Journal of Scientific Insights
E-ISSN: 3062-8571
DOI: https://doi.org/10.69930/jsi.v2i1.222
Research Article

ScienceTech. Group
Vol. 2 (1), 2025
Page: 14-34

Honey Price Classification using K-Nearest Neighbor Machine Learning

Budi Aribowo [1,*], Aprilia Tri Purwandari [1], Ni'mah Tsabitah [1], Reudinta Zesha [1], Dwi Astharini [2]

[1] Department of Industrial Engineering, Faculty of Sains and Technology, Universitas Al-Azhar Indonesia, Indonesia
[2] Department of Electical Engineering, Faculty of Sains and Technology, Universitas Al-Azhar Indonesia, Indonesia
*Email (corresponding author): budiaribowo@gmail.com

Abstract. *The global honey market faces significant challenges due to price inconsistencies, which often do not correlate with the actual quality of the honey. This research aims to develop a honey price classification model based on quality using the K-Nearest Neighbor (K-NN) machine learning method. The contribution in this research is a machine learning model that can classify honey prices validated by measured variables of honey concentration. Data was collected on 14 types of honey, focusing on price per 100 ml, pcategorized into 'cheap' and 'expensive' classifications. Data processing includes statistical testing using T-Test to determine the significance of price differences, followed by applying the K-NN algorithm for classification. Model performance is evaluated using metrics such as accuracy, the Receiver Operating Characteristic (ROC) curve, a graph used to evaluate the performance of binary classification models, and the Area Under the Curve (AUC). The results show that the K-NN model achieves an optimal accuracy of 100% and an AUC of 1.00 when the K parameter is set to 3, indicating excellent classification ability. It is hoped that these findings will increase market transparency, set fairer price standards, and help consumers and producers in making decisions in purchasing honey.*

Keywords: *Honey price classification, k-nearest neighbor, cross-validation technique*

1. Introduction

The global honey market has experienced significant growth over the past few decades. However, one of the main challenges this market faces is the uncertainty in setting prices that correspond to the quality of honey. This variability can lead to significant price differences that do not always reflect the actual quality of the honey. This uncertainty confuses consumers and opens opportunities for unethical producers to manipulate prices. This research addresses this issue by developing a price classification model for honey based on its quality using the K-Nearest Neighbor (K-NN) machine learning method. This method was chosen for its ability to identify hidden patterns in data, thus providing more accurate and fair price estimates. Through the development of this model, it is hoped to enhance transparency in the honey market, provide more consistent pricing standards, and assist consumers and producers in making better decisions.

Various works has been done in providing classification system for honey. An embedded system was developed by Hidayatullah et al (2021), to classify honey nectar based on brightness and color using naive Bayes. Classification system of honey types was developed on (Alhamdani et.al. 2022) based on color, brightness, and pH using the JST

Received: 14 November 2024 / Accepted: 28 December 2024 / Available online: 14 January 2025

backpropagation method. Rochman and Mukhtar (2019) had introduces classification system of honey quality using machine learning based on spectrofotometer data. There has been no publication concerning classification of market price of honey. The main update and contribution of this work is the development of classification model for honey price based on its quality using the K-NN.

Honey (Genus Apis) is a sweet liquid with numerous benefits as a sugar substitute. Each type of honey has different characteristics in color and taste. Honey comes from nectar collected by bees in their nests, known as "honeycombs." Nectar is stored in the honeycomb without going through food digestion because there is a separation between the digestive sac and the honey sac. The chemical process of honey occurs through heat, air exchange, and evaporation (Hidayatullah et al., 2021). While the national regulatory bodies regulate the honey quality standards, an international platform is also available to provide guidelines for international honey trading. The Alimentarius regulation strictly states that the consumers have the right to receive accurate information about the food they are going to consume. The honey should not have any added ingredients, any foreign material, aroma, tainted absorbed substances during processing or storage (Kumar Joshi et al., 2024).

Indonesian honey has great potential due to diverse nectar sources but is limited by high water content (25-29%), exceeding the ASEAN and Indonesian National Standard (SNI) No. 013545 the Year 1994 standards of 22%. This affects its quality and storage durability (Sigit Amanto et al., 2012).

Honey varieties can be monofloral or multifloral, depending on whether the nectar comes from one or several plant species to produce honey. Monofloral honey is more expensive than multifloral honey (Suhandy et al., 2020).

Multifloral honey from various regions in Indonesia, such as Sumbawa, Riau, and Bangka, has characteristics that are based on location. Honey production in Indonesia is around 4000 tons annually, mostly from wild honey hunting in forests. Riau is one of the largest honey producers in Indonesia (Hasan et al., 2020). The plant origin of honey affects its appearance, color, water activity, and chemical composition. Honey mainly contains glucose, fructose, and about 200 minor components like pollen, minerals, vitamins, and proteins. Quality indicators, such as hydroxymethylfurfural content, moisture, enzyme activity, and pesticide residues, are unrelated to its geographical or botanical source (Demir and Bayram, 2018).

The nutritional composition of honey varies across Apis species, particularly in moisture, ash, and protein content. Wild species like apis dorsata and apis florea have lower moisture levels (7.88% and 10.18%) compared to cultivated apis mellifera (12.19%). Moisture content in other species ranges from 20.5% to 26% (Mustafa et al., 2023).

Honey is rich in antioxidants, enzymes (catalase, glucose oxidase, peroxidase), non-enzymatic nutrients (ascorbic acid, α-tocopherol, carotenoids, amino acids, flavonoids, phenolic acids), and Maillard reaction products. Its antioxidant content varies with the nectar source. Honey also contains vitamins like B1, B2, B6, C, niacin, pantothenic acid, biotin, folic acid, and vitamin K (Wulandari, 2017). Fructose in honey may lower blood glucose levels by prolonging gastric emptying time, reducing intestinal absorption, and stimulating glucokinase in the liver, which aids in glucose uptake and storage (Kadirvelu and Gurtu, 2013).

Honey composition consists of approximately 80-85% carbohydrates, 15-17% water, 0.3% protein, 0.2% ash, and several amino acids and vitamins. Honey quality affects shelf life and consumer benefits. According to the Indonesian National Standard (SNI) 01-3545-2013, quality honey must have a maximum water content of 22% and a maximum acidity level of 50 ml NaOH/kg (Sjamsiah et al., 2018). Honey is primarily composed of simple carbohydrates, mainly fructose (38.5%) and glucose (31%), making up 82.4% of its carbohydrate content. It also contains maltose, sucrose, and other complex sugars, along with minor components like proteins, enzymes, amino acids, organic acids, lipids, vitamins, minerals, and bioactive compounds such as flavonoids and phenolics (Engin Gündoğdu, Songül Çakmakçı and İhsan Güngör Şat, 2019).

Honey quality is assessed based on its physicochemical and microbiological properties. SNI 3545:2013 establishes essential parameters to ensure honey quality. SNI 7899:2013 and SNI 8664:2018 regulate honey management as raw material, with SNI 8664:2018 explaining production requirements from harvest to post-harvest. These parameters include water content, enzyme content, acidity levels, and metal contamination limits such as lead, cyanide, and mercury (Ferdian et al., 2015).

Data mining as an information extraction process requires machine learning to process the data. Machine learning is a computer application and mathematical algorithm that learns from data to make future predictions. Generally, machine learning methods are divided into three categories: supervised, unsupervised, and reinforcement (Prabowo et al., 2020). Data mining is the extraction of implicit, previously unknown, and potentially useful information from data. The idea is to build computer programs that sift through databases automatically, seeking regularities or patterns. Strong patterns, if found, will likely generalize to make accurate predictions on future data (Marnoto, 2014).

The supervised learning model predicts future events based on historical data. The algorithms used include Regression, Naive Bayesian, K-Nearest Neighbor, Decision Tree, Logistic Regression, and Neural Network. Supervised learning techniques are further divided into classification for categorical data and prediction for numerical data (Hidayatullah et al., 2021). K-Nearest Neighbor (K-NN) is a supervised learning algorithm, meaning it requires labeled training data to make predictions. This is similar to other classification methods like Decision Trees and Logistic Regression (Suyal and Goyal, 2022).

Classification in machine learning is one of the main topics in data mining, where the data used has classes or labels. Classification algorithms fall under supervised learning (Lawry, 2006; Sumathi et al., 2019). Classification methods include Decision Tree, Naive Bayes, K-Nearest Neighbor, and Logistic Regression (Sumathi and Sivanandam, 2019).

Cross Validation helps balance data by systematically dividing datasets and identifying an optimal K value to improve training and test data distribution (Hulu and Sihombing, 2020). K Fold Cross Validation ensures all data points are used for both training and validation across iterations, unlike a single train-test split, which may underutilize the dataset (Chamorro-Atalaya et al., 2023). K Fold Cross Validation (K-FCV) method is often employed as an evaluation technique to partition data into multiple subsets, reduce overfitting, and ensure the model performs optimally across different datasets (Tembusai, Mawengkang and Zarlis, 2021). Cross-validation should be a standard procedure in data analysis, either as a model-selection procedure or as validation of models selected in other ways. Focusing on out-of-sample performance increases the chances that obtained results are

replicable (Yarkoni & Westfall, 2017), or as Bokhari and Hubert (2018) declared, "the lack of cross validation can lead to inflated results and spurious conclusions" (de Rooij and Weeda, 2020).

The K-Nearest Neighbor (K-NN) method searches for the closest values in the training data relative to the test data by calculating distances. The value of K determines the number of nearest neighbors that are evaluated. The Euclidean distance calculation, commonly used and the default in Python, measures the distance between two points in space based on the length of the straight line connecting them (Irfanuddin et al., 2024), as in formula (1).

$$d_i = \sqrt{\Sigma_{i=1}^{\Pi}(x_{21} - x1_i)^2} \quad (1)$$

di = Euclidean distance to the i-th data point
$x2i$ = training data for the i-th point
$x1i$ = testing data for the i-th point
i = the i-th row

K-Nearest Neighbor (K-NN) is a supervised method that requires training data to classify objects based on their nearest distances. The principle of K-NN is to find the nearest distance between the data to be evaluated and the K nearest neighbours in the training data (Devita et al., 2018).

K-Nearest Neighbor (K-NN) is a straightforward yet highly effective classification method that operates without requiring any assumptions about data distribution. It can be utilized to accurately distinguish between best-selling and non-best-selling products (Danny, Muhidin and Jamal, 2024). K-Nearest Neighbor (K-NN)is often used for smaller datasets or when interpretability is crucial. In contrast, more complex algorithms like deep learning are preferred for large datasets with high-dimensional data, such as images, where they can leverage their capacity to learn complex patterns (Najwaini et al., 2023).

Karl Pearson created the confusion matrix in 1904. The confusion matrix is a square matrix of size N × N, where N denotes the number of output classes. Each row of the matrix represents the number of instances of a predicted class and each column represents the number of instances of the actual class. This provides a class-by-class breakdown of the number of accurate and inaccurate predictions made by a classifier for classification tasks. Categorization can be binary or multiclass. The confusion matrix reveals the classifier's performance, what it is getting right, and the many kinds of mistakes the classifier could make. The metrics derived from the confusion matrix help choose the best course of action to enhance the performance of the model. As confusion matrices can be constructed for datasets with known target/output values, they are used in supervised learning methods (Sathyanarayanan, 2024).

2. Methods

This research study aims to classify honey prices using a machine learning approach with the K-Nearest Neighbor method.

First of all, this research started with data collection by purchasing 14 types of honey from various stores. The data was then divided into two categories, namely low price and high price. The collected honey was measured using a refractometer to ensure the accuracy of

honey concentration measurement. Next, T-Test was conducted to determine whether there was a significant difference between the cheap and expensive price categories. Provided the test results show a significant difference, the process continues with honey classification using the K-Nearest Neighbors (KNN) algorithm, which will predict the price category based on the existing data.

The classification results are then evaluated using Confusion Matrix and ROC Curve to assess the performance of the KNN model. In addition, the model is validated using Cross Validation techniques to ensure that the model does not experience overfitting and has consistent performance.

Figure 1 shows flowchart illustrates the research process that begins with a literature study and problem identification, followed by data collection using questionnaires at 14 test stations. The collected data is verified, where the valid data is directly tested using factor analysis with p-value, while the invalid data is returned to the collection stage if the results are not significant. If significant, a reference value is determined and the data is fed into SQLite for model accuracy testing. Once the model is tested, the data is used for simulation, which is the final step in the research process

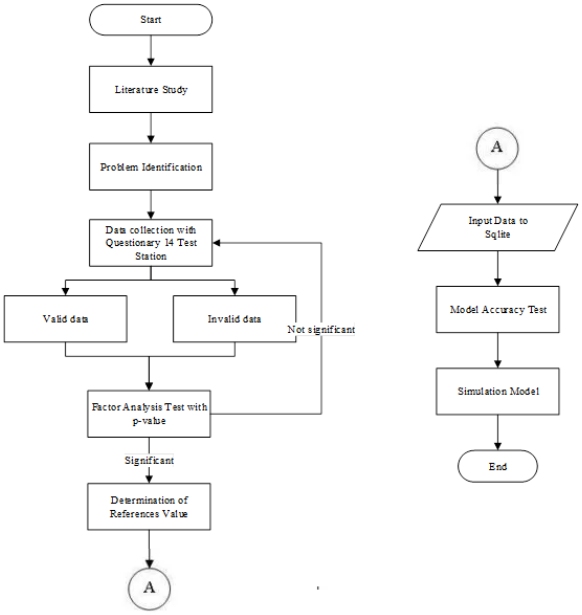

Figure 1. Development flowchart

3. Results and Discussion

Explains in detail the data collected, data processing, modeling, and performance evaluation of the applied KNN model.

3.1. Data Collection

Table 1 displays information about 14 types of honey, including the original size, original price, price per 100 ml, and their classification based on price per 100 ml. The first column shows the type of honey, the second column the size in millilitres (mL), the third column the original price in rupiah (Rp), the fourth column the price per 100 mL, and the last column the classification "Cheap Price" or "Expensive Price." This classification is based on price per 100 mL, with "Expensive Price" for honey that costs more than Rp 30,000 per 100 mL.

Table 1. Honey price

Types of Honey	Original size (mL)	Original price	Price per 100ml	Classification
Honey A	650	Rp. 40,800.00	Rp. 6,276.92	Cheap price
Honey B	350	Rp. 25,000.00	Rp. 7,142.86	Cheap price
Honey C	1000	Rp. 79,000.00	Rp. 7,900.00	Cheap price
Honey D	500	Rp. 39,999.00	Rp. 7,999.80	Cheap price
Honey E	150	Rp. 12,400.00	Rp. 8,266.67	Cheap price
Honey F	650	Rp. 55,000.00	Rp. 8,461.54	Cheap price
Honey G	350	Rp. 35,800.00	Rp. 10,228.57	Cheap price
Honey H	450	Rp. 50,000.00	Rp. 11,111.11	Cheap price
Honey I	250	Rp. 28,800.00	Rp. 11,520.00	Cheap price
Honey J	250	Rp. 75,000.00	Rp. 30,000.00	Expensive Price
Honey K	125	Rp. 39,000.00	Rp. 31,200.00	Expensive Price
Honey L	250	Rp. 99,000.00	Rp. 39,600.00	Expensive Price
Honey M	250	Rp. 149,000.00	Rp. 59,600.00	Expensive Price
Honey N	100	Rp. 250,000.00	Rp. 250,000.00	Expensive Price

For example, Honey A measuring 650 mL with a price of Rp 40,800 has a price per 100 mL of Rp 6,276.92 and is categorised as a "Cheap Price." Honey J, measuring 250 mL with a price of Rp 75,000, has a price per 100 mL of Rp 30,000 and is also categorised as a "Cheap Price." Honey M measuring 250 mL with a price of Rp 149,000 has a price per 100 mL of Rp 59,600 and is categorized as "Expensive Price." An extreme example is Honey N measuring 100 mL with a price of Rp 250,000, resulting in a price per 100 mL of Rp 250,000 and is categorized as "Expensive Price."

Table 2. Honey content data collection

Types of Honey	Brix value	Be	Water	Temperature	PPM 10 grams of honey (100)	Classification
Honey A	73	38.9	25.5	31.8	21	Cheap price
Honey B	76.00	40.3	22.5	31.7	5	Cheap price
Honey C	75.5	40	23	31.9	14	Cheap price
Honey D	73.6	39.3	24.5	31.8	33	Cheap price
Honey E	76.00	40.5	21.5	31.8	5	Cheap price
Honey F	73.5	39	25	31.8	125	Cheap price
Honey G	72	39.5	26	31.6	20	Cheap price
Honey H	76.8	40.8	21.5	31.4	20	Cheap price
Honey I	75.6	40.3	22.5	31.8	14	Cheap price
Honey J	78.5	41.8	19.5	31.1	80	Expensive Price
.Honey K	80.5	42.5	18	30.2	158	Expensive Price
Honey L	79.5	42	19	30.5	107	Expensive Price
Honey M	81.5	43	17.5	30.6	164	Expensive Price
Honey N	83	43	15.5	30.2	284	Expensive Price

The Table 2 shows 14 honey (AN) types based on chemical and physical parameters and price. The Brix value (sugar content) ranges from 73 (Honey A) to 83 (Honey N). The lowest density (Be) is 38.9 (Honey A), and the highest is 43 (Honey N & Honey M). The lowest water content is 15.5% (Honey N), and the highest is 25.5% (Honey A). The highest PPM is 284 (Honey N), and the lowest is 5 (Honey B). Honey is classified based on price: "Cheap Price" (below Rp 30,000/100 mL) and "Expensive Price" (above Rp 30,000/100 mL). Honey A is included in "Cheap Price," while Honey N is included in "Expensive Price."

3.2. Data Processing

```
import pandas as pd
# Importing or loading the dataset
dataset = pd.read_csv(r'C:\Users\tsabi\Downloads\Honey\Classification of Honey.csv',index_col='Types of Honey')
print(dataset)
```

Figure 2. Non parametric test

The figure 2 shows the results of the Kruskal-Wallis H test on two groups of honey prices: cheap and expensive. The affordable group has prices per 100 mL [6276.92, 7142.86, 7900, 7999.80, 8266.67, 8461.54, 10228.57, 11111.11, 11520], while the expensive group has prices [30000, 31200, 39600, 59600, 250000]. The Kruskal-Wallis H test produces a statistical value of 9.0 and a p-value of 0.0027, indicating a significant difference between the two price groups. With a p-value below 0.05, we reject the null hypothesis and conclude that the prices of cheap and expensive honey are significantly different.

3.2.1. Data Cleaning

Data Cleaning is performed before modeling to ensure the data is ready to be processed, reduce noise, and minimize errors. The goal is to maximize the classification results. This data-cleaning process involves detecting missing values (empty, infinite, or NaN entries) and avoiding mistakes by deleting dataset rows if the disappeared values exceed a certain threshold.

Figure 3 shows a snippet of Python code using the pandas library to read and analyze a dataset on honey classifications. The dataset includes various parameters such as Brix Value, Be Water, pH, Temperature, and PPM (Parts Per Million) levels in three sizes (100, 200, and 300 grams) for different types of honey. Based on this data, each type of honey is classified into price categories: Cheap Price or Expensive Price.

```
import pandas as pd
# Importing or loading the dataset
dataset = pd.read_csv(r'C:\Users\tsabi\Downloads\Honey\Classification of Honey.csv',index_col='Types of Honey')
print(dataset)
```

```
              Brix Value    Be  Water   PH  Temperature  \
Types of Honey
Honey A             73.0  38.9   25.5  4.52         31.8
Honey B             76.0  40.3   22.5  4.25         31.7
Honey C             75.5  40.0   23.0  4.09         31.9
Honey D             73.6  39.3   24.5  3.74         31.8
Honey E             76.0  40.5   21.5  4.36         31.8
Honey F             73.5  39.0   25.0  4.13         31.8
Honey G             72.0  39.5   26.0  4.60         31.6
Honey H             76.8  40.8   21.5  4.54         31.4
Honey I             75.6  40.3   22.5  3.97         31.8
Honey J             78.5  41.8   19.5  3.87         31.1
Honey K             80.5  42.5   18.0  3.86         30.2
Honey L             79.5  42.0   19.0  3.51         30.5
Honey M             81.5  43.0   17.5  3.83         30.6
Honey N             83.0  43.0   15.5  4.90         30.2

              PPM 10 grams of honey (100)  PPM 10 grams of honey (200)  \
Types of Honey
Honey A                                21                           12
Honey B                                 5                            2
Honey C                                14                           10
Honey D                                33                           21
Honey E                                 5                            2
Honey F                               125                           79
Honey G                                20                           12
Honey H                                20                           13
Honey I                                14                           10
Honey J                                80                           50
Honey K                               158                          101
Honey L                               107                           61
Honey M                               164                           89
Honey N                               284                          166

              PPM 10 grams of honey (300)   Classification
Types of Honey
Honey A                                 9      Cheap Price
Honey B                                 1      Cheap Price
Honey C                                 7      Cheap Price
Honey D                                16      Cheap Price
Honey E                                 1      Cheap Price
Honey F                                57      Cheap Price
Honey G                                 9      Cheap Price
Honey H                                10      Cheap Price
Honey I                                 7      Cheap Price
Honey J                                36  Expensive Price
Honey K                                71  Expensive Price
Honey L                                46  Expensive Price
Honey M                                64  Expensive Price
Honey N                               121  Expensive Price
```

Figure 3. Output import dataset

```
dataset.isnull()
```

Types of Honey	Brix Value	Be	Water	PH	Temperature	PPM 10 grams of honey (100)	PPM 10 grams of honey (200)	PPM 10 grams of honey (300)	Classification
Honey A	False	False	False	False	False	False	False	False	False
Honey B	False	False	False	False	False	False	False	False	False
Honey C	False	False	False	False	False	False	False	False	False
Honey D	False	False	False	False	False	False	False	False	False
Honey E	False	False	False	False	False	False	False	False	False
Honey F	False	False	False	False	False	False	False	False	False
Honey G	False	False	False	False	False	False	False	False	False
Honey H	False	False	False	False	False	False	False	False	False
Honey I	False	False	False	False	False	False	False	False	False
Honey J	False	False	False	False	False	False	False	False	False
Honey K	False	False	False	False	False	False	False	False	False
Honey L	False	False	False	False	False	False	False	False	False
Honey M	False	False	False	False	False	False	False	False	False
Honey N	False	False	False	False	False	False	False	False	False

Figure 4. Output data after data cleaning

Figure 4 is the image of data cleaning that has been done to detect missing values. The result is that no missing values are found in the data. So, there is no need to add data because there is no empty data. Furthermore, the data can be processed by data reduction using feature selection.

3.2.2. Data Reduction

This data selection aims to select attributes that influence the classification of honey prices.

```
#Feature Selection

from sklearn.feature_selection import SelectKBest, f_classif
import numpy as np

x = dataset[["Brix Value","Be", "Water","PH","Temperature","PPM 10 grams of honey (100)","PPM 10 grams of honey (200)","PPM 10 grams of honey (300)"]]
y = dataset[["Classification"]]
```

```
best_features = SelectKBest(score_func = f_classif, k = 1)
fit = best_features.fit(x, np.ravel(y))
print(fit.scores_)
```

```
[39.63472604 52.41568187 36.9131364   1.35523842 78.         18.29566119
 17.25211572 17.19517885]
```

```
print(fit.pvalues_)
```

```
[3.97364701e-05 1.02883555e-05 5.53787364e-05 2.66987967e-01
 1.34695486e-06 1.07404565e-03 1.33805070e-03 1.35451762e-03]
```

Figure 5. Output data feature selection

Figure 5 shows the process of conducting data selection. The selected attribute data has the highest feature score and the lowest values. Based on the image process above, the top five with the highest score are taken.

Table 3. F-score and P-values

No	Variables	f-score	p-value
1	Brix value	39,6347	3,9736
2	Be	52,4157	1,0288
3	Water	36,9131	5,5379
4	PH	1,3552	2,6699
5	Temperature	78,0000	1,3469
6	PPM 10 grams of honey (100)	18,2957	1,0740
7	PPM 10 grams of honey (200)	17,2521	1,3381
8	PPM 10 grams of honey (300)	17,1952	1.3545

Table 3 show variables,f-score, and p-value It can be concluded that the variables taken based on the results of feature selection are the Brix, Be, Water, Temperature, and PPM values of 10 grams of honey (100).

3.2.3. Data Modeling

Data modeling is performed at this stage using KNeighbors with different N values (1, 3, 5, 7, and 9) to determine the most appropriate model. Testing is carried out to obtain the accuracy and AUC values of each model, which will be used for classification analysis and determining the accuracy level of the machine learning model.

3.2.3.1 Data Collection

Before the data is processed, the vendor data and its criteria values are imported and displayed using the panda's module in Jupyter Notebook from a CSV file.

```
import pandas as pd
# Importing or Loading the dataset
dataset = pd.read_csv(r'C:\Users\tsabi\Downloads\Honey\Classification of Honey (after feature selection).csv',index_col='Types of Honey')
print(dataset)
```

```
                Temperature    Be  Brix Values  Water  \
Types of Honey
Honey A                31.8  38.9         73.0   25.5
Honey B                31.7  40.3         76.0   22.5
Honey C                31.9  40.0         75.5   23.0
Honey D                31.8  39.3         73.6   24.5
Honey E                31.8  40.5         76.0   21.5
Honey F                31.8  39.0         73.5   25.0
Honey G                31.6  39.5         72.0   26.0
Honey H                31.4  40.8         76.8   21.5
Honey I                31.8  40.3         75.6   22.5
Honey J                31.1  41.8         78.5   19.5
Honey K                30.2  42.5         80.5   18.0
Honey L                30.5  42.0         79.5   19.0
Honey M                30.6  43.0         81.5   17.5
Honey N                30.2  43.0         83.0   15.5

                PPM 10 grams of Honey (100)   Classification
Types of Honey
Honey A                                  21      Cheap Price
Honey B                                   5      Cheap Price
Honey C                                  14      Cheap Price
Honey D                                  33      Cheap Price
Honey E                                   5      Cheap Price
Honey F                                 125      Cheap Price
Honey G                                  20      Cheap Price
Honey H                                  20      Cheap Price
Honey I                                  14      Cheap Price
Honey J                                  80  Expensive Price
Honey K                                 158  Expensive Price
Honey L                                 107  Expensive Price
Honey M                                 164  Expensive Price
Honey N                                 284  Expensive Price
```

Figure 6. Output import data using pandas module

Figure 6 show Python code with a dataset that has gone through a feature selection process. The dataset includes selected parameters such as Temperature, Be, Brix Values, and Water, which are relevant for the classification of honey types. The data also includes the PPM (Parts Per Million) level for 10 grams of honey and price classification into Cheap Price and Expensive Price. This process aims to simplify the analysis by using only features that are significant to the economic value classification of honey.

3.2.3.2 Defining Independent Variables (X) and Dependent Variables (Y)

Machine Learning must define the independent and dependent variables to be processed at this stage. The following is the program code to define variables X and Y.

```
x = dataset[["Temperature","Be","Brix Values","Water","PPM 10 grams of Honey (100)"]]
y = dataset[["Classification"]]
```

```
x
```

Types of Honey	Temperature	Be	Brix Values	Water	PPM 10 grams of Honey (100)
Honey A	31.8	38.9	73.0	25.5	21
Honey B	31.7	40.3	76.0	22.5	5
Honey C	31.9	40.0	75.5	23.0	14
Honey D	31.8	39.3	73.6	24.5	33
Honey E	31.8	40.5	76.0	21.5	5
Honey F	31.8	39.0	73.5	25.0	125
Honey G	31.6	39.5	72.0	26.0	20
Honey H	31.4	40.8	76.8	21.5	20
Honey I	31.8	40.3	75.6	22.5	14
Honey J	31.1	41.8	78.5	19.5	80
Honey K	30.2	42.5	80.5	18.0	158
Honey L	30.5	42.0	79.5	19.0	107
Honey M	30.6	43.0	81.5	17.5	164
Honey N	30.2	43.0	83.0	15.5	284

Figure 7. Output import data using pandas module

Figure 7 the variable `x` contains independent features from the dataset, such as Temperature, Be, Brix Value, Water, and PPM of 10 grams of honey. While the variable `y` contains the "Classification" column which is the classification target. This step is important in data pre-processing before statistical analysis or machine learning modeling.

3.2.3.3 Converting Categorical Data to Numeric

Figure 8 displays an easy and yet effective technique for converting categorical data into numeric data before using it to train a machine learning model. The dataset is subjected to encoding to transform it into a format suitable for analysis. The specific method applied in this stage is Encoding, primarily applied to the independent variables within the dataset. Encoding is employed to convert categorical variables into a numerical format that can be utilized effectively in machine learning algorithms. This transformation ensures that the categorical attributes are appropriately represented for subsequent modeling and analysis (Eldora, Fernando and Winanti, 2024).

```
#Label Encoder

from sklearn import preprocessing

label_encoder = preprocessing.LabelEncoder()

dataset['Classification'] = label_encoder.fit_transform(dataset['Classification'])
dataset['Classification'].unique()
```

```
array([0, 1])
```

Figure 8. Label encoding

3.2.3.4 Sharing Training Data and Testing Data

According to research conducted by (Gholamy et al., 2018). Using 20-30% for test data and 70-80% for training data shows the best empirical results. In this study, the author divided each data set into 70% or 0.7 training data and 30% or 0.3 testing data. Training data is used to train the classification model, while testing data is used to test model performance with data assumed to have never been 'seen' or digested by the model. The separation between training and testing data can keep the testing process pure from data manipulation. The method of dividing training and testing data, or train-test-split in this case, uses the sci-kit-learn library. The following in Figure 9 is the program code to display the results of the train-test-split process.

.

```
#Split Training & Testing

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3)
```

Figure 9. Traint & test split

Figure 9 showThe code uses the train_test_split function from the sklearn library to split the dataset into training data and testing data. This function is imported using the command from sklearn.The dataset is divided into four parts.

Table 4. Train-Test-Split data results

Information	***Training Data***	***Testing Data***	**Total**
Amount	9	5	14
Percentage	70%	30%	100%

Table 4 contains the training and testing data percentage from the total data. In the table, nine training data are equivalent to 70% of the total data. Meanwhile, the testing data is 5, comparable to 30% of the total data. Training data is used to train the Machine Learning model. The model will learn the patterns and relationships between input and output data in this process. Test data is used to evaluate the performance of the trained machine-learning model. The model will be tested with data that has never been seen before to determine its ability to predict the correct results. Dividing data into training and testing is important to ensure that the machine-learning model does not overfit. Overfitting occurs when the model is too trained on the training data to not predict well on new data. Using testing data, we can determine how well the machine-learning model will work on accurate data.

3.2.3.5 Machine Learning Model Testing Using Training Data

```
x_train
```

Types of Honey	Temperature	Be	Brix Values	Water	PPM 10 grams of Honey (100)
Honey E	31.8	40.5	76.0	21.5	5
Honey L	30.5	42.0	79.5	19.0	107
Honey N	30.2	43.0	83.0	15.5	284
Honey F	31.8	39.0	73.5	25.0	125
Honey A	31.8	38.9	73.0	25.5	21
Honey D	31.8	39.3	73.6	24.5	33
Honey I	31.8	40.3	75.6	22.5	14
Honey B	31.7	40.3	76.0	22.5	5
Honey K	30.2	42.5	80.5	18.0	158

Figure 10. Output variable after train-test-split data

Figure 10 reveals the process after dividing the training and testing data, where the next stage is the classification analysis using five classification methods. The classification method tested in this study is K-Nearest Neighbors (KNN). The modeling process uses the scikit-learn library. The programming code above is the classification program code used for modeling training data.

3.2.3.6 Testing the accuracy level of each Machine Learning model using testing data and prediction data.

Next, each model is trained using training data that has gone through the previous training and testing data process. Model testing produces a fi score that researchers use to compare the performance of each model using the scikit-learn library. The following is the classification program code for each model.

3.2.4 K-Nearest Neighbors (KNN) Classification Analysis

K-Nearest Neighbors (KNN) is a type of supervised machine learning whose classification system considers the distance between observation data. KNN has parameters symbolized by k, which will later be optimized to determine the best parameters with the best classification performance results (Al Aziez & Anuraga, 2021). The following are the stages of KNN classification modeling.

3.2.4.1 Machine Learning Model Testing Using Training Data

```
#Train the Model
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors = 1)
knn.fit(x_train, y_train)
```

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=1)
```

Figure 11. Train the model support vector machine

Figure 11 shows the steps of importing the KNeighborsClassifier class from the sklearn.neighbors module, initializing a KNeighborsClassifier object with the n_neighbors parameter set to 3, and training a KNN model using the training dataset x_train and training labels y_train. Then_neighbors parameter specifies the number of nearest neighbors to be used for prediction. A smaller n_neighbors value will result in a model that is more sensitive to outliers, while a larger n_neighbors value will result in a more stable model.

3.2.4.2 Confusion matrix

The confusion matrix shows the performance of the K.N.N. model algorithm in classifying visually. This matrix allows one to compare the actual classification to its prediction. The following is the Confusion matrix program code for the K.N.N. model.

```
#Confusion Matrix

import matplotlib.pyplot as plt
import numpy
from sklearn import metrics

actual = y_test
predicted = knn.predict(x_test)

confusion_matrix = metrics.confusion_matrix(actual, predicted)
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [False, True])

cm_display.plot()
plt.show()
```

Figure 12. Input code confusion matrix

Figure 12 show code is used to create and display a confusion matrix as an evaluation of the performance of the classification model. First, libraries such as matplotlib, numpy, and the metrics module from sklearn are imported.

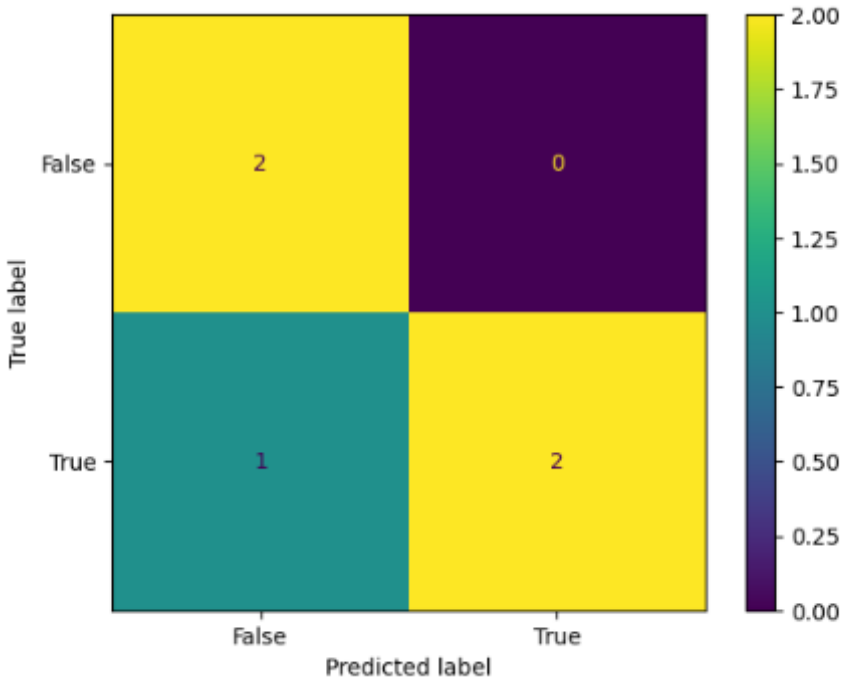

Figure 13. Confusion matrix model support vector machine

Figure 13 shows the number of correct and incorrect predictions for each class. For the negative class (False), there are two correct predictions (True Negative) and no incorrect predictions (False Positive). For the positive class (True), there are two correct predictions (True Positive) and one incorrect prediction (False Negative). In other words, the KNN model predicts 2 instances as negative and true negative, 2 as positive and true positive, and 1 that is predicted negative when it is positive.

This confusion matrix helps to understand the performance of the model on each class and shows where the prediction errors occur. In this case, the KNN model has a fairly good performance with 4 correct predictions and only 1 incorrect prediction, with no false positives.

3.2.4.3 Testing the level of accuracy using the KNN Machine Learning model using testing data and prediction data

Based on the confusion matrix results, the values contained in the confusion matrix are used to calculate the accuracy value. This value shows the performance of the KNN model in classifying repeat-order decisions. The following is the program code to determine the KNN model's accuracy value.

The accuracy calculation results in the figure 14 show various evaluation metrics, including precision, recall, f1-score, and support for each target class (0 and 1). For class 0, the precision is 0.67, recall 1.00, and f1-score 0.80 with support 2. While for class 1, the precision is 1.00, recall 0.67, and f1-score 0.80 with support 3. The overall accuracy of the model is 0.80 or 80%. The macro averages of precision, recall, and f1-score are 0.83, 0.83, and 0.80, respectively, while the weighted averages considering the class distribution are 0.87, 0.80, and 0.80.

This evaluation shows that the KNN model has a fairly good performance with an overall accuracy of 80%. Although the precision for class 1 is higher than class 0, the recall for class 0 is higher than class 1. The F1-score for both classes is the same, which is 0.80, indicating

a balance between precision and recall. The macro average shows the average performance of the model without considering the class distribution, while the weighted average takes into account the proportion of examples in each class. Overall, this KNN model has a good performance but shows a difference in prediction performance between class 0 and class 1.

```
#Find the Accuracy

from sklearn.metrics import classification_report

y_pred = knn.predict(x_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.67      1.00      0.80         2
           1       1.00      0.67      0.80         3

    accuracy                           0.80         5
   macro avg       0.83      0.83      0.80         5
weighted avg       0.87      0.80      0.80         5
```

Figure 14. Accuracy of support vector machine model

3.2.4.4 Creating a ROC (Receiver Operating Characteristic) Curve

After calculating the accuracy value of the KNN model, the next step is to create an ROC curve as a measuring tool to assess the ability of the classification system. The following Figure 15 is the program code to create an ROC curve on the KNN model.

```
#Create ROC Curve
from sklearn.metrics import roc_curve
y_pred = knn.predict(x_test)
y_pred = knn.predict_proba(x_test)[:, 1]
import matplotlib.pyplot as plt
knn_fpr, knn_tpr,_= roc_curve(y_test, y_pred, pos_label = 1)
plt.plot(knn_fpr, knn_tpr)
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title('ROC Curve of K-Nearest Neighbors')
plt.plot(knn_fpr, knn_tpr, 'b', label = 'AUC = %0.2f' % knn_auc)
plt.legend(loc = 'lower right')
plt.show()
```

Figure 15. Input code ROC plot

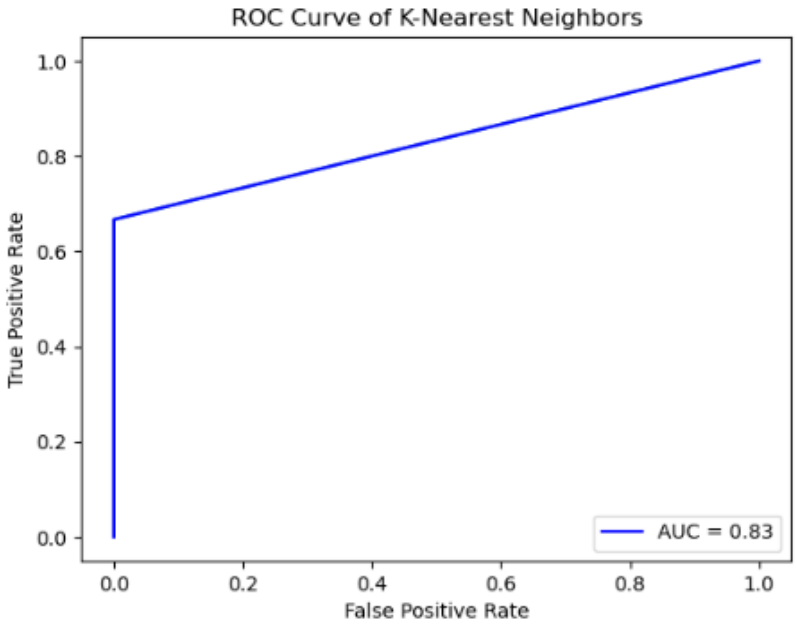

Figure 16. Support vector machine model roc curve

Figure 16 shows the ROC (Receiver Operating Characteristic) curve for the KNN model. The ROC curve is a graph used to evaluate the performance of a binary classification model. This graph shows the trade-off between the model's True Positive Rate (TPR) and False Positive Rate (FPR) at various threshold values. The X-axis shows the False Positive Rate (FPR). FPR is the proportion of negative examples that are incorrectly classified as positive.

Meanwhile, the Y-axis shows the True Positive Rate (TPR). TPR is the proportion of positive examples that are correctly classified as positive. The ROC curve in the figure shows that the KNN model has good performance. This curve is close to the ideal curve and has a high AUC value (0.83). This means that the model is able to classify most of the positive and negative examples correctly.

3.2.4.3 Validation of K-Nearest Neighbors Parameters

KNN has a parameter symbolised by k which will later be validated to determine the best parameter with the best classification performance results. Table 4 is the result of the validation of the k parameter in the KNN algorithm to classify honey price decisions.

Table 4. KNN Parameter Validation

Parameter	Accuracy	AUC
K = 1	0.80	0.83
K = 3	1.00	1.00
K = 5	0.80	1.00
K = 7	0.60	0.92
K = 9	0.60	0.50

Table 4 shows the results of the validation of the k parameter using the KNN algorithm. The results were obtained using training data, namely 70% of the dataset. The table shows the results of evaluating the K-Nearest Neighbors (KNN) model with various K

parameter values, using the Accuracy and Area Under the Curve (AUC) metrics. When K is set to 1, the model classifies a data point based solely on the nearest neighbor. This can lead to high accuracy on the training data but may result in overfitting, where the model captures noise rather than the underlying pattern. For instance, in the study, K=1 achieved an accuracy of 80% and an AUC of 83%, indicating decent performance but not optimal. Increasing K to 3 or 5 generally improves the model's robustness by averaging the classifications of multiple neighbors, which helps to mitigate the effects of noise. The study found that K=3 provided perfect accuracy and AUC of 100%, indicating excellent classification capability. K=5 also performed well, maintaining an accuracy of 80% with a perfect AUC, suggesting strong class discrimination ability.

As K increases further, the model may start to lose sensitivity to local patterns in the data. For example, at K=7, the accuracy dropped to 60%, although the AUC remained relatively high at 92%, indicating that while the model could still discriminate between classes, its overall classification ability had diminished. At K=9, the accuracy remained low at 60%, and the AUC fell to 50%, suggesting performance equivalent to random guessing.

Determining the optimal K value in a K-NN model can be done through cross-validation (e.g., 3-fold, with K=3 resulting in an average validation score of 1.0), evaluation of performance metrics** such as accuracy, AUC, and confusion matrix to find bias-variance balance, and visual analysis by plotting accuracy or AUC against K to identify "elbow points." Proper K selection is important to avoid overfitting at small K and underfitting at large K, with these strategies ensuring good generalization to new data.

3.3. Validation Test (Cross Validation)

Figure 17 shows the cross-validation which is an important technique for assessing the performance of machine learning models, such as the k-nearest neighbours (k-NN) model with k = 3. The dataset, consisting of features `x` and labels `y`, is then split into a training (80%) and a testing (20%) set. The model is evaluated using 3-fold cross-validation via `cross_val_score,` which divides the training set into three equal parts (folds). The model is trained on two folds and validated on the remaining fold, repeating this process three times with different validation folds. The validation scores from each fold are averaged to estimate the model performance. In this case, the cross-validation scores are all 1.0, indicating excellent performance on each fold, and the average cross-validation score is also 1.0, indicating that the model performs very well across different subsets of the training data. This method ensures that the model performance is not overly dependent on the particular training-test split, providing a more reliable measure of the model's generalization ability.

```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score

# Assuming `x` and `y` are your features and labels
# Split the data into training and testing sets (80-20 split)
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)

# Convert y_train and y_test to numpy arrays and flatten them
y_train = y_train.values.ravel()
y_test = y_test.values.ravel()

# Evaluate the model using cross-validation on the training set
cv_scores = cross_val_score(knn, X_train, y_train, cv=3)

# Print cross-validation scores
print("Cross-validation scores:", cv_scores)
print("Mean cross-validation score:", cv_scores.mean())
```

```
Cross-validation scores: [1. 1. 1.]
Mean cross-validation score: 1.0
```

Figure 17. Cross validation

Despite the promising results, this study has several limitations that should be acknowledged.

1. Limited Sample Size
   The dataset comprised only 30 types of honey, which may not be representative of the broader honey market. A larger and more diverse sample could provide a more comprehensive understanding of the factors influencing honey prices.
2. Type of Honey
   The study focused on specific types of honey, which may limit the generalizability of the findings. Different honey varieties may exhibit unique characteristics that affect pricing, and including a wider range of honey types could enhance the model's applicability.
3. Feature Selection
   While the study utilized several physicochemical variables, other factors such as geographical origin, production methods, and seasonal variations were not considered. These factors could significantly influence honey quality and pricing.
   To build upon this study, future research could explore the following avenues:

1. Expanded Dataset
   Collecting a larger dataset that includes a variety of honey types from different regions and production methods would improve the robustness of the model and its applicability to real-world scenarios.
2. Incorporating Additional Features
   Future studies could integrate more variables, such as sensory attributes (taste, aroma) and market dynamics (supply and demand), to enhance the model's predictive capabilities.
3. Comparative Analysis
   Conducting comparative studies with other classification algorithms (e.g., Support Vector Machines, Random Forest) could provide insights into the relative strengths and weaknesses of K-NN in honey price classification.
4. Longitudinal Studies
   Implementing longitudinal studies to track price changes over time could help in understanding the stability of the model and its adaptability to market fluctuations.

Conclusions

Based on the research that has been conducted on honey price classification using the K-Nearest Neighbor (KNN) machine learning method, the following are the conclusions that can be drawn from the research results. The Kruskal-Wallis H test shows a significant difference between cheap and expensive honey prices, with an H value of 9.0 and a p-value of 0.0027, which is far below the threshold of 0.05. The study measured chemical and physical variables of honey such as Brix value, density, water content, temperature, and PPM, with results showing significant differences between honey samples. The KNN model with K=3 gives the best accuracy and AUC of 100% respectively. K=1 and K=5 also give good performance, but the performance decreases at K=7 and K=9. The 3-fold cross-validation technique was used to evaluate the KNN model with K=3, and the results showed excellent performance with a validation score and average of 1.0, indicating a high generalisation ability of the model.

The study successfully developed a honey price classification model using the K-Nearest Neighbor (K-NN) machine learning method. The update in this research is in the form of a machine learning model that can classify honey prices based on the measured variables.demonstrating its effectiveness in distinguishing between cheap and expensive honey based on various physicochemical properties. The results indicated significant differences in honey prices, supported by statistical tests, and the K-NN model achieved optimal accuracy and AUC values, particularly at K=3, showcasing its potential for enhancing market transparency and aiding consumer decision-making.

The system development was limited on small sample number, narrow type of honey, and few feature. By addressing these limitations and exploring these directions, future research can contribute to a more nuanced understanding of honey pricing and improve the effectiveness of machine learning models in this domain.

Funding

This study was funded by UAI Research Grant 2024.

Conflict of Interest

All authors declare that this article has no conflict of interest.

References

A Comparative Study of Different Monofloral and Multifloral Honey Samples from Northern India for their Health Promoting Activities and Physicochemical Parameters. ARC Journal of Nutrition and Growth, [online] 10, pp.2455–2550.

Alhamdani, M. H. J., Syauqy, D., & Prasetio, B. H. (2022). Sistem klasifikasi kualitas jenis-jenis madu berdasarkan warna, kecerahan, dan pH menggunakan metode JST backpropagation. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 6(6), 2584-2590.

Chamorro-Atalaya, O., Arévalo-Tuesta, J., Balarezo-Mares, D., Gonzáles-Pacheco, A., Mendoza-León, O., Quipuscoa-Silvestre, M., Tomás-Quispe, G. and Suarez-Bazalar, R., 2023. K-Fold Cross-Validation through Identification of the Opinion Classification Algorithm for the Satisfaction of University Students. International journal of online and biomedical engineering, 19(11), pp.140–158.

Danny, M., Muhidin, A. and Jamal, A., 2024. Application of the K-Nearest Neighbor Machine Learning Algorithm to Preduct Sales of Best-Selling Products. Brilliance: Research of Artificial Intelligence, 4(1), pp.255–264.

de Rooij, M. and Weeda, W., 2020. Cross-Validation: A Method Every Psychologist Should Know. Advances in Methods and Practices in Psychological Science, 3(2), pp.248–263.

Demir, E. and Bayram, N.E., 2018. Specifying Some Quality Characteristics of Monofloral and Multifloral Honey Samples. Hacettepe Journal of Biology and Chemistry, 3(46), pp.417–423.

Devita, R. N., Herwanto, H. W., & Wibawa, A. P. (2018). Perbandingan kinerja metode naive bayes dan k-nearest neighbor untuk klasifikasi artikel berbahasa indonesia. J. Teknol. Inf. dan Ilmu Komput, 5(4).

Eldora, K., Fernando, E. and Winanti, W., 2024. Comparative Analysis of KNN and Decision Tree Classification Algorithms for Early Stroke Prediction: A Machine Learning Approach. Journal of Information Systems and Informatics, 6(1), pp.313–338.

Engin Gündoğdu, Songül Çakmakçı and İhsan Güngör Şat, 2019. An Overview of Honey: Its Composition, Nutritional and Functional Properties. Journal of Food Science and Engineering, 9(1).

Ferdian, S., Rihiantoro, T., & Handayani, R. S. (2017). Pengaruh Madu Terhadap Kualitas Tidur Pada Lansia. Jurnal Ilmiah Keperawatan Sai Betik, 11(2), 310-317.

Hasan, A. E. Z., Herawati, H., Purnomo, P., & Amalia, L. (2020). Fisikokimia madu multiflora asal Riau dan potensinya sebagai antibakteri Escherichia coli dan Staphylococcus aureus. Chemistry Progress, 13(2).

Hidayatullah, S. T., Syauqy, D., & Fitriyah, H. (2021). Klasifikasi Sumber Nektar Madu berdasarkan Kecerahan dan Warna dengan Metode Naive Bayes berbasis Embedded System. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 5(8), 3455-3461.

Hulu, S. and Sihombing, P., 2020. Analysis of Performance Cross Validation Method and K-Nearest Neighbor in Classification Data. International Journal of Research and Review (ijrrjournal.com), 7(4), p.69.

Irfanuddin, M. S., Nurkamid, M., & Khotimah, T. (2024). Klasifikasi Tingkat Kematangan Buah Belimbing Madu Berdasarkan Karakteristik Warna Menggunakan Algoritma K-Nearest Neighbor. Jurnal Dialektika Informatika (Detika), *4*(2), 65–73.

Kadirvelu, A. and Gurtu, S., 2013. Potential benefits of honey in type 2 diabetes mellitus: A review. International Journal of Collaborative Research on Internal Medicine & Public Health,5(4).

Lawry, J. (2006). Modelling And Reasoning With Vague Concepts. Springer.

Mustafa, G., Iqbal, A., Javid, A., Hussain, A., Bukhari, S.M., Ali, W., Saleem, M., Azam, S.M., Sughra, F., Ali, A., Rehman, K.U., Andleeb, S., Sadiq, N., Hussain, S.M., Ahmad, A. and Ahmad, U., 2023. Variations in nutritional profile of honey produced by various species of genus Apis. Brazilian Journal of Biology, 83.

Najwaini, E., Thomas Edyson Tarigan, Fajri Profesio Putra and Sulistyowati, 2023. Application of the K-Nearest Neighbors (KNN) Algorithm on the Brain Tumor Dataset. International Journal of Artificial Intelligence in Medical Issues, 1(1), pp.18–26.

Prabowo, S., Yuliani, Y., Prayitno, Y. A., Lestari, K., & Kusesvara, A. (2020). Penentuan Karakteristik Fisikokimia Beberapa Jenis Madu Menggunakan Metode Konvensional dan Kimia. Jurnal Pertanian Pangan Tropis, *1*(2), 66.

Rochman, S., & Mukhtar, M. N. A. (2019). Klasifikasi Kualitas Madu Menggunakan Sistem Spektrofotometer Dan Machine Learning Berdasarkan Komputer Single Board. Tibuana, 2(01), 45-49.

Sathyanarayanan, S., 2024. Confusion Matrix-Based Performance Evaluation Metrics. African Journal of Biomedical Research, pp.4023–4031.

Sigit Amanto, B., Her Riyadi, N. P., & Staf pengajar Fakultas Program Studi Ilmu dan Teknologi Pangan. (2012). Studi Karakteristik Alat Pengurangan Kadar Air Madu dengan Sistem Vakum Kental. Jurnal Hasil Teknologi Pertanian,Vol. V (Issue 2).

Sjamsiah, Sikanna, R., Aazmalaeni Rifkah, & Saleh, A. (2018). Penentuan Sifat Fisikokimia Madu Hutan (Apis Dorsata) dari Wilayah Maros, Pangkep dan Gowa Provinsi Sulawesi Selatan. Al-Kimia, *6*(2), 43–53.

Suhandy, D., Yulia, M., & Kusumiyati, K. (2020). Klasifikasi Madu Berdasarkan Jenis Lebah (Apis Dorsata versus Apis mellifera) Menggunakan Spektroskopi Ultraviolet dan Kemometrika Klasifikasi Madu Berdasarkan Spesies Lebah (Apis Dorsata versus Apis Mellifera) Menggunakan Spektroskopi Ultraviolet dan Kemometrik. *Jurnal Ilmu Pertanian Indonesia*, *25*(4), 564–573

Sumathi, S. , & Sivanandam, S. , N. (2019). *Introduction To Data Mining And It's Application. Springer-Verlag. Berlin, Heidelber*

Suyal, M. and Goyal, P., 2022. A Review on Analysis of K-Nearest Neighbor Classification Machine Learning Algorithms based on Supervised Learning. International Journal of Engineering Trends and Technology, 70(7).

Tembusai, Z.R., Mawengkang, H. and Zarlis, M., 2021. K-Nearest Neighbor with K-Fold Cross Validation and Analytic Hierarchy Process on Data Classification. International Journal of Advances in Data and Information Systems, 2(1).

Wulandari, D. D. (2017). Kualitas madu (keasaman, kadar air, dan kadar gula yang berkurang) berdasarkan suhu penyimpanan yang berbeda. Jurnal Kimia Riset, 2(1), 16–22.