Data Science Team
Quality Analytics
R&D Quality Community
Rrober14@its.jnj.com

MLOPS Plan Template – Data Theory

(Syslog)

6.4.1.3.4 Verify dynamic persistence
6.4.1.3.5 Data quality checks
6.4.1.3.5.1 Verify duplication rule adherence
6.4.1.3.5.2 Check that the columns we expected are there
6.4.1.3.5.3 Check the datatypes are as expected
6.4.1.3.5.4 Check that dataset meets minimum observation requirements
6.4.1.3.5.5 Check renaming features to a universal format is performed as expected
6.4.1.3.5.6 Check that minimum feature requirements are met
6.4.1.3.5.7 Check data conformance

Check if the data looks the same whether it's received through the app or the console. In some cases, the way the form is filled out in the app might add or remove line breaks in the text.

6.4.1.3.5.8 Check validity of user-list

6.4.1.4 4-Aggregation
6.4.1.4.1 Correctness-feature
6.4.1.4.1.1 Acronyms/stop words, aggregations, etc
6.4.1.4.1.2 Data Type Inference
6.4.1.4.1.3 Missing Values Inference
6.4.1.4.1.4 Unique Values Inference
6.4.1.4.1.5 Outliers Inference

6.4.1.5 5-Analysis
6.4.1.5.1 Syslog quality
6.4.1.5.2 Operational statistical diagnostics
6.4.1.5.3 Operational models
6.4.1.5.3.1 Ensure that the models are running behind the hook
6.4.1.5.4 Training/target metadata consistency
6.4.1.5.4.1 Check if the advanced configurations are reflected in the results generated
6.4.1.5.4.2 Order of columns passed for prediction is matched order of columns in training
6.4.1.5.4.3 Check language mismatch (trained on English but query is French)
6.4.1.5.4.4 Safeguard for embedding vector size too large (raises allocated more memory than is available error)
6.4.1.5.4.5 NLP models raised errors and root causes
6.4.1.5.4.6 ML models raised errors and root causes
6.4.1.5.4.7 High Performance Computing Thresholds

6.4.1.6 6-Application
6.4.1.6.1 Multiple users functionality
6.4.1.6.2 Functional UI elements
6.4.1.6.2.1 UI functionality - upload the dataset
6.4.1.6.2.2 UI functionality - select the source/columns
6.4.1.6.2.3 UI functionality - select hidden/optional parameters

6.4.1.6.2.4 UI functionality - fail gracefully if provided input files or parameter files are badly formatted
6.4.1.6.2.5 Check if the user can perform advanced configurations
6.4.1.6.2.6 Configurations should reset per query
6.4.1.6.3 UI functionality – dead button test
6.4.1.6.4 Check if the results generated are clickable and downloadable
6.4.1.6.5 Check persona warnings/alerts are set
6.4.1.6.6 Check that an operation is completed within the expected time interval
6.4.1.6.7 Platform functionality
6.4.1.6.7.1 User access point
6.4.1.6.7.2 Hosting provider availability
6.4.1.6.7.3 Redirecting to the correct location
6.4.1.6.8 Assets sanity check
6.4.1.6.8.1 Check if the graphs are correctly loaded
6.4.1.6.8.2 Verifying sanity of engineered features

6.4.2 Regression Testing:

Comparison between golden standard output of the application Vs current output

Occurs after any pipeline change to ensure no unintended bugs introduced

Testing if application deployment takes the same amount of time after any application related change

6.4.3 Robustness Testing

Robustness means system can withstand a change, disturbance or anomalies

Robustness Test = Unit Test + Known User Anti-Pattern Behavior

6.4.3.1 1-Governance & Provenance
6.4.3.2 2-Generation
6.4.3.2.1 Credentials verification
6.4.3.2.2 Input format/size compatibility
6.4.3.2.2.1 Incompatible input type
6.4.3.2.2.2 Large input/small input

6.4.3.3 3-Collection & Standardization
6.4.3.3.1 Data quality/quantity checks:
6.4.3.3.1.1 Duplicate rows/columns of data
6.4.3.3.1.2 Missing columns
6.4.3.3.1.3 Wrong data types

6.4.3.3.1.4 Many urls

6.4.3.4 4-Aggregation

6.4.3.4.1 Verifying sanity of engineered features

6.4.3.4.1.1 Entire input is stopwords or acronyms

6.4.3.5 5-Analysis

6.4.3.5.1 Training/target metadata consistency

6.4.3.5.1.1 Wrong columns orders

6.4.3.5.1.2 Check language mismatch (trained on English but query is French)

6.4.3.5.1.3 Safeguard for embedding vector size too large (raises allocated more memory than is available error)

6.4.3.5.2 Concept drift

6.4.3.5.2.1 Input/training data (drift shouldn't be a lot)

6.4.3.5.2.2 Output of the models

6.4.3.5.2.3 Performance metrics conformance

What changed?			
More time has passed	Level 1	Fresh data	Retrain existing model
The populations within data client gives us has changed	Level 2	Data distributions change	Parameter tuning
The features client gives has changed +/-	Level 3	Data structure change	AUTO-ML (bag o tricks)
Change strategy, change the target variable	Level 4	Start from scratch	Start from scratch

6.4.3.6 6-Application

6.4.3.6.1 Functional UI elements

6.4.3.6.1.1 UI functionality - fail gracefully if provided input files or parameter files are badly formatted (or swapped with a differently structured file)

6.4.3.6.1.2 Very complex filters/ acronyms/stop words, choosing techniques from dropdown for searching, etc.

6.4.3.6.1.3 Configurations should reset per query

6.4.4 Antifragility Testing

Improve the system's performance and adaptability when exposed to randomness, and unforeseen events

Antifragility Test = Unit Test + User Expectations

Redundancy - Having backup components or systems to take over in case of failure

Fault Tolerance/Resilience - continue operating properly if some components fails
Scalability - The ability of the ML pipeline to handle increasing loads or expand its capacity

6.5 Notifications

6.5.1 UI Interactive Notifications:

6.5.1.1 Credentials verification -> notification to user?/
6.5.1.2 Check if the dataset uploaded is within the threshold of the application (alert to user on UI or email)
6.5.1.3 Check if no empty dataset is uploaded (alert to user on UI or email)
6.5.1.4 stop words/acronyms -> notify user on UI
6.5.1.5 Check language mismatch (trained on English but query is French) -> notification to user on UI
6.5.1.6 UI functionality - upload the dataset + notification to the user
6.5.1.7 Check if the user can perform advanced configurations (adding filters/ acronyms/stop words, choosing techniques from dropdown for searching, etc.)
6.5.1.8 Check persona warnings/alerts are set

6.5.2 Reminder notification -> notification to user

6.5.2.1 If the user X hasn't logged in to the app in X days, then send out a reminder to use it
6.5.2.2 In case of an incomplete session
6.5.2.3 Provide feedback on their experience with app
6.5.2.4 Subscribe to DS newsletter

6.5.3 Update Notification -> notification to user

6.5.3.1 Upcoming Downtime/breakdown of app
6.5.3.2 Models re-training
6.5.3.3 Data refresh
6.5.3.4 App update

6.5.4 Engagement Notification -> notification to user

6.5.4.1 If a user completes specific tasks within an app, the app can then sent them a message to motivate them
6.5.4.2 User training request

6.5.5 Unit Tests Notifications:

6.5.5.1 1-Governance & Provenance
6.5.5.1.1 Dependencies verification
6.5.5.1.1.1 versions, availability of packages and libraries, etc -> notification to developer (log)

6.5.5.2 2-Generation
6.5.5.2.1 Credentials verification -> notification to developer (log/email)
6.5.5.2.2 Environment variables verification -> notification to developer (log)
6.5.5.2.3 Verify read/write accessibility -> notification to developer (log)
6.5.5.2.4 Input format/size compatibility -> notification to developer (log)
6.5.5.2.4.1 Verify input files type compatibility
6.5.5.2.4.2 Check if the dataset uploaded is within the threshold of the application
6.5.5.2.4.3 Check if no empty dataset is uploaded

6.5.5.3 3-Collection & Standardization
- **6.5.5.3.1** Verify ACID adherence (1-Governance & Provenance?)
- **6.5.5.3.2** Verify set frequency (Syslog) persistence -> have a test in pipline to check date of latest log -> notification to developer (email)
- **6.5.5.3.3** Verify dynamic persistence
- **6.5.5.3.4** Data quality/quantity checks: -> notification to developer (log)
  - 6.5.5.3.4.1 Verify duplication rule adherence
  - 6.5.5.3.4.2 Check that the columns we expected are there
  - 6.5.5.3.4.3 Check the datatypes are as expected
  - 6.5.5.3.4.4 Check that dataset meets minimum observation requirements
  - 6.5.5.3.4.5 Check renaming features to a universal format is performed as expected
  - 6.5.5.3.4.6 Check that minimum feature requirements are met

- **6.5.5.3.5** Check validity of user-list (valid emails?) ->manual labor

6.5.5.4 **4-Aggregation** -> notification to developer (log)
- **6.5.5.4.1** Verifying sanity of engineered features
- **6.5.5.4.2** acronyms/stop words, aggregations, etc

6.5.5.5 5-Analysis
- **6.5.5.5.1** Syslog quality
- **6.5.5.5.2** Operational models:
  - 6.5.5.5.2.1 Ensure that the models are running behind the hook -> notification to developer (log)
- **6.5.5.5.3** Training/target metadata consistency -> notification to developer (log)
  - 6.5.5.5.3.1 Check if the advanced configurations are reflected in the results generated
  - 6.5.5.5.3.2 Order of columns passed for prediction is matched order of columns in training
  - 6.5.5.5.3.3 Check language mismatch (trained on English but query is French)
  - 6.5.5.5.3.4 Safeguard for embedding vector size too large (raises allocated more memory than is available error)
  - 6.5.5.5.3.5 High Performance Computing Thresholds

6.5.5.6 6-Application
- **6.5.5.6.1** Functional UI elements -> notification to the developer (log)
  - 6.5.5.6.1.1 UI functionality - upload the dataset
  - 6.5.5.6.1.2 UI functionality - select the source/columns
  - 6.5.5.6.1.3 UI functionality – select hidden/optional parameters
  - 6.5.5.6.1.4 UI functionality – fail gracefully if provided input files or parameter files are badly formatted (or swapped with a differently structured file)
  - 6.5.5.6.1.5 Check if the user can perform advanced configurations (adding filters/ acronyms/stop words, choosing techniques from dropdown for searching, etc.)
  - 6.5.5.6.1.6 Configurations should reset per query
    - 6.5.5.6.1.6.1 UI functionality – dead button test

6.5.5.6.1.6.2 Check if the results generated are clickable and downloadable

6.5.5.6.2 Check persona warnings/alerts are set
6.5.5.6.2.1 Platform functionality -> notification to the developer (log)
6.5.5.6.2.1.1 user access point (web browsers testing)
6.5.5.6.2.1.2 hosting provider availability?
6.5.5.6.2.1.3 redirecting to the correct location
6.5.5.6.2.1.4 "check that an operation is completed within the expected time interval"
6.5.5.6.3 Assets sanity check -> notification to the developer (log)
6.5.5.6.3.1 Check if the graphs are correctly loaded

Check if the results are generated when the analysis is completed

6.6 Operational Health
6.6.1 Data Integrity Diagnostics
6.6.1.1 Missingness
6.6.1.2 Correctness
6.6.1.3 Granularity
6.6.1.4 Relevant

6.6.2 Application Diagnostics
6.6.2.1 Customer acquisition
6.6.2.2 Customer retention
6.6.2.3 Uptime

6.7 Performance & Impact
6.7.1.1 User Feedback
6.7.1.2 Unified Behaviors Standard Metrics
6.7.1.3 Data Quality (2.0-business impact)
6.7.1.4 Code and Scripts
6.7.1.4.1 www folder
6.7.1.4.2 application-orchestrators
6.7.1.4.3 scripts