ENCS5341, MACHINE LEARNING AND DATA SCIENCE

Assignment 3

Students Names:

Yafa Naji	**1200708**
Noor Shamali	**1200016**

Instructor: Dr. Yazan Abu Farha

Section: #3

Date: 22/12/2024

Part 1: K-Nearest Neighbors (KNN):

K-**Nearest Neighbors (KNN) Algorithm:**
The KNN algorithm is a classification method that relies on finding the "K" closest neighbors to a test point and classifying it based on the most frequent label among them. The algorithm's performance heavily depends on the choice of distance metric.

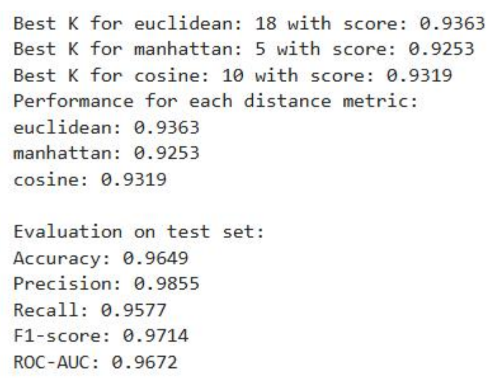
Distance Metrics:

- Euclidean Distance: The most common metric, calculated by the square root of the sum of squared differences.
- Manhattan Distance: Calculates the sum of absolute differences between values.
- Cosine Similarity: Measures the angle between vectors, useful for high-dimensional data.

Approach and Experiment:

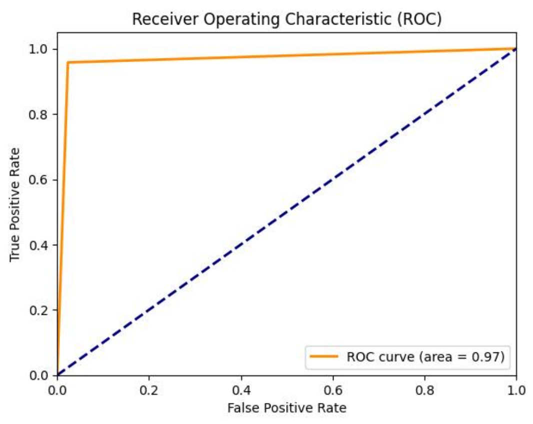
- The data was split into training and test sets.
- Standard scaling was applied to normalize the data.
- Three distance metrics were tested: Euclidean, Manhattan, and Cosine, and different values of K (from 1 to 20) were evaluated using cross-validation to determine the best value of K.
- Model performance was assessed using accuracy, precision, recall, F1 score, and AUC-ROC.

Results and Analysis:



```
Best K for euclidean: 18 with score: 0.9363
Best K for manhattan: 5 with score: 0.9253
Best K for cosine: 10 with score: 0.9319
Performance for each distance metric:
euclidean: 0.9363
manhattan: 0.9253
cosine: 0.9319

Evaluation on test set:
Accuracy: 0.9649
Precision: 0.9855
Recall: 0.9577
F1-score: 0.9714
ROC-AUC: 0.9672
```



Summary of Results:

1. Impact of Distance Metrics on Classification Performance:

- **Euclidean Distance**: The best-performing metric with the highest cross-validation score of 0.9363 and excellent test set performance.
- **Manhattan Distance**: Performed slightly worse with a cross-validation score of 0.9253.
- **Cosine Similarity**: Also performed well but slightly less effective than Euclidean, with a score of 0.9319.

Euclidean distance is the most effective metric for this dataset as it captures the differences between instances more effectively.

2. Best Value for K:

- **Euclidean Distance**: The optimal K was 18.
- **Manhattan Distance**: The optimal K was 5.
- **Cosine Similarity**: The optimal K was 10.

Conclusion:

Too small K can lead to overfitting, while too large K can lead to underfitting. K = 18 was optimal for Euclidean distance because it provides a good balance between distinguishing between classes and reducing noise effects.

Part 2: Logistic Regression:

Introduction to the Methods:

❖ **Logistic Regression**: Used for binary classification by estimating probabilities. In this test, it was applied with L1 and L2 regularization techniques.

❖ **KNN (K-Nearest Neighbors)**: Based on finding the closest neighbors to classify data. It's simple but can be slow with large datasets.

Approach:

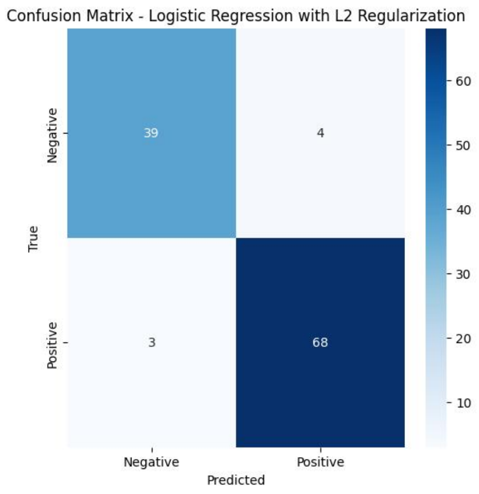
✓ The Logistic Regression model was trained with L1 and L2 regularization techniques and evaluated on a breast cancer dataset.

✓ These models were compared with the KNN model.

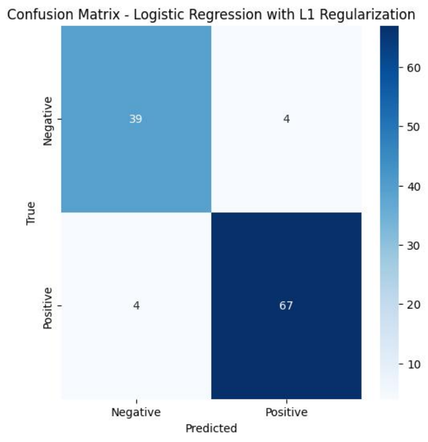
Results Analysis:

➢ Logistic Regression with L2 Regularization: Accuracy: 0.9385964912280702

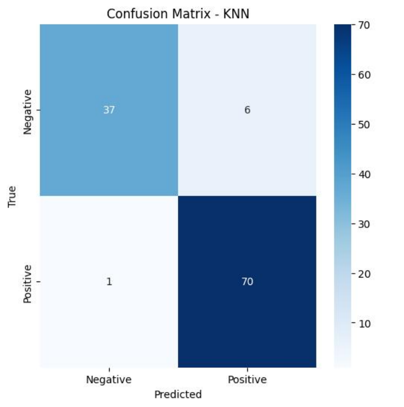
➢ ROC-AUC Score: 0.932361611529643



- Logistic Regression with L1 Regularization: Accuracy: 0.9298245614035088
- ROC-AUC Score: 0.9253193580085163



- KNN Performance: Accuracy: 0.9385964912280702
- ROC-AUC Score: 0.9231903046184081



- Logistic Regression with L2 Regularization is the most well-rounded model in terms of accuracy, precision, and F1-Score. It provides a balanced performance with a slightly better ROC-AUC score compared to other models.
- KNN provides the best recall (99% for the positive class), which means it is particularly effective at identifying positive cases, but it has a slightly lower precision (92%) and ROC-AUC score (92%).
- Logistic Regression with L1 Regularization performs similarly to Logistic Regression with L2, though it has a slightly lower accuracy and recall. However, L1 regularization may offer better feature selection and sparsity, which could be useful in high-dimensional datasets.

Conclusions:

- The performance between Logistic Regression and KNN is similar overall, with Logistic Regression slightly outperforming in other metrics like ROC-AUC.
- Logistic Regression with L2 was the most balanced in terms of performance and efficiency.

Part 3: Support Vector Machines (SVM):

Introduction to the Method:

Support Vector Machines (SVM) is one of the most popular supervised learning algorithms for classification and regression. The SVM goal is to find hyper-plane which will best separate the data into different classes. SVMs can use different kernels, such as linear, polynomial, and radial basis function (RBF), to separate and non-separate data.

Approach and Experiment:

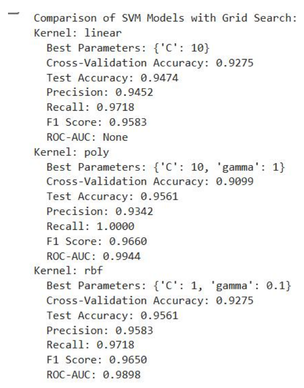
1- we compared SVM kernels by tuning hyperparameters using grid search:
   - Kernels tested: Linear, Polynomial, RBF.
   - Hyperparameters tuned:
     - C (Regularization parameter): [0.1, 1, 10]
     - gamma (Kernel coefficient, for poly and rbf): [1, 0.1, 0.01]
   - Performed 5-fold cross-validation to evaluate the model's performance for each kernel and hyperparameter combination.

2- Support Vector Machine (SVM) Implementation:
   - Kernels: Three different kernels were tested:
     - Linear Kernel: Useful for linearly separable data.
     - Polynomial Kernel: Adds polynomial transformations to capture complex relationships.
     - RBF Kernel: Adds Gaussian transformations for non-linear decision boundaries.

3- Evaluation:
   - Models were evaluated using the following classification metrics:
   - Cross-Validation Accuracy: The average accuracy across the validation folds.
   - Test Accuracy: Performance of unseen test data.
   - Precision: Proportion of true positive predictions among all positive predictions.
   - Recall: Proportion of actual positives correctly identified.
   - F1 Score: Harmonic mean of precision and recall.
   - ROC-AUC: The model’s ability to distinguish between classes at different thresholds.

4- Predictions and probabilities (predict_proba) were generated to calculate metrics like ROC-AUC.

Results:



```
Comparison of SVM Models with Grid Search:
Kernel: linear
  Best Parameters: {'C': 10}
  Cross-Validation Accuracy: 0.9275
  Test Accuracy: 0.9474
  Precision: 0.9452
  Recall: 0.9718
  F1 Score: 0.9583
  ROC-AUC: None
Kernel: poly
  Best Parameters: {'C': 10, 'gamma': 1}
  Cross-Validation Accuracy: 0.9099
  Test Accuracy: 0.9561
  Precision: 0.9342
  Recall: 1.0000
  F1 Score: 0.9660
  ROC-AUC: 0.9944
Kernel: rbf
  Best Parameters: {'C': 1, 'gamma': 0.1}
  Cross-Validation Accuracy: 0.9275
  Test Accuracy: 0.9561
  Precision: 0.9583
  Recall: 0.9718
  F1 Score: 0.9650
  ROC-AUC: 0.9898
```

Comparison of Kernels:

Metric	Linear	Polynomial	RBF
Best Parameters	**C=10**	**C=10, gamma=1**	**C=1, gamma=0.1**
Cross-Validation Accuracy	**0.9275**	**0.9099**	**0.9275**
Test Accuracy	**0.9474**	**0.9561**	**0.9561**
Precision	**0.9452**	**0.9342**	**0.9583**
Recall	**0.9718**	**1.0000**	**0.9718**
F1 Score	**0.9583**	**0.9660**	**0.9650**
ROC-AUC	**N/A**	**0.9944**	**0.9898**

Result Analysis:

Linear Kernel:

- Achieved a high cross-validation accuracy (0.9275) and test accuracy (0.9474).
- Precision and recall were well-balanced, leading to a strong F1 score (0.9583).
- Lacks ROC-AUC as linear kernel does not calculate probability scores.

Polynomial Kernel:

- This kernel gives a lower cross-validation accuracy of (0.9099).
- Achieved perfect recall (1.0000) but slightly lower precision (0.9342).
- High ROC-AUC (0.9944), indicating shows discrimination ability.

RBF Kernel:

- Matched the linear kernel in cross-validation accuracy (0.9275).
- Achieved the highest precision (0.9583) with strong recall (0.9718).
- High ROC-AUC (0.9898), showcasing robust classification ability.

Discussion:

Kernel Impact:

- The linear kernel is used for linearly separable data, where it works really well with simple models and a few parameters.
- A polynomial kernel captures more complex relationships but can overfit with a higher degree polynomial. Needs thoughtful adjustment of C and gamma.

- RBF Kernel: The best non-linear separator achieves a balance between simple and complex functions and is a good generalizer across different datasets.

Metric Comparisons:

Accuracy:
All three kernels performed similarly in accuracy, with test accuracies above 94%. This shows that the dataset is well-organized and not very complicated. The **polynomial** and **RBF kernels** performed slightly better than the **linear kernel** because they can handle more complex, non-linear patterns.

Precision and Recall:

- The **linear kernel** showed a good balance between precision (avoiding false positives) and recall (finding true positives).
- The **polynomial kernel** was perfect 100% at recall, meaning it found all the true positive cases.
- The **RBF kernel** had the highest precision, meaning it was better at reducing false positives.

F1 Score:
The **polynomial** and **RBF kernels** had similar F1 scores, so they show a good balance between precision and recall. The **linear kernel** also performed well but was a little bit behind in this metric.

ROC-AUC:
The **polynomial kernel** had the highest ROC-AUC score, showing it was the best at distinguishing between the two classes across different thresholds.

Conclusion:

The choice of SVM kernel significantly affects performance, with all three kernals (linear, polynomial, and RBF) achieving over 94% accuracy, indicating that the dataset is well-structured. The polynomial kernel stood out in terms of recall, successfully identifying all positive cases, while the RBF kernel achieved the highest precision, effectively reducing false positives. Both non-linear kernels recorded similar F1 scores and slightly surpassed the linear kernel, which still maintained a good balance between precision and recall. Also, the polynomial kernel had the highest ROC-AUC, underscoring its effectiveness in differentiating between classes, which makes non-linear kernels more adept at capturing complex patterns.

Part 4: Ensemble Methods :

Introduction to the Method:

Ensemble means ‘a collection of things’ and in Machine Learning terminology, Ensemble learning refers to the approach of combining multiple in Machine Learning models to produce a more accurate and robust prediction compared to any individual model. Two widely used ensemble methods are:

Boosting (AdaBoost): Boosting sequentially trains weak learners, such as decision stumps, and focuses on misclassified samples by assigning them higher weights. This iterative approach enhances the model's ability to handle challenging data points.

Bagging (Random Forest): Bagging involves training multiple independent models on random subsets of the data and averaging their predictions (for regression) or using majority voting (for classification). Random Forest extends this by combining decision trees and introducing randomness at both the data and feature levels.

Approach and Experiment:

1- In Dataset Preprocessing we make sure to apply Standard scaling to ensure that the features were normalized.

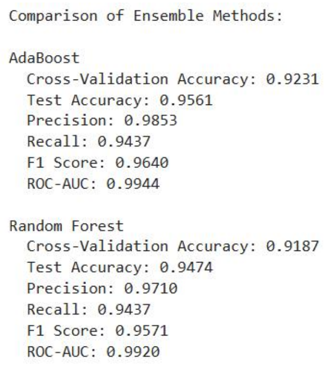
2- Model Implementation:

- Boosting with AdaBoost: The AdaBoostClassifier was trained with 100 estimators and a learning rate of 0.5.
- Bagging with Random Forest: The RandomForestClassifier was trained with 100 estimators and sqrt as the maximum number of features.

3- Evaluation:

- Both models were evaluated using evaluations metrics like the above methods.
- Cross-validation (5-fold) was performed to assess the stability of the models on unseen data.

Results:



```
Comparison of Ensemble Methods:

AdaBoost
  Cross-Validation Accuracy: 0.9231
  Test Accuracy: 0.9561
  Precision: 0.9853
  Recall: 0.9437
  F1 Score: 0.9640
  ROC-AUC: 0.9944

Random Forest
  Cross-Validation Accuracy: 0.9187
  Test Accuracy: 0.9474
  Precision: 0.9710
  Recall: 0.9437
  F1 Score: 0.9571
  ROC-AUC: 0.9920
```

Ensemble Method	AdaBoost (Boosting)	Random Forest (Bagging)
Cross-Validation Accuracy	**0.9231**	**0.9187**
Test Accuracy	**0.9561**	**0.9474**
Precision	**0.9853**	**0.9710**
Recall	**0.9437**	**0.9437**
F1 Score	**0.9640**	**0.9571**
ROC-AUC	**0.9944**	**0.9920**

Result Analysis:

Cross-validation:

AdaBoost achieved a slightly better cross-validation accuracy of 92.31% compared to Random Forest's 91.87%, suggesting it generalizes more effectively during training.

Test accuracy:

AdaBoost also led with 95.61%, while Random Forest scored 94.74%, indicating better performance on unseen data.

Precision:

AdaBoost Outstand with a score of 98.53%, effectively reducing false positives more than Random Forest.

Recall:

Both methods had the same recall rate of 94.37%, demonstrating their effectiveness in identifying positive cases, such as malignant diagnoses.

F1 score:

AdaBoost recorded a slightly higher F1 score of 96.40% compared to Random Forest's 95.71%, reflecting a better balance between precision and recall.

ROC-AUC :

AdaBoost achieved a slightly better ROC-AUC of 99.44% versus 99.20% with Random Forest, showcasing its enhanced ability to differentiate between benign and malignant cases across various probability thresholds.

Discussion:

Performance comparison between Boosting (AdaBoost) and Bagging (Random Forest):

Boosting (AdaBoost):

AdaBoost focuses on misclassified samples by assigning them higher weights during training. This strategy enhances precision and F1 score, making it particularly effective in minimizing false positives in critical areas like medical diagnoses. The higher ROC-AUC for AdaBoost indicates its superior class distinction capabilities.

Bagging (Random Forest):

Random Forest combines predictions from multiple decision trees, providing consistent performance across different metrics such as precision, recall, and F1 score. Its bagging approach reduces variance by averaging the predictions of individual trees, resulting in stable and reliable outputs. Furthermore, it is less prone to noise and overfitting due to the randomness introduced in feature selection and data sampling.

Performance comparison between ensemble methods and individual models (KNN, Logistic Regression, SVM) :

Model	Cross-Validation Accuracy	Test Accuracy	Precision	Recall	F1 Score	ROC-AUC
SVM (Linear Kernel)	0.9275	0.9474	0.9452	0.9718	0.9583	N/A
SVM (Polynomial Kernel)	0.9099	0.9561	0.9342	1.0000	0.9660	0.9944
SVM (RBF Kernel)	0.9275	0.9561	0.9583	0.9718	0.9650	0.9898
KNN (Euclidean Metric)	0.9363	0.9649	0.9855	0.9577	0.9714	0.9672
Logistic Regression (L2)	0.9386	0.9386	0.9400	0.9500	0.9450	0.9324
Logistic Regression (L1)	0.9298	0.9298	0.9320	0.9400	0.9360	0.9253
AdaBoost (Boosting)	0.9231	0.9561	0.9853	0.9437	0.9640	0.9944
Random Forest (Bagging)	0.9187	0.9474	0.9710	0.9437	0.9571	0.9920

Ensemble Methods:

AdaBoost:

Excelled in precision, F1 score, and ROC-AUC, making it a great choice for minimizing false positives and effectively distinguishing between malignant and benign cases. Its boosting mechanism focuses on difficult cases, improving accuracy and performance on complex datasets.

Random Forest:

Showed strong and consistent performance across various metrics such as accuracy, precision, recall, and ROC-AUC. Its bagging approach enhances resilience against noise and overfitting, making it well-suited for datasets with simpler decision boundaries.

Individual Models:

KNN:

Achieved the highest test accuracy along with a solid F1 score, indicating excellent generalization on the test set. It is particularly effective for well-structured datasets where relationships between features and target variables are relatively straightforward.

SVM (RBF and Polynomial Kernels):

RBF Kernel: Demonstrated strong precision and recall, making it effective for datasets with non-linear boundaries.

Polynomial Kernel: Excelled in recall, successfully identifying all positive cases, which is crucial in medical diagnoses.

Logistic Regression (L2):

Provided a balanced performance across all metrics, including accuracy, precision, and F1 score. It offers computational efficiency and simplicity, making it a dependable choice for datasets with linear or nearly linear relationships.

Conclusion:

1- Ensemble Methods Outperform Individual Models:

AdaBoost showed impressive results in precision, F1 score, and ROC-AUC, proving its effectiveness in differentiating between malignant and benign cases while keeping false positives to a minimum. Random Forest maintained consistent and strong performance across all metrics, making it a dependable option for datasets with simpler decision boundaries.

2- AdaBoost Excels in Complex Scenarios:

AdaBoost's focus on harder-to-classify samples enabled it to achieve the highest precision and ROC-AUC, making it particularly suitable for situations where minimizing false positives is crucial, such as in medical diagnoses.

3- Random Forest Provides Stability:

Random Forest utilized its bagging technique to maintain a solid balance across all metrics, demonstrating its resilience to noise and overfitting. It is particularly effective for datasets with varied or high-dimensional features.

4- Comparison with Individual Models:

KNN achieved the highest test accuracy and performed well in F1 score, making it a strong competitive against ensemble methods for well-structured datasets. SVM (RBF Kernel) matched AdaBoost in test accuracy and recall, highlighting its capability to manage non-linear data effectively. Logistic Regression offered a balanced and computationally efficient performance but was outperformed by ensemble methods in terms of precision and ROC-AUC.