- Getting Started with cuML's accelerator mode

Using Accelerated Hardware

- Train a CNN to classify handwritten digits on the MNIST dataset using Flax NNX API
- Train a Vision Transformer (ViT) for image classification with JAX
- Text classification with a transformer language model using JAX

Featured examples

- Train a miniGPT language model with JAX AI Stack
- LoRA/QLoRA finetuning for LLM using Tunix
- Parameter-efficient fine-tuning of Gemma with LoRA and QLoRA
- Loading Hugging Face Transformers Checkpoints
- 8-bit Integer Quantization in Keras
- Float8 training and inference with a simple Transformer model
- Pretraining a Transformer from scratch with KerasHub
- Simple MNIST convnet
- Image classification from scratch using Keras 3
- Image Classification with KerasHub

```
import math
from scipy.stats import norm

def hypothesis_test_cv(mu, xbar, sigma, n, alpha=0.05, tail=1):
    # Standard error
    se = sigma / math.sqrt(n)

    # TWO-TAILED
    if tail == 1:
        zc = norm.ppf(1 - alpha/2)
        LCV = mu - zc * se
        UCV = mu + zc * se

        decision = (
            "Fail to Reject H0"
            if LCV <= xbar <= UCV
            else "Reject H0"
        )

        return {
            "Test": "Two-tailed",
            "z_critical": zc,
            "LCV": LCV,
            "UCV": UCV,
            "xbar": xbar,
            "\nDecision": decision
        }

    # RIGHT-TAILED
    elif tail == 2:
        zc = norm.ppf(1 - alpha)
        UCV = mu + zc * se

        decision = (
            "Fail to Reject H0"
            if xbar <= UCV
            else "Reject H0"
        )

        return {
            "Test": "Right-tailed",
            "z_critical": zc,
            "UCV": UCV,
            "xbar": xbar,
            "\nDecision": decision
        }

    # LEFT-TAILED
    elif tail == 3:
        zc = norm.ppf(1 - alpha)
        LCV = mu - zc * se

        decision = (
            "Fail to Reject H0"
            if xbar >= LCV
            else "Reject H0"
        )

        return {
            "Test": "Left-tailed",
            "z_critical": zc,
            "LCV": LCV,
            "xbar": xbar,
            "\nDecision": decision
        }

    else:
        raise ValueError("tail must be 1 (two), 2 (right), or 3 (left)")


if __name__ == "__main__":
    print("Hypothesis Testing - Critical Value Method\n")

    sigma = float(input("Enter population standard deviation (sigma): "))
    population_mean = float(input("Enter hypothesized population mean (μ₀): "))
    sample_mean = float(input("Enter sample mean (x̄): "))
    sample_size = int(input("Enter sample size (n): "))
    alpha = float(input("Enter significance level (alpha): "))

    print("\nSelect test type:")
    print("1. Two-tailed test (H₁: μ ≠ μ₀)")
    print("2. Right-tailed test (H₁: μ > μ₀)")
    print("3. Left-tailed test (H₁: μ < μ₀)")

    test_choice = int(input("Enter choice (1/2/3): "))

    print("\nResults: ")

    result = hypothesis_test_cv(population_mean, sample_mean, sigma, sample_size, alpha, test_choice)

    for k, v in result.items():
        print(f"{k}: {v}")

    print("\n")
```

```
Hypothesis Testing - Critical Value Method

Enter population standard deviation (sigma): 10
Enter hypothesized population mean (μ₀): 50
Enter sample mean (x̄): 54
Enter sample size (n): 36
Enter significance level (alpha): 0.05

Select test type:
1. Two-tailed test (H₁: μ ≠ μ₀)
2. Right-tailed test (H₁: μ > μ₀)
3. Left-tailed test (H₁: μ < μ₀)
Enter choice (1/2/3): 2

Results:
Test: Right-tailed
z_critical: 1.6448536269514722
UCV: 52.741422711585784
xbar: 54.0

Decision: Reject H0
```