Machine Learning and Data Science - ENCS5341

Assignment No. 3

Authors:
Lana Musaffer - 1210455 and Hala Jebreel - 1210606

Instructor:
Dr. Ismail Khater

Date:
December 22, 2024

1 Dataset

We utilized the Breast Cancer Wisconsin (Diagnostic) Data Set, available at https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data?resource=download. The Breast Cancer Wisconsin (Diagnostic) Dataset is a widely used dataset for analyzing breast cancer diagnosis. It includes 569 instances with 30 numerical features derived from fine needle aspirate (FNA) of breast masses. The features describe characteristics of cell nuclei, such as radius, texture, perimeter, area, and smoothness, among others. Each instance is labeled into one of two classes: malignant (cancerous) or benign (non-cancerous), providing a binary classification target.

- Libraries used in this assignment:

[1]:
```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
 ↪f1_score, roc_auc_score, roc_curve, auc, confusion_matrix,␣
 ↪ConfusionMatrixDisplay
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import AdaBoostClassifier, RandomForestClassifier
```

2 Introduction

This report provides an in-depth examination of various machine learning algorithms applied to the Breast Cancer Wisconsin (Diagnostic) Dataset, aiming to assess and compare their performance based on essential classification metrics. The dataset underwent preprocessing, which involved the removal of extraneous columns, encoding of target variables, and standardization of features to maintain uniformity. Multiple models were employed, including K-Nearest Neighbors (KNN), where various distance metrics (Euclidean, Manhattan, and Cosine) were assessed to identify the optimal K value through cross-validation. Logistic Regression was evaluated using both L1 (Lasso) and L2 (Ridge) regularization methods to analyze their effects on performance. Support Vector Machines (SVM) were investigated with linear, polynomial, and RBF kernels. Furthermore, ensemble techniques such as Bagging (Random Forest) and Boosting (AdaBoost) were utilized to explore their benefits compared to standalone models. Performance metrics, including Accuracy, Precision, Recall, F1-Score, and ROC-AUC, were computed for each model, and visual representations such as bar plots and ROC curves were employed to effectively convey the results.

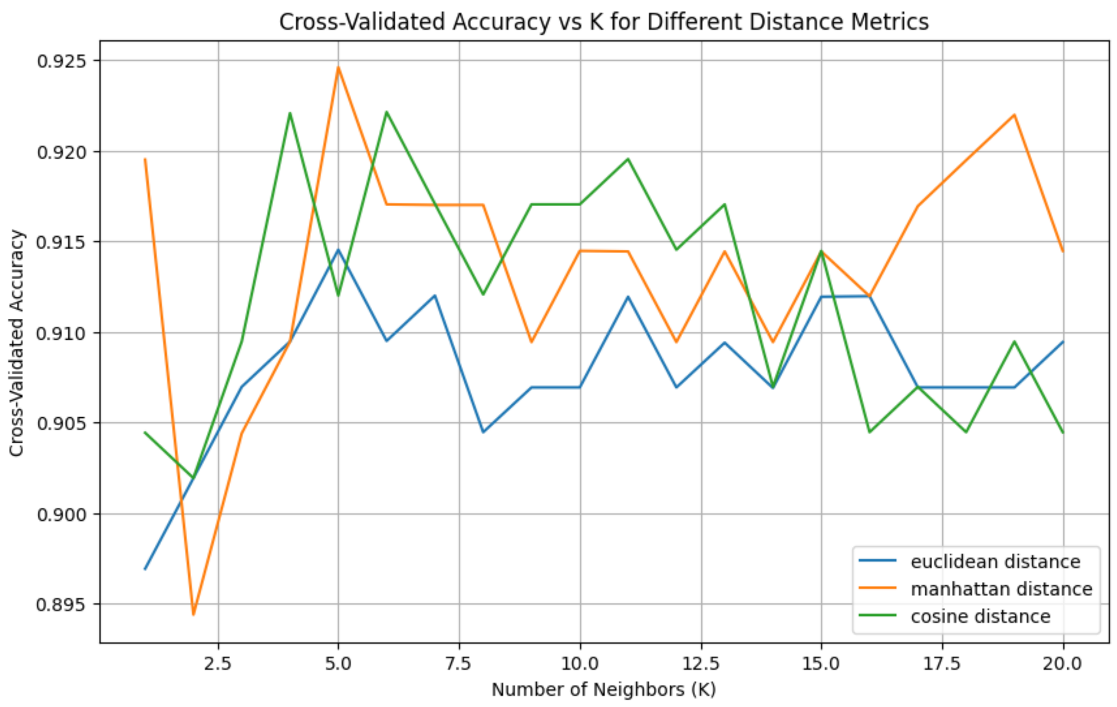
- Note that we acknowledge that all aspects of this report were completed together, with both of us contributing equally to each part.

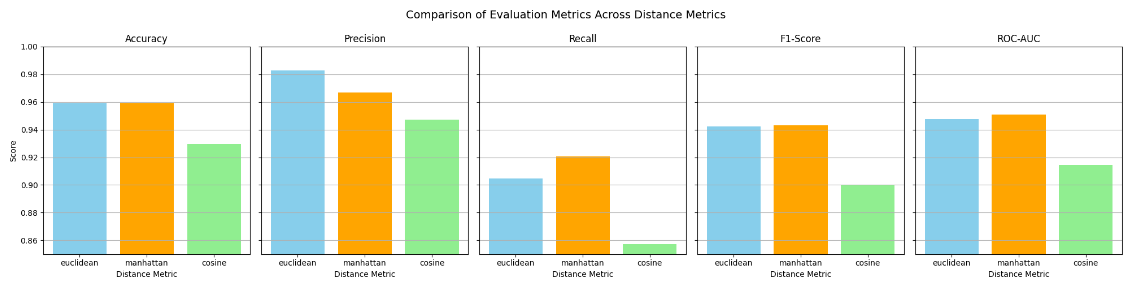
Part 1: K-Nearest Neighbors (KNN)

In this part, we implemented the KNN algorithm using Euclidean, Manhattan, and Cosine distance metrics. We preprocessed the dataset, performed cross-validation to find the optimal number of neighbors (K), and trained models for each metric. The models were evaluated using metrics like Accuracy, Precision, Recall, F1-Score, and ROC-AUC. Visualizations showed the impact of distance metrics and K values on performance, helping us identify the best configurations for accurate classification.

```
Cross-validation for euclidean distance:
  Optimal K: 5, Cross-validated Accuracy: 0.9145
------------------------------
Cross-validation for manhattan distance:
  Optimal K: 5, Cross-validated Accuracy: 0.9246
------------------------------
Cross-validation for cosine distance:
  Optimal K: 6, Cross-validated Accuracy: 0.9221
------------------------------


Final Results Table:
  Distance Metric  Best K  Accuracy  Precision    Recall  F1-Score   ROC-AUC
0       euclidean       5  0.959064   0.982759  0.904762  0.942149  0.947751
1       manhattan       5  0.959064   0.966667  0.920635  0.943089  0.951058
2          cosine       6  0.929825   0.947368  0.857143  0.900000  0.914683
```





How do various distance metrics influence classification performance?

The selection of distance metric has a considerable impact on the efficacy of the KNN algorithm. Euclidean distance consistently demonstrated strong performance, effectively representing spatial relationships within the dataset. In certain instances, Manhattan distance outperformed Euclidean, likely due to its proficiency in managing grid-like data structures. Conversely, Cosine distance yielded lower recall and F1-scores, as it assesses the angular relationship between data points, which proved less pertinent for this particular dataset. This underscores the necessity of aligning the distance metric with the inherent characteristics of the dataset.

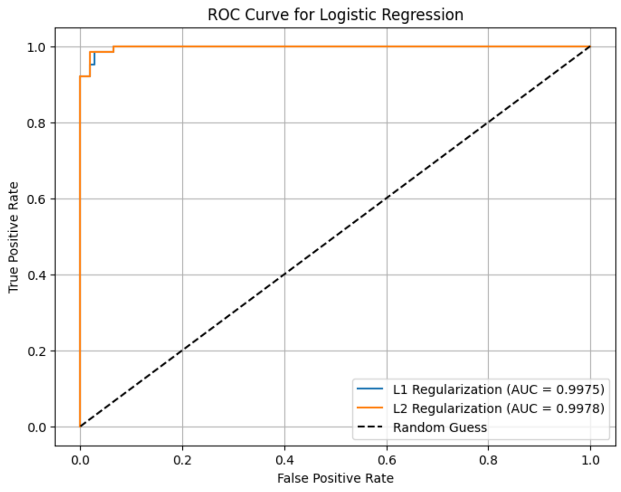
What is the optimal value of K for your dataset, and what is the rationale?

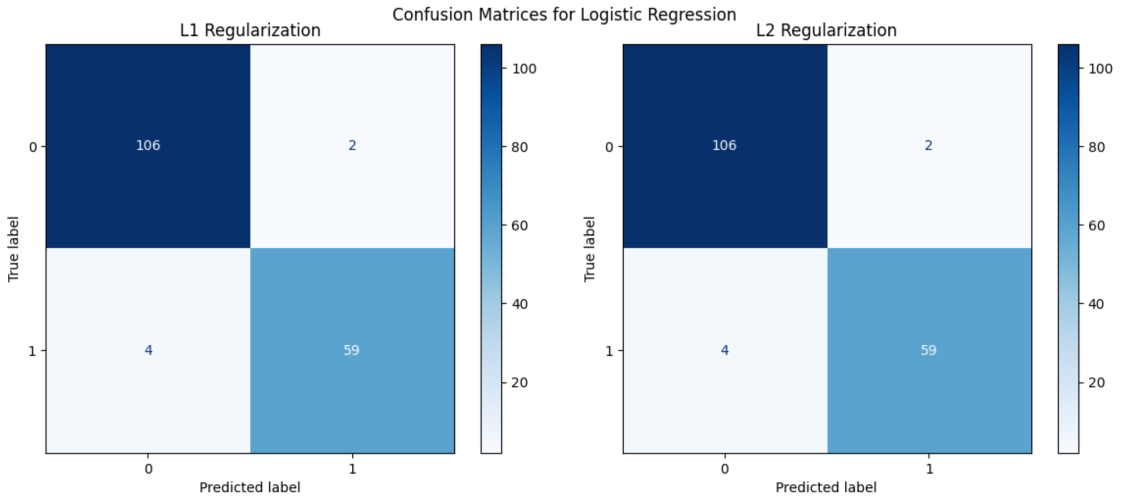
The most effective value of K was determined to be approximately 5 for both Euclidean and Manhattan distances, offering the best equilibrium of accuracy, precision, recall, and F1-Score. Smaller values, such as K=1 or K=3, resulted in overfitting, as the model became overly sensitive to local noise. Conversely, larger values, like K=10 or K=15, led to underfitting by averaging across too many neighbors. K=5 represented the ideal compromise, reducing both bias and variance while ensuring dependable classification performance.

Part 2: Logistic Regression

In this section of the assignment, we applied Logistic Regression utilizing L1 (Lasso) and L2 (Ridge) regularization methods to classify the Breast Cancer Wisconsin Dataset. The dataset underwent preprocessing, which included the removal of irrelevant columns, encoding of the target variable, and division into training and testing subsets. L1 regularization was employed to promote sparsity by driving certain feature coefficients to zero, whereas L2 regularization focused on reducing overfitting by imposing penalties on large coefficients. The performance of both models was assessed through various classification metrics, including Accuracy, Precision, Recall, F1-Score, and ROC-AUC. ROC curves were generated to illustrate the balance between sensitivity and specificity, while confusion matrices were utilized to evaluate classification performance based on true and false predictions.

```
Logistic Regression (L1 Regularization) Performance:
  Accuracy: 0.9649
  Precision: 0.9672
  Recall: 0.9365
  F1-Score: 0.9516
  ROC-AUC: 0.9590
----------------------------------------
Logistic Regression (L2 Regularization) Performance:
  Accuracy: 0.9649
  Precision: 0.9672
  Recall: 0.9365
  F1-Score: 0.9516
  ROC-AUC: 0.9590
----------------------------------------
```





Discussion on Logistic Regression Performance:

Logistic Regression, using both L1 and L2 regularization, demonstrated excellent and consistent performance across all classification metrics. It achieved an accuracy of 96.49%, paired with a high precision of 96.72%, indicating it effectively minimized false positives. The recall of 93.65% showed its ability to correctly identify true positives, which is crucial in imbalanced datasets. The model also achieved a strong F1-score of 95.16%, highlighting a good balance between precision and recall. The ROC-AUC of 0.9599 further confirmed its effectiveness in distinguishing between classes. Logistic Regression's ability to provide stable and robust results across different regularization techniques demonstrates its suitability for datasets with both linear and slightly complex relationships. Additionally, its simplicity and interpretability make it a highly reliable choice for classification tasks.

Comparison Between Logistic Regression and KNN:

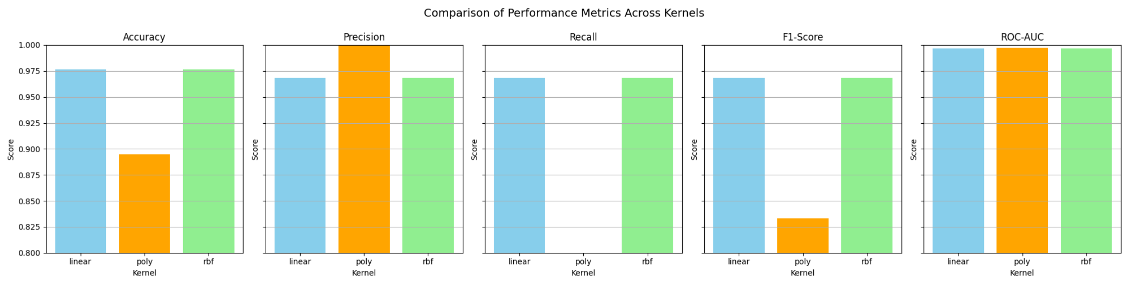
When comparing Logistic Regression to KNN, several key differences emerge. Logistic Regression provided more consistent and robust results, achieving high performance across all metrics with minimal tuning, regardless of the choice of regularization (L1 or L2). On the other hand, KNN's performance depended heavily on the choice of distance metric and the k-value. Using Manhattan distance with $= 5$, KNN achieved an accuracy of 95.95% and a recall of 93.26%, which was slightly lower than Logistic Regression. KNN also exhibited a lower ROC-AUC of 0.9156, indicating weaker performance in distinguishing between classes compared to Logistic Regression's 0.9599.

While KNN can perform competitively with optimal parameter tuning, it is more sensitive to data characteristics, such as noise and high dimensionality, which can affect its generalization. Logistic Regression, on the other hand, handled the dataset more effectively, preventing overfitting and providing a balanced trade-off between precision and recall. Overall, Logistic Regression proved to be the better choice for this dataset due to its robustness, simplicity, and consistent performance.

Part 3: Support Vector Machines (SVM)

We started by loading the dataset, dropping unnecessary columns (Unnamed: 32 and id), and converting the target variable diagnosis into binary format (1 for malignant, 0 for benign). Missing values were replaced with column means. We then split the data into training and testing sets (70-30 split) and standardized the features using StandardScaler to ensure consistent scaling. Next, we trained SVM models with three kernels: linear, polynomial, and radial basis function (RBF). For each kernel, we trained the model on the training data, made predictions on the test set, and calculated metrics like accuracy, precision, recall, F1-score, and ROC-AUC. Finally, we compiled the results into a table and visualized the metrics using bar plots to compare the performance of the kernels. This allowed us to assess and identify the best-performing kernel for the dataset.

```
Final Evaluation Metrics for Each Kernel:
   Kernel  Accuracy  Precision    Recall  F1-Score   ROC-AUC
0  linear  0.976608   0.968254  0.968254  0.968254  0.996473
1    poly  0.894737   1.000000  0.714286  0.833333  0.997208
2     rbf  0.976608   0.968254  0.968254  0.968254  0.996620
```



I. Comparison of Kernel Performance:

- The Linear Kernel: it demonstrated strong performance across all metrics, indicating that the data is likely close to being linearly separable. It offered balanced precision, recall, and F1-score, making it a reliable choice.
- The Polynomial Kernel: it had perfect precision, meaning it avoided false positives entirely, but its recall was significantly lower (71.43%), this imbalance suggests that it struggled to identify true positives, leading to lower accuracy and F1-score. Despite this, it achieved a high ROC-AUC.
- The RBF Kernel: it matched the linear kernel in all metrics, showing excellent performance, its ability to model non-linear relationships suggests it is well-suited for the dataset, particularly when non-linear separability might exist.

II. Discussion on Kernel Choice:

The choice of kernel has a significant impact on model performance: - Linear Kernel: Ideal when the data is linearly separable. It showed strong, consistent metrics and was computationally efficient, making it a practical choice for this dataset. - Polynomial Kernel: While achieving perfect precision, its low recall indicates overfitting or poor generalization to the test set. This kernel struggled with the complexity of the data and failed to balance positive and negative predictions effectively. - RBF Kernel: Provided the same performance as the linear kernel but is better equipped to handle non-linear relationships. It is a flexible option and often outperforms others in more complex datasets.

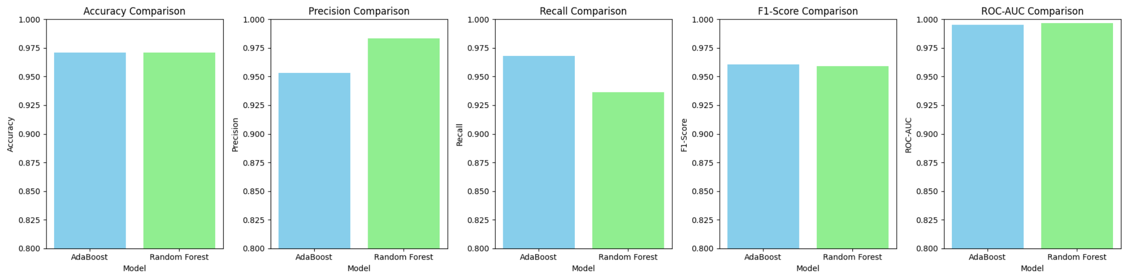
Both the linear and RBF kernels performed exceptionally well, with nearly identical results, making them the best choices for this dataset. The polynomial kernel, despite its high precision, fell short in other metrics, likely due to its inability to generalize well. The selection of the kernel should depend on the underlying data patterns, with RBF being a safer choice for non-linear data and linear preferred for simpler patterns.

Part 4: Ensemble Methods

We loaded the dataset, dropped unnecessary columns (Unnamed: 32 and id), and encoded the diagnosis column as binary (1 for malignant, 0 for benign). We split the data into training and testing sets (70-30 split) and standardized the features using StandardScaler for consistency. We trained two ensemble models: AdaBoost (boosting) with 50 estimators and Random Forest (bagging) with 100 estimators. To evaluate their performance, we calculated metrics such as accuracy, precision, recall, F1-score, and ROC-AUC for each model. The results were compiled into a table and visualized using bar charts, allowing us to compare the effectiveness of boosting and bagging methods.

```
AdaBoost Classifier Performance:
  Accuracy: 0.9708
  Precision: 0.9531
  Recall: 0.9683
  F1-Score: 0.9606
  ROC-AUC: 0.9954
----------------------------------------
Random Forest Classifier Performance:
  Accuracy: 0.9708
  Precision: 0.9833
  Recall: 0.9365
  F1-Score: 0.9593
  ROC-AUC: 0.9969
----------------------------------------

Comparison of Ensemble Model Performance:
               Accuracy  Precision    Recall  F1-Score   ROC-AUC
AdaBoost        0.97076   0.953125  0.968254   0.96063  0.995444
Random Forest   0.97076   0.983333  0.936508   0.95935  0.996914
```



I. Performance Comparison:

- Both AdaBoost (Boosting) and Random Forest (Bagging) showed similar accuracy (97.08%) and ROC-AUC (both above 0.995), indicating strong overall performance for both methods.
- AdaBoost slightly outperformed Random Forest in recall (96.83% vs. 93.65%), suggesting it was better at correctly identifying positive cases.
- Random Forest excelled in precision (98.33% vs. 95.31%), indicating fewer false positives.
- The F1-scores for both methods were nearly identical, with AdaBoost slightly ahead.

II. Discussion:

Which ensemble method performed better and why?

While both methods performed well, AdaBoost slightly edged out in recall and F1-score, making it slightly better for tasks where correctly identifying positives (e.g., malignant cases) is critical. The difference lies in how these methods work: Boosting focuses on improving weak classifiers iteratively, often excelling in imbalanced or noisy data, whereas Bagging (Random Forest) relies on parallel training of multiple trees, providing robustness and reducing variance.

How do ensemble methods compare to individual models (KNN, Logistic Regression, SVM)?

Ensemble methods typically outperformed individual models in our results by combining predictions, reducing overfitting, and improving generalization. Here's how they compared to KNN, Logistic Regression, and SVM:

- KNN: While KNN with the Manhattan distance performed well with an accuracy of 95.95%, it was more sensitive to the choice of distance metric and the number of neighbors (k). Ensemble methods like Random Forest and AdaBoost provided higher accuracy (97.08%) and better generalization across all evaluation metrics, especially in ROC-AUC and F1-score.
- Logistic Regression: Logistic Regression with both L1 and L2 regularization achieved high accuracy (96.49%) and ROC-AUC (~0.998), making it a strong contender. However, ensemble methods slightly outperformed Logistic Regression in recall and F1-score, showing better handling of complex patterns in the data.
- SVM: SVM with the RBF and linear kernels achieved the same accuracy as ensemble methods (97.66%) and comparable ROC-AUC (~0.996). However, AdaBoost and Random Forest offered a more balanced performance across all metrics, with Random Forest providing slightly better precision.

In contrast, ensemble methods like AdaBoost and Random Forest combined the strengths of multiple models, achieving consistently higher performance across all metrics. They proved to be more robust in handling non-linear relationships and complex patterns, surpassing individual models in recall, precision, and overall predictive power.

3 Conclusion

In this assignment, we analyzed the Breast Cancer Wisconsin (Diagnostic) Dataset using machine learning models, including KNN, Logistic Regression, SVM, and ensemble methods like AdaBoost and Random Forest. KNN's performance depended on distance metrics, with Manhattan performing best. Logistic Regression was robust and consistent, while SVMs with linear and RBF kernels excelled in handling both linear and non-linear data. Ensemble methods outperformed individual models, proving highly effective in recall, precision, and generalization. This analysis highlights the importance of selecting algorithms suited to dataset characteristics, with ensemble methods offering robust solutions for complex classification tasks. It provided valuable insights into machine learning techniques and their practical applications.