Assignment 2 Report
Data Mining and Machine Learning

Anjan Mondal
MDS202208

March 14, 2023

1 Introduction

This assignment consists of two tasks:

Task 1
Use decision tree regression to predict the revenue for a movie in the dataset Hindi Movies from 2005-2017.

Task 2
Build the following ensemble classifiers for both datasets and compare the performance with the classifiers built in Assignment 1.

- Bagging with decision trees (or a random forest).
- Boosting with decision stumps

2 Task 1: Movies Revenue Prediction

In this task, we use the movies data to perform a regression task. We are given a set of attributes like Number of Screens, Budget, Lead Star, Director etc. and we need to predict the revenue for the movie. The data was preprocessed to prepare it for the modeling stage. This involved converting categorical variables into numerical variables, specifically One Hot Vectors using the `get_dummies` function in pandas. The final preprocessed data was split into a training set and a testing set in a ratio of 80:20. We use `train_test_split` function from the `sklearn` library and set the parameter `stratify=y` to gurantee equal proportion of all class labels in the test and the training sets. `random_state=40` is used for the split.

2.1 Modeling

In this section, we describe the steps taken to fit the decision tree regression model to the preprocessed movie data.

2.1.1 Decision Tree

The decision tree model was fit to the training data using `sklearn` library's `GridSearchCV` function with `DecisionTreeRegressor()` as its estimator. The parameters `max_depth`, `min_samples_split`, `min_samples_leaf` were fitted for a range of values to obtain a best combination of the parameters on the training set. The defualt 5-fold cross validation was implemented for each combination of the parameters. The model was trained using the default Gini impurity as the splitting criterion. The best parameters turned out to be the following : `{'max_depth': 4, 'min_samples_leaf': 4, 'min_samples_split': 2}`. The time required to fit the model and find the best parameters is: `100.61 seconds`.

2.2 Evaluation

In this section, we compare the performance of the two models using various evaluation metrics such as coefficient of determination (R^2), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE). The performance metrics of the models were calculated by comparing the predictions of the models with the actual values in the testing set. The following results were obtained:

Table 1: Decision Tree Results

R^2	0.758
MSE	1.4937154784984432e+16
RMSE	122217653.328
MAE	0.843

3 Task 2: Bank & Movie Data Prediction

- We use a bank dataset to perform a classification task. We are given a set of attributes like Age, Education, Credit Card Defualt etc. and we need to predict whether a client subscribes to a term deposit.
- We use a bollywood movie dataset to perform a task of classification if a movie would be hit or flop based on features like Whether it is a remake, Genre, Lead Actor etc.

For the movie dataset, we had to create a separate column indicating whether the film was a hit or a flop. This target variable is set to `'hit'` if the revenue is greater than the budget and `'flop'` otherwise.
Both datasets were preprocessed in a similar way as Task 1. The final preprocessed data was split into a training set and a testing set in a ratio of 80:20.

3.1 Modeling

In this section, we describe the steps taken to fit the bagging and boosting models to the preprocessed bank data.

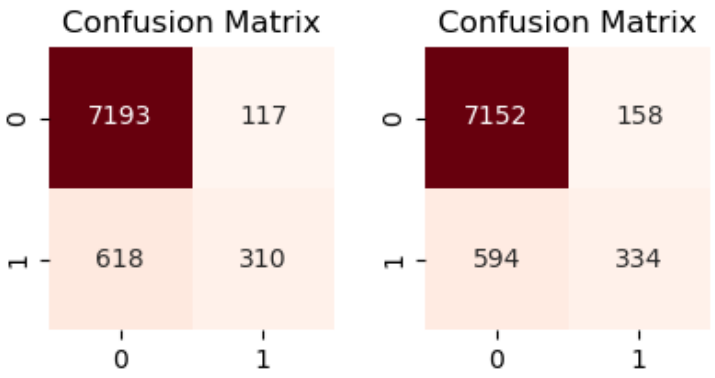
3.1.1 Bagging with Decision Trees

The bagging with decision trees model was fit using `BaggingClassifier` method of `scikit-learn` with `DecisionTreeClassifier()` as the `estimator` parameter. The parameters used were: `n_estimators = 200, max_features = 0.5`. The time required to fit the model is: `17.48 seconds` for the bank-data and `3.19 seconds` for the movie data.

3.1.2 Boosting with Stumps

The boosting with decision stumps model was fit using `AdaBoostClassifier` method of `scikit-learn` with `DecisionTreeClassifier(max_depth=1)` as the `estimator` parameter. The time required to fit the model is: `5.81 seconds` for the bank-data and `1.37 seconds` for the movie data.

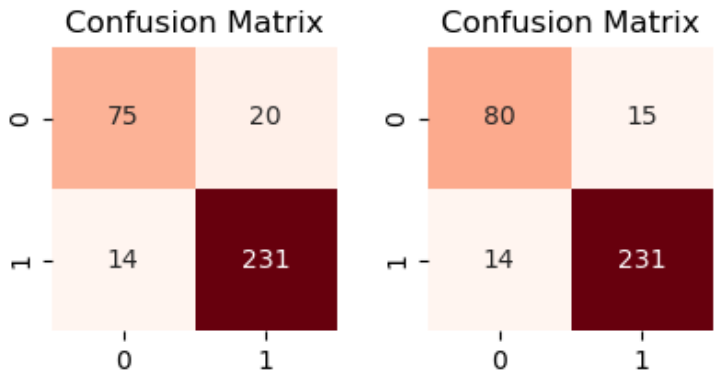
3.2 Evaluation



(a) Bagging Classifier (b) Boosting Classifier

Figure 1: Confusion Matrices for models trained on Bank Data

In this section, we compare the performance of the two models using various evaluation metrics such as accuracy, precision, recall, and F1 score.
The accuracy of the models was calculated by comparing the predictions of the models with the actual values in the testing set.



(a) Bagging Classifier (b) Boosting Classifier

Figure 2: Confusion Matrices for models trained on Movies Data

Table 2: Bank Dataset Results

Metric	Yes	No
Accuracy	0.911	0.911
Precision	0.726	0.921
Recall	0.334	0.984
F1 Score	0.458	0.951

Table 3: Bagging Results

Metric	Yes	No
Accuracy	0.909	0.909
Precision	0.679	0.923
Recall	0.923	0.978
F1 Score	0.470	0.950

Table 4: Boosting Results

Metric	Yes	No
Accuracy	0.913	0.913
Precision	0.646	0.939
Recall	0.510	0.965
F1 Score	0.570	0.952

Table 5: Decision Tree Results

Metric	Yes	No
Accuracy	0.890	0.890
Precision	0.514	0.920
Recall	0.348	0.958
F1 Score	0.415	0.939

Table 6: Naive Bayes

Table 7: Movie Dataset Results

Metric	Flop	Hit
Accuracy	0.900	0.900
Precision	0.843	0.920
Recall	0.920	0.943
F1 Score	0.815	0.931

Table 8: Bagging Results

Metric	Flop	Hit
Accuracy	0.915	0.915
Precision	0.851	0.939
Recall	0.842	0.943
F1 Score	0.847	0.941

Table 9: Boosting Results

Metric	Flop	Hit
Accuracy	0.915	0.915
Precision	0.851	0.939
Recall	0.842	0.943
F1 Score	0.847	0.941

Table 10: Decision Tree Results

Metric	Flop	Hit
Accuracy	0.803	0.803
Precision	0.671	0.845
Recall	0.579	0.890
F1 Score	0.621	0.867

Table 11: Naive Bayes Results