P556 Homework 1

Fall 2021

Due on Sep 13th, 11:59pm

Question 1: Regression Sydney Dataset (20 points)

You can load the Sydney dataset from https://www.kaggle.com/shree1992/housedata where you can also find a description. The goal is to predict the 'price' column. For this task, you can ignore the date.

- Determine which features are continuous vs. categorical. Drop rows without a valid sales price.
- Visualize the univariate distribution of each continuous variable, and the distribution of the target. Do you notice anything? Is there something that might require special treatment?
- Visualize the dependency of the target on each continuous feature (2d scatter plot).
- Split the data in training and testing set. Use ColumnTransformer to encode categorical variables. Impute missing values using SimpleImputer. Evaluate Linear Regression (OLS), Ridge, Lasso and Elasticnet (although we haven't talked about these methods yet, but you can easily find references online and you can use provided functions by Scikit-learn or other packages directly) using cross-validation with the default parameters. Dose scaling the data with StandardScaler help? Use the preprocessing that works best going forward.

Question 2: Classification on the 'credit-g' dataset (20 points)

You can download the dataset with 'fetch_openml('credit-g')' and see its description at https://www.openml.org/d/31

- Determine which features are continuous and which ones are categorical.
- Visualize the univariate distribution of each continuous variable, and the distribution of the target.
- Split the data in training and testing set. Preprocess the data (such as treatment of categorical variables) and evaluate and initial Logistic Regression model (directly use the provided function) with with a training/validation split.
- Use ColumnTransformer to encode categorical variables. Evaluate Logistic Regression, Linear Support Vector Machines and nearest neighbors (You can directly call these functions). How different are the results? How dose scaling the continuous features with StandardScaler influence the results?

Question 3: Binary Classification (30 points)

Consider the binary classification problem of mapping a given input to two classes. Let $X = \mathbb{R}^d$ and Y =-1,+1 be the input space and output space, respectively. In simple words, it means that the input has d features and all of them are real valued, whereas the output can only take values -1,+1. This is one of the most common problems in machine learning and many sophisticated methods exist to solve it.

- In the question, we will solve it using the concepts we have already learned in class. Let us assume the two sets of points can be separated using a straight line i.e. the samples are linearly separable. So if $d = 2$, one should be able to draw a line to distinguish between the two classes. All points lying on side of the line should belong to a particular class (say 1) and the points lying on the other side should belong to another class (say 2). To see what this would look like, your first task is as follows:

Write a function that will randomly generate a dataset for this problem. Your function should randomly choose a line l, which can be denoted as $ax + by + c = 0$ and the line divides the plane into two sides. On one side, $ax + by + c > 0$ while on the other $ax + by + c < 0$. Use this fact to randomly generate k_0 points on the side of class 0 (i.e. $y = -1$) and k_1 points on the side of class 1 (i.e. $y = 1$). Create a plot of this dataset where all the points corresponding to one class are blue and those of the other class are green, the line dividing both classes should be red. Axes should be labeled.

Note: Do not confuse the x and y in the equation of line $ax + by + c = 0$ with X and Y. Instead imagine these x and y as the $2D$ coordinate system on which you have different points which should lie on 2 sides of the line $ax + by + c = 0$. For example, there is a point $(2, 3)$ in the $2D$ system where $x = 2$ and $y = 3$.

- If Y is the variable you are trying to predict using a feature X then in a typical Machine Learning problem, you are tasked with a target function f which maps X to Y i.e. Find f such that $Y = f(X)$.

  When you are given a dataset for which you do not have access the target function f, you have to learn it from the data. In this problem, we are going to learn the parameters of the line that separates the two classes for the dataset that we constructed in problem above. As we previously mentioned, that line can be represented as $ax + by + c = 0$.

  The goal here is to correctly find out the coefficients a, b, and c, represented below as $\mathbf{w}$ which is a vector. The algorithm to find it is a simple iterative process:

  – Randomly choose a $\mathbf{w}$ to begin with.

  – Keep on adjusting the value of $\mathbf{w}$ as follows until all data samples are correctly classified:

    ∗ Randomly choose a sample from the dataset without replacement and see if it is correctly classified. If yes, move on to another sample.

    ∗ If not, then update the weights as $\mathbf{w}^{t+1} = \mathbf{w}^t + y\mathbf{x}$ and go back to the previous step (of randomly choosing a sample)

      · $\mathbf{w}^{t+1}$ is value of $\mathbf{w}$ at iteration $t + 1$

      · $\mathbf{w}^{t+1}$ is value of $\mathbf{w}$ at iteration t

      · y is the class label for the sample under consideration

      · $\mathbf{x}$ is the data-point under consideration

  Write a function that implements this learning algorithm. The input to the function is going to be a dataset represented by the input variable X and the target variable Y. The output of the function should be the chosen $\mathbf{w}$.

Question 4: K-Nearest Neighbor (30 points)

Implement the K-nearest neighbor algorithm and apply it to the Iris dataset https://archive.ics.uci.edu/ml/datasets/iris. Here $K = 3$.

Requirements:

- Split the data into training and testing (You can directly use the function provided by Python packages).

- In our class, we have the basic kNN algorithm based on nearest neighbor labels. In this implementation, you need to make it in an opposite way: the training data is labeled as $\mathcal{D}$ and start with $\mathcal{D}_k = \mathcal{D}$, during each iteration, remove the data point which has the farthest distance from the query point until $\mathcal{D}_k$ has k points left.

- Implement a heap to make it faster for the search.

Answer the following questions:

1) What is your precision on your test data?

2) What is the Big-O complexity for kNN if I adopt heap for the search?