**Leveraging Machine Learning for Microsoft Teams Call Quality Enhancement**

**Predictive Analytics and Issue Resolution**

**The Challenge: Proactive Call Quality Management**

* **Ensuring high-quality communication is paramount for productivity.**
* **Poor call quality leads to frustration, lost productivity, and increased support costs.**
* **Traditional reactive troubleshooting is time-consuming and often too late.**
* **Solution: Employing Machine Learning to move from reactive to proactive and predictive call quality management.**

**Data Sources & Key Metrics**

**Microsoft Teams Call Quality Dashboard (CQD) and Call Analytics provide rich telemetry:**

* **Dimensions & Measures:**
  + **Network (e.g., Reflexive Local IP, Subnet, Reflexive Transport, Round Trip Time, Jitter, Packet Loss).**
  + **User & Device (e.g., User Id, Endpoint Name, Device Type, Capture Device, Render Device, OS, CPU Name, Processor Speed).**
  + **Media (e.g., Media Type - Audio/Video/VBSS, Codec, Frame Loss Percentage).**
  + **Call State (e.g., Call State, QoE Record Available).**
  + **User Feedback (e.g., Star rating).**
  + ***Reference:*** [**Dimensions and measures available in Call Quality Dashboard**](https://learn.microsoft.com/en-us/microsoftteams/dimensions-and-measures-available-in-call-quality-dashboard)
* **Stream Classification: CQD classifies streams as Good, Poor, or Unclassified based on thresholds for RTT, Packet Loss, Jitter, etc.**
  + ***Reference:*** [**Stream classification in Call Quality Dashboard**](https://learn.microsoft.com/en-us/microsoftteams/stream-classification-in-call-quality-dashboard)
* **Intelligent Media Quality Classifiers (ML-driven): CQD already uses ML to pinpoint specific problem areas (Poor Microphone, Audio Glitches, Video Freezes, Frozen Video, Insufficient Bandwidth, Excessive CPU) across network, compute, and input devices (Local/Remote).**
  + ***Reference:*** [**CQD intelligent media quality classifiers**](https://learn.microsoft.com/en-us/microsoftteams/cqd-intelligent-media-quality-classifiers)

**1) Detect, Analyze & Predict: Reflexive Network & Subnets**

**Problem: Identifying network performance bottlenecks causing poor call quality, especially related to the local network (reflexive IPs) and specific subnets.**

**Key Data Points: Reflexive Local IP, Reflexive Transport, Subnet, Round Trip Time (RTT), Jitter, Packet Loss Rate.**

**Machine Learning Models:**

* **Anomaly Detection (Unsupervised/Semi-supervised):**
  + **Models: Isolation Forest, One-Class SVM, Autoencoders.**
  + **Application: Identify unusual spikes or sustained high values in RTT, Jitter, or Packet Loss from specific reflexive IPs or subnets that deviate significantly from historical norms, indicating a localized network issue.**
* **Classification (Supervised):**
  + **Models: Decision Trees, Random Forests, Gradient Boosting Machines (XGBoost, LightGBM).**
  + **Application: Classify calls as "Good" or "Poor" (using CQD's Stream Classification as target) based on network metrics. The models will highlight which network features (e.g., high packet loss on a specific subnet) are most influential in predicting poor quality.**
* **Regression (Supervised):**
  + **Models: Linear Regression, Ridge/Lasso Regression, Neural Networks.**
  + **Application: Predict continuous quality scores (e.g., estimated MOS score if available, or a composite quality index) based on network metrics, allowing for more granular prediction of quality degradation.**

**Visualization:**

* **Heatmaps: Call quality (e.g., percentage of poor calls) per Subnet or Reflexive Local IP.**
* **Time-Series Graphs: Trends of RTT, Jitter, Packet Loss over time, segmented by subnet.**
* **Scatter Plots: Packet Loss vs. Jitter vs. Call Quality to identify clusters of problematic network conditions.**
* **Geospatial Maps: Overlay network quality issues onto geographic locations if subnet mapping is available.**

**2) Detect, Analyze & Predict: Users, Devices, Media Type**

**Problem: Understanding how individual users, specific devices, and different media types (audio, video, screen share) impact call quality.**

**Key Data Points: User Id, Endpoint Name, Device Type, Media Type (audio, video, VBSS), Codec, Capture Device, Render Device, Intelligent Media Quality Classifiers (e.g., Poor Microphone, Video Freezes).**

**Machine Learning Models:**

* **Classification (Supervised):**
  + **Models: Decision Trees, Random Forests, Support Vector Machines (SVMs), Logistic Regression.**
  + **Application: Predict if a call will be "Poor" based on Device Type, Capture/Render Device, Media Type. This helps pinpoint which devices or media scenarios are problematic.**
* **Clustering (Unsupervised):**
  + **Models: K-Means, DBSCAN, Hierarchical Clustering.**
  + **Application: Group users or devices based on their historical call quality patterns and associated Intelligent Media Quality Classifiers. This can reveal cohorts of users experiencing similar issues, or identify device models with inherent problems.**
* **Association Rule Mining (Unsupervised):**
  + **Models: Apriori, Eclat.**
  + **Application: Discover relationships between device types, software versions, and specific quality issues (e.g., "Users with Device X and OS Version Y frequently experience Poor Microphone issues").**

**Visualization:**

* **Bar Charts: Count of Poor calls by Device Type, Media Type, or specific Capture/Render Device.**
* **Pie Charts/Treemaps: Proportion of different Intelligent Media Quality Classifiers (e.g., how many poor calls were due to Poor Microphone vs. Insufficient Bandwidth).**
* **User-Specific Dashboards: Provide drill-down views for individual users showing their call history and associated quality metrics.**

**3) Feedback Analysis & Correlation with Other Call Quality Metrics**

**Problem: Bridging the gap between objective call quality metrics and subjective user experience.**

**Key Data Points: User Feedback (e.g., 1-5 star rating from post-call survey), CQD Metrics (RTT, Jitter, Packet Loss, MOS, Intelligent Media Quality Classifiers).**

**Machine Learning Models:**

* **Sentiment Analysis (if free-text feedback is available):**
  + **Models: Naive Bayes, SVM, Transformer models (e.g., BERT for more complex text).**
  + **Application: Categorize open-ended user comments as positive, negative, or neutral.**
* **Regression (Supervised):**
  + **Models: Linear Regression, Random Forests, Neural Networks.**
  + **Application: Predict the User Feedback rating based on objective CQD Metrics. This helps determine which technical metrics most strongly influence user perception of quality.**
* **Correlation Analysis (Statistical/ML):**
  + **Techniques: Pearson Correlation Coefficient, Spearman's Rank Correlation.**
  + **Application: Quantify the statistical relationship between User Feedback and various CQD Metrics to identify strong drivers of user satisfaction/dissatisfaction.**

**Visualization:**

* **Scatter Plots: User Feedback Score vs. MOS Score or Packet Loss to visualize correlation.**
* **Box Plots/Violin Plots: Distribution of CQD Metrics for different User Feedback categories (e.g., "1-star feedback" vs. "5-star feedback").**
* **Word Clouds/Topic Models: (If free-text feedback) Highlight common themes or issues mentioned by users in negative feedback.**

**4) Call Drops and Setup Failures Analysis**

**Problem: Identifying the root causes and predicting the likelihood of call drops and setup failures.**

**Key Data Points: Call State (e.g., Dropped, Failed to Setup), Session ID, Endpoint Name, Network Type, Signaling Protocol, Authentication Type, QoE Record Available (false for failures).**

**Machine Learning Models:**

* **Classification (Supervised):**
  + **Models: Logistic Regression, Random Forests, Gradient Boosting, Neural Networks.**
  + **Application: Predict the probability of a call drop or setup failure based on pre-call (e.g., device health, network type) and early-call metrics.**
* **Decision Tree/Rule-based Models (Supervised):**
  + **Models: C4.5, CART.**
  + **Application: Generate interpretable rules that explain why calls drop or fail (e.g., "If Network Type is VPN AND Endpoint Name is VDI Client, then Call Drop probability is high"). These are excellent for root cause analysis.**
* **Survival Analysis (Specialized):**
  + **Models: Cox Proportional Hazards Model.**
  + **Application: Analyze the "time to failure" (how long a call lasts before dropping) and identify factors that shorten call duration or increase the risk of an early drop.**

**Visualization:**

* **Funnels Charts: Illustrate success/failure rates at different stages of call setup.**
* **Drill-down Dashboards: Allow investigation into specific Call State categories to see contributing factors.**
* **Geographic Maps: Identify regions or office locations with higher rates of call drops/failures.**

**5) Device Issues and Its Analysis**

**Problem: Deep dive into specific device-related problems impacting call quality.**

**Key Data Points: Capture Device, Render Device, Device Vendor, Driver Version, OS, CPU Name, Processor Speed, Intelligent Media Quality Classifiers (e.g., Poor Microphone, Audio Glitches, Video Freezes).**

**Machine Learning Models:**

* **Classification/Clustering (Supervised/Unsupervised):**
  + **Models: Decision Trees, Random Forests, K-Means.**
  + **Application:**
    - **Classification: Predict which Intelligent Media Quality Classifiers (e.g., Poor Microphone) are associated with specific Capture/Render Devices or Driver Versions.**
    - **Clustering: Group devices based on their performance profiles and quality issues, identifying "bad batches" or models that consistently underperform.**
* **Time-Series Analysis (Anomaly Detection):**
  + **Models: ARIMA, Prophet, LSTM.**
  + **Application: Monitor trends in device-specific quality metrics (e.g., Capture Device Jitter) over time to detect gradual degradation or sudden performance drops for a particular device type or model.**

**Visualization:**

* **Device Performance Leaderboards/Heatmaps: Rank devices by average call quality or frequency of issues.**
* **Trend Lines: Show performance evolution of specific device models or driver versions over time.**
* **Scatter Plots: CPU Usage vs. Video Frame Rate to identify performance bottlenecks.**
* **Bar Charts: Frequency of Intelligent Media Quality Classifiers per device model or driver version.**

**6) Timeline-Based Analysis**

**Problem: Understanding temporal patterns, identifying trends, and predicting future call quality.**

**Key Data Points: All CQD metrics with associated timestamps.**

**Machine Learning Models:**

* **Time Series Forecasting:**
  + **Models: ARIMA (AutoRegressive Integrated Moving Average), Prophet (Facebook's forecasting tool), Exponential Smoothing, Recurrent Neural Networks (RNNs) like LSTMs (Long Short-Term Memory).**
  + **Application: Predict future call quality trends (e.g., average MOS score, Packet Loss) for the entire organization, specific sites, or user groups. Identify seasonality (e.g., quality drops during peak hours, or certain days of the week).**
* **Change Point Detection (Unsupervised):**
  + **Models: Ruptures, CUSUM (Cumulative Sum).**
  + **Application: Automatically detect significant, sudden shifts or anomalies in call quality metrics that indicate a new underlying problem (e.g., a network change, a faulty server, or a new software deployment impacting quality).**
* **Seasonal Decomposition:**
  + **Technique: Decompose a time series into trend, seasonal, and residual components to better understand underlying patterns.**

**Visualization:**

* **Line Graphs: Overlay multiple key metrics (e.g., RTT, Packet Loss, Jitter, MOS) over time, showing trends and fluctuations.**
* **Stacked Area Charts: Illustrate the proportion of different Intelligent Media Quality Classifiers over time.**
* **Forecasting Plots: Show historical data with predicted future values and confidence intervals.**
* **Anomaly Plots: Highlight detected change points or unusual events on a timeline.**

**7) Best Way to Visualize**

**Effective visualization is key to making ML insights actionable.**

**Key Principles:**

* **Interactive Dashboards: Allow users to filter, drill down, and explore data dynamically.**
* **Real-time/Near Real-time Updates: Essential for proactive issue detection.**
* **Clarity & Actionability: Visualizations should clearly convey insights and suggest immediate actions.**
* **Contextual Information: Provide context (e.g., user, device, network details) with each visualization.**

**Recommended Tools & Techniques:**

* **Business Intelligence (BI) Tools: Power BI, Tableau, Grafana (integrating with data warehouses/lakes).**
* **Web-based Dashboards: Custom-built using D3.js, React + charting libraries (e.g., Recharts, Chart.js) for highly customized and interactive experiences.**
* **Specific Chart Types:**
  + **Network: Heatmaps (subnet-to-subnet quality matrix), Geographic Maps (issues by location).**
  + **User/Device: Bar charts (top N problematic devices/users), Treemaps (proportion of issues by device type).**
  + **Trends: Line charts with forecasting, area charts showing issue breakdown over time.**
  + **Correlations: Scatter plots, bubble charts, parallel coordinate plots.**
  + **Alerts: Dashboards displaying active alerts, alert history, and frequency.**

**8) Generate Alerts & 9) Look for Repeated Alert Patterns**

**Problem: Proactive notification of critical issues and identifying recurring problems that require systematic solutions.**

**Machine Learning Models for Alerts:**

* **Threshold-Based Alerting (Rule-based):**
  + **Technique: Simple rules (e.g., "Alert if Packet Loss for any Subnet exceeds 5% for 15 minutes").**
  + **ML Enhancement: ML models can *learn* optimal thresholds rather than static ones.**
* **Anomaly Detection (Unsupervised):**
  + **Models: Isolation Forest, One-Class SVM, Z-score analysis, Moving Average with Standard Deviation.**
  + **Application: Trigger alerts when ML models detect statistically significant deviations from normal behavior for any key call quality metric or group of metrics.**
* **Predictive Alerting:**
  + **Models: Time Series Forecasting, Classification (e.g., "predict likelihood of poor call within next hour").**
  + **Application: Generate alerts *before* an issue fully manifests, based on predicted degradation.**

**Machine Learning Models for Repeated Alert Patterns:**

* **Sequence Mining:**
  + **Models: PrefixSpan, GSP (Generalized Sequential Pattern mining).**
  + **Application: Discover frequently occurring sequences or patterns of alerts (e.g., "Alert: High CPU on Device X is always followed by Alert: Video Freezes"). This helps identify root causes of cascading failures.**
* **Clustering (Unsupervised):**
  + **Models: DBSCAN, K-Means.**
  + **Application: Group similar alerts together (e.g., alerts related to the same device type, or same network segment) to reduce noise and identify underlying common problems.**
* **Association Rule Mining:**
  + **Models: Apriori.**
  + **Application: Find associations between different types of alerts or alerts and other contextual data (e.g., "Alert: Poor Microphone is often accompanied by Old Driver Version").**

**Alerting Mechanisms:**

* **Integrations: Microsoft Teams notifications, email, SMS, integration with ITSM (IT Service Management) tools (e.g., ServiceNow) via webhooks.**
* **Prioritization: Assign severity levels to alerts based on ML confidence scores or impact assessment.**

**Visualization of Alerts:**

* **Dedicated Alert Dashboard: List active and historical alerts with severity, timestamps, and contributing factors.**
* **Frequency Graphs: Show the number of alerts by type, time, or location.**
* **Correlation Matrix/Network Graph: Visualize relationships between different alert types.**

**Conclusion & Next Steps**

**Summary of Benefits:**

* **Proactive Issue Resolution: Identify and address problems before they significantly impact users.**
* **Optimized Resource Allocation: Focus troubleshooting efforts on the most impactful issues.**
* **Improved User Experience: Ensure consistent, high-quality communication for all Teams users.**
* **Data-Driven Decision Making: Leverage insights for network upgrades, device standardization, and policy adjustments.**

**Key Considerations for Implementation:**

* **Data Quality: Ensure clean, consistent, and complete data from CQD and Call Analytics.**
* **Feature Engineering: Expert knowledge is crucial to create effective features for ML models.**
* **Model Monitoring: Continuously monitor model performance and retrain as data patterns evolve.**
* **Iterative Improvement: Start small, demonstrate value, and expand ML capabilities over time.**