FEATURE STORE SUMMIT

12-13 OCTOBER | 08:30 AM - 4:00 PM PT

ORGANIZED BY HOPSWORKS

getML

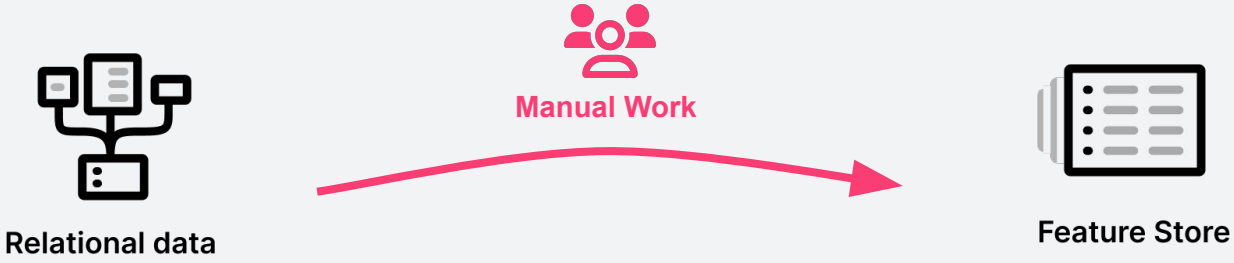
Why Relational Learning Matters

Automated Feature Engineering on Relational Data and Time Series

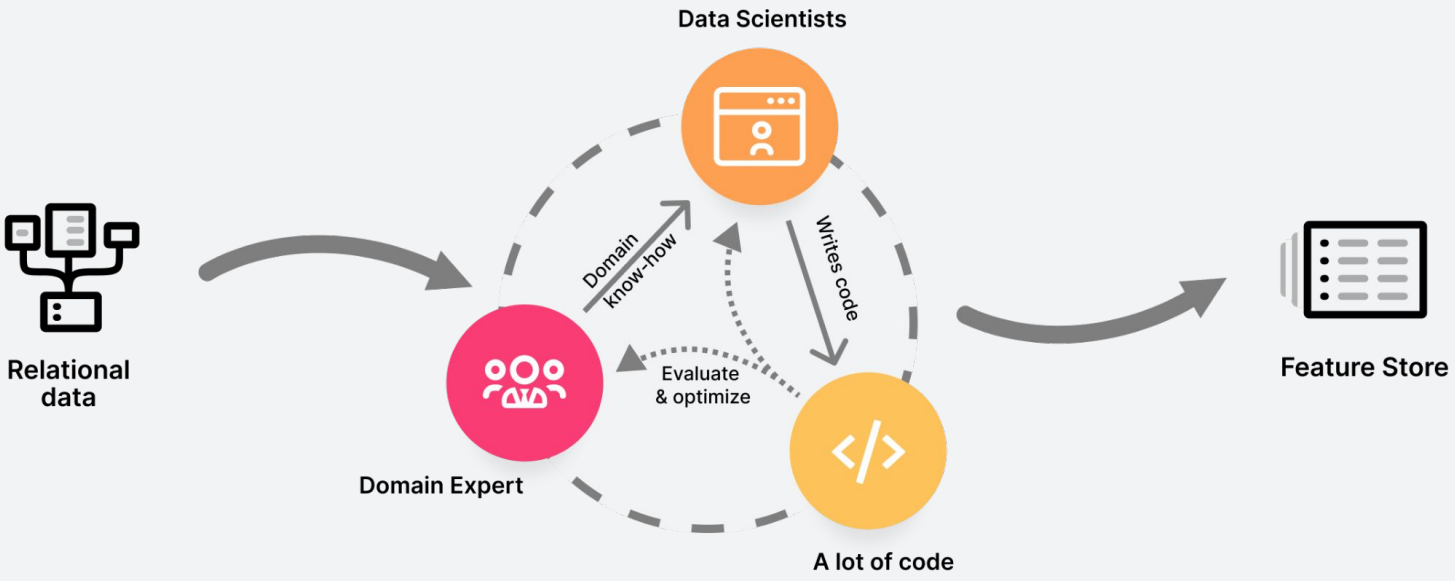
Dr. Patrick Urabanke
CTO

getML.com

Feature engineering is an expensive process



This is how you spend 90% of your time



getML

Feature Stores for ML

getML automates that using relational learning

An introduction to automated feature engineering

Propositionalization

Multi-Relational Decision Trees

Relational Boosting

getML

Feature Stores for ML

Propositionalization

Feature engineering by brute force

```
AVG( t2.col_1 )  AS feature_1
FROM POPULATION_TABLE t1
LEFT JOIN TRANSACTIONS t2
ON t1.customer_id = t2.customer_id
GROUP BY t2.customer_id;


SUM( t2.col_1 )  AS feature_2
FROM POPULATION_TABLE t1
LEFT JOIN TRANSACTIONS t2
ON t1.customer_id = t2.customer_id
GROUP BY t2.customer_id;
```

getML

Feature Stores for ML

Propositionalization

PRO

- Simple, interpretable features, similar to manual features
- Easy to implement
- Many implementations (featuretools, tsfresh, H2O, XPanse Analytics, dot.data, getML, ...)

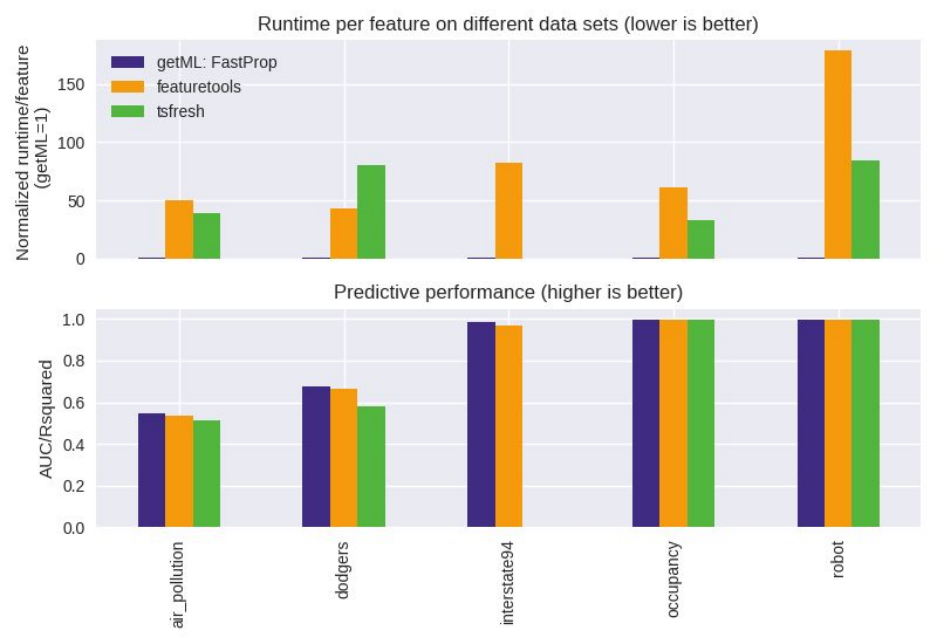
CON

- Feature explosion
- Hard to implement efficiently
- Does not capture complex relationships
- Many garbage features

Propositionalization is hard to implement efficiently

getML FastProp

Our custom-built database engine is **34x to 179x faster** than implementations based on pandas.



https://getml.com/blog/article/introducing-fast-prop-the-fastest-approach-to-propositionalization

getML

Feature Stores for ML

MRTDL
Multi-Relational Decision Tree Learning

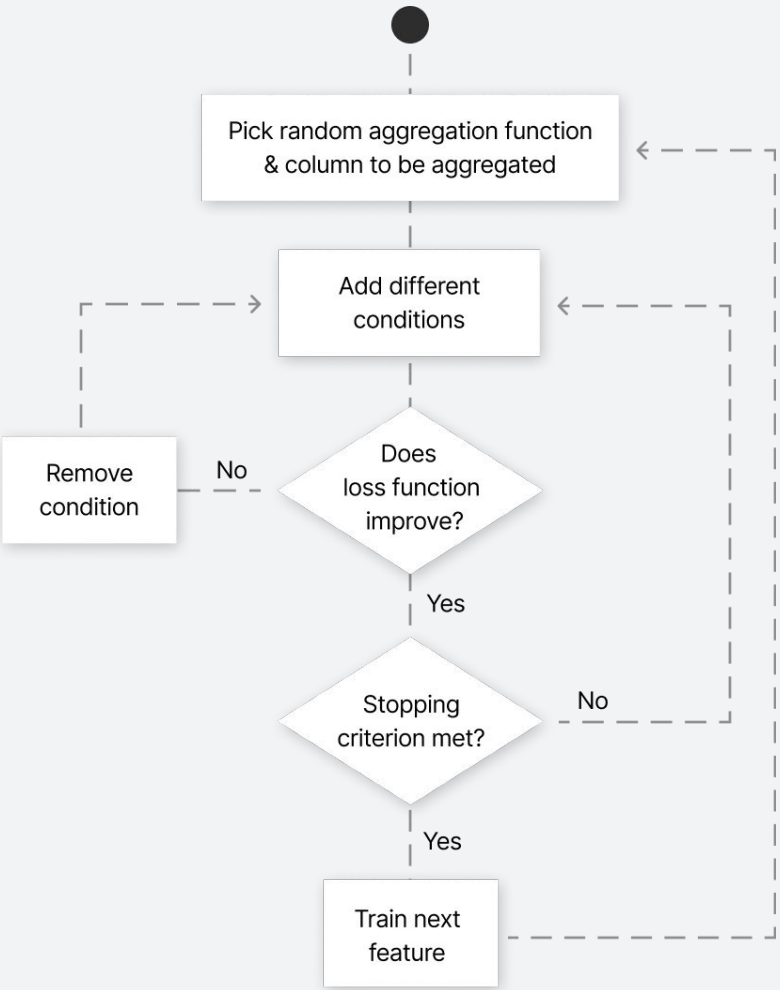
```
SUM( t2.col_1 ) AS feature_1
    FROM POPULATION_TABLE t1
    LEFT JOIN TRANSACTIONS t2
    ON t1.customer_id = t2.customer_id
    WHERE t1.reference_date - t2.transaction_date <= 90.0
AND   t2.transaction_type IN ('sala', 'rent', 'cc')
GROUP BY t2.customer_id;
```

getML

Feature Stores for ML

MRTDL

The algorithm



getML

Feature Stores for ML

MRTDL
Multi-Relational Decision Tree Learning

PRO

- Captures more complex relationships
- Features still reasonably close to manual features

CON

- Feature explosion
- Hard to implement efficiently
- Information might be lost due to greediness

Feature explosion is an underestimated problem

```
SOME_AGGREGATION( some_column )
FROM SOME_TABLE t1
LEFT JOIN SOME_OTHER_TABLE t2
ON t1.join_key = t2.join_key
WHERE some_condition BASED ON some_other_column
GROUP BY t2.join_key;
```

Problem:

The feature space grows **quadratically** with the number of columns.
This also affects manual feature engineering.

Relboost: Aggregate learnable weights

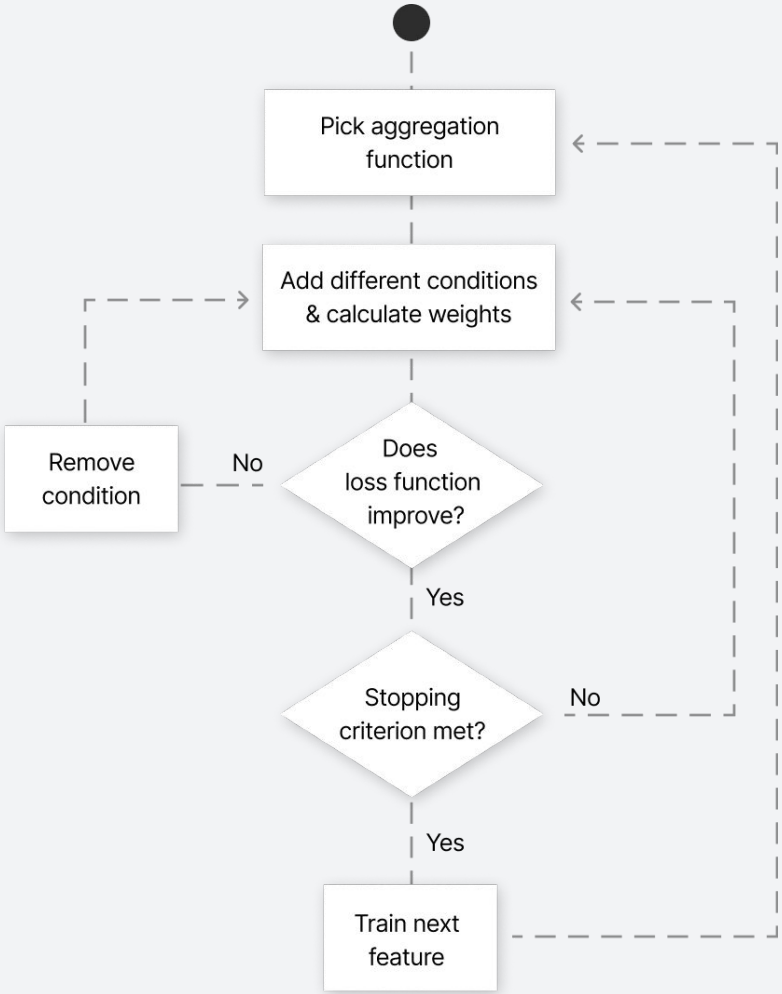
```
SOME_AGGREGATION(
  CASE WHEN some_condition THEN some_weight
  CASE WHEN some_other_condition THEN some_other_weight
) FROM SOME_TABLE t1
LEFT JOIN SOME_OTHER_TABLE t2
ON t1.join_key = t2.join_key
GROUP BY t2.join_key;
```

Solution:

The feature space grows **linearly** with the number of columns.

Relboost

The algorithm



getML

Feature Stores for ML

Relboost

PRO

- No feature explosion
- Can build any number of features
- Builds on state-of-the-art machine learning paradigms

CON

- Does not allow for more complex aggregation (COUNT DISTINCT, MEDIAN, ...)
- Harder to interpret

Relboost performs best on complex data sets

	Number of columns	Accuracy: **FastProp**	Accuracy: **Relboost**
CORA	2	89.9%	89.9%
MOVIELENS	13	77.8%	81.6%

https://nbviewer.getml.com/github/getml/getml-demo/blob/master/cora.ipynb

https://nbviewer.getml.com/github/getml/getml-demo/blob/master/movie_lens.ipynb

Live demo
Customer Churn Prediction

Demo notebook
https://notebooks.getml.com/github/getml/getml-demo/blob/master/adventure_works.ipynb

Data set
Microsoft - AdventureWorks
github.com/microsoft/sql-server-samples

getML

Feature Stores for ML

Thank you!

Dr. Patrick Urbanke • patrick@getml.com

linkedin.com/company/getml

https://getml.com

getML

ML framework
for relational learning

More notebooks & benchmarks - https://github.com/getml/getml-demo/

Web-hosted live demo - https://demo.getml.com/

Step 1: Get the data

```
In [2]: conn = getml.database.connect_mariadb(
            host="relational.fit.cvut.cz",
            dbname="AdventureWorks2014",
            port=3306,
            user="guest",
            password="relational"
        )

        conn

Out[2]: Connection(conn_id='default',
                   dbname='AdventureWorks2014',
                   dialect='mysql',
                   host='relational.fit.cvut.cz',
                   port=3306)
```

getML

Feature Stores for ML

Step 2: Assign roles

In [14]:
```
special_offer.set_role(["SpecialOfferID"], getml.data.roles.join_key)
special_offer.set_role(["MinQty", "DiscountPct"], getml.data.roles.numerical)
special_offer.set_role(["Category", "Description", "Type"], getml.data.roles.categorical)
special_offer.set_role(["StartDate", "EndDate"], getml.data.roles.time_stamp)

special_offer
```

Out[14]:

name	StartDate	EndDate	SpecialOfferID	Category	Description	Type	MinQty	DiscountPct	MaxQty	rowguid	ModifiedDate
role	time_stamp	time_stamp	join_key	categorical	categorical	categorical	numerical	numerical	unused_float	unused_string	unused_string
unit	time stamp, comparison only	time stamp, comparison only									
0	2011-05-01	2014-11-30	1	No Discount	No Discount	No Discount	0	0	nan	0290C4F5-191F-4337-AB6B-0A2DDE03...	2011-04-01 00:00:00
1	2011-05-31	2014-05-30	2	Reseller	Volume Discount 11 to 14	Volume Discount	11	0.02	14	D7542EE7-15DB-4541-985C-5CC27AEF...	2011-05-01 00:00:00
2	2011-05-31	2014-05-30	3	Reseller	Volume Discount 15 to 24	Volume Discount	15	0.05	24	4BDBCC01-8CF7-40A9-B643-40EC5B71...	2011-05-01 00:00:00
3	2011-05-31	2014-05-30	4	Reseller	Volume Discount 25 to 40	Volume Discount	25	0.1	40	504B5E85-8F3F-4EBC-9E1D-C1BC5DEA...	2011-05-01 00:00:00

getML

Feature Stores for ML

Step 3: Define training and testing sets

In [20]:
```
container = getml.data.Container(population=sales_order_header, split=split)

container.add(
    product=product,
    sales_order_detail=sales_order_detail,
    sales_order_header=sales_order_header,
    sales_order_reason=sales_order_reason,
    special_offer=special_offer,
    store=store,
)

container
```

Out[20]:

population

	subset	name	rows	type
0	test	SalesOrderHeaderRefined	3879	View
1	train	SalesOrderHeaderRefined	15825	View

peripheral

	alias	name	rows	type
0	product	Product	504	DataFrame
1	sales_order_detail	SalesOrderDetail	121317	DataFrame
2	sales_order_header	SalesOrderHeaderRefined	19704	DataFrame
3	sales_order_reason	SalesOrderHeaderSalesReason	27647	DataFrame
4	special_offer	SpecialOffer	16	DataFrame
5	store	Store	701	DataFrame

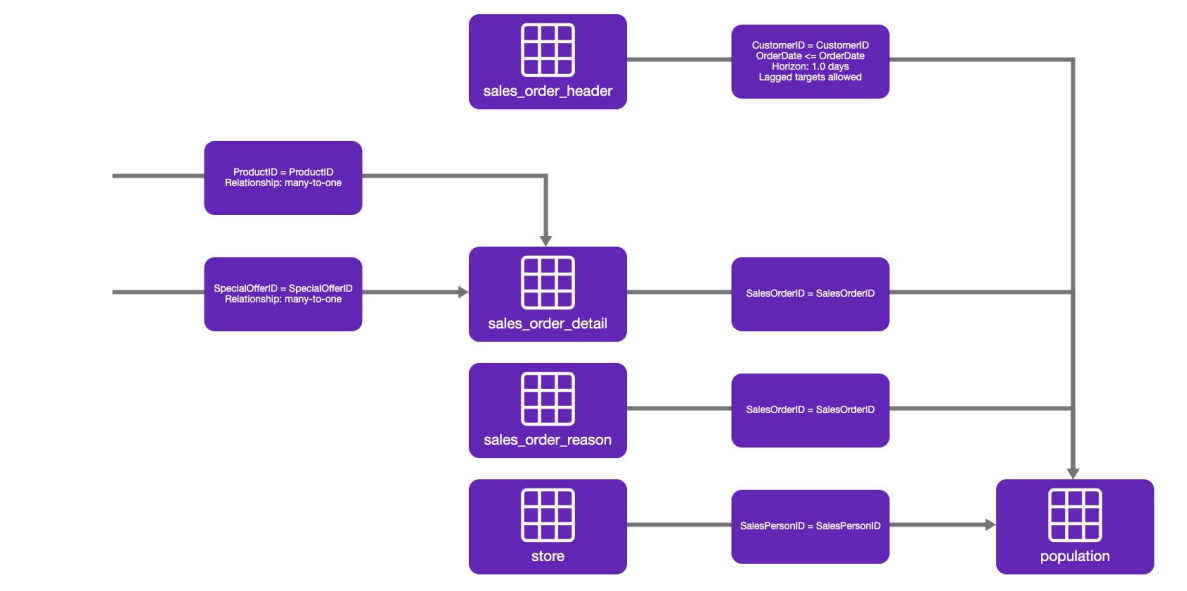
Step 4: Define the data model

```
In [21]: dm = getml.data.DataModel(sales_order_header.to_placeholder("population"))

         dm.add(getml.data.to_placeholder(
             product=product,
             sales_order_detail=sales_order_detail,
             sales_order_header=sales_order_header,
             sales_order_reason=sales_order_reason,
             special_offer=special_offer,
             store=store,
         ))


         dm.population.join(
             dm.sales_order_header,
             on="CustomerID",
             time_stamps="OrderDate",
             lagged_targets=True,
             horizon=getml.data.time.days(1),
         )
```

getML

Feature Stores for ML

Step 4: Define the data model



getML

Feature Stores for ML

Step 5: Define the pipeline

In [24]:
```
pipe2 = getml.Pipeline(
    tags=['relboost'],
    data_model=dm,
    preprocessors=[seasonal, mapping],
    feature_learners=[relboost],
    predictors=[predictor],
    include_categorical=True,
)

pipe2
```

Out[24]:
```
Pipeline(data_model='population',
         feature_learners=['Relboost'],
         feature_selectors=[],
         include_categorical=True,
         loss_function=None,
         peripheral=['product', 'sales_order_detail', 'sales_order_header',
                     'sales_order_reason', 'special_offer', 'store'],
         predictors=['XGBoostClassifier'],
         preprocessors=['Seasonal', 'Mapping'],
         share_selected_features=0.5,
         tags=['relboost'])
```

Step 6: Fit the pipeline

```
In [*]: pipe2.fit(container.train)

Checking data model...


Staging...
[========================================] 100%


INFO [FOREIGN KEYS NOT FOUND]: When joining POPULATION__STAGING_TABLE_1 and SALES_ORDER_REASON__STAGING_TABLE_4 over
'SalesOrderID' and 'SalesOrderID', there are no corresponding entries for 33.769352% of entries in 'SalesOrderID' in
'POPULATION__STAGING_TABLE_1'. You might want to double-check your join keys.
INFO [FOREIGN KEYS NOT FOUND]: When joining POPULATION__STAGING_TABLE_1 and STORE__STAGING_TABLE_5 over 'SalesPersonI
D' and 'SalesPersonID', there are no corresponding entries for 84.941548% of entries in 'SalesPersonID' in 'POPULATIO
N__STAGING_TABLE_1'. You might want to double-check your join keys.


Staging...
[========================================] 100%

Preprocessing...
[========================================] 100%

Relboost: Training features...
[============                            ] 32%
```

getML

Feature Stores for ML

Step 7: Evaluate your results

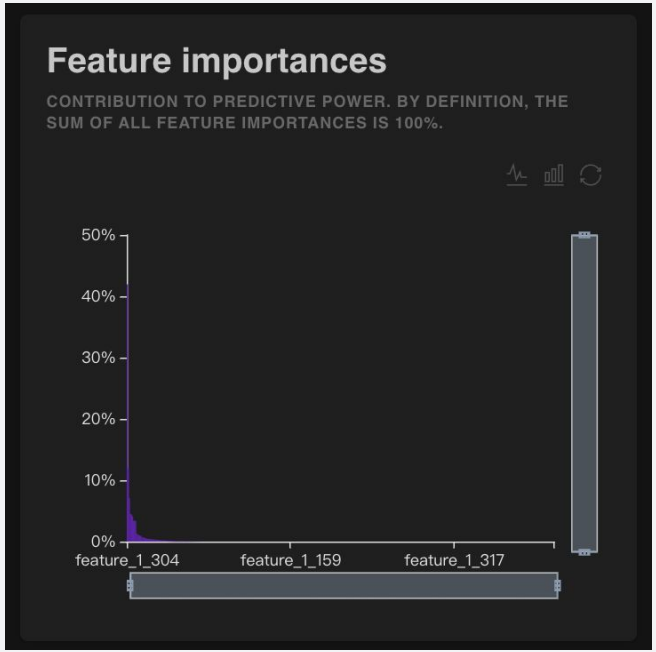
Fitted pipelines

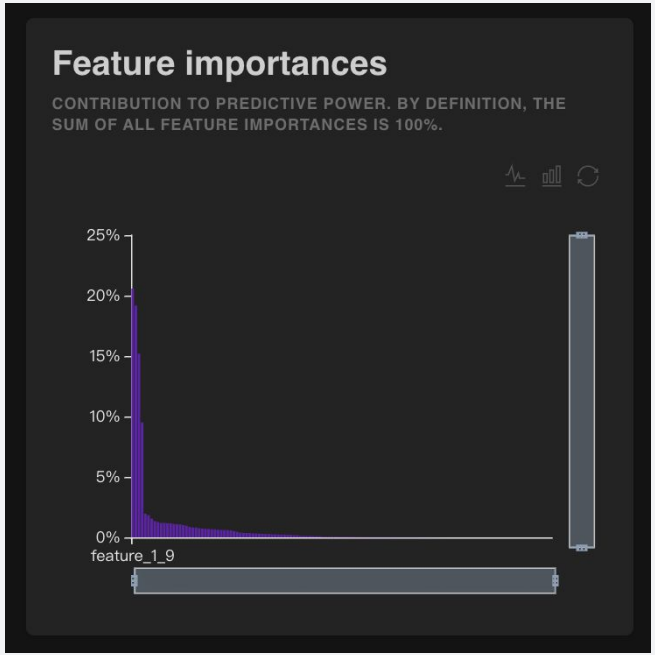
PREDICTIVE PERFORMANCE OF PIPELINES THAT HAVE ALREADY BEEN FITTED

ID	Tags	Created	Accuracy	AUC	Cross entropy	Targets	Set used
RfvhrY	relboost, container-vjRbft	2021-10-03 12:29:32	0.927	0.9781	0.1906	churn	test
M75uIB	fast_prop, container-vjRbft	2021-10-03 12:28:17	0.9129	0.9712	0.2236	churn	test

Rows per page: 5 1-2 of 2

Step 8: Study your features





getML

Feature Stores for ML

Step 8: Study your features

```sql
DROP TABLE IF EXISTS "FEATURE_1_304";

CREATE TABLE "FEATURE_1_304" AS
SELECT MIN( t1."orderdate" - t2."t4__startdate" ) AS "feature_1_304",
       t1.rowid AS "rownum"
FROM "POPULATION__STAGING_TABLE_1" t1
LEFT JOIN "SALES_ORDER_DETAIL__STAGING_TABLE_2" t2
ON t1."salesorderid" = t2."salesorderid"
GROUP BY t1.rowid;
```

Step 8: Study your features

```
DROP TABLE IF EXISTS "FEATURE_1_9";

CREATE TABLE "FEATURE_1_9" AS
SELECT AVG(
    CASE
        WHEN ( t1."orderdate" - t2."t4__startdate" > 66070588.235294 ) AND ( t1."salespersonidcat" IN ( '279', '282', '276', '280', '283', '277', '275', '278', '281', '289'
        WHEN ( t1."orderdate" - t2."t4__startdate" > 66070588.235294 ) AND ( t1."salespersonidcat" IN ( '279', '282', '276', '280', '283', '277', '275', '278', '281', '289'
        WHEN ( t1."orderdate" - t2."t4__startdate" > 66070588.235294 ) AND ( t1."salespersonidcat" NOT IN ( '279', '282', '276', '280', '283', '277', '275', '278', '281', '
        WHEN ( t1."orderdate" - t2."t4__startdate" > 66070588.235294 ) AND ( t1."salespersonidcat" NOT IN ( '279', '282', '276', '280', '283', '277', '275', '278', '281', '
        WHEN ( t1."orderdate" - t2."t4__startdate" <= 66070588.235294 OR t1."orderdate" IS NULL OR t2."t4__startdate" IS NULL ) AND ( t1."territoryidcat" IN ( '2', '10', '9
        WHEN ( t1."orderdate" - t2."t4__startdate" <= 66070588.235294 OR t1."orderdate" IS NULL OR t2."t4__startdate" IS NULL ) AND ( t1."territoryidcat" IN ( '2', '10', '9
        WHEN ( t1."orderdate" - t2."t4__startdate" <= 66070588.235294 OR t1."orderdate" IS NULL OR t2."t4__startdate" IS NULL ) AND ( t1."territoryidcat" NOT IN ( '2', '10'
        WHEN ( t1."orderdate" - t2."t4__startdate" <= 66070588.235294 OR t1."orderdate" IS NULL OR t2."t4__startdate" IS NULL ) AND ( t1."territoryidcat" NOT IN ( '2', '10'
        ELSE NULL
    END
) AS "feature_1_9",
     t1.rowid AS "rownum"
FROM "POPULATION__STAGING_TABLE_1" t1
LEFT JOIN "SALES_ORDER_DETAIL__STAGING_TABLE_2" t2
ON t1."salesorderid" = t2."salesorderid"
GROUP BY t1.rowid;
```

getML

Feature Stores for ML