Homework - Machine Learning

Product Performance Analysis

Overview

In this assignment, you will analyze supermarket product sales data using machine learning techniques. You will implement **K-means clustering** to discover product groupings and use **regression** models to predict product performance. This project emphasizes understanding machine learning concepts, proper data preprocessing, model comparison, and clear visualization of results.

Main Objectives

- Understand and implement K-means clustering algorithm from scratch
- Apply regression techniques to predict continuous outcomes
- Perform proper data preprocessing (handling missing values, normalization, outliers)
- Compare multiple models and interpret their performance
- Visualize and communicate results effectively
- Think from a user's perspective when presenting analytical results

You are strongly encouraged to use **Generative AI** tools (ChatGPT, Claude, GitHub Copilot, etc.) to help you understand concepts, debug code, and improve your implementation, such as:

- **Learning and Understanding**: Ask AI to explain concepts, algorithms, or debugging techniques
- **Code Generation**: Use AI to generate boilerplate code, UI components, or helper functions
- **Debugging Assistance**: Get help identifying and fixing bugs in your code
- **Documentation**: Assist with writing clear README files and comments
- **Optimization**: Ask for suggestions on improving code efficiency or structure
- **Design Ideas**: Brainstorm UI/UX design approaches

However, you must thoroughly comprehend all the code you submit and demonstrate your comprehension in the report.

Important Notes

- **Start early**: Don't underestimate preprocessing complexity
- **Test incrementally**: Verify each component before moving to the next
- **Document as you go**: Don't leave report writing to the last minute
- **Ask for help**: Use office hours if you're stuck
- **Focus on understanding**: This is about learning, not just getting results
- **Use AI wisely**: It's a tool to help you learn faster, not do the work for you

Dataset: Product Sales Data

You will work with `product_sales.csv`, containing 200 product records with the following features:

- **product_id**: Unique product identifier
- **product_name**: Name of the product
- **category**: Product category (Dairy, Bakery, Produce, Meat, Beverages, Snacks)
- **price**: Retail price in dollars
- **cost**: Cost per unit in dollars
- **units_sold**: Number of units sold per month
- **promotion_frequency**: Number of promotions per month (0-4)
- **shelf_level**: Shelf position (1=bottom, 3=eye-level, 5=top)
- **profit**: Monthly profit in dollars (TARGET VARIABLE)

Data Quality Issues (Intentional):

The dataset contains realistic data quality problems that you must handle:

- **Missing values**: ~10-15 records have missing data
- **Outliers**: ~5-8 products with unusual values
- **Scale differences**: Features have different ranges requiring normalization
- **Inconsistencies**: Some data entry errors

Project Requirements

1. Data Preprocessing (25%)

Implement comprehensive data cleaning and preparation:

Required Tasks:

- Missing Value Analysis and Handling
    - Identify all missing values in the dataset
    - Decide on appropriate handling strategy for each case:
        - Drop records (if too many missing values)
        - Impute with mean/median/mode
        - Other reasonable approaches
    - Document your decisions and rationale
- Outlier Detection and Treatment
    - Detect outliers using appropriate methods (e.g., IQR, Z-score)
    - Decide how to handle outliers:
        - Remove them
        - Cap them at threshold
        - Keep them if justified
    - Explain your reasoning
- Feature Normalization/Standardization
    - Normalize or standardize numerical features
    - Choose appropriate method:
        - Min-Max Normalization (0-1 scaling)
        - Z-score Standardization (mean=0, std=1)
    - Explain why normalization is necessary for K-means

2. K-means Clustering Analysis (40%)

2.1 K-means Implementation

You must implement K-means algorithm from scratch. You may use NumPy for basic operations but not use `sklearn.cluster.KMeans.fit()` or similar built-in clustering functions.

Required Implementation:

1. Initialize centroids (random or K-means++)
2. Assign each point to nearest centroid
3. Update centroids based on cluster assignments
4. Repeat until convergence
5. Return cluster labels and final centroids

2.2 Elbow Method for Optimal K

- Run K-means for k = 2, 3, 4, 5, 6, 7, 8
- Calculate WCSS (Within-Cluster Sum of Squares) for each k
- Plot elbow curve
- Recommend optimal k value based on the curve

2.3 Cluster Analysis and Interpretation

For your chosen k value:

- Run K-means clustering
- Calculate statistics for each cluster:
    - Number of products
    - Average price
    - Average units sold
    - Average profit

- Average profit
    - Average promotion frequency
- **Name each cluster** based on characteristics:
    - Example: "Budget Best-Sellers", "Premium Low-Volume", "Mid-Range Steady"
- Provide business insights for each cluster, e.g.,

```
Cluster 0: "Budget Best-Sellers"
- Products: 65
- Avg Price: $2.15
- Avg Units Sold: 780
- Avg Profit: $650
- Characteristics: Low-price, high-volume products
- Business Insight: Focus on maintaining stock levels

Cluster 1: "Premium Low-Volume"
- Products: 48
- Avg Price: $8.99
- Avg Units Sold: 120
- Avg Profit: $580
- Characteristics: High-price, specialty items
- Business Insight: Consider targeted promotions
```

2.4 Visualization

Create clear visualizations:

- **Elbow curve**: k vs WCSS
- **Cluster scatter plot**: 2D plot showing clusters
    - Use 2 most relevant features (e.g., price vs units_sold)
    - Different colors for different clusters
    - Mark centroids clearly
- Add titles, labels, and legends

3. Regression Analysis (35%)

Predict product **profit** or **units_sold** (your choice) using regression models.

3.1 Model Implementation and Training

You may use libraries like **scikit-learn** for regression.

Required tasks:

1. Split data into training and testing sets (70-30 or 80-20)
2. Implement at least **TWO different regression models**, such as:
    - Linear Regression
    - Polynomial Regression (you can define the degree based on your insight into the dataset.)
3. Train each model on training data
4. Make predictions on test data

Example Code Structure:

```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import train_test_split

# Prepare data
X = features  # price, cost, units_sold, etc.
y = target    # profit or units_sold

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)

# Model 1: Linear Regression
model1 = LinearRegression()
model1.fit(X_train, y_train)
predictions1 = model1.predict(X_test)

# Model 2: Polynomial Regression
poly = PolynomialFeatures(degree=2)
X_train_poly = poly.fit_transform(X_train)
```

```
X_train_poly = poly.fit_transform(X_train)
X_test_poly = poly.transform(X_test)
model2 = LinearRegression()
model2.fit(X_train_poly, y_train)
predictions2 = model2.predict(X_test_poly)
```

3.2 Model Comparison and Evaluation

Evaluate each model using:

- **MSE (Mean Squared Error)**: Average squared prediction error
- **MAE (Mean Absolute Error)**: Average absolute prediction error

Compare models:

- Which model performs best?
- Why do you think this model performs better?
- Are there signs of overfitting?
- What are the tradeoffs?

3.3 Visualization

Create visualizations for your best model:

- **Actual vs Predicted scatter plot**
  - X-axis: Actual values
  - Y-axis: Predicted values
  - Diagonal line showing perfect prediction
  - Points should cluster around this line
- **Regression line/curve** (if using 2D feature space)
- Optional (bonus): Residual plot

Visualization and User Interface

Basic Requirements (No separate points, but required):

Create a simple presentation of your results suitable for non-technical users. This can be:

- A simple web page (HTML/JavaScript)
- A Jupyter notebook with clear explanations
- A Python GUI (tkinter, streamlit, etc.)
- A simple dashboard

Must include:

1. **Data Overview Section**
    - Basic dataset statistics
    - Preprocessing summary
2. **Clustering Results Section**
    - Elbow curve
    - Cluster visualization
    - Cluster statistics table
    - Clear interpretation
3. **Regression Results Section**
    - Model comparison table
    - Actual vs Predicted plot
    - Performance metrics
    - Interpretation

Deliverables

Required Submissions:

1. Source Code

- Well-organized, commented code
- Clear file structure
- If not directly executable, include README.md with setup instructions

2. REPORT.pdf

Include the following sections:

- Introduction: Brief overview of the project
- Data Preprocessing
    - Missing value handling approach and justification
    - Outlier detection method and treatment
    - Normalization approach and reason
    - Preprocessing summary statistics
- K-means Clustering Analysis
    - Implementation approach
    - Elbow method results and optimal k selection
    - Cluster analysis with statistics
    - Cluster interpretation and naming
    - Business insights from clustering
- Regression Analysis
    - Models chosen and why
    - Training process
    - Performance comparison table

    - Best model selection and justification
    - Discussion of overfitting/underfitting

- Visualizations (embedded in sections above)
    - Elbow curve
    - Cluster scatter plot
    - Model comparison charts
    - Actual vs Predicted plot
- Conclusion
    - Key findings
    - Limitations of your analysis
    - Potential improvements
    - AI tool usage summary

G. References (if applicable)

- Any external resources used

Submission Format:

Option A: GitHub Repository (Preferred ⭐)

- Create a public or private repository
- Submit repository link via Canvas
- Well-organized repositories may receive bonus points (2-3 points)

Option B: ZIP Archive

- Compress your entire project folder
- Name: `LastName_FirstName_StudentID_ML_Assignment.zip`
- Upload to Canvas

Recommended Project Structure:

```
LastName_FirstName_StudentID_ML_Assignment/
├── source code/
│   ├── main.html
│   ├── preprocessing.xx
│   ├── kmeans.xx
│   ├── regression.xx
│   └── visualization.xx
├── data/
│   └── product_sales.csv
├── results/
│   └── (generated plots, if any)
├── README.md (only if code needs setup instructions)
└── REPORT.pdf
```

Final Notes

- Learning Philosophy: The goal is to learn artificial intelligence concepts and develop practical skills. Gen-AI is a tool that can accelerate your development and help you learn more effectively, but it's not a substitute for understanding. Think of AI as a highly knowledgeable teaching assistant available 24/7.
    - **Remember**: In professional software development, using AI tools is standard practice. Learning to use them effectively is a valuable skill.
- Using GitHub demonstrates professional development practices and is highly encouraged in the software industry
- Focus on making your application usable by non-technical users
- Test your application on the provided dataset - we will use it for grading
- Ensure your code runs on a standard environment (provide clear setup instructions) without any limitations on Operating Systems or Integrated Development Environments.
- The quality of your documentation (README and/or report) is part of your grade
- Bonus points (up to 20 points) are awarded for a rational add-on, such as
    - Performance comparison visualization
    - Advanced visualization (network graphs, etc.)
    - Additional UI/UX optimization
    - Other innovative features
    - Well-organized GitHub submission
    - etc.

Good luck with your assignment! This is an opportunity to build a complete machine learning application from scratch while learning to leverage modern AI development tools.