EXPONEA

Live predictions with schemaless data at scale

Ondrej Brichta

EXPONEA

MARKETING CLOUD

WE ARE HIRING

Our problem at Exponea

What we have:

- Terabytes of customer data
- Means to process all the data fast
- Customers with business understanding (or our consultants)

What we needed:

- Probability classificator
  - For any chosen event or attribute
  - Able to process any given data and return a reasonable output **without human interaction**

Our problem at Exponea

One specific problem:

- How to choose which is the best banner to show right now?
- Web page personalization?

Solution?

In session prediction

- Predict probability of buying for each customer live when they are online

What steps do we need to do and automate?

1. Business understanding
2. Data understanding
3. Data preparation
4. Modeling
5. Evaluation
6. Deployment

1. Business understanding

Key questions:

- What problem do we have?
- What are our use cases?
- Whom are we creating this model for?

1. Business understanding

Key questions:

- What problem do we have? - We want to optimize our campaigns, ...
- What are our use cases? - Should I show this banner?
  - Translate this use case into machine learning - predict whether customer will buy something
- Whom are we creating this model for? - NOT FOR US, our customers!

1. Business understanding

Key questions:

- What problem do we have? - We want to optimize our campaigns, ...
- What are our use cases? - Should I show this banner? ...send this email?
  - Translate this use case into machine learning - predict whether customer will buy something
- Whom are we creating this model for? - NOT FOR US, our customers!

Hard or impossible to automate, but!

We can delegate this to customers or consultants as easier problems.

- Create few templates to choose from
  - At least few easy to use and few advanced to modify the model

2. Data understanding

- Closely related to business understanding
- Most important part of machine learning!
- Key questions
  - What data do i have?
  - Is the data valuable?
  - What do I want to predict?
  - Should I train on all data or only subset? <- crucial
- Changes here can influence outcome by the largest amount

2. **Data understanding**

Automation?

Yes, but difficult...

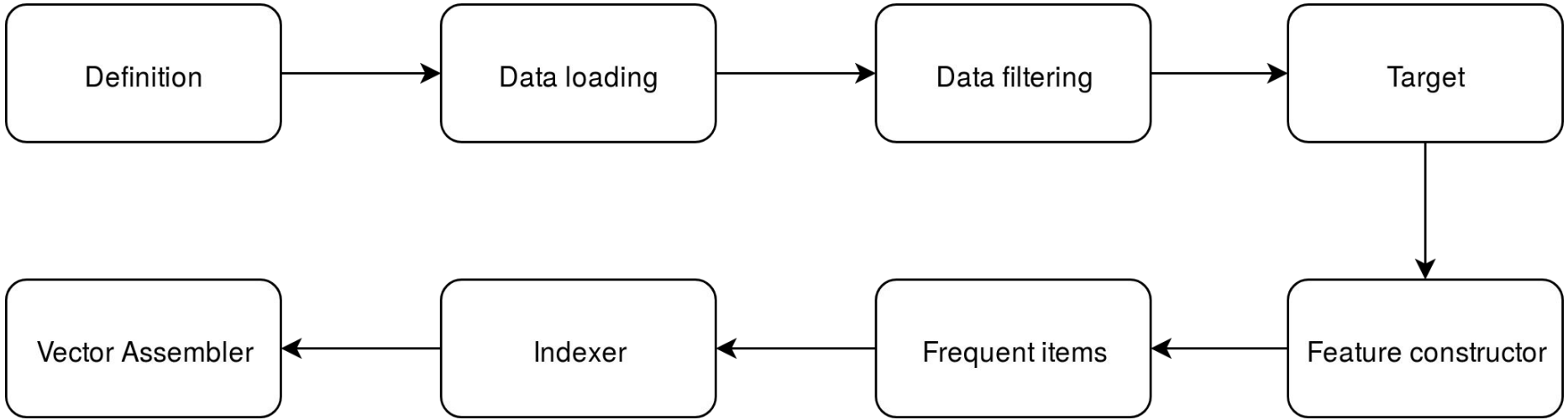
2. Data understanding

- Have pre sets for each template
  - We have in session predictions just for this problem
- Experiment with variables (date ranges, data filters,...)
  - We have variables set to best values for all projects. Can be manually adjusted by user
- Have a model providing unselected variables

3. Data preparation

- Cleaning the data - have some common structure for easy selection
- Derive attributes - have general easy aggregates (count, first, last,...)
- Dataset balancing - easy rules to determine which technique to use
- Aggregates - have a model which selects which complex aggregates to use
- Reducing dimensionality - feature selection
- Mapping data to “clean” values - most frequent items, vector indexer
- Casting data to vectors - vector assembler

3. Data preparation



3. Data preparation

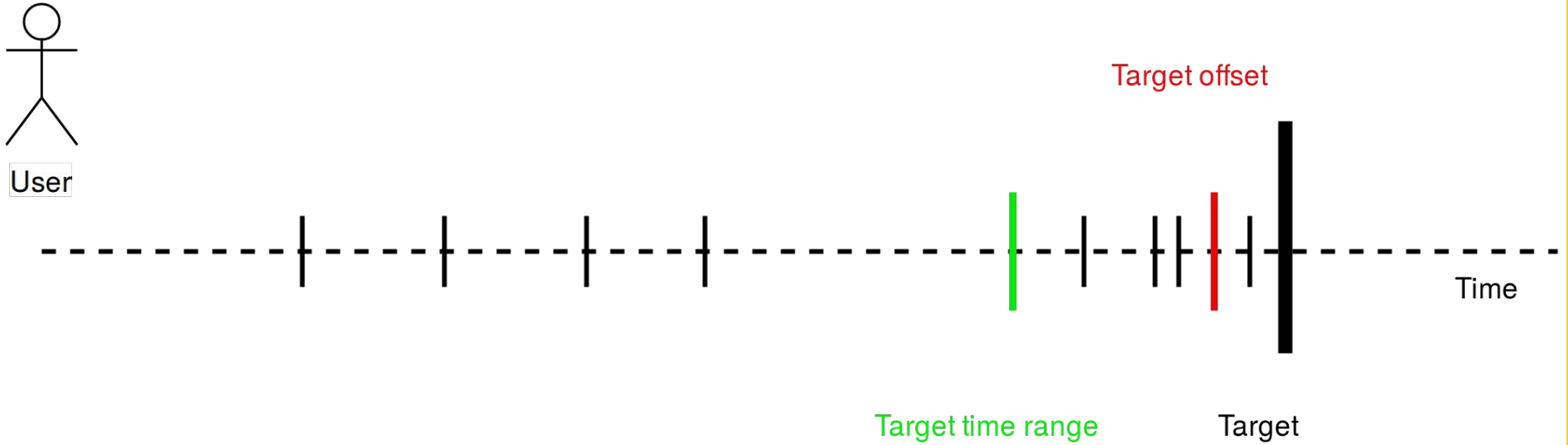

```
{
  "data": {
    "customer_id": "123456789",
    "project_id": "123456",
    "properties": {
      "browser": "Chrome",
      "device": "Other",
      "discount_code": "",
      "items_count": 2,
      "location": "https://randomshop.com/index.php?route=checkout/checkout/success",
      "os": "Windows",
      "payment_method": "PayPal Express",
      "total_price": "25.15392"
    },
    "timestamp": 1509433681.215142,
    "type": "purchase"
  },
  "timestamp": 1509433682.032801,
  "type": "add_event"
}
```

3. Data preparation

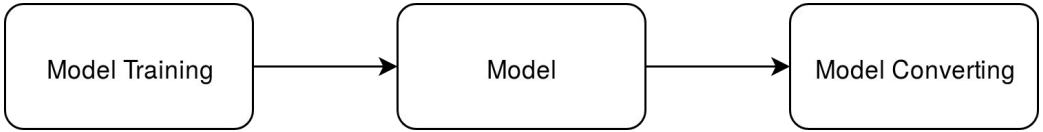
- Customers connect from different platforms
- Customers log in after few events

```
customer_df = customer_df.select("id", "id_history", "properties")
unmerged_customers = customer_df \
        .withColumnRenamed('id', 'original_id') \
        .withColumn("id", explode("id_history")) \
        .select("id", "id_history", "properties", "original_id") \
        .union(customer_df.select("id", "id_history", "properties").withColumn("original_id", customer_df.id))
```

3. Data preparation



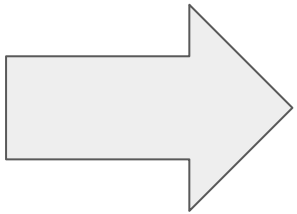
4. Modeling




```
def train(conf, inv_mapping, inv_map_vector_indexer, atts_all, rdd_train_raw, df, atts_cat, min_instances_per_node=50,
          max_depth=5):
    dt = DecisionTreeRegressor(labelCol="target",
                               featuresCol="indexed",
                               minInstancesPerNode=min_instances_per_node,
                               maxDepth=max_depth
                               )

    tree_model = dt.fit(df.select("indexed", 'target'))
    tree_converter = TreeConverter(tree_model, inv_mapping, inv_map_vector_indexer, atts_all)
    json_tree = tree_converter.convert_model()
    importances = tree_model.featureImportances
    predictor_ids = importances.indices
    predictor_importances = importances.values

    predictors = [atts_all[predictor_id] for predictor_id in predictor_ids]
    attributes = get_attributes(predictor_ids, predictor_importances, atts_all)

    enhanced_tree = create_json_tree(conf, rdd_train_raw, json_tree)
    return enhanced_tree, predictors, attributes, json_tree
```

4. Modeling

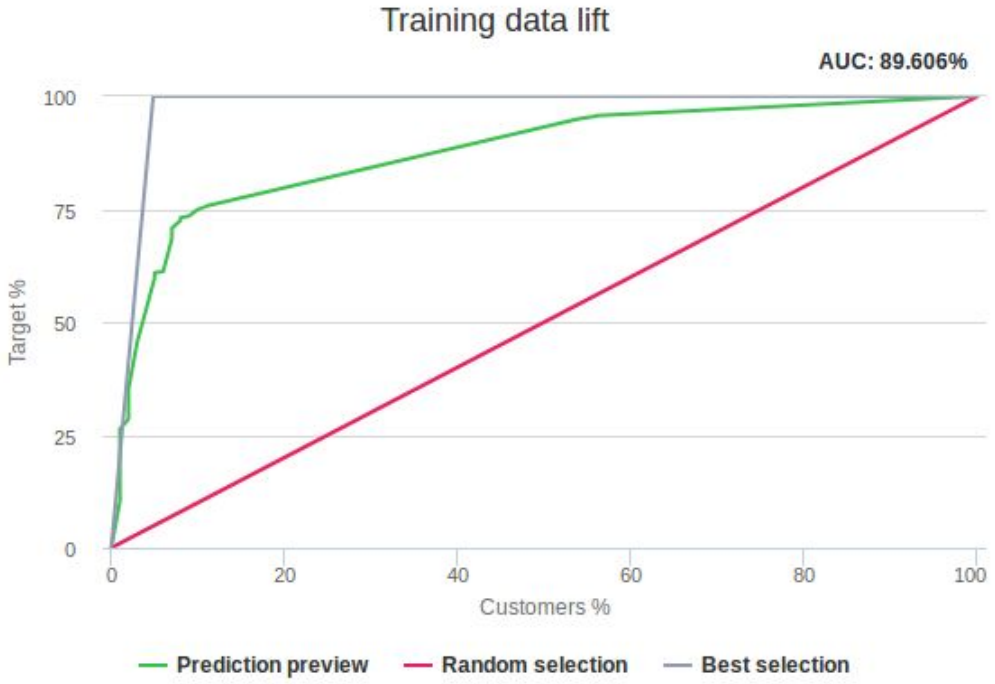
```
('DecisionTreeRegressionModel (uid=DecisionTreeRegressor_
 'of depth 5 with 43 nodes\n'
 '  If (feature 31 in {0.0,3.0,4.0})\n'
 '   If (feature 203 <= 13.0)\n'
 '    If (feature 3 in {5.0,7.0})\n'
 '     If (feature 199 <= 3.0)\n'
 '      If (feature 56 in {0.0,1.0,3.0,6.0,7.0})\n'
 '       Predict: 5.337023127100218E-4\n'
 '      Else (feature 56 not in {0.0,1.0,3.0,6.0,7.0})\n'
 '       Predict: 0.07352941176470588\n'
 '     Else (feature 199 > 3.0)\n'
 '      If (feature 200 <= 12.0)\n'
 '       Predict: 0.14516129032258066\n'
 '      Else (feature 200 > 12.0)\n'
 '       Predict: 0.01694915254237288\n'
 '    Else (feature 3 not in {5.0,7.0})\n'
 '     If (feature 193 <= 26.0)\n'
 '      If (feature 26 in {0.0,2.0,3.0,6.0})\n'
 '       Predict: 0.003889734483341789\n'
 '      Else (feature 26 not in {0.0,2.0,3.0,6.0})\n'
 '       Predict: 0.023227544516722518\n'
 '     Else (feature 193 > 26.0)\n'
 '      If (feature 199 <= 3.0)\n'
 '       Predict: 0.03893983167943726\n'
 '      Else (feature 199 > 3.0)\n'
 '       Predict: 0.16521739130434782\n'
 '   Else (feature 203 > 13.0)\n'
 '    If (feature 236 <= 0.0)\n'
 '     If (feature 205 <= 25.0)\n'
 '      If (feature 8 in {1.0,2.0})\n'
 '       Predict: 0.04894671623296159\n'
 '      Else (feature 8 not in {1.0,2.0})\n'
 '       Predict: 0.09539656128674431\n'
 '     Else (feature 205 > 25.0)\n'
 '      If (feature 3 in {1.0,5.0,6.0,8.0})\n'
 '       Predict: 0.01904761904761905\n'
 '      Else (feature 3 not in {1.0,5.0,6.0,8.0})\n'
 '       Predict: 0.15716486902927582\n'
 '    Else (feature 236 > 0.0)\n'
```



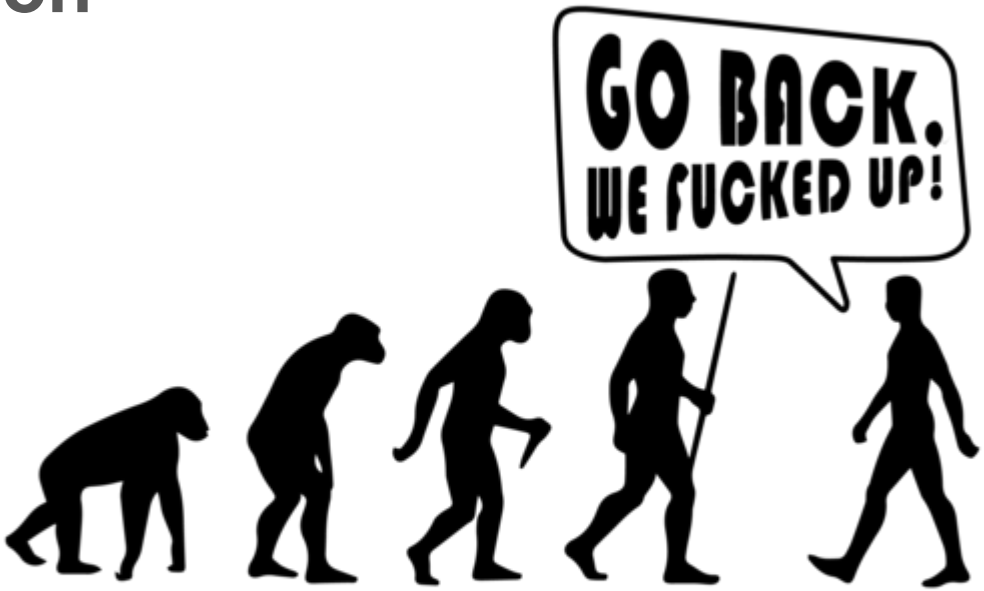
```
lDict([('attribute_id', None),
       ('attribute', 'Root'),
       ('attribute_name', 'Root'),
       ('threshold', None),
       ('prediction', 0.028257350303899805),
       ('cats', None),
       ('rules', 'All (All %) ... Positive %'),
       ('node_id', 1),
       ('split_type', None),
       ('negation', None),
       ('condition', None),
       ('children',
        [OrderedDict([('attribute_id', 31),
                      ('attribute',
                       'campaign_L_campaign_policy'),
                      ('attribute_name',
                       'campaign_L_campaign_policy'),
                      ('threshold', None),
                      ('prediction',
                       0.016529412342092O9),
                      ('cats',
                       ['newsletter', '', '<n/a>']),
                      ('rules',
                       'campaign_L_campaign_policy in '
                       '{newsletter, , <n/a>}'),
                      ('node_id', 2),
                      ('split_type', 'categorical'),
                      ('negation', False),
                      ('condition', 'in'),
                      ('children',
                       [OrderedDict([('attribute_id',
                                      203),
                                     ('attribute',
                                      'page_visit_C_7day'),
                                     ('attribute_name',
                                      'page_visit_C_7day'),
                                     ('threshold',
                                      13.0),
                                     ('prediction',
                                      0.011574389566274913),
                                     ('cats', []),
                                     ('rules',
                                      'page_visit_C_7day '
                                      '<= 13.0'),
```

5. Evaluation

- How good is my model?
- ROC, F1, Lift curve



5. Evaluation



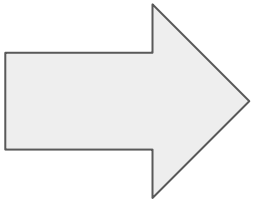
6. **Deployment**

How do you want to use it? Must it be fast? Online or Offline?

6. **Deployment**

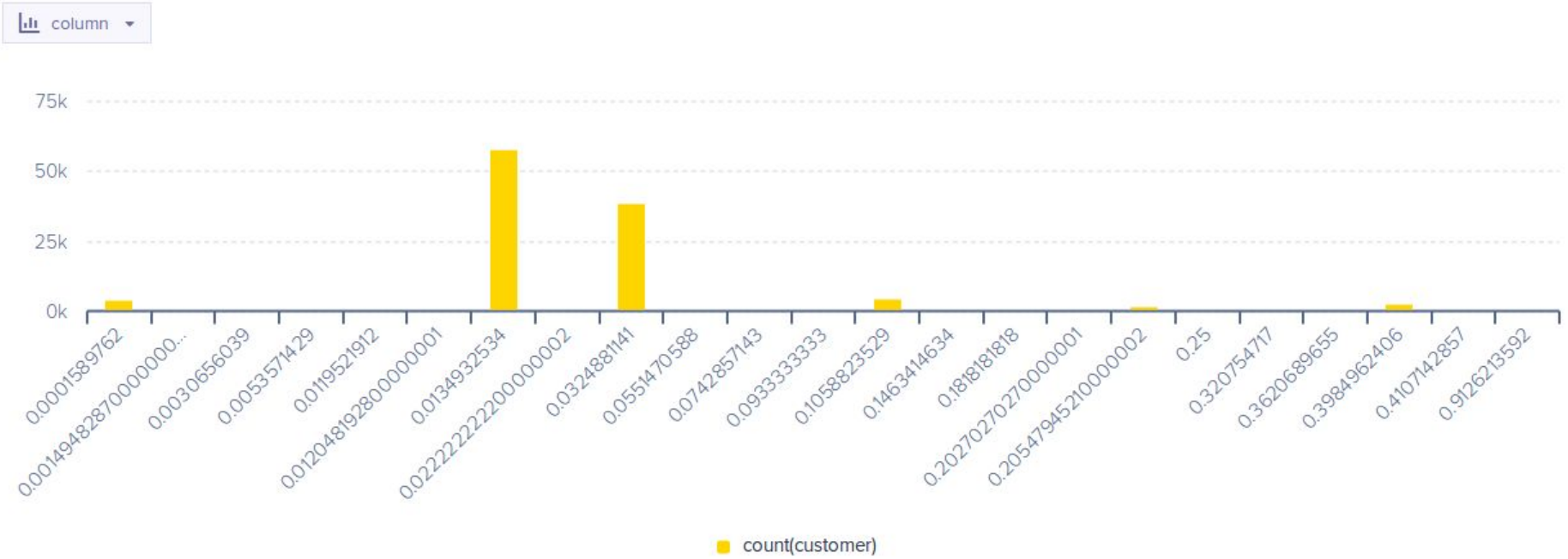
For us, Fast and Online!

```
lDict([('attribute_id', None),
       ('attribute', 'Root'),
       ('attribute_name', 'Root'),
       ('threshold', None),
       ('prediction', 0.028257350303899805),
       ('cats', None),
       ('rules', 'All (All %) ... Positive %'),
       ('node_id', 1),
       ('split_type', None),
       ('negation', None),
       ('condition', None),
       ('children',
        [OrderedDict([('attribute_id', 31),
                      ('attribute',
                       'campaign_L_campaign_policy'),
                      ('attribute_name',
                       'campaign_L_campaign_policy'),
                      ('threshold', None),
                      ('prediction',
                       0.01652941234209209),
                      ('cats',
                       ['newsletter', '', '<n/a>']),
                      ('rules',
                       'campaign_L_campaign_policy in '
                       '{newsletter, , <n/a>}'),
                      ('node_id', 2),
                      ('split_type', 'categorical'),
                      ('negation', False),
                      ('condition', 'in'),
                      ('children',
                       [OrderedDict([('attribute_id',
                                      203),
                                     ('attribute',
                                      'page_visit_C_7day'),
                                     ('attribute_name',
                                      'page_visit_C_7day'),
                                     ('threshold',
                                      13.0),
                                     ('prediction',
                                      0.011574389566274913),
                                     ('cats', []),
                                     ('rules',
                                      'page_visit_C_7day '
                                      '<= 13.0'),
```



```
                                                'type': 'customer'},
                                               {'filter': {'attribute': {'expression': {'formula': 'ifnull(V0, '
                                                                                                   'V1)',
                                                                                        'members': [{'attribute': {'aggregate': {
                                                                                                                   'attribute': {'property': 'total_quantity',
                                                                                                                                 'type': 'property'},
                                                                                                                   'event': {'filter': [],
                                                                                                                             'type': 'cart_update'},
                                                                                                                   'independent_if_reused': True,
                                                                                                                   'type': 'last'},
                                                                                                                                 'type': 'embedded_aggregate'},
                                                                                                                   'type': 'customer'},
                                                                                                                  {'type': 'constant',
                                                                                                                   'value': 0,
                                                                                                                   'value_type': 'number'}]},
                                                                                   'type': 'embedded_expression'},
                                                                'constraint': {'operands': [{'type': 'constant',
                                                                                             'value': 2.0}],
                                                                               'operator': 'greater '
                                                                                           'than',
                                                                               'type': 'number'},
                                                                'type': 'attribute'},
                                                 'type': 'customer'}],
                                    'formula': 'if F0 then if not F1 then if F2 then if not F3 '
                                               'then if not F4 then V0 else V1 end else if not F5 '
                                               'then V2 else V3 end end else if not F6 then if F7 '
                                               'then V4 else V5 end else if not F3 then V6 else '
                                               'V7 end end end else if not F8 then if not F9 then '
                                               'if F10 then V8 else V9 end else if F11 then V10 '
                                               'else V11 end end else if F12 then if not F13 then '
                                               'V12 else V13 end else if F14 then V14 else V15 '
                                               'end end end end else if F15 then V16 else if F16 '
                                               'then V17 else if F17 then if F18 then V18 else '
                                               'V19 end else if not F19 then V20 else V21 end end '
                                               'end end end',
                                    'members': [{'type': 'constant',
                                                 'value': 0.0005337023127100218,
                                                 'value_type': 'number'},
                                                {'type': 'constant',
                                                 'value': 0.07352941176470588,
                                                 'value_type': 'number'},
                                                {'type': 'constant',
                                                 'value': 0.14516129032258066,
                                                 'value_type': 'number'},
                                                {'type': 'constant',
                                                 'value': 0.01694915254237288,
                                                 'value_type': 'number'},
                                                {'type': 'constant',
```

6. Deployment

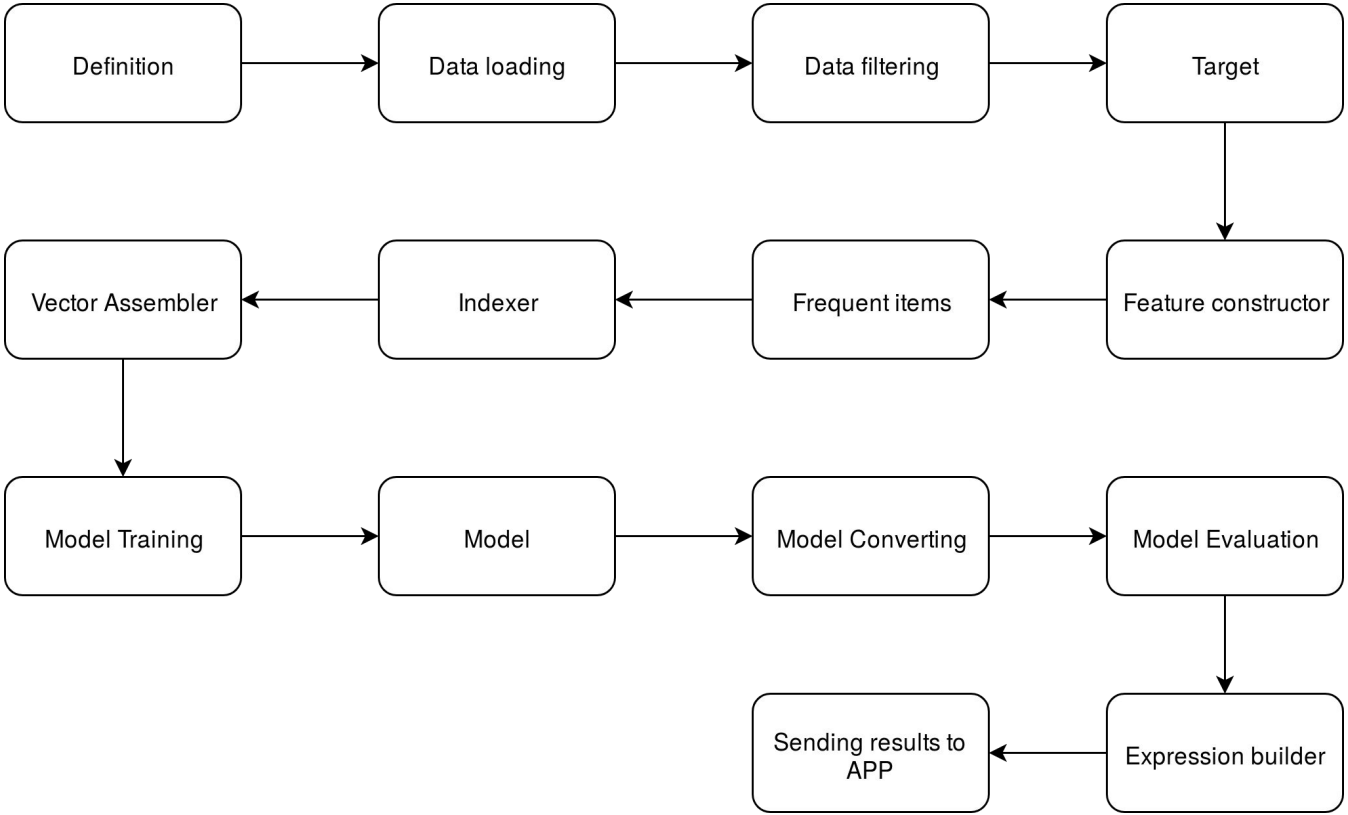


6. Deployment

Other deployments:

- Fast and Offline -> pre computed results for each customer
- Slow and Online -> improved results with slower algorithms
- Slow and Offline -> you messed up. Rework your solution.

App overview



Ask me anything

ondrej.brichta@exponea.com

EXPONEA

MARKETING CLOUD

WE ARE HIRING