Kidney Disease Prediction Analysis

Introduction

This document details the steps and methods used to preprocess and analyze a dataset aimed at predicting kidney disease. We explore data cleaning, exploratory data analysis (EDA), and machine learning model evaluation.

Importing Libraries

The analysis begins by importing essential Python libraries:

- **NumPy** and **Pandas**: For data manipulation and analysis.
- **Matplotlib** and **Seaborn**: For data visualization.
- **Warnings**: To suppress unnecessary warnings during execution.

Importing the Dataset

The dataset was loaded using `pandas.read_csv()` which reads the dataset from a specified path. The initial view of the dataset is displayed to understand its structure and the first few rows.

Data Understanding and Cleaning

Dataset Overview

The shape of the dataset and the list of its columns was checked to understand its dimensions and features. The `info()` method provides details on each column's data type and non-null counts.

Data Type Conversion

Certain columns with numeric values might be read as strings. These columns were converted to numeric types using `pd.to_numeric()`, handling any errors by coercing them into `NaN`.

Identifying Categorical and Numerical Columns

The columns were categorize into categorical and numerical based on their data types. This helps in applying appropriate preprocessing techniques.

Missing Values

The percentage of missing values in each column was computed to understand the extent of data completeness. This helps in deciding how to handle missing data.

Handling Missing Values in Categorical Variables

Missing values in categorical variables was handle by:

- **Cleaning**: inconsistent entries were replaced with standardized values.
- **Imputation**: missing values was filled with the most frequent value (mode) in each column.

Handling Missing Values in Numerical Columns

The distributions of numerical columns and handle missing values was analyze using median imputation for skewed features and mean imputation for normally distributed features.

Exploratory Data Analysis (EDA)

Target Feature Distribution
The distribution of the target variable (kidney disease classification) was examined to understand the class balance.

Visualization

- **Bar Plots**: Used to visualize the distribution of categorical variables.
- **Box Plots**: Used to identify outliers and understand the spread of numerical variables.

Preprocessing for Machine Learning

Encoding Categorical Variables
Categorical variables are converted to binary values to facilitate machine learning algorithms. This involves mapping categorical values to numerical representations.

Saving Cleaned Data
The cleaned and preprocessed dataset is saved as a new CSV file for future use.

Machine Learning Algorithms

Importing Libraries
Essential libraries for machine learning are imported:

- **Train-Test Split**: To split the dataset into training and test sets.
- **Standard Scaler**: To normalize feature values.
- **Classifiers**: Including Support Vector Classifier (SVC), K-Nearest Neighbors (KNN), and Random Forest.

Data Preparation
The dataset is split into features (X) and target labels (y). Features are standardized using `StandardScaler()`. The data is then split into training and test sets.

Model Initialization
Three different classifiers are initialized for comparison:

- **Support Vector Classifier (SVC)**
- **K-Nearest Neighbors (KNN)**
- **Random Forest Classifier**

Model Training
Each model is trained on the training set to learn from the data.

Model Evaluation

Models are evaluated using accuracy scores, which measure the proportion of correctly predicted instances. The accuracy of each model is compared to determine the best performer.

Enhanced Elitism for Strategy Generation Algorithm

Importing Libraries
Libraries for genetic algorithms and model evaluation are imported:
- **DEAP**: For evolutionary algorithms.
- **Cross-Validation**: For model evaluation.

Data Preparation
The dataset is scaled and split into training and test sets.

Genetic Algorithm Components
A genetic algorithm is used to optimize the feature selection process. It involves:
- **Creating Individuals**: Representing potential solutions.
- **Evaluation Function**: Measures performance using cross-validation.
- **Genetic Operations**: Includes crossover and mutation to explore different solutions.

Running the Genetic Algorithm
The algorithm iterates through generations, evolving the population of solutions to find the best feature subset for model training.

Final Model Evaluation
The best feature subset identified by the genetic algorithm is used to train a Support Vector Classifier. The model's accuracy on the test set is reported to evaluate its performance.