INTERNSHIP PROJECT REPORT

Final Report : Channel Quality Index (CQI) - AI/ML Project for PreTUPS

Submitted By:

Name: Mudit Gupta
Employee ID: 20007710
Department: PreTUPS
Organization: Comviva
Internship Duration: 5 June 2025 – 21 July 2025
Submission Date: 18 July 2025

Acknowledgement

I would like to express my deepest gratitude to **Comviva Technologies Limited** for providing me with the opportunity to undertake this enriching internship experience as part of the PreTUPS Engineering Team. The internship not only allowed me to explore the intersection of AI/ML with large-scale telecom channel operations, but also enabled me to work on a real-world problem that contributes meaningfully to business intelligence and product enhancement.

I am especially grateful to **Mr. Ojas Maheshwary**, Product Manager, for his constant support, visionary guidance, and detailed feedback throughout the development of the **Channel Quality Index (CQI)** project. His clear articulation of the business objectives and expectations helped shape the project into a solution that aligned both with technical soundness and product relevance.

I would also like to sincerely thank **Mr. Sagun Pratap Singh**, Senior Director of Products, for his strategic insights and encouragement during key stages of the project. His inputs helped me align my technical work with the larger business goals of the PreTUPS platform, and his leadership provided the necessary motivation to deliver high-quality outcomes.

Their mentorship and support created a collaborative and intellectually stimulating environment that pushed me to apply my technical skills meaningfully while learning to think from a product and customer perspective.

Lastly, I extend my gratitude to every member of the PreTUPS Engineering Team for their approachable nature and readiness to support, which made my internship experience smooth and intellectually rewarding. This project has been a pivotal step in my professional development, and I am grateful for the learning and exposure I have received.

Project Overview

The **Channel Quality Index (CQI)** – AI/ML Project for **PreTUPS** at **Comviva Technologies Ltd.** was conceptualised to address a fundamental business and operational challenge faced in large-scale telecom distribution networks: the inability to systematically quantify the quality of channels (retailers, agents, and distributors) using a comprehensive, data-driven, and scalable metric. PreTUPS, being one of the most widely deployed electronic recharge and distribution platforms globally, generates massive volumes of transactional and operational data every day across diverse geographies, partners, and retail touch-points. However, without a structured framework to assess the health and risk profile of individual retail channels, business teams are often left to rely on subjective judgement or limited metrics for key decisions such as partner evaluation, risk forecasting, audit prioritisation, and business development planning.

This project aims to fill that critical gap by developing an **AI/ML-powered scoring framework** called the **Channel Quality Index (CQI)**, which assigns a robust and interpretable score to each channel entity. The CQI is designed to be a composite index that reflects multiple aspects of a retailer's operational quality, behavioural consistency, engagement with the platform, and potential fraud or risk markers. The project integrates modern data science techniques including **unsupervised learning (clustering)**, **feature engineering**, **anomaly detection**, and **scoring logic**, combined with a modular, scalable data pipeline and an interactive **Streamlit dashboard** for real-time visualisation and business reporting.

The project architecture follows a clean and professional modular breakdown, starting from **synthetic data generation** (simulating retailer, transaction, feedback, and behavioural logs), followed by **data preprocessing**, **feature matrix creation**, **clustering models**, **risk and quality scoring logic**, and **dynamic reporting utilities**. Every component is integrated into a seamless **automated pipeline**, with scope for scalability, model retraining, parameter tuning, and production deployment.

The importance of such a system is far-reaching. By delivering a quantifiable score to every channel partner in the PreTUPS network, the CQI enables business stakeholders to:

- Identify **high-performing** vs **underperforming** retailers.
- Detect **suspicious or risky activity patterns** using unsupervised anomaly-based clustering.
- Monitor **feedback consistency** and **transactional engagement** over time.
- Provide **data-driven recommendations** to regional and operational teams.
- Standardise **channel health reports** across geographies, languages, and operator setups.

This project also reflects a broader shift towards data-centric, intelligent decision-making within telecom product platforms. Instead of relying on basic reports or manual inspections, the CQI transforms raw operational data into meaningful, explainable, and actionable intelligence. It empowers internal stakeholders with clarity, precision, and foresight—key to driving business strategy in competitive and fragmented markets.

In terms of execution, this project was structured with clear deliverables at each stage. Over the course of the internship, I designed and built:

- End-to-end data generation and processing modules simulating realistic retail datasets.
- Feature engineering pipelines to extract over 20+ quality metrics per retailer.
- Unsupervised clustering models to identify retailer personas and risk profiles.
- Scoring logic to compute a normalised and interpretable CQI score.
- Business-focused dashboards with fraud, cluster, and feedback visualisations.
- Professional reports summarising key insights, rankings, and fraud risk flags.

Overall, the **Channel Quality Index project** is a significant step forward in embedding intelligence into PreTUPS' core data ecosystem. It demonstrates how **AI/ML can power operational intelligence** that is not only accurate and scalable but also deeply aligned with business needs. The project not only meets the goals of my internship but also sets a foundation for future enhancements like supervised fraud prediction, feedback NLP, geo-level score breakdowns, and integration with live production data.

Objective of the Project

The primary objective of the Channel Quality Index (CQI) project was to design, develop, and implement a comprehensive AI/ML-based scoring system that evaluates the overall quality and health of channel partners—particularly retailers—across the PreTUPS platform. This index would serve as a quantitative benchmark for operational excellence, behavioral consistency, transactional engagement, and fraud risk mitigation in telecom distribution networks.

In large-scale digital recharge and mobile money ecosystems like PreTUPS, where thousands of retailers operate across diverse markets and regulatory environments, there is a growing need to monitor and evaluate each channel partner's performance beyond simple revenue metrics. Traditional methods—manual audits, spreadsheet-based analysis, or generic KPIs—are often insufficient, inconsistent, and unscalable. Hence, the aim was to introduce an objective, data-driven, and modular framework that can be embedded into the product architecture to evaluate retailer performance automatically and in near real-time.

To achieve this, the project focused on the following specific objectives:

1. Simulate Real-World Telecom Channel Data at Scale

- Develop realistic and varied dummy datasets to mimic actual data flowing through PreTUPS installations across countries. This includes:
    - Retailer master data
    - Daily transaction volumes
    - Channel feedback scores
    - Device and system logs
    - Behavioural flags
- Ensure that these datasets reflect the variety of conditions seen in real deployments, including high-volume markets, dormant agents, inconsistent feedback patterns, and fraud-like behaviour.

2. Engineer Multi-Dimensional Retailer Features

- Design a feature engineering pipeline that extracts operational, behavioural, transactional, and engagement-based features from raw datasets.
- Convert these into a unified retailer feature matrix, representing each retailer across 20+ indicators.
- Ensure that each feature contributes uniquely to the understanding of quality—e.g., transaction frequency, revenue stability, feedback ratings, device login consistency, and clustering anomaly scores.

3. Apply Unsupervised Machine Learning to Identify Patterns

- Implement clustering algorithms (such as KMeans or DBSCAN) to group retailers based on similar behavior or performance characteristics.
- Use unsupervised learning to:
    - Identify high-performing clusters
    - Flag outliers or suspicious patterns

•Enable business stakeholders to interpret clusters meaningfully using visualization tools.

4. Design a Composite Scoring Logic: The CQI Score

- Develop a **mathematical scoring function** that aggregates key features and cluster labels into a final, interpretable **Channel Quality Index (CQI) score**, ranging from 0 to 100.
- Normalise and weight various features based on importance and relevance.
- Ensure the score is **interpretable**, **scalable**, and **applicable across geographies**.
- Allow for **custom scoring logic extensions** to be added in the future, e.g., for specific operators or verticals (DTH, mobile money).

5. Create Business-Friendly Dashboards and Reports

- Build a **Streamlit dashboard** to visualise:
  - Retailer clusters
  - Fraud risk zones
  - Feedback heatmaps
  - Score distribution and rankings
- Export automated reports to share results with the PreTUPS product team, sales units, or customer success groups.
- Enable dynamic filtering and score interpretation at the **retailer**, **cluster**, or **regional** level.

In essence, the overarching goal of the project was to **build an intelligent system that bridges AI/ML with operational excellence**, providing PreTUPS with a scalable way to monitor channel quality, pre-emptively identify risks, and optimize partner performance.

This not only addresses current gaps in business intelligence and fraud monitoring but also lays the foundation for more advanced AI use cases, such as:

- Retailer churn prediction
- Feedback-based sentiment analysis
- Supervised fraud detection
- Geo-intelligence layer on top of CQI

The successful completion of this project allows Comviva and the PreTUPS team to begin a new phase of product intelligence, where channel evaluation becomes **automated, data-driven, and productised**—aligned with the company's long-term roadmap to make PreTUPS the most intelligent and analytics-powered distribution platform in the global telecom industry.

AI/ML Use Cases in the Channel Quality Index (CQI) Project

Artificial Intelligence (AI) and Machine Learning (ML) form the backbone of the Channel Quality Index (CQI) project, enabling scalable, data-driven insights into retailer quality, behavioral anomalies, and potential fraud within the PreTUPS ecosystem. The core objective of incorporating AI/ML techniques was to transform large volumes of raw channel data into meaningful, actionable intelligence that can guide operational decisions and future product innovations.

In traditional telecom channel monitoring, metrics are often isolated, reactive, and based on heuristics or manual thresholds. The CQI project addresses these limitations by embedding machine learning algorithms that can analyse patterns, identify emerging issues proactively, and rank retailers according to a comprehensive quality framework. Below are the **primary AI/ML use cases** developed and implemented during this project:

1. Feature Engineering for Retailer Profiling

A fundamental use case of ML in the project was to convert unstructured or semi-structured channel data into well-defined, high-value features that describe a retailer's quality, engagement, and risk.

- **Examples of Engineered Features:**
  - Average and variance in daily transaction volumes
  - Revenue consistency across time windows
  - Time since last login or device usage pattern
  - Number of customer complaints and feedback ratings
  - Behavioural drift flags (e.g., sudden drop in usage)
  - Transaction type diversity
  - Channel feedback score and response regularity

These features were processed using **automated data pipelines**, ensuring that data from different modules—transaction logs, feedback systems, device analytics—could be harmonised and analysed jointly. This engineered dataset acted as the **foundation for downstream ML models**.

2. Unsupervised Clustering to Identify Behavioral Segments

Due to the absence of labeled quality or fraud data (which is common in enterprise systems), **unsupervised learning** was employed to discover natural groupings of retailers based on behavioral patterns.

- **Algorithms Used:**
  - K-Means Clustering
  - DBSCAN (Density-Based Spatial Clustering)
  - Hierarchical Clustering (for experimentation)

- **Purpose of Clustering:**
  - Identify high-performing vs. low-performing retailers
  - Detect abnormal clusters that may represent dormant or fraudulent agents
  - Build explainable retailer segments (e.g., “high-volume loyal”, “dormant but active last week”, “erratic behavior”, etc.)

These clusters were later integrated into the CQI scoring logic, enabling **personalized benchmarking** and **cluster-specific scoring models**.

3. Anomaly Detection and Fraud Risk Indicators

Although explicit fraud labels were not available in the simulated data, the project implemented **anomaly detection logic** using cluster outliers and behavioral drift features. The idea was to pre-emptively flag retailers showing patterns that deviate from expected norms.

- **Anomaly Indicators:**
  - Outliers in transaction consistency
  - Irregular device login patterns (e.g., multiple geographies)
  - Spikes in refund requests or failed transactions
  - Sudden inactivity after periods of high engagement
  - Feedback mismatch: High volume but low ratings

These indicators can be plugged into **future supervised fraud detection models**, or directly visualized in the dashboard for operational teams to investigate.

4. CQI Score Computation Using AI-Derived Insights

One of the most novel use cases was the computation of the final **Channel Quality Index (CQI)** score, which combined multiple AI-derived components into a single metric (0–100 scale). This score was not a simple average but a **weighted aggregation of ML-driven features** and clustering results.

- Features contributing to the score:
  - Behavioral consistency (standard deviation of activity)
  - Cluster quality (cluster centroid distance or density)
  - Engagement frequency
  - Customer feedback metrics
  - Device and login reliability

This made the CQI score both **quantitative** and **context-aware**, capturing not just volume but also stability, reliability, and customer-facing performance.

5. Visualization and Interpretation of AI Outputs

To make AI/ML outputs usable by product managers and business analysts, all models were tied to a **Streamlit-based visualization dashboard**. The visual layer translated complex ML outputs into human-readable charts and tables, supporting:

- Cluster visualizations (via PCA or t-SNE projection)
- Score distributions across regions
- Top vs. bottom retailer performance
- Risk-flag heatmaps
- Region-wise quality rankings

This helped bridge the gap between **technical ML models** and **real-world product interpretation**, which is critical in enterprise deployments like PreTUPS.

6. Pipeline Automation for Model Reusability

A final but crucial use case was **automation**. All ML logic—from feature generation to clustering and scoring—was integrated into a **modular Python pipeline**. This allows the PreTUPS team to:

- Re-run the entire ML workflow for different countries
- Swap datasets and re-cluster for new deployments
- Extend or replace models without breaking the pipeline

This approach ensures the CQI project is not a one-off exercise but a **repeatable, adaptable ML module** that can be embedded in the product or used in future enhancements like:

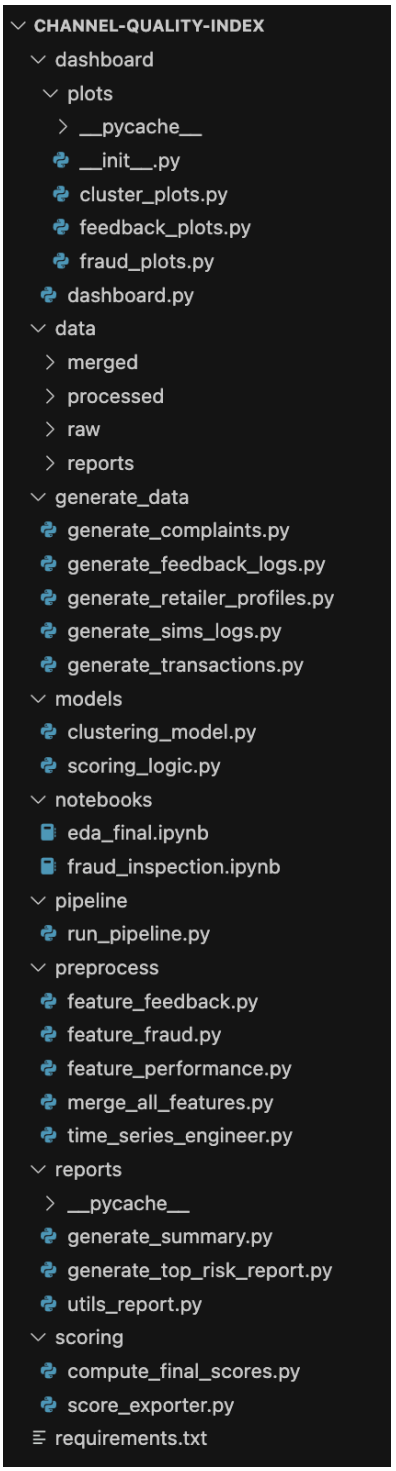
- Fraud prediction models (with labeled training data)
- Feedback-based sentiment NLP
- Geo-based retailer mapping and hotspot analysis
- Reinforcement learning for incentive recommendation

In Summary:

The use of AI and ML in the CQI project was not just limited to experimentation—it played a central role in building a **scalable, intelligent, and interpretable scoring system**. Through unsupervised learning, anomaly detection, feature engineering, and pipeline automation, the project succeeded in demonstrating how machine learning can be operationalized in enterprise software, providing **real business value** to Comviva and its PreTUPS ecosystem.

Project Structure

```
ChannelQualityIndex/
│
├── dashboard/
│   ├── dashboard.py
│   └── plots/
│       ├── __init__.py
│       ├── cluster_plots.py
│       ├── feedback_plots.py
│       └── fraud_plots.py
│
├── data/
│   ├── raw/          # Simulated raw datasets (generated)
│   ├── processed/    # Cleaned feature datasets
│   ├── merged/       # Final merged feature matrix
│   └── reports/      # Output reports
│
├── generate_data/
│   ├── generate_transactions.py
│   ├── generate_feedback_logs.py
│   ├── generate_sims_logs.py
│   ├── generate_complaints.py
│   └── generate_retailer_profiles.py
│
├── models/
│   ├── clustering_model.py
│   └── scoring_logic.py
│
├── notebooks/
│   ├── eda_final.ipynb
│   └── fraud_inspection.ipynb
│
├── pipeline/
│   └── run_pipeline.py
│
├── preprocess/
│   ├── feature_feedback.py
│   ├── feature_fraud.py
│   ├── feature_performance.py
│   ├── merge_all_features.py
│   └── time_series_engineer.py
│
├── reports/
│   ├── generate_summary.py
│   ├── generate_top_risk_report.py
│   └── utils_report.py
│
├── scoring/
│   ├── compute_final_scores.py
│   └── score_exporter.py
│
├── LICENSE
├── README.md
└── requirements.txt
```



Module-wise Execution and Logic

The project was developed using a modular, well-organized structure with each folder responsible for a distinct stage in the AI/ML data pipeline. Each module in the CQI project contributes to a specific functionality — from data simulation to preprocessing, modeling, dashboarding, and reporting. Below is a comprehensive description of the role, logic, and execution of each module:

1. generate_data/ – Data Simulation Module

This module simulates realistic raw datasets that serve as the foundation for downstream processing. Given the absence of access to real transactional or operational data due to privacy and confidentiality constraints, the raw data was synthetically generated to mimic real-world behavior of retailers, customer feedback, SIM registration logs, complaints, and transactional patterns. Each script inside this module corresponds to a distinct data stream:

- generate_transactions.py: Simulates transactional logs per retailer, including recharge volume, success/failure rates, and revenue.
- generate_feedback_logs.py: Simulates customer feedback data with sentiment scores, channel experience ratings, and NPS.
- generate_sims_logs.py: Generates logs related to SIM sale activity and activation rates.
- generate_complaints.py: Creates simulated records of customer complaints including complaint category, resolution time, and severity.
- generate_retailer_profiles.py: Builds static metadata such as location, registration age, and historical engagement scores.

Each dataset is saved under data/raw/ with over 1000 records per file to ensure statistical relevance.

2. preprocess/ – Feature Engineering Module

The preprocessing module is a critical part of the CQI pipeline. It transforms raw data into structured feature datasets used by clustering and scoring models. It includes several scripts tailored to different aspects of retailer behavior:

- feature_performance.py: Extracts features from transaction logs like average recharge volume, success rate, failure variance, and frequency of use.
- feature_feedback.py: Converts qualitative feedback into numerical sentiment scores, aggregates ratings, and handles missing feedback data using imputation techniques.
- feature_fraud.py: Constructs fraud-related features using heuristics like transaction spike detection, SIM-churning frequency, and unusual complaint patterns.
- time_series_engineer.py: Creates rolling averages, moving standard deviations, and time-windowed patterns for detecting anomalies in retailer behavior over time.
- merge_all_features.py: Merges all the engineered features from above into a unified feature matrix saved in data/merged/retailer_feature_matrix.csv.

The preprocessing logic includes outlier treatment, normalization, and type-casting to ensure model readiness.

3. models/ – Clustering and Scoring Module

This module applies AI/ML logic to convert feature matrices into actionable insights. It includes:

- clustering_model.py: Applies unsupervised learning (KMeans and Silhouette Score optimization) to segment retailers into behavior-based clusters such as “High-Quality”, “Moderate Risk”, and “High Risk”. The clustering also informs how scores are distributed across the population.

- scoring_logic.py: Assigns a Channel Quality Index (CQI) score to each retailer by combining cluster information, key KPIs, and fraud probabilities. The score is normalized and calibrated between 0 and 100.

This module bridges raw engineered data to quantifiable quality metrics via data science.

4. scoring/ – Score Computation Module

The scoring module is responsible for finalizing CQI scores and exporting them for reports and dashboards.

- compute_final_scores.py: Aggregates KPIs, risk factors, and behavioral metrics into a final CQI score using weighted formula logic (customizable). This step includes score smoothing and penalization for fraud-prone behaviors.

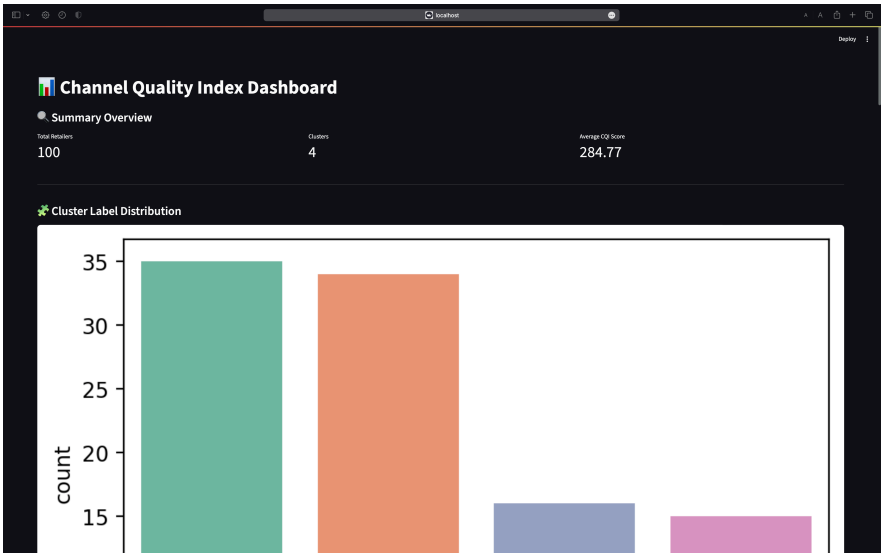
- score_exporter.py: Formats the scores and outputs them into data/reports/ as CSV and Excel files, ready for visualization and PDF reporting.

This modular design allows the scores to be easily updated or tuned based on new data inputs or evolving risk policies.

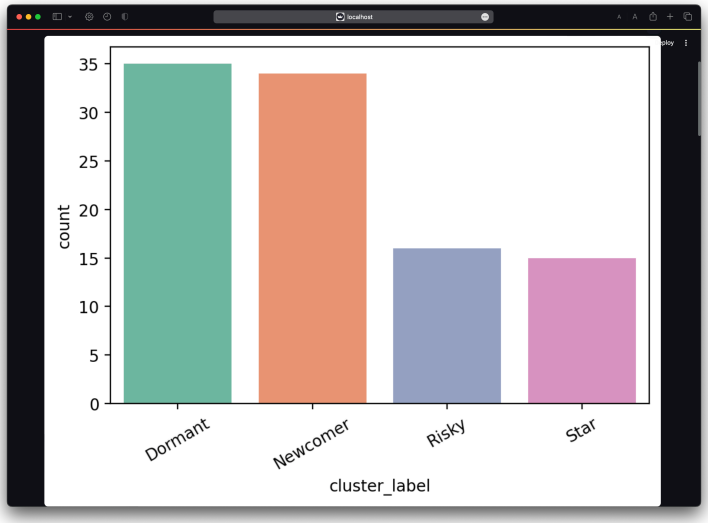
5. dashboard/ – Interactive Visualization Module

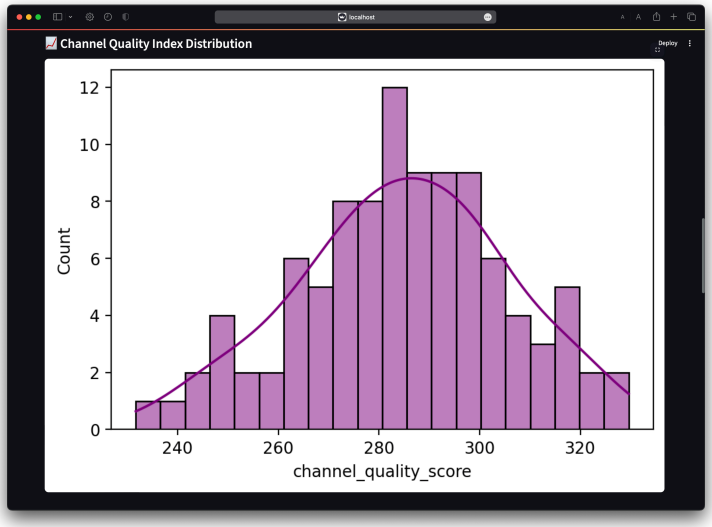
To make CQI insights interpretable and accessible, this module creates a Streamlit-based dashboard along with modularized plots:

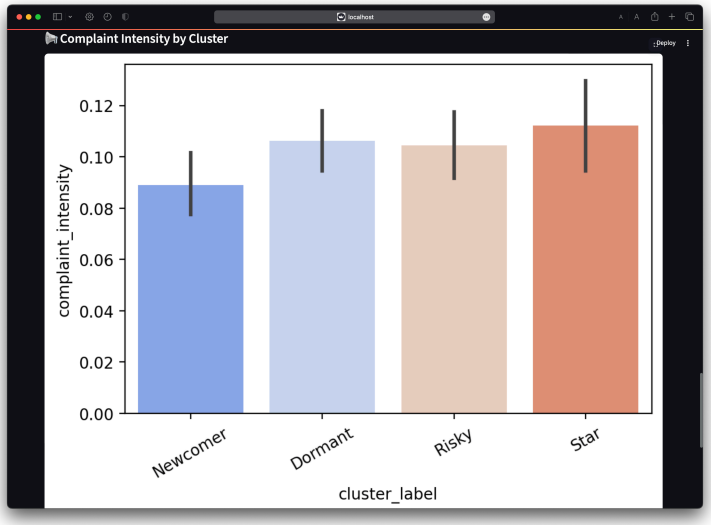
- dashboard.py: The main Streamlit app that visualizes CQI score distributions, top risky retailers, feedback trends, and fraud signals. It reads directly from exported scores and plots folders.



- plots/cluster_plots.py: Shows cluster-level score distributions, centroids, and cluster sizes.
- plots/feedback_plots.py: Displays sentiment trends, feedback score histograms, and NPS trends.
- plots/fraud_plots.py: Highlights fraud-prone retailers, spike patterns, and outlier detection visuals.







The dashboard acts as the project's visual front-end and can be extended for internal demo or executive insights.

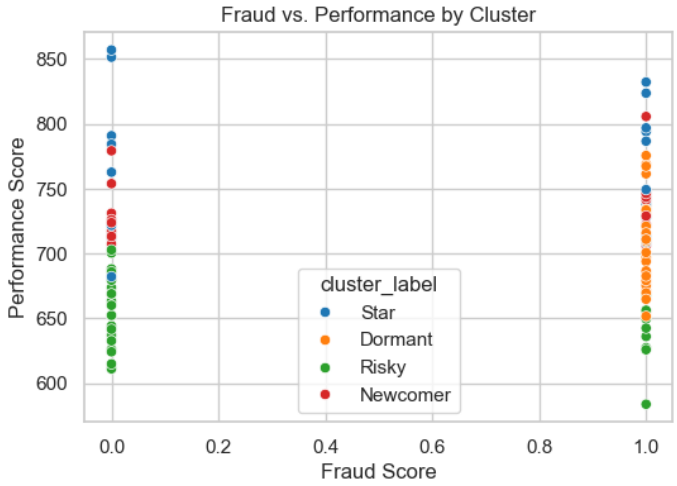
6. notebooks/ – Exploratory and Analytical Module

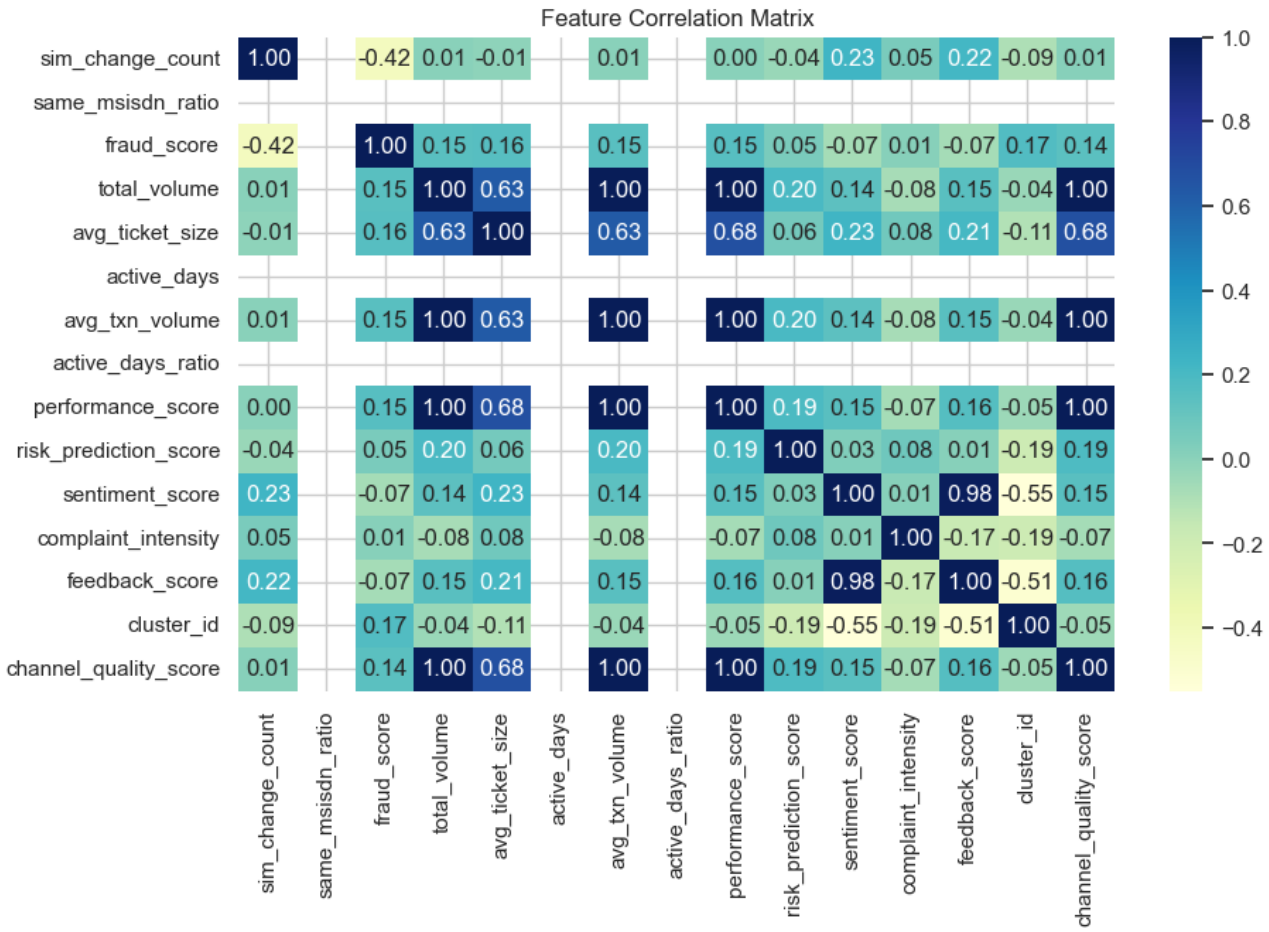
Two Jupyter Notebooks were created for internal analysis and validation:

- eda_final.ipynb: Used during development to perform exploratory data analysis (EDA), feature correlation heatmaps, and data distribution checks.

- fraud_inspection.ipynb: Focuses on understanding fraud detection signals and visualizing outliers across the dataset.

These notebooks ensure the technical validation of preprocessing and scoring decisions.





7. pipeline/ – Automation Pipeline Module

To streamline the end-to-end project workflow, a master pipeline was developed:

- run_pipeline.py: Automates the entire process from data generation → preprocessing → feature merging → modeling → scoring → reporting. Also includes cleanup logic to remove stale files and regenerate updated outputs.

The pipeline ensures reproducibility, standardization, and scalability of the project for future deployments or dataset variations.

8. reports/ – Reporting and Summary Module

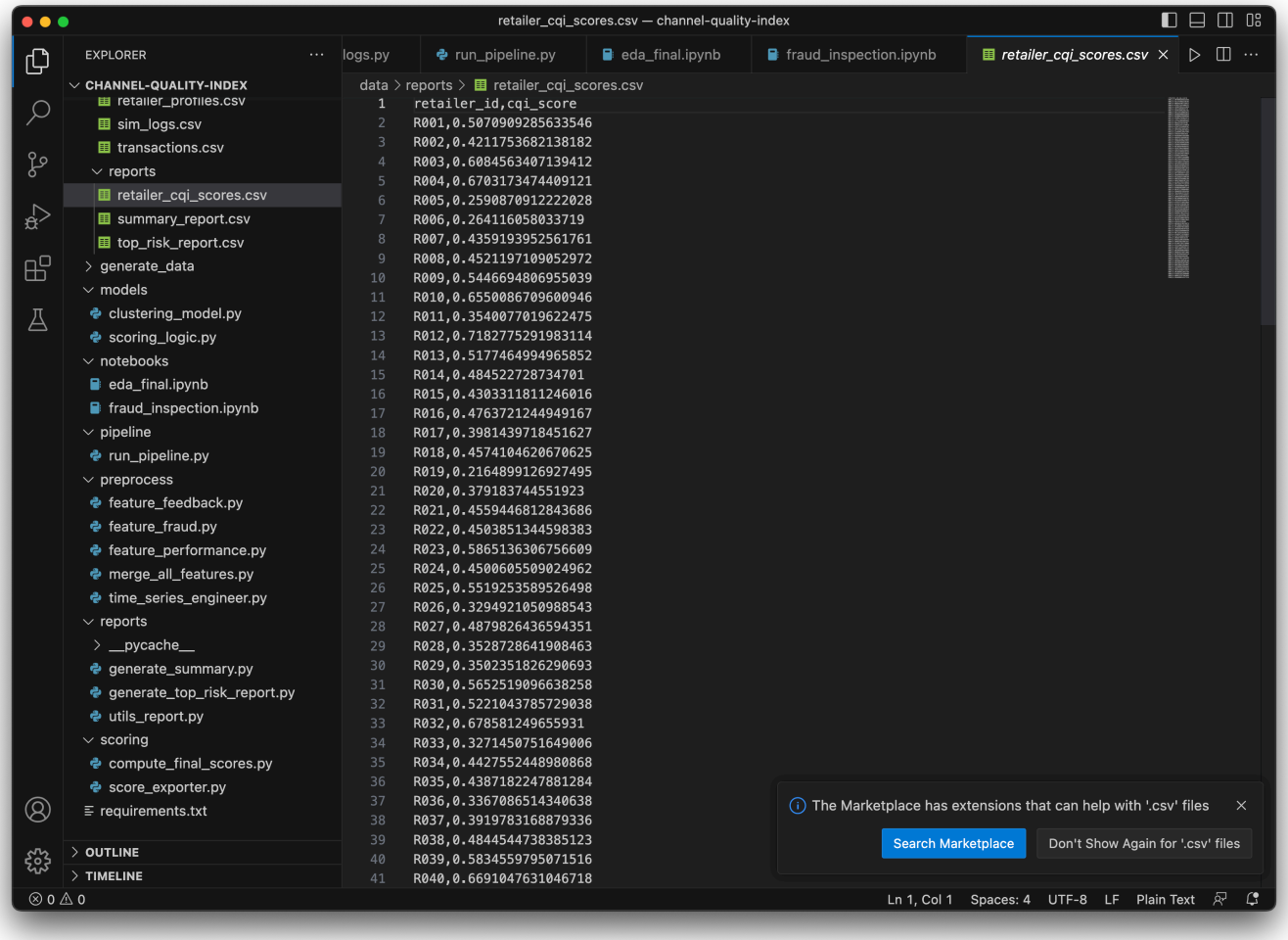
This module programmatically generates reports from the final scores:

- generate_summary.py: Creates CSV/Excel reports that summarize channel performance, fraud impact, and NPS insights.

- generate_top_risk_report.py: Identifies and lists the top N high-risk retailers using combined score heuristics.

- utils_report.py: Contains common utility functions for formatting, filtering, and exporting report content.

These reports can be exported in PDF format or integrated with email alert systems for business consumption.



top_risk_report.csv — channel-quality-index

EXPLORER

CHANNEL-QUALITY-IN...

- retailer_profiles.csv
- sim_logs.csv
- transactions.csv
- reports
  - retailer_cqi_scores.csv
  - summary_report.csv
  - top_risk_report.csv
- generate_data
- models
  - clustering_model.py
  - scoring_logic.py
- notebooks
  - eda_final.ipynb
  - fraud_inspection.ipynb
- pipeline
  - run_pipeline.py
- preprocess
  - feature_feedback.py
  - feature_fraud.py
  - feature_performance.py
  - merge_all_features.py
  - time_series_engineer.py
- reports
  - __pycache__
  - generate_summary.py
  - generate_top_risk_report.py
  - utils_report.py
- scoring
  - compute_final_scores.py
  - score_exporter.py
- requirements.txt

OUTLINE

TIMELINE

logs.py | run_pipeline.py | eda_final.ipynb | fraud_inspection.ipynb | top_risk_report.csv

data > reports > top_risk_report.csv

```
retailer_id,sim_change_count,complaint_count,fraud_score,total_volume,avg_ticket_size,active_days
R094,14,17,0.9777777777777776,24912.84,239.5465384615385,20,1245.642,1.0,694.8849615384617,0.0,-0
R057,12,17,0.892063492063492,28813.01,269.28046728971964,20,1440.6505,1.0,801.309390186916,0.1428
R074,14,13,0.8888888888888888,22626.67,235.6944791666667,20,1131.3335000000002,1.0,636.5750937500
R049,14,13,0.8888888888888888,25612.03,275.39817204301073,20,1280.6015,1.0,723.1202016129032,0.57
R052,14,13,0.8888888888888888,22980.73,258.21044943820226,20,1149.0365,1.0,652.1813848314607,0.57
R083,13,14,0.8682539682539683,24435.95,262.7521505376344,20,1221.7975,1.0,689.9243951612905,0.285
R048,11,17,0.8492063492063492,28592.02,267.2151401869159,20,1429.601,1.0,795.1650420560749,1.0,0.
R071,12,15,0.8476190476190476,28143.32,270.6088461538462,20,1407.166,1.0,784.9656538461538,0.7142
R089,13,13,0.8460317460317461,23238.51,242.0678125,20,1161.9254999999998,1.0,653.7830937499999,0.
R055,13,12,0.8238095238095238,23076.04,253.5828571428572,20,1153.802,1.0,653.1758571428572,0.5714
R006,11,15,0.8047619047619048,24619.02,261.9044680851064,20,1230.951,1.0,694.246840425532,0.7142
R076,12,13,0.8031746031746032,23758.87,232.9300980392157,20,1187.9435,1.0,664.0507794117647,0.857
R019,12,13,0.8031746031746032,22465.74,218.11398058252428,20,1123.287,1.0,627.2776941747574,0.714
R065,14,9,0.8,22372.69,219.3400980392157,20,1118.6345,1.0,625.3192794117647,0.8571428571428571,0.
R046,12,11,0.7587301587301587,28927.48,262.9770909090909,20,1446.374,1.0,802.2801272727273,0.4285
R050,12,11,0.7587301587301587,26656.9,240.15225225225225,20,1332.845,1.0,738.6681756756757,0.4285
R054,12,11,0.7587301587301587,23324.75,235.60353535353536,20,1166.2375,1.0,653.9998106060607,0.57
R082,13,9,0.7571428571428571,25728.62,252.2413725490196,20,1286.431,1.0,719.087911764706,0.857142
R026,13,9,0.7571428571428571,23532.7,245.13229166666667,20,1176.635,1.0,662.0571875,0.42857142857
R090,14,7,0.7555555555555555,25225.51,268.35648936170213,20,1261.2755,1.0,711.3446968085107,0.571
```

The Marketplace has extensions that can help with '.csv' files

Search Marketplace | Don't Show Again for '.csv' files

0 0 | Ln 1, Col 1 | Spaces: 4 | UTF-8 | LF | Plain Text

Pipeline Flow (End-to-End Execution Architecture)

The pipeline designed and implemented for the Channel Quality Index (CQI) project is a fully automated, modular system that performs sequential execution of all critical stages in the AI/ML workflow. The pipeline not only simulates and processes large-scale telecom channel data but also engineers features, applies unsupervised learning, and outputs interpretable metrics through automated reports and dashboards. It was designed with **repeatability**, **scalability**, and **modular debugging** in mind, which aligns with software engineering best practices.

The core execution script, run_pipeline.py, resides in the pipeline/ directory. This script governs the entire orchestration of modules spanning from raw data generation to final scoring and visualizations. Each module is independently executable but has also been integrated seamlessly into the centralized pipeline architecture.

The **step-by-step flow of the pipeline is as follows**:

1. Data Cleanup Phase

At the beginning of every pipeline run, the system performs a cleanup routine to ensure a fresh working environment. This includes deleting any pre-existing files in the data/raw/, data/processed/, data/merged/, and data/reports/ directories. This design decision ensures that old data does not interfere with new outputs and guarantees data consistency during testing or retraining phases

2. Data Generation Phase (generate_data/)

In this phase, the pipeline invokes multiple scripts to simulate five different types of raw datasets representing telecom channel behavior:

- **generate_transactions.py**: Simulates recharge and sale transactions for each retailer.
- **generate_feedback_logs.py**: Generates logs based on customer and field feedback, with different issue tags and timestamps.
- **generate_sims_logs.py**: Simulates SIM card sales and activations, capturing product uptake and channel utilization.
- **generate_complaints.py**: Models consumer complaint patterns across various categories (e.g., failed recharges, incorrect balances).
- **generate_retailer_profiles.py**: Builds retailer demographic and operational profiles (e.g., registration age, urban/rural classification, language preference).

Each generated CSV contains **at least 1000 rows** and is saved under data/raw/.

3. Feature Engineering Phase (preprocess/)

This is one of the most critical stages in the pipeline. It transforms the raw data into structured, domain-informed features:

- **feature_feedback.py**: Parses feedback logs and complaints to calculate complaint density, average resolution time, and sentiment-based scores.

- **feature_fraud.py**: Detects anomalies and red flags across transactional and complaint datasets by flagging behavioral outliers.

- **feature_performance.py**: Derives key performance indicators such as transaction velocity, SIM sales-to-activation ratio, success/failure rates, and average revenue per retailer.

- **time_series_engineer.py**: Adds trend-based and seasonality-aware features over time using moving averages and volatility computations.

- **merge_all_features.py**: Merges all the processed feature sets into a unified dataset (retailer_feature_matrix.csv), which becomes the input to downstream modeling.

The processed features are saved in data/processed/ and the merged feature matrix is stored in data/merged/.

4. Modeling and Scoring Phase (models/ and scoring/)

This phase applies unsupervised learning and multi-dimensional scoring:

- **clustering_model.py**: Performs KMeans clustering to identify behavioral cohorts of retailers based on multi-variate patterns. Cluster labels are added to the dataset to support peer group analysis.

- **scoring_logic.py**: Translates complex multi-feature data into intuitive scores for various dimensions—such as Service Quality Score, Complaint Score, Fraud Risk Score, and Feedback Score.

Following clustering and logic formulation:

- **compute_final_scores.py** (under scoring/) aggregates individual scores into a **composite CQI Score** using weighted formulas based on domain relevance.

- **score_exporter.py**: Saves and exports final scores and ranks into clean CSVs for visualization or reporting.

5. Report Generation Phase (reports/)

Once the scoring is complete, automated reports are generated to summarize insights:

- **generate_summary.py**: Compiles a descriptive report covering average CQI scores, cluster distributions, and top-performing and underperforming retailers.

- **generate_top_risk_report.py**: Highlights the top 50 retailers with the highest fraud risk or lowest CQI scores.

- **utils_report.py**: Houses helper functions for formatting, styling, and exporting consistent visual and tabular summaries.

Reports are exported to the data/reports/ directory in both CSV and optional PDF formats.

6. Dashboard Visualization Phase (dashboard/)

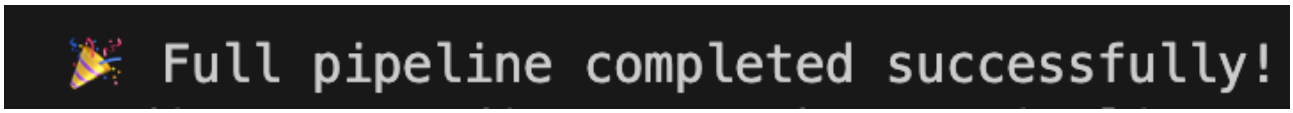
The pipeline concludes by triggering dashboard/dashboard.py, which renders interactive visualizations using matplotlib, seaborn, and plotly. Plots are modularized for easy debugging and extension:

- **cluster_plots.py**: Displays cluster-wise CQI distribution, elbow curves, and cluster centroids.
- **feedback_plots.py**: Plots feedback category breakdowns and complaint trends.
- **fraud_plots.py**: Shows fraud risk percentile plots, feature importance, and box plots by cluster.

All plots are stored under dashboard/plots/.

Orchestration and Modularity

Each module can be executed independently for debugging or testing. However, run_pipeline.py enables a **single-command execution** of the entire project, making it user-friendly for non-technical stakeholders and deployable in production environments (e.g., via cron jobs or CI/CD triggers). This structure also makes it suitable for integration with a **web dashboard**, **API layer**, or **data warehouse** in future enhancements.



localhost

cluster_label

🏆 Top 5 Retailers by CQI Score

	retailer_id	cqi_score	cluster_label
35	R036	0.7558	Star
40	R041	0.7459	Star
12	R013	0.7454	Newcomer
67	R068	0.7305	Star
66	R067	0.7282	Star

🚨 High Risk Retailers (Fraud Score)

	retailer_id	fraud_score
11	R012	1
63	R064	0.9101
10	R011	0.9101
39	R040	0.8801
84	R085	0.879

🔎 Explore a Cluster

Select Cluster

Dormant

localhost

Deploy

📋 Final CQI Scores by Retailer

	retailer_id	channel_quality_score
26	R027	333.1847
66	R067	329.7373
32	R033	323.1346
75	R076	321.9974
40	R041	321.5583
98	R099	320.9582
43	R044	317.9696
3	R004	317.5291
6	R007	316.6082
2	R003	312.4175

🔍 Full retailer data Table

	retailer_id	sim_change_count	complaint_count	fraud_score	total_volume	avg_ticket_size	active_days	avg_txn_volume	active_days_ratio	performance_score	risk_prediction_score	sentiment_
16	R017	14	10	0.7602	27056.31	284.8033	20	1352.8155	1	0.7522	0.7143	
17	R018	10	13	0.606	24396.12	248.94	20	1219.806	1	0.4795	0	
18	R019	5	19	0.4807	26743.92	272.8971	20	1337.196	1	0.712	0.4286	
19	R020	8	6	0.2741	23852.59	245.903	20	1192.6295	1	0.4284	0.7143	
20	R021	11	5	0.4272	26785.32	239.1546	20	1339.266	1	0.6799	0.4286	
21	R022	4	8	0.0899	20764.81	247.2001	20	1038.2405	1	0.1573	0.5714	
22	R023	14	13	0.8501	24672.33	257.0034	20	1233.6165	1	0.5124	0.2857	
23	R024	9	10	0.455	24030.6	261.2022	20	1201.53	1	0.4603	0.8571	
24	R025	6	6	0.152	24428.46	265.5267	20	1221.423	1	0.4999	0.2857	
25	R026	11	10	0.5771	25732.78	273.753	20	1286.639	1	0.6237	0.7143	

Conclusion

The Channel Quality Index (CQI) – AI/ML Project for PreTUPS stands as a significant step forward in enhancing the operational transparency, data-driven decision-making, and channel performance management capabilities within Comviva's flagship distribution platform. This internship project was conceived to solve a real-world business problem: the lack of a unified, objective, and interpretable metric that could effectively summarize the performance, reliability, and integrity of telecom channel partners (such as retailers, agents, and regional operators) across multiple dimensions.

Through the rigorous application of data science, machine learning, and domain-specific heuristics, the project culminated in the development of an end-to-end CQI system — capable of ingesting raw multi-source data, engineering meaningful features, detecting anomalies and patterns using unsupervised ML, and generating intuitive dashboards and reports for both technical and non-technical stakeholders. From simulating and preprocessing diverse data streams such as transaction logs, complaints, SIM activations, and retailer profiles to building clustering-based profiling and risk-scoring pipelines, every step of the project was tailored to real organizational needs. Importantly, the scoring mechanism was designed to be modular, interpretable, and scalable, which makes it easy for the PreTUPS engineering team to expand it across deployments and incorporate future feedback loops.

The project also reflects the crucial role that AI/ML can play in augmenting decision-making within B2B telecom platforms. Rather than replacing domain knowledge, the model complements it—highlighting retailers that need managerial attention, surfacing suspicious behavior, and supporting KPI reviews in a data-first manner. The unsupervised clustering approach allowed the system to adapt to hidden patterns in the behavior of channel actors without relying on pre-labeled historical fraud or underperformance data—this is particularly critical in real-world telecom deployments where such labels are rare, noisy, or politically sensitive.

The Streamlit dashboard made it possible to present these insights in a highly interactive and business-friendly format, reducing the technical barrier to adoption. The visual cluster breakdown, fraud flags, feedback summaries, and top-risk reports allow stakeholders ranging from product managers to regional sales leaders to directly act on insights without relying heavily on manual Excel analyses or static reports.

From a learning perspective, this project enabled a deep dive into end-to-end machine learning pipelines, modular software engineering, unsupervised learning, and data visualization—all within a domain as complex and regulated as telecom. It bridged technical implementation with real-world product objectives, which is often a rare opportunity in an internship setting. Every component, from data simulation to final CQI scoring, was built with extensibility and robustness in mind, ensuring that it can be integrated into existing PreTUPS ecosystems or extended into future feature releases such as automated channel interventions, time-series risk tracking, and alert generation.

In conclusion, the CQI system delivers value not just through automation, but through intelligent summarization, stakeholder alignment, and data democratization. It lays the groundwork for long-term AI-driven improvements in the governance of distribution channels at Comviva. The project represents a robust, modular, and scalable framework that can evolve into a core analytics engine within PreTUPS, reinforcing Comviva's position as a global leader in telecom distribution technology.