A STUDY ON SUPERVISED LEARNING MODEL – K-NN CLASSIFICATION

[1]**Dr. UdayaSri Kompalli** , M.B.A., M.Sc.,(IT),M.Tech.(CSE),MA(Astro),Ph.D., Gold Medal, World Book of Records, FSIESRP,
Assistant Professor, Dept. of Computer Science, PB Siddhartha College of Arts & Science, Vijayawada
kudayasri@pbsiddhartha.ac.in
[2]**Avanigadda Sivayya**, [3]**Gandham Mani Saketh**, [4]**Abdul Faheem**,
[2,3,4] Student, B.Sc.-AIML, PB Siddhartha College of Arts & Science.

DOI : 10.48047/IJFANS/V11/I12/740

Abstract:
In the world of machine learning, the k-Nearest Neighbors (kNN) algorithm has emerged as a powerful tool for classification tasks. With its simplicity and effectiveness, kNN has gained popularity across various domains. In this article, we will explore the fundamentals of kNN, its applications, and implementation. Whether you are a beginner or an experienced practitioner in machine learning, this article will provide valuable insights into the capabilities of kNN and how it can be leveraged to make accurate predictions. So, let's dive into the world of k-Nearest Neighbors and unlock its potential!

Keywords: KNN, Machine Learning, Classification, Supervised Learning

1. Introduction : MACHINE LEARNING

Machine learning is a field of study and practice that focuses on developing algorithms and models that enable computers to learn and make predictions or decisions without being explicitly programmed. It is a subset of artificial intelligence (AI) and involves the use of statistical techniques and computational models to analyze and interpret complex data patterns.

The goal of machine learning is to enable computers to learn from experience or historical data, and improve their performance or decision-making abilities over time. This is achieved by training machine learning models on large datasets, where the models learn patterns, relationships, and trends within the data. Once trained, these models can be used to make predictions, classify new data, or discover insights from previously unseen data.

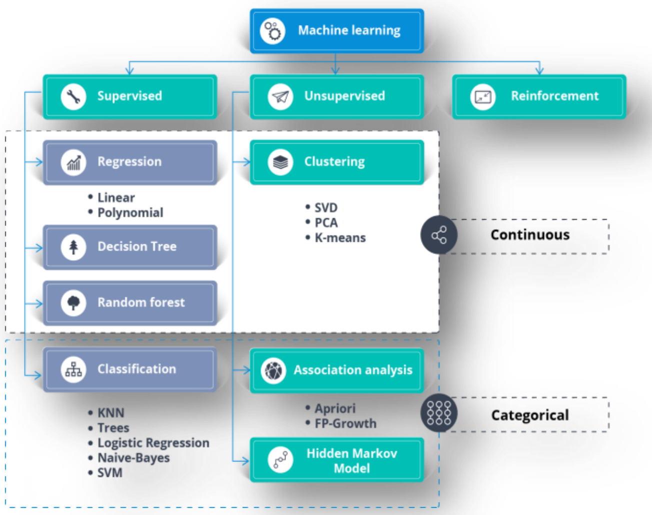
Machine learning can be categorized into three types:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Supervised Learning: In supervised learning, the model learns from labeled data to make predictions or classify new data. It uses input variables and corresponding output labels to find patterns and make accurate predictions.

Unsupervised Learning: Unsupervised learning deals with unlabeled data. The model discovers patterns and structures within the data without any predefined output labels. It helps in clustering similar data points or finding hidden relationships.

Reinforcement Learning: Reinforcement learning involves an agent learning to make decisions by interacting with an environment. It receives feedback in the form of rewards or penalties and learns to maximize its cumulative reward by optimizing its actions.

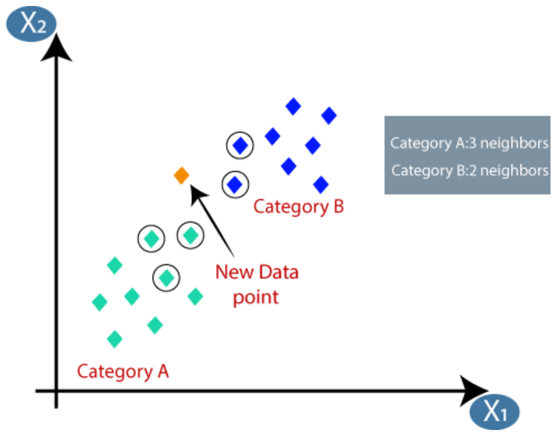


k-Nearest Neighbors (kNN)

k-Nearest Neighbors (kNN) is a machine learning algorithm that determines the class or value of a new data point based on its proximity to the neighboring data points in the training set. It belongs to the supervised learning type of algorithms, where the data is labelled and used to make predictions. The algorithm finds the "k" closest neighbors to the new data point and assigns the majority class or calculates the average value of their labels as the prediction. kNN is easy to understand and implement, making it a popular choice for simple classification and regression tasks.

UNDERSTANDING K-NEAREST NEIGHBORS (KNN)

The k-Nearest Neighbors (kNN) algorithm is a simple yet powerful classification algorithm in machine learning. It follows the principle that similar instances tend to belong to the same class. The kNN algorithm makes predictions based on the labels of its nearest neighbors in the feature space.



The key idea behind kNN is to measure the proximity or similarity between data points. This is done by calculating the distance between the feature vectors of the instances. The choice of distance metric, such as Euclidean distance or Manhattan distance, depends on the nature of the data and the problem at hand. The distance metric defines the notion of "closeness" between two data points.

In the kNN algorithm, the value of k determines the number of neighbors to consider when making a prediction. For example, if k is set to 5, the algorithm will find the five nearest neighbors to a given data point. The class label that occurs most frequently among these neighbors is assigned to the data point being classified.

One of the advantages of kNN is its simplicity and interpretability. The algorithm does not make strong assumptions about the underlying data distribution and can be easily implemented. Additionally, kNN allows for flexibility in choosing the value of k and the distance metric, making it adaptable to different scenarios.

However, there are some considerations and limitations when using kNN. The performance of kNN can be sensitive to the choice of k and thedistance metric. The algorithm may also be computationally expensive, especially when dealing with large datasets. Furthermore, kNN is more suitable for problems with balanced class distributions and continuous features.

APPLICATIONS OF K-NEAREST NEIGHBORS (KNN)

k-Nearest Neighbors (kNN) algorithm has found numerous applications across various domains. Its simplicity and versatility make it a popular choice for solving classification and regression problems. Some common real-world applications of kNN include:

1. Customer Segmentation
2. Recommendation Systems
3. Fraud Detection
4. Image and Speech Recognition
5. Medical Diagnosis

The versatility of kNN allows its application in a wide range of domains, from finance and marketing to healthcare and image processing. Its ability to handle both categorical and numerical data, along with its intuitive nature, makes it a valuable tool in various problem-solving scenarios.

IMPLEMENTING K-NEAREST NEIGHBORS (KNN)

To implement the k-Nearest Neighbors (kNN) algorithm, follow these steps:

Data Preprocessing:

- **Handle missing values:** Fill in or remove any missing values in the dataset.
- **Normalize the data:** Scale the features to a common range to ensure all features contribute equally to the distance calculations.
- **Split the data:** Divide the dataset into a training set and a test set for model evaluation.

Distance Calculation:

- **Select a distance metric:** Choose a suitable distance metric, such as Euclidean distance or Manhattan distance, based on the nature of the data.
- **Calculate distances:** Compute the distance between the test instance and each instance in the training set using the chosen distance metric.

Selecting the Value of k:

- **Determine the value of k**: Decide on the number of nearest neighbors (k) to consider when making predictions. This can be based on domain knowledge or using techniques like cross-validation to find the optimal value.

Finding Neighbors:

- **Identify the k nearest neighbors:** Sort the distances in ascending order and select the k instances with the shortest distances to the test instance.

Class Prediction:

- **Make predictions:** Assign the class label of the majority of the k nearest neighbors to the test instance. In the case of regression problems, calculate the average of the k nearest neighbors' values.

Evaluate the Model:

- **Assess the model's performance:** Use evaluation metrics such as accuracy, precision, recall, or mean squared error to measure the effectiveness of the kNN algorithm on the test set.

PERFORMANCE AND EVALUATION OF K-NEAREST NEIGHBORS (KNN)

When evaluating the performance of the k-Nearest Neighbors (kNN) algorithm, several metrics can be used to assess its effectiveness. These metrics provide insights into the model's predictive accuracy and its ability to correctly classify or predict instances. Some commonly used evaluation metrics include:

1. Accuracy:

- **Formula:** - $(Number\ of\ correctly\ classified\ instances)$ $/(Total\ number\ of\ instances)$
- Accuracy measures the overall correctness of the model's predictions by calculating the ratio of correctly classified instances to the total number of instances.

2. Precision:

- **Formula:**

$$(True\ Positives)\ /\ (True\ Positives + \ False\ Positives)$$

- Precision quantifies the proportion of correctly predicted positive instances out of all instances predicted as positive. It focuses on the accuracy of positive predictions.

3. Recall (Sensitivity):

- **Formula:**

$$(True\ Positives)\ /\ (True\ Positives + \ False\ Negatives)$$

- Recall calculates the proportion of correctly predicted positive instances out of all actual positive instances. It emphasizes the ability of the model to capture positive instances.

4. F1-score:

- **Formula:**

$$2 * (Precision * Recall) / (Precision + Recall)$$

- The F1-score is the harmonic mean of precision and recall. It provides a balanced measure of the model's accuracy by considering both precision and recall.

The choice of distance metric in kNN can significantly impact its performance. Different distance metrics, such as Euclidean distance, Manhattan distance, or cosine similarity, have varying effects on the calculated distances between instances. The choice of the distance metric should align with the characteristics of the data and the problem at hand.

Cross-validation is an essential technique in evaluating the performance of the kNN algorithm. It involves partitioning the data into multiple subsets (folds) and iteratively using each fold as a test set while training the model on the remaining folds. This allows for a more robust evaluation of the model's performance, reducing the risk of overfitting or underfitting.

Parameter tuning is another crucial aspect of kNN performance. The choice of the value of k, the number of nearest neighbors to consider, can impact the model's performance. Using cross-validation or other techniques, finding the optimal value of k can help maximize the model's accuracy and prevent overfitting or underfitting.

By considering these evaluation metrics, selecting appropriate distance metrics, employing cross-validation, and tuning the parameters, you can effectively assess and optimize the performance of the kNN algorithm for your specific classification or regression tasks.

CASE STUDY

Title: Categorizing Skill Levels using kNN

Statement: We begin by working with a dataset consisting of 100 instances, which includes information on aptitude, English proficiency, programming skills, and the corresponding skill levels. The skill levels range from novice to expert, representing different stages of skill development in a particular domain. The problem statement revolves around building a model that can predict the skill level of an individual based on their aptitude, English proficiency, and programming skills.

Before we begin, it's important to mention that the test data used in this project was collected from my class students via a Google Form. The goal of this implementation is to use the k-Nearest Neighbors (kNN) algorithm to predict the proficiency level of students in subjects like "Apptitude," "English," and "Programming" based on their skills.

To achieve this, we have a pre-existing training dataset that already contains these attributes along with the corresponding class labels indicating each student's proficiency level. The training dataset was prepared beforehand, ensuring it accurately represents the range of skill levels in the class.

For evaluating the performance of the kNN algorithm, we extracted the test data solely from the students' responses in the Google Form. It has the same attributes as the training dataset. The aim is to use the trained kNN model to predict the proficiency level of students in the test data by finding similar patterns in the training dataset.

By leveraging the kNN algorithm, we can make predictions for each student in the test data by considering the similarities between their skill levels and those of the students in the training dataset.

This implementation allows us to assess how well the kNN algorithm predicts the proficiency level of students based on their skills. It demonstrates the practical application of machine learning in education, where such predictions can help educators assess students and provide appropriate support.

Note: The implementation is carried out using Python programming language, leveraging libraries such as pandas, numpy, matplotlib, seaborn, and scikit-learn.

Implementation

1. Importing Required Libraries

```
Import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
```

2. Loading Dataset

```
#Loading Test and Train Datasets
 • Loading Training Dataset
 •Loading Testing Dataset
```

3. Data Preprocessing

```
• Checking Nulls, Duplicates
• Encode categorical variables if
  necessary.
```

Splitting the Dataset

```
#Splitting Input& Output Features
x_train -> Training Input Features
y_train -> Training OutputFeatures
x_test -> Test Input Features
y_test -> Test Output Features
```

Applying kNN:

```
#Fitting Input and Output Features
from sklearn.neighbors import KNeighb
orsClassifier
knn=KNeighborsClassifier()
knn.fit(x_train, y_train)
```

Predicting Skill Level:

```
#Predicting the Skill Level
y_pred=knn.predict(x_test)
x_test['Skill Level ^']=y_pred
x_test
```

Performance Of The Model -

```
#Accuracy
from sklearn.metrics import accuracy_
score
print('Model Accuracy Score - {0: 0.4
f}'.format(accuracy_score(y_test, y_p
red)))
```

Output –

```
Model Accuracy Score -  0.9355
```

Inference:

Based on the provided dataset, the following table represents the predicted skill levels for each student:

Apptitude	English	Programming	Skill Level ^
5	5	5	Genius
4	5	5	Genius
4	5	4	Genius
4	4	3	Lexical Logician
2	5	4	Code Linguist

4	5	2	Lexical Logician
5	5	5	Genius
4	5	4	Genius
4	5	2	Lexical Logician
4	4	0	Lexical Logician
5	5	5	Genius
5	5	4	Genius
3	5	3	Speaker
3	4	3	Novice
4	4	3	Lexical Logician
1	4	2	Speaker
0	2	1	Novice
5	5	4	Genius
4	5	5	Genius
5	5	3	Lexical Logician
5	5	4	Genius
4	5	1	Lexical Logician
2	1	1	Novice
1	3	3	Novice

This table displays the predicted skill levels for each student based on the attributes "Aptitude," "English," and "Programming." The predicted skill levels include categories such as "Genius," "Lexical Logician," "Code Linguist," "Speaker," and "Novice." These predictions were made using the implemented k-Nearest Neighbors (kNN) algorithm, which considered the similarities between the students' skill levels and those in the training dataset to assign the appropriate skill level category.

By examining the predicted skill levels, we can gain insights into the proficiency of each student in different subjects. This information can be valuable for educators and stakeholders in identifying areas where students excel or require additional support. The kNN algorithm's ability to predict skill levels based on similar instances demonstrates its potential in educational contexts for personalized learning and student assessment.

Conclusion

The k-Nearest Neighbors (kNN) model achieved an accuracy of **93.55%** on the given dataset. This indicates that the kNN algorithm performed well in classifying the instances based on their features. The accuracy score demonstrates the effectiveness of the kNN model in predicting the class labels of the test data. The kNN algorithm's simplicity and interpretability make it a suitable choice for various classification tasks.

ACKNOWLEDGMENTS

We are deeply grateful to **Dr. Kompalli Udaya Sri**, M.B.A., M.Sc.,(IT), M.Tech.(CSE), MA(Astro), Ph.D., Gold Medal, World Book of Records, FSIESRP, our mentor and guide, for her exceptional guidance and unwavering support throughout this project. Her profound expertise and insightful feedback have been invaluable in shaping our research and achieving our goals. Her expertise in the field of Artificial Intelligence and Machine Learning has been instrumental in shaping our understanding and driving the success of our work.

Furthermore, we would like to express our appreciation to our peers and fellow students for their encouragement and collaboration. Their constructive discussions and feedback have greatly enriched our work and inspired us to push the boundaries of our knowledge.

AUTHORS

- Avanigadda Sivayya (216202P)
- Gandham Mani Saketh (216221P)
- Abdul Faheem (216270P)

We, the authors of this article, are second-year **B.Sc.** students specializing in **Artificial Intelligence and Machine Learning** in the **Department of Computer Science** at **P B Siddhartha College of Arts and Science.** With a keen interest in exploring the applications of machine learning algorithms, we collaborated to investigate and implement the k-Nearest Neighbors (kNN) classification algorithm.

Our collective efforts and passion for the subject have driven us to conduct this study and share our findings. By combining our knowledge and skills in AI and ML, we have delved into the intricacies of the kNN algorithm, analyzed its performance, and explored its applications in predicting skill levels of students.

It is our sincere hope that this article provides valuable insights and contributes to the field of machine learning. We extend our gratitude to our peers for their constant support and encouragement throughout this endeavor.

BIBLIOGRAPHY

Survey Form:

https://docs.google.com/forms/d/e/1FAIpQLSdpT4P3DXF2Ttjj2QHeSpSnXz1OrEYAX6KP6VwHa ywNVgJ6g/viewform?usp=sf link

1. N. Kola, M. Kumar, “Supervised Learning Algorithms of Machine Learning: Prediction of Brand Loyalty,” International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 8, no. 11, pp. 3886-3889, 2019
2. Muhammad, Z. Yan, “Machine Learning Approaches: A Survey,” ICTACT Journal on Soft Computing, vol. 5, no. 3, pp. 946-952, 2015.
3. A. Juyal, C. Pande, “Performance Analysis of Supervised Machine Learning Algorithms on Medical Dataset,” International Journal of Recent Technology and Engineering (IJRTE), vol. 8, no. 6, pp. 1637-1642, 2020.