CS6220
Big Data System and Analytics
Assignment 5

Sathvik Karatattu Padmanabha

October 2024

Student Session: cs6220-A
GT ID: xx4032361

Topic: Programming

Problem 2.3:

The Power of Random Forest

Contents

1 Introduction and Readme

1.1 Problem

I have chosen the Power of Random Forests option for this assignment. I have chosen the adult income dataset for this experiment [7]. I have compared 2 variations of Decision Tree and 6 configurations for random forests in this assignment.

1.2 Execution Environment

I have chosen scikit-learn (sklearn) as the ML library [8]. This is because it has a very good random forest library. I have used the PACE environment with NVIDIA Tesla V100-PCIE-32GB (GPU not necessary) to execute the experiments and used the jupyter interactive app in the PACE dashboard for convenience. I used anaconda for library management.

The .ipynb book submitted in the zip file contains the complete code. The majority of the code is mostly inspired from two online sources [1]. However, I have added changes as required by the problem statement and other references are mostly from sklearn documentation [8] and other notebooks such as [2], [3] and [5].

The github repo containing the code is provided below. However, you need to request access from me as it is private. I have anyways provided the .ipynb book along with the submission.

https://github.com/SathvikKP/Random_forests_analysis

To setup the environment, I have provided the anaconda .yml file. But here is a list of commands that I used to setup the environment.

```
module load anaconda3
conda create --prefix ./bda_5 python=3.9
conda activate ./bda_5/
conda install -c conda-forge matplotlib numpy seaborn ipykernel
scikit-learn category_encoders pandas pydot graphviz
conda install -c conda-forge ipykernel
python -m ipykernel install --user --name bda_5 --display-name "bda_5"
conda install -c anaconda python-graphviz
conda env export > environment.yml
```

2 Notebook Details

I have neatly added the markup cells describing the actions taking place in the code cells.

Most of the code was taken from [1], [2] and [3]. I have modified the code according to the assignment requirements and modularized the functions in original code to make it easy for the reader to analyze.

- Step 1 : Import necessary libraries
- Step 2: Data Cleaning functions
- Step 3: Exploratory Data Analysis Function
- Step 4: Comparison of Various ML Models (for chosen dataset)
- Step 5: Auxillary functions for Random Forest Testing (for various tasks such as visualization, training and testing)
- Step 6: Main decision tree and random forest comparison function
- Step 7: Read the data and clean it
- Step 8: Perform Exploratory Data Analysis and Train:Test Split
- Step 9: Compare Different ML Models
- Step 10: Compare Decision Tree and Random Forests with various configurations

Note: Steps 4 and 9 are not required by the assignment, but I did it for some interesting observations.

3 Dataset Details

3.1 Dataset description and examples

I have chosen the adult income dataset for this experiment [7] (dataset info from [6]). This is a good dataset with lots of features and many examples. I chose a subset of 15000 examples total. This is because I wanted to limit the accuracy for analysis and for faster execution of the models. I have chosen a train:test split of 80:20. Table 1 gives description of raw data (before pre-processing).

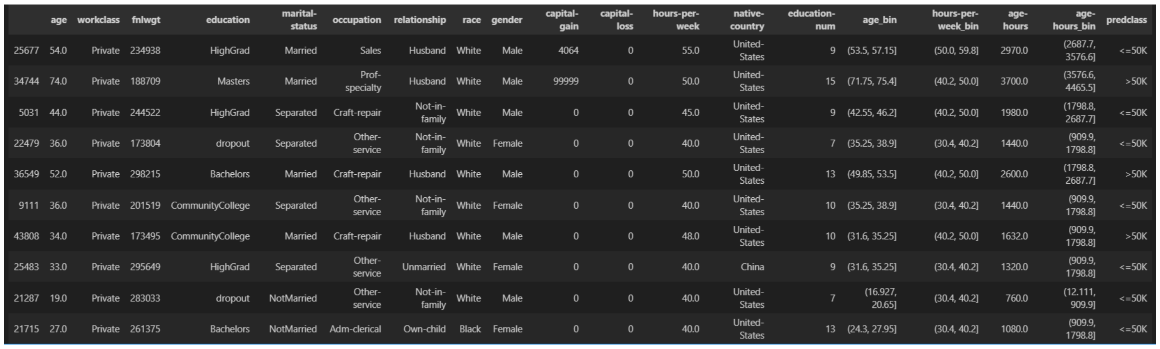
Attribute	Value Range / Description
age	Continuous (numerical)
workclass	Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked
fnlwgt	Continuous (numerical)
education	Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool
education-num	Continuous (numerical)
marital-status	Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse
occupation	Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces
relationship	Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried
race	White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black
sex	Female, Male
capital-gain	Continuous (numerical)
capital-loss	Continuous (numerical)
hours-per-week	Continuous (numerical)
native-country	United-States, Cambodia, England, Puerto-Rico, Canada, Germany, ...
income	<=50K, >50K

Table 1: Attributes and Value Ranges in the Dataset. Continuous value ranges are present in Figure 5. (Note: This is description for before data cleaning, encoding and scaling; Post this process,the ranges are [-1,1] for all features except income)

The data required significant pre-processing, some feature engineering to make it fit before executing the training process. Some redundant and unwanted features were removed, some new features were created to improve the performance.

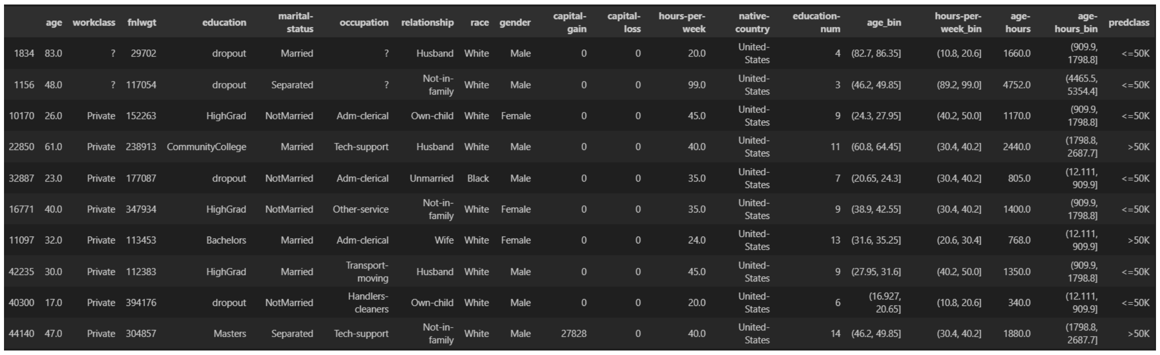
Then, I performed exploratory data analysis to get some insights. After this process, I decided to apply label encoding and scaled the values to [-1, 1]. This was done for performance and accuracy reasons.

Figures 1, 2, 3 and 4 show training and testing examples. Figures 1 and 2 show the training examples after some feature engineering (creating new features and renaming some features). Figures 3 and 4 are same examples as shown in Figure 1 and 2 respectively, but they are the final data representation that will be used for training.



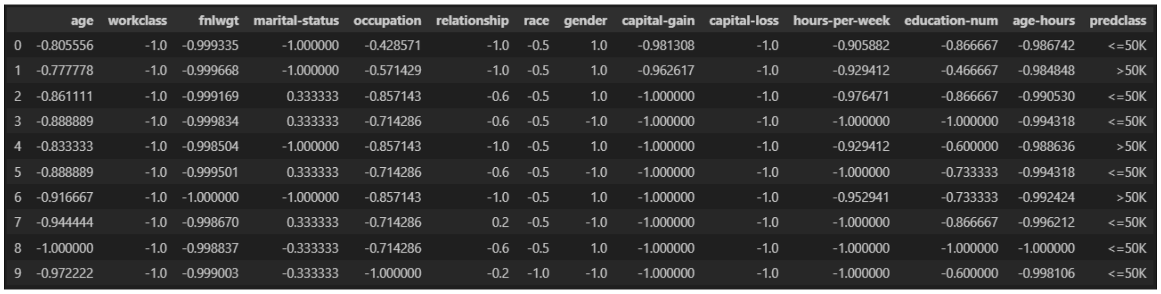
	age	workclass	fnlwgt	education	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	native-country	education-num	age_bin	hours-per-week_bin	age-hours	age-hours_bin	predclass
25677	54.0	Private	234938	HighGrad	Married	Sales	Husband	White	Male	4064	0	55.0	United-States	9	(53.5, 57.15]	(50.0, 59.8]	2970.0	(2687.7, 3576.6]	<=50K
34744	74.0	Private	188709	Masters	Married	Prof-specialty	Husband	White	Male	99999	0	50.0	United-States	15	(71.75, 75.4]	(40.2, 50.0]	3700.0	(3576.6, 4465.5]	>50K
5031	44.0	Private	244522	HighGrad	Separated	Craft-repair	Not-in-family	White	Male	0	0	45.0	United-States	9	(42.55, 46.2]	(40.2, 50.0]	1980.0	(1798.8, 2687.7]	<=50K
22479	36.0	Private	173804	dropout	Separated	Other-service	Not-in-family	White	Female	0	0	40.0	United-States	7	(35.25, 38.9]	(30.4, 40.2]	1440.0	(909.9, 1798.8]	<=50K
36549	52.0	Private	298215	Bachelors	Married	Craft-repair	Husband	White	Male	0	0	50.0	United-States	13	(49.85, 53.5]	(40.2, 50.0]	2600.0	(1798.8, 2687.7]	>50K
9111	36.0	Private	201519	CommunityCollege	Separated	Other-service	Not-in-family	White	Female	0	0	40.0	United-States	10	(35.25, 38.9]	(30.4, 40.2]	1440.0	(909.9, 1798.8]	<=50K
43808	34.0	Private	173495	CommunityCollege	Married	Craft-repair	Husband	White	Male	0	0	48.0	United-States	10	(31.6, 35.25]	(40.2, 50.0]	1632.0	(909.9, 1798.8]	>50K
25483	33.0	Private	295649	HighGrad	Separated	Other-service	Unmarried	White	Female	0	0	40.0	China	9	(31.6, 35.25]	(30.4, 40.2]	1320.0	(909.9, 1798.8]	<=50K
21287	19.0	Private	283033	dropout	NotMarried	Other-service	Not-in-family	White	Male	0	0	40.0	United-States	7	(16.927, 20.65]	(30.4, 40.2]	760.0	(12.111, 909.9]	<=50K
21715	27.0	Private	261375	Bachelors	NotMarried	Adm-clerical	Own-child	Black	Female	0	0	40.0	United-States	13	(24.3, 27.95]	(30.4, 40.2]	1080.0	(909.9, 1798.8]	<=50K

Figure 1: Sample Training Examples (Before Label Encoding and Scaling).



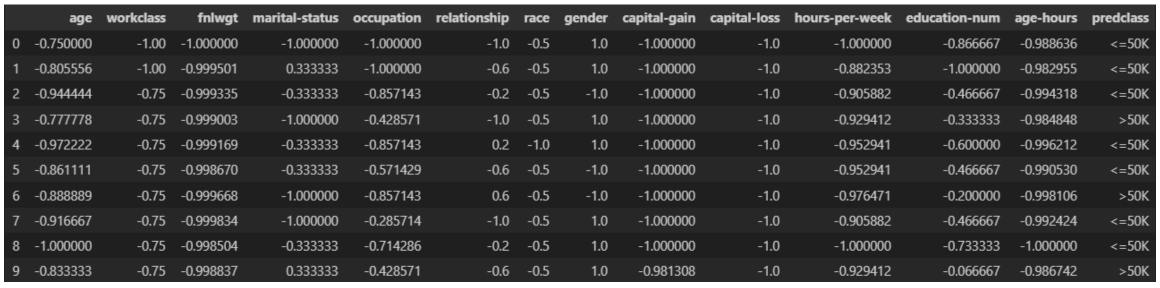
	age	workclass	fnlwgt	education	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	native-country	education-num	age_bin	hours-per-week_bin	age-hours	age-hours_bin	predclass
1834	83.0	?	29702	dropout	Married	?	Husband	White	Male	0	0	20.0	United-States	4	(82.7, 86.35]	(10.8, 20.6]	1660.0	(909.9, 1798.8]	<=50K
1156	48.0	?	117054	dropout	Separated	?	Not-in-family	White	Male	0	0	99.0	United-States	3	(46.2, 49.85]	(89.2, 99.0]	4752.0	(4465.5, 5354.4]	<=50K
10170	26.0	Private	152263	HighGrad	NotMarried	Adm-clerical	Own-child	White	Female	0	0	45.0	United-States	9	(24.3, 27.95]	(40.2, 50.0]	1170.0	(909.9, 1798.8]	<=50K
22850	61.0	Private	238913	CommunityCollege	Married	Tech-support	Husband	White	Male	0	0	40.0	United-States	11	(60.8, 64.45]	(30.4, 40.2]	2440.0	(1798.8, 2687.7]	>50K
32887	23.0	Private	177087	dropout	NotMarried	Adm-clerical	Unmarried	Black	Male	0	0	35.0	United-States	7	(20.65, 24.3]	(30.4, 40.2]	805.0	(12.111, 909.9]	<=50K
16771	40.0	Private	347934	HighGrad	NotMarried	Other-service	Not-in-family	White	Female	0	0	35.0	United-States	9	(38.9, 42.55]	(30.4, 40.2]	1400.0	(909.9, 1798.8]	<=50K
11097	32.0	Private	113453	Bachelors	Married	Adm-clerical	Wife	White	Female	0	0	24.0	United-States	13	(31.6, 35.25]	(20.6, 30.4]	768.0	(12.111, 909.9]	>50K
42235	30.0	Private	112383	HighGrad	Married	Transport-moving	Husband	White	Male	0	0	45.0	United-States	9	(27.95, 31.6]	(40.2, 50.0]	1350.0	(909.9, 1798.8]	<=50K
40300	17.0	Private	394176	dropout	NotMarried	Handlers-cleaners	Own-child	White	Male	0	0	20.0	United-States	6	(16.927, 20.65]	(10.8, 20.6]	340.0	(12.111, 909.9]	<=50K
44140	47.0	Private	304857	Masters	Separated	Tech-support	Not-in-family	White	Male	27828	0	40.0	United-States	14	(46.2, 49.85]	(30.4, 40.2]	1880.0	(1798.8, 2687.7]	>50K

Figure 2: Sample Testing Examples (Before Label Encoding and Scaling).



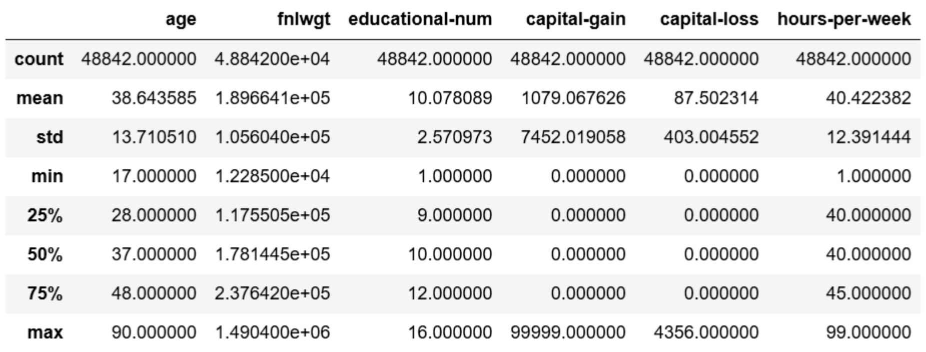
	age	workclass	fnlwgt	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	education-num	age-hours	predclass
0	-0.805556	-1.0	-0.999335	-1.000000	-0.428571	-1.0	-0.5	1.0	-0.981308	-1.0	-0.905882	-0.866667	-0.986742	<=50K
1	-0.777778	-1.0	-0.999668	-1.000000	-0.571429	-1.0	-0.5	1.0	-0.962617	-1.0	-0.929412	-0.466667	-0.984848	>50K
2	-0.861111	-1.0	-0.999169	0.333333	-0.857143	-0.6	-0.5	1.0	-1.000000	-1.0	-0.976471	-0.866667	-0.990530	<=50K
3	-0.888889	-1.0	-0.999834	0.333333	-0.714286	-0.6	-0.5	-1.0	-1.000000	-1.0	-1.000000	-1.000000	-0.994318	<=50K
4	-0.833333	-1.0	-0.998504	-1.000000	-0.857143	-1.0	-0.5	1.0	-1.000000	-1.0	-0.929412	-0.600000	-0.988636	>50K
5	-0.888889	-1.0	-0.999501	0.333333	-0.714286	-0.6	-0.5	-1.0	-1.000000	-1.0	-1.000000	-0.733333	-0.994318	<=50K
6	-0.916667	-1.0	-1.000000	-1.000000	-0.857143	-1.0	-0.5	1.0	-1.000000	-1.0	-0.952941	-0.733333	-0.992424	>50K
7	-0.944444	-1.0	-0.998670	0.333333	-0.714286	0.2	-0.5	-1.0	-1.000000	-1.0	-1.000000	-0.866667	-0.996212	<=50K
8	-1.000000	-1.0	-0.998837	-0.333333	-0.714286	-0.6	-0.5	1.0	-1.000000	-1.0	-1.000000	-1.000000	-1.000000	<=50K
9	-0.972222	-1.0	-0.999003	-0.333333	-1.000000	-0.2	-1.0	-1.0	-1.000000	-1.0	-1.000000	-0.600000	-0.998106	<=50K

Figure 3: Sample Training Examples (After Label Encoding and Scaling).



	age	workclass	fnlwgt	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	education-num	age-hours	predclass
0	-0.750000	-1.00	-1.000000	-1.000000	-1.000000	-1.0	-0.5	1.0	-1.000000	-1.0	-1.000000	-0.866667	-0.988636	<=50K
1	-0.805556	-1.00	-0.999501	0.333333	-1.000000	-0.6	-0.5	1.0	-1.000000	-1.0	-0.882353	-1.000000	-0.982955	<=50K
2	-0.944444	-0.75	-0.999335	-0.333333	-0.857143	-0.2	-0.5	-1.0	-1.000000	-1.0	-0.905882	-0.466667	-0.994318	<=50K
3	-0.777778	-0.75	-0.999003	-1.000000	-0.428571	-1.0	-0.5	1.0	-1.000000	-1.0	-0.929412	-0.333333	-0.984848	>50K
4	-0.972222	-0.75	-0.999169	-0.333333	-0.857143	0.2	-1.0	1.0	-1.000000	-1.0	-0.952941	-0.600000	-0.996212	<=50K
5	-0.861111	-0.75	-0.998670	-0.333333	-0.571429	-0.6	-0.5	-1.0	-1.000000	-1.0	-0.952941	-0.466667	-0.990530	<=50K
6	-0.888889	-0.75	-0.999668	-1.000000	-0.857143	0.6	-0.5	-1.0	-1.000000	-1.0	-0.976471	-0.200000	-0.998106	>50K
7	-0.916667	-0.75	-0.999834	-1.000000	-0.285714	-1.0	-0.5	1.0	-1.000000	-1.0	-0.905882	-0.466667	-0.992424	<=50K
8	-1.000000	-0.75	-0.998504	-0.333333	-0.714286	-0.2	-0.5	1.0	-1.000000	-1.0	-1.000000	-0.733333	-1.000000	<=50K
9	-0.833333	-0.75	-0.998837	0.333333	-0.428571	-0.6	-0.5	1.0	-0.981308	-1.0	-0.929412	-0.066667	-0.986742	>50K

Figure 4: Sample Testing Examples (After Label Encoding and Scaling).



	age	fnlwgt	educational-num	capital-gain	capital-loss	hours-per-week
count	48842.000000	4.884200e+04	48842.000000	48842.000000	48842.000000	48842.000000
mean	38.643585	1.896641e+05	10.078089	1079.067626	87.502314	40.422382
std	13.710510	1.056040e+05	2.570973	7452.019058	403.004552	12.391444
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.175505e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.781445e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.376420e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.490400e+06	16.000000	99999.000000	4356.000000	99.000000

Figure 5: Data Value ranges for continuous variables

3.2 Exploratory Data Analysis

Figures 6-14 show various graphs and plots for various features to get an understanding of the data.

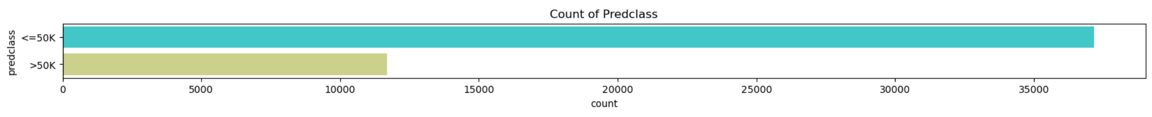


Figure 6: Count of Predclass

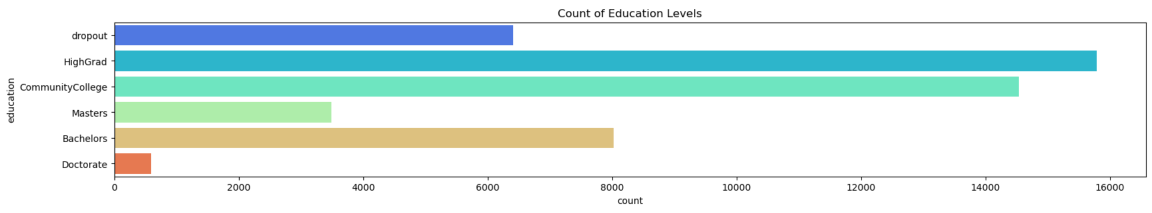


Figure 7: Count of Education Levels

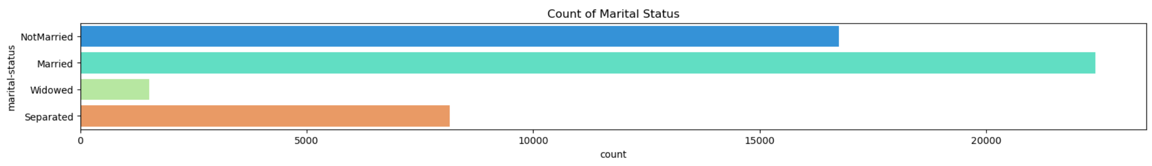


Figure 8: Count of Marital Status

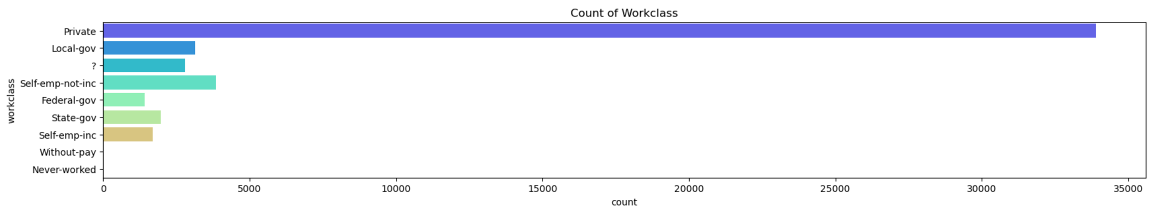


Figure 9: Count of Workclass

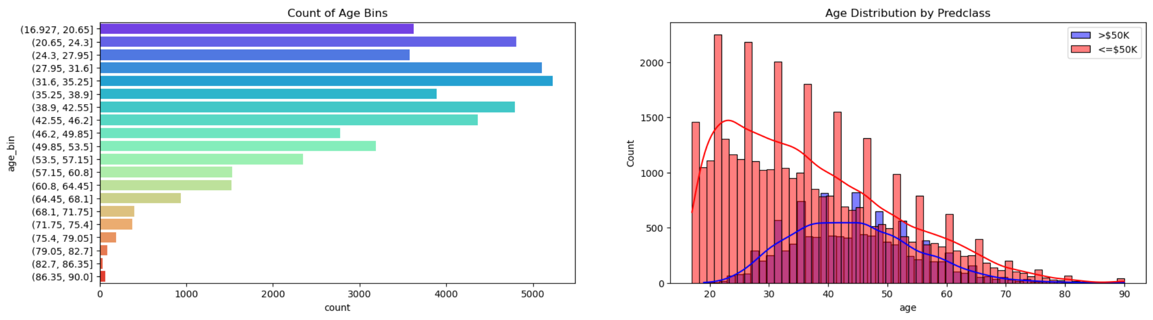


Figure 10: Age Distribution by Predclass

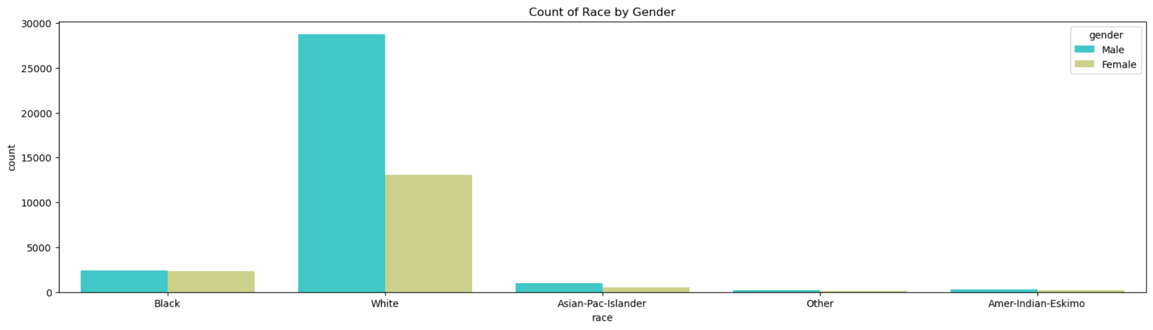


Figure 11: Count of Race by Gender

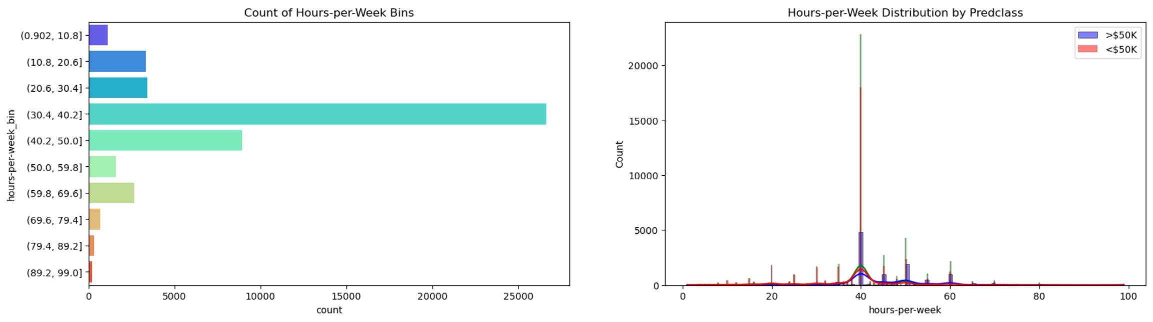


Figure 12: Hours-per-Week Distribution by Predclass

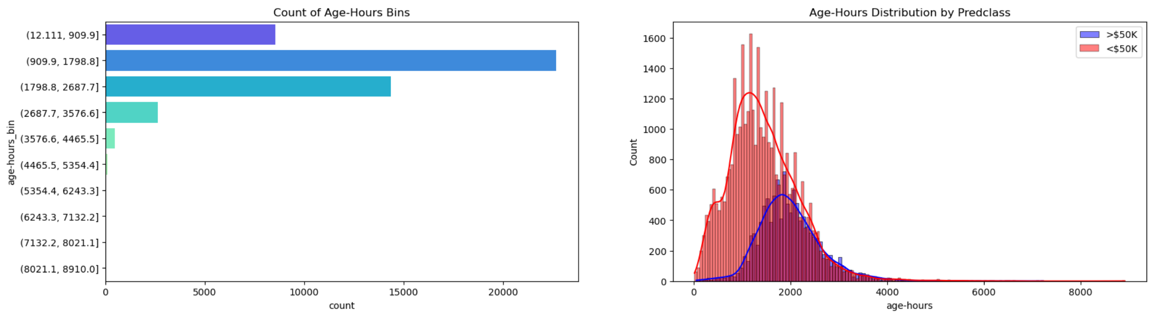


Figure 13: Age-Hours Distribution by Predclass

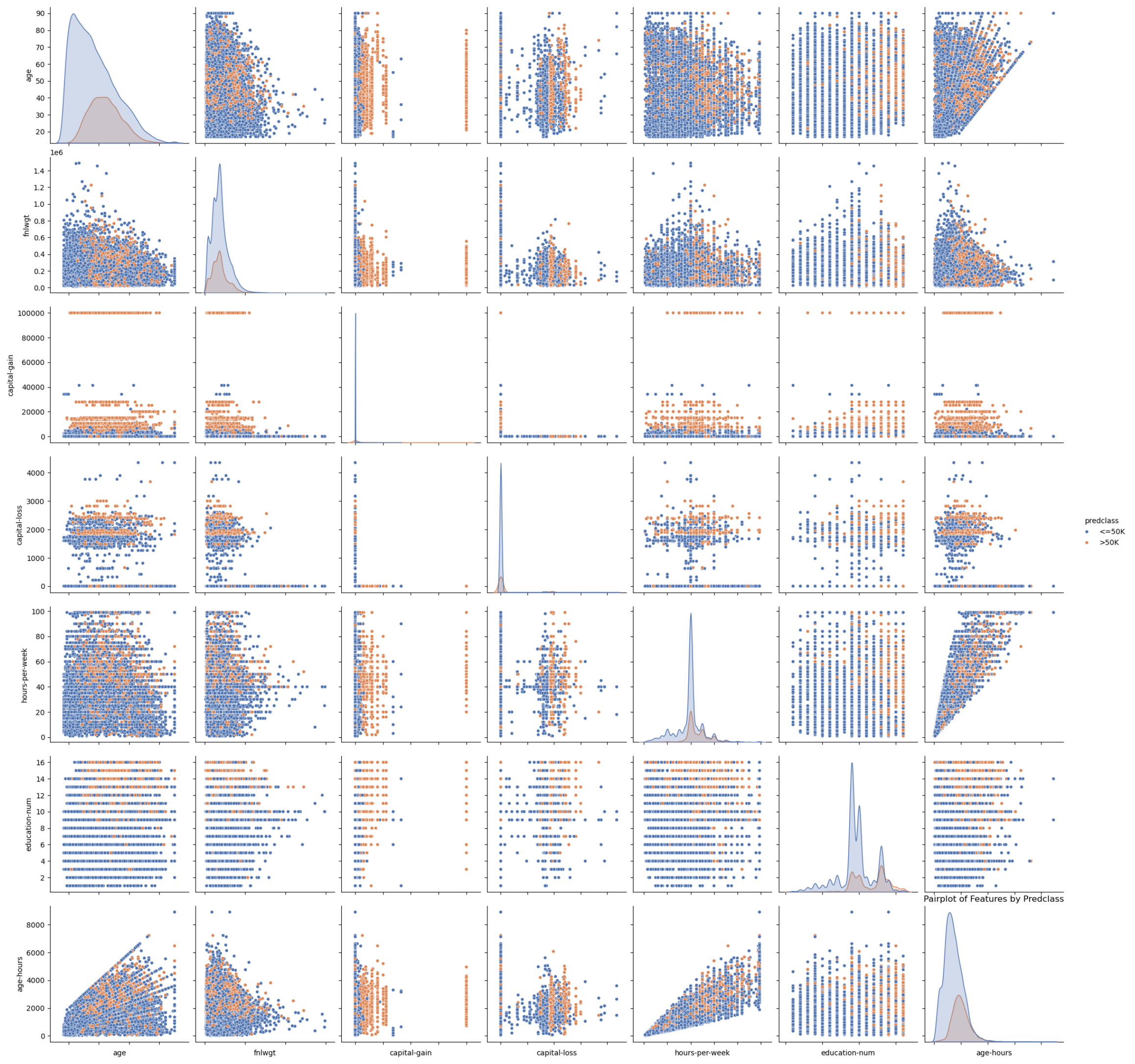


Figure 14: Pairplot of Features by Predclass

Analysis: Some interesting observations are listed below

- We have a lot of <=50k examples than > 50k examples. This might affect the overall accuracy. In an ideal scenario, number of examples must be comparable.
- Education levels, maritial status are new features created (feature engineering) and is much better representation than what was originally in the dataset (Table 1).
- Age, hours-per week was a continuous variable which we have split into bins (now a discrete variable) for random forest classifier.
- The final Pairplot figure (Figure 14) shows us that there exists some correlation among certain features. We should ideally resolve such issues.

4 Decision Trees and Random Forests

4.1 Introduction

Decision Trees and Random Forests are two versatile machine learning models that are applicable to many machine learning tasks.

Decision tree is an independent model that makes predictions based on a series of decisions whereas random forest is group of multiple decision trees, which work to improve the overall prediction accuracy. The accuracy of decision tree is low and sensitive to variations in training data whereas random forest provides an improved accuracy.

Random Forest algorithm is a powerful tree learning technique in Machine Learning. It works by creating a number of Decision Trees during the training phase. Each tree is constructed using a random subset of the data set to measure a random subset of features in each partition. This randomness introduces variability among individual trees, reducing the risk of overfitting and improving overall prediction performance.

In prediction, the algorithm aggregates the results of all trees, either by voting (for classification tasks) or by averaging (for regression tasks) This collaborative decision-making process, supported by multiple trees with their insights, provides an example stable and precise results. Random forests are widely used for classification and regression functions, which are known for their ability to handle complex data, reduce overfitting, and provide reliable forecasts in different environments.

(References : [4], [9])

4.2 Experiment Details

I have chosen decision tree with two different configuration and random forest with 6 different configurations and have performed a comparative analysis using all of them.

Sklearn has very good DecisionTree and Random Forest classifiers and we can call the functions DecisionTreeClassifier() and RandomForestClassifier(n_estimators=25) respectively.

The CART Decision Tree branches based on information gain (gini). However there is one more configuration which branches based on entropy. This is almost similar to C4.5 but not exactly C4.5. I have chosen both these configurations. Next I have chosen random forests classifier with 5, 10, 25, 50, 75 and 100 trees.

Table 2 shows the hyper-parameter configurations for all the models. Most of the hyper-parameters are same. This is because I wanted a fair comparison among all models. The only difference between decision trees is criterion (gini vs entroypy), and among random forests, the difference is number of trees (n_estimators). In a real world scenario, all the hyper-parameters need to be experimented upon. For example, trees might need to be pruned for performance and resource constraints.

I have neatly modularized the code so that it prints the various required statistics making it easy to analyze.

Table 2: Hyperparameters for Decision Tree and Random Forest Models

Hyperparameter	**Decision Tree (CART)**	**Decision Tree (Entropy)**	**Random Forest (5 trees)**	**Random Forest (10 trees)**	**Random Forest (25 trees)**	**Random Forest (50 trees)**	**Random Forest (75 trees)**	**Random Forest (100 trees)**
ccp_alpha	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
class_weight	None	None	None	None	None	None	None	None
criterion	gini	entropy	gini	gini	gini	gini	gini	gini
max_depth	NA	NA	NA	NA	NA	NA	NA	NA
max_features	NA	NA	sqrt	sqrt	sqrt	sqrt	sqrt	sqrt
max_leaf_nodes	NA	NA	NA	NA	NA	NA	NA	NA
min_impurity_decrease	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
min_samples_leaf	1	1	1	1	1	1	1	1
min_samples_split	2	2	2	2	2	2	2	2
min_weight_fraction_leaf	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
monotonic_cst	NA	NA	NA	NA	NA	NA	NA	NA
random_state	100	100	100	100	100	100	100	100
splitter	best	best	NA	NA	NA	NA	NA	NA
bootstrap	NA	NA	True	True	True	True	True	True
n_estimators	NA	NA	5	10	25	50	75	100
n_jobs	NA	NA	NA	NA	NA	NA	NA	NA
oob_score	NA	NA	False	False	False	False	False	False
verbose	NA	NA	0	0	0	0	0	0
warm_start	NA	NA	False	False	False	False	False	False

4.3 Experiment Results

Figures 15 and 16 show the accuracy curves and ROC Curves. We can see that Random forests outperform decision trees. We also observe that Entropy based decision tree slightly outperforms information gain based decision trees.

We can see increasing number of trees beyond 25 does not yield much improvement. From the ROC curves, we see arrive at the same conclusion, all the configurations above 25 trees have almost identical ROC curves.

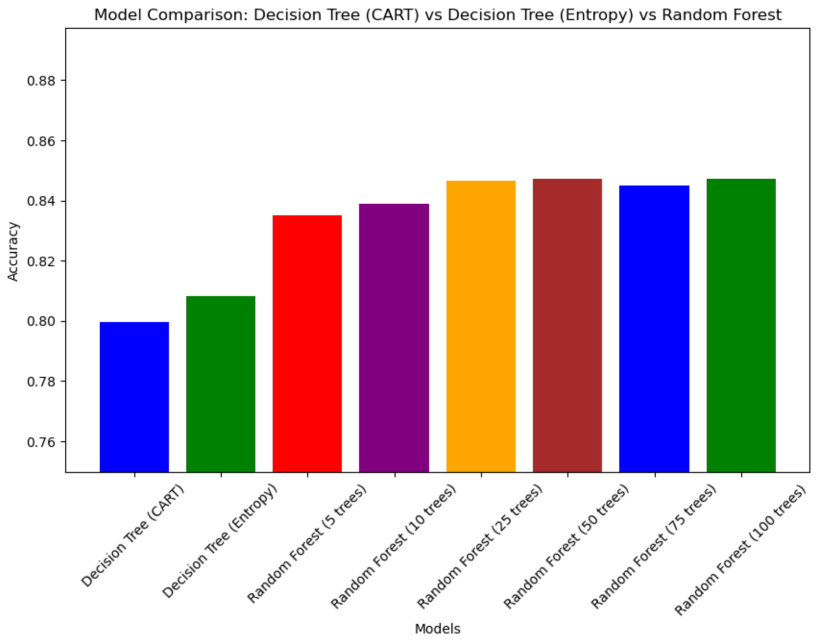


Figure 15: Model Comparison: Decision Tree (CART) vs Decision Tree (Entropy) vs Random Forest

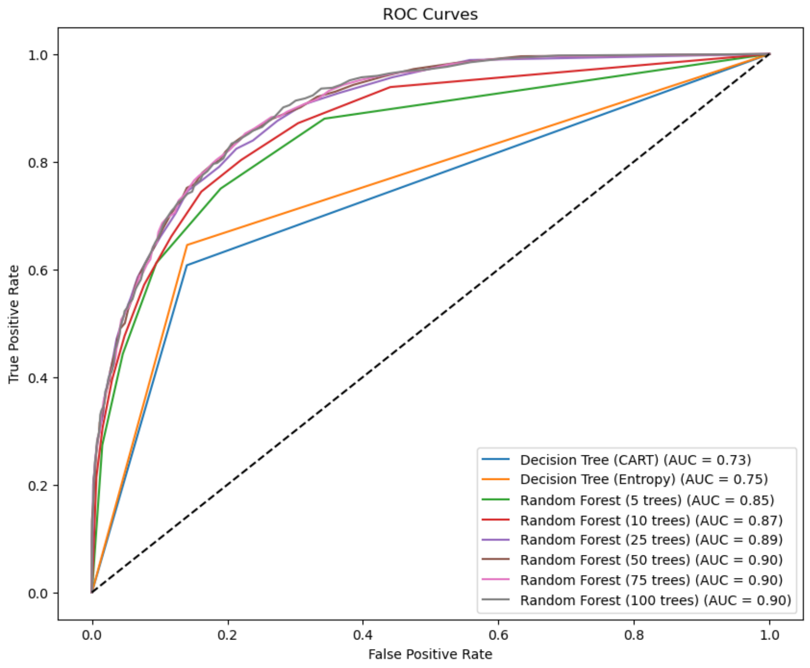


Figure 16: ROC Curves

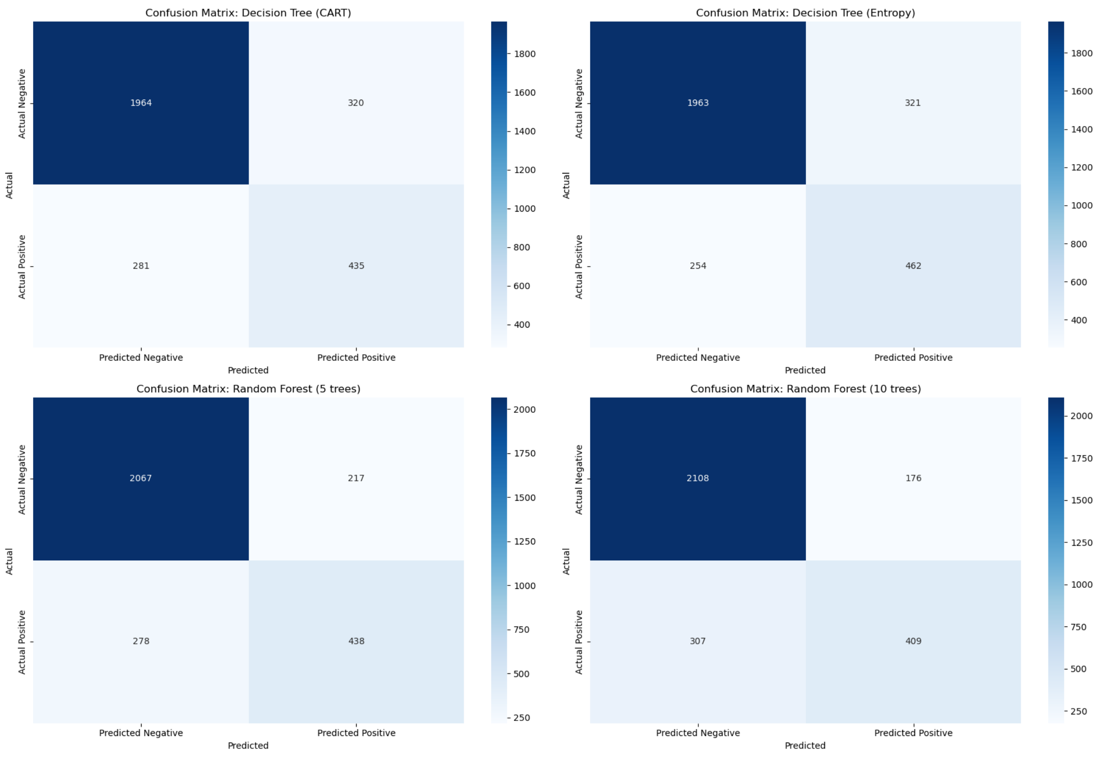


Figure 17: Confusion Matrices for first 4 experiments

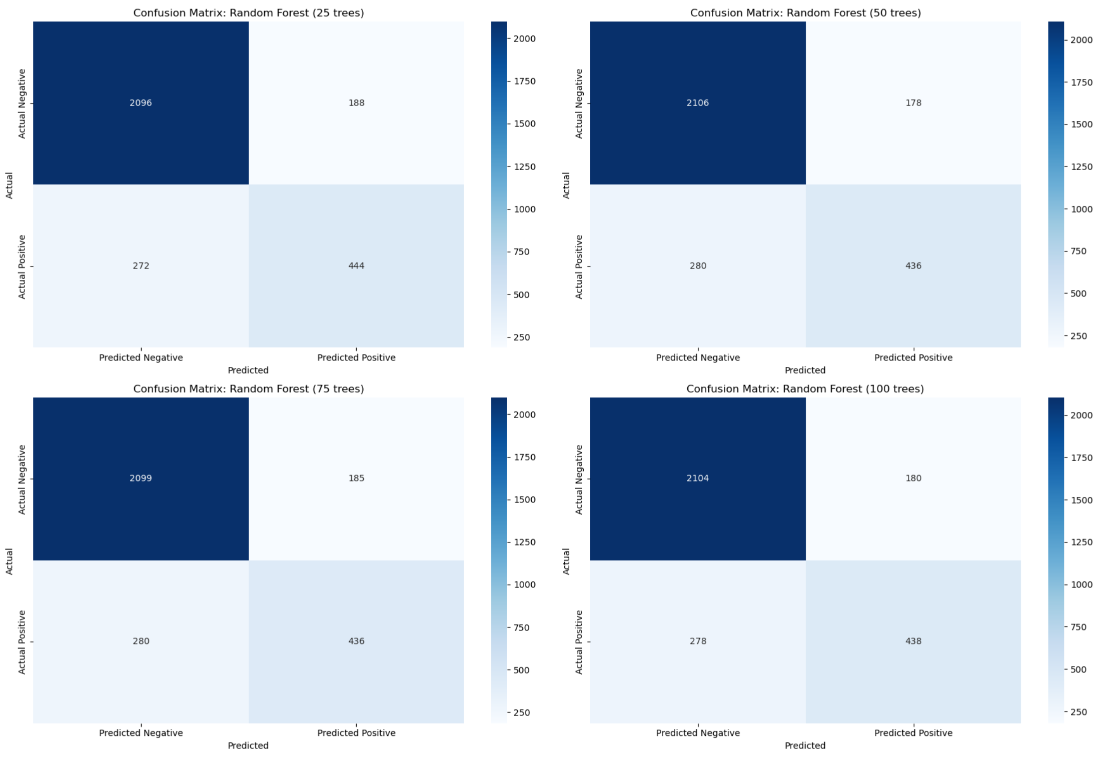


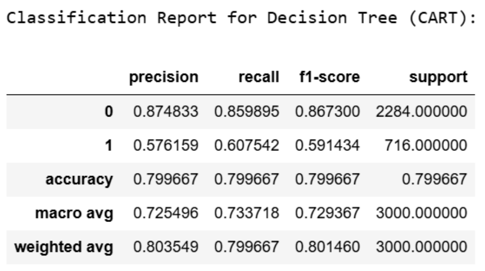
Figure 18: Confusion Matrices for next 4 experiments

Figures 17 and 18 show confusion matrices for all the experiments. Observations:

Analysis

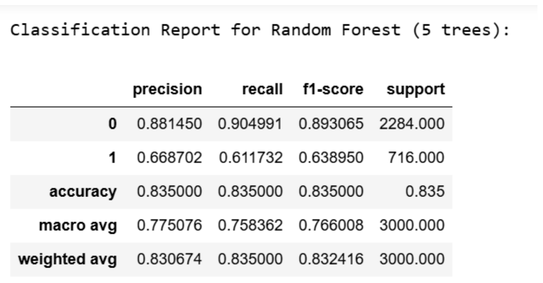
- We can see there are more negative classifications than positive classifications. This follows from the exploratory data analysis which showed a skewed balance of predicted class (Figure 6).
- Entropy based decision tree outperforms information gain based decision tree especially when predicting actual positive.
- Increasing random forests from 5 to 10 will increase accuracy for predicting actual negative but decrease accuracy for predicting actual positive
- There is almost no change in Figure 18. This shows blindly increasing number of decision trees from 25 –> 100 do not impact much in terms of accuracy.

Figures 19-26 show the classification reports showing precision and recall which cement the above observations made from the confusion matrices. See the recall value for random forests with 5 trees vs 10 trees. It is a considerable decrease. When we compare figures 23-26, they are almost similar which proves increasing trees beyond some threshold will not gain improvements.



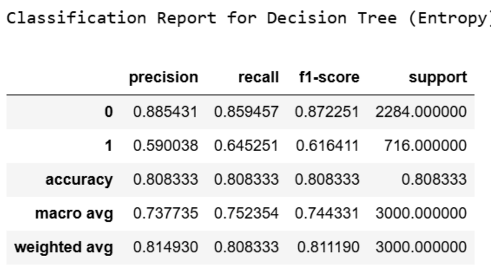
Classification Report for Decision Tree (CART):

	precision	recall	f1-score	support
0	0.874833	0.859895	0.867300	2284.000000
1	0.576159	0.607542	0.591434	716.000000
accuracy	0.799667	0.799667	0.799667	0.799667
macro avg	0.725496	0.733718	0.729367	3000.000000
weighted avg	0.803549	0.799667	0.801460	3000.000000



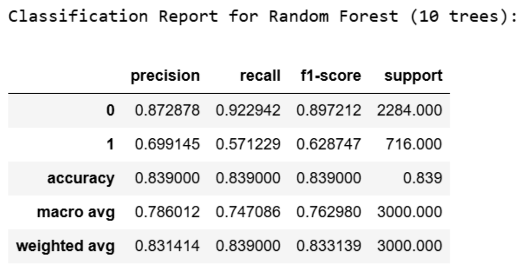
Classification Report for Random Forest (5 trees):

	precision	recall	f1-score	support
0	0.881450	0.904991	0.893065	2284.000
1	0.668702	0.611732	0.638950	716.000
accuracy	0.835000	0.835000	0.835000	0.835
macro avg	0.775076	0.758362	0.766008	3000.000
weighted avg	0.830674	0.835000	0.832416	3000.000



Classification Report for Decision Tree (Entropy

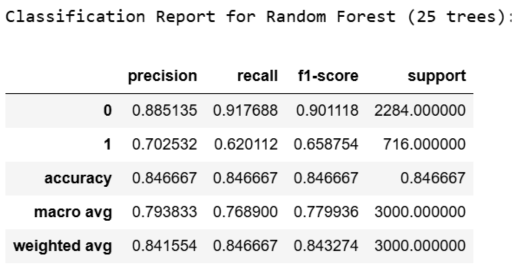
	precision	recall	f1-score	support
0	0.885431	0.859457	0.872251	2284.000000
1	0.590038	0.645251	0.616411	716.000000
accuracy	0.808333	0.808333	0.808333	0.808333
macro avg	0.737735	0.752354	0.744331	3000.000000
weighted avg	0.814930	0.808333	0.811190	3000.000000



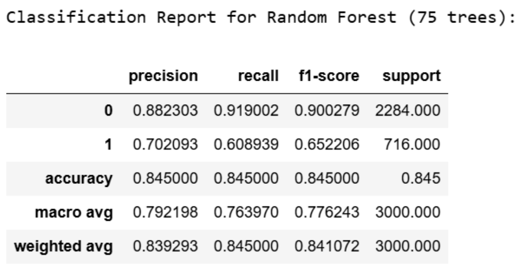
Classification Report for Random Forest (10 trees):

	precision	recall	f1-score	support
0	0.872878	0.922942	0.897212	2284.000
1	0.699145	0.571229	0.628747	716.000
accuracy	0.839000	0.839000	0.839000	0.839
macro avg	0.786012	0.747086	0.762980	3000.000
weighted avg	0.831414	0.839000	0.833139	3000.000

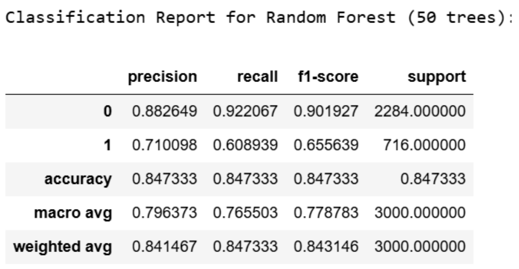
Figures 19 - 22: Classification reports for the first 4 experiments


Classification Report for Random Forest (25 trees):

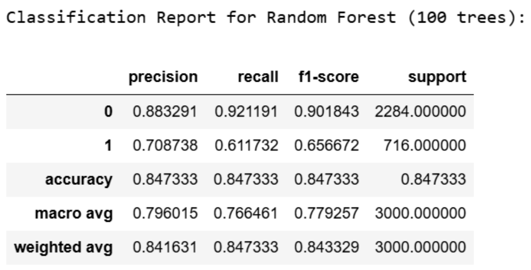
	precision	recall	f1-score	support
0	0.885135	0.917688	0.901118	2284.000000
1	0.702532	0.620112	0.658754	716.000000
accuracy	0.846667	0.846667	0.846667	0.846667
macro avg	0.793833	0.768900	0.779936	3000.000000
weighted avg	0.841554	0.846667	0.843274	3000.000000


Classification Report for Random Forest (75 trees):

	precision	recall	f1-score	support
0	0.882303	0.919002	0.900279	2284.000
1	0.702093	0.608939	0.652206	716.000
accuracy	0.845000	0.845000	0.845000	0.845
macro avg	0.792198	0.763970	0.776243	3000.000
weighted avg	0.839293	0.845000	0.841072	3000.000


Classification Report for Random Forest (50 trees):

	precision	recall	f1-score	support
0	0.882649	0.922067	0.901927	2284.000000
1	0.710098	0.608939	0.655639	716.000000
accuracy	0.847333	0.847333	0.847333	0.847333
macro avg	0.796373	0.765503	0.778783	3000.000000
weighted avg	0.841467	0.847333	0.843146	3000.000000

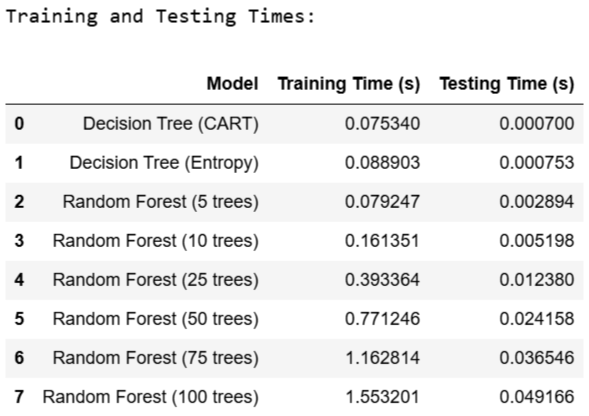

Classification Report for Random Forest (100 trees):

	precision	recall	f1-score	support
0	0.883291	0.921191	0.901843	2284.000000
1	0.708738	0.611732	0.656672	716.000000
accuracy	0.847333	0.847333	0.847333	0.847333
macro avg	0.796015	0.766461	0.779257	3000.000000
weighted avg	0.841631	0.847333	0.843329	3000.000000

Figures 23 - 26: Classification reports for the last 4 experiments

The training time and testing time reports is present in Figure 27. We can see that as we increase the number of trees in the random forests, the training time also increases. But overall, the training time is quick. I will show in later sections on how random forests are relatively faster when we compare with other models like SVM.

As accuracy remains almost same on increasing the number of trees from 25 to 100, it is a waste of resources and time to use 100 trees.


Training and Testing Times:

	Model	Training Time (s)	Testing Time (s)
0	Decision Tree (CART)	0.075340	0.000700
1	Decision Tree (Entropy)	0.088903	0.000753
2	Random Forest (5 trees)	0.079247	0.002894
3	Random Forest (10 trees)	0.161351	0.005198
4	Random Forest (25 trees)	0.393364	0.012380
5	Random Forest (50 trees)	0.771246	0.024158
6	Random Forest (75 trees)	1.162814	0.036546
7	Random Forest (100 trees)	1.553201	0.049166

Figure 27: Training and Testing time for all 8 experiments

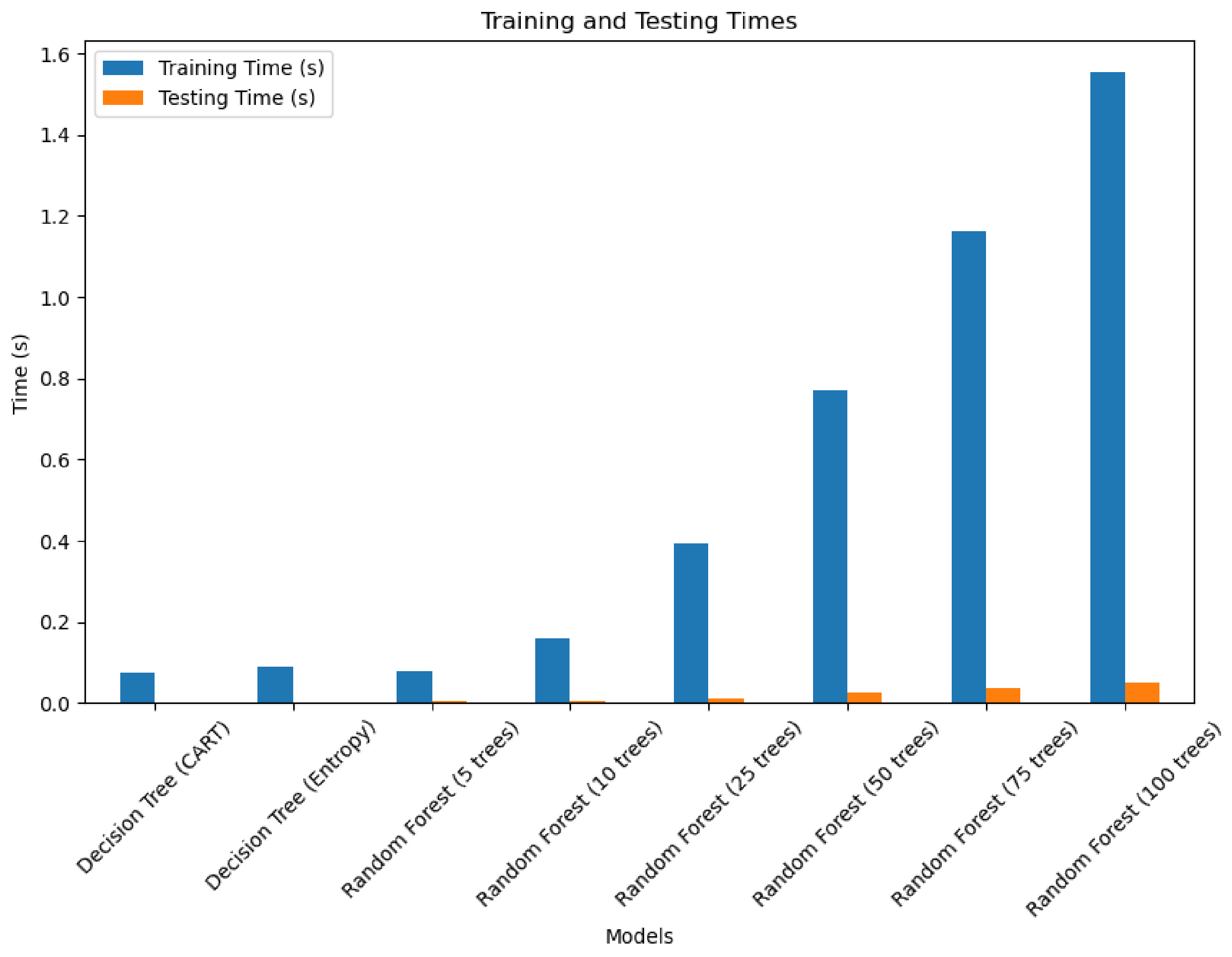


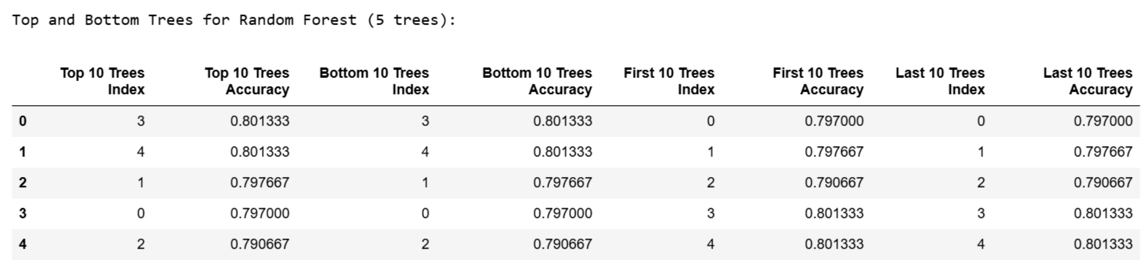
Figure 28: Training and Testing time bar graph

4.4 Analysis of Individual Trees in Random Forests

I have chosen 4 classes of trees within a single random forest.

- Top 10 Trees (based on accruacy): The trees with the highest accuracy
- Bottom 10 Trees (based on accuracy): The trees with the least accuracy
- First 10 Trees (based on index): The trees with indices at the start
- Last 10 Trees (based on index): The trees with the indices at the end

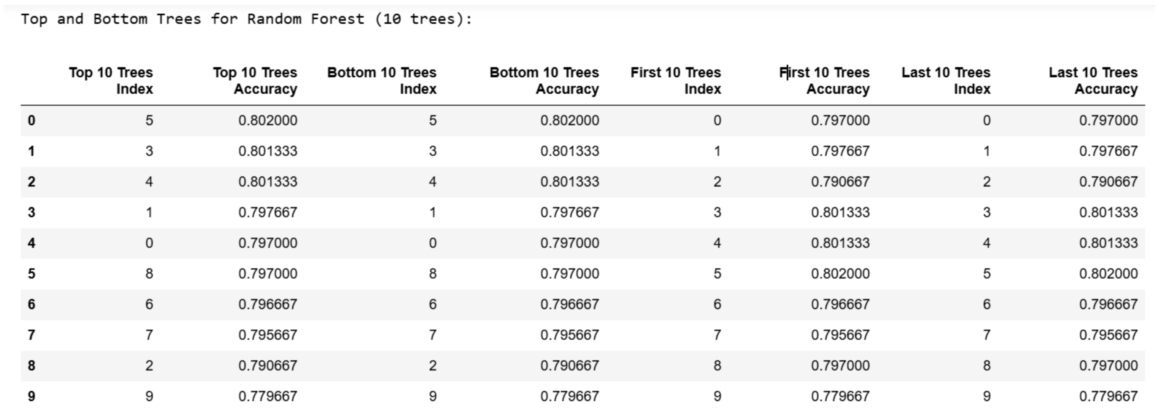
Figures 29 - 34 show the accuracies for each random forest experiment that I conducted.



```
Top and Bottom Trees for Random Forest (5 trees):
```

	Top 10 Trees Index	Top 10 Trees Accuracy	Bottom 10 Trees Index	Bottom 10 Trees Accuracy	First 10 Trees Index	First 10 Trees Accuracy	Last 10 Trees Index	Last 10 Trees Accuracy
0	3	0.801333	3	0.801333	0	0.797000	0	0.797000
1	4	0.801333	4	0.801333	1	0.797667	1	0.797667
2	1	0.797667	1	0.797667	2	0.790667	2	0.790667
3	0	0.797000	0	0.797000	3	0.801333	3	0.801333
4	2	0.790667	2	0.790667	4	0.801333	4	0.801333

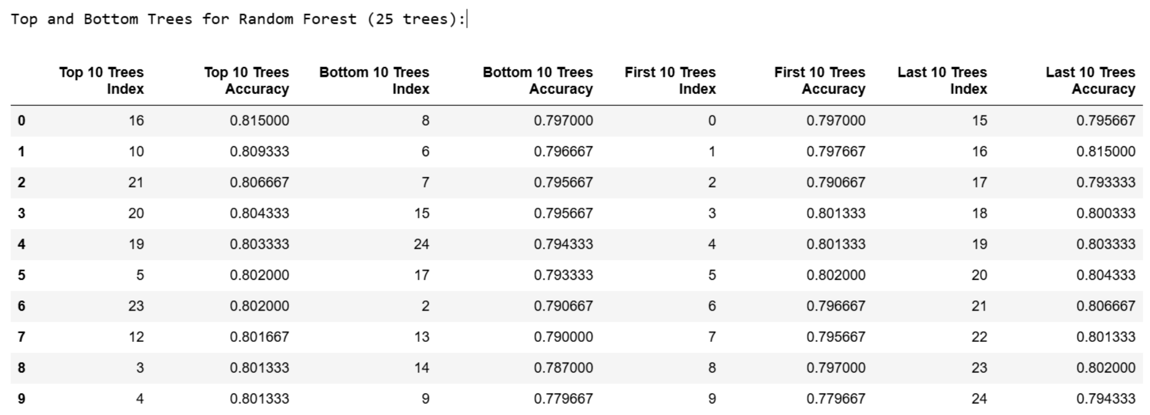
Figure 29: Accuracy for Random Forests (5 trees)



```
Top and Bottom Trees for Random Forest (10 trees):
```

	Top 10 Trees Index	Top 10 Trees Accuracy	Bottom 10 Trees Index	Bottom 10 Trees Accuracy	First 10 Trees Index	First 10 Trees Accuracy	Last 10 Trees Index	Last 10 Trees Accuracy
0	5	0.802000	5	0.802000	0	0.797000	0	0.797000
1	3	0.801333	3	0.801333	1	0.797667	1	0.797667
2	4	0.801333	4	0.801333	2	0.790667	2	0.790667
3	1	0.797667	1	0.797667	3	0.801333	3	0.801333
4	0	0.797000	0	0.797000	4	0.801333	4	0.801333
5	8	0.797000	8	0.797000	5	0.802000	5	0.802000
6	6	0.796667	6	0.796667	6	0.796667	6	0.796667
7	7	0.795667	7	0.795667	7	0.795667	7	0.795667
8	2	0.790667	2	0.790667	8	0.797000	8	0.797000
9	9	0.779667	9	0.779667	9	0.779667	9	0.779667

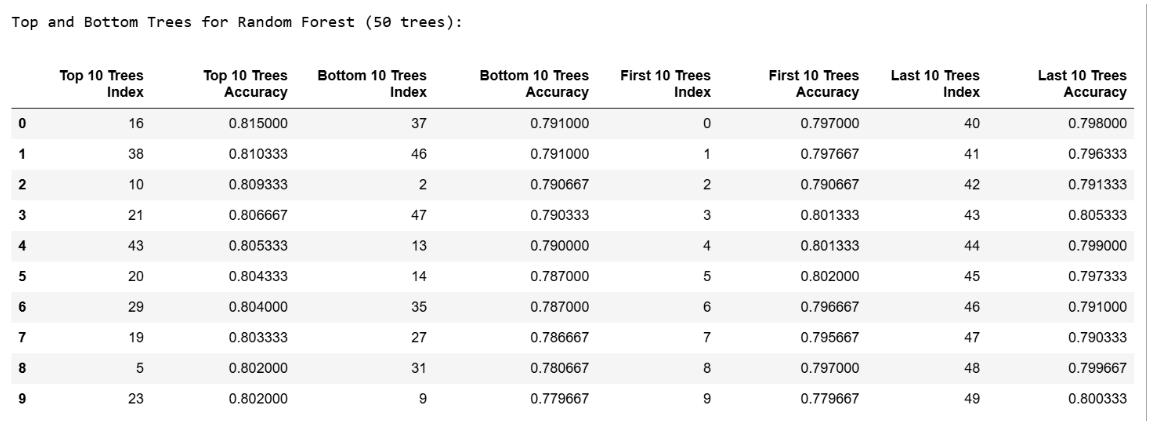
Figure 30: Accuracy for Random Forests (10 trees)



```
Top and Bottom Trees for Random Forest (25 trees):
```

	Top 10 Trees Index	Top 10 Trees Accuracy	Bottom 10 Trees Index	Bottom 10 Trees Accuracy	First 10 Trees Index	First 10 Trees Accuracy	Last 10 Trees Index	Last 10 Trees Accuracy
0	16	0.815000	8	0.797000	0	0.797000	15	0.795667
1	10	0.809333	6	0.796667	1	0.797667	16	0.815000
2	21	0.806667	7	0.795667	2	0.790667	17	0.793333
3	20	0.804333	15	0.795667	3	0.801333	18	0.800333
4	19	0.803333	24	0.794333	4	0.801333	19	0.803333
5	5	0.802000	17	0.793333	5	0.802000	20	0.804333
6	23	0.802000	2	0.790667	6	0.796667	21	0.806667
7	12	0.801667	13	0.790000	7	0.795667	22	0.801333
8	3	0.801333	14	0.787000	8	0.797000	23	0.802000
9	4	0.801333	9	0.779667	9	0.779667	24	0.794333

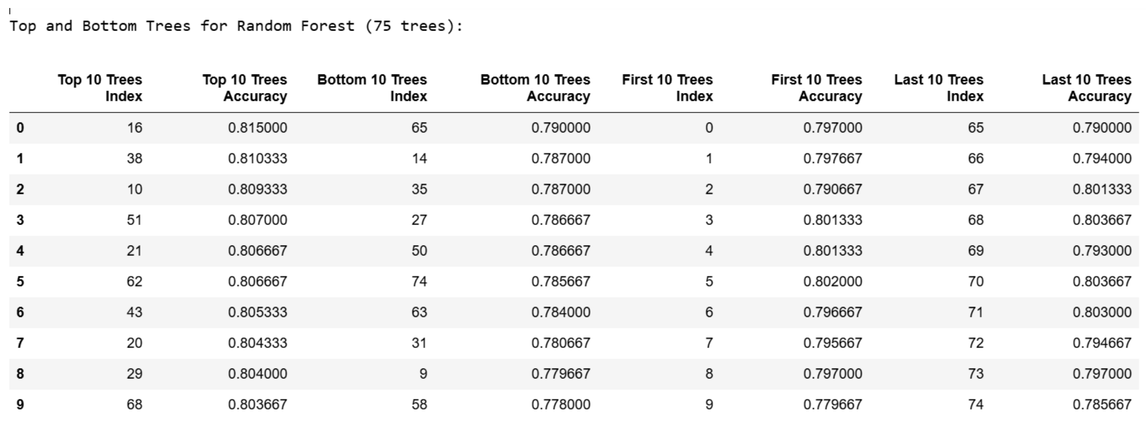
Figure 31: Accuracy for Random Forests (25 trees)



```
Top and Bottom Trees for Random Forest (50 trees):
```

	Top 10 Trees Index	Top 10 Trees Accuracy	Bottom 10 Trees Index	Bottom 10 Trees Accuracy	First 10 Trees Index	First 10 Trees Accuracy	Last 10 Trees Index	Last 10 Trees Accuracy
0	16	0.815000	37	0.791000	0	0.797000	40	0.798000
1	38	0.810333	46	0.791000	1	0.797667	41	0.796333
2	10	0.809333	2	0.790667	2	0.790667	42	0.791333
3	21	0.806667	47	0.790333	3	0.801333	43	0.805333
4	43	0.805333	13	0.790000	4	0.801333	44	0.799000
5	20	0.804333	14	0.787000	5	0.802000	45	0.797333
6	29	0.804000	35	0.787000	6	0.796667	46	0.791000
7	19	0.803333	27	0.786667	7	0.795667	47	0.790333
8	5	0.802000	31	0.780667	8	0.797000	48	0.799667
9	23	0.802000	9	0.779667	9	0.779667	49	0.800333

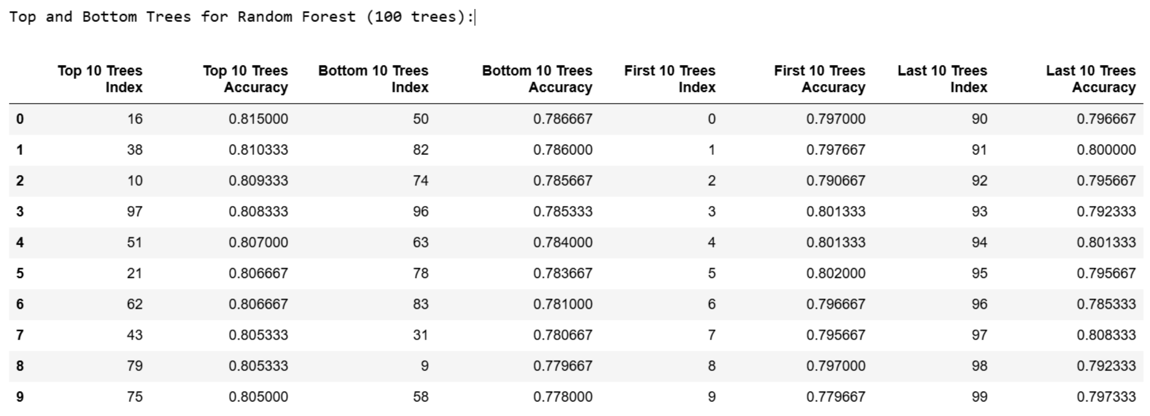
Figure 32: Accuracy for Random Forests (50 trees)



```
Top and Bottom Trees for Random Forest (75 trees):
```

	Top 10 Trees Index	Top 10 Trees Accuracy	Bottom 10 Trees Index	Bottom 10 Trees Accuracy	First 10 Trees Index	First 10 Trees Accuracy	Last 10 Trees Index	Last 10 Trees Accuracy
0	16	0.815000	65	0.790000	0	0.797000	65	0.790000
1	38	0.810333	14	0.787000	1	0.797667	66	0.794000
2	10	0.809333	35	0.787000	2	0.790667	67	0.801333
3	51	0.807000	27	0.786667	3	0.801333	68	0.803667
4	21	0.806667	50	0.786667	4	0.801333	69	0.793000
5	62	0.806667	74	0.785667	5	0.802000	70	0.803667
6	43	0.805333	63	0.784000	6	0.796667	71	0.803000
7	20	0.804333	31	0.780667	7	0.795667	72	0.794667
8	29	0.804000	9	0.779667	8	0.797000	73	0.797000
9	68	0.803667	58	0.778000	9	0.779667	74	0.785667

Figure 33: Accuracy for Random Forests (75 trees)



```
Top and Bottom Trees for Random Forest (100 trees):
```

	Top 10 Trees Index	Top 10 Trees Accuracy	Bottom 10 Trees Index	Bottom 10 Trees Accuracy	First 10 Trees Index	First 10 Trees Accuracy	Last 10 Trees Index	Last 10 Trees Accuracy
0	16	0.815000	50	0.786667	0	0.797000	90	0.796667
1	38	0.810333	82	0.786000	1	0.797667	91	0.800000
2	10	0.809333	74	0.785667	2	0.790667	92	0.795667
3	97	0.808333	96	0.785333	3	0.801333	93	0.792333
4	51	0.807000	63	0.784000	4	0.801333	94	0.801333
5	21	0.806667	78	0.783667	5	0.802000	95	0.795667
6	62	0.806667	83	0.781000	6	0.796667	96	0.785333
7	43	0.805333	31	0.780667	7	0.795667	97	0.808333
8	79	0.805333	9	0.779667	8	0.797000	98	0.792333
9	75	0.805000	58	0.778000	9	0.779667	99	0.797333

Figure 34: Accuracy for Random Forests (100 trees)

Analysis

We can see that there are not much differences between accuracies. Let us choose 25 trees example (Figure 31). The highest accuracy is of Tree 16 with 0.815 and lowest accuracy is Tree 9 with

0.779667. The difference is not much. However when we ensemble the tree while making a decision, it leads to increased accuracy; we get accuracy of about 0.8475 (Figure 15). This is the benefit of random forests which uses the ensemble of many trees to improve accuracy and robustness of decision making process. Inaccuracies in some trees while be outvoted (assuming we choose majority voting) by many accurate predictions.

(Note : Sometimes, this observation may change for a different dataset. There can be considerable differences in accuracies between trees. These differences in accuracies between trees are the reason on the basis of which tree pruning can be done for performance and resource constraint issues.)

4.5 Feature Importance

Decision Trees and Random Forests can help us to analyze the importance of features, i.e, which feature has most important contribution in deciding the predicted class. Sklearn has allows us to determine feature importance using model.feature_importances_ attribute.

I have plotted the feature importance plots in Figures 35-42.

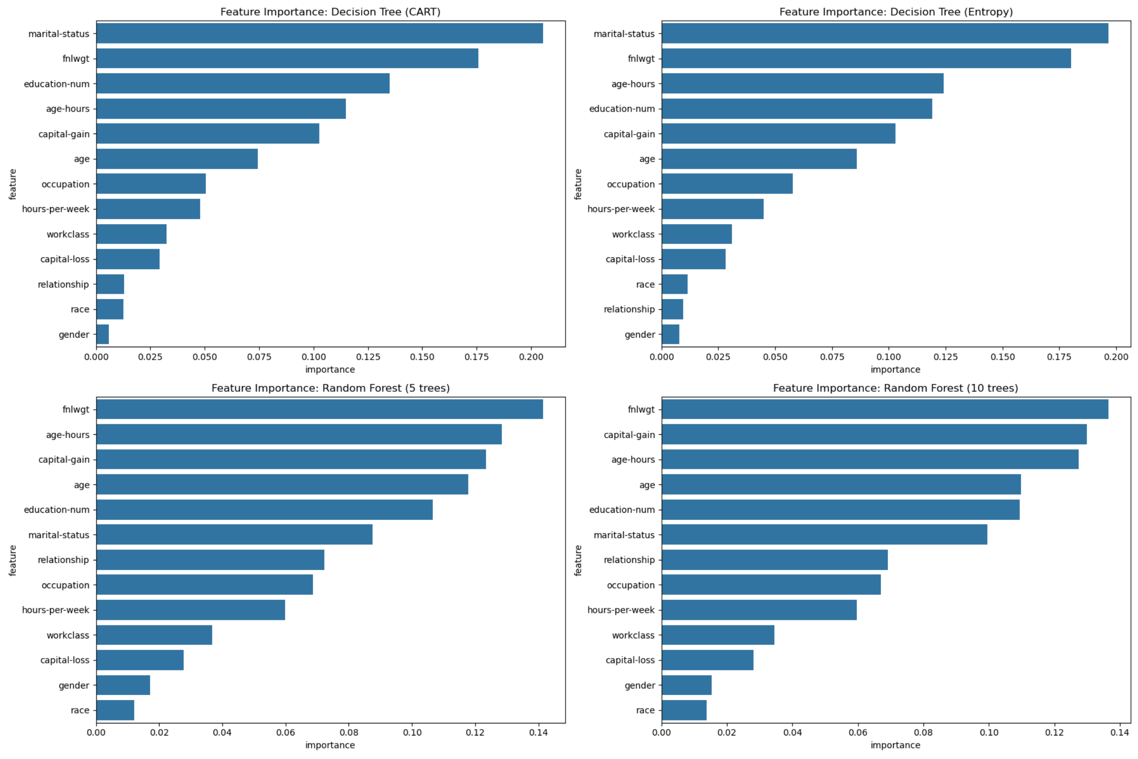


Figure 35-38: Feature Importance comparison for first 4 experiments

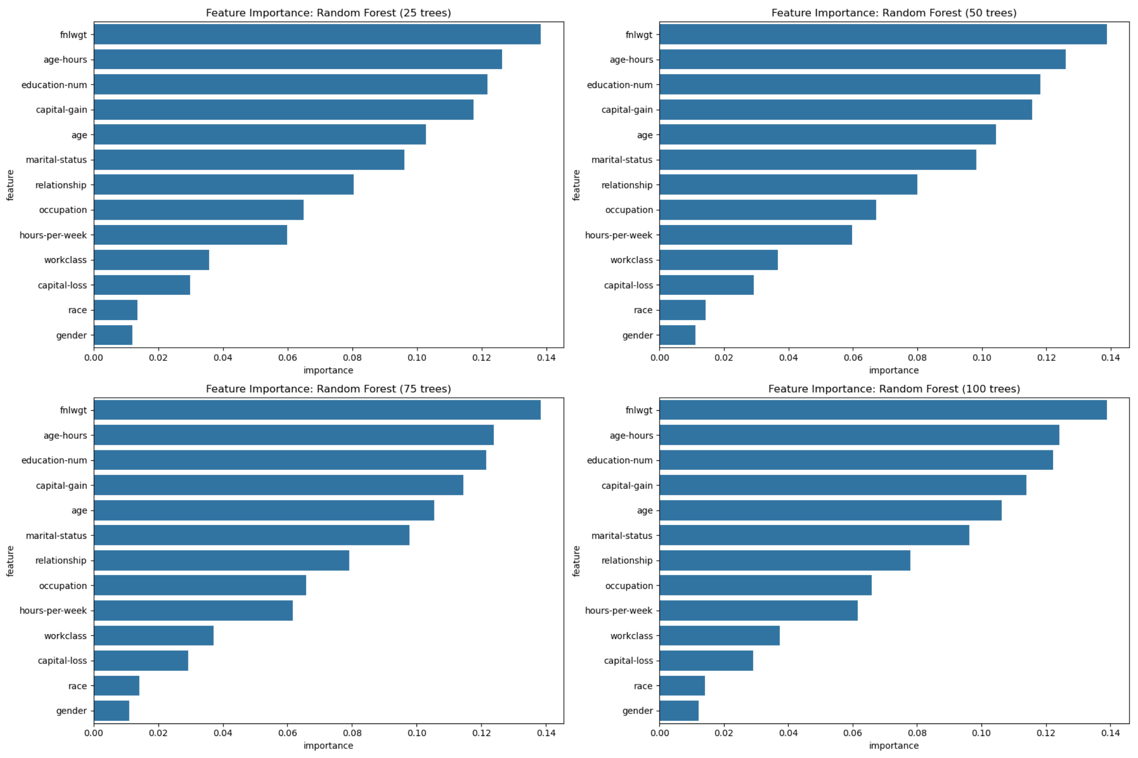


Figure 39-42: Feature Importance comparison for last 4 experiments

Analysis

- Decision Trees provide more importance to marital-status than fnlwgt whereas Random forests give more importance to fnlwgt. This could be because random forests behave like an ensemble and use more information from various trees to come to a better decision.

- When we compare Random Forests with 5 trees vs 10 trees, we see that 5 tree forest gives second importance to age-hours whereas 10 trees gives second importance to capital-gain. However after increasing to 25 trees, we see that age-hours again is the second most important feature. Therefore, sometimes, intermediate number of trees may not be desirable.

- We can see that the feature importances are almost same for 25-100 trees but slight differences in the values (Figures 39-42). This shows us that increasing number of trees blindly beyond a threshold will not yield increasing results.

4.6 Visualizing the Trees

We can use the graphviz package to print the decision trees. As there are a lot of features in this experiment, the trees are very deep and dense. Hence, I restricted the tree depth to 2 to visualize. In the code, it is an easily configurable parameter which stores the tree images in a directory named ’output’. Figures 43-45 show the tree images (depth 2) for Decision Tree (CART), Decision Tree (Entropy) and Random Forest with 25 trees.

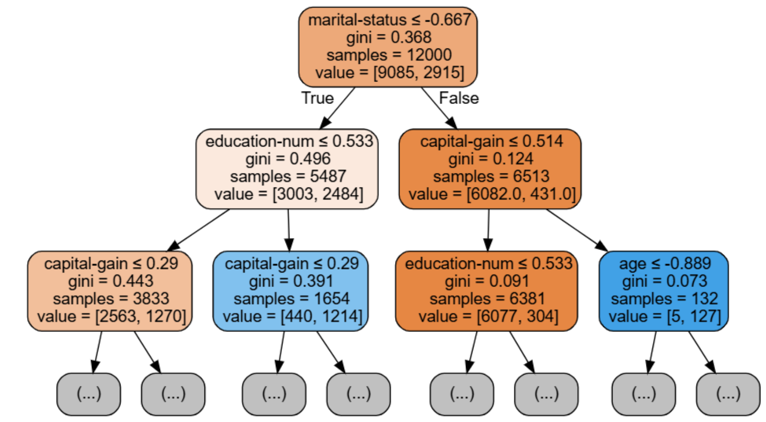


Figure 43: Decision Tree (CART)

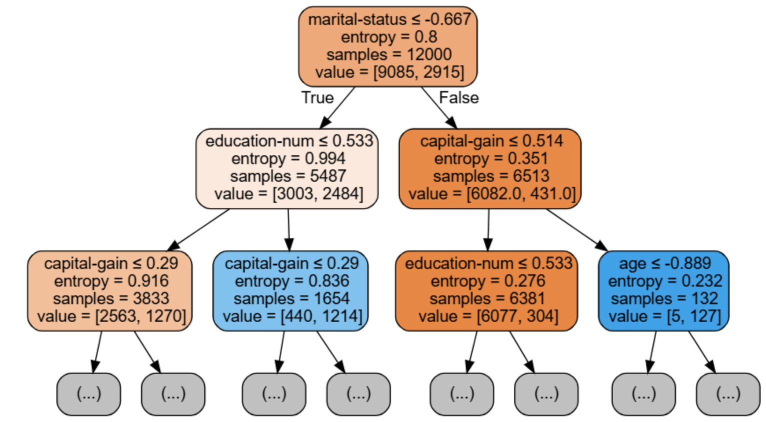


Figure 44: Decision Tree (Entropy)

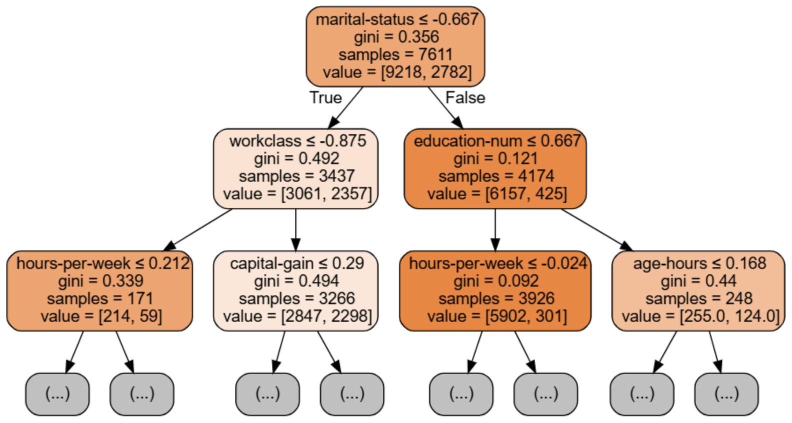


Figure 45: Random Forest (25 trees)

Analysis

Even though fnlwgt is most important feature in random forest with 25 trees (Figure 39), we can see that it is not present in the root of the random forest. In fact, maritial-status (most important factor in decision trees) is present at the root. This shows us that it is not directly possible to interpret the importance from the visualization and we need specific methods and specific functions to determine the importance of each feature (Figures 35-42).

Note: for depth = 2, all random forests will have the same tree image. The differences will be visible only at higher depth levels. Hence, I have not visualized them here.

5 Comparison of Random Forests with Other ML Models

Note: This section is not needed by the assignment, but I did this anyway to compare and see how random forests fare with other ML models. Figures 46-50 compare the various models. I compared the following algorithms on the same dataset.

- Naive Bayes
- Linear SVM
- Radial SVM
- Logistic Regression
- Decision Tree
- KNN
- Random Forest (25 trees)

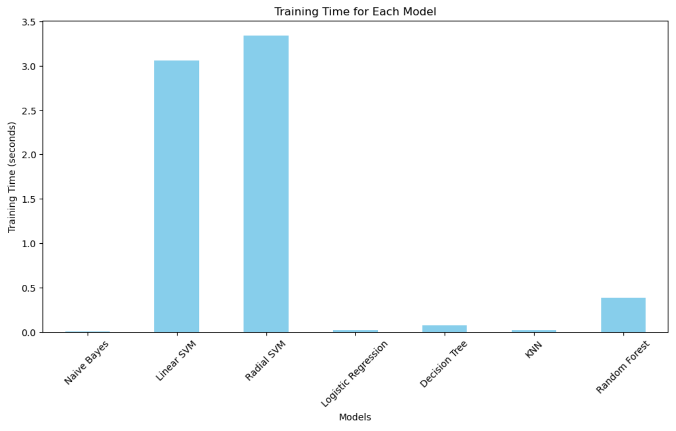


Figure 46: Training Time for Each Model

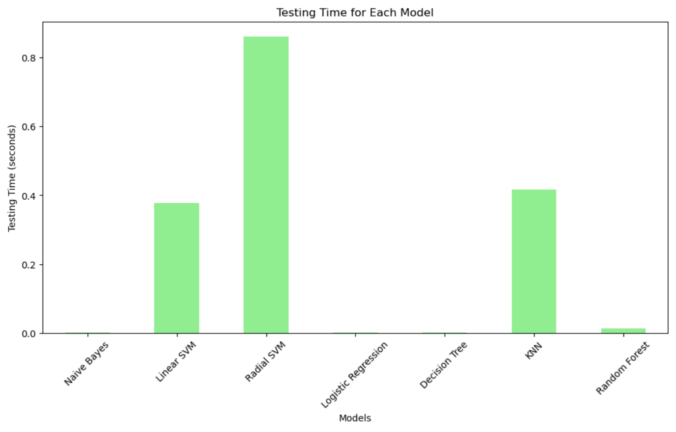


Figure 47: Testing Time for Each Model

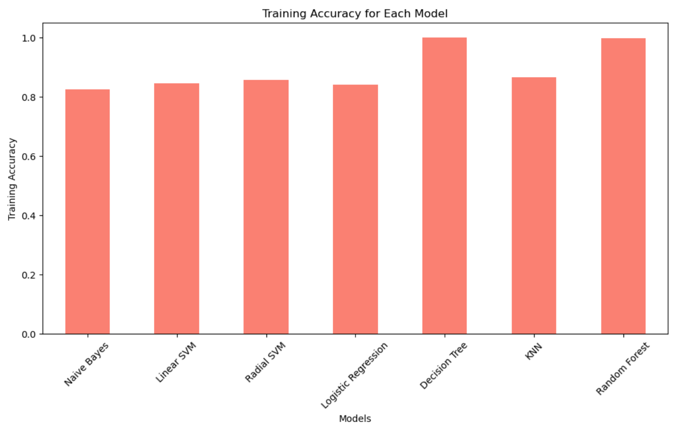


Figure 48: Training Accuracy for Each Model

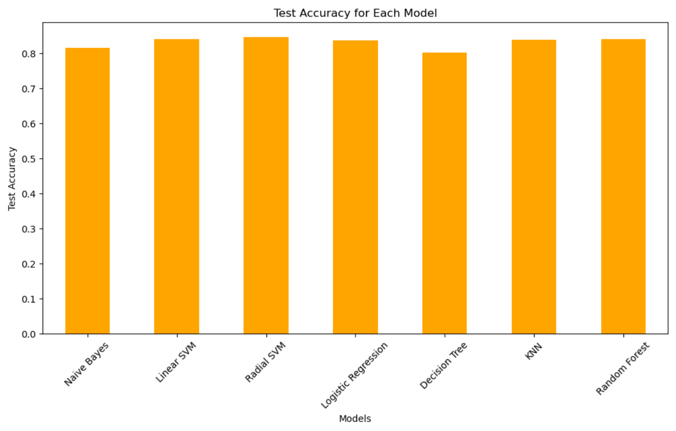


Figure 49: Testing Accuracy for Each Model

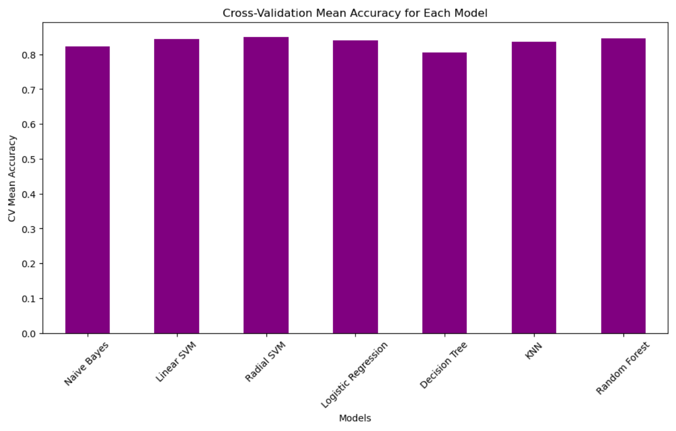


Figure 50: Cross Validation Mean Accuracy for Each Model

Analysis

- We can see that random forests and decision trees have highest training accuracies. Random forests also has highest test accuracy which proves its effectiveness for this dataset.
- We can see that random forests and decision trees have high training accuracy but not much **improvement** in test accuracy when we compare with other models. In fact, decision trees have the worst test accuracy of all. This shows us that decision trees and random forests slightly overfit the data and work too well for this dataset. We also see that due to the ensemble nature of random forests, test accuracy is mitigated (not considerable improvement when we compare to training accuracy, but still the highest among all models).
- The training and testing times for random forest is too low when we compare with similar high performance models like SVM or KNN. This is an advantage of random forests. We can use this in resource constrained environments.

6 Conclusion

- I was successfully able to implement the decision trees and random forests and also compared them with other models. Random forests outperformed all other standard ML models like SVM, Naive Bayes and Logistic Regression.
- Efficient data-preprocessing reduces training time and improves overall accuracy. Hence, during data preprocessing and exploration, I used feature engineering to create new features, then I encoded them and scaled them to gain the best performance.
- Entropy based decision trees outperform Information gain based decision tree (Note : This is only for this dataset. For some other dataset, information gain may be better).
- I observed that random forests will outperform single decision tree due to a variety of factors [10].
  - Random forests is a form of ensemble learning that combines multiple decision trees to make a prediction. This will help mitigate the limitations of individual decision trees and make stable predictions. This will also reduce impact of bias and variance.
  - Feature Importance (Section 4.5): I observed from my experiments that random forests and decision trees have different values for the most important feature. This is because of the nature of random forests and how it creates multiple trees which lead to the model gaining a better understanding of the most influential features.
  - Training Times: Random forests are faster than decision trees in terms of number of nodes processed / time. This is due to the parallelized nature of multiple trees. We can see from Figure 28 that 5 tree random forest is faster than entropy based single decision tree. Of course, the 25 tree random forest takes more time but that is because there is high density of nodes.
  - Robustness against overfitting: We can see from figures 48 and 49 as to how decision trees can tend to overfit and it is mitigated by random forests. This is due to the many voting and other ensemble mechanisms used in random forests which prevent noise and other outliers from affecting overall accuracy as opposed to a single decision tree.
  - Random forests provide a stable result. This is evidenced when we printed the top 10, bottom trees etc in section 4.4. We saw that all trees have almost similar accuracies.
- Visualizing the trees is difficult as trees can be dense and deep (section 4.6) and hence, we need alternate mechanisms to understand the feature importance (section 4.5).
- We observed that blindly increasing the number of trees beyond a threshold will not have improved results (Sections 4.3, 4.4 and 4.5). In my experiments, 25 trees random forest performed similarly to 100 trees random forest. The ideal threshold is usually slightly more than the number of features (In my experiment, I had 19 features before dropping unnecessary redundant features).
- In fact, increasing trees blindly may have detrimental effects when dataset is small. I re-conducted the experiment with limited data (2500 examples). Figure 51 shows the results. In fact, accuracy reduced when we increased trees from 10 –> 100.

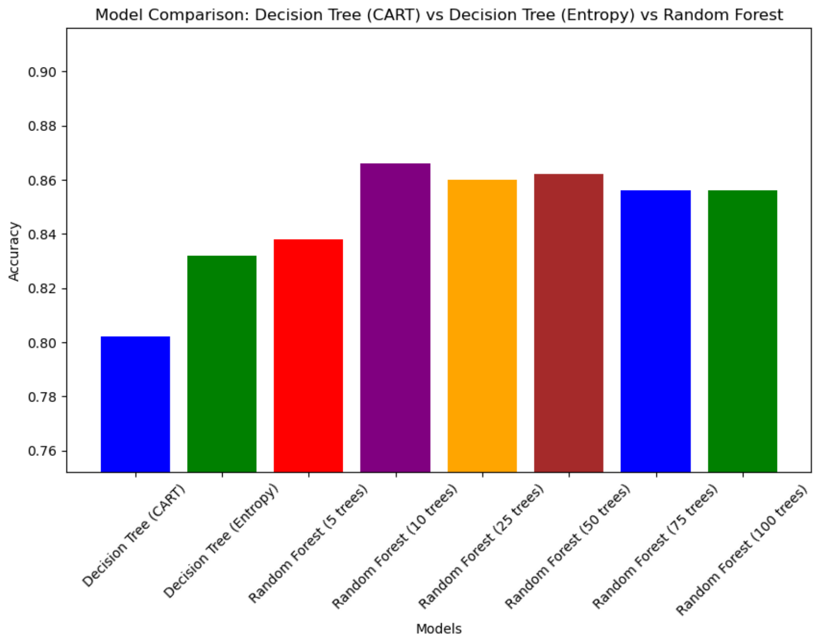
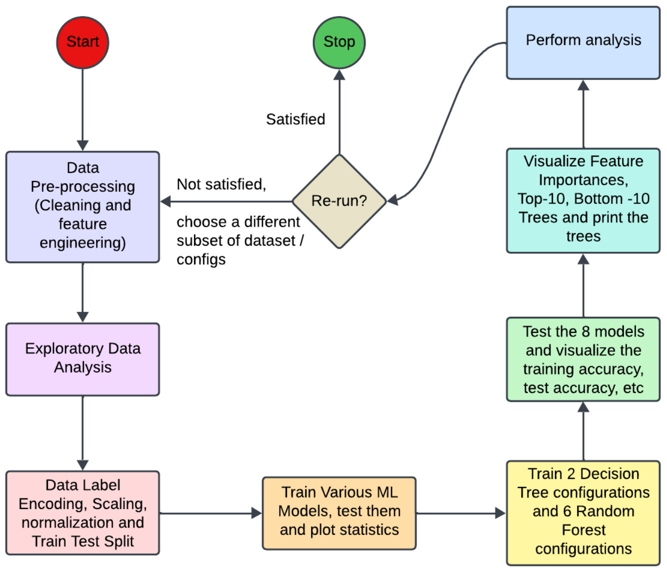


Figure 51: Testing accuracies for limited dataset

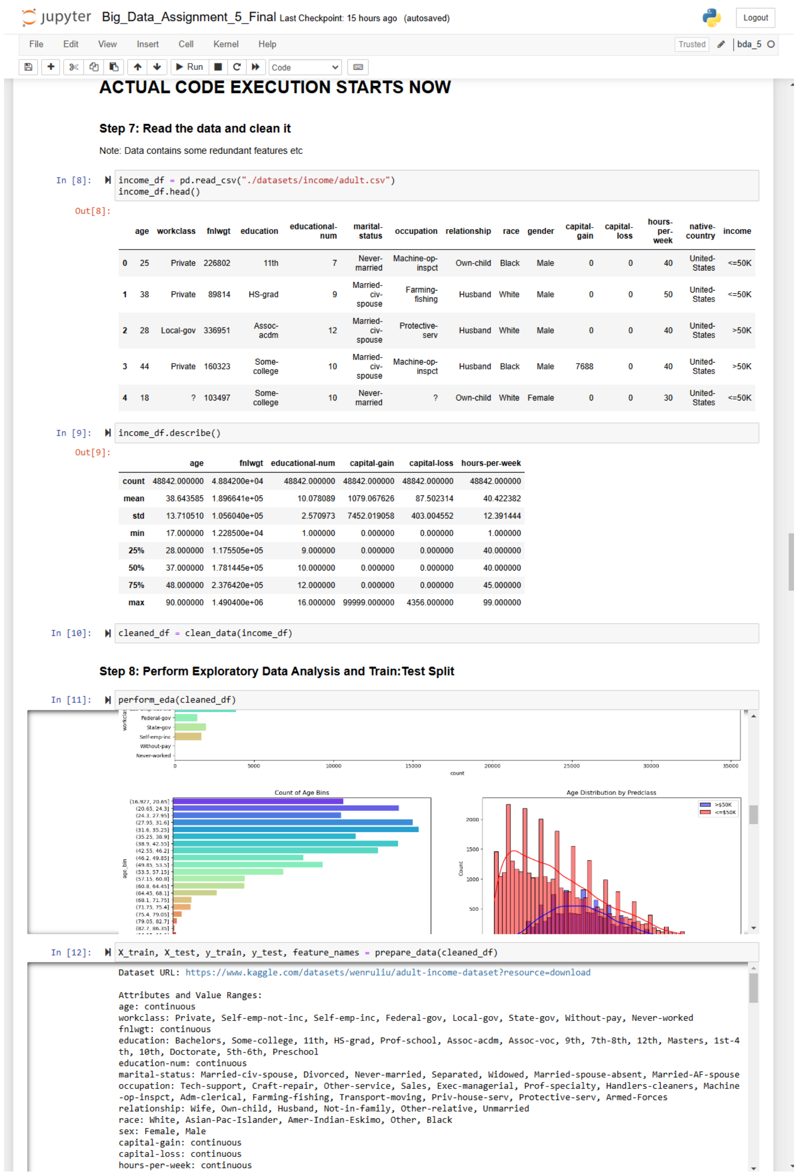
7 Workflow diagram



Start
Stop
Perform analysis
Satisfied
Data Pre-processing (Cleaning and feature engineering)
Not satisfied,
choose a different subset of dataset / configs
Re-run?
Visualize Feature Importances, Top-10, Bottom -10 Trees and print the trees
Exploratory Data Analysis
Test the 8 models and visualize the training accuracy, test accuracy, etc
Data Label Encoding, Scaling, normalization and Train Test Split
Train Various ML Models, test them and plot statistics
Train 2 Decision Tree configurations and 6 Random Forest configurations

Figure 52: Workflow diagram

8 Screenshots of Execution



ACTUAL CODE EXECUTION STARTS NOW

Step 7: Read the data and clean it

Note: Data contains some redundant features etc

```
In [8]:  income_df = pd.read_csv("./datasets/income/adult.csv")
         income_df.head()
```

Out[8]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male	0	0	40	United-States	<=50K
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male	0	0	50	United-States	<=50K
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male	0	0	40	United-States	>50K
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	7688	0	40	United-States	>50K
4	18	?	103497	Some-college	10	Never-married	?	Own-child	White	Female	0	0	30	United-States	<=50K

```
In [9]:  income_df.describe()
```

Out[9]:

	age	fnlwgt	educational-num	capital-gain	capital-loss	hours-per-week
count	48842.000000	4.884200e+04	48842.000000	48842.000000	48842.000000	48842.000000
mean	38.643585	1.896641e+05	10.078089	1079.067626	87.502314	40.422382
std	13.710510	1.056040e+05	2.570973	7452.019058	403.004552	12.391444
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.175505e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.781445e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.376420e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.490400e+06	16.000000	99999.000000	4356.000000	99.000000

```
In [10]:  cleaned_df = clean_data(income_df)
```

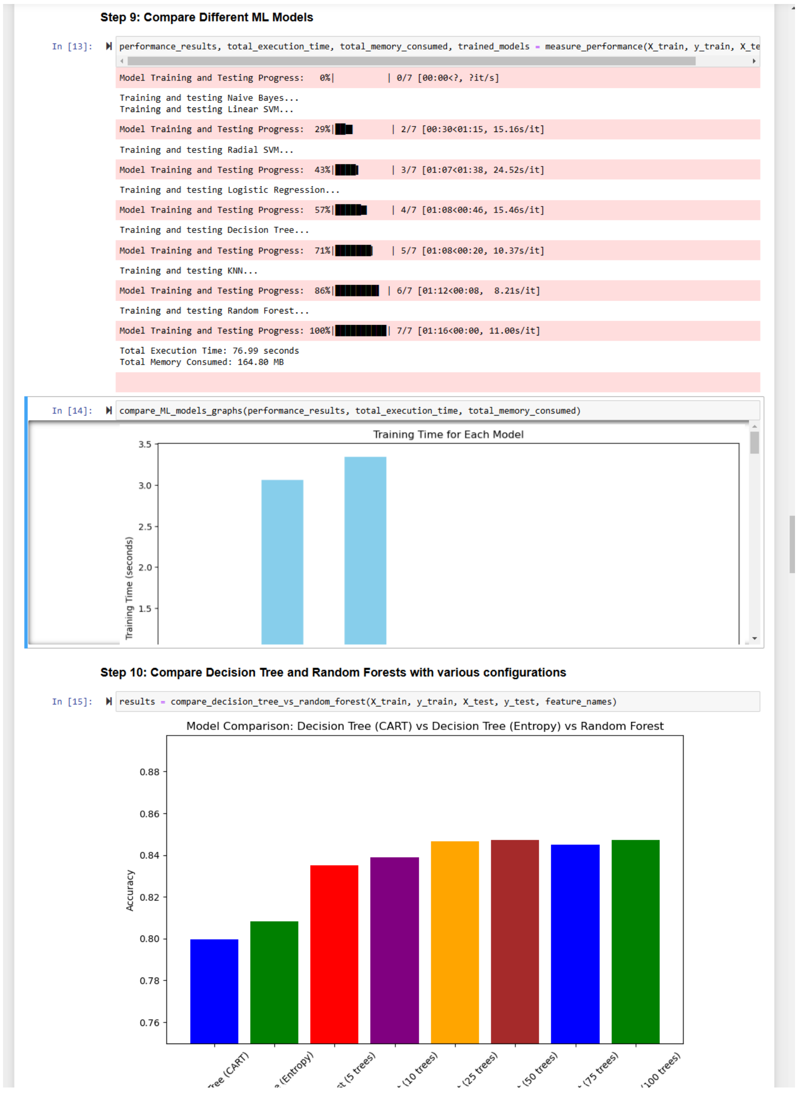
Step 8: Perform Exploratory Data Analysis and Train:Test Split

```
In [11]:  perform_eda(cleaned_df)
```

```
In [12]:  X_train, X_test, y_train, y_test, feature_names = prepare_data(cleaned_df)
```

```
Dataset URL: https://www.kaggle.com/datasets/wenruliu/adult-income-dataset?resource=download

Attributes and Value Ranges:
age: continuous
workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked
fnlwgt: continuous
education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4
th, 10th, Doctorate, 5th-6th, Preschool
education-num: continuous
marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse
occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine
-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces
relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried
race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black
sex: Female, Male
capital-gain: continuous
capital-loss: continuous
hours-per-week: continuous
```

Figure 53: Jupyter notebook execution - 1



Step 9: Compare Different ML Models

```
In [13]: performance_results, total_execution_time, total_memory_consumed, trained_models = measure_performance(X_train, y_train, X_te
```

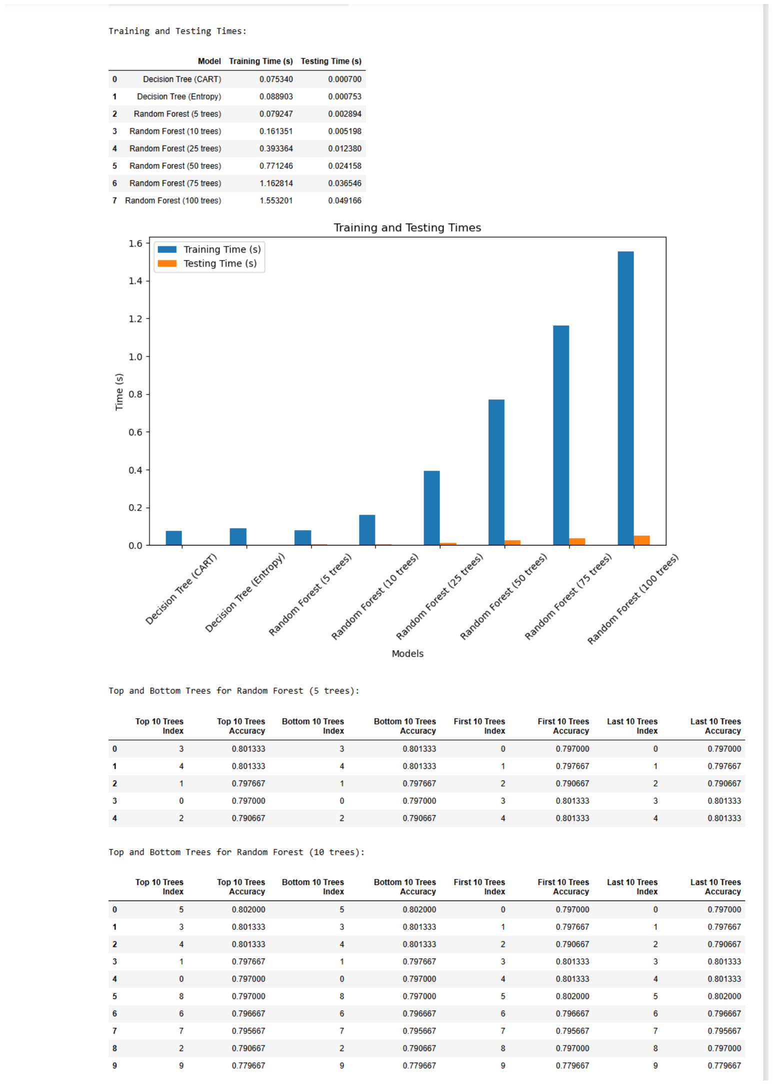
```
Model Training and Testing Progress:   0%|          | 0/7 [00:00<?, ?it/s]

Training and testing Naive Bayes...
Training and testing Linear SVM...

Model Training and Testing Progress:  29%|██▊       | 2/7 [00:30<01:15, 15.16s/it]

Training and testing Radial SVM...

Model Training and Testing Progress:  43%|████▎     | 3/7 [01:07<01:38, 24.52s/it]

Training and testing Logistic Regression...

Model Training and Testing Progress:  57%|█████▋    | 4/7 [01:08<00:46, 15.46s/it]

Training and testing Decision Tree...

Model Training and Testing Progress:  71%|███████▏  | 5/7 [01:08<00:20, 10.37s/it]

Training and testing KNN...

Model Training and Testing Progress:  86%|████████▌ | 6/7 [01:12<00:08,  8.21s/it]

Training and testing Random Forest...

Model Training and Testing Progress: 100%|██████████| 7/7 [01:16<00:00, 11.00s/it]

Total Execution Time: 76.99 seconds
Total Memory Consumed: 164.80 MB
```

```
In [14]: compare_ML_models_graphs(performance_results, total_execution_time, total_memory_consumed)
```

Step 10: Compare Decision Tree and Random Forests with various configurations

```
In [15]: results = compare_decision_tree_vs_random_forest(X_train, y_train, X_test, y_test, feature_names)
```

Figure 54: Jupyter notebook execution - 2



```
Training and Testing Times:
```

	Model	Training Time (s)	Testing Time (s)
0	Decision Tree (CART)	0.075340	0.000700
1	Decision Tree (Entropy)	0.088903	0.000753
2	Random Forest (5 trees)	0.079247	0.002894
3	Random Forest (10 trees)	0.161351	0.005198
4	Random Forest (25 trees)	0.393364	0.012380
5	Random Forest (50 trees)	0.771246	0.024158
6	Random Forest (75 trees)	1.162814	0.036546
7	Random Forest (100 trees)	1.553201	0.049166

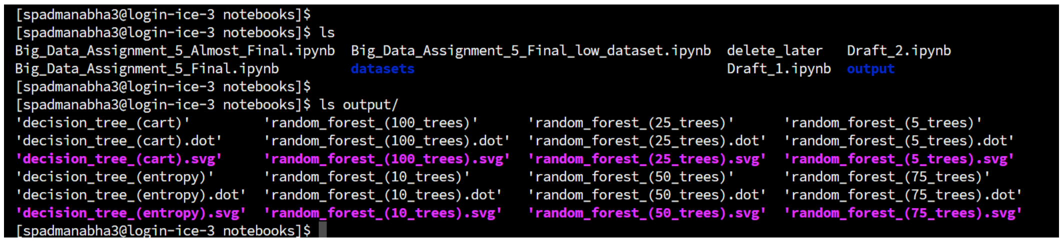
```
Top and Bottom Trees for Random Forest (5 trees):
```

	Top 10 Trees Index	Top 10 Trees Accuracy	Bottom 10 Trees Index	Bottom 10 Trees Accuracy	First 10 Trees Index	First 10 Trees Accuracy	Last 10 Trees Index	Last 10 Trees Accuracy
0	3	0.801333	3	0.801333	0	0.797000	0	0.797000
1	4	0.801333	4	0.801333	1	0.797667	1	0.797667
2	1	0.797667	1	0.797667	2	0.790667	2	0.790667
3	0	0.797000	0	0.797000	3	0.801333	3	0.801333
4	2	0.790667	2	0.790667	4	0.801333	4	0.801333

```
Top and Bottom Trees for Random Forest (10 trees):
```

	Top 10 Trees Index	Top 10 Trees Accuracy	Bottom 10 Trees Index	Bottom 10 Trees Accuracy	First 10 Trees Index	First 10 Trees Accuracy	Last 10 Trees Index	Last 10 Trees Accuracy
0	5	0.802000	5	0.802000	0	0.797000	0	0.797000
1	3	0.801333	3	0.801333	1	0.797667	1	0.797667
2	4	0.801333	4	0.801333	2	0.790667	2	0.790667
3	1	0.797667	1	0.797667	3	0.801333	3	0.801333
4	0	0.797000	0	0.797000	4	0.801333	4	0.801333
5	8	0.797000	8	0.797000	5	0.802000	5	0.802000
6	6	0.796667	6	0.796667	6	0.796667	6	0.796667
7	7	0.795667	7	0.795667	7	0.795667	7	0.795667
8	2	0.790667	2	0.790667	8	0.797000	8	0.797000
9	9	0.779667	9	0.779667	9	0.779667	9	0.779667

Figure 55: Jupyter notebook execution - 3



```
[spadmanabha3@login-ice-3 notebooks]$
[spadmanabha3@login-ice-3 notebooks]$ ls
Big_Data_Assignment_5_Almost_Final.ipynb  Big_Data_Assignment_5_Final_low_dataset.ipynb  delete_later   Draft_2.ipynb
Big_Data_Assignment_5_Final.ipynb         datasets                                       Draft_1.ipynb  output
[spadmanabha3@login-ice-3 notebooks]$
[spadmanabha3@login-ice-3 notebooks]$ ls output/
'decision_tree_(cart)'         'random_forest_(100_trees)'      'random_forest_(25_trees)'      'random_forest_(5_trees)'
'decision_tree_(cart).dot'     'random_forest_(100_trees).dot'  'random_forest_(25_trees).dot'  'random_forest_(5_trees).dot'
'decision_tree_(cart).svg'     'random_forest_(100_trees).svg'  'random_forest_(25_trees).svg'  'random_forest_(5_trees).svg'
'decision_tree_(entropy)'      'random_forest_(10_trees)'       'random_forest_(50_trees)'      'random_forest_(75_trees)'
'decision_tree_(entropy).dot'  'random_forest_(10_trees).dot'   'random_forest_(50_trees).dot'  'random_forest_(75_trees).dot'
'decision_tree_(entropy).svg'  'random_forest_(10_trees).svg'   'random_forest_(50_trees).svg'  'random_forest_(75_trees).svg'
[spadmanabha3@login-ice-3 notebooks]$
```

Figure 56: PACE environment directory with outputs

9 References

[1] https://www.kaggle.com/code/jieyima/income-classification-model

[2] https://github.com/WillKoehrsen/Machine-Learning-Projects/blob/master/Random%20Forest%20Tutorial.ipynb

[3] https://nthu-datalab.github.io/ml/labs/03_Decision-Tree_Random-Forest/03_Decision-Tree_Random-Forest.html

[4] https://www.restack.io/p/decision-making-models-answer-comparing-decision-trees-random-forests-cat-ai

[5] https://gist.github.com/pb111/88545fa33780928694388779af23bf58

[6] https://archive.ics.uci.edu/dataset/2/adult

[7] https://www.kaggle.com/datasets/wenruliu/adult-income-dataset?resource=download

[8] https://scikit-learn.org/stable/

[9] https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/

[10] https://pratikbarjatya.medium.com/unleashing-the-power-of-random-forest-why-it-outperforms-decision-trees-and-expert-rules-472a9bea1b8a