Amirkabir University of Technology
(Tehran Polytechnic)

Machine Learning Course By Dr. Nazerfard

CE5501 | Fall 2024

Teaching Assistants

AmirHossein Babaeayan (Head TA)

Ehsan Shobeiri

Ali Raei

Mohammadreza Kaviani Nia

Sara Firoozi Nia

Assignment (2)

Outlines. In this assignment, we will focus on Regression, KNN, and Classification Methods.

Deadline. Please submit your answers before the end of November 9th in courses.aut.ac.ir. Other methods like sending via email or in social networks are not accepted and will not be considered.

Assignment Manual

Delay policy. During the semester, you have extra 5 days for submitting your answers with delay. Mentioned time is for all assignments. After that, for each day of delay you loss 20% points of that assignment. After 4 days you miss all points and any submit doesn`t acceptable. Remember that saving this time doesn`t have any extra point.

Sharing is not caring. Students are free to discuss and share their ideas about problems with others. But sharing source codes, solutions, answers and other results is not allowed and based on university`s rule, both sides will be graded zero.

Problems are waiting you. Some problems are required to be implemented within a programming language and obtain some charts, images, results, etc; then discuss about it. These types of questions are tagged by #Implementation. Some other problems are required to be solved or computed by hand or research about them. These types of questions are tagged by #Theorical. You are not allowed to use programming language or other technical tools to answer theorical problems.

Report is the key. All students' explanations, solutions, results, discuss and answers must be compacted into a single pdf report. A clean and explicit report is expected and may followed by extra pts; so, you may need to write any related detail or experience during the solving problems. Report file should started within a cover page that it includes course and assignment information as well as identical details like name, student number and email address. Second page should be table of contents that indicates student`s answer to each question. Please repeat your name and student number in left side of footer in other pages. Also, you are free to write in Persian or English. If typing is bothering you, so write in a paper and put its picture with acceptable readability quality in report file.

Organize the upload items. Students should upload their implementation source codes as well as results and report. You should upload a single .zip file with the following structure:

```
ML_00_[std-number].zip
        Report
                ML_0_[std-number].pdf
                [other material and results]

        Source codes
                P[problem-number]_[a-z].py
                P[problem-number]_[a-z].ipynb
                …
```

Python is the power. Students are free to use any programming language like python, matlab, C++ , etc. However it is recommended strongly to use python in jupyter notebook environment; so, you may need to upload your .py or .ipynb sources.

Feel free to contact. If you have any question or suggestion, need guide or any comment be comfortable to ask via email as well as Telegram group. Good luck with your learning journey

Problem 1: Predicting House Price Using Stacked Regression

In today’s dynamic real estate market, accurately predicting the sale price of residential properties is a critical challenge for both homeowners and real estate professionals. The sale price of a house is influenced by numerous factors, including location, size, condition, and various economic indicators. With the ever-evolving housing market landscape, making informed decisions about buying or selling property has become increasingly complex. To address this challenge, we aim to leverage data science and machine learning techniques. This project will focus on developing a Stacked Regression model that utilizes the predictive capabilities of multiple regression algorithms.

Implementation Steps

1. **Load the dataset** and display 5 sample records.
2. **Identify and remove outliers** using z-score on relevant columns.
3. **Normalize the target column** to ensure a balanced distribution.
4. **Preprocess the dataset** to prepare it for modeling.
5. **Train models** including Lasso, Elastic Net, Kernel Ridge, and Gradient Boosting Regression. Optimize each model by finding the best parameters and learning rates. Briefly explain how the learning rate impacts model performance (using the sklearn library).
6. **Evaluate model performance** by reporting the Mean Squared Error (MSE) and R^2 on the test data for each model, and compare the results.
7. **Explain model stacking**: What is model stacking, and how does Stacked Regression work?
8. **Train a Stacked Regression model** (implementing Stacked Regression and using sklearn library for base models).
9. **Report the MSE and R^2** for the Stacked Regression model on test data, and compare these results with those from step 6.

Problem 2: Predicting Song Sales Using Machine Learning

Music plays a significant role in our digital age, with streaming and social media platforms driving song sales. Various factors, such as musical features, social trends, and marketing strategies, impact song sales. Machine learning offers a powerful tool for predicting song sales by analyzing these features, which can benefit music streaming services. The objective of this assignment is to utilize machine learning techniques to predict song sales based on musical features.

Implementation Steps

1. **Load the dataset** and identify categorical and numerical features by counting unique values in each feature.
2. **Analyze the distribution** of the target variable and numeric features using suitable charts.
3. **Outlier Detection:** Discuss the differences between descriptive and prescriptive outlier detection methods. Identify the suitable method for this problem and explain your choice. Use box plots to identify columns with outliers and remove them from the dataset.
4. **Feature Relationships:** Investigate relationships between all features using appropriate charts and provide detailed explanations.

5. **Multicollinearity Analysis:** Explain the concept of multicollinearity and discuss suitable techniques for handling it. Determine if multicollinearity exists in this dataset (without using charts from Step 4). If present, apply necessary preprocessing steps.
6. **Implement Linear Regression:** Implement linear regression from scratch and compare the results and runtime with scikit-learn’s linear regression.
7. **Lasso vs. Ridge Regression:** Discuss the differences between Lasso and Ridge regression. Implement both models and compare their performance with the previous linear regression results.
8. **Overfitting Analysis:** Explore the possibility of overfitting on the test or validation set. Discuss methods for detecting and preventing overfitting in this project.
9. **Target Intervals for Sales:** Consider using intervals for the target value, with each interval representing a specific category. Divide the target data into intervals of 10,000, where the range from 0 to 10,000 is labeled as "worst_seller" and 40,000 to 50,000 as "best_seller." Discuss if using these intervals allows for linear regression or if the problem is better suited for classification. Analyze potential challenges with this approach.
10. **Model Evaluation:** Ensure model generalization by evaluating its performance using appropriate metrics. Analyze errors and provide a detailed report.
11. **Suggestions for Improvement:** Propose a new idea to improve the performance of the discussed models (extra points).

Problem 3: Feature Selection and Classification Using Weighted K-Nearest Neighbors (KNN)

You have joined an insurance company with extensive customer data. Your initial task is to identify the most informative features within this dataset. In class, you learned about K-Nearest Neighbors (KNN) for classification, but KNN can also provide deeper insights into the data. In this problem, we will use KNN to learn weights for available features and then select those with the highest weights.

Distance Formulas

Euclidean Distance (Vanilla KNN):

In the standard KNN approach, we use Euclidean distance to measure similarity between data points. The Euclidean distance between two points $p = (p_1, p_2, ..., p_n)$ and $q = (q_1, q_2, ..., q_n)$ is calculated as follows:

$$d(p, q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + ... + (q_n - p_n)^2}$$

Weighted Distance (Weighted KNN):

In weighted KNN, each feature is assigned a corresponding weight w_i, which adjusts the distance calculation to give certain features more or less influence. The weighted distance between two points $A = (A_1, A_2, ..., A_n)$ and $B = (B_1, B_2, ..., B_n)$ is calculated as:

$$d(A, B) = \sqrt{\sum w_i * (A_i - B_i)^2}$$

Implementation Steps

1. Data Preparation:

- Load the dataset and perform necessary preprocessing steps for KNN. Specify and apply these steps as needed.
- Split the dataset into training and test sets in a 0.9:0.1 ratio.

2. Weighted KNN Loss Calculation:

- Implement a function to calculate the loss of a weighted KNN. Follow these steps:
- For each observation in the dataset, calculate the weighted distance between it and all other observations (using pairwise_distances from scikit-learn is recommended). Use the weighted distance formula provided above.
- Use the calculated distances to select the k nearest neighbors for each observation and predict the label.
- Compute a performance metric (e.g., accuracy, AUC, or Log Loss) based on predictions. Return the result without using pre-built functions for metric calculations.
- Function Inputs and Outputs:
- This function should accept the weights, dataset, and k as inputs and output the calculated loss.

3. Weight Optimization:

- Initialize weights to zero and optimize the function from Step 2 to find the best weights that minimize KNN classification loss (using scipy.optimize.minimize for optimization is recommended).

4. Weighted and Vanilla KNN Classification:

- Implement the weighted KNN classifier using the optimized weights from Step 3 to predict labels for the test set.
- Additionally, classify the test set using vanilla KNN (weighted KNN with all weights set to one) and compare the results.

5. Dimensionality Reduction Using Random Feature Subsets:

- Since KNN performance is affected by the curse of dimensionality, reducing the number of features can improve results.
- Select 8 random subsets of features, each containing 5 features, and use them for classification. Report the results.

6. Feature Selection Based on Weights:

- Select the 5 features with the highest weights from Step 3 and use these to classify the test set. Report the classification results.

7. Comparison and Analysis:

- Compare results from random feature selection, no feature selection, and weighted feature selection. Provide a justification for the observed results.