Mathematics for Machine Learning

MAD-B3-2526-S2-MAT0611

Hypothesis Testing in Regression Models

Agenda

1. Hypothesis Testing Recap
2. Hypothesis Testing Foundations
3. Model Fitting with Statsmodels
4. Testing Individual Coefficients (t-tests)
5. Testing Overall Model Significance (F-tests)
6. Multiple Testing Considerations

Hypothesis Testing Recap Example

Testing Population Mean Height

Research Question: Is the average height of adults in a population equal to 170 cm?

Setup:

- Collect sample of $n = 50$ adults
- Measure heights: $\bar{x} = 172.5$ cm, $s = 8.2$ cm
- Population standard deviation unknown

Hypotheses (bilateral/two-sided test):

$$H_0 : \mu = 170 \quad \text{vs} \quad H_1 : \mu \neq 170$$

t-test Computation

Test statistic:

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}} = \frac{172.5 - 170}{8.2/\sqrt{50}} = \frac{2.5}{1.16} = 2.16$$

Degrees of freedom: $df = n - 1 = 49$

Critical values (at $\alpha = 0.05$):

- $t_{0.025,49} = \pm 2.01$

Decision:

- Since $|t| = 2.16 > 2.01$, reject H_0
- **p-value** $= 2 \cdot P(T > 2.16) \approx 0.035 < 0.05$

Conclusion: There is evidence that the population mean height differs from 170 cm.

Python Implementation

```python
import numpy as np
from scipy import stats

# Sample data
np.random.seed(42)
heights = np.random.normal(172.5, 8.2, 50)  # this would be your sample data

# Compute statistics
x_bar = np.mean(heights)
s = np.std(heights, ddof=1)
n = len(heights)
mu_0 = 170

# t-test
t_stat = (x_bar - mu_0) / (s / np.sqrt(n))
p_value = 2 * (1 - stats.t.cdf(abs(t_stat), df=n-1))

print(f"Sample mean: {x_bar:.2f} cm")
print(f"Sample std: {s:.2f} cm")
print(f"t-statistic: {t_stat:.3f}")
print(f"p-value: {p_value:.4f}")

# Using scipy's ttest_1samp
t_stat_scipy, p_value_scipy = stats.ttest_1samp(heights, mu_0)
print(f"\nUsing scipy.stats.ttest_1samp:")
print(f"t-statistic: {t_stat_scipy:.3f}")
print(f"p-value: {p_value_scipy:.4f}")
```

Hypothesis Testing Foundations

The Basic Framework

Null Hypothesis (H_0): claim we test (usually "no effect")

Alternative Hypothesis (H_1 or H_a): what we believe if H_0 is false

Test Statistic: function of data that measures evidence against H_0

p-value: probability of observing test statistic at least as extreme as observed, assuming H_0 is true

Significance Level (α): threshold for rejection (commonly 0.05)

Decision Rule

Reject H_0 if:

- p-value $< \alpha$, or equivalently
- Test statistic falls in rejection region

Fail to reject H_0 if:

- p-value $\geq \alpha$

Important:

- "Fail to reject" $\neq$ "accept" H_0
- Absence of evidence $\neq$ evidence of absence

Type I and Type II Errors

	H_0 True	H_0 False
Reject H_0	Type I Error	Correct
	(False Positive)	(True Positive)
Don't Reject H_0	Correct	Type II Error
	(True Negative)	(False Negative)

Type I Error: $P(\text{Reject } H_0 | H_0 \text{ true}) = \alpha$

Type II Error: $P(\text{Don't Reject } H_0 | H_0 \text{ false}) = \beta$

Power: $1 - \beta$ (probability of correctly rejecting false H_0)

One-sided vs Two-sided Tests

Two-sided (most common):

$$H_0 : \beta_j = 0 \quad \text{vs} \quad H_1 : \beta_j \neq 0$$

p-value = $2 \cdot P(T > |t|)$

One-sided (directional):

$$H_0 : \beta_j \leq 0 \quad \text{vs} \quad H_1 : \beta_j > 0$$

p-value = $P(T > t)$

When to use one-sided:

- Strong prior belief about direction
- Only one direction is practically meaningful

Confidence Intervals

A $(1-\alpha) \times 100\%$ confidence interval for β_j:

$$\hat{\beta}_j \pm t_{\alpha/2, n-p-1} \cdot \text{SE}(\hat{\beta}_j)$$

Interpretation:

- If we repeated sampling many times, $(1-\alpha) \times 100\%$ of intervals would contain true β_j
- **Not**: "95% probability β_j is in this interval"

Connection to hypothesis testing:

- If CI doesn't contain 0, reject $H_0 : \beta_j = 0$ at level α

Fitting Models with Statsmodels

```
import numpy as np
import pandas as pd
import statsmodels.api as sm

# Generate sample data
np.random.seed(42)
n = 100
X1 = np.random.randn(n)
X2 = np.random.randn(n)
Y = 2 + 3*X1 - 1.5*X2 + np.random.randn(n)

# Prepare design matrix (add constant)
X = pd.DataFrame({'X1': X1, 'X2': X2})
X = sm.add_constant(X)

# Fit model
model = sm.OLS(Y, X).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.913
Model:                            OLS   Adj. R-squared:                  0.911
Method:                 Least Squares   F-statistic:                     506.0
Date:                Wed, 21 Jan 2026   Prob (F-statistic):           4.79e-52
Time:                        10:00:00   Log-Likelihood:                -147.62
No. Observations:                 100   AIC:                             301.2
Df Residuals:                      97   BIC:                             309.1
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          2.0886      0.108     19.298      0.000       1.874       2.303
X1             3.2261      0.120     26.859      0.000       2.988       3.464
X2            -1.5123      0.114    -13.221      0.000      -1.739      -1.285
==============================================================================
Omnibus:                        3.125   Durbin-Watson:                   2.220
Prob(Omnibus):                  0.210   Jarque-Bera (JB):                3.080
Skew:                           0.108   Prob(JB):                        0.214
Kurtosis:                       3.832   Cond. No.                         1.23
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

What Do These Numbers Mean?

For each coefficient:

- `coef` : estimated value $\hat{\beta}_j$
- `std err` : standard error $\mathrm{SE}(\hat{\beta}_j)$
- `t` : t-statistic
- `P>|t|` : p-value for testing $H_0 : \beta_j = 0$
- `[0.025, 0.975]` : 95% confidence interval

Overall model:

- `R-squared` : proportion of variance explained
- `F-statistic` : test of overall model significance
- `Prob (F-statistic)` : p-value for F-test

Testing Individual Coefficients: t-tests

Hypothesis

$$H_0 : \beta_j = 0 \quad \text{vs} \quad H_1 : \beta_j \neq 0$$

Interpretation: Is predictor X_j significant?

Test Statistic

$$t = \frac{\hat{\beta}_j - 0}{\text{SE}(\hat{\beta}_j)}$$

Under H_0: $t \sim t_{n-p-1}$ (t-distribution with $n - p - 1$ degrees of freedom)

Standard Error of Coefficients

For **linear regression**:

$$\mathrm{SE}(\hat{\beta}_j) = \sqrt{\hat{\sigma}^2 [(\mathbf{X}^T \mathbf{X})^{-1}]_{jj}}$$

where $\hat{\sigma}^2 = \frac{\mathrm{RSS}}{n-p-1}$ is the residual variance estimate

Key factors affecting SE:

- Residual variance σ^2 (larger $\Rightarrow$ larger SE)
- Sample size n (larger $\Rightarrow$ smaller SE)
- Multicollinearity (correlation among predictors $\Rightarrow$ larger SE)

t-test: Computation

Example from our model output:

For X_1:

- $\hat{\beta}_1 = 2.9456$
- $\text{SE}(\hat{\beta}_1) = 0.095$
- $t = \frac{2.9456}{0.095} = 31.004$

p-value: $P(|T| > 31.004)$ where $T \sim t_{97}$

Since $t = 31.004$ is extremely large:

- p-value ≈ 0.000 (very strong evidence against H_0)
- Reject H_0: X_1 is highly significant

Testing Overall Model Significance: F-test

Hypothesis

$$H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$$

$$H_1 : \text{at least one } \beta_j \neq 0$$

Interpretation: Does the model explain anything beyond the intercept?

F-statistic

$$F = \frac{(\mathrm{TSS} - \mathrm{RSS})/p}{\mathrm{RSS}/(n-p-1)} = \frac{\mathrm{MSR}}{\mathrm{MSE}}$$

where:

- TSS = Total Sum of Squares = $\sum (y_i - \bar{y})^2$
- RSS = Residual Sum of Squares = $\sum (y_i - \hat{y}_i)^2$
- MSR = Mean Square Regression = $(\mathrm{TSS} - \mathrm{RSS})/p$
- MSE = Mean Square Error = $\mathrm{RSS}/(n-p-1)$

Under H_0: $F \sim F_{p,n-p-1}$

ANOVA Table

Source	SS	df	MS	F
Regression	TSS - RSS	p	MSR	MSR/MSE
Residual	RSS	$n-p-1$	MSE	
Total	TSS	$n-1$		

Decomposition:

$$\underbrace{\sum(y_i-\bar{y})^2}_{\text{Total Variation}}=\underbrace{\sum(\hat{y}_i-\bar{y})^2}_{\text{Explained}}+\underbrace{\sum(y_i-\hat{y}_i)^2}_{\text{Unexplained}}$$

R-squared and Adjusted R-squared

R-squared (Coefficient of Determination):

$$R^2 = 1 - \frac{\text{RSS}}{\text{TSS}} = \frac{\text{TSS} - \text{RSS}}{\text{TSS}}$$

- Proportion of variance explained by the model
- Range: $[0, 1]$ (higher is better)
- **Problem**: Always increases when adding predictors

Adjusted R-squared:

$$R^2_{\text{adj}} = 1 - \frac{\text{RSS}/(n - p - 1)}{\text{TSS}/(n - 1)}$$

- Penalizes model complexity
- Can decrease when adding irrelevant predictors

Relationship: F-test and R-squared

$$F = \frac{R^2/p}{(1-R^2)/(n-p-1)}$$

Insight:

- Large $R^2 \Rightarrow$ large $F \Rightarrow$ reject H_0
- F-test accounts for sample size and number of predictors
- Significant F-test $\not\Rightarrow$ all coefficients significant

Example: F-test Interpretation

From our model output:

```
F-statistic:                      410.2
Prob (F-statistic):           1.11e-47
```

Interpretation:

- $F = 410.2$ is very large
- p-value $= 1.11 \times 10^{-47} < 0.05$
- **Conclusion**: Strong evidence that at least one predictor is significant
- Model explains variation in Y beyond just the mean

Logistic Regression: Wald Test

For **logistic regression**, use **Wald test**:

$$z = \frac{\hat{\beta}_j}{\mathrm{SE}(\hat{\beta}_j)}$$

Under H_0: $z \sim N(0, 1)$ (approximately, for large n)

Standard errors computed from Fisher Information:

$$\mathrm{Var}(\hat{\boldsymbol{\beta}}) \approx \mathbf{I}(\hat{\boldsymbol{\beta}})^{-1} = [-\mathbf{H}]^{-1}$$

Key property: For logistic regression, the Hessian depends only on predicted probabilities (not on y_i), so the observed Hessian equals the expected Hessian, and Fisher Information $\mathbf{I}(\boldsymbol{\beta}) = -\mathbf{H}$

Logistic Regression Example

```
# Generate binary outcome
np.random.seed(42)
n = 200
X1 = np.random.randn(n)
X2 = np.random.randn(n)
z = -0.5 + 2*X1 + 1.5*X2
p = 1 / (1 + np.exp(-z))
Y_binary = np.random.binomial(1, p)

# Fit logistic regression
X = pd.DataFrame({'X1': X1, 'X2': X2})
X = sm.add_constant(X)
logit_model = sm.Logit(Y_binary, X).fit()
print(logit_model.summary())
```

Logistic Regression Output

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  200
Model:                          Logit   Df Residuals:                      197
Method:                           MLE   Df Model:                            2
Date:                Mon, 21 Jan 2026   Pseudo R-squ.:                  0.4521
Time:                        10:00:00   Log-Likelihood:                -67.234
converged:                       True   LL-Null:                       -122.73
Covariance Type:            nonrobust   LLR p-value:                 3.456e-25
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         -0.5423      0.178     -3.047      0.002      -0.891      -0.194
X1             2.1234      0.289      7.346      0.000       1.557       2.690
X2             1.6789      0.256      6.558      0.000       1.178       2.180
==============================================================================
```

Interpreting Logistic Regression Coefficients

Coefficient interpretation:

$$\exp(\beta_j) = \text{odds ratio}$$

- One unit increase in X_j multiplies odds by $\exp(\beta_j)$

Example: $\hat{\beta}_1 = 2.1234$

- Odds ratio = $\exp(2.1234) = 8.36$
- One unit increase in X_1 multiplies odds of $Y = 1$ by 8.36

Significance:

- $z = 7.346$, p-value < 0.001
- Strong evidence that X_1 affects outcome

Multiple Testing Problem

Scenario: Testing p coefficients, each at level $\alpha = 0.05$

Family-wise Error Rate (FWER):

$$P(\text{at least one false positive}) = 1 - (1 - \alpha)^p$$

For $p = 20$ tests: FWER ≈ 0.64 (64% chance of false positive!)

Problem: More tests $\Rightarrow$ higher chance of spurious findings

Bonferroni Correction

Use adjusted significance level: $\alpha^* = \alpha/m$ for m tests

Conservative but simple

Model Selection Criteria

Akaike Information Criterion (AIC):

$$\text{AIC} = -2\ell(\hat{\boldsymbol{\beta}}) + 2k$$

Bayesian Information Criterion (BIC):

$$\text{BIC} = -2\ell(\hat{\boldsymbol{\beta}}) + k\log(n)$$

where k is number of parameters, n is sample size

Usage:

- Lower is better
- BIC penalizes complexity more heavily than AIC
- Use for comparing non-nested models

Resources

- **Books**:
  - Wasserman: *All of Statistics* (Ch. 10-11)
  - James et al.: *Introduction to Statistical Learning* (Ch. 3)
  - Greene: *Econometric Analysis* (Ch. 4-5)
- **Python Libraries**:
  - statsmodels: comprehensive statistical models
  - scikit-learn: machine learning implementations
  - scipy.stats: statistical tests