Table of Contents

Hackathon Report

Introduction

The actual sales data that were obtained consists of customers' records, product characteristics, and sales reports. The main aim of this report was to classify customers according to their gender (CustomerGender) with the help of sales and customer characteristics. This classification could be utilised for marketing and sales promotions, market segmentation and the improvement of sales results.

The analysis focused on:

1. Data cleaning and data preprocessing where all the dust is removed in order to deal with all the gaps and the irregularities within the given set.

2. KNN model, Random forest model used in building & training of the model to predict gender of customer.

3. The process of evaluating these models to pin point the best classifier in executing the classification task.

Methodology

1. Data Preprocessing

To ensure robust and accurate modeling, the dataset underwent extensive preprocessing:

1. **Handling Missing Values:**
    - Critical missing values (e.g., CustomerID) were dropped.

- Missing values in numerical columns (Quantity, UnitPrice, etc.) were filled with the median to avoid skewing the data.
- Categorical columns were imputed using the mode.

2. **Data Transformation:**
   - **Date Conversion:** InvoiceDate was converted to integers for the sake of order preservation of temporal data.
   - **Categorical Encoding:**
     - CustomerGender was encoded using LabelEncoder (0 for Male, 1 for Female).
     - Country was one-hot encoded, creating dummy variables for each country.
3. **Feature Scaling:**
   - Standardization was applied on quantity, unit price, and customer age, which helps to bring all feature data to a more standard scale especially for distance measure algorithms like KNN.
4. **Duplicate Removal:**
   - o The records were cleaned by deleting duplicated rows.

2. Model Selection

Two machine learning models were selected based on their relevance and effectiveness for classification tasks:

1. **K-Nearest Neighbors (KNN):**
   - Chosen for its simplicity and to be able to identify complex, non-linear forms of relationship between variable.
   - K, the number of neighbors, has been set at 5 because it provides good, non-biased variance between the dataset groups. The number of neighbors (k) was set to 5, as it balances bias and variance.
2. **Random Forest:**
   - A powerful type of model that could work with the interactions of high dimensions.
   - The number of trees that define the forests was set to 100 because a higher number may be computationally expensive to calculate but would improve the model's accuracy. trees (n_estimators) was set to 100 for a good tradeoff between accuracy and computational efficiency.

3. Train-Test Split

- The dataset was split into training (80%) and testing (20%) sets using train_test_split with a fixed random state for reproducibility.

4. Evaluation Metrics

To assess model performance:

- **Accuracy:** Proportion of correct predictions.
- **Precision:** Ability to avoid false positives.
- **Recall:** Ability to identify all true positives.
- **F1 Score:** Harmonic mean of precision and recall.
- **Confusion Matrix:** Visual representation of true and false predictions across classes.

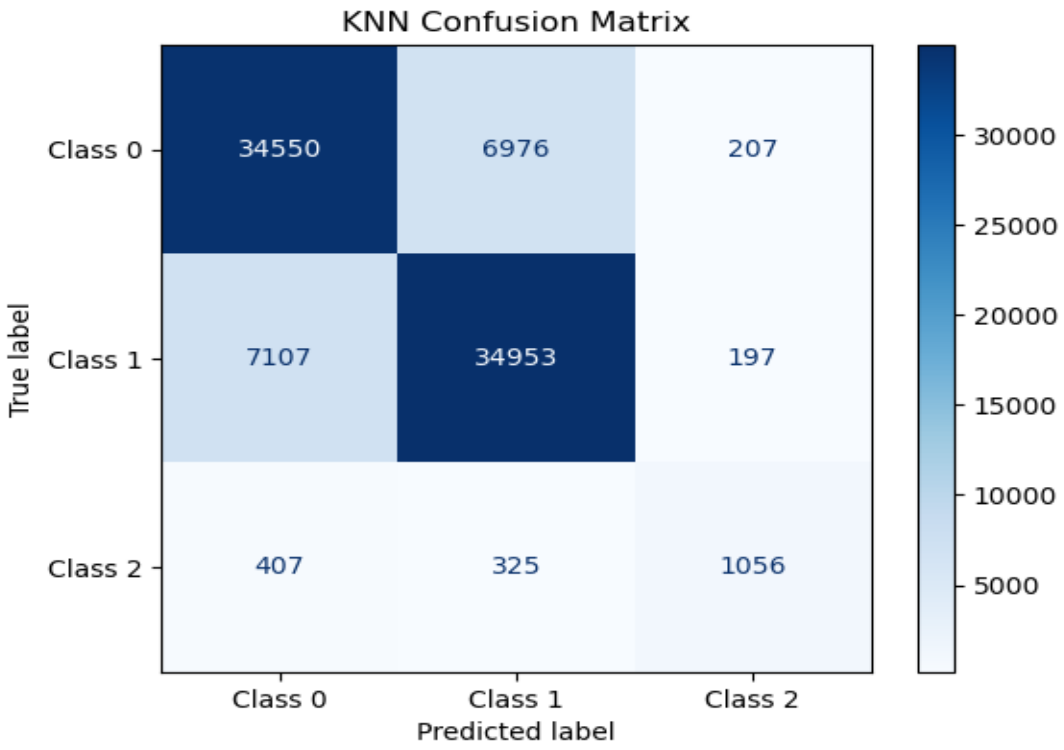
Analysis and Results

1. K-Nearest Neighbors (KNN)

- **Performance Metrics:**
  - Accuracy: **82.26%**
  - Precision: **82.22%**
  - Recall: **82.26%**
  - F1 Score: **82.22%**

Confusion Matrix

- Add the confusion matrix visualization here:



Observation:

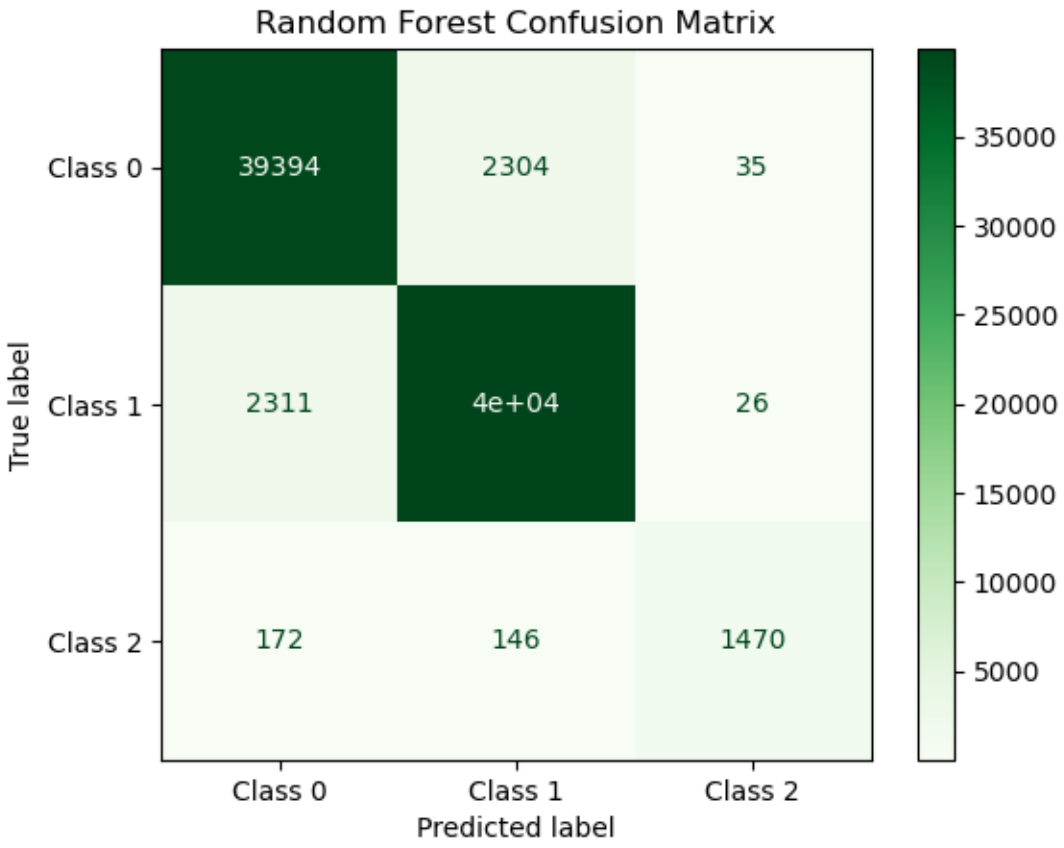
The major shortcoming that KNN observed was it was highly sensitive to the class imbalance problem.

2. Random Forest

- **Performance Metrics:**
  - Accuracy: **94.18%**
  - Precision: **94.18%**
  - Recall: **94.18%**
  - F1 Score: **94.17%**

Confusion Matrix

- Add the confusion matrix visualization here:



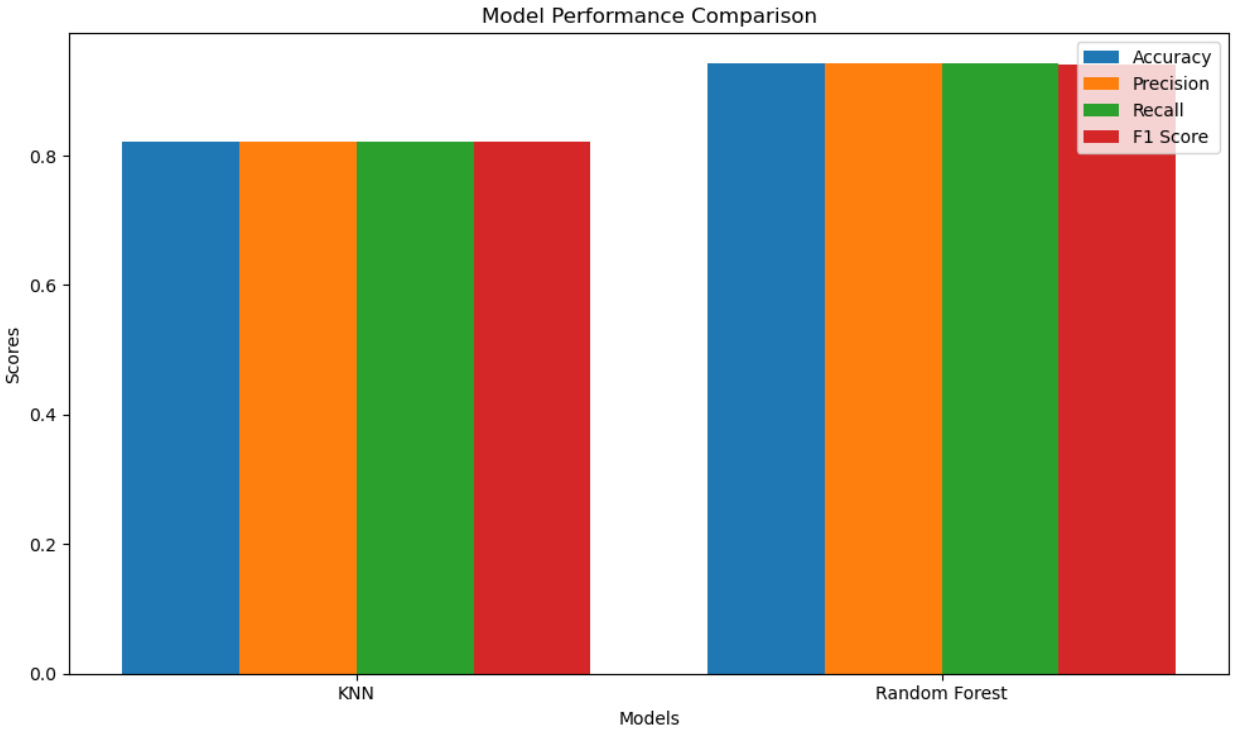
Observation:

Random Forest outperformed KNN more especially for the minority class because RF was able to learn from both the numerical columns and was able to handle with the imbalanced data and intricate relationships.

Model Comparison and Selection

Metric	**KNN**	**Random Forest**
Accuracy	82.26%	94.18%
Precision	82.22%	94.18%
Recall	82.26%	94.18%
F1 Score	82.22%	94.17%

- Add a bar chart or grouped bar chart to compare model performance metrics visually:



Conclusion and Recommendations

Key Insights:

1. Specifically, Random Forest yielded better accuracy with a ratio of 94.18%.
2. Despite being easy to understand andapply, the KNN algorithm was powerless to overcome the problem of class imbalance, especiallyin the context of the minority class.

Recommendations:

1. **For Classification:**
    - Random Forest to be used as a main model because of its stability and high level of performing as per expectation.
    - Tuning of hyperparameters such as max_depth, n_estimators etc in order to get best of the best resulthyperparameters (e.g., max_depth, n_estimators) to further enhance performance.
2. **Data Enhancements:**
    - Balance the classes for the dataset by methods such as SMOTE or ADASYN.
    - Develop fresh attributes, for example TotalPrice exists when Quantity multiplied on UnitPrice. Engineer new features, such as TotalPrice = Quantity * UnitPrice.

3. **Future Models:**
    - Try other models like XGBoost or LightGBM to get a better performance.

Reflection

During the hackathon:

- We enhanced our understanding of preprocessing techniques, including handling missing values, encoding, and scaling.
- Contributed to the tasks of applying and assessing machine learning models. Acquired experience in model comparison and in choosing the best classifier models.
- Gained insights into model comparison and selecting the best classifier.

Improvements for Future Projects:

- • Spend more time fine tuning the hyperparameters. use other models and look at ensembles differently.
- Emphasis with preprocessing steps on increasing the datasets to be processed for producing good results and ensemble techniques.
- Focus on automating preprocessing steps to handle larger datasets efficiently.

References:

Hastie, T., Tibshirani, R. and Friedman, J. (2009) [2]9). The Elements of Statistical Learning: Data Mining, Inference and Predictions Second Edition.

Breiman, L. (2001). Random Forests. Machine Learning vol. 45 no 1 2002 pp 5to32.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, pp.2825to2830.

Breiman, L. (2001). Random Forests. Machine Learning, 45(1), pp.5to32.

Friedman, J.H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics, 29(5), pp.1189to1232.

Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P. (2002). SMOTE: Minstrel synthesizing technique that involves oversampling of the minority class of data. Journal of Artificial Intelligence Research, 16, pp. 321-357.

Bishop, C.M. (2006). Pattern Recognition and Machine Learning. 1st India ed. New York: Springer.