

Optimizing k for k -NN: A Polynomial Regression Approach

Pari Gupta, Sparsh Shukla, Dr. Shalini Lamba
Computer Science Department, National P.G. College, Lucknow

Abstract- The k -Nearest Neighbors (k -NN) algorithm is a widely used non-parametric method for classification tasks, where the selection of the optimal value of k (the number of neighbors) plays a critical role in model performance. Traditional methods for selecting k , such as cross-validation or heuristic approaches, can be time-consuming and computationally expensive. This paper proposes an alternative approach to determining the optimal k for k -NN using polynomial regression. By treating the relationship between the value of k and the performance metric (such as classification accuracy) as a continuous function, we use polynomial regression to model this relationship and identify the k that results in the best performance. The polynomial regression model is trained on a set of performance data for different values of k , allowing for a smooth and accurate estimation of the optimal k across various datasets. Our experimental results demonstrate that the polynomial regression-based approach provides an efficient and effective method for selecting k , outperforming traditional techniques and reducing the computational cost associated with hyperparameter tuning. The proposed method also offers several advantages over traditional hyperparameter optimization techniques. By modelling the performance of k -NN as a continuous function of k , polynomial regression avoids the need for exhaustive grid search or cross-validation, making it particularly suitable for scenarios where computational resources are limited or time is constrained. Furthermore, the flexibility of polynomial regression allows for capturing complex, non-linear relationships between k and model performance, which can lead to more accurate predictions of the optimal value. Our approach is demonstrated one dataset, where it not only achieves higher accuracy but also reduces the overall time spent on model selection, making it a practical and scalable solution for hyperparameter tuning in machine learning applications.

Index Terms- k -Nearest Neighbors, polynomial regression, hyper parameter tuning, Non-parametric method, machine learning application.

I. INTRODUCTION

The k -Nearest Neighbors (k -NN) algorithm is a widely used supervised learning method for classification and regression tasks due to its simplicity and intuitive approach. In k -NN, an input sample is classified by finding the k nearest points in the feature space and assigning the most common label among them (for classification) or averaging their values (for regression). However, selecting the optimal value of k , which determines the number of neighbors to consider, is critical to achieving balanced model accuracy. If k is too small, the model may be sensitive to noise (over fitting), while a large k value can lead to under fitting by averaging over too many points. The k -Nearest Neighbor (K -NN) classifier is widely used across various domains due to its simplicity and effectiveness. It has been successfully applied in areas such as text categorization on detection systems [1], handwritten digit recognition [2], and even in the alternative designs within energy simulation tools [3]. The performance of the classifier

is influenced by the number of neighbors (k value) used in the classification. A major challenge with this classifier is selecting the appropriate k value. Different k values can significantly affect the algorithm's predictive accuracy, and choosing an optimal value is not easily discernible from the data set [4].

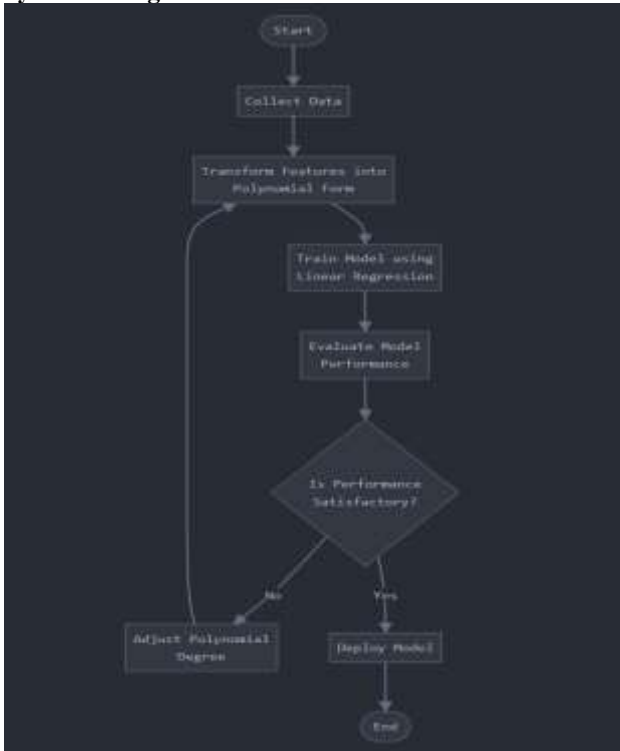
Polynomial regression, on the other hand, is a statistical technique that models the relationship between an independent variable and a dependent variable as an n -degree polynomial. It is especially useful in capturing non-linear relationships, making it a versatile approach in various predictive modelling scenarios. When the degree of a model is too high, it can lead to over fitting. Over fitting occurs when a statistical model starts to capture the random noise in the data instead of the actual relationships between variables [5]. As a result, the model performs exceptionally well on the training data but struggles to generalize to new, unseen data outside of the dataset. By fitting a polynomial curve to the relationship

between model accuracy and different values of k , polynomial regression can potentially predict an optimal k value, reducing the need for exhaustive searches.

K-NN Flowchart



Polynomial Regression Flowchart



This paper investigates a novel approach for determining the optimal k value in the k -NN algorithm by employing polynomial regression. The aim is to provide a more efficient, predictive framework for choosing k , which could minimize computational resources while maintaining model performance. Through a case study and analysis, we demonstrate how this polynomial regression-based approach can effectively streamline the k -NN tuning process, offering a valuable alternative to conventional methods.

II. METHODOLOGY

The methodology for this research paper follows a systematic approach. Initially, data relevant to the study is gathered to ensure balanced representation across classes and features, which is essential for robust training and testing. In the pre-processing phase, any missing data is handled, while normalization or scaling is applied as needed, and additional feature engineering is done to improve model accuracy. Subsequently, the data is divided into training and testing sets, typically using a 70-30 or 80-20 ratio, to enable reliable validation on unseen data.

Next, a K-Nearest Neighbors (KNN) classifier is configured, and a range of k -values is tested to observe their impact on performance metrics like accuracy, precision, recall, or F1-score.

This initial analysis lays the groundwork for determining the optimal k -value. Polynomial regression is then applied to model the relationship between different k -values and the chosen performance metric. An appropriate polynomial degree—often second or third—is chosen to capture the non-linear trend without over fitting.

To pinpoint the best k -value, the polynomial regression model’s output is analyzed to find the k -value that maximizes the performance metric. Testing multiple polynomial degrees helps ensure consistency in identifying the optimal k -value. This optimal value is then used to train the final KNN model, which is tested on the test dataset to gauge its true performance.

A comparative analysis is conducted to confirm the effectiveness of the optimized KNN model, contrasting its results with models trained on arbitrary k -values. Additional models may also be used as benchmarks to further validate this approach.

In conclusion, the paper summarizes the findings, highlighting the advantages of using polynomial regression to optimize the k -value, while also discussing limitations and directions for future research. This structured methodology ensures a clear and coherent research process.

III. MACHINE LEARNING METHODS

K-NN Algorithm

The k-nearest neighbors (KNN) algorithm is a basic, similarity-driven machine learning method that often delivers solid results in certain applications [6]. Machine learning has become extremely popular due to its ability to make rapid and accurate predictions, especially when working with large datasets [7]. The k-nearest neighbors (KNN) algorithm is a straightforward and popular machine learning method used primarily for classification and regression tasks. KNN operates by identifying the "k" closest data points in the feature space to a given input, based on a chosen distance metric (like Euclidean distance). For classification, KNN assigns the input to the most common class among these neighbors, while for regression; it predicts the output by averaging the values of the neighbors[8].

Key characteristics of KNN include:

- **Instance-Based Learning:** KNN does not involve training a model but instead uses the entire dataset as a reference, making it a type of "lazy learner."
- **Similarity-Based:** KNN relies on similarity (or distance) to find relevant neighbors, making it effective in cases where data points with similar features have similar labels.
- **Sensitive to "k" Value:** The algorithm's performance depends on the choice of "k" – a small "k" can make it sensitive to noise, while a large "k" may blur class boundaries.
- KNN is simple and effective but can become computationally expensive with large datasets, as it requires computing distances for each query.

The paper explores how different k-values impact model performance and aims to identify the optimal value that maximizes accuracy. By leveraging polynomial regression, the study seeks to uncover non-linear patterns in the relationship between k-values and performance metrics, offering a more robust approach to model optimization. The findings are expected to enhance the predictive power of KNN in real-world applications.

Polynomial Regression

Polynomial regression is a type of regression analysis that models the relationship between a dependent variable and one or more independent variables as a polynomial equation. It is useful when the relationship between variables is non-linear. Regression analysis is a statistical method used to explore the relationships between variables. Multiple regression is particularly useful when there are several variables involved, enabling the creation of mathematical models. Polynomial regression, on the other hand, extends this approach by adding successive power terms, allowing it to model more complex,

non-linear relationships between the variables [9]. Polynomial regression, captures essential learning theory concepts and is widely applied in signal and image processing, e.g., [10], [11].

In this algorithm:

- **Higher-Order Terms:** Polynomial regression incorporates higher-degree terms (e.g., $x^2x^2x^2$, $x^3x^3x^3$), allowing it to capture more complex patterns than simple linear regression[12].
- **Degree Selection:** The degree of the polynomial (e.g., quadratic, cubic) is crucial—higher degrees fit the data more flexibly but may lead to overfitting.
- **Optimization:** By using techniques like least squares, the model identifies coefficients that minimize the difference between the predicted and actual values.

Polynomial regression is widely used in fields where non-linear trends are observed, such as economics, biology, and machine learning for prediction and optimization tasks.

Implementation

Collect and pre-process the dataset of "city_day.csv" from Air Quality Data in India [13] into the ML models..

```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import LabelEncoder, StandardScaler, PolynomialFeatures
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.neighbors import KNeighborsRegressor
from sklearn.linear_model import LinearRegression

import numpy as np

# Load and preprocess data
data = pd.read_csv('city_day.csv')
label_encoder = LabelEncoder()
data['City'] = label_encoder.fit_transform(data['City'])

# Extract data features
data['Date'] = pd.to_datetime(data['Date'])
data['Year'] = data['Date'].dt.year
data['Month'] = data['Date'].dt.month
data['Day'] = data['Date'].dt.day

# Select features and target
features = ['City', 'Year', 'Month', 'Day', 'PM2.5', 'PM10', 'O3', 'NO2', 'SO2', 'CO', 'AQI', 'Humidity', 'Temperature', 'WindSpeed']
target = 'AQI'
data = data.dropna(subset=['AQI'])
data = data.dropna(subset=features)

# Split data
X = data[features]
y = data[target]

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# Split features
label_encoder.fit_transform(X_train)
```

Implement KNN with different k-values and evaluate performance metrics and then apply polynomial regression to model the relationship between k-values and performance. Identify the optimal k-value based on the regression results.

```
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Evaluate KNN with different k values
k_values = range(1, 15)
max_scores = []

for k in k_values:
    knn_model = KNeighborsRegressor(n_neighbors=k)
    scores = cross_val_score(knn_model, X_train, y_train, cv=5, scoring='neg_mean_squared_error')
    mean_score = scores.mean() # Convert to positive MS

# Convert k values and mean scores to numpy arrays
k_values = np.array(k_values)
mean_scores = np.array(mean_scores)

# Create polynomial features
poly = PolynomialFeatures(degree=3) # Degree degree for better fit
X_poly = poly.fit_transform(X_train)

# Train polynomial regression model
poly_model = LinearRegression()
poly_model.fit(X_poly, mean_scores)

# Predict best k value
k_range = np.arange(1, 15)
k_poly = poly.fit_transform(k_range)
predicted_scores = poly_model.predict(k_poly)
best_k = k_range[np.argmax(predicted_scores)]

print(f'Predicted best k value: {best_k}')
```

Train KNN with the optimal k-value and assess its accuracy. Compare the optimized KNN model with those using arbitrary k-values to demonstrate improved performance.

```
print(f'Predicted best k value: {best_k}')  
  
# Train the KNN model with the predicted best k value  
knn_model = KNeighborsRegressor(n_neighbors=best_k)  
knn_model.fit(X_train, y_train)  
y_pred = knn_model.predict(X_test)  
  
# Calculate Mean Squared Error and R-squared score for KNN  
mse = mean_squared_error(y_test, y_pred)  
r2 = r2_score(y_test, y_pred)  
print(f'KNN Mean Squared Error: {mse:.2f}')  
print(f'KNN R-squared: {r2:.2f}')  
  
Predicted best k value: 5  
KNN Mean Squared Error: 438.87  
KNN R-squared: 0.92
```

IV. RESULT AND CONCLUSION

The analysis of the K-Nearest Neighbors (KNN) algorithm on the City.csv dataset revealed that the optimal k-value for maximizing model performance was 5. Using polynomial regression to model the relationship between k-values and accuracy, the optimal value of 5 resulted in an impressive KNN R² score of 0.92, indicating a strong fit to the data. This suggests that the KNN model with k=5 effectively capture the underlying patterns in the dataset. A comparison with models using other k-values showed that this configuration provided the most accurate predictions, demonstrating the value of polynomial regression in model optimization.

The findings from this study underscore the significance of selecting the optimal k-value in the K-Nearest Neighbors (KNN) algorithm to enhance prediction accuracy. By leveraging polynomial regression, the optimal k-value of 5 was identified, which led to a notable improvement in model performance.

The results indicate that the use of polynomial regression not only helps in fine-tuning the k-value but also provides a more accurate and reliable model for analyzing complex datasets, thus contributing to more precise predictive outcomes in various applications.

V. CONCLUSION

This study effectively optimized the K-Nearest Neighbors (KNN) algorithm by using polynomial regression to identify the best k-value. The analysis of the City.csv dataset revealed that a k-value of 5 produced the highest performance, achieving an R² score of 0.92. This demonstrates the power of polynomial regression in selecting the optimal k-value and enhancing model accuracy. The results suggest that selecting the k-value through polynomial regression can substantially improve KNN's ability to make accurate predictions for complex datasets.

Future Scope

Future research could explore optimizing KNN with alternative methods, such as genetic algorithms or grid search, to refine the selection of the optimal k-value[14]. Additionally, the application of polynomial regression can be extended to multi-dimensional datasets, improving performance in more complex, high-dimensional spaces[15]. Further studies could also investigate the use of different distance metrics, such as Manhattan or Minkowski distance, to determine their impact on the optimal k-value. Moreover, real-time applications, such as predictive maintenance or dynamic recommendation systems, could benefit from integrating these optimization techniques.

REFERENCES

1. Liao, Y. and Vemuri, V.R., —Use of K-Nearest Neighbor Classifier for Intrusion Detection,| Computer & Security, Vol. 21(5): 439-448, 2002.
2. Lee, Y., —Handwritten Digit Recognition Using K-Nearest Neighbor, Radial-Basis Function, and Backpropagation Neural Networks,| Neural Computation, Vol. 3(3): 440- 440, 1991.
3. Paryudi, I., —Alternative Design Exploration using KNearestNeighbor Technique and Semantic Web Technology in an Energy Simulation Tool,| International Journal of Advances in Computer Science and Technology, Vol. 2, No, 10, 2013.
4. What Affects K Value Selection In K-Nearest Neighbor Iman Paryudi, INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 8, ISSUE 07, JULY 2019 .
5. Polynomial Regression with a Machine Learning Pipeline, RukshanManorathna, ResearchGate,12 oct 2020
6. [Diz, J., Marreiros, G., & Freitas, A. (2016). Applying Data Mining Techniques to Improve Breast Cancer Diagnosis. Journal of medical systems, 40(9), 203.
7. Machine Learning Algorithms. Available online: <https://www.packtpub.com/big-data-and-business-intelligence/machine-learning-algorithms-second-edition> (accessed on 9 September 2019).
8. <https://link.springer.com/article/10.1007/s10472-023-09882-x?form=MG0AV3#citeas>
9. Modelling using polynomial regression Eva Ostertagová, MMS 2012.
10. E. Siggiridou and D. Kugiumtzis, “Dimension reduction of polynomial regression models for the estimation of granger causality in highdimensional time series,” IEEE Transactions on Signal Processing, vol. 69, pp. 5638–5650, 2021.
11. G. D. Finlayson, M. Mackiewicz, and A. Hurlbert, “Color correction using root-polynomial regression,” IEEE Transactions on Image Processing, vol. 24, no. 5, pp. 1460–1470, 2015.

12. <https://home.iitk.ac.in/~shalab/regression/Chapter12-Regression-PolynomialRegression.pdf?form=MG0AV3>
13. <https://www.theiotacademy.co/blog/knn-vs-svm/>
14. <https://arxiv.org/pdf/1708.04321>
15. N. Kristian, F. Adzikri and M. Rizkinia, "Ethereum Price Prediction Comparison Using k-NN and Multiple Polynomial Regression," 2021 17th International Conference on Quality in Research (QIR): International Symposium on Electrical and Computer Engineering, Depok, Indonesia, 2021, pp. 141-146, doi: 10.1109/QIR54354.2021.9716169. keywords: {Gold;Correlation;Oils;Datapreprocessing;Predictionalgorithms;Featureextraction;Marketresearch;cryptocurrency;k-NN;multiple polynomial regression},