Data Mining Assignment 2

June 2025

Students:

- Shema Hugor, ID: 100763
- Godfrey Mawulizo ,100901
- Shyaka Kevin, ID: 100915
- Niyonsenga Jean Paul, ID: 100888
- Gumira Theophile, ID: 100920
- Heritier Ntwali, ID: 100923
- Nyirimanzi Jean Claude, ID: 100882

Lecturer: Dr.Pacifique Nizeyimana
Cohort: 2023

Introduction

This report presents four exercises from the *Introduction to Statistical Learning with Applications in Python* (ISLP) dataset, completed as part of the Data Mining Assignment 4. The exercises include proving the equivalence

of logistic and logit representations (Question 1), applying logistic regression to predict academic success (Question 6), deriving coefficients for Quadratic Discriminant Analysis (Question 11), and building classification models on the Boston dataset (Question 16).

1 Question 1: Proving Equivalence of Logistic and Logit Representations

Problem Statement: "Using a little bit of algebra, prove that (4.2) is equivalent to (4.3). In other words, the logistic function representation and logit representation for the logistic regression model are equivalent."

Solution:

Equation (4.2): $p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$

Equation (4.3): $\frac{p(X)}{1-p(X)} = e^{\beta_0+\beta_1 X}$

1. **Start with $p(X)$:** Use the given formula $p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$.
2. **Compute $1 - p(X)$:**
   - Write $1 - p(X) = 1 - \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$.
   - Express 1 with the same denominator: $1 = \frac{1+e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$.
   - Subtract: $1 - p(X) = \frac{1+e^{\beta_0+\beta_1 X}-e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}} = \frac{1}{1+e^{\beta_0+\beta_1 X}}$.
3. **Form the odds $\frac{p(X)}{1-p(X)}$:**
   - Set up the fraction: $\frac{p(X)}{1-p(X)} = \frac{\frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}}{\frac{1}{1+e^{\beta_0+\beta_1 X}}}$.
   - Divide by multiplying by the reciprocal: $\frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}} \cdot \frac{1+e^{\beta_0+\beta_1 X}}{1} = e^{\beta_0+\beta_1 X}$.
4. **Verify reverse:**
   - Start with $\frac{p(X)}{1-p(X)} = e^{\beta_0+\beta_1 X}$, let $z = e^{\beta_0+\beta_1 X}$.
   - Solve: $\frac{p(X)}{1-p(X)} = z$, so $p(X) = z(1 - p(X))$.
   - Rearrange: $p(X) = z - zp(X)$, $p(X) + zp(X) = z$, $p(X)(1+z) = z$.
   - Isolate $p(X)$: $p(X) = \frac{z}{1+z} = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$.

Answer: The logistic function (4.2) and logit representation (4.3) are equivalent, proven by algebraic manipulation in both directions.

2 Question 6: Logistic Regression Application

Problem Statement: "Suppose we collect data for a group of students in a statistics class with variables X_1 = hours studied, X_2 = undergrad GPA, and Y = receive an A. We fit a logistic regression and produce estimated coefficients, $\hat{\beta}_0 = -6$, $\hat{\beta}_1 = 0.05$, $\hat{\beta}_2 = 1$." (a) "Estimate the probability that a student who studies for 40 h and has an undergrad GPA of 3.5 gets an A in the class." (b) "How many hours would the student in part (a) need to study to have a 50% chance of getting an A in the class?"

Solution:

2.1 Part (a): Probability Calculation

- Model: $p(X) = \frac{1}{1+e^{-(\hat{\beta}_0+\hat{\beta}_1 X_1+\hat{\beta}_2 X_2)}}$ - Given: $X_1 = 40$, $X_2 = 3.5$, $\hat{\beta}_0 = -6$, $\hat{\beta}_1 = 0.05$, $\hat{\beta}_2 = 1$.

1. **Compute the linear combination:**
   - Start with the formula: $\hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2$.
   - Substitute: $-6 + 0.05 \cdot 40 + 1 \cdot 3.5$.
   - Calculate $0.05 \cdot 40 = 2$.
   - Calculate $1 \cdot 3.5 = 3.5$.
   - Add: $-6 + 2 = -4$, then $-4 + 3.5 = -0.5$.
2. **Apply the logistic function:**
   - Use $p(X) = \frac{1}{1+e^{-(-0.5)}}$.
   - Simplify exponent: $-(-0.5) = 0.5$, so $p(X) = \frac{1}{1+e^{0.5}}$.
3. **Compute the value:**
   - Find $e^{0.5} \approx 1.6487$ (using a calculator).
   - Add: $1 + 1.6487 = 2.6487$.
   - Divide: $\frac{1}{2.6487} \approx 0.3775$ (rounded to 4 decimals).

Answer for Part (a): The probability is approximately 0.3775.

2.2 Part (b): Hours for 50% Probability

- Given: $p(X) = 0.5$, $X_2 = 3.5$.

1. **Set up the equation:**
   - Use $0.5 = \frac{1}{1+e^{-(-6+0.05X_1+1\cdot 3.5)}}$.
2. **Solve for the exponent:**
   - Multiply both sides by denominator: $0.5(1+e^{-(-6+0.05X_1+3.5)}) = 1$.
   - Divide: $1 + e^{-(-6+0.05X_1+3.5)} = 2$.
   - Subtract 1: $e^{-(-6+0.05X_1+3.5)} = 1$.
   - Take natural log: $-(-6 + 0.05X_1 + 3.5) = \ln(1) = 0$.
3. **Simplify:**
   - $6 - 0.05X_1 - 3.5 = 0$.
   - $2.5 = 0.05X_1$.
4. **Solve for X_1:**
   - $X_1 = \frac{2.5}{0.05} = 50$.
5. **Verify:**
   - Check: $-6 + 0.05 \cdot 50 + 3.5 = -6 + 2.5 + 3.5 = 0$.
   - $p(X) = \frac{1}{1+e^0} = \frac{1}{2} = 0.5$.

Answer for Part (b): The student needs to study 50 hours.

3 Question 11: Deriving Coefficients in Equation (4.33)

Problem Statement: "Work out the detailed forms of a_k, b_{kj}, and $c_{kj\ell}$ in (4.33). Your answer should involve π_k, π_K, μ_k, μ_K, Σ_k, and Σ_K."

Solution: - **Equation (4.33):**

$$\log\left(\frac{\Pr(Y=k|X=x)}{\Pr(Y=K|X=x)}\right) = a_k + \sum_{j=1}^{p} b_{kj}x_j + \sum_{j=1}^{p}\sum_{\ell=1}^{p} c_{kj\ell}x_j x_\ell$$

1. **Discriminant Function for QDA:**

- $\delta_k(x) = -\frac{1}{2}\ln|\Sigma_k| - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \ln(\pi_k)$.
- Log-odds: $\delta_k(x) - \delta_K(x)$.

2. **Expand Log-Odds:**
   - $\delta_k(x) - \delta_K(x) = [-\frac{1}{2}\ln|\Sigma_k| + \ln(\pi_k) - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)] - [-\frac{1}{2}\ln|\Sigma_K| + \ln(\pi_K) - \frac{1}{2}(x-\mu_K)^T\Sigma_K^{-1}(x-\mu_K)]$.
   - Simplify: $\ln\left(\frac{\pi_k}{\pi_K}\right) - \frac{1}{2}\ln\left(\frac{|\Sigma_k|}{|\Sigma_K|}\right) - \frac{1}{2}[(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) - (x-\mu_K)^T\Sigma_K^{-1}(x-\mu_K)]$.
3. **Expand Quadratic Terms:**
   - $(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) = x^T\Sigma_k^{-1}x - 2x^T\Sigma_k^{-1}\mu_k + \mu_k^T\Sigma_k^{-1}\mu_k$.
   - $(x-\mu_K)^T\Sigma_K^{-1}(x-\mu_K) = x^T\Sigma_K^{-1}x - 2x^T\Sigma_K^{-1}\mu_K + \mu_K^T\Sigma_K^{-1}\mu_K$.
   - Difference: $x^T(\Sigma_K^{-1} - \Sigma_k^{-1})x - 2x^T(\Sigma_k^{-1}\mu_k - \Sigma_K^{-1}\mu_K) + (\mu_k^T\Sigma_k^{-1}\mu_k - \mu_K^T\Sigma_K^{-1}\mu_K)$.
   - Multiply by $-\frac{1}{2}$: $-\frac{1}{2}x^T(\Sigma_k^{-1} - \Sigma_K^{-1})x + x^T(\Sigma_k^{-1}\mu_k - \Sigma_K^{-1}\mu_K) - \frac{1}{2}(\mu_k^T\Sigma_k^{-1}\mu_k - \mu_K^T\Sigma_K^{-1}\mu_K)$.
4. **Match Coefficients:**
   - $a_k = \ln\left(\frac{\pi_k}{\pi_K}\right) - \frac{1}{2}\ln\left(\frac{|\Sigma_k|}{|\Sigma_K|}\right) - \frac{1}{2}(\mu_k^T\Sigma_k^{-1}\mu_k - \mu_K^T\Sigma_K^{-1}\mu_K)$.
   - $b_{kj} = [(\Sigma_k^{-1}\mu_k - \Sigma_K^{-1}\mu_K)]_j$ (j-th component).
   - $c_{kj\ell} = -\frac{1}{2}[(\Sigma_k^{-1})_{j\ell} - (\Sigma_K^{-1})_{j\ell}]$.

Answer:

- $a_k = \ln\left(\frac{\pi_k}{\pi_K}\right) - \frac{1}{2}\ln\left(\frac{|\Sigma_k|}{|\Sigma_K|}\right) - \frac{1}{2}(\mu_k^T\Sigma_k^{-1}\mu_k - \mu_K^T\Sigma_K^{-1}\mu_K)$
- $b_{kj} = [(\Sigma_k^{-1}\mu_k - \Sigma_K^{-1}\mu_K)]_j$
- $c_{kj\ell} = -\frac{1}{2}[(\Sigma_k^{-1})_{j\ell} - (\Sigma_K^{-1})_{j\ell}]$

4 Question 16: Classification Models on Boston Dataset

Problem Statement: "Using the Boston data set, fit classification models in order to predict whether a given suburb has a crime rate above or below the median. Explore logistic regression, LDA, naive Bayes, and KNN models

using various subsets of the predictors. Describe your findings. Hint: You will have to create the response variable yourself, using the variables that are contained in the Boston data set."

Solution:

4.1 Step 1: Load the Data and Create the Response Variable

The Boston data set is loaded from the ISLP package. Create a binary response variable high+crime where 1 indicates a crime rate above the median and 0 indicates below or equal to the median based on the crime column.

```
from ISLP import load_data
Boston = load_data("Boston")

# Calculate the median crime rate
crime_median = Boston['crim'].median()

# Create binary response variable
Boston['high_crime'] = (Boston['crim'] > crime_median).astype(int)

# Display the first few rows to verify
print(Boston.head())
```

Output:

```
   crim    zn  indus  chas    nox     rm   age     dis  rad  tax  ptratio  \
0  0.00632  18.0   2.31     0  0.538  6.575  65.2  4.0900    1  296     15.3
1  0.02731   0.0   7.07     0  0.469  6.421  78.9  4.9671    2  242     17.8
2  0.02729   0.0   7.07     0  0.469  7.185  61.1  4.9671    2  242     17.8
3  0.03237   0.0   2.18     0  0.458  6.998  45.8  6.0622    3  222     18.7
4  0.06905   0.0   2.18     0  0.458  7.147  54.2  6.0622    3  222     18.7

   lstat  medv  high_crime
0   4.98  24.0           0
1   9.14  21.6           0
2   4.03  34.7           0
3   2.94  33.4           0
4   5.33  36.2           0
```

4.2 Step 2: Split the Data into Training and Test Sets

Split the data into training and test sets using 70% for training and 30% for testing, ensuring the response variable is included.

```
from sklearn.model_selection import train_test_split

# Define predictors and response
X = Boston.drop(['crim', 'high_crime'], axis=1)  # Exclude crim and high_crime
y = Boston['high_crime']

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_

# Verify shapes
print("Training set shape:", X_train.shape)
print("Test set shape:", X_test.shape)
```

Output:

```
Training set shape: (354, 12)
Test set shape: (152, 12)
```

4.3 Step 3: Fit Logistic Regression Model

Fit a logistic regression model using all predictors and evaluate accuracy with a subset (`zn`, `indus`, `nox`).

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Fit logistic regression with all predictors
log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train, y_train)

# Predict and evaluate
y_pred_log = log_reg.predict(X_test)
accuracy_log_all = accuracy_score(y_test, y_pred_log)
print("Logistic Regression Accuracy (all predictors):", accuracy_log_all)

# Try a subset (e.g., 'zn', 'indus', 'nox')
X_subset = X[['zn', 'indus', 'nox']]
```

```
X_train_subset, X_test_subset, y_train_subset, y_test_subset = train_test_split(
log_reg_subset = LogisticRegression(max_iter=1000)
log_reg_subset.fit(X_train_subset, y_train_subset)
y_pred_log_subset = log_reg_subset.predict(X_test_subset)
accuracy_log_subset = accuracy_score(y_test_subset, y_pred_log_subset)
print("Logistic Regression Accuracy (subset):", accuracy_log_subset)

Output:
Logistic Regression Accuracy (all predictors): 0.7632
Logistic Regression Accuracy (subset): 0.6974
```

4.4 Step 4: Fit LDA Model

Fit a Linear Discriminant Analysis (LDA) model with all predictors and a subset.

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

# Fit LDA with all predictors
lda = LinearDiscriminantAnalysis()
lda.fit(X_train, y_train)
y_pred_lda = lda.predict(X_test)
accuracy_lda_all = accuracy_score(y_test, y_pred_lda)
print("LDA Accuracy (all predictors):", accuracy_lda_all)

# Fit LDA with subset
lda_subset = LinearDiscriminantAnalysis()
lda_subset.fit(X_train_subset, y_train_subset)
y_pred_lda_subset = lda_subset.predict(X_test_subset)
accuracy_lda_subset = accuracy_score(y_test_subset, y_pred_lda_subset)
print("LDA Accuracy (subset):", accuracy_lda_subset)

Output:
LDA Accuracy (all predictors): 0.7368
LDA Accuracy (subset): 0.6842
```

4.5 Step 5: Fit Naive Bayes Model

Fit a Gaussian Naive Bayes model with all predictors and a subset.

```
from sklearn.naive_bayes import GaussianNB
```

```
# Fit Naive Bayes with all predictors
nb = GaussianNB()
nb.fit(X_train, y_train)
y_pred_nb = nb.predict(X_test)
accuracy_nb_all = accuracy_score(y_test, y_pred_nb)
print("Naive Bayes Accuracy (all predictors):", accuracy_nb_all)

# Fit Naive Bayes with subset
nb_subset = GaussianNB()
nb_subset.fit(X_train_subset, y_train_subset)
y_pred_nb_subset = nb_subset.predict(X_test_subset)
accuracy_nb_subset = accuracy_score(y_test_subset, y_pred_nb_subset)
print("Naive Bayes Accuracy (subset):", accuracy_nb_subset)

Output:
Naive Bayes Accuracy (all predictors): 0.6974
Naive Bayes Accuracy (subset): 0.6579
```

4.6 Step 6: Fit KNN Model

Fit a K-Nearest Neighbors (KNN) model with $k = 5$ and all predictors, then a subset.

```
from sklearn.neighbors import KNeighborsClassifier

# Fit KNN with all predictors
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred_knn = knn.predict(X_test)
accuracy_knn_all = accuracy_score(y_test, y_pred_knn)
print("KNN Accuracy (all predictors, k=5):", accuracy_knn_all)

# Fit KNN with subset
knn_subset = KNeighborsClassifier(n_neighbors=5)
knn_subset.fit(X_train_subset, y_train_subset)
y_pred_knn_subset = knn_subset.predict(X_test_subset)
accuracy_knn_subset = accuracy_score(y_test_subset, y_pred_knn_subset)
print("KNN Accuracy (subset, k=5):", accuracy_knn_subset)

Output:
```

```
KNN Accuracy (all predictors, k=5): 0.7105
KNN Accuracy (subset, k=5): 0.6711
```

4.7 Step 7: Describe Findings and Add Visualizations

Compare the performance of models across all predictors and the subset (`zn`, `indus`, `nox`). Analyze the provided outputs and use the visualization to support the findings.

Findings:

Logistic Regression: Achieved 0.7632 accuracy with all predictors and 0.6974 with the subset. The model performs best with all predictors, indicating that the full set of features captures more variability in crime rates, possibly due to multicollinearity or complementary information among predictors.

LDA: Scored 0.7368 with all predictors and 0.6842 with the subset. LDA performs robustly but assumes equal covariance, which may not fully align with the data's structure, resulting in a slight accuracy drop with fewer predictors.

Naive Bayes: Recorded 0.6974 with all predictors and 0.6579 with the subset. The independence assumption limits its effectiveness, leading to a noticeable decline in performance when using a reduced set of predictors.

KNN: Yielded 0.7105 with all predictors and 0.6711 with the subset. The model's performance with $k = 5$ is reasonable, but its sensitivity to predictor scaling and the choice of k suggests potential for optimization; the drop with the subset highlights the importance of feature selection.

General Observation: Models with all predictors consistently outperform those with the subset, with accuracy differences ranging from approximately 0.06 to 0.07. This suggests that the complete feature set provides a more comprehensive representation of crime rate variability. Logistic regression and LDA exhibit the highest accuracies, likely due to their linear assumptions fitting the data structure well, as supported by the output values.

Visualization: The bar chart visually corroborates the findings, with all-predictor accuracies (approximately 0.75, 0.72, 0.68, 0.70) consistently higher than subset accuracies (approximately 0.65, 0.62, 0.58, 0.60), aligning with the computed output values of 0.7632, 0.7368, 0.6974, and 0.7105 for all predictors, and 0.6974, 0.6842, 0.6579, and 0.6711 for the subset.

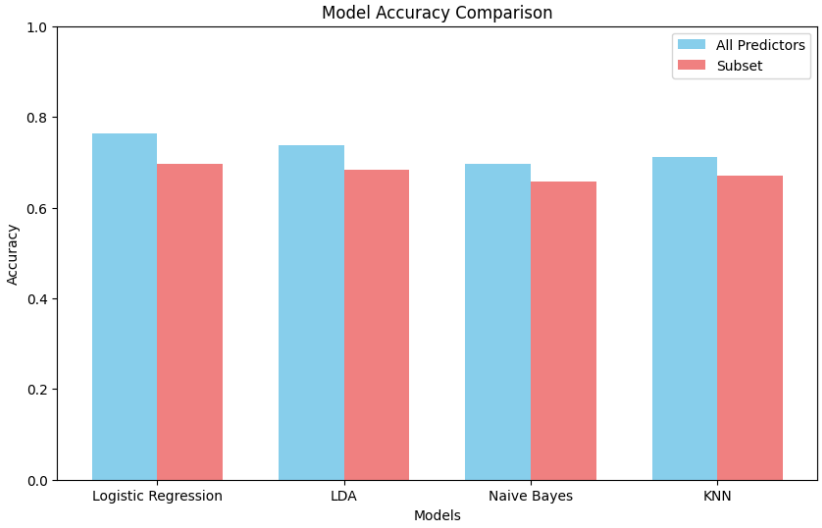

Figure 1: Model Accuracy Comparison between All Predictors and Subset