Data Analysis and Machine Learning with Python Midterm Test

B09602017 白宗民

I. T, F, T, T, T

II. a, b, d, a, d

III. In the ipynb files.

IV.

1. Data cleaning is an essential step in the data analysis process because it directly impacts the **validity and reliability** of the results. Cleaning data can lead to more accurate models and analyses, enabling better decision-making and reducing the likelihood of skewed or biased outcomes.

a. Outlier detection and treatment: In financial data analysis, an extremely high transaction value that is significantly different from the rest could be an outlier. Detecting and addressing these outliers is important to prevent them from skewing averages or other statistical analyses. Also in my recent paper work for COLM 2024, we also do the outlier detection in our experiments.

b. Handling the missing data: In healthcare data, missing values for a patient's blood pressure readings could be imputed using the average of the available readings, assuming the missingness is random and not biased. (Ex: in out Titanic.csv)

c. Data Type Conversion: In a retail dataset, the 'Product_Code' column might be mistakenly treated as an integer type when it should be a categorical type, since mathematical operations on product codes don't make sense. So the data type is quite important to understand the dataset.

2. Overfitting is a common issue for lots of the experiments, it is that the neurons just memorize all the data without generalize(泛化) them. This would cause a lot of problems like never think, just remember it, as a result, the performance would be well on training dataset but poor at valid or testing dataset since the model has never seen the data. In my previous work, I generally **decrease the complexity of the model** (layers, dimension) to reduce the ability of the model to remember all the data, forcing them to generalize the data and learn it. And also do the **Dropout** and **Early Stop.** But I think the most important and fundamental issue is the **data quality (I have encountered a lot of these issues in my previous lab and internship).** The data imbalance could cause a lot of issues, one of them is overfitting.

3.

Decision Tree: A decision tree is a flowchart-like tree structure where an internal node represents a feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition based on the attribute value. It partitions the tree in a recursive manner called recursive partitioning. This flowchart-like structure helps you in decision making.

Advantages:

Simple to understand, interpret, and visualize.

Little data preparation is needed, and it can handle both numerical and categorical data.

The cost of using the tree for inference is logarithmic in the number of data points used to train the tree.

Disadvantages:

Prone to overfitting, especially if the tree is very deep.

Can be unstable because small variations in the data might result in a completely different tree being generated.

Decision tree learners create biased trees if some classes dominate.

Application Scenarios: Decision trees are suitable for classification tasks where transparency and interpretability are important, such as in finance for credit scoring or in medicine for diagnosing patients.

Random Forest: Random Forest is an ensemble learning method for classification and regression that works by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

Advantages:

Reduces the risk of overfitting by averaging multiple trees.

Can handle a large dataset with higher dimensionality.

They automatically handle missing values and maintain accuracy for a large proportion of data missing.

Disadvantages:

Model interpretability: Random forest models are not all that interpretable; they are like black boxes.

For very large data sets, the size of the trees can take up a lot of memory.

It can take a long time to train as it combines a lot of decision trees to determine the class.

Application Scenarios: Random Forest is suitable for situations where performance and accuracy are more critical than understanding the model's inner workings. It can be used in e-commerce to recommend products to users or in finance to assess the likelihood of customers defaulting on loans.

4.

Matplotlib:

Pros: Highly customizable, powerful, and can be used to create almost any type of static, animated, or interactive visualization.

Cons: Can be verbose for complex visualizations, steep learning curve for beginners, not the best for highly interactive visualizations.

Usage Methods: Matplotlib is best when you need to make customized plots or when you are building a project that requires the plotting of data in various complex ways.

Seaborn:

Pros: Built on top of Matplotlib and provides a high-level interface for drawing attractive and informative statistical graphics. Easier to generate complex plots with less code.

Cons: Less customizable than Matplotlib since it's a higher-level API, and it's not intended for interactive graphics or animations.

Usage Methods: Seaborn is better suited for exploratory data analysis and for making standard statistical visualizations quickly with its beautiful default styles.

5. It is **not typically appropriate** to use random forest regression or decision tree regression for anomaly detection. Random Forest and Decision Tree Regression are **supervised** learning methods which mean they **require labeled data** to train the model. They work well for prediction tasks when the target variable is continuous (regression).

While Anomaly Detection is often an **unsupervised learning problem** because you **don't have labels that indicate whether a data point is an anomaly.** The goal is to identify rare items, events, or observations which raise suspicions by differing significantly from the majority of the data.es.

Anomaly detection requires algorithms that are tailored to **detecting items that are unusual and do not fit well with the pattern or distribution of the majority of the data.** While decision trees and random forests **could theoretically be adapted for anomaly detection, it's not their strength or common use case.**