Introduction to Statistical Learning with Python

Chapter 2 Exercise Solutions

Questions 1 & 10: Conceptual Analysis and Boston Housing Dataset

Group 1

Group Members:

Shema Hugor
Godfrey
Kevin
Jean paul
Jean Claude

Course: Data Mining
Institution: American University of Central Asia
Department: Computer Science
Date: June 12, 2025

Contents

Abstract

This report presents comprehensive solutions to Chapter 2 exercises from "Introduction to Statistical Learning with Python" (ISLP). We analyze the conceptual differences between flexible and inflexible statistical learning methods, and conduct an extensive exploratory data analysis of the Boston housing dataset. Our analysis demonstrates practical applications of statistical learning concepts and provides insights into real-world data mining challenges. The solutions include both theoretical explanations and Python implementations, showcasing the relationship between statistical theory and practical data science applications.

1 Introduction

Statistical learning forms the foundation of modern data mining and machine learning applications. This report addresses two fundamental exercises from Chapter 2 of ISLP: a conceptual analysis of method selection criteria and a practical data exploration exercise using the Boston housing dataset.

The first question examines when to choose flexible versus inflexible statistical learning methods based on data characteristics such as sample size, dimensionality, relationship complexity, and noise levels. The second question involves comprehensive exploratory data analysis of a real-world dataset, demonstrating practical data mining techniques.

2 Question 1: Conceptual Analysis - Flexible vs Inflexible Methods

2.1 Problem Statement

For each of the following scenarios, we must determine whether a flexible or inflexible statistical learning method would generally perform better:

(a) The sample size n is extremely large, and the number of predictors p is small

(b) The number of predictors p is extremely large, and the number of observations n is small

(c) The relationship between the predictors and response is highly non-linear

(d) The variance of the error terms, i.e., $\sigma^2 = \text{Var}(\epsilon)$, is extremely high

2.2 Theoretical Background

The choice between flexible and inflexible methods involves the fundamental bias-variance tradeoff in statistical learning:

$$\text{Expected Test Error} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error} \tag{1}$$

- **Flexible methods** have low bias but high variance
- **Inflexible methods** have high bias but low variance

2.3 Solutions and Justifications

2.3.1 Scenario (a): Large n, Small p

Answer: Flexible Method Better

Justification:

- With extremely large sample size and few predictors, we have sufficient data to train complex models without overfitting
- The abundance of data helps reduce the variance of flexible methods
- Flexible methods can capture underlying patterns that simpler models might miss
- The curse of dimensionality is not a concern with small p
- Risk of overfitting is minimized due to large training set

2.3.2 Scenario (b): Large p, Small n

Answer: Inflexible Method Better
Justification:

- This represents the classic "curse of dimensionality" scenario
- With many predictors and few observations, flexible methods will severely overfit
- The ratio p/n is large, making it difficult to estimate complex relationships reliably
- Simpler, more constrained models will generalize better to new data
- Regularization techniques (which add inflexibility) become crucial

2.3.3 Scenario (c): Highly Non-linear Relationship

Answer: Flexible Method Better
Justification:

- Inflexible methods like linear regression assume specific functional forms
- Linear models cannot adequately capture complex non-linear patterns
- Flexible methods can adapt their form to match the underlying relationship
- The bias of inflexible methods would be very high for non-linear relationships
- Methods like decision trees, neural networks, or kernel methods excel here

2.3.4 Scenario (d): High Error Variance

Answer: Inflexible Method Better
Justification:

- High noise (σ^2) means low signal-to-noise ratio
- Flexible methods will attempt to fit the noise, leading to overfitting
- Simpler models are more robust to high variance in the data
- The irreducible error is already high, so minimizing model variance becomes crucial
- Regularization helps prevent the model from fitting random fluctuations

3 Question 10: Boston Housing Dataset Analysis

3.1 Dataset Overview

The Boston housing dataset contains information about housing values in suburbs of Boston, collected in the 1970s. It serves as a classic benchmark dataset for regression analysis and exploratory data analysis.

3.2 Data Loading and Exploration

3.2.1 Part (a): Loading the Dataset

Listing 1: Loading the Boston Dataset

```python
from sklearn.datasets import load_boston
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load Boston dataset
boston = load_boston()
df = pd.DataFrame(boston.data, columns=boston.feature_names)
df['target'] = boston.target

print("Dataset loaded successfully!")
```

3.2.2 Part (b): Dataset Dimensions and Structure

The Boston housing dataset contains:

- **Rows (observations):** 506 Boston suburbs/towns
- **Columns:** 14 total (13 predictors + 1 target variable)
- **Target variable:** Median value of owner-occupied homes in $1000s

Table 1: Boston Housing Dataset Variables

Variable	Description
CRIM	Per capita crime rate by town
ZN	Proportion of residential land zoned for lots over 25,000 sq.ft
INDUS	Proportion of non-retail business acres per town
CHAS	Charles River dummy variable (1 if bounds river; 0 otherwise)
NOX	Nitric oxides concentration (parts per 10 million)
RM	Average number of rooms per dwelling
AGE	Proportion of owner-occupied units built prior to 1940
DIS	Weighted distances to employment centres
RAD	Index of accessibility to radial highways
TAX	Full-value property-tax rate per $10,000
PTRATIO	Pupil-teacher ratio by town
B	1000(Bk - 0.63)^2 where Bk is proportion of blacks by town
LSTAT	Percentage of lower status of the population
target	Median value of owner-occupied homes in $1000s

3.2.3 Part (c): Pairwise Scatterplots Analysis

Listing 2: Creating Pairwise Scatterplots

```python
# Create correlation matrix
correlation_matrix = df.corr()

# Create pairwise scatterplot matrix for subset of variables
pd.plotting.scatter_matrix(df.iloc[:, :6], figsize=(15, 12), alpha=0.6)
plt.suptitle('Pairwise Scatterplots - Boston Housing Variables')
plt.tight_layout()
plt.show()

# Correlation heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
            center=0, square=True, fmt='.2f')
plt.title('Correlation Matrix of All Variables')
plt.show()
```

Key Findings from Pairwise Analysis:

- Strong positive correlation between RAD and TAX (highway access and taxes)
- Strong negative correlation between DIS and NOX (distance to employment and pollution)
- Moderate correlations suggest multicollinearity among some predictors
- Clear relationships between socioeconomic variables and housing values

3.2.4 Part (d): Crime Rate Associations

Listing 3: Crime Rate Correlation Analysis

```python
# Calculate correlations with crime rate
crime_correlations = df.corr()['CRIM'].abs().sort_values(ascending=
    False)

print("Correlations with CRIM (absolute values):")
for var, corr in crime_correlations.items():
    if var != 'CRIM':
        print(f"{var}: {corr:.3f}")
```

Strongest Relationships with Crime Rate:

- **RAD (0.625):** Highway accessibility correlates with higher crime
- **TAX (0.583):** Higher property taxes correlate with higher crime
- **INDUS (0.406):** More industrial areas have higher crime rates
- **NOX (0.421):** More polluted areas have higher crime rates

Interpretation: Urban, accessible, industrial areas with higher taxes tend to have more crime, suggesting crime is associated with urban density and accessibility.

3.2.5 Part (e): High Values and Range Analysis

Listing 4: Range and Outlier Analysis

```python
# Descriptive statistics
print(df.describe())

# High value analysis
high_crime = df[df['CRIM'] > df['CRIM'].quantile(0.95)]
high_tax = df[df['TAX'] > df['TAX'].quantile(0.95)]
high_ptratio = df[df['PTRATIO'] > df['PTRATIO'].quantile(0.95)]

print(f"High crime suburbs (top 5%): {len(high_crime)}")
print(f"High tax suburbs (top 5%): {len(high_tax)}")
print(f"High pupil-teacher ratio suburbs (top 5%): {len(high_ptratio)}"
    )
```

Key Observations:

- **Crime rate range:** 0.006 to 88.98 (extreme variation)
- **Tax rate range:** 187 to 711 per $10,000
- **Pupil-teacher ratio range:** 12.6 to 22.0
- Some suburbs show extremely high values, indicating significant inequality

3.2.6 Part (f): Charles River Analysis

Listing 5: Charles River Proximity Analysis

```python
charles_river_suburbs = df[df['CHAS'] == 1]
print(f"Suburbs bounding Charles River: {len(charles_river_suburbs)}")
print(f"Percentage: {len(charles_river_suburbs)/len(df)*100:.1f}%")
```

Result: 35 suburbs (6.9%) bound the Charles River, indicating that waterfront access is relatively rare and likely premium.

3.2.7 Part (g): Pupil-Teacher Ratio

Listing 6: Educational Resource Analysis

```python
median_ptratio = df['PTRATIO'].median()
print(f"Median pupil-teacher ratio: {median_ptratio}")
```

Result: Median pupil-teacher ratio is 19.05, indicating potential educational resource constraints across Boston suburbs.

3.2.8 Part (h): Lowest Home Value Analysis

Listing 7: Disadvantaged Area Analysis

```python
lowest_value_idx = df['target'].idxmin()
lowest_value_suburb = df.loc[lowest_value_idx]

print(f"Lowest median home value: ${lowest_value_suburb['target']:.1f}k
    ")
print("Characteristics of this suburb:")
```

```python

for col in df.columns:
    if col != 'target':
        percentile = (df[col] <= lowest_value_suburb[col]).mean() * 100
        print(f"{col}: {lowest_value_suburb[col]:.3f} "
              f"({percentile:.1f}th percentile)")
```

Characteristics of Lowest Value Suburb ($5,000 median):

- Crime rate in top 1% (extremely high)
- Very high industrial proportion
- High pollution levels (NOX)
- Poor pupil-teacher ratio
- High percentage of lower-status population
- Represents a severely disadvantaged area with multiple challenges

3.2.9 Part (i): Room Count Analysis

Listing 8: Housing Size Distribution Analysis

```python
more_than_7_rooms = df[df['RM'] > 7]
more_than_8_rooms = df[df['RM'] > 8]

print(f"Suburbs with >7 rooms per dwelling: {len(more_than_7_rooms)}")
print(f"Suburbs with >8 rooms per dwelling: {len(more_than_8_rooms)}")

if len(more_than_8_rooms) > 0:
    print("\nCharacteristics of suburbs with >8 rooms:")
    print(more_than_8_rooms.describe())
```

Housing Size Analysis:

- **64 suburbs** average more than 7 rooms per dwelling
- **13 suburbs** average more than 8 rooms per dwelling
- Large homes represent luxury/affluent areas (top 2.6%)

Characteristics of ¿8 Room Suburbs:

- Much lower crime rates than average
- Lower tax rates (affluent areas)
- Better pupil-teacher ratios (better educational resources)
- Much higher median home values
- Lower percentage of lower-status population
- Clear indicators of affluent neighborhoods

4 Statistical Learning Insights

4.1 Bias-Variance Tradeoff Applications

The conceptual analysis in Question 1 demonstrates practical applications of the bias-variance tradeoff:

- **Data abundance** enables complex modeling without overfitting
- **High dimensionality** requires regularization and simpler models
- **Non-linear relationships** demand flexible methods despite higher variance
- **High noise** necessitates bias toward simpler, more stable models

4.2 Real-World Data Mining Lessons

The Boston housing analysis reveals important data mining principles:

- **Exploratory analysis** is crucial for understanding data structure
- **Correlation analysis** reveals important feature relationships
- **Outlier detection** identifies interesting cases and data quality issues
- **Domain knowledge** enhances interpretation of statistical findings

5 Conclusions

5.1 Method Selection Guidelines

Based on our analysis, we propose the following guidelines for choosing between flexible and inflexible methods:

1. **Assess the data regime:** Consider the ratio of observations to predictors
2. **Evaluate relationship complexity:** Use domain knowledge and exploratory analysis
3. **Consider noise levels:** High variance data requires more conservative approaches
4. **Account for interpretability needs:** Balance performance with explainability

5.2 Data Mining Applications

The Boston housing analysis demonstrates several real-world applications:

- **Urban planning:** Identifying areas requiring intervention or investment
- **Real estate modeling:** Predicting property values using multiple factors
- **Policy analysis:** Understanding relationships between public services and outcomes
- **Social equity research:** Examining patterns of advantage and disadvantage

5.3 Future Research Directions

This analysis suggests several areas for further investigation:

- Developing robust methods for high-dimensional, small-sample scenarios
- Creating adaptive algorithms that adjust flexibility based on data characteristics
- Investigating temporal changes in housing patterns and their predictors
- Applying modern machine learning techniques to update this classic analysis

6 References

1. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An Introduction to Statistical Learning with Applications in Python.* Springer.
2. Harrison, D., & Rubinfeld, D. L. (1978). Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management*, 5(1), 81-102.
3. Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer.

A Complete Python Implementation

Listing 9: Complete Implementation Code

```python
# Complete implementation for ISLP Chapter 2 solutions
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_boston
import warnings
warnings.filterwarnings('ignore')

# Load and prepare data
boston = load_boston()
df = pd.DataFrame(boston.data, columns=boston.feature_names)
df['target'] = boston.target

# Basic information
print(f"Dataset shape: {df.shape}")
print(f"Columns: {list(df.columns)}")

# Correlation analysis
correlation_matrix = df.corr()
crime_correlations = correlation_matrix['CRIM'].abs().sort_values(
    ascending=False)

# Charles River analysis
charles_river_count = (df['CHAS'] == 1).sum()

# Median pupil-teacher ratio
median_ptratio = df['PTRATIO'].median()

```

```
# Lowest home value analysis
lowest_idx = df['target'].idxmin()
lowest_suburb = df.loc[lowest_idx]

# Room analysis
rooms_7_plus = (df['RM'] > 7).sum()
rooms_8_plus = (df['RM'] > 8).sum()

# Summary statistics
summary_stats = df.describe()

print("Analysis complete!")
```