# DMML Assignment 2

Paarth Iyer - MCS202218
Varun Agrawal - MDS202251

March 14, 2023

## Preparation of data

The data for both parts has been prepared the same way as done in assignment 1 (of paarth).

# Task 1

## Decision Tree Regressor

The metric for evaluation used is r2 score as that gives an idea of how much variance of the true value is captured by the model.

Grid search is used to find the best parameters for the given data. The random state is fixed so we have an idea of a possible good values of these params.

Performing grid search took 1.4s, when using all the threads. It was performed on $1 \times 4 \times 1 \times 4 \times 4$ number of possible combinations

Here, the following params were deemed to give a good model :
`criterion='poisson', max_depth=8, max_features=0.85, min_samples_leaf=10`

## Result

The model, with random state 42, the r2 score is 0.8385. This is a good explainer of the variance of the data, ie the model does predicts the Revenue with a good accuracy.

# Task 2 : Bank dataset

To compare the results to the previous assignment, a decision tree classifier has been made along with a Random forest and Boosted decision stumps.

## Creating the classifiers

f1 score of 'yes' is used in this dataset so that we get a good metric to detect the possible 'yes' answers. Accuracy is not used as the percentage of no is about 90%, so accuracy would be high if majority of predictions are no, and the f1 score of 'no' will also be high due to a similar reason.

Grid search is used on the all three above classifiers to find the optimal hyper parameters to get the highest f1 score.

### Decision Tree Classifier

Grid search is used to vary the classes : min_samples_leaf, max_depth, criterion and max_features. The following were the results with random state = 42 :

`class_weight='balanced', criterion='entropy', max_depth=8, max_features=0.93, min_samples_leaf=6`
We are using class weight balanced as that helps remove the imbalance in the class frequency. The gridsearch takes 21.5s on $4 \times 4 \times 2 \times 5$ number of possible combinations after some hit and trial to find a good range of possible values.

### Random Forest Classifier

Grid search is used to vary the classes : n_estimator, max_depth and criterion. The following were the results with random state = 42 :

```
class_weight='balanced', max_depth=13, n_estimators=120
```

class_weight = balanced is used again for the reason stated above. The gridsearch takes 84.7s on $3 \times 4 \times 2$ number of possible combinations

### Boosted Decision Stumps

For boosting, Adaboost from sklearn is used with the estimator being a decision tree classifier with max depth being 1.

Grid search is used to vary the classes : n_estimator and learning_rate .The following were the results with random state = 42 for the adaboost classifier :

```
learning_rate=1.9, n_estimators=20
```

class_weight = balanced is used again for the reason stated above. The gridsearch takes 25.0s on $5 \times 5$ number of possible combinations

## Results and Comparison

Using decision tree classifiers :

| . | precision | recall | f1-score |
|---|---|---|---|
| no | 0.95 | 0.84 | 0.89 |
| yes | 0.34 | 0.64 | 0.44 |
| accuracy | . | . | 0.82 |

Using Random forest classifier :

| . | precision | recall | f1-score |
|---|---|---|---|
| no | 0.94 | 0.90 | 0.92 |
| yes | 0.42 | 0.58 | 0.48 |
| accuracy | . | . | 0.86 |

Using Adaboosted decision stumps :

| . | precision | recall | f1-score |
|---|---|---|---|
| no | 0.94 | 0.90 | 0.92 |
| yes | 0.41 | 0.53 | 0.46 |
| accuracy | . | . | 0.86 |

From the above, we can see that the random forest and boosted decision stumps perform slightly better than the decision tree in f1 score for both classes, and the overall performance in general is better.

Comparing random forest and boosted stumps, the random forest is slightly better in recall and f1 score of 'yes', and has similar performance otherwise.

The decision tree has better recall but worse precision for 'yes' than the other two models, with worse accuracy overall.

# Task 2 : Bollywood dataset

To compare the results to the previous assignment, a decision tree classifier has been made along with a Random forest and Boosted decision stumps.

## Creating the classifiers

In this task, accuracy is used as a metric to be maximized as unlike the bank dataset, the classes are about equally distributed. Grid search is used on the all three above classifiers to find the optimal hyper parameters to get the highest accuracy score.

### Decision Tree Classifier

Grid search is used to vary the classes : min_samples_leaf, max_depth, criterion and max_features. The following were the results with random state = 100 :

```
criterion='entropy', max_depth=7, max_features=0.85, min_samples_leaf=10
```

The gridsearch takes 2.7s on 3 × 4 × 2 × 4 number of possible combinations after some hit and trial to find a good range of possible values.

### Random Forest Classifier

Grid search is used to vary the classes : n_estimator and criterion. The following were the results with random state = 100 :

```
class_weight='balanced', max_depth=13, n_estimators=120
```

The gridsearch takes 3.6s on 4 × 2 number of possible combinations

### Boosted Decision Stumps

For boosting, Adaboost from sklearn is used with the estimator being a decision tree classifier with max depth being 1.

Grid search is used to vary the classes : n_estimator and learning_rate .The following were the results with random state = 100 for the adaboost classifier :

```
learning_rate=1.9, n_estimators=20
```

The gridsearch takes 2.8s on 4 × 5 number of possible combinations

## Results and Comparison

Using decision tree classifiers :

| . | precision | recall | f1-score |
|---|---|---|---|
| no | 0.80 | 0.85 | 0.83 |
| yes | 0.93 | 0.91 | 0.92 |
| accuracy | . | . | 0.89 |

Using Random forest classifier :

| . | precision | recall | f1-score |
|---|---|---|---|
| no | 0.82 | 0.80 | 0.81 |
| yes | 0.91 | 0.92 | 0.92 |
| accuracy | . | . | 0.89 |

Using Adaboosted decision stumps :

| . | precision | recall | f1-score |
|---|---|---|---|
| no | 0.81 | 0.84 | 0.82 |
| yes | 0.93 | 0.91 | 0.92 |
| accuracy | . | . | 0.89 |

In this dataset, it appears that all the three models perform equally. Even on other random states, we get similar results.

Decision tree performs slightly better on some metrics, random forest is better on some and boosted stumps on some. Thus, it may mean that on this particular dataset, we don't have to use more expensive models like random forest and can go with a decision tree or boosted stumps.