

LOGISTIC REGRESSION: BINARY

CLASSIFICATION

I. OBJECTIVE

Upon successful completion of this laboratory exercise, the student shall be able to:

1. Explain the theoretical foundations of logistic regression, including the logistic (sigmoid) function, odds ratios, and maximum likelihood estimation.

2. Interpret logistic regression coefficients in terms of odds ratios and their implications for business decision-making.

3. Implement logistic regression models using Python and the scikit-learn library to classify binary outcomes in business contexts.

4. Calculate predicted probabilities for new observations and apply appropriate decision thresholds based on business requirements.

5. Evaluate model performance using confusion matrices, accuracy, precision, recall, F1-score, and the ROC-AUC metric.

6. Apply logistic regression to practical business problems including credit scoring, customer churn prediction, and marketing response modeling.

II. APPARATUS AND MATERIALS

The following items are required for the proper execution of this laboratory exercise:

1. Personal computer with Python 3.8 or later installed

2. Required Python libraries:

- scikit-learn (sklearn) version 1.0 or later
- numpy version 1.20 or later
- matplotlib version 3.4 or later
- pandas version 1.3 or later

3. Integrated Development Environment (IDE) such as:

- Jupyter Notebook (recommended)
- Spyder
- Visual Studio Code with Python extension

4. Scientific calculator (for verification computations)

5. Laboratory notebook or report template

6. This laboratory manual

III. THEORETICAL BACKGROUND

A. Introduction and Motivation

Logistic regression is a statistical method for modeling the probability of a binary outcome. Unlike linear regression, which predicts continuous values, logistic regression predicts the probability that an observation belongs to a particular class.

The method was developed by statistician David Cox in 1958 and has since become one of the most widely used techniques in business analytics, medical research, and social sciences.

The primary advantage of logistic regression over simpler classifiers (such as the perceptron) is its ability to output calibrated probabilities rather than mere class labels, enabling more nuanced business decisions.

B. The Linear Predictor

Logistic regression begins with a linear combination of input features, termed the linear predictor or log-odds:

$$x = \sum_{i=1}^n \beta_i x_i + \beta_0$$

where:

z = linear predictor (log-odds)

β_i = coefficient for feature i

x_i = value of feature i

β_0 = intercept term

n = number of features

C. The Logistic (Sigmoid) Function

The linear predictor z can range from negative infinity to positive infinity. To convert this to a probability bounded between 0 and 1, we apply the logistic (sigmoid) function:

$$p(y = 1|x) = \frac{1}{1 + e^{-z}}$$

Alternatively expressed as:

$$p(y = 1|x) = \frac{e^z}{1 + e^z}$$

Properties of the sigmoid function:

- When $z = 0$, $p = 0.5$

- When z approaches +infinity, p approaches 1
- When z approaches -infinity, p approaches 0
- The function is symmetric about the point (0, 0.5)

D. Odds and Log-Odds

The odds of an event is defined as the ratio of the probability of the event occurring to the probability of it not occurring:

$$\text{Odds} = \frac{p}{1-p}$$

The log-odds (logit) is the natural logarithm of the odds:

$$\text{logit}(p) = \ln(\text{Odds}) = \ln\left(\frac{p}{1-p}\right) = z$$

E. Interpretation of Coefficients

Each coefficient β_i represents the change in log-odds for a one-unit increase in x_i , holding all other variables constant.

The odds ratio for feature i is computed as:

$$OR_i = e^{\beta_i}$$

Interpretation:

- If $OR > 1$: increasing x_i increases the odds of $y=1$
- If $OR < 1$: increasing x_i decreases the odds of $y=1$
- If $OR = 1$: x_i has no effect on the odds

Example: If $\beta_1 = 0.693$, then $OR_1 = e^{0.693} = 2.0$, meaning

a one-unit increase in x_1 doubles the odds of $y=1$.

F. Maximum Likelihood Estimation

Unlike linear regression which uses least squares, logistic regression estimates coefficients by maximizing the likelihood function. The log-likelihood for n observations is:

$$LL = \text{SUM}[y_i * \ln(p_i) + (1-y_i) * \ln(1-p_i)]$$

where y_i is the actual class (0 or 1) and p_i is the predicted probability for observation i .

Scikit-learn uses iterative optimization algorithms (such as L-BFGS or Newton-CG) to find coefficient values that maximize this log-likelihood.

G. Decision Boundary

The decision boundary for logistic regression occurs where

$p(y=1|x) = 0.5$, which corresponds to $z = 0$:

$$\beta_0 + \beta_1*x_1 + \beta_2*x_2 = 0$$

For two features, solving for x_2 :

$$x_2 = -(\beta_1/\beta_2)*x_1 - (\beta_0/\beta_2)$$

H. Regularization

Scikit-learn's LogisticRegression applies L2 regularization by default, controlled by the parameter C (inverse regularization strength):

- Large C: weak regularization, may overfit
- Small C: strong regularization, may underfit

I. Scikit-learn Implementation

Key parameters of LogisticRegression:

C : inverse regularization strength (default=1.0)
solver : optimization algorithm
max_iter : maximum iterations for convergence
random_state : seed for reproducibility

Key attributes after fitting:

coef_ : coefficient vector (β_1, β_2, \dots)
intercept_ : intercept term (β_0)
classes_ : class labels

IV. PROCEDURE

PART A: ENVIRONMENT SETUP AND LIBRARY IMPORT

Step 1. Open your Python IDE (Jupyter Notebook recommended) and
create a new notebook file named "Lab8_LogisticRegression.ipynb"

Step 2. Import the required libraries by executing the following code:

```
# Import required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import roc_curve, roc_auc_score
import warnings
warnings.filterwarnings('ignore')
```

Step 3. Verify installation by checking library versions:

```
print("NumPy version:", np.__version__)
print("Pandas version:", pd.__version__)
import sklearn
```

```
print("Scikit-learn version:", sklearn.__version__)
```

PART B: MANUAL COMPUTATION EXERCISE

Step 4. Consider the following scenario for a loan approval model.

A logistic regression has been trained with the following parameters:

β_0 (intercept) = -3.5

β_1 (income coefficient) = 2.0

β_2 (credit score coefficient) = 1.5

A loan applicant has:

x_1 (normalized income) = 0.8

x_2 (normalized credit score) = 0.7

Step 5. Compute the linear predictor (log-odds):

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

$$z = -3.5 + (2.0)(0.8) + (1.5)(0.7)$$

$$z = -3.5 + 1.6 + 1.05$$

$$z = -0.85$$

Step 6. Compute the predicted probability using the sigmoid function:

$$p = 1 / (1 + e^{-z})$$

$$p = 1 / (1 + e^{-(-0.85)})$$

$$p = 1 / (1 + e^{(0.85)})$$

$$p = 1 / (1 + 2.3396)$$

$$p = 1 / 3.3396$$

$$p = 0.2994$$

Step 7. Make the classification decision:

If using threshold = 0.5:

Since $p = 0.2994 < 0.5$, predict class 0 (Reject loan)

Step 8. Calculate the odds:

$$\text{Odds} = p / (1 - p)$$

$$\text{Odds} = 0.2994 / (1 - 0.2994)$$

$$\text{Odds} = 0.2994 / 0.7006$$

$$\text{Odds} = 0.4274$$

Interpretation: The odds of approval are 0.43 to 1 (or about 1 to 2.3), meaning rejection is more than twice as likely.

Step 9. Calculate odds ratios for interpretation:

$$\text{OR}_1 = e^{(\beta_1)} = e^{(2.0)} = 7.389$$

$$\text{OR}_2 = e^{(\beta_2)} = e^{(1.5)} = 4.482$$

Interpretation:

- A 1-unit increase in normalized income multiplies the odds of approval by 7.39
- A 1-unit increase in credit score multiplies the odds of approval by 4.48

Step 10. Complete the following table for different applicants:

[TABLE 1: Student Work Area - Manual Probability Calculations]

Insert table with 5 rows and 7 columns

Headers: Applicant | x_1 | x_2 | z | p | Odds | Decision (threshold=0.5)

Row 1: A | 0.8 | 0.7 | -0.85 | 0.2994 | 0.4274 | Reject

Row 2: B | 0.9 | 0.9 | [blank] | [blank] | [blank] | [blank]

Row 3: C | 0.5 | 0.5 | [blank] | [blank] | [blank] | [blank]

Row 4: D | 0.6 | 0.8 | [blank] | [blank] | [blank] | [blank]

PART C: PYTHON IMPLEMENTATION WITH SCIKIT-LEARN

Step 11. Create the training data using NumPy arrays:

```
# Define training data for credit approval
X_train = np.array([
    [0.9, 0.8], # Applicant 1: High income, good credit
    [0.3, 0.2], # Applicant 2: Low income, poor credit
```

```
[0.7, 0.7], # Applicant 3: Good income, good credit  
[0.2, 0.3], # Applicant 4: Low income, low credit  
[0.8, 0.6], # Applicant 5: High income, fair credit  
[0.4, 0.4], # Applicant 6: Fair income, fair credit  
[0.6, 0.9], # Applicant 7: Fair income, excellent credit  
[0.1, 0.5] # Applicant 8: Very low income, fair credit  
])
```

```
y_train = np.array([1, 0, 1, 0, 1, 0, 1, 0])
```

```
print("Feature matrix shape:", X_train.shape)  
print("Target vector shape:", y_train.shape)  
print("\nClass distribution:")  
print(" Class 0 (Rejected):", np.sum(y_train == 0))  
print(" Class 1 (Approved):", np.sum(y_train == 1))
```

Step 12. Create and configure the Logistic Regression model:

```
# Initialize logistic regression with specified parameters  
log_reg = LogisticRegression(  
    C=1.0, # Regularization strength (inverse)  
    solver='lbfgs', # Optimization algorithm  
    max_iter=1000, # Maximum iterations  
    random_state=42 # For reproducibility  
)
```

Step 13. Train the logistic regression model:

```
# Fit the model to training data  
log_reg.fit(X_train, y_train)  
  
# Display learned parameters  
print("Learned coefficients:")  
print(" beta_1 (income):", log_reg.coef_[0][0])  
print(" beta_2 (credit):", log_reg.coef_[0][1])  
print(" beta_0 (intercept):", log_reg.intercept_[0])
```

Step 14. Calculate and display odds ratios:

```
# Compute odds ratios  
odds_ratios = np.exp(log_reg.coef_[0])  
  
print("\nOdds Ratios:")  
print(" OR for income (x_1):", odds_ratios[0])  
print(" OR for credit (x_2):", odds_ratios[1])  
  
# Interpretation  
print("\nInterpretation:")  
print(f" A 1-unit increase in income multiplies odds by {odds_ratios[0]:.3f}")  
print(f" A 1-unit increase in credit multiplies odds by {odds_ratios[1]:.3f}")
```

Step 15. Generate predicted probabilities:

```

# Get predicted probabilities

y_prob = log_reg.predict_proba(X_train)

print("\nPredicted Probabilities:")

print(" Column 0 = P(y=0), Column 1 = P(y=1)")

for i in range(len(X_train)):

    print(f" Sample {i+1}: P(Reject)={y_prob[i,0]:.4f}, "
          f"P(Approve)={y_prob[i,1]:.4f}")

```

Step 16. Make class predictions and evaluate accuracy:

```

# Generate class predictions

y_pred = log_reg.predict(X_train)

# Calculate accuracy

accuracy = accuracy_score(y_train, y_pred)

print("\nTraining accuracy:", accuracy * 100, "%")

# Display predictions vs actual

print("\nSample-by-sample results:")

for i in range(len(y_train)):

    status = "Correct" if y_train[i] == y_pred[i] else "INCORRECT"

    print(f" Sample {i+1}: Actual={y_train[i]}, "
          f"Predicted={y_pred[i]}, P(y=1)={y_prob[i,1]:.4f} - {status}")

```

Step 17. Generate confusion matrix and classification report:

```
# Compute confusion matrix  
cm = confusion_matrix(y_train, y_pred)  
  
print("\nConfusion Matrix:")  
  
print("      Predicted")  
  
print("      Reject Approve")  
  
print(f" Actual Reject {cm[0,0]} {cm[0,1]}")  
print(f" Actual Approve {cm[1,0]} {cm[1,1]}")  
  
  
# Display classification report  
  
print("\nClassification Report:")  
  
print(classification_report(y_train, y_pred,  
                           target_names=['Rejected', 'Approved']))
```

Step 18. Calculate ROC-AUC score:

```
# Compute ROC-AUC  
auc_score = roc_auc_score(y_train, y_prob[:, 1])  
  
print("ROC-AUC Score:", auc_score)
```

Step 19. Record the results in your laboratory notebook:

Learned coefficient beta_1: _____

Learned coefficient beta_2: _____

Learned intercept beta_0: _____

Odds ratio for x_1: _____

Odds ratio for x_2: _____

Training accuracy: _____ %

ROC-AUC score: _____

PART D: VISUALIZATION

Step 20. Create a scatter plot with predicted probabilities:

```
# Create figure with two subplots
fig, axes = plt.subplots(1, 2, figsize=(14, 6))
```

```
# Subplot 1: Training data with class labels
```

```
ax1 = axes[0]
class_0 = X_train[y_train == 0]
class_1 = X_train[y_train == 1]
```

```
ax1.scatter(class_0[:, 0], class_0[:, 1],
c='red', marker='x', s=150,
```

```

linewidths=2, label='Class 0 (Rejected)')

ax1.scatter(class_1[:, 0], class_1[:, 1],
            c='blue', marker='o', s=150,
            label='Class 1 (Approved)')

ax1.set_xlabel('x_1 (Normalized Income)', fontsize=12)
ax1.set_ylabel('x_2 (Normalized Credit Score)', fontsize=12)
ax1.set_title('Training Data', fontsize=14)
ax1.legend()
ax1.grid(True, alpha=0.3)
ax1.set_xlim(0, 1)
ax1.set_ylim(0, 1)

```

Step 21. Add the decision boundary to the plot:

```

# Extract coefficients
beta1, beta2 = log_reg.coef_[0]
beta0 = log_reg.intercept_[0]

# Calculate decision boundary line (where p = 0.5, z = 0)
x1_range = np.linspace(0, 1, 100)
x2_boundary = -(beta1/beta2) * x1_range - (beta0/beta2)

# Plot decision boundary
ax1.plot(x1_range, x2_boundary, 'g-',
          linewidth=2, label='Decision Boundary (p=0.5)')

```

```

ax1.legend()

# Print decision boundary equation
print(f"\nDecision Boundary Equation:")
print(f" {beta1:.4f}*x_1 + {beta2:.4f}*x_2 + {beta0:.4f} = 0")
print(f" x_2 = {-beta1/beta2:.4f}*x_1 + {-beta0/beta2:.4f}")

```

Step 22. Create probability surface visualization:

```

# Subplot 2: Probability contours
ax2 = axes[1]

# Create meshgrid for probability surface
xx1, xx2 = np.meshgrid(np.linspace(0, 1, 100),
                       np.linspace(0, 1, 100))
X_grid = np.c_[xx1.ravel(), xx2.ravel()]

# Get probabilities for grid
Z = log_reg.predict_proba(X_grid)[:, 1].reshape(xx1.shape)

# Plot probability contours
contour = ax2.contourf(xx1, xx2, Z, levels=20, cmap='RdYlBu', alpha=0.8)
plt.colorbar(contour, ax=ax2, label='P(Approved)')

# Overlay training points
ax2.scatter(class_0[:, 0], class_0[:, 1],
            color='blue', s=50, alpha=0.5)
ax2.scatter(class_1[:, 0], class_1[:, 1],
            color='red', s=50, alpha=0.5)

```

```

c='red', marker='x', s=150, linewidths=2)

ax2.scatter(class_1[:, 0], class_1[:, 1],
            c='blue', marker='o', s=150, edgecolors='black')

# Add decision boundary

ax2.contour(xx1, xx2, Z, levels=[0.5], colors='green', linewidths=2)

ax2.set_xlabel('x_1 (Normalized Income)', fontsize=12)
ax2.set_ylabel('x_2 (Normalized Credit Score)', fontsize=12)
ax2.set_title('Probability Surface with Decision Boundary', fontsize=14)

plt.tight_layout()

plt.savefig('logistic_regression_visualization.png', dpi=150)

plt.show()

```

Step 23. Plot the ROC curve:

```

# Calculate ROC curve

fpr, tpr, thresholds = roc_curve(y_train, y_prob[:, 1])

plt.figure(figsize=(8, 6))

plt.plot(fpr, tpr, 'b-', linewidth=2,
         label=f'ROC Curve (AUC = {auc_score:.3f})')

plt.plot([0, 1], [0, 1], 'k--', linewidth=1, label='Random Classifier')

plt.xlabel('False Positive Rate', fontsize=12)
plt.ylabel('True Positive Rate', fontsize=12)

```

```
plt.title('Receiver Operating Characteristic (ROC) Curve', fontsize=14)  
plt.legend(loc='lower right')  
plt.grid(True, alpha=0.3)  
plt.savefig('roc_curve.png', dpi=150)  
plt.show()
```

PART E: TESTING WITH NEW DATA AND THRESHOLD ANALYSIS

Step 24. Classify new applicants and obtain probabilities:

```
# Define new test instances
```

```
X_new = np.array([
```

[0.7, 0.5],

[0.4, 0.6],

[0.5, 0.5],

[0.85, 0.75]

1)

```
# Get predictions and probabilities
```

```
new_predictions = log_reg.predict(X_new)
```

```
new_probabilities = log_reg.predict_proba(X_new)
```

```
print("New Applicant Predictions:")
```

```
for i, (x, pred, prob) in enumerate(zip(X_new, new_predictions,
```

```
new_probabilities)):
```

```

decision = "Approved" if pred == 1 else "Rejected"

print(f" Applicant {i+1}: x_1={x[0]}, x_2={x[1]}")

print(f" P(Approved) = {prob[1]:.4f} -> {decision}")

```

Step 25. Analyze effect of different thresholds:

```

# Test different thresholds

thresholds_to_test = [0.3, 0.4, 0.5, 0.6, 0.7]

print("\nThreshold Analysis for Training Data:")
print("-" * 50)

for thresh in thresholds_to_test:

    y_pred_thresh = (y_prob[:, 1] >= thresh).astype(int)

    acc = accuracy_score(y_train, y_pred_thresh)

    cm_thresh = confusion_matrix(y_train, y_pred_thresh)

    print(f"\nThreshold = {thresh}:")
    print(f" Accuracy: {acc*100:.1f}%")
    print(f" True Negatives: {cm_thresh[0,0]}, False Positives: {cm_thresh[0,1]}")
    print(f" False Negatives: {cm_thresh[1,0]}, True Positives: {cm_thresh[1,1]}")

```

Step 26. Record threshold analysis results:

Optimal threshold for this problem: _____

Justification: _____

V. INDIVIDUAL ASSIGNMENT

Complete the problems corresponding to your assigned variant number.

Show all work, include Python code, and provide screenshots of output.

VARIANT 1

Problem 1.1 (Computational)

A logistic regression model for customer churn has the following parameters:

$\beta_0 = -2.5$

β_1 (usage decline) = 3.2

β_2 (complaint count) = 1.8

A customer has $x_1 = 0.6$ and $x_2 = 0.4$.

- a) Calculate the log-odds (z) for this customer.
- b) Calculate the probability of churn.
- c) Calculate the odds of churn.
- d) If the company uses a threshold of 0.4, what is the prediction?

e) Calculate and interpret the odds ratios for both features.

Problem 1.2 (Applied Business Problem)

A telecommunications company wants to predict customer churn.

[TABLE 2: Training Data for Problem 1.2]

Insert table with 9 rows and 4 columns

Headers: Customer | x_1 (Usage Decline) | x_2 (Complaints) | Churned

1 | 0.8 | 0.7 | 1

2 | 0.2 | 0.1 | 0

3 | 0.7 | 0.6 | 1

4 | 0.1 | 0.3 | 0

5 | 0.6 | 0.8 | 1

6 | 0.3 | 0.2 | 0

7 | 0.9 | 0.5 | 1

8 | 0.2 | 0.4 | 0

a) Using scikit-learn with C=1.0, solver='lbfgs', and random_state=42,

train a logistic regression model. Report the coefficients,

intercept, and odds ratios for each feature.

b) Generate predicted probabilities for all training samples.

Create a table showing actual class, predicted probability,

and predicted class (using threshold=0.5).

c) Compute and display the confusion matrix, accuracy, precision, recall, and F1-score. Calculate the ROC-AUC score.

Problem 1.3 (Extension - Business Decision Analysis)

The company has the following cost structure:

- Cost of retention offer to at-risk customer: \$50
- Lost revenue if chunner is not retained: \$500
- Retention offer success rate: 40%

- a) Calculate the expected value of offering retention to a customer with $P(\text{churn}) = 0.7$.
- b) Determine the probability threshold above which it is profitable to offer retention.
- c) Using this optimal threshold, re-classify the training data and compare results to the 0.5 threshold.

VARIANT 2

Problem 2.1 (Computational)

A logistic regression model for loan default has the following parameters:

$$\beta_0 = -1.8$$

$$\beta_1 \text{ (debt ratio)} = 2.5$$

$$\beta_2 \text{ (payment history)} = -2.0$$

A borrower has $x_1 = 0.7$ and $x_2 = 0.6$.

- a) Calculate the log-odds (z) for this borrower.
- b) Calculate the probability of default.
- c) Calculate the odds of default.
- d) If the bank uses a threshold of 0.3, what is the prediction?
- e) Calculate and interpret the odds ratios for both features.

Problem 2.2 (Applied Business Problem)

A credit card company wants to predict payment default.

[TABLE 3: Training Data for Problem 2.2]

Insert table with 9 rows and 4 columns

Headers: Account | x_1 (Debt Ratio) | x_2 (Payment History) | Defaulted

1 | 0.9 | 0.2 | 1

2 | 0.2 | 0.8 | 0

3 | 0.8 | 0.3 | 1

4 | 0.3 | 0.9 | 0

5 | 0.7 | 0.1 | 1

6 | 0.1 | 0.7 | 0

7 | 0.6 | 0.4 | 1

8 | 0.4 | 0.6 | 0

a) Train a logistic regression model with C=1.0, solver='lbfgs',

random_state=42. Report all parameters and odds ratios.

b) Generate the probability surface plot with decision boundary.

c) Calculate ROC-AUC and plot the ROC curve.

Problem 2.3 (Extension - Business Decision Analysis)

The company faces these costs:

- False positive (denying good customer): \$200 lost revenue
- False negative (approving defaulter): \$1,500 loss

a) Calculate the cost ratio and optimal threshold.

b) What is the total expected cost on a test set of 1,000 accounts
with 5% false positive rate and 12% false negative rate?

c) If the default rate is 8%, what is the cost per account?

VARIANT 3

Problem 3.1 (Computational)

A logistic regression model for email response has the following parameters:

$$\beta_0 = -4.0$$

$$\beta_1 (\text{recency}) = 2.8$$

$$\beta_2 (\text{frequency}) = 1.5$$

A customer has $x_1 = 0.5$ and $x_2 = 0.7$.

- a) Calculate the log-odds (z) for this customer.
- b) Calculate the probability of response.
- c) Calculate the odds of response.
- d) If the company uses a threshold of 0.2, what is the prediction?
- e) Calculate and interpret the odds ratios for both features.

Problem 3.2 (Applied Business Problem)

A marketing firm wants to predict email campaign response.

[TABLE 4: Training Data for Problem 3.2]

Insert table with 9 rows and 4 columns

Headers: Customer | x_1 (Recency Score) | x_2 (Frequency Score) | Responded

1 | 0.9 | 0.8 | 1

2 | 0.1 | 0.2 | 0

3 | 0.8 | 0.6 | 1

4 | 0.2 | 0.3 | 0

5 | 0.7 | 0.9 | 1

6 | 0.3 | 0.1 | 0

7 | 0.6 | 0.7 | 1

8 | 0.4 | 0.4 | 0

a) Train a logistic regression model. Report coefficients and calculate odds ratios with interpretation.

b) Calculate predicted probabilities and classify using thresholds of 0.3, 0.5, and 0.7. Compare confusion matrices.

c) A new customer has $x_1 = 0.55$ and $x_2 = 0.65$. Predict both the probability and class using the optimal threshold.

Problem 3.3 (Extension - Campaign Profitability)

Campaign economics:

- Cost per email: \$0.50
- Revenue per responder: \$25
- Historical response rate: 12%

- a) Calculate break-even response probability.
 - b) Using logistic regression predictions, determine which customers should receive the campaign (expected profit > 0).
 - c) Calculate expected campaign profit for 10,000 customers.
-

VARIANT 4

Problem 4.1 (Computational)

A logistic regression model for insurance claim approval has:

$$\beta_0 = -3.0$$

$$\beta_1 (\text{claim amount}) = -1.5$$

$$\beta_2 (\text{policy tenure}) = 4.0$$

A claim has $x_1 = 0.4$ and $x_2 = 0.8$.

- a) Calculate the log-odds (z) for this claim.
- b) Calculate the probability of approval.
- c) Calculate the odds of approval.
- d) If the company uses a threshold of 0.6, what is the prediction?
- e) Calculate and interpret the odds ratios for both features.

Problem 4.2 (Applied Business Problem)

An insurance company wants to automate claim screening.

[TABLE 5: Training Data for Problem 4.2]

Insert table with 9 rows and 4 columns

Headers: Claim | x_1 (Claim Amount) | x_2 (Policy Tenure) | Approved

1 | 0.2 | 0.9 | 1

2 | 0.8 | 0.2 | 0

3 | 0.3 | 0.7 | 1

4 | 0.9 | 0.3 | 0

5 | 0.4 | 0.8 | 1

6 | 0.7 | 0.1 | 0

7 | 0.1 | 0.6 | 1

8 | 0.6 | 0.4 | 0

a) Train a logistic regression model. Report all parameters.

b) Generate the decision boundary equation and plot.

c) Evaluate model with confusion matrix and all metrics.

Problem 4.3 (Extension - Cost-Sensitive Classification)

Cost structure:

- Wrongly denied claim (legal costs): \$10,000
- Wrongly approved claim (fraud loss): \$5,000

- a) What is the optimal classification threshold given these costs?
 - b) Compare total costs using thresholds 0.3, 0.5, and 0.67.
 - c) Recommend the best threshold with justification.
-

VARIANT 5

Problem 5.1 (Computational)

A logistic regression model for subscription renewal has:

$$\beta_0 = -2.0$$

$$\beta_1 (\text{engagement}) = 3.5$$

$$\beta_2 (\text{tenure}) = 1.2$$

A subscriber has $x_1 = 0.7$ and $x_2 = 0.3$.

- a) Calculate the log-odds (z) for this subscriber.
- b) Calculate the probability of renewal.
- c) Calculate the odds of renewal.
- d) If the company uses a threshold of 0.5, what is the prediction?

e) Calculate and interpret the odds ratios for both features.

Problem 5.2 (Applied Business Problem)

A streaming service wants to predict subscription renewals.

[TABLE 6: Training Data for Problem 5.2]

Insert table with 9 rows and 4 columns

Headers: Subscriber | x_1 (Engagement) | x_2 (Tenure) | Renewed

1 | 0.9 | 0.6 | 1

2 | 0.1 | 0.3 | 0

3 | 0.8 | 0.5 | 1

4 | 0.2 | 0.2 | 0

5 | 0.7 | 0.8 | 1

6 | 0.3 | 0.4 | 0

7 | 0.6 | 0.7 | 1

8 | 0.4 | 0.1 | 0

a) Train and evaluate a logistic regression model.

b) Create probability surface plot showing renewal likelihood.

c) Analyze feature importance using odds ratios.

Problem 5.3 (Extension - Revenue Optimization)

Subscription economics:

- Monthly subscription: \$15
- Retention discount offer: \$5/month for 3 months
- Discount acceptance rate: 60%

- a) Calculate customer lifetime value for 12-month horizon.
 - b) Determine P(renewal) threshold for offering discount.
 - c) Expected revenue impact of targeting based on model predictions.
-

VARIANT 6

Problem 6.1 (Computational)

A logistic regression model for product purchase has:

$$\beta_0 = -3.5$$

$$\beta_1 (\text{browse time}) = 2.0$$

$$\beta_2 (\text{cart value}) = 2.5$$

A visitor has $x_1 = 0.6$ and $x_2 = 0.5$.

- a) Calculate the log-odds (z) for this visitor.

- b) Calculate the probability of purchase.
- c) Calculate the odds of purchase.
- d) If the site uses a threshold of 0.4, what is the prediction?
- e) Calculate and interpret the odds ratios for both features.

Problem 6.2 (Applied Business Problem)

An e-commerce site wants to predict purchase completion.

[TABLE 7: Training Data for Problem 6.2]

Insert table with 9 rows and 4 columns

Headers: Session | x_1 (Browse Time) | x_2 (Cart Value) | Purchased

1 | 0.8 | 0.9 | 1

2 | 0.2 | 0.1 | 0

3 | 0.7 | 0.7 | 1

4 | 0.1 | 0.3 | 0

5 | 0.9 | 0.6 | 1

6 | 0.3 | 0.2 | 0

7 | 0.6 | 0.8 | 1

8 | 0.4 | 0.4 | 0

- a) Train model and report coefficients with interpretations.
- b) Plot ROC curve and calculate AUC.

c) Determine optimal threshold using Youden's J statistic
($J = \text{sensitivity} + \text{specificity} - 1$).

Problem 6.3 (Extension - Conversion Optimization)

Site economics:

- Average order value: \$75
- Popup discount offer: 10% off
- Popup increases conversion by 25%

- a) Calculate break-even $P(\text{purchase})$ for showing popup.
 - b) Expected revenue change from popup targeting strategy.
 - c) Compare "show to all" vs "model-based targeting" approaches.
-

VARIANT 7

Problem 7.1 (Computational)

A logistic regression model for employee attrition has:

$$\beta_0 = -1.5$$

$$\beta_1 (\text{satisfaction}) = -3.0$$

$$\beta_2 (\text{workload}) = 2.2$$

An employee has $x_1 = 0.4$ and $x_2 = 0.7$.

- a) Calculate the log-odds (z) for this employee.
- b) Calculate the probability of attrition.
- c) Calculate the odds of attrition.
- d) If HR uses a threshold of 0.35, what is the prediction?
- e) Calculate and interpret the odds ratios for both features.

Problem 7.2 (Applied Business Problem)

A company wants to predict employee turnover.

[TABLE 8: Training Data for Problem 7.2]

Insert table with 9 rows and 4 columns

Headers: Employee | x_1 (Satisfaction) | x_2 (Workload) | Left

1 | 0.2 | 0.8 | 1

2 | 0.9 | 0.2 | 0

3 | 0.3 | 0.7 | 1

4 | 0.8 | 0.3 | 0

5 | 0.1 | 0.9 | 1

6 | 0.7 | 0.4 | 0

7 | 0.4 | 0.6 | 1

8 | 0.6 | 0.5 | 0

- a) Train logistic regression and analyze coefficients.
- b) Which feature has stronger influence? Justify with odds ratios.
- c) Predict attrition probability for employee with $x_1=0.5$, $x_2=0.5$.

Problem 7.3 (Extension - HR Cost Analysis)

HR costs:

- Turnover cost per employee: \$25,000
- Retention intervention cost: \$3,000
- Intervention success rate: 50%

- a) Calculate expected value of intervention for $P(\text{leave})=0.6$.
 - b) Determine optimal intervention threshold.
 - c) Expected annual savings with model-based targeting (100 employees).
-

VARIANT 8

Problem 8.1 (Computational)

A logistic regression model for hospital readmission has:

$\beta_0 = -2.8$

$\beta_1 (\text{severity}) = 2.5$

$\beta_2 (\text{age factor}) = 1.8$

A patient has $x_1 = 0.5$ and $x_2 = 0.6$.

- a) Calculate the log-odds (z) for this patient.
- b) Calculate the probability of readmission.
- c) Calculate the odds of readmission.
- d) If the hospital uses a threshold of 0.4, what is the prediction?
- e) Calculate and interpret the odds ratios for both features.

Problem 8.2 (Applied Business Problem)

A hospital wants to predict 30-day readmissions.

[TABLE 9: Training Data for Problem 8.2]

Insert table with 9 rows and 4 columns

Headers: Patient | x_1 (Severity Index) | x_2 (Age Factor) | Readmitted

1 | 0.8 | 0.7 | 1

2 | 0.2 | 0.3 | 0

3 | 0.7 | 0.8 | 1

4 | 0.3 | 0.2 | 0

5 | 0.9 | 0.6 | 1

6 | 0.1 | 0.4 | 0

7 | 0.6 | 0.9 | 1

8 | 0.4 | 0.1 | 0

- a) Train and evaluate logistic regression model.
- b) Generate predicted probabilities and rank patients by risk.
- c) Identify top 25% highest risk patients.

Problem 8.3 (Extension - Healthcare Economics)

Cost structure:

- Readmission penalty: \$15,000 per case
- Enhanced discharge program cost: \$2,000 per patient
- Program effectiveness: 35% reduction in readmission

- a) Calculate expected savings from targeting high-risk patients.
 - b) Determine probability threshold for program enrollment.
 - c) Compare targeted vs universal program implementation.
-

VARIANT 9

Problem 9.1 (Computational)

A logistic regression model for fraud detection has:

$$\beta_0 = -5.0$$

$$\beta_1 (\text{amount deviation}) = 3.5$$

$$\beta_2 (\text{location risk}) = 2.8$$

A transaction has $x_1 = 0.8$ and $x_2 = 0.6$.

- a) Calculate the log-odds (z) for this transaction.
- b) Calculate the probability of fraud.
- c) Calculate the odds of fraud.
- d) If the bank uses a threshold of 0.3, what is the prediction?
- e) Calculate and interpret the odds ratios for both features.

Problem 9.2 (Applied Business Problem)

A payment processor wants to detect fraudulent transactions.

[TABLE 10: Training Data for Problem 9.2]

Insert table with 9 rows and 4 columns

Headers: Transaction | x_1 (Amount Score) | x_2 (Location Risk) | Fraud

1 | 0.9 | 0.8 | 1

2 | 0.1 | 0.1 | 0

3 | 0.8 | 0.7 | 1

4 | 0.2 | 0.2 | 0

5 | 0.7 | 0.9 | 1

6 | 0.3 | 0.3 | 0

7 | 0.9 | 0.6 | 1

8 | 0.1 | 0.4 | 0

- a) Train logistic regression model with analysis.
- b) Evaluate using precision, recall, and F1-score.
- c) Plot precision-recall trade-off for different thresholds.

Problem 9.3 (Extension - Fraud Economics)

Transaction costs:

- False positive (blocked legitimate): \$30 customer service
- False negative (missed fraud): \$800 average loss
- Fraud rate: 0.5% of transactions

- a) Calculate optimal threshold given cost structure.
 - b) Expected monthly loss on 1 million transactions.
 - c) Compare thresholds 0.1, 0.3, and 0.5 on cost basis.
-

VARIANT 10

Problem 10.1 (Computational)

A logistic regression model for lead conversion has:

$$\beta_0 = -3.2$$

$$\beta_1 (\text{engagement score}) = 2.8$$

$$\beta_2 (\text{company size}) = 1.5$$

A lead has $x_1 = 0.7$ and $x_2 = 0.5$.

- a) Calculate the log-odds (z) for this lead.
- b) Calculate the probability of conversion.
- c) Calculate the odds of conversion.
- d) If sales uses a threshold of 0.4, what is the prediction?
- e) Calculate and interpret the odds ratios for both features.

Problem 10.2 (Applied Business Problem)

A B2B company wants to predict sales lead conversion.

[TABLE 11: Training Data for Problem 10.2]

Insert table with 9 rows and 4 columns

Headers: Lead | x_1 (Engagement) | x_2 (Company Size) | Converted

1 | 0.9 | 0.7 | 1

2 | 0.1 | 0.2 | 0

3 | 0.8 | 0.5 | 1

4 | 0.2 | 0.4 | 0

5 | 0.7 | 0.8 | 1

6 | 0.3 | 0.1 | 0

7 | 0.6 | 0.6 | 1

8 | 0.4 | 0.3 | 0

a) Build and evaluate logistic regression model.

b) Rank leads by conversion probability for prioritization.

c) Create decision boundary visualization.

Problem 10.3 (Extension - Sales Resource Allocation)

Sales economics:

- Average deal size: \$50,000
- Sales rep cost per lead: \$500 (time investment)
- Close rate if pursued: Based on model probability

a) Calculate expected value of pursuing leads at different probabilities.

b) Determine minimum P(conversion) for lead to be worthwhile.

c) Expected revenue from top 20 leads ranked by probability.

VI. LABORATORY REPORT REQUIREMENTS

The laboratory report shall conform to the following specifications:

1. COVER PAGE

- Course name and section number
- Laboratory exercise title and number
- Student name and identification number
- Date of submission
- Variant number assigned

2. MAIN BODY

- All manual calculations shown with intermediate steps
- Proper use of significant figures (4 decimal places)
- Clear labeling of all computed values
- Interpretation of results in business context

3. PYTHON CODE DOCUMENTATION

- Complete, executable Python code for all problems
- Code must include comments explaining each section
- Output must be clearly visible in screenshots
- All variable names should be descriptive

4. REQUIRED SCREENSHOTS

- Screenshot of imported libraries and version check
- Screenshot of model training and coefficient output
- Screenshot of odds ratios with interpretation
- Screenshot of predicted probabilities table
- Screenshot of confusion matrix and classification report
- Screenshot of ROC curve with AUC value
- Screenshot of decision boundary and probability surface plots

5. INTERPRETATION REQUIREMENTS

- Written interpretation of each coefficient's meaning
- Business implications of odds ratios
- Recommendation for classification threshold with justification
- Discussion of model limitations

6. FORMAT SPECIFICATIONS

- Report typed in 12-point Times New Roman or similar
- Pages numbered consecutively
- Code formatted in Courier New or monospace font
- Figures labeled and captioned

7. FILE SUBMISSION

- PDF of complete laboratory report
- Jupyter notebook file (.ipynb) with all code
- Saved figure files (.png) of all plots

VII. ANALYSIS QUESTIONS

Answer the following questions in complete sentences:

1. Explain why logistic regression uses the sigmoid function rather than directly using the linear predictor for classification.

What problems would arise if we used a linear model for probability estimation?

Response: _____

2. A credit scoring model has an odds ratio of 2.5 for the feature "years employed." Explain in plain language what this means for a loan applicant. How would you communicate this finding to a non-technical business stakeholder?

Response: _____

-
-
-
3. In many business applications, the costs of false positives and false negatives are not equal. Explain how you would determine the optimal classification threshold when:
- a) False negatives are much more costly than false positives
 - b) False positives are much more costly than false negatives

Response: _____

4. Compare logistic regression to the single-layer perceptron studied in the previous laboratory. What are the key differences in terms of:

- a) Output interpretation
- b) Training method
- c) Decision boundary
- d) Practical applicability

Response: _____

5. A logistic regression model achieves 95% accuracy on training data but only 60% accuracy on new test data. What phenomenon does this illustrate? What modifications to the model or training process might improve generalization performance?

Response: _____

VIII. REFERENCE DATA

A. Scikit-learn LogisticRegression Class Parameters

```
LogisticRegression(  
    penalty='l2',      # Regularization type  
    C=1.0,           # Inverse regularization strength  
    fit_intercept=True,  # Whether to fit intercept  
    solver='lbfgs',     # Optimization algorithm  
    max_iter=100,       # Maximum iterations  
    random_state=None,   # Random seed  
    tol=1e-4,          # Convergence tolerance  
    class_weight=None,   # Class weights for imbalanced data  
    multi_class='auto'    # Multiclass strategy  
)
```

Available solvers:

- 'lbfgs' - Good for small datasets (default)
- 'liblinear' - Good for small datasets, L1 penalty
- 'newton-cg' - Good for multiclass
- 'sag' - Fast for large datasets
- 'saga' - Fast, supports L1 and elastic net

B. Key LogisticRegression Attributes After Fitting

```
model.coef_      # Coefficients, shape (1, n_features)  
model.intercept_ # Intercept, shape (1,)  
model.classes_  # Class labels  
model.n_iter_   # Actual iterations per class
```

C. Key LogisticRegression Methods

```
model.fit(X, y)      # Train the model  
model.predict(X)     # Predict class labels  
model.predict_proba(X) # Predict probabilities  
model.predict_log_proba(X) # Predict log-probabilities  
model.score(X, y)    # Return accuracy  
model.decision_function(X) # Return log-odds (z values)
```

D. Summary of Logistic Regression Formulas

Linear Predictor (Log-Odds):

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots$$

Sigmoid Function:

$$p = 1 / (1 + e^{-z})$$

Odds:

$$\text{Odds} = p / (1 - p)$$

Odds Ratio:

$$\text{OR}_i = e^{(\beta_i)}$$

Decision Boundary (2 features):

$$x_2 = -(\beta_1/\beta_2)x_1 - (\beta_0/\beta_2)$$

Accuracy:

$$(TP + TN) / (TP + TN + FP + FN)$$

Precision:

$$TP / (TP + FP)$$

Recall (Sensitivity):

$$TP / (TP + FN)$$

F1-Score:

$$2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$$

Specificity:

$$TN / (TN + FP)$$

E. Useful Code Snippets

```
# Calculate odds ratios  
  
odds_ratios = np.exp(model.coef_[0])  
  
  
# Custom threshold prediction  
  
threshold = 0.3  
  
y_pred_custom = (model.predict_proba(X)[:, 1] >= threshold).astype(int)  
  
  
# Calculate optimal threshold (Youden's J)  
  
fpr, tpr, thresholds = roc_curve(y_true, y_prob)  
  
optimal_idx = np.argmax(tpr - fpr)  
  
optimal_threshold = thresholds[optimal_idx]  
  
  
# Cost-sensitive threshold  
  
# threshold = cost_FN / (cost_FP + cost_FN)  
  
  
# Calculate log-odds manually  
  
z = np.dot(X, model.coef_.T) + model.intercept_
```

F. Confusion Matrix Interpretation

		Predicted	
		Negative	Positive
Actual Negative	TN	FP	
	FN		TP

Actual Positive FN TP

Where:

TN = True Negatives (correctly rejected)

FP = False Positives (Type I error)

FN = False Negatives (Type II error)

TP = True Positives (correctly accepted)