COMP534- Applied Artificial Intelligence

Assignment 1 – Binary Classification

Submitted by,

Jishnu Prakash Kunnanath Poduvattil	201581347	j.kunnanath-poduvattil@liverpool.ac.uk
Akhil Raj	201594703	a.raj@liverpool.ac.uk

Introduction

This report consists of discussion and analysis of binary classification using different machine learning algorithms using Pima Indian diabetes dataset.

Libraries Used

To perform binary classification, we mainly used the following libraries: -

Library	Utilisation	Link
Scikit-learn it consists of simple and advanced machine learning techniques for predictive data analysis.	Load machine learning models, data pre-processing, model performance analysis etc	https://scikit-learn.org/stable/
Pandas To perform data frame operations in python	Load data, data frame operations like value count etc	https://pandas.pydata.org/
Numpy To perform numerical, array operations	Using mean operator	https://numpy.org/
Matplotlib To visualise data in python	Plotting diagrams, charts for exploratory data analysis	https://matplotlib.org/
Seaborn Advanced data visualisation library in python	Plotting pair plots, heatmap etc for exploratory data analysis	https://seaborn.pydata.org/
Pydot Python interface to GraphViz	Used for visualising decision trees	https://pypi.org/project/pydot/

Classification Methods

For predicting diabetes state using Pima Indian diabetes dataset, we are focussing on three following techniques.

K-Nearest Neighbour (KNN) Classification

KNN is a simple technique that can be used for classification. The dataset is memorised and the prediction for an unknown data point is done based on the closest neighbours. The value of K and the metric used to measure distance between datapoints impacts the performance of KNN. We used KNN Classifier algorithm from scikit-learn to fit the diabetes data.

Hyperparameter tuning: -

1) **Elbow method**: We used elbow method to find a perfect value of K. The metric used to measure the distance is *minkowski* with p value of 2, that is equivalent to Euclidean distance. An iteration of KNN with different values of K was performed.

2) **K-Fold Cross Validation:** We also tried K-fold cross validation method to find the perfect K-value. An iteration for different K values with 4-fold cross validation. Basically, we split the train data in to 4 folds and use 3 folds for training and 1-fold for validating as iterations. This process is repeated for different k values.

Decision Tree Classification

Decision tree is a simple machine learning technique that decides by performing sequence of tests on each attribute, building a tree of possible decisions. The parameters that influence a decision tree is the criteria Gini impurity or entropy and information gain (maximum gain leads to low entropy). We used sci-kit learn decision tree algorithm to fit our diabetes data.

Hyperparameter tuning: -

GridSearchCV: It is a scikit module used for hyperparameter tuning. We passed a set of parameter values to the GridSearchCV and it did an exhaustive search on the grid using different combinations and gives out some parameter values where the model performs the best.

Logistic Regression

Logistic regression is a process of modelling the probability of a discrete outcome given an input variable. The most common logistic regression models a binary outcome; something that can take two values such as true/false, yes/no, and so on. Used sci-kit learn Logistic regression algorithm to fit the dataset. We used **GridSearchCV** for Hyperparameter tuning.

Training and Testing Process

Data Pre-processing

After performing an exploratory data analysis on Pima Indian diabetes dataset, we replaced zero values in 5 highly correlated feature vectors like Glucose, BloodPressure, SkinThickness, Insulin, BMI with respective median values. No duplicates were found.

Train-test split

Train-test split ratio was defined as 80:20. In KNN Classifier, a 4-fold cross validation method was tried out to find the value of K. Similarly, a 4-fold cross validation was used in hyperparameter tuning for decision tree classifier.

Evaluation

Comparison of confusion matrix: -

Method	True Positives	True Negatives	False Positive	False Negative
KNN	99	28	9	19
Decision Tree	96	32	11	15
Logistic regression	98	28	9	19

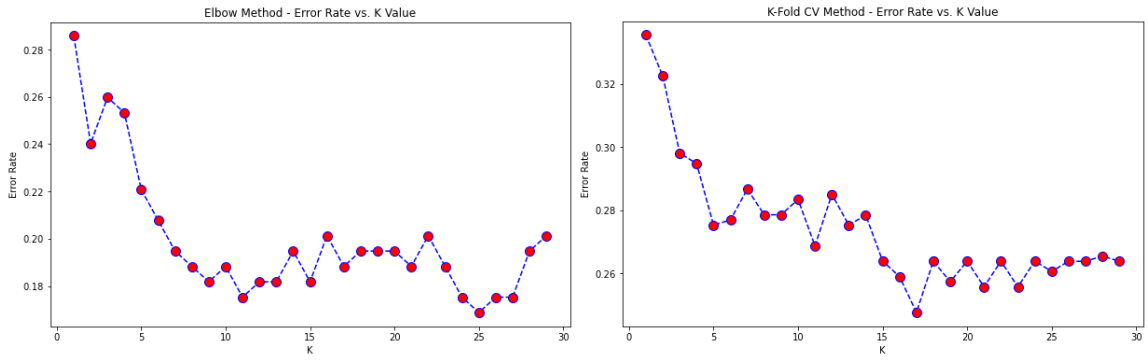
Comparison of Performances: -

Method	Precision		Recall		F1-score		Accuracy	
	0	1	0	1	0	1	Train	Test
KNN	0.84	0.76	0.92	0.60	0.88	0.67	79 %	82 %
Decision Tree	0.86	0.74	0.90	0.68	0.88	0.71	82 %	83 %
Logistic regression	0.84	0.76	0.92	0.60	0.88	0.67	77%	82%

Conclusions: -

KNN Classification

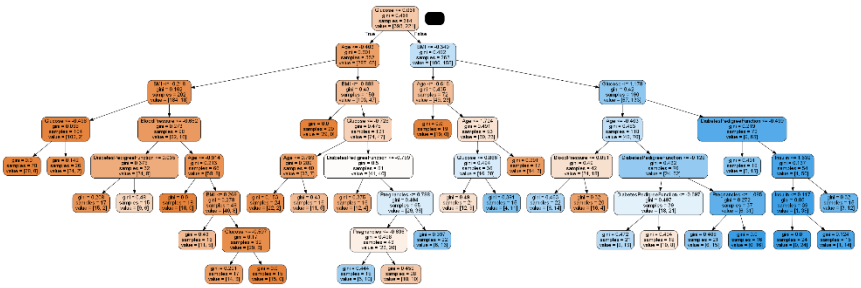
To find the optimal value of K, Elbow method and K-fold cross validation (4 folds, mean accuracy for each k) methods were used. Both methods suggested different values of K and we decided to choose K value using elbow method as it yielded good results. The charts show the error rate yielded in both methods. A reduced error rate was found for K=25 in elbow method and K=17 in k-fold cross validation. But at these values, the model is overfitting. The final value of K was selected as 9.



Decision Tree

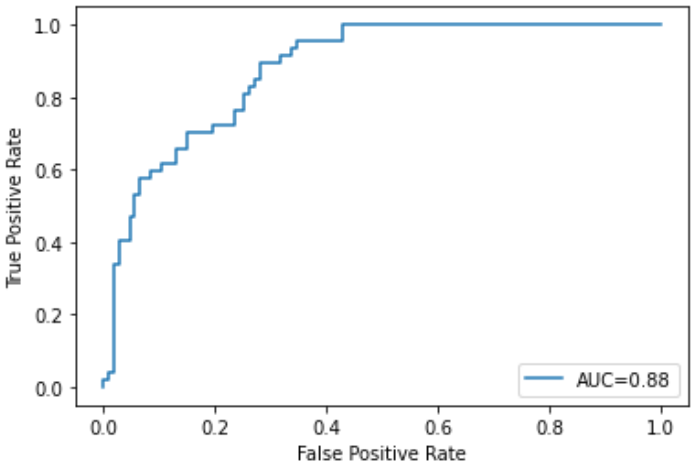
GridsearchCV was used for hyperparameter tuning and the below table shows the parameters searched and the results from the grid search algorithm that yielded the best results and the final decision tree.

Parameter	Values	Best performance
criterion	['Gini', 'entropy']	Gini
max_depth	Range (2, 100)	16
min_samples_leaf	Range (1, 20)	15
min_samples_split	Range (1, 20)	12
splitter	['best', 'random']	best



Logistic Regression

When GridsearchCV is used for hyperparameter tuning and the default parameters give the best accuracy. Without any parameter turning, the model gives 81% accuracy. So, the base model is used as the final model.



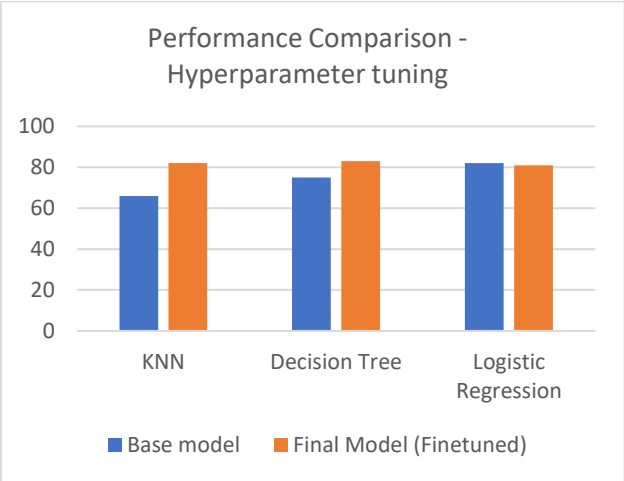
The graph shows roc curve of the logistic regression model. The more the curve hugs the top left corner of the plot, the better the model. To quantify this, area under the curve(AUC) is calculated. The closer AUC is to 1, the better model. This model has an AUC of 0.88.

Summary

In comparison, decision tree yields better results. As the data is unbalanced, KNN tends to overfit and **decision tree** with the above parameters yielded the best results when compared with KNN and logistic regression. The base model of logistic regression gave a good test result, but it tends to be overfitting when compared with train accuracy.

Performance Improvement After Hyperparameter tuning

Significant improvement in model performance was observed after hyperparameter tuning. The test accuracy of each method before and after hyperparameter tuning is depicted below.



Final Conclusions

Challenges

The challenges of the project are: -

- the unbalanced data
- the missing values (zero values)
- Overfitting models

More information on data could improve the performance of the model. In general, the above discussed machine learning models tends to overfit on an unbalanced data. GridsearchCV, used for hyperparameter tuning doesn't always suggests the best parameters necessarily.

Task Allocation

Jishnu Prakash Kunnanath Poduvattil

- Project Repository in GitHub: For keeping track and maintaining code
- binaryClassification.py
    - Read the data
    - Exploratory data analysis (EDA)
    - Missing value imputation
    - Pre-processing the data for training
        - Cleaning and Standardising
    - KNN Classification – Fitting the data, hyperparameter tuning, model performance
    - Decision Tree - Fitting the data, hyperparameter tuning, model performance
- ReportBC.pdf
    - Prepared report and compared the results of each method
- EDA_Diabetes.ipynb (Colab file link)
  A detailed exploratory data analysis was done for better understanding of the dataset. Relationships between the feature vectors were visualised and explained in this interactive notebook.
- Readme.txt
  Consists of file information and how to run the script to reproduce the results.

Akhil Raj

- binaryClassification.py
    - Logistic regression – Fitting the data, hyperparameter tuning, roc curve and auc.
- ReportBC.pdf
    - Documentation of Logistic regression and the performance.