	KNN1	The KNN model applied to the Iris dataset achieved high accuracy, around 97%. Optimal performance was observed with K=3, evident from the error rate plot. KNN, effective for classification tasks, accurately predicts flower species based on measurements. This model's success validates its applicability for identifying unknown Iris species. The algorithm's simplicity and effectiveness make it a solid choice for similar classification problems.
	KNN2	The K-Nearest Neighbors (KNN) classifier was applied to the Titanic dataset after preprocessing. Data cleaning included handling missing values and encoding categorical variables. With K=5, the model achieved decent classification results, though exact accuracy isn't explicitly stated. The algorithm effectively distinguished survivors from non-survivors based on the processed features. KNN's simplicity and ability to handle mixed data types make it suitable for such classification tasks, providing insights into survival patterns on the Titanic.
	Naïve Bayes	The code demonstrates the application of the Categorical Naive Bayes classifier on a simple weather-based play decision dataset. After encoding categorical data into numerical format, the algorithm learns the probabilistic relationship between weather conditions, temperature, and the decision to play. The model predicts playing under 'Overcast' weather and 'Mild' temperatures with an unspecified accuracy. Naive Bayes assumes feature independence, simplifying computation. It's useful for quick predictions in scenarios where features can be considered independent, showcasing the algorithm's capability to make decisions based on given conditions.
	Decision Tree	The code employs a Decision Tree Classifier on a weather dataset to predict if a game will occur based on weather and temperature conditions. It transforms categorical data into numerical form for model compatibility. Using entropy as the criterion, the tree learns to make decisions, predicting a 'Yes' (game on) for 'Overcast' weather and 'Mild' temperature conditions. While accuracy isn't explicitly shown, decision trees are adept at handling both categorical and numerical data, making them versatile for various classification tasks. The visualized tree reveals the decision-making process, showing conditions leading to 'Play' or 'Don't Play' outcomes, demonstrating the algorithm's capability to generate interpretable models for decision-making based on specific criteria
	DBSCAN	The code implements the DBSCAN clustering algorithm on customer data from a mall, focusing on annual income and spending score attributes. DBSCAN identifies six distinct clusters and a number of noise points within the data. Visual inspection of the scatter plot reveals clear groupings, suggesting the algorithm successfully segments customers based on spending habits and income levels. The silhouette score of 0.47 indicates moderate cohesion and separation between clusters. DBSCAN's ability to find arbitrarily shaped clusters without specifying the number of

		clusters beforehand makes it valuable for exploratory data analysis, particularly in scenarios with noisy data or unknown cluster structures.
	AgglomerativeClustering	The code applies Agglomerative Hierarchical Clustering to segment mall customer data based on annual income and spending score. Three clusters were identified, with no noise points detected. The silhouette score of 0.48 suggests moderate cohesion within clusters and separation between them. The dendrogram visually represents the merging of clusters, aiding in determining the optimal number of clusters. This method is advantageous for understanding hierarchical relationships in data and can be particularly useful for applications requiring a detailed view of data structure, such as customer segmentation for targeted marketing strategies
	KmeansClusttering	The code demonstrates K-Means clustering on various datasets, including synthetic data and real-world spine measurement data. It visualizes data distribution and clusters, showing how K-Means groups similar data points together. Through the Elbow Method, it determines the optimal number of clusters by minimizing distortion, a measure of the squared distances between points and their closest centroid. Silhouette scores indicate the quality of clustering, with higher scores suggesting better-defined clusters. The algorithm successfully identifies distinct groups in the data, which can be crucial for tasks like customer segmentation, pattern recognition, and data compression, offering insights into underlying data structure without prior labeling.
	AprioriMarketAnalysis	The code performs market basket analysis using the Apriori algorithm on a dataset representing grocery transactions. It identifies frequent itemsets with a support threshold of 0.6, revealing items frequently bought together. Filtering for itemsets longer than two items and with high support uncovers complex shopping patterns. Association rules generated from these itemsets, with a confidence threshold of 0.7, highlight strong conditional dependencies between products. Such insights are valuable for retail strategy, enabling personalized recommendations, optimizing store layouts, and informing inventory management. The algorithm's ability to detect co-occurrence patterns aids in understanding customer buying behavior without requiring labeled data.
	ImageClassification	The code performs image classification on fruit images using color histograms as features. It extracts histograms from the HSV color space, trains models (k-NN, SVM, and Neural Network), and evaluates their performance. The k-NN model with k=9 achieved an accuracy of around 70%, while the SVM and Neural Network models showed similar accuracies. These algorithms effectively classify fruits based on color characteristics, demonstrating the utility of color histograms for image categorization tasks. The models can identify different fruits, making them useful for applications in food industry automation, quality control,

		and computer vision systems.
	Linear Regression	The code builds models to predict sales based on advertising expenditures across TV, radio, and newspapers. It uses both Simple Linear Regression (SLR) and Multiple Linear Regression (MLR). SLR with TV as the predictor achieved an R-squared value of 81.10%, indicating a good fit but not accounting for other variables. MLR, incorporating all three advertising mediums, improved the model's explanatory power with an R-squared of 90.21%, highlighting the combined effect of advertising channels on sales. The MLR model's lower Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) compared to SLR suggest better predictive accuracy. This algorithm helps understand the impact of advertising spend on sales, aiding in budget allocation and forecasting sales based on marketing investments.