Expedition ML Hack

Expedition is no longer supported and new PANOS versions are not recognised in ML tasks

This is a workaround to avoid these errors and it’s NOT a fix

It tricks expedition to think the csv data belongs to older supported versions, it removes columns added on newer versions

1. Install XSV. we use this to trim large csv files. It’s a Rust binary, much faster than awk or other tools, export the attached binary to path below and verify its path and version

```
# which xsv
/usr/local/bin/xsv

# xsv --version
0.13.0
```

2. Install Parallels so that XSV can process files in parallel

```
sudo apt-get update
sudo apt-get install -y parallel
```

3. create trimmer.sh file in your temp data structure folder, update paths accordingly in script below

```
#!/bin/bash

# Usage: ./trim_all_csvs_parallel.sh /path/to/dir
CSV_DIR="$1"
DROP_COUNT=8
EXPECTED_COLS=123
LOG_FILE="/ExpeditionLogs/TempData/trimmer.log"
```

```
if [[ -z "$CSV_DIR" || ! -d "$CSV_DIR" ]]; then
    echo "$(date '+%F %T') [ERROR] Directory '$CSV_DIR' not found." >
> "$LOG_FILE"
    exit 1
fi

# Function to process each CSV file
process_file() {
    FILE="$1"
    DROP_COUNT="$2"
    EXPECTED_COLS="$3"
    LOG_FILE="$4"

    {
        echo "$(date '+%F %T') [INFO] Processing: $FILE"

        # Skip if file is in use
        if lsof "$FILE" &>/dev/null; then
            echo "$(date '+%F %T') [SKIP] File is in use: $FILE"
            exit 0
        fi

        # Count number of columns
        TOTAL_COLS=$(/usr/local/bin/xsv headers "$FILE" | wc -l)

        # Skip if column count doesn't match expected
        if [[ "$TOTAL_COLS" -ne "$EXPECTED_COLS" ]]; then
            echo "$(date '+%F %T') [SKIP] Column count $TOTAL_COLS !=
$EXPECTED_COLS: $FILE"
            exit 0
        fi

        # Determine which columns to keep (1-based)
        KEEP_COUNT=$((TOTAL_COLS - DROP_COUNT))
        COLS=$(seq -s, 1 "$KEEP_COUNT")

        # Create temp output file
        TMP_FILE="$(dirname "$FILE")/.$(basename "$FILE").trimmed"
```

```
        # Run /usr/local/bin/xsv to select columns and overwrite original fil
e
        if /usr/local/bin/xsv select "$COLS" "$FILE" > "$TMP_FILE"; then
            mv "$TMP_FILE" "$FILE"
            chown www-data:www-data "$FILE"
            echo "$(date '+%F %T') [DONE] Trimmed last $DROP_COUNT c
olumns and set ownership to www-data: $FILE"
        else
            echo "$(date '+%F %T') [FAIL] Failed to process: $FILE"
            rm -f "$TMP_FILE"
        fi
    } >> "$LOG_FILE" 2>&1
}

# Export function for GNU parallel
export -f process_file

# Process all CSV files in the directory using 6 parallel jobs
find "$CSV_DIR" -maxdepth 1 -name '*.csv' -print0 |
    parallel -0 -j6 process_file {} "$DROP_COUNT" "$EXPECTED_COLS"
"$LOG_FILE"
```

4. Make the file executable

```
chmod +x trimmer.sh
```

5. Test run with below command

```
./trimmer.sh $PATHtoYourCSVFolder
eg
./trimmer.sh /ExpeditionLogs/TempData
```

6. Add a cronjob to run before your daily ML schedule, make sure to leave enough time in between. XSV may take a couple hours for large files Update the paths accordingly

```
00 07 * * * /bin/bash /ExpeditionLogs/TempData/trimmer.sh /Expedition Logs/TempData >> /ExpeditionLogs/TempData/cron.log 2>&1
```

7. Monitor log file trimmer.log for errors on same directory
8. If all goes well, Expedition should now recognize the csv files and process them.