SmartData

From click to predict and back: ML pipelines at OK

Dmitry Bugaychenko

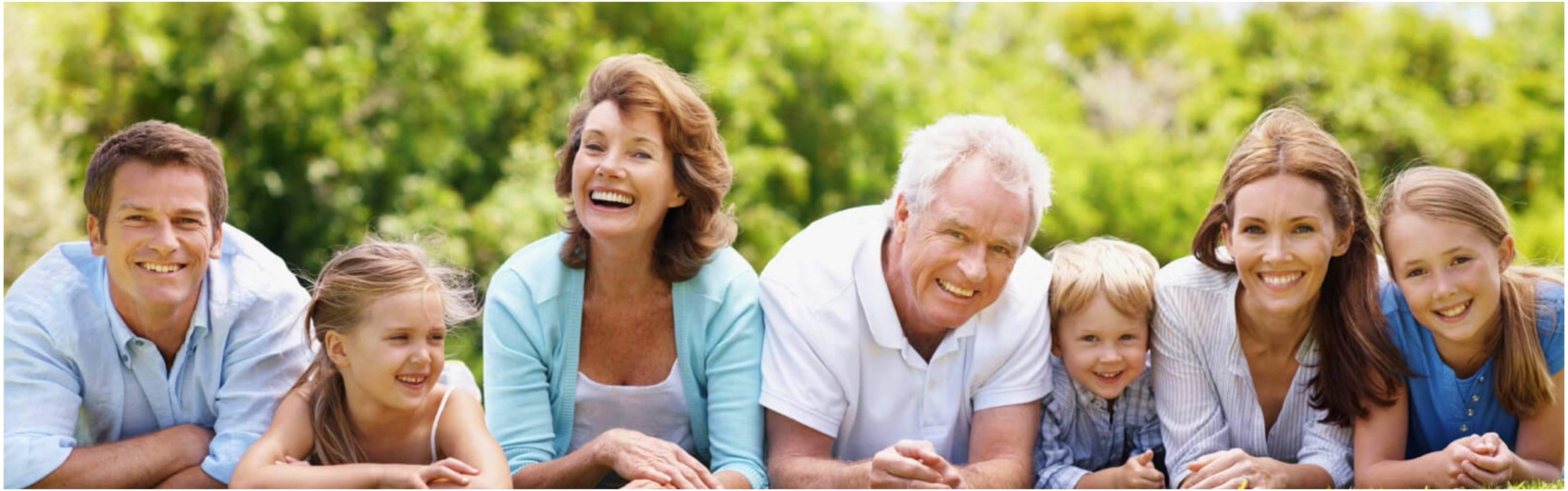
@mail.ru group

OK is...

70 000 000+

monthly unique users

OK is…



800 000 000+ family links in the social graph

OK is…

A place where people share their positive feelings

OK is...

- 10000+ servers around the globe
- 1+Tb/s of outgoing traffic
- 400+ software components
- High Load, Big Data, Fault Tolerance...

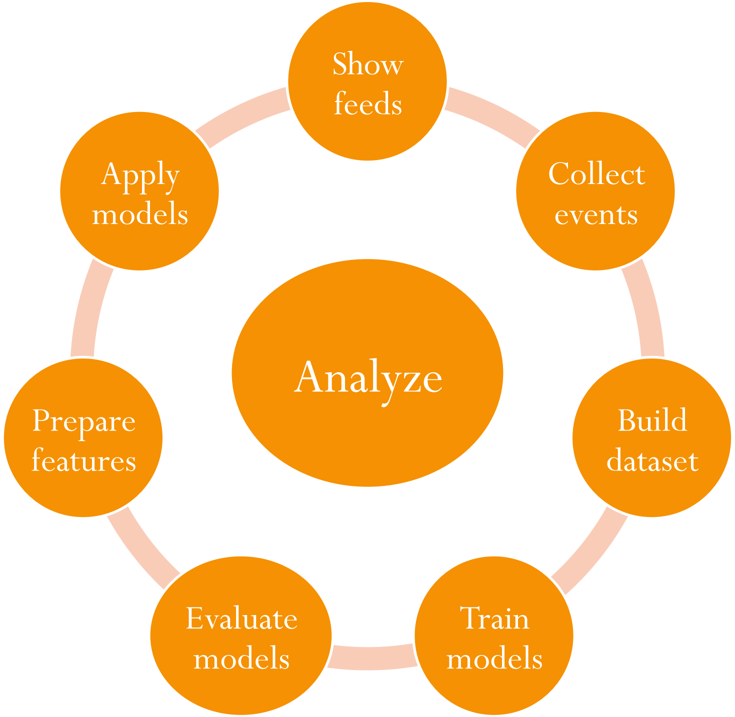
OK is…

- 3 Hadoop clusters
- 30+ petabytes storage (+16Tb daily)
- 10000+ cores
- 40+ TB RAM
- 300+ regular jobs

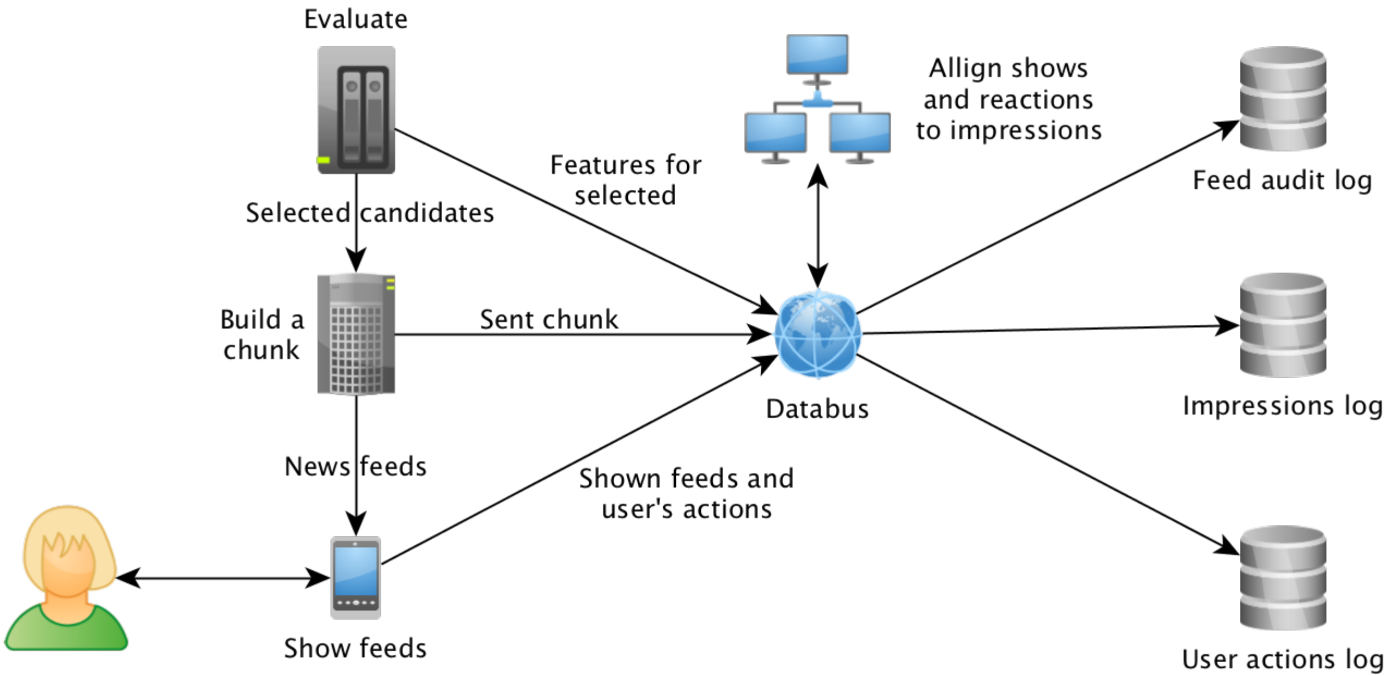
News feed at OK

- 14 000 000 000 news feed records sent to users daily
- 1000+ servers involved in preparation
  - Collect 400 000+ of impressions per second
  - Build a 10 000 000 000+ records dataset
  - Train 5000+ personalization models
  - Extract features in real time handling 8 000 000+ reads per second
  - Store features for 1 500 000 000+ objects
  - Evaluate 5 500 000+ candidates per second
  - Store 3 000 000+ selected records per second

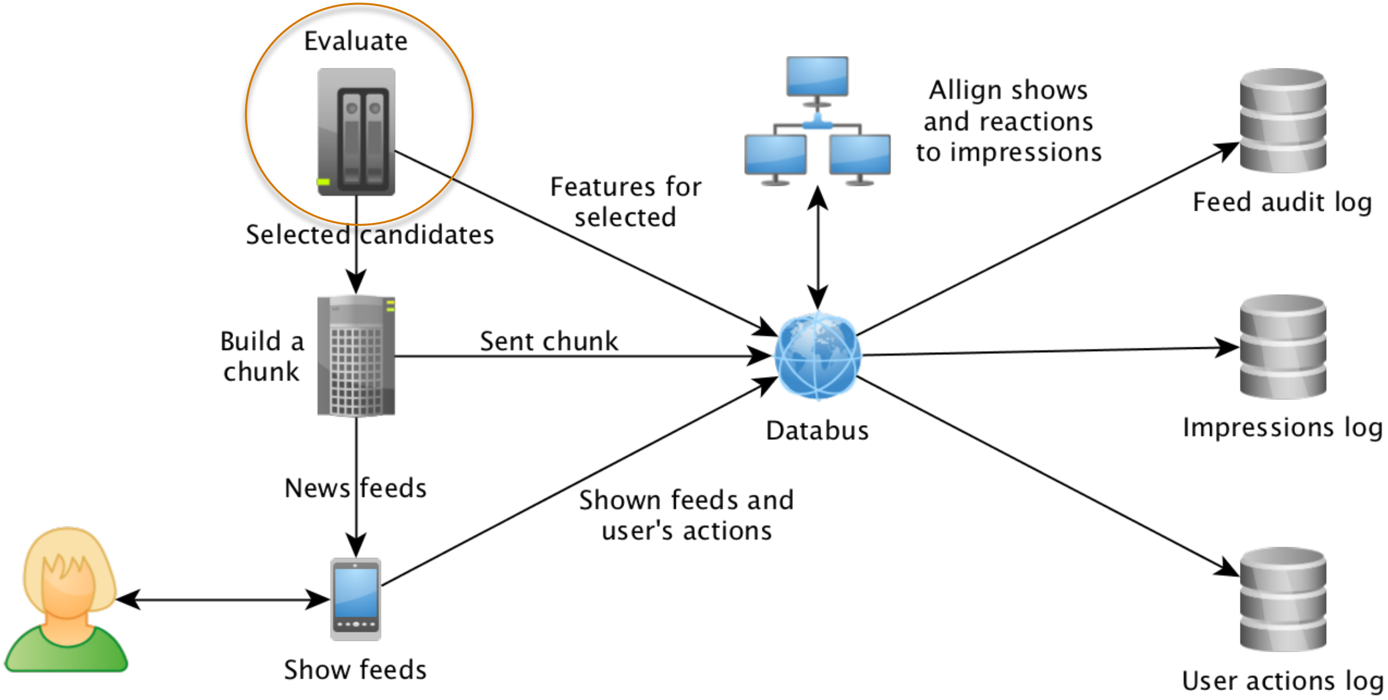
News feed preparation at OK



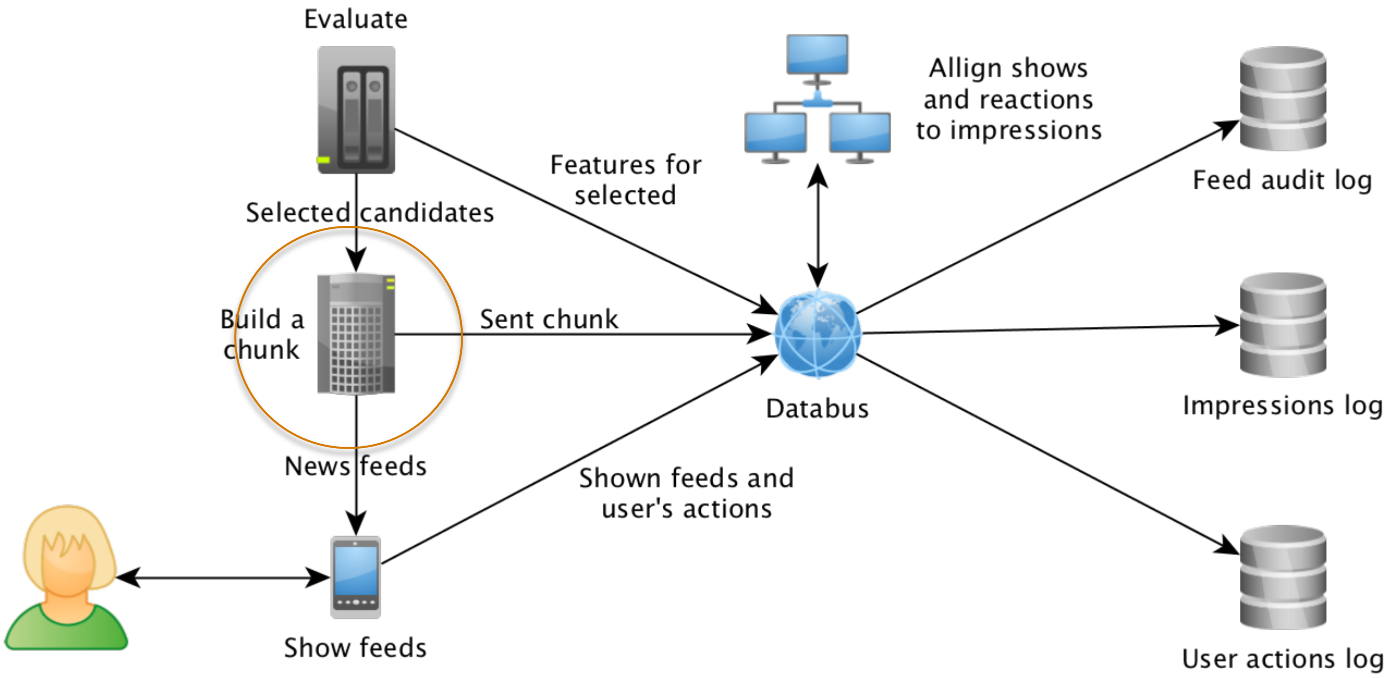
Show feeds and collect data



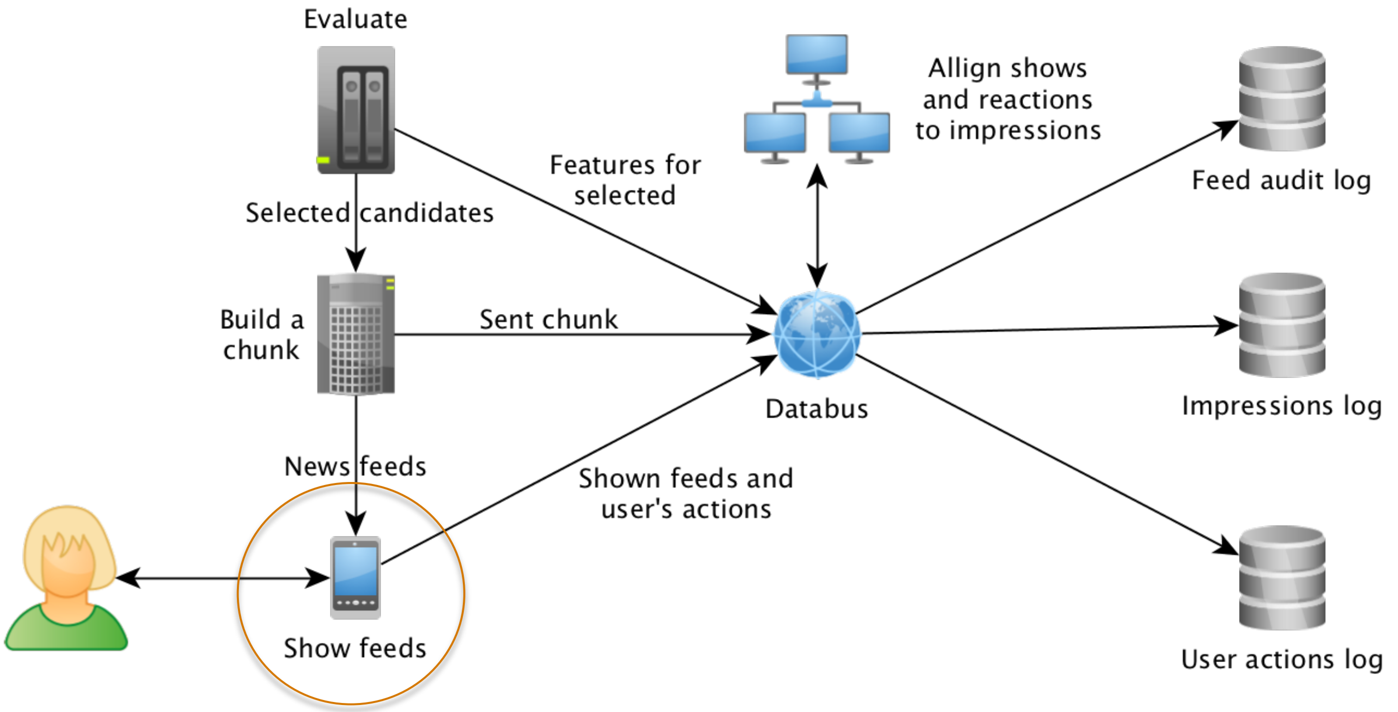
Show feeds and collect data



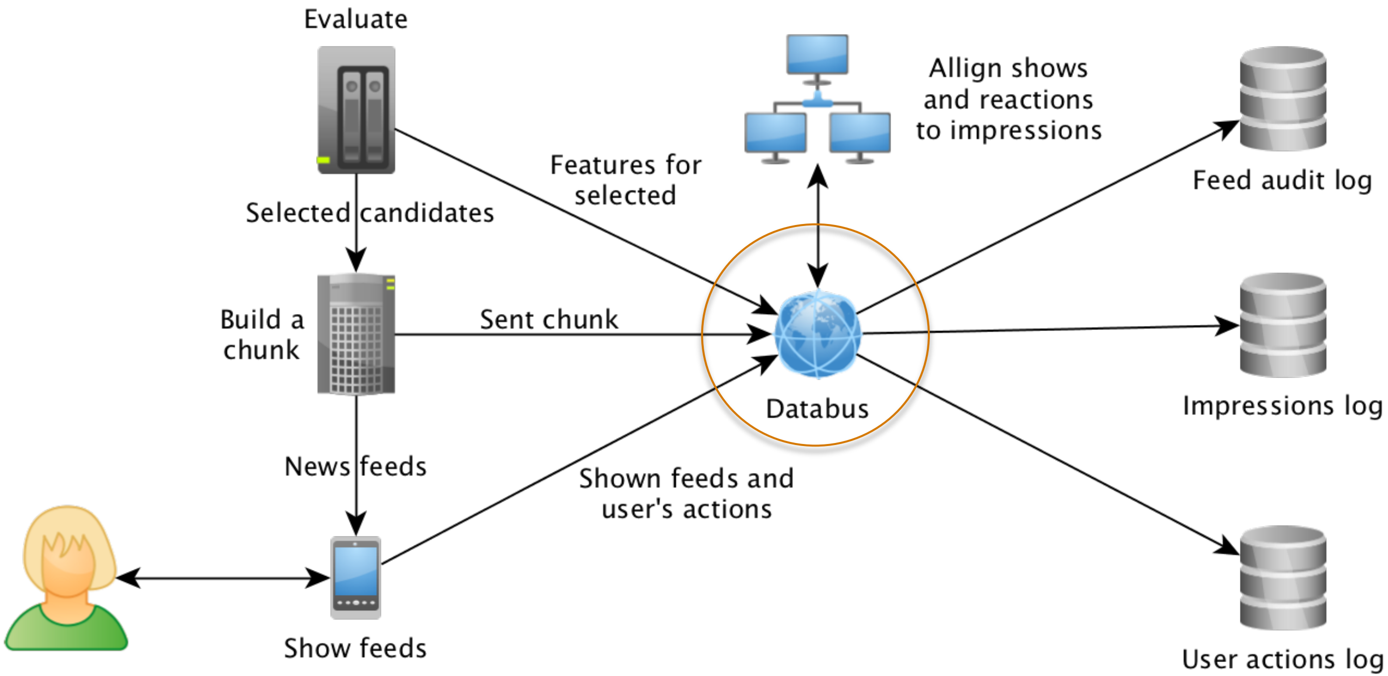
Show feeds and collect data



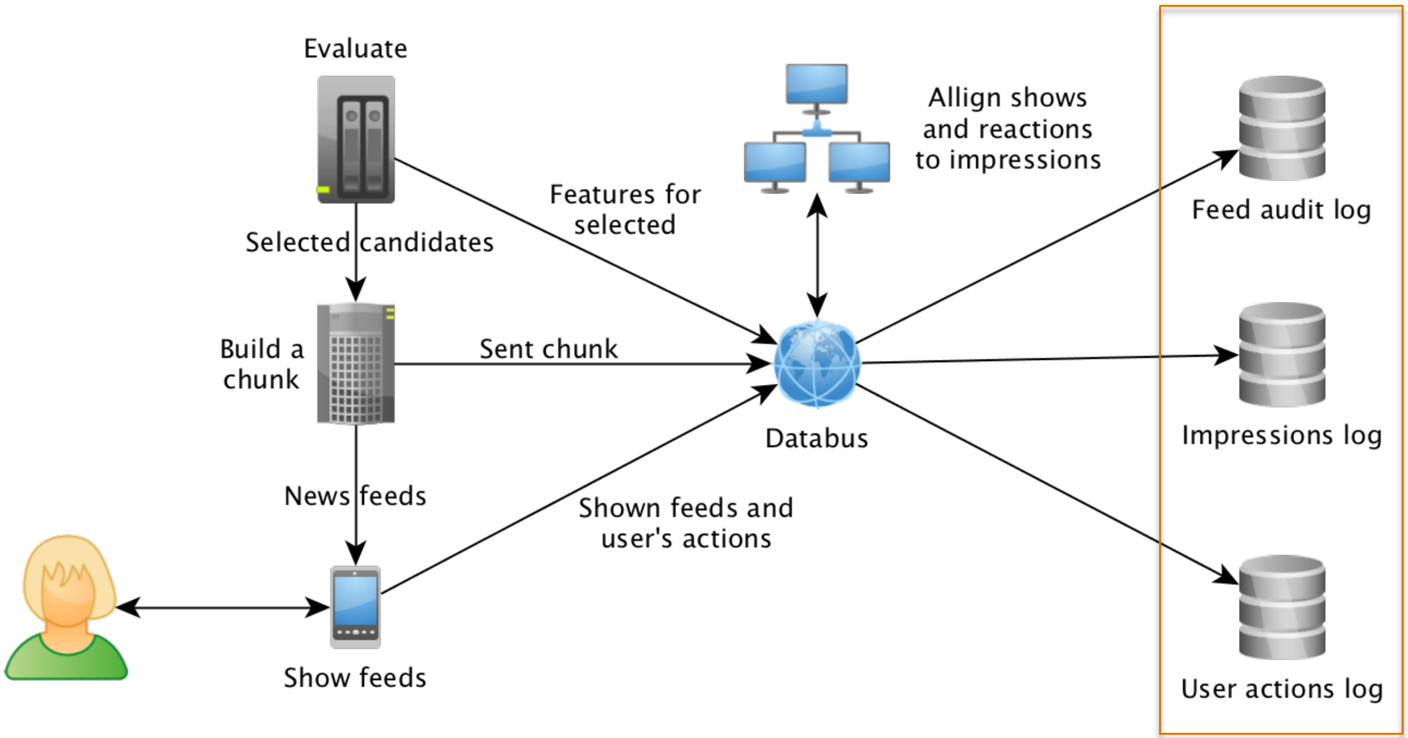
Show feeds and collect data



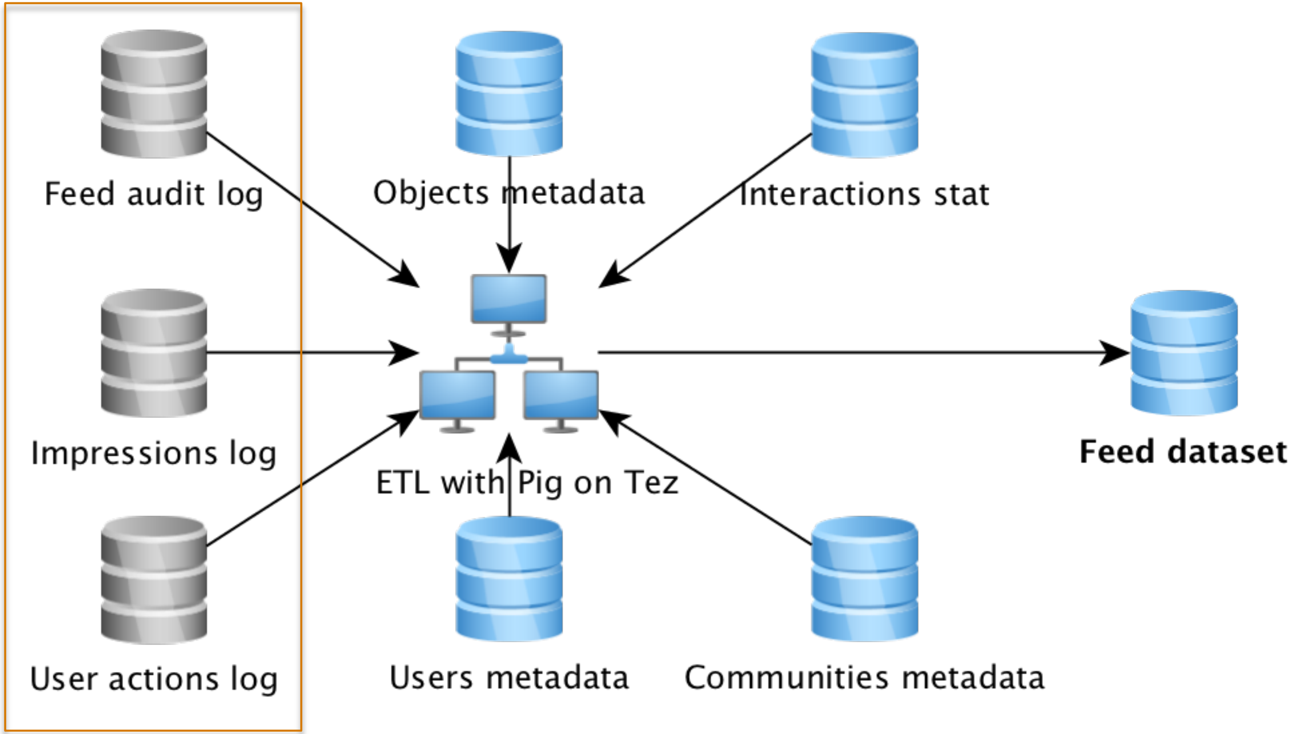
Show feeds and collect data



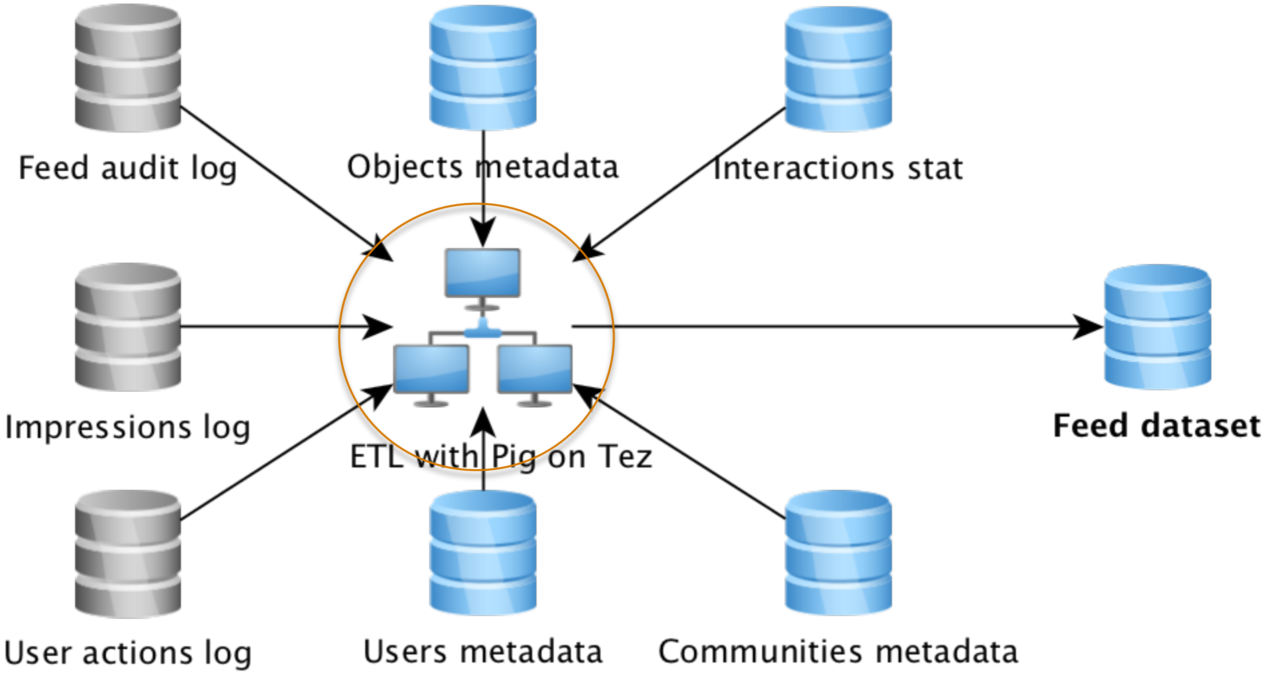
Show feeds and collect data



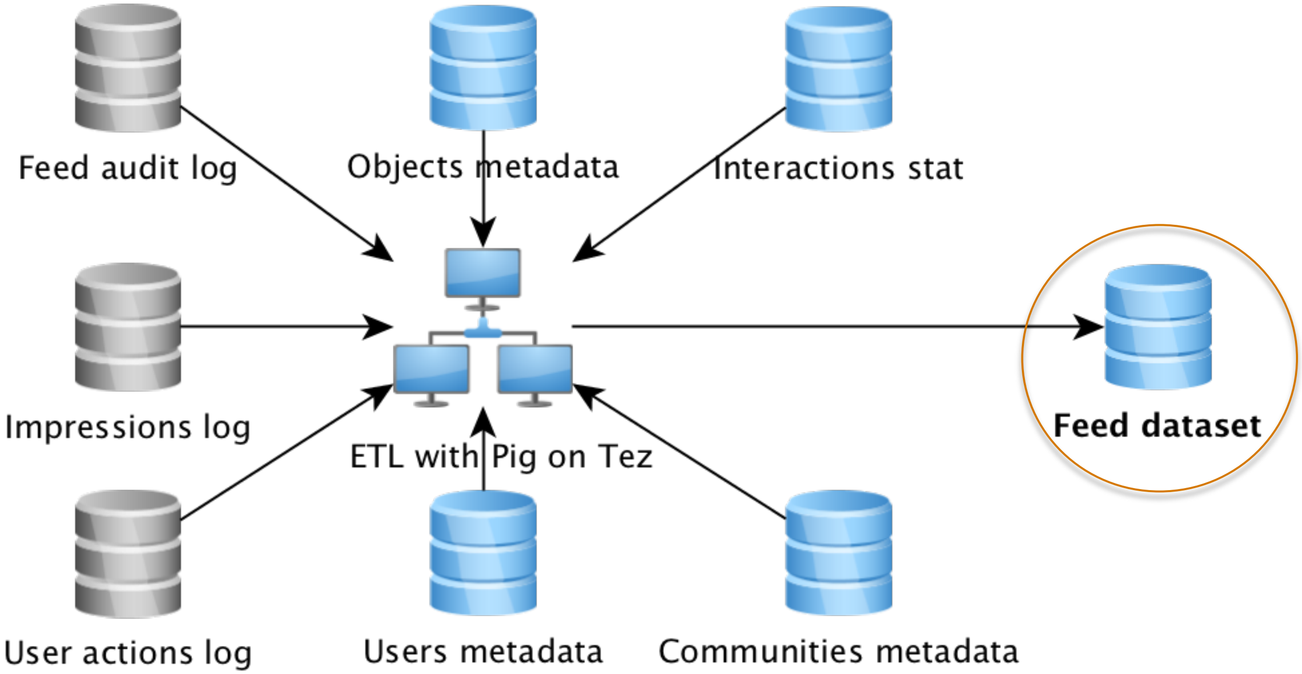
Build a dataset



Build a dataset



Build a dataset

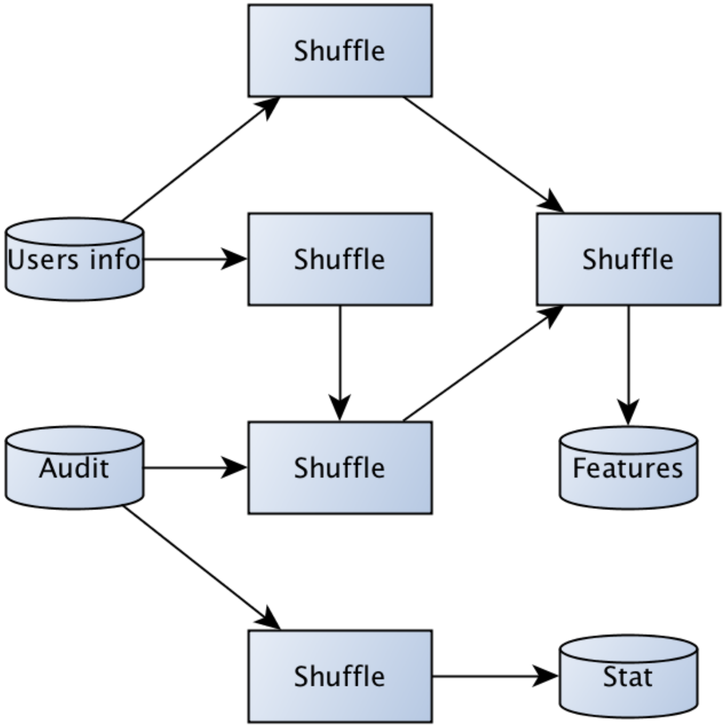


Why Pig, not Spark?

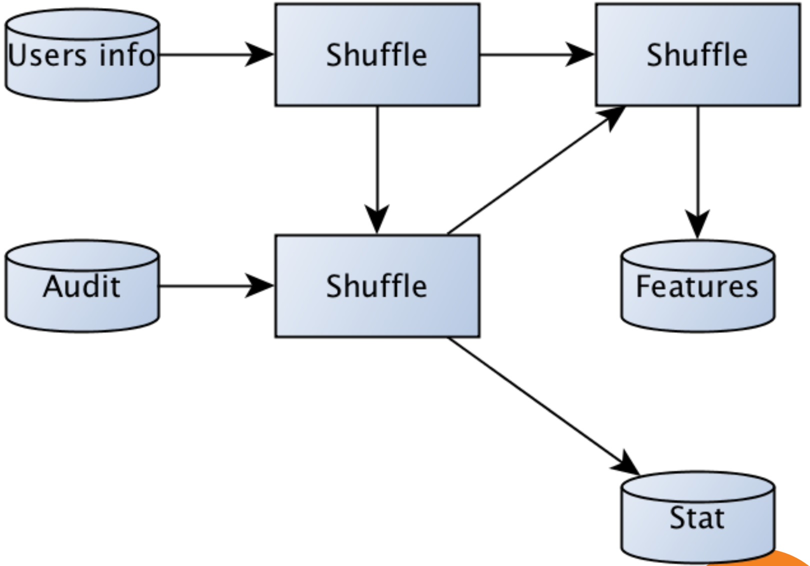
Why Pig, not Spark?

- Better cluster utilization
  - Faster downscaling
  - Larger upscaling
- Better DAG optimization
  - Multi output DAGs
  - Diamond splitters
  - Shuffle reuse
- Better shuffle handling
  - Parallelism estimation
  - Parallelism hints
  - Controllable memory usage

Spark



Pig on Tez



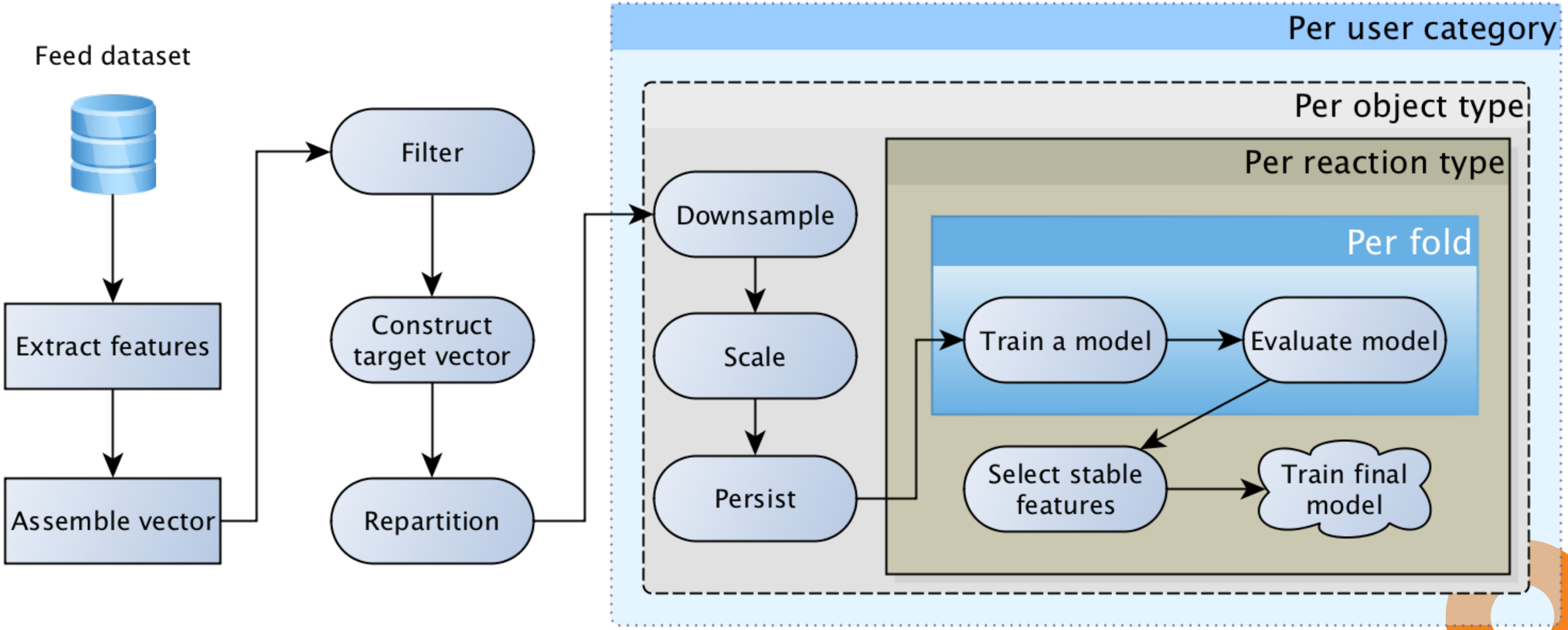
Train models

- Users as split into categories
- Objects are divided by type
- Were are multiple possible reactions for an user to an object

Train models

- Users as split into categories
- Objects are divided by type
- Were are multiple possible reactions for an user to an object
- We need to predict probabilities ☺

Train model



Feed dataset
Extract features
Assemble vector
Filter
Construct target vector
Repartition
Per user category
Per object type
Per reaction type
Per fold
Downsample
Scale
Persist
Train a model
Evaluate model
Select stable features
Train final model

Why Spark, not Python?

Why Spark, not Python?

- Smother transition from ETL to training
- High parallelism:
  - 9 user categories
  - 16 object types
  - 6 reactions types
  - 5 + 1 folds
  - **5 184** models to train in total

Spark ML Pipelines

- Two types of entities:
  - **Transformers** modify (transform) dataset
  - **Estimators** create (fit) transformers
- **Pipeline** is an estimator built as a chain of estimators and transformers
- **Fitting pipeline** replace each estimator with a transformer it fits
- **Pipeline model** is a chain of transformers created by a pipeline

Spark ML Pipelines limitations

- Train-only stages remain in the final result (sampling, repartitioning, caching, etc.)
- No built-in parallelism
- Some data transformations might be eliminated by updating final transformer (eg. feature scaling)
- Hard to get an overview of the resulting model
- Crazy execution plans for large pipelines

ML Pipeline extensions at OK

- Unwrapped Stage
  - Sampling
  - Caching
  - Projection
  - Persist to temp
  - Ordered cut
  - Repartition
  - …
- Forked Estimator
  - Type selector
  - Multi-class
  - Folded
- Model transformers
  - Un-scaler
  - Un-interceptor
- Model With Summary
- Evaluators

OK ML pipelines in action

```
val clusteredEstimator = new Pipeline().setStages(Array(
    new KMeans().setK(numClusters).setPredictionCol("cluster")
    new ColumnsExtractor()
        .withColumns("features", "label", "labelVector")
        .withExpresions("cluster" -> "CONCAT(IF(gender = 1, 'M_', 'F_'), CAST(cluster AS string))"),
    CombinedModel.perType(
        typeColumn = "cluster", parallel = true,
        estimator = Scaler.scale(
            scaler = new ScalerEstimator().setWithMean(true),
            estimator = Interceptor.intercept(
                UnwrappedStage.cacheAndMaterialize(
                    Evaluator.crossValidate(
                        numFolds = 10, parallel = false,
                        estimator = new LogisticRegressionLBFGS().setRegParam(0.02).setRegularizeLast(false),
                        evaluator = new Evaluator.TrainTestEvaluator(new BinaryClassificationEvaluator()))))
            )
    )))

val model = clusteredEstimator.fit(data)
```

OK ML pipelines in action

```
val clusteredEstimator = new Pipeline().setStages(Array(
      new KMeans().setK(numClusters).setPredictionCol("cluster")
      new ColumnsExtractor()
          .withColumns("features", "label", "labelVector")
          .withExpresions("cluster" -> "CONCAT(IF(gender = 1, 'M_', 'F_'), CAST(cluster AS string))"),
      CombinedModel.perType(
          typeColumn = "cluster", parallel = true,
          estimator = Scaler.scale(
              scaler = new ScalerEstimator().setWithMean(true),
              estimator = Interceptor.intercept(
                  UnwrappedStage.cacheAndMaterialize(
                      Evaluator.crossValidate(
                          numFolds = 10, parallel = false,
                          estimator = new LogisticRegressionLBFGS().setRegParam(0.02).setRegularizeLast(false),
                          evaluator = new Evaluator.TrainTestEvaluator(new BinaryClassificationEvaluator()))))
              )
      )))

val model = clusteredEstimator.fit(data)
```

OK ML pipelines in action

```
val clusteredEstimator = new Pipeline().setStages(Array(
        new KMeans().setK(numClusters).setPredictionCol("cluster")
        new ColumnsExtractor()
            .withColumns("features", "label", "labelVector")
            .withExpresions("cluster" -> "CONCAT(IF(gender = 1, 'M_', 'F_'), CAST(cluster AS string))"),
        CombinedModel.perType(
            typeColumn = "cluster", parallel = true,
            estimator = Scaler.scale(
                scaler = new ScalerEstimator().setWithMean(true),
                estimator = Interceptor.intercept(
                    UnwrappedStage.cacheAndMaterialize(
                        Evaluator.crossValidate(
                            numFolds = 10, parallel = false,
                            estimator = new LogisticRegressionLBFGS().setRegParam(0.02).setRegularizeLast(false),
                            evaluator = new Evaluator.TrainTestEvaluator(new BinaryClassificationEvaluator()))))
                )
        )))

val model = clusteredEstimator.fit(data)
```

OK ML pipelines in action

```
val clusteredEstimator = new Pipeline().setStages(Array(
        new KMeans().setK(numClusters).setPredictionCol("cluster")
        new ColumnsExtractor()
            .withColumns("features", "label", "labelVector")
            .withExpresions("cluster" -> "CONCAT(IF(gender = 1, 'M_', 'F_'), CAST(cluster AS string))"),
        CombinedModel.perType(
            typeColumn = "cluster", parallel = true,
            estimator = Scaler.scale(
                scaler = new ScalerEstimator().setWithMean(true),
                estimator = Interceptor.intercept(
                    UnwrappedStage.cacheAndMaterialize(
                        Evaluator.crossValidate(
                            numFolds = 10, parallel = false,
                            estimator = new LogisticRegressionLBFGS().setRegParam(0.02).setRegularizeLast(false),
                            evaluator = new Evaluator.TrainTestEvaluator(new BinaryClassificationEvaluator()))))
                )
        )))

val model = clusteredEstimator.fit(data)
```

OK ML pipelines in action

```
val clusteredEstimator = new Pipeline().setStages(Array(
      new KMeans().setK(numClusters).setPredictionCol("cluster")
      new ColumnsExtractor()
          .withColumns("features", "label", "labelVector")
          .withExpresions("cluster" -> "CONCAT(IF(gender = 1, 'M_', 'F_'), CAST(cluster AS string))"),
      CombinedModel.perType(
          typeColumn = "cluster", parallel = true,
          estimator = Scaler.scale(
              scaler = new ScalerEstimator().setWithMean(true),
              estimator = Interceptor.intercept(
                  UnwrappedStage.cacheAndMaterialize(
                      Evaluator.crossValidate(
                          numFolds = 10, parallel = false,
                          estimator = new LogisticRegressionLBFGS().setRegParam(0.02).setRegularizeLast(false),
                          evaluator = new Evaluator.TrainTestEvaluator(new BinaryClassificationEvaluator()))))
              )
      )))

val model = clusteredEstimator.fit(data)
```

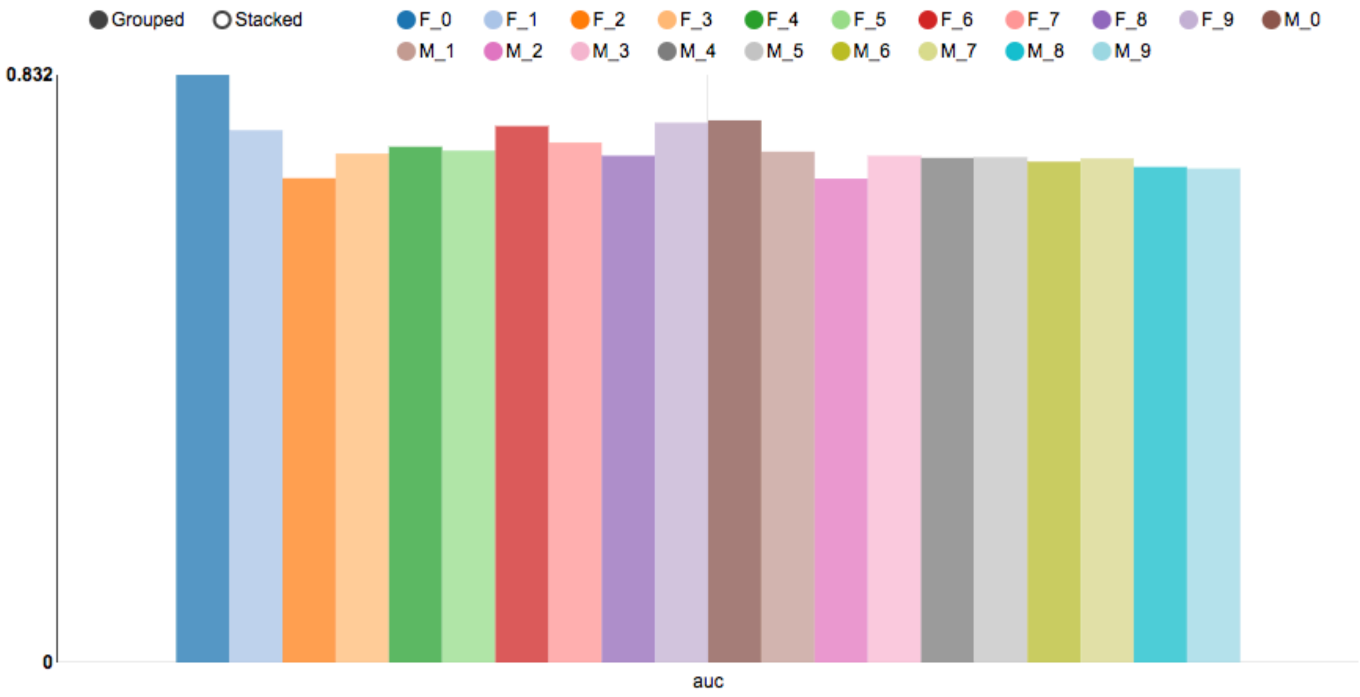
OK ML pipelines in action

```
val clusteredEstimator = new Pipeline().setStages(Array(
    new KMeans().setK(numClusters).setPredictionCol("cluster")
    new ColumnsExtractor()
        .withColumns("features", "label", "labelVector")
        .withExpresions("cluster" -> "CONCAT(IF(gender = 1, 'M_', 'F_'), CAST(cluster AS string))"),
    CombinedModel.perType(
        typeColumn = "cluster", parallel = true,
        estimator = Scaler.scale(
            scaler = new ScalerEstimator().setWithMean(true),
            estimator = Interceptor.intercept(
                UnwrappedStage.cacheAndMaterialize(
                    Evaluator.crossValidate(
                        numFolds = 10, parallel = false,
                        estimator = new LogisticRegressionLBFGS().setRegParam(0.02).setRegularizeLast(false),
                        evaluator = new Evaluator.TrainTestEvaluator(new BinaryClassificationEvaluator()))))
            )
    )))

val model = clusteredEstimator.fit(data)
```

OK ML pipelines in action

```
val clusteredEstimator = new Pipeline().setStages(Array(
        new KMeans().setK(numClusters).setPredictionCol("cluster")
        new ColumnsExtractor()
            .withColumns("features", "label", "labelVector")
            .withExpresions("cluster" -> "CONCAT(IF(gender = 1, 'M_', 'F_'), CAST(cluster AS string))"),
        CombinedModel.perType(
            typeColumn = "cluster", parallel = true,
            estimator = Scaler.scale(
                scaler = new ScalerEstimator().setWithMean(true),
                estimator = Interceptor.intercept(
                    UnwrappedStage.cacheAndMaterialize(
                        Evaluator.crossValidate(
                            numFolds = 10, parallel = false,
                            estimator = new LogisticRegressionLBFGS().setRegParam(0.02).setRegularizeLast(false),
                            evaluator = new Evaluator.TrainTestEvaluator(new BinaryClassificationEvaluator()))))
                )
        )))

val model = clusteredEstimator.fit(data)
```

OK ML pipelines in action



Evaluate model

```
// Compute hell a lot of metrics
new PartitionedRankingEvaluator()
  .setMetrics(
    Seq(
      numNegatives(),
      numPositives(),
      ndcgStrong(),
      ndcgStrongAt(10),
      auc(),
      precision(),
      precisionAt(10),
      recall(),
      recallAt(10),
      f1(),
      f1At(10),
      foundPositives(),
      foundNegatves())
      ++ typeSelector.nested.keys.map(t => countIf(t, r => r.getString(2).equals(t)))
      ++ typeSelector.nested.keys.map(
        t => countRelevantIf(s"${t}_relevant", r => r.getString(2).equals(t)))
      ++ typeSelector.nested.keys.map(
        t => countDistinctIf(s"${t}_distinctOwner", r => r.getString(2).equals(t), r => r.getLong(3)))
      ++ typeSelector.nested.keys.map(
        t => countDistinctRelevantIf(
          s"${t}_distinctRelevantOwner", r => r.getString(2).equals(t), r => r.getLong(3))
    ): _*
  )
  .setModelThreshold(0.0)
  .setGroupByColumns("userId", "ownerType", "objectType")
  .setExtraColumns("type", "ownerId"),
// Include only users with both positive and negative instances for certain type
new SqlFilter().setWhere("BOTH_POSITIVE(metrics)"),
new VectorStatCollector()
  .setInputCol("metrics")
  .setGroupByColumns("label", "score", "ownerType", "objectType")
  .setNumPartitions(settings.aggregateMetricsPartitions)
  .setNumShufflePartitions(settings.preAggregateShufflePartitions),
new NameAssigner().setInputCols("label", "score"),
new VectorExplode()
```

- Time based per-user validation
- Evaluate model for it own task
- Evaluate combinations for global performance
- Most informative metrics: AUC, NDCG, distinct relevant owners

Why offline evaluation sucks?

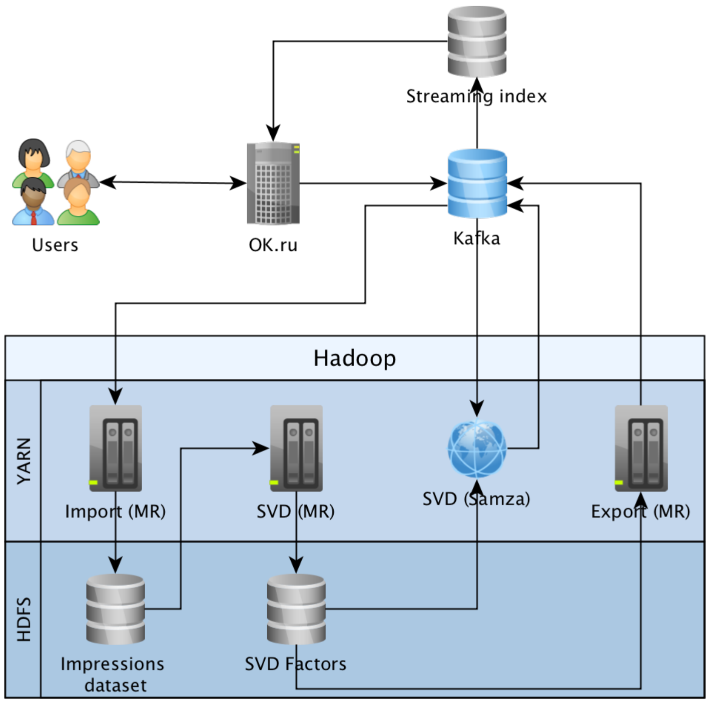
Why offline evaluation sucks?

- User behavior depends on the whole feed, not only on a single feed record
- Many important KPI’s are system wide and can not be deduced from models’ scores directly
- Evaluation results are biased by the model’s which were active during training and evaluation time
- **To conclude**: offline evaluation can show if the model is meaningful or not, but not more

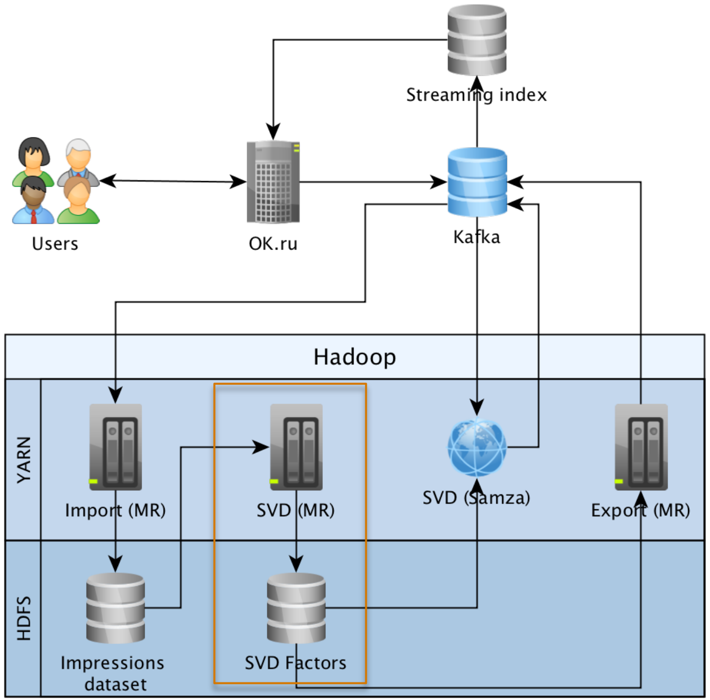
Prepare features

- Deduce from the feed input
  - Number of friends’ likes
  - First/last event date, etc.
  - …
- Read from existing service
  - User and owners’ demography
  - Communities metadata
  - Relation masks
  - PYMK relevance
  - …
- Compute offline
  - SVD/LDA profiles
  - …
- Compute in real time
  - CTR’s
  - Document LDA
  - Document SVD bias
  - …
- Compute online
  - SVD/LDA prediction
  - …

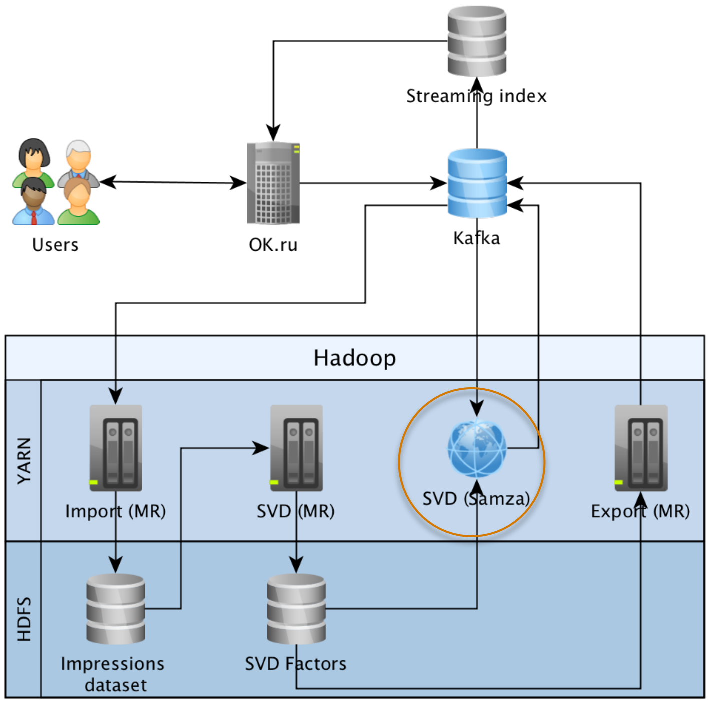
Streaming ML



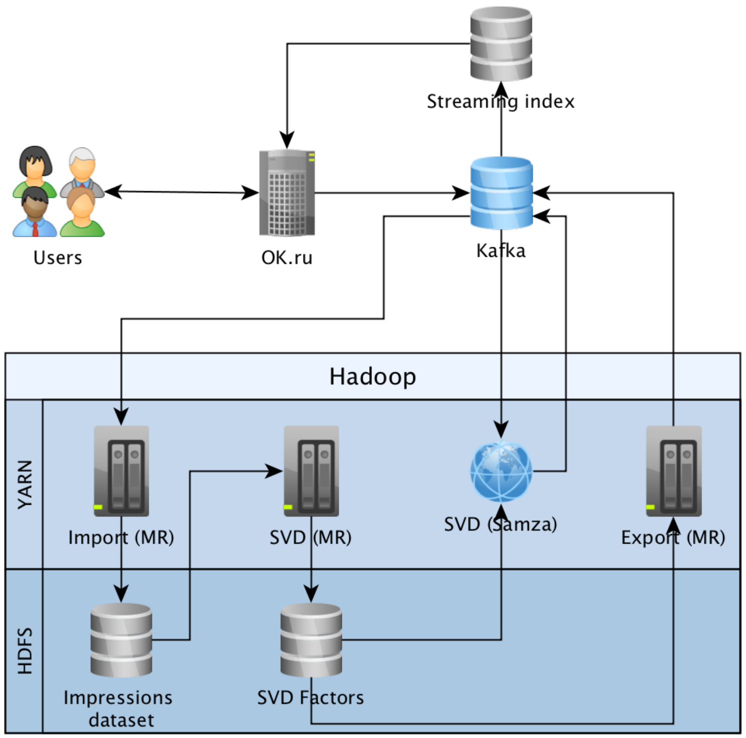
Streaming ML



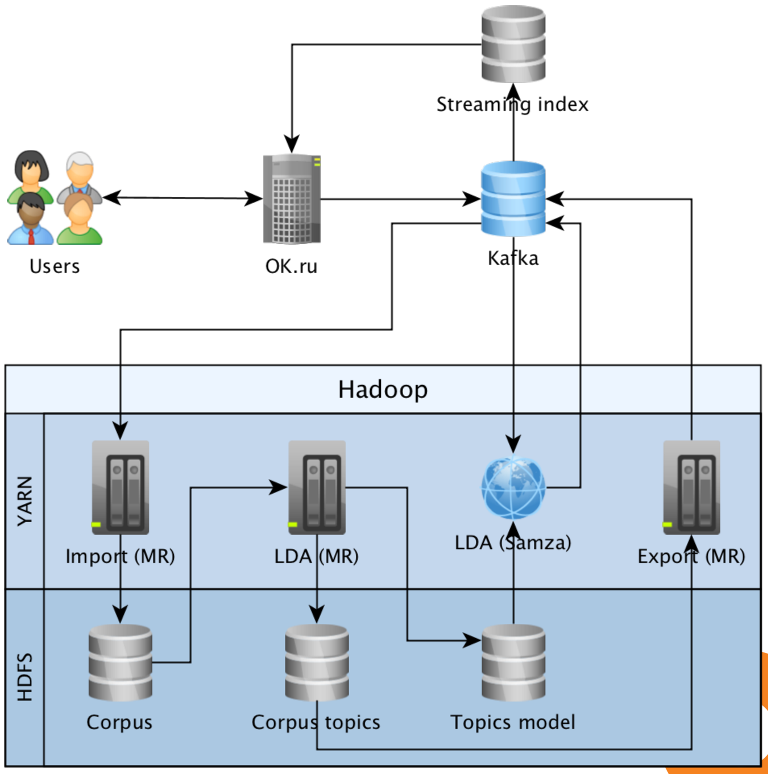
Streaming ML



Streaming SVD



Streaming LDA



Why Samza, not Spark?

Why Samza, not Spark?

- Easy to test with unit test
- Simple maintenance procedures (failure recovery, update, monitoring)
- Transparency and performance
- Time to market

Apply models

1. Get user's subscriptions
2. Read recent events
3. Extract objects and actors
4. Fetch all the features
5. Evaluate predictions
6. Apply business rules
7. Store the result

WTF are business rules for?

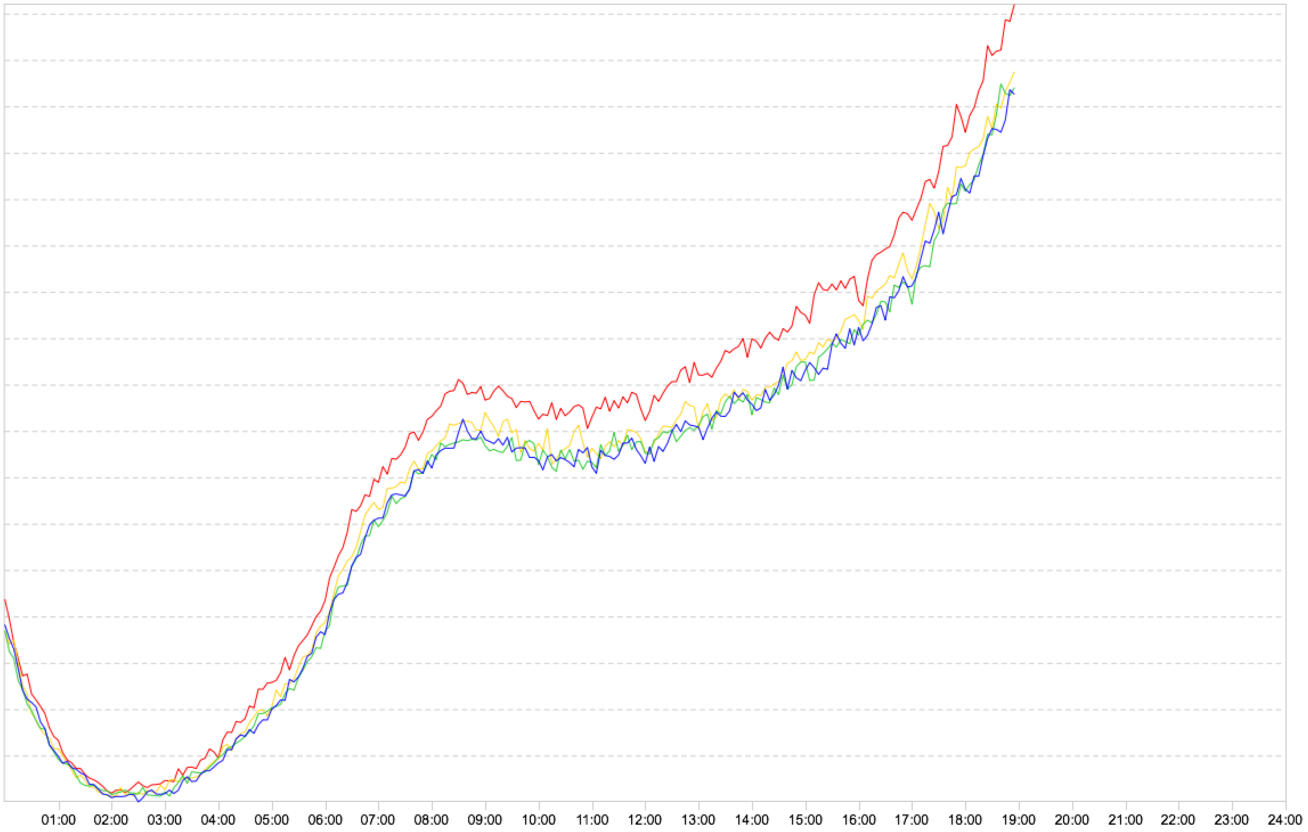
- Per-object rules
  - Counteract spammers' tricks
  - Consider user value
- List-wise rules
  - Improve diversity
  - Inject non-personalized content
- System-wise rules
  - Distribute feedback evenly

Experiment and Analyze

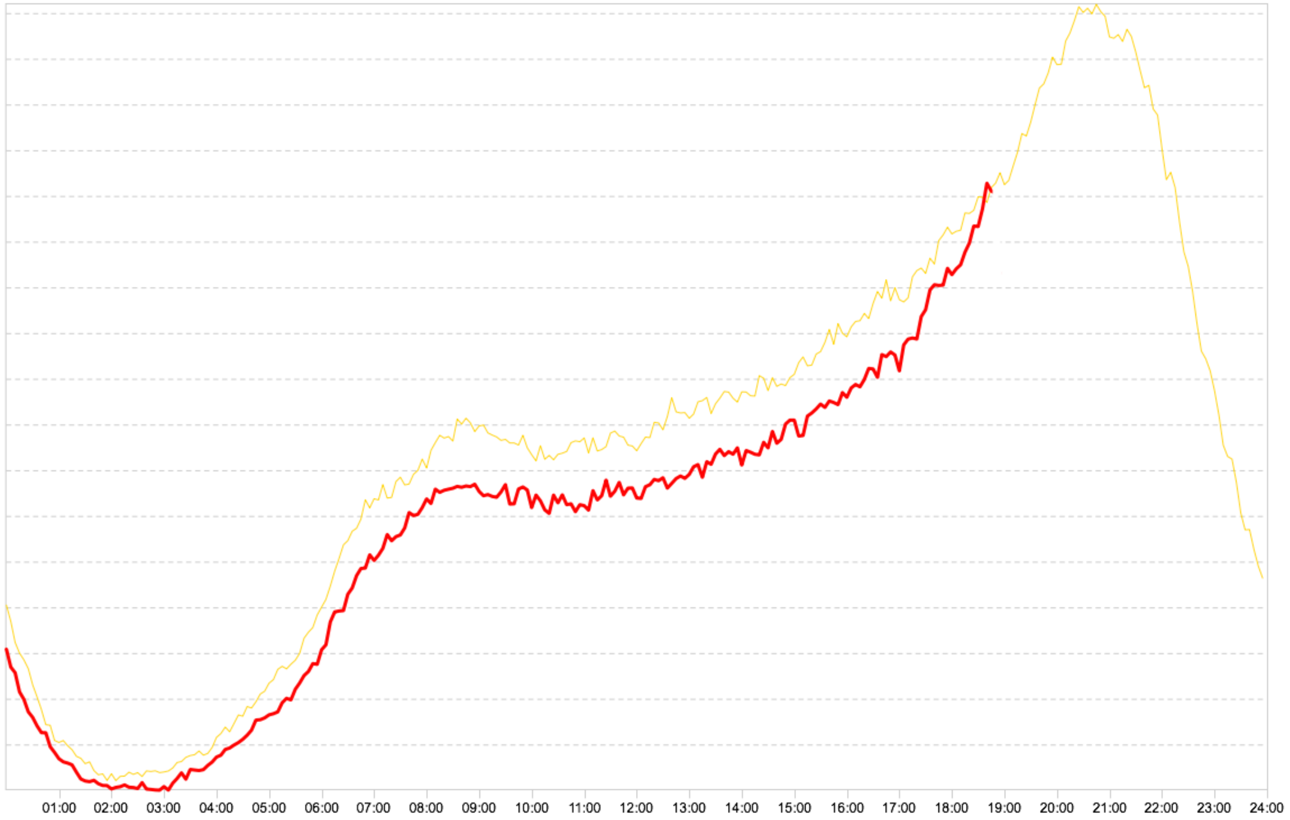
Experiment and Analyze

Experimenting with news feed and analyzing result is a pain in the sensitive place!

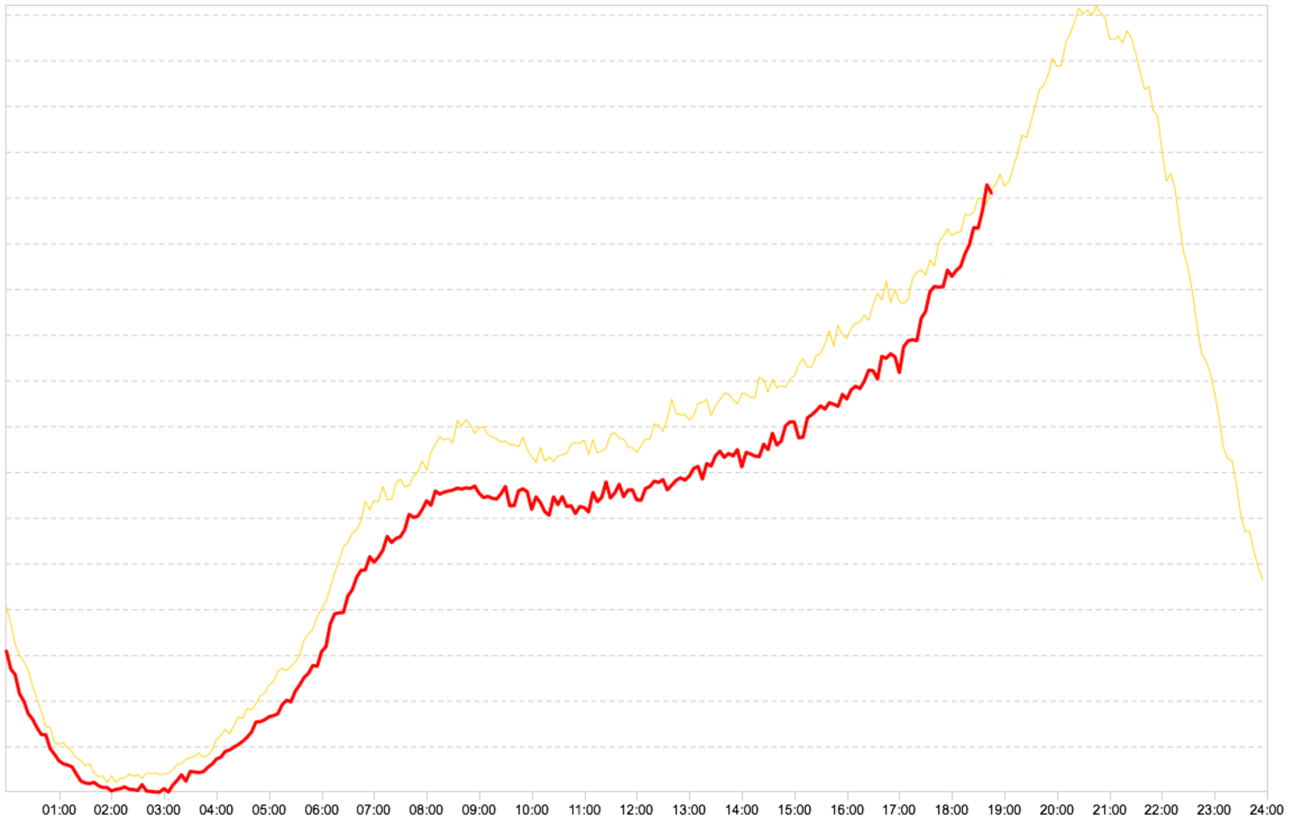
Analyze A/B experiment, started at 17:30



Response to the experiment from Split 12 in Y/T view



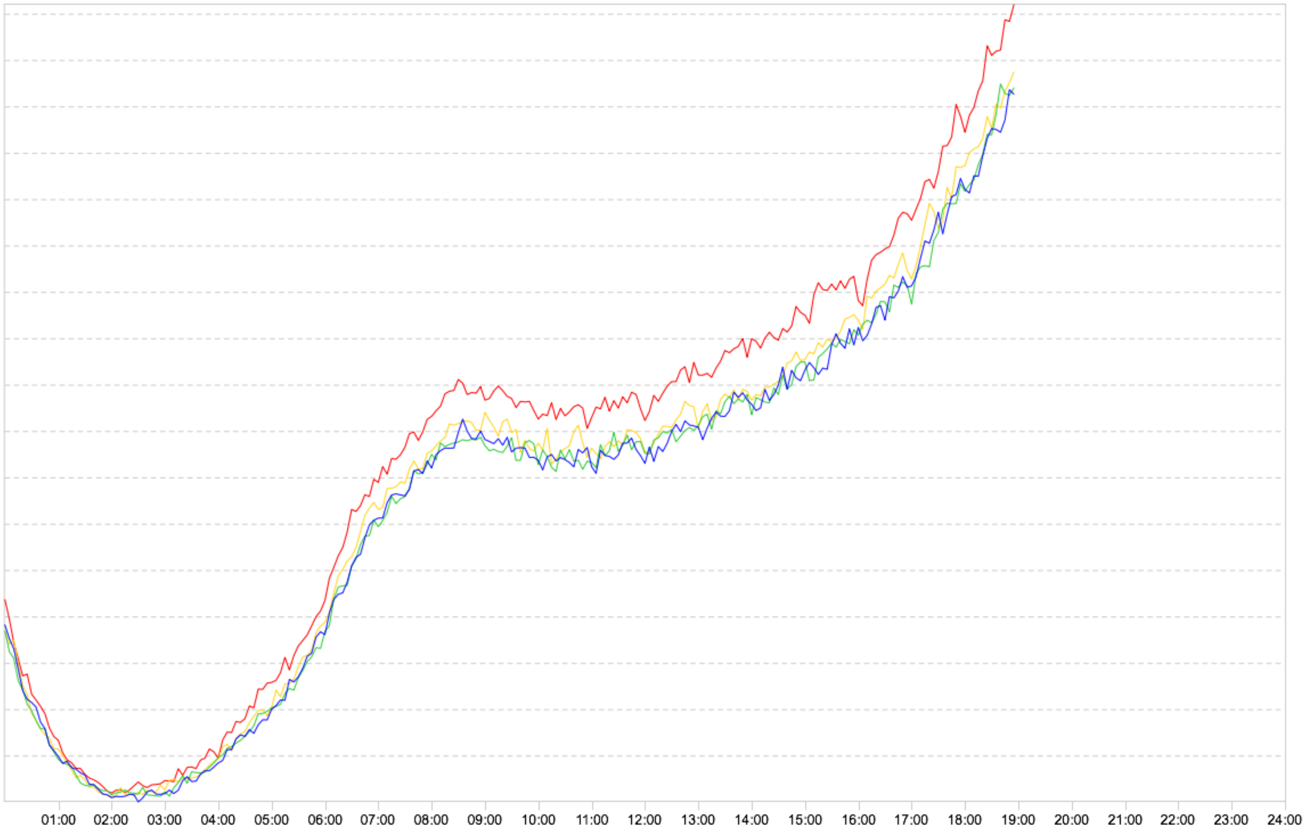
Response to the experiment from Split 12 in Y/T view



BUT!!!!

The experiment is on **Split 6**

Are you still a fan of A/B tests?

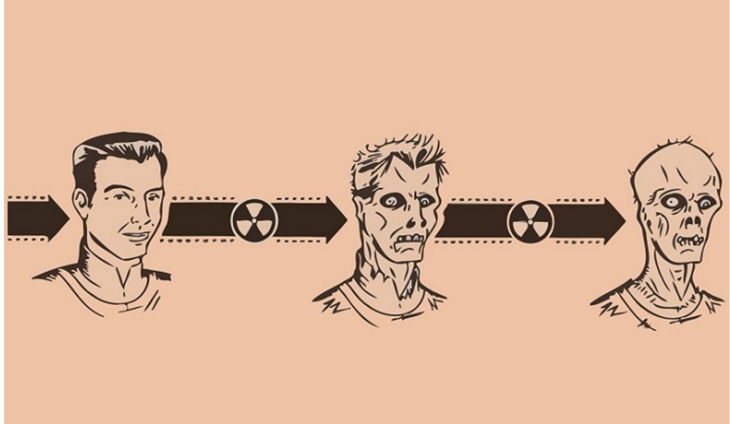


Why ~~shit~~ it happens?

$$\eta(t) = \eta_0 \exp\left(\frac{k-1}{\tau} t\right)$$

Why ~~shit~~ it happens?

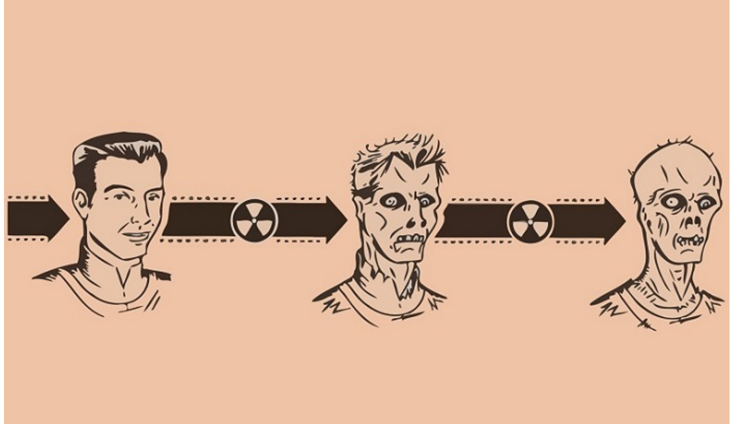
$$\eta(t) = \eta_0 \exp\left(\frac{k-1}{\tau} t\right)$$



$k < 1$

Why ~~shit~~ it happens?

$$\eta(t) = \eta_0 \exp\left(\frac{k-1}{\tau} t\right)$$

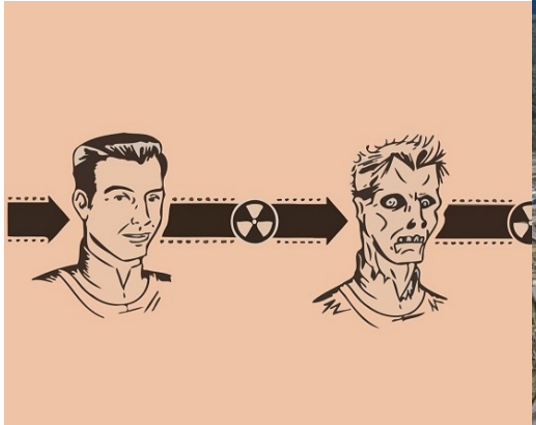


k < 1

k > 1

Why ~~shit~~ it happens?

$$\eta(t) = \eta_0 \exp\left(\frac{k-1}{\tau} t\right)$$



k < 1

k = 1

k > 1

Few recommendations for the analysis

- Keep an eye on the long-running most important KPIs
- Look for correlations between long-running and more instant KPIs
- Not limit analyses to A/B – monitor global trends change
- Measure thrice and cut once

Few recommendations for the analysis

- Keep an eye on the long-running most important KPIs
- Look for correlations between long-running and more instant KPIs
- Not limit analyses to A/B – monitor global trends change
- Measure thrice and cut once
- **Keep calm and drill deeper**

We are hiring!

- Like industrial technologies and scale?
- ML and high load challenges?
- Mail us:

cv@odnoklassniki.ru