

Validation Unveiled: Striking the Balance for Data Integrity and Compliance in R, Python, and SAS

Sy Truong, Meta-Xceed, Inc.

Agenda

Introduction

Risk-Based Validation

User Acceptance Testing

Performance Qualification Testing

AI Approach Summary



Introduction

- Traditional SAS Validation Process
- AI-Enhanced Techniques for R and Python System Validation

Systems Validation Methodologies

- Python/R Risk-Based Analysis: Less formal validation compared to SAS.
- User Acceptance: Effective execution of user-representative scripts, akin to SAS OQ testing.
- Performance Qualification: Testing performance with parallel execution of Python, R, and SAS code.
- AI-Generated Test Scripts: ChatGPT generates useful tests for Python and R while protecting data.

Risk Based Validation

Traditional SAS
vs
AI Enhance for R and Python

Traditional Risk Assessment vs AI

- STEP 1: Conduct meetings with SAS users to discuss data integrity and accuracy risks during analysis.
 - STEP 2: Document risk factors using Excel or MS Word to organize insights from interviews.
 - STEP 3: Assign risk scores to tasks based on predefined rules, utilizing numerical or grading systems.
 - STEP 4: Consolidate scores to evaluate overall risk level, providing insights into system validation needs.
- STEP 1: Direct interview-style questions to AI models, specifying user roles and scenarios.
 - STEP 2: Summarize interview questions and answers in CSV format for easy analysis.
 - STEP 3: Combine established business rules with AI-generated scores for accurate risk assessment.
 - STEP 4: Use AI to summarize risk analysis, prompting for a summary based on provided data.

AI Risk Assessment STEP 1

- *Prompt: As a R programmer involved in clinical trials, when performing survival analysis using Kaplan-Meier, list the top 10 risks I might face when using the system.*

ChatGPT

When working with survival analysis using R's Kaplan-Meier method in the pharmaceutical industry, here are ten common risks you might encounter:

- **Censoring Issues:** Incomplete follow-up or data might lead to censoring, impacting the accuracy of survival estimates.
- **Data Quality:** Ensuring the accuracy and reliability of data inputs is crucial; any errors could significantly affect results.
- **Assumptions Violation:** The assumptions of Kaplan-Meier analysis (such as independent censoring) may be violated, affecting the reliability of results.
- **Small Sample Sizes:** Limited data could result in imprecise estimates, affecting the robustness of your conclusions.
- **Time-Dependent Covariates:** Handling variables that change over time requires careful consideration and appropriate methodologies.
- **Competing Risks:** Failure to account for multiple causes of failure could lead to biased estimates of survival probabilities.

AI Risk Assessment STEP 1

- *Prompt: As a R programmer involved in clinical trials, when performing survival analysis using Kaplan-Meier, list the top 10 risks I might face when using the system.*
 - **Confounding Factors:** Uncontrolled confounding variables might impact the observed association between survival and covariates.
 - **Model Complexity:** Overfitting models or choosing overly complex models might result in poor generalization to new data.
 - **Model Validation:** Failing to validate the model appropriately could lead to misleading conclusions about the predictive performance.
 - **Interpretation Challenges:** Communicating complex survival analysis results to non-statistical stakeholders might pose difficulties.

To mitigate these risks, it's vital to thoroughly understand the underlying assumptions of the Kaplan-Meier method, validate models rigorously, perform sensitivity analyses, and collaborate closely with domain experts for context-specific interpretations.

AI Risk Assessment STEP 2

Prompt: Please provide the above response in CSV format.



ChatGPT



Certainly! Here's the information captured in a CSV (Comma-Separated Values) format:

CSV

Copy code

Risk,Description

Censoring Issues,Incomplete follow-up or data might lead to cer

Data Quality,Ensuring the accuracy and reliability of data input

Assumptions Violation,The assumptions of Kaplan-Meier analysis

Small Sample Sizes,Limited data could result in imprecise estim

Time-Dependent Covariates,Handling variables that change over t

Competing Risks,Failure to account for multiple causes of failu

Confounding Factors,Uncontrolled confounding variables might in

Model Complexity,Overfitting models or choosing overly complex

AI Risk Assessment STEP 3

- *Prompt: What is the risk score, ranging from 0 to 100, for conducting survival analysis on clinical data using R (or specific risks mentioned)?*



ChatGPT



Assigning numerical risk scores between 0 and 100 for the listed risks involves subjective assessment and would depend on various factors such as context, data specifics, and expert judgment. Here's a sample subjective risk score representation:

```
CSV Copy code

Risk,Description,Risk Score (0-100)
Censoring Issues,Incomplete follow-up or data might lead to censoring, i
Data Quality,Ensuring the accuracy and reliability of data inputs is cru
Assumptions Violation,The assumptions of Kaplan-Meier analysis (such as
Small Sample Sizes,Limited data could result in imprecise estimates, aff
Time-Dependent Covariates,Handling variables that change over time requi
Competing Risks,Failure to account for multiple causes of failure could
Confounding Factors,Uncontrolled confounding variables might impact the
```

AI Risk Assessment STEP 4

Prompt: Write a summary of risk assessment based upon the following risk analysis: [paste csv]

ChatGPT

The risk assessment based on the provided analysis of various factors in survival analysis using Kaplan-Meier methodology is as follows:

- **Data Quality (80):** Ensuring the accuracy and reliability of data inputs is identified as the highest-risk factor. Errors in data could significantly impact the outcomes of survival analysis.
- **Confounding Factors (80):** Uncontrolled variables affecting the association between survival and covariates pose a considerable risk. Proper control and understanding of these factors are crucial.
- **Assumptions Violation (75):** Violation of key assumptions in Kaplan-Meier analysis, such as independent censoring, could affect the reliability of the results and pose a significant risk.
- **Competing Risks (75):** Failing to account for multiple causes of failure may lead to biased survival probability estimates, posing a substantial risk to the accuracy of the analysis.
- **Model Validation (75):** Inadequate validation of the model might lead to misleading conclusions about predictive performance, posing a significant risk to the analysis's reliability.
- **Interpretation Challenges (70):** Communicating complex survival analysis results to non-statistical stakeholders is identified as a risk due to potential difficulties in conveying the findings accurately.

AI Risk Assessment STEP 4

Prompt: Write a summary of risk assessment based upon the following risk analysis: [paste csv]

- **Censoring Issues (70):** Incomplete follow-up or data leading to censoring poses a moderate risk, impacting the accuracy of survival estimates.
- **Time-Dependent Covariates (70):** Handling variables changing over time requires careful consideration, posing a moderate risk to the analysis.
- **Model Complexity (65):** Overfitting or selecting overly complex models poses a moderate risk by potentially hindering generalization to new data.
- **Small Sample Sizes (60):** Limited data, while a risk, is ranked lower as it may lead to imprecise estimates but is relatively less impactful compared to other factors.

This summary ranks risks based on their potential impact on the accuracy, reliability, and interpretability of survival analysis using Kaplan-Meier methodology in the pharmaceutical context. Prioritizing attention to high-risk factors, such as data quality and confounding variables, is crucial for robust and reliable analysis outcomes.

User Acceptance Testing

Traditional SAS
vs
AI Enhance for R and Python

Traditional User Acceptance Test vs AI

- STEP 1: Capture User Requirements: Interview users to define necessary outputs and functionalities, then document in a detailed MS Word file.
 - STEP 2: Acquire or Create Test Scripts: Identify or develop scripts aligned with requirements, adjusting as needed for validation.
 - STEP 3: Execute Scripts: Manually run scripts or automate sequential execution to streamline validation.
 - STEP 4: Summarize Test Results: Review logs for issues, verify alignment with requirements, and document findings in Excel before finalizing in MS Word.
- STEP 1: Begin by prefacing prompts with user roles and describe the need for capturing requirements.
 - STEP 2: Use AI automation to create specific test scripts tailored to address gaps in existing scripts.
 - STEP 3: Streamline the process by automating the sequential execution of multiple test scripts and capturing outputs.
 - STEP 4: Incorporate AI-generated summaries into the test summary report to aid in drawing conclusions efficiently.

AI User Acceptance Testing STEP 1

Prompt: As a Biostatistician skilled in Python, identify the top 10 user requirements for a statistical computing environment that uses Python.

ChatGPT

Certainly! Here are the top 10 user requirements for a statistical computing environment utilizing Python for a Python Programmer analyzing clinical data:

- **Data Import and Preprocessing:** Efficient tools to import diverse clinical datasets (CSV, Excel, etc.) and preprocess them for analysis, including data cleaning and transformation.
- **Statistical Analysis Libraries:** Access to comprehensive statistical libraries (e.g., Pandas, NumPy, SciPy) to perform various analyses like regression, hypothesis testing, and survival analysis.
- **Visualization Capabilities:** Robust visualization modules (Matplotlib, Seaborn) to generate clear and insightful graphs and plots for data exploration and presentation.
- **Machine Learning Integration:** Seamless integration with machine learning libraries (Scikit-learn, TensorFlow) for predictive modeling and advanced analysis, such as predictive modeling for patient outcomes.
- **Reproducibility and Version Control:** Tools to ensure reproducibility of analyses and the ability to maintain version control for scripts and datasets to track changes.
- **Documentation and Reporting:** Features facilitating easy documentation of analysis steps, allowing for reproducibility, and creating comprehensive reports for regulatory compliance.

AI User Acceptance Testing STEP 1

Prompt: As a Biostatistician skilled in Python, identify the top 10 user requirements for a statistical computing environment that uses Python.

- **Compatibility and Interoperability:** Compatibility with various clinical data formats and the ability to interface with other tools or systems commonly used in clinical research.
- **Security and Data Privacy:** Robust security measures ensuring compliance with data privacy regulations (HIPAA, GDPR) when handling sensitive patient information.
- **Performance and Scalability:** Capability to handle large-scale clinical datasets efficiently and effectively, ensuring performance and scalability.
- **Community Support and Resources:** Access to a strong community and ample resources (forums, documentation, tutorials) for troubleshooting, learning, and staying updated with advancements in Python for clinical data analysis.

These requirements address the essentials needed for a Python-based statistical computing environment tailored for analyzing clinical data.

AI User Acceptance Testing STEP 2

Prompt: As a Python programmer analyzing oncology clinical trial studies, could you provide an example Python script using the Pandas library?

ChatGPT

Absolutely! Pandas is a powerful library for data manipulation and analysis in Python, often used in data science and analytics tasks. Here's a simple example demonstrating how you might use Pandas to perform basic analysis on oncology clinical trial data:

Let's assume you have a CSV file named `clinical_trials.csv` containing oncology clinical trial data with columns like `TrialID`, `PatientID`, `Treatment`, `Outcome`, etc. Here's an example script:

```
python Copy code  
  
import pandas as pd  
  
# Load the clinical trial data from a CSV file  
clinical_trials_data = pd.read_csv('clinical_trials.csv')  
  
# Display basic information about the dataset  
print("Summary of the clinical trials data:")  
print(clinical_trials_data.info())
```

AI User Acceptance Testing STEP 2

Prompt: As a Python programmer analyzing oncology clinical trial studies, could you provide an example Python script using the Pandas library?

This script does the following:

Imports Pandas: import pandas as pd.

Loads Data: Reads the clinical trial data from a CSV file into a Pandas DataFrame using `pd.read_csv()`.

Basic Data Summary: Prints the data summary and the first few rows of the dataset using `info()` and `head()` functions.

Summary Statistics: Provides basic statistics (mean, standard deviation, etc.) of numerical columns using `describe()`.

Treatment Counts: Counts the occurrences of different treatments using `value_counts()`.

Filtering Data: Filters the data to show only patients with positive outcomes.

Replace 'clinical_trials.csv' with the correct file path and column names according to your dataset.

This is just a starting point; Pandas offers numerous functions for data manipulation, cleaning, analysis, and visualization. You can perform various operations, such as data aggregation, grouping, merging, and more, depending on the specific analysis or insights you aim to derive from the clinical trial data.

AI User Acceptance Testing STEP 3

Prompt: Please write a Python script that runs the following programs–v_test1.py, v_test2.py, v_test3.py–and captures their respective outputs.



ChatGPT



Certainly! Below is a Python script that runs three programs ('v_test1.py', 'v_test2.py', and 'v_test3.py') sequentially and captures their output:

```
python Copy code  
  
import subprocess  
  
# Define the programs to be run  
programs = ['v_test1.py', 'v_test2.py', 'v_test3.py']  
  
# Loop through each program and capture its output  
for program in programs:  
    print(f"Running {program}...")
```

AI User Acceptance Testing STEP 3

Prompt: Please write a Python script that runs the following programs—v_test1.py, v_test2.py, v_test3.py—and captures their respective outputs.

Replace `'v_test1.py'`, `'v_test2.py'`, and `'v_test3.py'` with the actual filenames of your programs. This script will execute each program sequentially and capture their respective outputs, displaying them in the console. Adjustments can be made to handle output formatting or error handling based on specific requirements.

AI User Acceptance Testing STEP 4

Prompt: Could you generate a summary of the test results based on the CSV list provided: [paste CSV text]?

- AI-Augmented Analysis: Integrating AI-generated summaries into test reports enhances efficiency.
- Initiating Test Summary: AI can quickly generate concise overviews based on provided data.
- Facilitating Conclusion Drawing: AI's role complements human analysis, expediting the summarization process.



Performance Qualification Testing

Traditional SAS
vs
AI Enhance for R and Python

Traditional Performance Qualification vs AI

- STEP 1: Define Performance Criteria: Determine the maximum concurrent users and acceptable speed decrease under load, adhering to business guidelines.
 - STEP 2: Select System-Intensive Test Scripts: Identify scripts for tasks like multivariate regression analysis from past user tests.
 - STEP 3: Execute and Measure Script Duration: Run scripts while tracking execution time to assess performance.
 - STEP 4: Organize Scripts for Simulated Users: Place scripts in separate folders representing user sessions to simulate varying loads.
 - STEP 5: Automate Script Execution: Develop a script to run multiple sessions concurrently and capture their durations.
 - STEP 6: Analyze Performance: Manually record and compare durations for smaller servers, automate analysis for larger servers to assess performance under varying user loads.
- STEP 1: Gain Performance Insights: Use an AI to enhance understanding.
 - STEP 2: Generate Additional Test Scripts: Create scripts for CPU and memory performance testing with AI.
 - STEP 3: Automate Script Execution and Timing: Develop an AI assisted Python scripts to run and time multiple test scripts concurrently, then calculate the average execution time for assessment.

AI Performance Qualification Testing STEP 1

- *Prompt: How would you recommend testing the performance of a server used to run Python scripts upon clinical trials data for five users.*

ChatGPT

To test the performance of a server used for running Python scripts on clinical trials data for multiple users, especially five users in this case, here's a suggested approach:

- **Define Test Scenarios:** Identify key scenarios that represent typical use cases for these Python scripts on clinical trials data. These scenarios should cover various types of operations or analyses that the users commonly perform.
- **Create Test Data:** Generate or acquire a representative dataset similar in size and complexity to the actual clinical trials data. This data will be used during testing to ensure the simulation closely mimics real-world usage.
- **Script Execution Management:** Copy the scripts into individual folders, each representing a separate user session. Ensure these scripts can be run independently and simultaneously.

AI Performance Qualification Testing STEP 1

- *Prompt: How would you recommend testing the performance of a server used to run Python scripts upon clinical trials data for five users.*
 - **Automate Execution:** Develop a script or use testing tools to automate the simultaneous execution of the Python scripts for the five user sessions. This will simulate concurrent users interacting with the system.
 - **Monitor Resource Usage:** During the script execution, closely monitor the server's resource usage (CPU, memory, disk I/O) to identify any potential bottlenecks or limitations. Tools like top (for Linux) or Task Manager (for Windows) can be used for this purpose.
 - **Measure Execution Time:** Capture and record the start and end times for each script execution for the five users. Calculate the elapsed time for each session.
 - **Analyze Performance:** Compare the performance metrics (elapsed time, CPU usage, memory consumption) against predefined success criteria. Evaluate if the server meets the defined criteria in terms of acceptable speed and resource utilization when five users are simultaneously running the Python scripts.
 - **Iterate and Optimize:** If the server doesn't meet the defined criteria, analyze the bottlenecks identified during testing. Optimize the scripts, consider parallel processing, or adjust server resources (if feasible) to enhance performance.
 - **Documentation and Reporting:** Document the testing process, results, any optimizations made, and recommendations for improvement. Provide a comprehensive report detailing the server's performance under the simulated load.

AI Performance Qualification Testing STEP 2

Prompt: Generate a Python program to test the performance on large data utilizing the CPU processing and memory of the server.



ChatGPT

Certainly! Here's an example Python script that generates a large dataset, performs some CPU-intensive operations, and measures the elapsed time:

```
python Copy code

import numpy as np
import time

# Define the size of the dataset (adjust according to your server's cap
data_size = 1000000 # For example, 1 million data points

# Generate a large dataset (random numbers for illustration purposes)
large_data = np.random.rand(data_size)

# Function to perform CPU-intensive operations on the data
def perform_operations(data):
    # Example: Squaring each element in the dataset
    squared_data = np.square(data)
    # Add more CPU-intensive operations here as needed

# Simulate memory usage by storing the result in a variable
result = squared_data # In a real scenario, this might involve mor
```

AI Performance Qualification Testing STEP 2

Prompt: Generate a Python program to test the performance on large data utilizing the CPU processing and memory of the server.

This script utilizes the NumPy library to generate a large dataset of random numbers and performs CPU-intensive operations (squaring each element in this example). You can modify the operations within the `perform_operations()` function to simulate different types of CPU-intensive tasks.

Please adjust the `data_size` variable according to the capacity of your server and monitor the script's execution, as large data sizes or highly intensive operations might consume substantial memory and CPU

resources. This script aims to stress the CPU and memory, providing an example of performance testing with Python.

AI Performance Qualification Testing STEP 3

- Prompt: Write a Python script to capture the elapsed time of how long it takes to run scripts: v_test1.py, v_test2.py, v_test3.py, v_test4.py and v_test5.py. Then calculate the average time to took to run these five program.

ChatGPT

Certainly! Here's an example of a Python script that measures the elapsed time for running five Python scripts (**v_test1.py** to **v_test5.py**) and calculates the average time taken:

```
python Copy code

import subprocess
import time

# Function to run a Python script and measure its execution time
def measure_execution_time(script_name):
    start_time = time.time()
    subprocess.run(["python", script_name]) # Replace "python" with th
    end_time = time.time()
    elapsed_time = end_time - start_time
    return elapsed_time
```

AI Performance Qualification Testing STEP 3

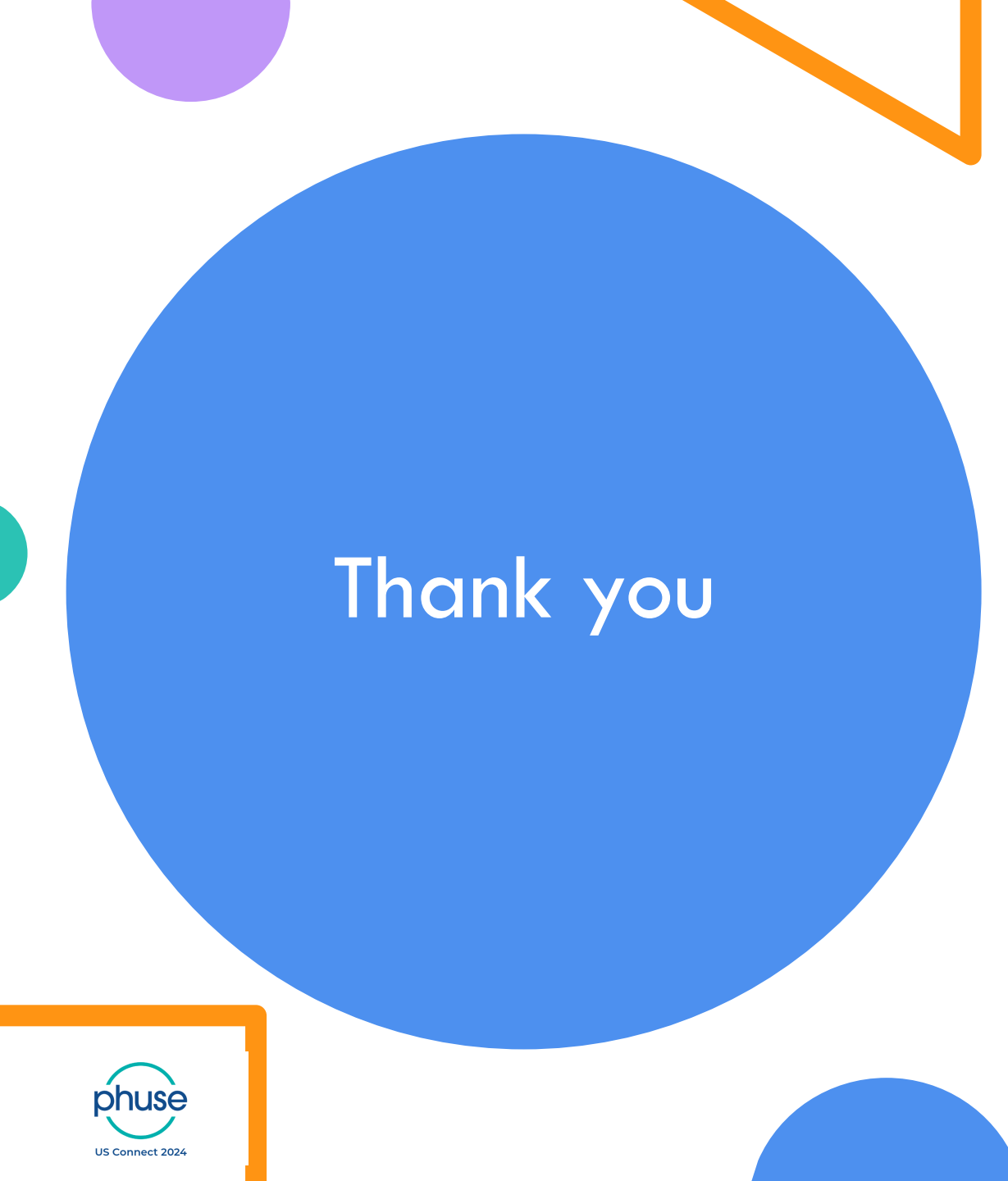
- *Prompt: Write a Python script to capture the elapsed time of how long it takes to run scripts: v_test1.py, v_test2.py, "v_test3.py, v_test4.py and v_test5.py. Then calculate the average time to took to run these five program.*

Replace "v_test1.py", "v_test2.py", "v_test3.py", "v_test4.py", and "v_test5.py" with the actual filenames of your Python scripts.

This script measures the elapsed time for running each script, stores the elapsed times in a list, and then calculates the average time taken for all five scripts. Adjust it as needed to match your specific environment or requirements.

AI Approach Summary

- Traditional Validation: Test script creation and processing are crucial for SAS and extend to R and Python systems.
- AI Integration: Leveraging platforms like GitHub, AI efficiently generates diverse functional scripts for validation.
- Automation Benefits: AI streamlines script generation and validation tasks, requiring human oversight for alignment with business needs.
- Future Prospects: Ongoing prompt engineering development and better AI models increased accuracy and efficiency, advancing AI's role in system validation.



Thank you

Sy Truong

510-979-9333

Sy.Truong@meta-x.com

www.meta-x.com