2024

Comparing the performance of two implementations of identically named machine learning techniques across different datasets

MACHINE LEARNING

ISAAC UMUKORO

NCIRL | ix23215640@student.ncirl.ie

Table of Contents

1. Abstract

This project aims to compare the performance of the implementations of two identically named machine-learning techniques: K-nearest neighbors (KNN) and Decision Tree Classifiers across different software libraries. The research uses three datasets; Breast Cancer, Wine and Heart Disease, to understand whether different implementations of identically named Machine learning techniques perform similarly or differently and which implementation performs better than the others in each dataset.

Each of the datasets used in this study presents distinct difficulties. The Breast cancer and the Heart Disease dataset are binary classification problems while the Wine dataset is a multiple-category classification problem. The research employs a consistent empirical design, feature selection, dimensionality reduction via PCA, cross-validation, and hyperparameter tuning to assess the model techniques.

The findings highlight the variability in model performance between implementations and emphasize the importance of careful optimization. The hyperparameter tuning, feature selection, and consistent evaluation metrics significantly affect classification accuracy. Our research shows that well-chosen preprocessing and model optimization steps can improve machine learning methods, enabling data scientists to optimize solutions.

Keywords: k-Nearest Neighbors, Decision Tree, Breast Cancer Dataset, Diabetes Dataset, Wine Dataset, Cross-Validation, PCA, Feature Selection.

2. Introduction

In machine learning, techniques often share names across different software libraries and frameworks, but their specific implementation can differ remarkably. These differences can lead to changes in performance, especially when applied to different datasets. The main objective of this work is to compare the performance of two different implementations of two commonly used machine learning techniques: k-Nearest Neighbors (kNN) (Cover, 1967) and Decision Tree Classifiers (page, 2024) using three different datasets which include: Breast cancer, Wine and Heart Disease - acquired from Sciki-learn's library (Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, et al, 2011) and UCI repository (Repository, n.d.). The purpose of this study is to demonstrate how these changes affect the effectiveness of the model.

The three selected datasets exhibit a range of challenges that provide an extensive testing ground for each implementation to adapt to different data structures and complexities. The Breast cancer data set is a binary classification problem focusing on the detection of malignant tumours. The Diabetes Dataset is also a binary classification problem, and it indicates whether a patient has a diabetes risk above or below the median threshold. The Wine Dataset is a multiple-category classification problem, where different types of wines are categorized based on their chemical composition.

The goal is to unravel insights into the differences in performance between these implementations, which can help data scientists understand how software choices and dataset characteristics interact. K-Nearest Neighbor (KNN) classifier (Cunningham, P., & Delany, S. J., 2020) is well-known for its simplicity and versatility, and it relies on distance metrics to classify data based on its nearest neighbors. The Decision Tree classifier (Quinlan, 1986) uses a tree-like model to predict the target variable through a series of decision rules. The comparison of the various implementations of these two techniques aims to distinguish key factors that can impact their accuracy and efficiency.

This work also offers valuable insights into the strengths and limitations of different implementations of identically named Machine-learning techniques. It will help practitioners select the optimal solution for their specific needs and datasets, thereby improving decision-making and efficient use of machine learning the findings also provide useful suggestions for data preprocessing, hyperparameter tuning, and model selection for similar classification tasks. Finally, this report intends to contribute to a clearer understanding of the nuances of using kNN and Decision Tree models in different libraries and data types.

3. Research Question

In this project, we will examine the following research questions:

- Do different implementations of identically named machine learning techniques perform exactly the same? If not, what are the outstanding implementations of these identically named techniques for specific empirical designs and evaluation measures? Specifically, which implementation performs best for each dataset?
- How do the datasets employed in this study differ from each other? What are the differences in characteristics between the three datasets, and how might these differences impact the effectiveness of machine learning techniques?

4. Literature Review

In a preliminary review of the relevant literature, several papers were identified that contribute to understanding of the performance of the implementation of two identically Named Machine-learning techniques. These papers provide valuable insights into the methodologies, algorithms, and techniques employed in this research. Below is a brief review of the selected papers:

Cover, T., & Hart, P. (1967). "Nearest neighbor pattern classification." This influential paper laid the theoretical groundwork for the kNN algorithm. The authors presented how the nearest neighbor pattern classification works and demonstrated its robustness. They showed the effectiveness of the algorithm in pattern recognition, setting the basis for further advancements in nearest neighbor classification.

Shakhnarovich, G., Darrell, T., & Indyk, P. (Eds.) (2006). "Nearest-Neighbor Methods in Learning and Vision: Theory and Practice." This book compares various advancements and implementations of nearest-neighbor methods, offering insight into improvements in computational efficiency and accuracy. It also provides practical guidance for applying kNN to machine learning and computer vision tasks.

Cunningham, P., & Delany, S. J. “K-Nearest Neighbour Classifiers: 2nd Edition.” This paper provides detailed examples and discussions on kNN classifiers, including KDTree making it a comprehensive guide to kNN implementation.

Quinlan, J. R. (1986). "Induction of decision trees." This foundational work introduced the ID3 (Iterative Dichotomiser 3) algorithm, which uses entropy and information gain to create decision trees. Quinlan’s paper is pivotal for understanding tree-based machine learning methodologies.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). "The elements of statistical learning." The book provides a comprehensive comparison of different decision tree methodologies, including CART,

Random Forests, and Gradient Boosting. It explores the strengths and weaknesses of various implementations, offering readers guidance on the appropriate use of each method.

Chen, T., & Guestrin, C. (2016). "XGBoost: A scalable tree boosting system." This paper presents XGBoost, a highly efficient and scalable gradient-boosting library. The authors discuss the implementation and optimizations that enable XGBoost to outperform other gradient-boosting systems, making it a popular tool for machine learning competitions.

Criminisi, A., Shotton, J., & Konukoglu, E. (2012). "Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning." The authors discuss decision forests as a unified framework, comparing them against other classifiers and analyzing their versatility for different machine learning tasks.

Guyon, I., & Elisseeff, A. (2003). "An introduction to variable and feature selection." This work emphasizes the importance of feature selection and engineering. It discusses methods for selecting relevant features, which can significantly improve the performance of machine learning models.

5. Data Description

For this project, I used three datasets: Breast Cancer, Wine and Heart disease. Both the Breast Cancer and the Wine datasets were obtained from the Scikit-learn library while the Heart Disease data sets were obtained from the UCL Machine Learning Repository.

- **Breast Cancer Dataset**: This dataset used features derived from digital images to classify whether a breast tumour is benign or malignant. Below is a tabular representation of the Seven characteristics of the raw Breast Cancer dataset.

Category	**Details**
I. Number of Variables	Independent: 30, Dependent: 1
II. Number of Records	569

III.	Data Types of Combination	Numerical and Binary (Target)
IV.	Summary of Each Variable	See below
V.	Data Cleaning	Not applied at this stage
VI.	Data Normalization	Not applied
VII.	Data Balancing Characteristics and Splitting	Class 0:212, class 1:357

Summary Statistics

Variables	**Min**	**25% Quartile**	**Median**	**75% Quartile**	**Max**	**Mean**
Mean radius	6.981	11.700	13.370	15.780	28110	14.127
Mean texture	9.710	16.170	18.840	21.800	39.280	19.290
Mean perimeter	43.790	75.170	86.240	104.100	188.500	91.969
Mean area	143.5	420.3	55.11	782.7	2501.0	654.889
Mean smoothness	0.05263	0.08637	0.09587	0.1053	0.1634	0.09636
Mean compactness	0.01938	0.06492	0.09263	0.1304	0.3454	0.10434
Mean concavity	0.0	0.02956	0.06154	0.1307	0.4268	0.0888
Mean concave points	0.0	0.02031	0.0335	0.074	0.2012	0.04892
Mean symmetry	0.106	0.1619	0.1792	0.1957	0.304	0.18116
Mean fractal dimension	0.04996	0.0577	0.06154	0.06612	0.09744	0.0628
Radius error	0.115	0.2324	0.3242	0.4789	2.873	0.40517
Texture error	0.3602	0.8339	1.108	1.474	4.885	1.21685
Perimeter error	0.757	1.606	2.287	3.357	21.98	2.86606
Area error	6.802	17.85	24.53	45.19	542.2	40.3371
Smoothness error	0.001713	0.007041	0.008474	0.0109	0.03113	0.007041
Compactness error	0.002252	0.01308	0.02045	0.03245	0.1354	0.025478
Concavity error	0.0	0.01509	0.02589	0.04205	0.396	0.031894
Concave point error	0.0	0.007638	0.01093	0.01471	0.05279	0.011796
Symmetry error	0.007882	0.01516	0.01873	0.02348	0.07895	0.020542
Fractal dimension error	0.00895	0.002248	0.003187	0.004558	0.02984	0.003795
Worst radius	7.93	13.01	14.97	18.79	36.04	16.269
Worst texture	12.02	21.08	25.41	29.72	49.54	25.677
Worst perimeter	50.41	84.11	97.66	125.4	251.2	107.261
Worst area	185.2	515.3	686.5	1084	4254	880.583
Worst smoothness	0.07117	0.1166	0.1313	0.146	0.2226	0.132369
Worst compactness	0.02729	0.1472	0.2119	0.3391	1.058	0.254265
Worst concavity	0.0	0.1145	0.2267	0.3829	1.252	0.272188

Worst concave points	0.0	0.06493	0.09993	0.1614	0.291	0.114606
Worst Symmetry	0.1565	0.2504	0.2822	0.3179	0.6638	0.290076
Worst Fractal dimension	0.05504	0.07146	0.08004	0.09208	0.2075	0.083946
Target	0	0	1	1	1	0.627417

- **Wine Dataset**: This dataset is utilized for wine classification based on its chemical analysis. Below are the characteristics of the dataset.

Category		**Details**
I.	Number of Variables	Independent: 13, Dependent:1
II.	Number of Records	178
III.	Data Types of Combination	Numerical and Categorical (Target)
IV.	Summary of Each Variable	See Below
V.	Data Cleaning	Not applied
VI.	Data Normalization	Not applied

Summary of each variable

Variable	**Min**	**25% Quartile**	**Median**	**75% Quartile**	**Max**	**Mean**
Alcohol	11.03	12.36	13.05	13.68	14.83	13.00
Malic_acid	0.74	1.60	1.87	3.08	5.80	2.34
Ash	1.36	2.21	2.36	2.56	3.23	2.37
Alkalinity_of_ash	10.6	17.2	19.5	21.5	30.0	19.49
Magnesium	70	88	98	107	162	99.74
Total_phenols	0.98	1.74	2.36	2.8	3.88	2.30
Flavaniods	0.34	1.21	2.14	2.88	5.08	2.03
Nonflavanoid_phenold	0.13	0.27	0.34	0.44	0.66	0.36
Proanthocyanins	0.41	1.25	1.56	1.95	3.58	1.59
Color_intensity	1.28	3.22	4.69	6.2	13.0	5.06
Hue	0.48	0.78	0.97	1.12	1.71	0.96
Od280/0d315_of_diluted_wines	1.27	1.94	2.78	3.17	4.0	2.61
Prolinne	278	500.5	673.5	985	1680	746.89

- **Heart Disease Dataset:** This dataset aims to predict the presence of heart failure in patients based on various medical analyses. The characteristics of the dataset are listed

Category	Details
Number of Variables	Independent Variables:13, Dependent:1
Number of Records	303
Data Types of Combination	Binary, Nominal and Numerical
Summary of each Variable	See table below
Data Cleaning	Not applied
Data Normalization	Not applied
Data Balancing Characteristics and Splitting	Class Distribution: Class 0: 164, Class 1:55, Class 2:36, Class 3:35, Class 4, 13

Summary of Each Variable

Variable	Min	25% Quartile	Median	75% Quartile	Max	Mean
ages	29	48	56	61	77	54.44
sex	0	0	1	1	1	0.68
cp	1	3	3	4	4	3.16
trestbps	94	120	130	140	200	131.69
chol	126	211	241	275	564	246.69
Fbs	0	0	0	0	1	0..15
restecg	0	0	1	2	2	0.99
thalach	71	133.6	153	166	202	149.61
exang	0	0	0	1	1	0.33
oldpeak	0	0	0.8	1.6	6.2	1.04
slope	1	1	2	2	3	1.60
ca	0	0	0	1	4	0.67
thai	1	2	2	3	3	2.31
target	0	0	0	2	4	0.94

Methodology

To compare two implementations of identically named techniques across three datasets, I used the K-Nearest Neighbor (KNN)and the DecisionTree Technique. For the KNN Machine learning technique, I used two implementations: KNeighborsClassifier from sklearn library and the KDTree from the Scipy library. For the DecisionTree Machine learning technique, I used two Implementations: DecisionTreeClassifier and XGBoost classifier.

1. K-Nearest Neighbor Technique on Breast Cancer Dataset

Implementation (I) using K-Neighbors Classifier: The breast cancer dataset was loaded from sklearn datasets. Then preprocessing using Pandas for data manipulation and analysis. Handling of duplicates in the data was done using **X.drop_duplicates()** to avoid bias in the dataset. Normalization of the data was done using Standardscaler which is important for algorithms like K-Nearest Neighbors (KNN) which rely on distance calculations. The preprocessed data is then saved to a CSV file and later loaded for the training of the model. Then KNeighborClassifier is employed for the model training. KNeighborClassifier is initialized with 5 neighbours. I explored different values of K but K=5 seems to produce the most optimal result. The dataset is split into training and testing using train_test_split, 70 percent for training and 30 percent for testing. The KNN classifier is trained on the training data **(X_train, y_train).** The trained data is then used to predict the target variable on the test data **(X_test resulting in y_pred).**

Implementation II using Scipy KDTree: In this implementation, The KDTree (Documentation, 2008 - 2024) from the Scipy library is used. The preprocessed dataset saved to a CSV file was loaded for this implementation which helped for effective comparison of both implementations. The same settings employed in implementation 1 above are also used in this implementation. The model is trained using the KDTree. The KDTree is built using the training dataset, while the test dataset is used to query the KD-Tree for the k nearest neighbors of each point in X_test and this returns the distances of K-nearest neighbors and the indices of the neighbors in the training dataset. Finally, the indices of the k-nearest neighbors in each test dataset are used to identify their corresponding labels in the y_train.

Below is the table containing the seven characteristics of the preprocessed Breast cancer dataset.

Category	**Detail**
I. Number of Variables	Independent Variables:30, Dependent:1
II. Number of Records	569
III. Data Types of Combination	Numerical (Features), Binary (Target)
IV. Summary of each Variable	See the table below
V. Data Cleaning	
a. Number and proportion of irrelevant predictive/independent variables removed.	None
b. Number and Proportion of Duplicates Removed	0
c. Dimensionality Reduction (PCA/OLS)	Not applied 0
d. Number and Proportion of Missing Values	None
e. Number and Proportion of Outliers Filtered	See the table of Summary of each variable
f. First Four Characteristics of the Dataset after Data Cleaning	
VI. Data Normalization	Applied using Standardizer, 569(100%)
VII. Data Balancing Characteristics & Splitting.	
a. Number of records per class (0 and 1)	Class 0: 212, Class 1: 357
b. Number of Records in Training Data	398
c. Number of Records in Testing Data	171

Summary Statistics

Variables	Min	25% Quartile	Median	75% Quartile	Max	Mean
Mean radius	-2.029648	-0.689385	-0.215082	0.469393	3.971288	-1.373633e-16
Mean texture	-2.229249	-0.104636	-0.10436	0.584176	4.651889	7.492542e-17
Mean perimeter	-1.984504	-0691956	-0.235980	0.499677	3.976130	-1.248757e-16
Mean area	-1.454443	-0.667195	-0.295187	0.363507	5.250529	-2122887e-16
Mean smoothness	-3.112085	-0.710963	-0.034891	0.636199	4.770911	-8.116921e-16
Mean compactness	-1.610136	-0.747086	-0.221940	0.493857	4.568425	1.873136e-16
Mean concavity	-1.114873	-0.743748	-0.342240	0.526062	4.243589	4.995028e-17
Mean concave points	-1.261820	-0.737944	-0.397721	0.646935	3.927930	-3.746271e-17
Mean symmetry	-2.744117	-0.703240	-0.071627	0.530779	4.484751	1.748260e-16
Mean fractal dimension	-1.819865	-0.722639	0-178279	0.470983	4.910919	4.838933e-16
Radius error	-1.059924	-0.623571	-0.292245	0.266100	8.906909	2.247763e-16
Texture error	-1.554264	-0.694809	-0.197498	0.466552	6.655279	-1.155100e-16
Perimeter error	-1.044049	-0623768	-0.286652	0.243031	9.461986	-1.123881e-16
Area error	-0.737829	-0.494754	-0.347783	0.106773	11.041842	-1.248757e-16
Smoothness error	-1.776065	-0.624018	-0.220335	0.368355	8.029999	-1.498508e-16
Compactness error	-1.298098	-0.692926	-0.281020	0.389654	6.143482	1.810698e-16
Concavity error	-1.057501	-0.557161	-0.199065	0.336752	12.072680	1.685822e-16
Concave point error	-1.913447	-0.674490	-0.140496	0.472657	6.649601	0.000000e+00
Symmetry error	-1.532890	-0.651681	-0.219430	0.355692	7.071917	9.990056e-17

Fractal dimension error	-1.096968	-0.585118	-0229940	0.288642	9.851593	6.243785e-16
Worst radius	-1.726901	-0.674921	-0.269040	0.522016	4.094189	-8.491548e-16
Worst texture	-2.223994	-0.748629	-0.043516	0.658341	3.885905	1.248757e-17
Worst perimeter	-1.693361	-0.689578	-0.285980	0.540279	4.287337	-3.621395e-16
Worst area	-1.222423	-0.642136	-0.341181	0.357589	5.930172	0.000000e+00
Worst smoothness	-2.682695	-0.691230	-0046843	0.597545	3.955374	-2.247763e-16
Worst compactness	-1.443878	-0.681083	-0.269501	0.539669	5.112877	-3.496520e-16
Worst concavity	-1.305831	-0.756514	-0.218232	0.531141	4.700669	7.492542e-17
Worst concave points	-1.745063	-0.756400	-0.223469	0.712510	2.685877	2.497514e-16
Worst Symmetry	-2.160960	-0641864	-0127409	0.450138	6.046041	2.747265e-16
Worst Fractal dimension	-1.601839	-0.691912	-0.216444	0.450762	6.846856	-5619407e-16

1. Decision Tree Technique on Breast Cancer Dataset.

- **Implementation III using Decision Tree:** The Decision Tree Classifier from the sci-kit-learn library is utilized for this implementation. Pandas are used for data manipulation, the dataset was loaded from sklearn datasets, **train_test_split** is used for splitting the data into training and testing data, **StandardScaler** for normalizing the data, and DescisionTreeClassifier from sklearntree is used in the decision tree model.
  After loading the dataset, the data types are identified, and then handling of duplicates to make sure that the model doesn't overfit. Normalization is carried out using Standardscaler, the data is split into training and testing datasets using 70/30. The processed data is saved to a CSV file and then loaded from the CSV file.
  A Decision Tree classifier with a random state is utilized to ensure the reproducibility of the result. The Decision Tree Classifier is trained on the training dataset and then the trained model is used to make predictions on the testing set.

- **Implementation IV using XGBoost:** The XGBoost classifier from the XGBoost library is utilized for this implementation. The processed dataset saved to a CSV file in implementation 1 is loaded to train and fit the model. The preprocessed dataset is read into

a panda data frame containing x and y targets, and the data is split into training and testing sets. A seed value “Random_state=42” is used to ensure the reproducibility of the split.

The training dataset is used to fit the XGBoost model and then the trained model is used to predict the test dataset.

2. K-Nearest Neighbor Technique using the Wine Dataset

- **Implementation V using K Neighbors Classifier:** In this implementation, The K-Nearest Neighbor Classifier is utilized on the Wine dataset. The same preprocessing method and setting which were used for the earlier mentioned K-Nearest Neighbor Classifier were applied. However, we added dimensionality reduction using Principal Component Analysis (PCA) to reduce the dataset to 5 principal components, after normalizing the dataset, and it is saved to a CSV file to be used to fit and predict the model. The PCA was applied to help improve the model performance and there was a significant improvement to the model after applying PCA than the model without PCA.

Below are 7 characteristics of the Wine dataset.

Characteristic	Description
I. Number of Variables	Independent: 5, Dependent: 1,
II. Total Number of Records	178
III. Data Types of Combination	Numerical & Categorical (Target)
IV. Summary of Each Variable	See table below
V. Data Cleaning	
a. Number and proportion of irrelevant predictive/independent variables removed.	None
b. Number and proportion of duplications removed (if any in data) and technique employed towards duplication removal.	0
c. Dimensionality reduction based on PCA/OLS and self-observation.	PCA (5 Component) used for dimension reduction
d. Number and proportion of missing values in total and number of missing values dealt employing a technique to deal with missing value of your choice.	Not applied
e. Number and proportion of outliers filtered	Not applied
f.First four characteristics of datasets after performing (1-4) data cleaning steps.	See the table below for a Summary of Each Variable
VI. Data Normalization	All 178 were normalized using StandardScaler
VII. Data Balancing Characteristics & Splitting:	Class 0: 71, Class 1: 59, Class 2: 48 Records in Training dataset: 124 Records in Testing datasets: 54

Statistics	PC1	PC2	PC3	PC4	PC5	Target
Mean	-16.0e-16	2.00e-17	-4.99e-17	2.00e-17	-2.99e-17	0.94
Min	-4.28	-3.52	-4.59	-2.89	-2.02	0.00
25%	-2.17	-1.23	-0.83	-0.69	-0.57	0.00
Median	0.06	-0.26	-0.14	-0.03	-0.27	1.00
75%	2.00	1.40	0.76	0.59	0.36	2.002
Max	4.31	3.87	5.35	3.79	4.19	2.00

- **Implementation vi using the Scipy KDTree**: This implementation uses the same settings and preprocessed data as the KNN Classifier in Implementation v except that it uses the KDTree from the Scipy library to initialize the model.

3. Decision Tree Technique using Wine dataset.

- **Implementation VII using Decision Tree Classifier:** This implementation uses the Decision Tree classifier from sci-kit-learn as I have explained earlier on. The data preprocessing and settings involve loading the dataset, dropping duplicates contained in the data, normalizing the data using a standard scaler and then the data is saved to a CSV file. The saved data is loaded and then split into training and test data (70\30). The Decision tree classifier is used to train and evaluate the model.
- **Implementation VII using XGBoost:** This implementation follows the settings and also uses the same preprocessed dataset from the implementation**.** Below is the table showing the seven characteristics of the preprocessed dataset a.

Characteristics	**Detail**
I. Number of Variables	Independent: 13, Dependent 1
II. Number of Records	178
III. Data Types of combination	Numerical and Categorical (Target Variable)
IV. Summary of Each Variable	See table below
V. Data Cleaning a. Number and proportion of irrelevant predictive/independent variables removed.	0
b. Number and proportion of duplications removed (if any in data) and technique employed towards duplication removal.	0(0%), Duplicate checked using drop_duplicates()
c. Dimensionality reduction based on PCA/OLS and self-observation.	Not applied
d. Number and proportion of missing values in total and number of missing values dealt employing a technique to deal with missing value of your choice.	No missing value is present.
e. Number and proportion of outliers filtered.	Not applied
f. First four characteristics of datasets after performing (1-4) data cleaning steps	See the table of Summary of each variable
VI. Data Normalization	Normalized using StandardScaler

VII.	Data Balancing Characteristics & Splitting	Class distribution: Class: 0.398876 Class 1: 0.331461 Class 2: 0.269663 Records in Training Set: 124 Records in Testing Set: 54

Summary Statistics

Variable	Mean	Min	25%	Median	75%
alcohol	-8.382808e-16	-24342	-0.7882	0.0610	0.8361
Malic_acid	-7.983626e-17	-1.4330	0.6587	-0.4231	0.6698
ash	-8.170742e-17	-3.6792	-0.5721	-0.0238	0.6981
alkalinity_of-ash	-3.9918134e-17	-2.6710	-0.6891	0.0015	0.6021
magnesium	-3.991813e-17	-2.0883	-0.8244	-0.1223	0.5096
total_phenols	-3.991813e-17	-2.1072	-0.8855	0.0959	0.8089
flavanoids	-4.390994e-16	-1.6960	-0.8275	0.1061	0.8491
nonflavonoid_phenols	3.592632e-16	-1.8682	-0.7401	-0.1761	0.6095
proanthocyanins	-1.197544e-16	-2.0690	-0.5973	-0.0629	0.6292
Color_intensity	3.991813e-17	-1.6343	-0.7951	-0.1592	0.4939
hue	1.995907e-16	-2.0947	-0.7676	0.0331	0.7132
od280/od315_of_ diluted_wines	3.193450e-16	-1.8951	-0.9522	0.2377	0.7886
proline	-1.596725e-16	-1.4932	-0.7846	-0.2337	0.7582
target	0.938202	0.0000	0.0000	1,0000	2.000c

1. K-Nearest Neighbor Technique using Heart Disease Dataset

- **Implementation using K-Neighbors Classier:** In this implementation, The K-Nearest Neighbor Classifier is utilized on the heart disease dataset. The preprocessing of the data involves the following:
  - **a.** Splitting of the datasets into features
  - **b.** Encoding of categorical variables into numerical using label encoder.
  - **c.** Normalization of the data using a standard scaler.

Then the data is saved as a CSV file to ensure the same preprocessing is used for the second implementation. The saved dataset is then loaded and then split into training and testing data (70/30). The KNN classifier is then initialized with 5 neighbors and Euclidean distance metrics. Then the model that is trained is used to make predictions for the test dataset. Finally, accuracy and f1 performance are printed. Below is a table of the characteristics of the preprocessed heart disease dataset.

Characteristics	Details
1. Number of variables	Independent: 13, Dependent: 1
2. Number of Records	303
3. Data Types of Combination	Numerical:13, Categorical :1
4. Data Cleaning a. Number and proportion of irrelevant predictive/independent variables removed; b. Number and proportion of duplications removed (if any in data) and technique employed towards duplication removing; c.Dimensionality reduction based on PCA/OLS and self-observation; d.Number and proportion of missing values in total and number of missing values dealt employing a technique to deal with missing value of your choice e. Number and proportion of outliers filtered f.First four characteristics of datasets after performing (1-4) data cleaning steps	 Not applied 0 Not applied Not applied Not applied See the summary of each variable table
5. Summary of each variable	See table below
6. Data Normalization	All data normalized using standardscaler
7. Data Balancing Characteristics and Splitting	Class 0: 164, Class 1: 55, Class 2: 36, Class 3: 35, Class 4: 13. Records in Training dataset: 212 Records in Test dataset: 91

Summary of Each Variable

Variable	Min	Max	Median	Mean	25%	75%
Age	-2.82	2.50	0.17	1.91e-17	-0.71	0.73
sex	-1.46	-1.46	0.69	-2.93e-17	-1.46	0.69
CP	-2.25	-2.25	-0.17	-122e-16	-0.17	0.88
Trestbps	-2.15	-2.15	-0.10	4.54e-16	-0.67	0.47
chol	-2.33	-2.33	-0.11	2.42e-16	-0.69	0.55
Fbs	-0.42	-0.42	-0.42	5.86e-18	-0.42	-0.42
Restecg	-1.00	-1.00	0.01	-7.04e-17	-1.00	1.02
Thaclach	-3.44	-3.44	0.15	-1.17e-16	-0.71	0.72
Exang	-0.70	-0.70	-0.70	-909e-17	-0.70	1.44
Oldpeak	-0.90	-0.90	-0.21	2.35e-17	-0.90	0.48
Slope	-0.98	-0.98	0.65	1.41e-17	-0.98	0.65
Ca	-0.71	-0.71	-0.71	2.35e-17	-0.71	0.28
Thai	-0.88	-0.88	-0.88	7.62e-17	-0.88	1.19
Target	0.00	0.00	0.00	0.94	0.00	2.00

- **Implementation using the Scipy KDTree**: This implementation uses the same preprocessing above. The preprocessed data is loaded and split into training and testing datasets with a random value of 42 for reproducibility and the KDtree from Scipy is used using the same 5 neighbours as the K neighbors classifier above. The model is then trained with the training dataset and used to make predictions for the test dataset.

2. Decision Tree Technique using the Heart Disease Data

- **Implementation using Decision Tree Classifier:** in this implementation, the data was first loaded from the CSV file and then the following was applied to the dataset:
  - **a.** Data cleaning by removing and handling missing values, outliers removal and duplicate removal
  - **b.** Encoding of categorical variables using a label encoder to transform the categorical variables 'ca' and 'thal' into the numerical format.
  - **c.** Outlier detection using Inter quartile range.
  - **d.** Normalization/Feature scaling is carried out using the 'StandardScaler'
  - **e.** Dimensionality reduction is applied using PCA to reduce the dimensionality of the data while maintaining 95 per cent of the variance.

After the preprocessing steps, the dataset is then saved to a CSV file and then loaded. The processed data is split into training and test sets (70/30). The Decision tree classifier is initialized with a random state for reproducibility and then trained with the training dataset and then used to predict the outcome on the test dataset. The accuracy and f1 score of the model are printed out.

- Implementation using XGBoost: The preprocessed data is loaded and then this implementation., then the data is split as above. Then the XGBoost tree is used to initialized

the model, the model is trained with the trained dataset and then the trained model predicts the outcome on the test dataset. The model accuracy and F1 score are printed out.

Below is the table characteristics of the preprocessed Heart disease Dataset used for the above implementation.

Characteristics	**Details**
1. Number of variables	Independent: 13, Dependent: 1
2. Number of Records	303
3. Data Types of Combination	Numerical:14, Binary :3
4. Data Cleaning a. Number and proportion of irrelevant predictive/independent variables removed; b. Number and proportion of duplications removed (if any in data) and technique employed towards duplication removing; c.Dimensionality reduction based on PCA/OLS and self-observation; d.Number and proportion of missing values in total and number of missing values dealt employing a technique to deal with missing value of your choice e. Number and proportion of outliers filtered f.First four characteristics of datasets after performing (1-4) data cleaning steps	 Not applied 0 Applied to retained 95% variance 0 Outliers filtered See the summary of each variable table
5. Summary of each variable	It is the same as the table above of the summary of each variabe
6. Data Normalization	All data normalized using standardscaler
7. Data Balancing Characteristics and Splitting	Class 0: 164, Class 1: 55, Class 2: 36, Class 3: 35, Class 4: 13. Records in Training dataset: 212 Records in Test dataset: 91

6. Evaluation Methods and Results

To evaluate the result for the comparison of two different implementations of Identical machine learning techniques we employed:

Accuracy: Accuracy is simply a ratio of correctly predicted observations to the total observations. It is the proportion of true results (both true positives and true negatives) among the total number of cases examined.

F1 Score: The F1 score is the harmonic mean of precision and recall taking both false positives and false negatives into account. It is a good way to show that a classifier has a good value for both recall and precision.

K Neigbhbor Classifier vs KDtree on the Breast Cancer Dataset

KNN classifier	**KDtree KNN**
F1 Score: 0.98 Accuracy: 0.97	F1 Score: 0.98 Accuracy: 0.97

Both implementations perform the same because the main functionality of how the nearest neighbors contribute to the classification decision is fundamentally the same between the two implementations regardless of their different libraries.

Decision Tree classifier vs XGBoost Classifier on Breast Cancer Dataset

Decision Tree Classifier	**XGBoost Classifier**
F1 Score: 0.93 Accuracy: 0.92	F1 Score: 0.97 Accuracy: 0.96

The ability of XGBoost to combine the strengths of many trees and optimize their collective decisions helps it achieve superior accuracy and flexibility in the breast cancer dataset prediction, making it outperform the decision tree.

K Neigbhbor Classifier vs KDtree on the Wine Dataset

KNN classifier	**KDTree**
Model Accuracy: 0.96 F1 Score: 0.96	Model Accuracy: 0.96 F1 Score: 0.96

Both models performed the same way.

Decision Tree Classifier vs XGBoost Classifier

Decision Tree model	**XGBoost Model**
F1 Score: 0.96 Accuracy: 0.96	F1 Score: 1.00 Accuracy: 1.00

The XGBoost model performed better than the decision tree model on the Wine dataset.

K Neigbhbor Classifier vs KDtree on the Heart Disease Dataset

KNN	KDTree
Model Accuracy: 0.55 F1 Score: 0.51	Model Accuracy: 0.58 F1 Score: 0.56

The Scipy KDTree model performs slightly better than the KNN model.

Decision Tree Classifier vs XGBoost Classifier

Decision Tree	**XGBoost**
Accuracy: 0.75 F1 Score: 0.76	Accuracy: 0.80 F1 Score: 0.79

The XGBoost model performs better than the Decision Tree model because of its improved algorithms.

7. Discussion and Conclusion

Two research questions are asked in this project.

1. Do the two-implementation of identically named techniques perform differently or the same?

Yes, different implementations of identically named machine learning techniques do perform the same depending on the type of implementation used, as you can see in the case of the KNN and KDTree classifier (Documentation, 2008 - 2024)in the breast cancer dataset and the wine data set.

2. If they are performing differently, then what could be the reason? For example, one possible reason maybe that they are internally using different algorithms, or implicitly employing some data processing (confirm using the documentation) or some other reason.

Also, two implementations of identically named techniques do perform differently in some cases as you can see from the Evaluation method and results section of the project, The XGBoost classifier performs differently from the decision tree classifier even though they both use the tree method for classification but the ability of XGBoost (Chen, 2016)to combine the strengths of many trees, the use of their improved gradient tress and optimize their collective decisions helps it achieve superior accuracy and flexibility across the 3 datasets that I used in this project, making it outperform the decision tree.

There is also a situation where the SCipy KDTree performs slightly better than the K-Nearest Neighbor. This might be because KDTree is efficient in handling spatial queries, which can be crucial in high-dimensional spaces or datasets with complex distributions like the heart disease dataset. Also, the various preprocessing applied to the Heart Disease Dataset can affect the performance of the Decision tree. The KNN and KDTree internally implement different algorithms for processing data. For example, KNN from Sci-kit-learn uses various algorithms to compute the nearest neighbour and it then automatically chooses the best one depending on the dataset but the KDTree from Scipy manually uses partitional space for its computation which can be suitable for the nature of the heart disease dataset making it to slightly perform better than he K-nearest neighbor classifier.

Chen, T. &. G. C., 2016. *XGBoost: A scalable tree boosting system.,* s.l.: Association for Computing Machinery.

Cover, T. &. H. P., 1967. Nearest neighbor pattern classification. *Nearest neighbor pattern classification,* 13(1), pp. 21 - 27.

Criminisi, A. S. J. &. K. E., 2012. Decision forests: A unified framework for classification. *Decision Forests,* 7(now publishers inc), pp. 81-227.

Cunningham, P., & Delany, S. J., 2020. *k-Nearest Neighbour Classifiers: 2nd Edition (with Python examples),* s.l.: arXiv:2004.04523v2 [cs.LG.

Documentation, S. K., 2008 - 2024. *The Scipy community.* [Online]
Available at: https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.KDTree.html
[Accessed 10 05 2024].

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, et al, 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research 12 (2011) 2825-2830,* 12(10/11), pp. 2825-2830.

Friedman, J. H. T. &. T. R., 2001. *he elements of statistical learning.* s.l.:Springer.

Guyon, I. &. E. A., 2003. An introduction to variable and feature selection. *Journal of Machine Learning Research 3,* Issue 3, pp. 1157-1182.

page, C. M. i. M., 2024. *Decision Tree.* Dublin: NCIRL.

Quinlan, J. R., 1986. Induction of decision trees.. *Machine Learning*, pp. 81 - 106.

Repository, T. U. M. L., n.d. *UC Irvine Machine Learning Repository.* [Online]
Available at: https://archive.ics.uci.edu/
[Accessed 10 May 2024].

Shakhnarovich, G., Darrell, T., & Indyk, P. (Eds.), 2005. *"Nearest-Neighbor Methods in Learning and Vision: Theory and Practice.".* s.l.:The MIT Press Cambridge, Massachusetts London, England.

Wikipedia, t. f. e., 2024. *Wikipedia, the free encyclopedia.* [Online]
[Accessed 10 05 2024].

Wikipedia, t. f. e., 2024. *Wikipedia, the free encyclopedia*. [Online]
Available at: https://en.wikipedia.org/wiki/F-score
[Accessed 10 05 2024].