Predicting Poker Hands with KNN and Logistic Regression

By: Andy Pham

Introduction:

a. Problem Statement

The goal of this project is to build a machine learning model that can classify poker hands into their correct categories using only the card suits and ranks as input. The Poker Hand Dataset contains ten possible outcomes ranging from no hand to a royal flush, which makes this a supervised classification problem. The challenge is to determine whether a machine can learn to recognize poker hand patterns accurately.

b. Motivation

I selected this topic because the logic behind poker hand recognition is more complex than it looks. Humans can usually recognize pairs or high cards quickly, but identifying patterns like straights, flushes, or full houses requires comparing suits, ranks, and combinations of cards. A machine has to learn these relationships through examples, and that creates challenges since the data is heavily imbalanced. Most poker hands contain nothing significant while rare hands such as straight flush or royal flush occur extremely infrequently. This imbalance makes it difficult for a model to predict strong hands accurately and gives me a chance to analyze the strengths and weaknesses of different algorithms.

c. Approach

To approach this problem, I tested two machine learning models, K Nearest Neighbors and Logistic Regression. Both models were trained using the same card features which allowed me to compare their performance fairly. KNN attempts to classify hands based on similarity to other hands, while Logistic Regression tries to create decision boundaries that separate one hand type from another. Throughout this project I performed exploratory data analysis, prepared the data for training, and evaluated each model to see how well it could classify poker hands. The goal of this report is to present my findings, explain what worked and what did not, and reflect on what I learned during the process.

Data:

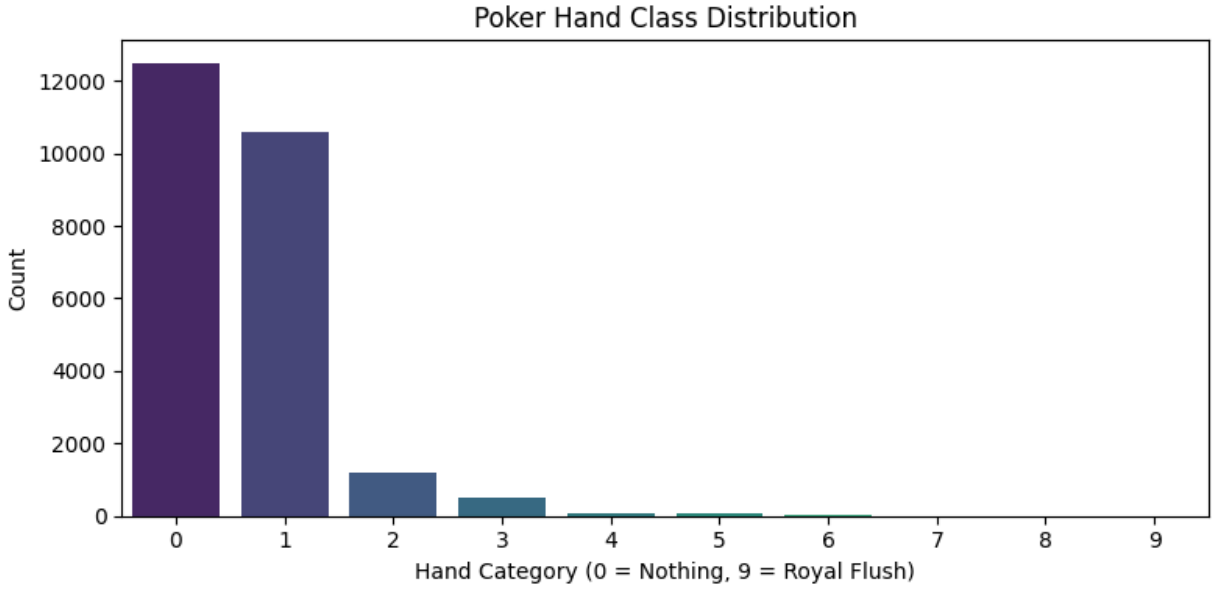
a. Dataset Introduction

The dataset used in this project is the Poker Hand Dataset from the UCI Machine Learning Repository. It contains examples of randomly dealt five card poker hands, and each record includes ten features representing the suit and rank of each of the five cards. Suits are encoded as values from one to four, and ranks are encoded from one to thirteen. The eleventh column represents the final hand category, which ranges from zero to nine. Zero means the hand contains no meaningful combination while nine represents a royal flush. The dataset is already numeric which made it convenient to work with for classification.

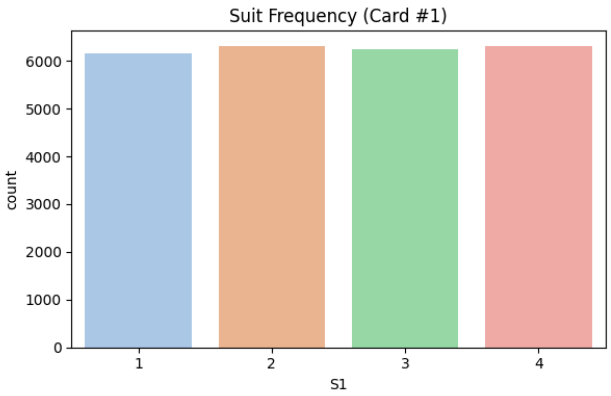
The training file contains 25,010 rows and 11 columns, while the testing file contains one million rows. Each row represents a unique poker hand, and every column contains exactly one value per card attribute which means the data has no missing entries. Summary statistics confirmed that the average suit value is around two point five and the average rank value is near seven which aligns with the midpoint of their ranges. This suggests that suits and ranks are evenly distributed overall. The mean of the class label is low because most hands belong to class zero meaning most hands are not strong hands. A count plot of the class distribution clearly showed a strong imbalance where class zero dominates the dataset, and extremely rare hands like straight flush and royal flush appear in very small numbers. This imbalance later impacted model performance and became an important factor during evaluation.

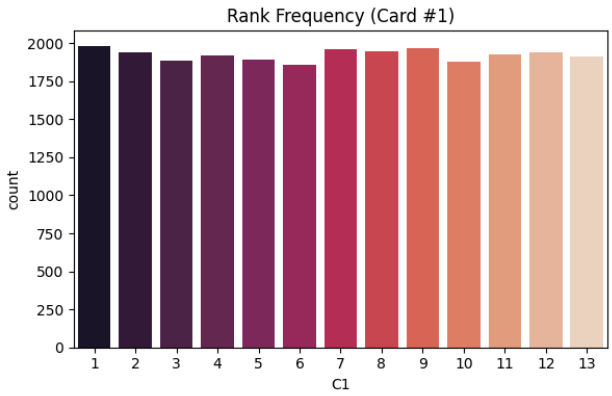
b. Visual Analysis

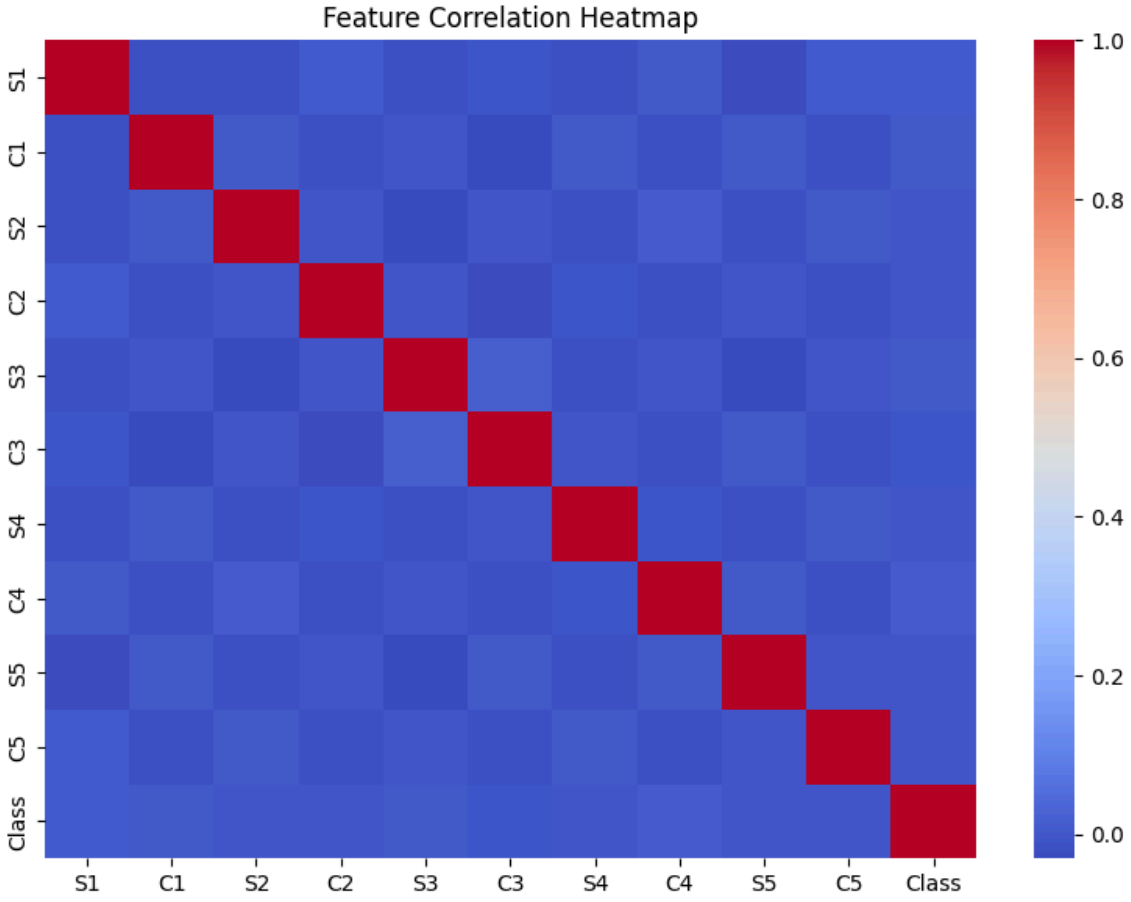
To better understand the data, I generated multiple visualizations. A class distribution plot revealed that the dataset is heavily imbalanced. Most samples fall into class zero, meaning the majority of hands contain no strong combinations. Rare outcomes such as straight flush or royal flush appear very infrequently. This imbalance later affected model performance, especially when predicting uncommon hand types.



I also plotted suit and rank frequencies for the first card column. The distributions were fairly even, with only slight natural variation, which suggests random card dealing. A correlation heatmap was created to observe relationships between features. The heatmap showed little correlation between suits and ranks, which is expected since each card is drawn independently. This indicates that poker hand recognition does not depend on single card values but rather on combinations of cards, which increases the difficulty of the classification task.







c. Preprocessing

Before training the models, I separated the input features from the output class. The first ten columns (S1, C1 through S5, C5) were used as features, and the class column was used as the target label. I split the training dataset into training and validation sets using an 80 to 20 split with stratification to maintain class distribution. The features were then scaled using StandardScaler. Scaling was important since K Nearest Neighbors calculates distances

between samples and Logistic Regression generally performs better when features are on a similar scale. After preprocessing, the data was prepared and ready for model training and evaluation.

Methods:

This project evaluates two machine learning models for poker hand classification: K Nearest Neighbors and Logistic Regression. Both models were trained using the scaled feature data and were evaluated using validation and test sets to compare their performance in predicting poker hands.

a. K Nearest Neighbors (KNN)

K Nearest Neighbors is a supervised learning algorithm that classifies a sample based on the majority label among its closest neighbors. The model determines similarity using distance in feature space. Since poker hands are represented by numerical values for suits and ranks, KNN can classify a hand by comparing it to hands that appear similar. One advantage of KNN is that it does not assume a linear relationship in the data. This makes it useful for problems where patterns are based on combinations of features, such as detecting pairs or flushes. KNN is sensitive to feature scaling, so the data was standardized before training. Multiple values for k were tested to find the best performing version of the model, and k = 9 achieved the highest validation accuracy.

b. Logistic Regression

Logistic Regression is a linear classification method that predicts class labels by estimating probability through the logistic function. In this project, a multinomial Logistic Regression model was used to handle all ten poker hand classes. The model attempts to separate classes by drawing decision boundaries between them. Logistic Regression is simple, interpretable, and serves as a strong baseline model even when the underlying relationships are not linear. Just like KNN, Logistic Regression benefits from feature scaling, so standardized training data was used. Although it can struggle with non linear patterns, including the many combinations involved in poker, it provides a clear comparison to evaluate how more flexible models perform.

c. Training Procedure

Both models were trained on the scaled training data and evaluated using the validation set to measure performance. After identifying the best k value for KNN, both models were tested on the full test dataset containing one million hands. Accuracy, classification reports, and confusion matrices were used to analyze the results. By comparing both models side by side, I was able to observe how well each one performed and what challenges they faced when predicting rare poker hands.

Results:

After training both models on the scaled poker hand dataset, I compared their performance using validation accuracy, test accuracy, and confusion matrix results. The K Nearest Neighbors model was tuned using several k values, and k = 9 produced the highest validation accuracy. Logistic Regression was trained using a multinomial classifier and evaluated using the same data split for a fair comparison.

a. Model Performance

KNN achieved the highest validation and test accuracy, performing slightly better than Logistic Regression. Both models performed above random guessing, but neither reached very high accuracy due to the complexity of poker hand classification and the imbalance of the dataset.

```
k = 1: Validation Accuracy = 0.4744
k = 3: Validation Accuracy = 0.4990
k = 5: Validation Accuracy = 0.5078
k = 7: Validation Accuracy = 0.5124
k = 9: Validation Accuracy = 0.5264
k = 11: Validation Accuracy = 0.5216
```

```
KNN (k=9) Test Accuracy: 0.5198
```

```
Logistic Regression Validation Accuracy: 0.4996
```

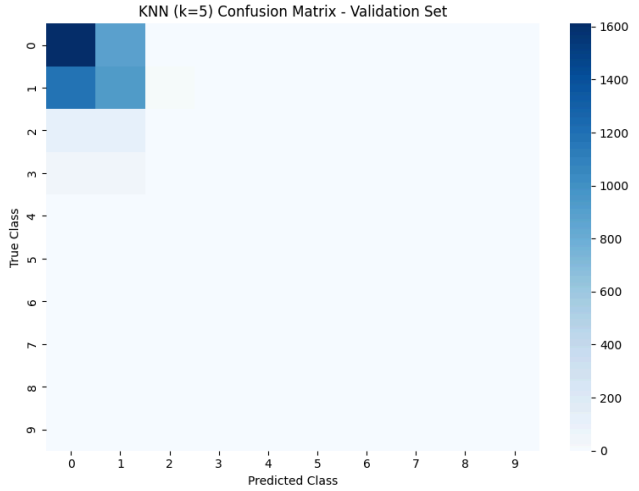
```
Logistic Regression Test Accuracy: 0.5012
```

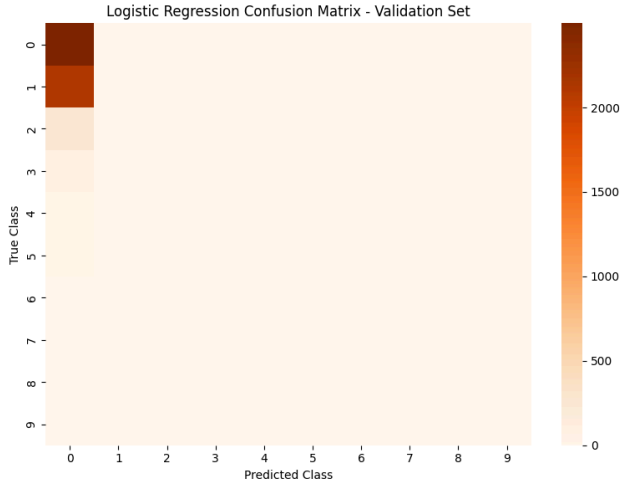
b. Observations from Evaluation

The class distribution plot showed that most hands are labeled as Class 0, which means no significant hand. Both models reflected this imbalance by predicting Class 0 more frequently than other classes. This led to good accuracy for common hands but poor accuracy for rare hands such as straight flush or royal flush. The classification reports and confusion matrices supported this observation. Most mistaken predictions occurred between classes with very low sample frequency, suggesting that the models struggled to learn patterns for rare hand types.

KNN performed better overall because it classifies based on similarity between hands. Poker hand recognition relies on combinations of card ranks and suits, which makes pattern based learning more effective for this task. Logistic Regression uses linear decision boundaries, which means it has difficulty capturing complex relationships such as three of a kind versus full house. As a result, Logistic Regression produced slightly lower performance on both the validation and test datasets.

The confusion matrices below show how each model performed across the ten poker classes. Most correct predictions occur in Class 0, which is expected due to class imbalance. Rare classes show sparse values across predictions.





c. Result Summary

The results show that K Nearest Neighbors is more suitable than Logistic Regression for poker hand classification in this experiment. While both models were limited due to class imbalance, KNN handled non linear relationships better and achieved higher accuracy. These results suggest that future models could benefit from feature engineering or more advanced classification approaches such as tree based models or ensemble methods.

Conclusion:

This project explored whether machine learning models could correctly classify poker hands based on card suits and ranks. Using the Poker Hand Dataset from the UCI Machine Learning Repository, I trained and evaluated two supervised models: K Nearest Neighbors and Logistic Regression. Through data exploration, preprocessing, and testing, I was able to compare how each model handled the classification task.

KNN produced better results than Logistic Regression with a validation accuracy of 52.64 percent and a test accuracy of 51.98 percent, while Logistic Regression achieved around 50 percent on both validations. These results suggest that KNN is better suited for poker hand prediction because it uses similarity between card patterns rather than relying on linear decision boundaries. Both models performed reasonably well on common hands but struggled with rare outcomes due to severe class imbalance in the dataset.

From this project, I learned the importance of data distribution and how imbalance can strongly influence model predictions. I also learned how scaling affects distance based algorithms like KNN and how model selection plays a critical role when relationships in the data are not linear. Although the models did not achieve very high accuracy, the project helped me understand the full machine learning workflow including data analysis, preprocessing, training, evaluation, and reflection.

If I were to continue this project, I would experiment with more advanced models such as Random Forest or Gradient Boosting, and I would consider applying class balancing techniques or feature engineering that detects pairs, flushes, and rank frequency patterns. These improvements could help the model understand poker relationships more effectively and increase prediction accuracy.

Git-Repository Link: https://github.com/AxPSaucey/ITCS-3156-ML-Final-Report

Reference

Lichman, M. "Poker Hand Dataset." UCI Machine Learning Repository, archive.ics.uci.edu/dataset/158/poker+hand. Accessed 2024.

Pedregosa, Fabian, et al. “Scikit-learn: Machine Learning in Python.” Journal of Machine Learning Research, vol. 12, 2011, pp. 2825–2830.

Seaborn Statistical Visualization Library. seaborn.pydata.org.

McKinney, Wes. “Data Structures for Statistical Computing in Python.” Proceedings of the 9th Python in Science Conference, 2010.

Acknowledgement

I used online documentation for Python libraries including pandas, scikit-learn, and seaborn for syntax reference when writing code. I also used ChatGPT as assistance for structuring the project report, generating code snippets, and explaining machine learning concepts during development. All decisions, interpretations, and final report writing were done by me.