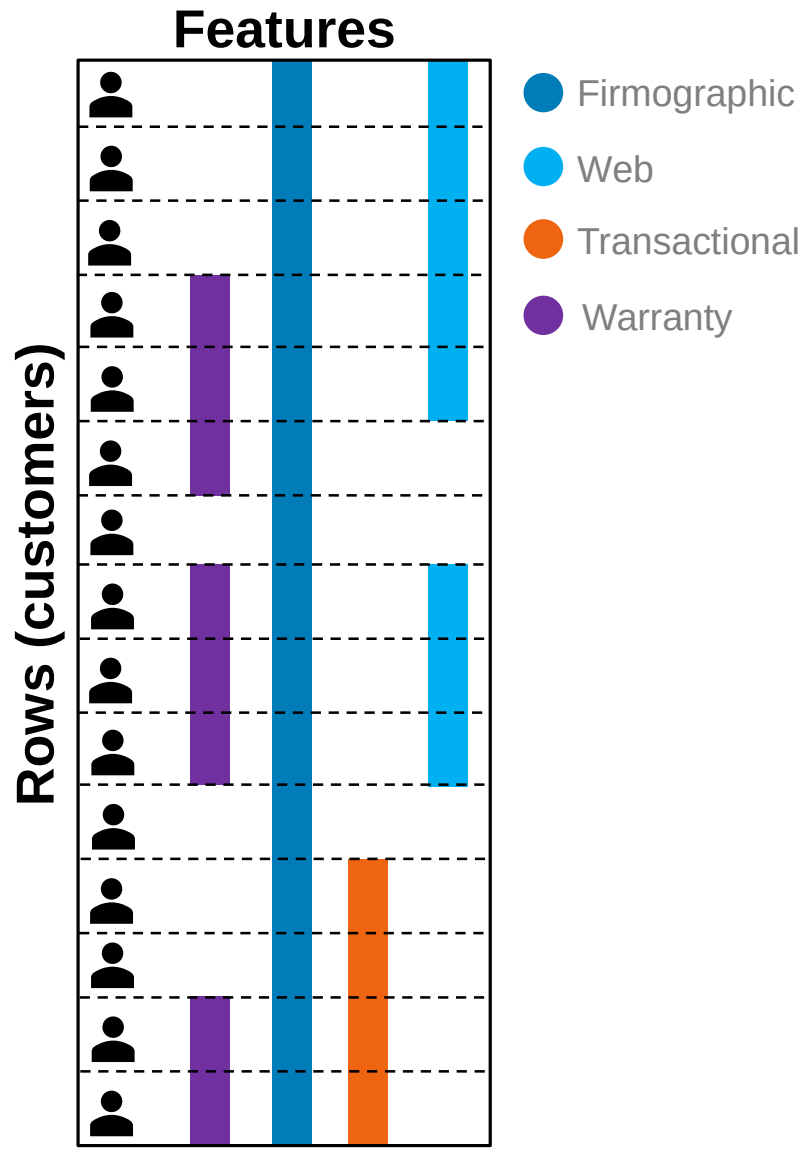
- We want to maximize purchasers contained in first deciles
- Customers segmented to deciles based on their purchase propensity
- Consider first segments (1-4) as purchasers
- We compute lift [1] and gain [2] curves
- The difference between validation and test curve can’t be to large
- Closely related to AUC [3]
  - *We also tested AUC/ROC AUC but found no significant improvement*





How our input data looks like

- Roughly 60% of the dataset have missing values
  - Only some customers visit online
  - Only some customers have previous purchase
  - Only some customers contacted e-support



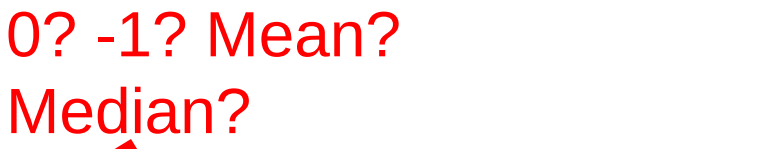
Is data sparsity OK or needs special handling?

- Would feeding data like below work in ML model ?

customer_id	#emails	#products	#bought_prdcts	#interactions	bought
John	NULL	NULL	1	3	0
Lukas	2	4	NULL	0	0
Anna	1	3	1	NULL	0
Beata	NULL	5	0	0	1

Is data sparsity OK or needs special handling?

- Null would not work for most models – they need to be replaced by a value or a placeholder value

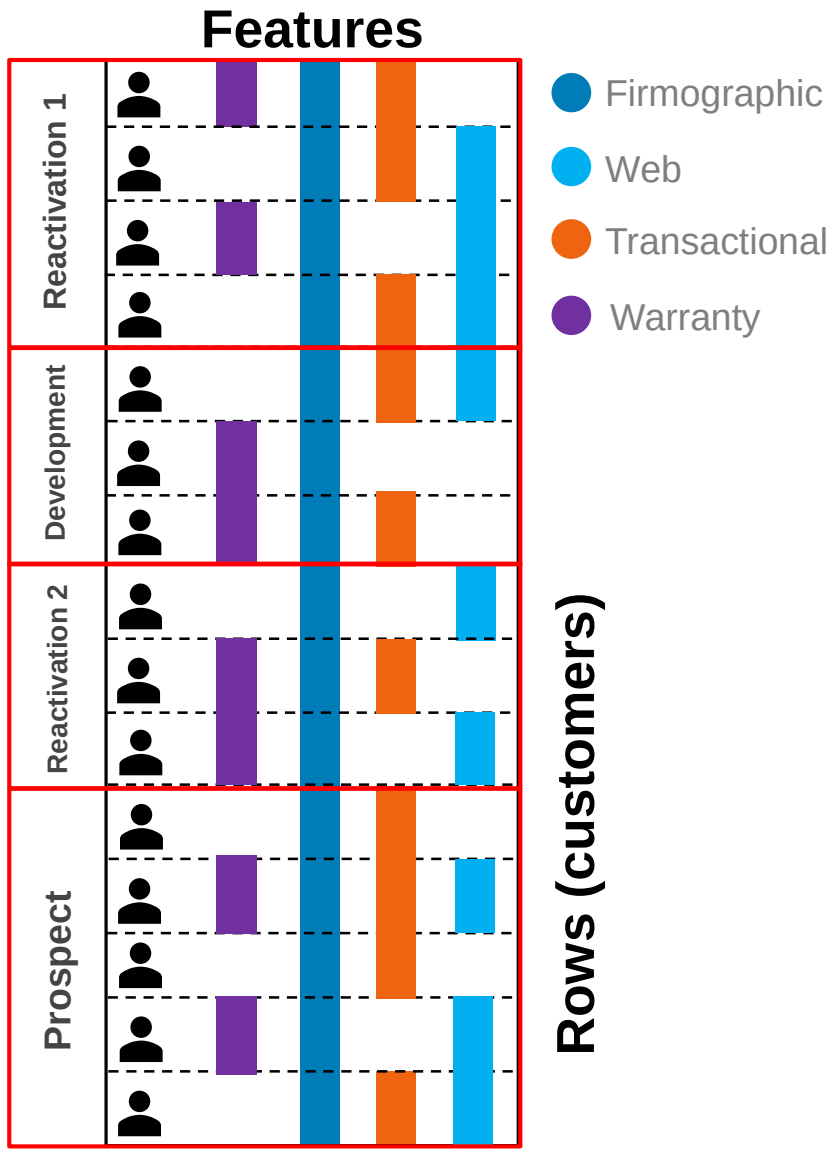


customer_id	#emails	#products	#bought_prdcts	#interactions	bought
John	NULL	NULL	1	3	0
Lukas	2	4	NULL	0	0
Anna	1	3	1	NULL	0
Beata	NULL	5	0	0	1

Is the way business uses the data aligned with how it is structured ?

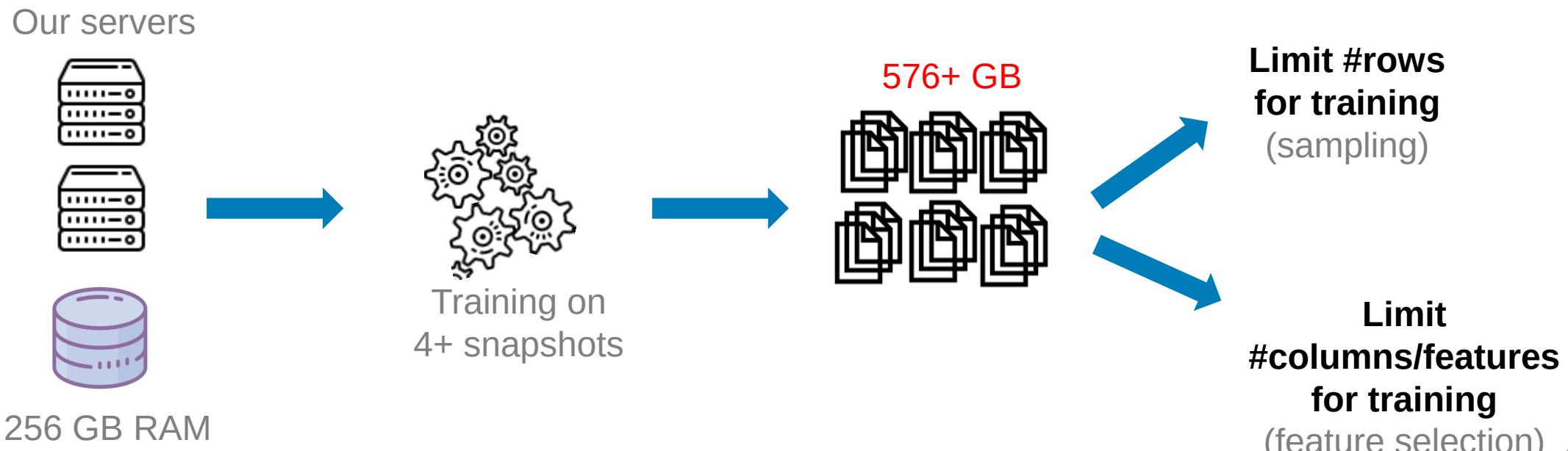
Would this help us to divide and conquer (to speed up training) ?

- Decisions/actions based on business segmentation
- Each segment contains very diverse customer behaviors
  - Represented by different columns/features being present
  - This diversity strongly impacts models performance
- Business segmentation **might not be the best way to create separate models**



Strategies to downsize training data & get better models

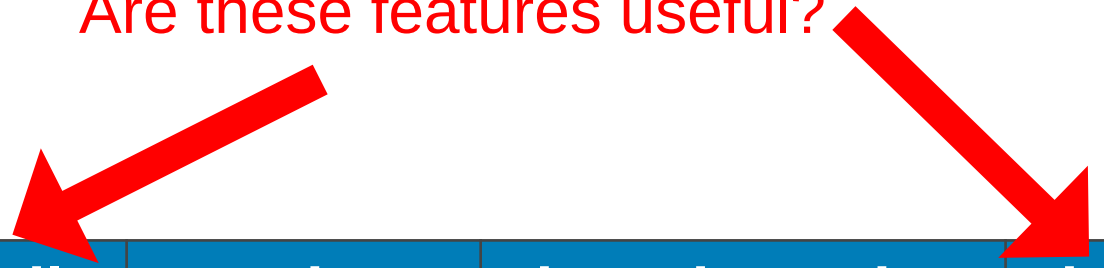
- Reduce the size of the dataset for training !
  - This speeds up the training & allows parallel execution
  - 90,000,000 rows * 400 integer features * 32 bits = 144 GB/snapshot



How to limit # columns/features before training

- First, check if there are columns that contain very little information
- If yes remove them !

Are these features useful?

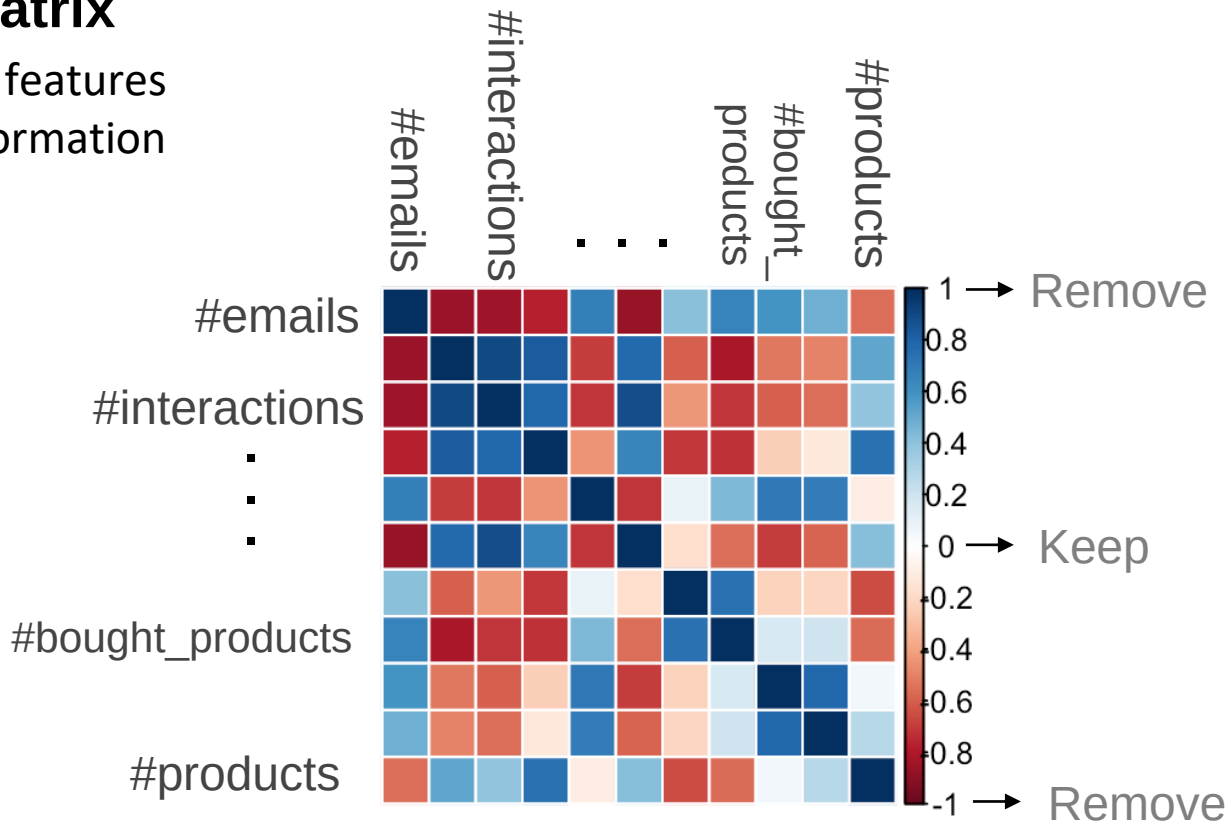


customer_id	#emails	#products	#bought_prdcts	#interactions	bought
John	2	1	1	3	0
Lukas	2	4	5	3	0
Anna	1	3	1	3	0
Beata	2	5	10	3	1

How to formally define feature usefulness

- **Invalid / Low variance** features
  - Values in the columns mostly stay the same and don't vary across customers
- **Redundant features** based on **covariance matrix**
  - Covariance matrix captures correlations between features
  - Highly correlated features likely contain same information hence one is only selected

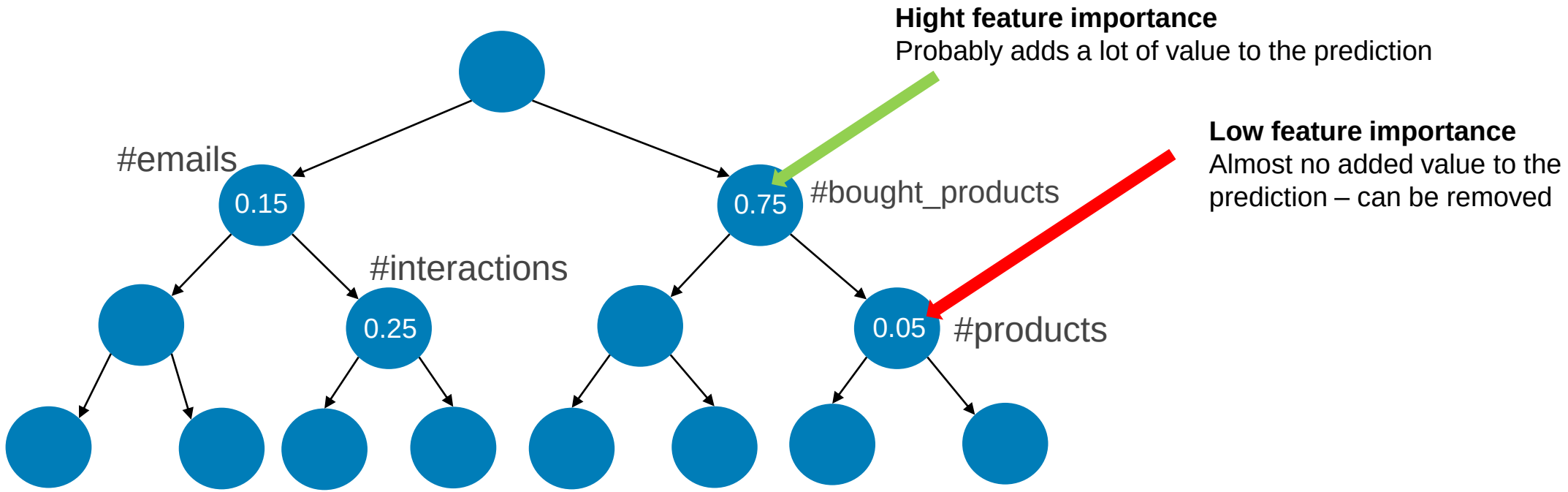
Low variance ~ low information



High Correlation ~ same/ redundant info

Use Random Forest (RF) as feature selector

- Another way to check for feature usefulness is to train a random forest
- RF identifies important features: importance is inherent part of the model



Our Solution

Training XGBoost (Decision Tree on steroids) for final predictions [4]

- Missing values are identified and passed to XGBoost
  - *From experience and tests this performs better than keeping imputed values*
- Hyperparameter tuning via cross-validation to choose best parameters

Key Takeaways

- With large datasets standard approaches **might** fail
- Even simple models might take very long time to train
- For large dataset even 1% might still be a lot of data especially if random sampling is incorrect

Key Takeaways

- More often than not “clean data” are **not clean data**
- Stratify sampling based on multiple features is usually way to go to reduce the size for training
- With more data more time is spend on data preparation

Key Takeaways

- If in doubt - XGBoost (*faster runtime and better results were achieve when running GPU implementation*)

- Rather simple than complex
  - If complex try multiple forms of regularization

- Do not focus only on model metrics
  - Always check were your model makes mistakes and adjust based on that

We are hiring!

ondrej.jariabka@dell.com

https://jobs.dell.com/slovakia