



Universiti Tunku Abdul Rahman
Faculty of Information and Communication Technology (FICT)

UCCD2063 Artificial Intelligence Techniques
Practical Assignment
May 2021

Group 20

No	Name	Student ID	Practical	Contribution
1.	Ong Kian Shon	18ACB02258	P6	33.33%
2.	Lim Jason	18ACB02103	P6	33.33%
3.	Lee Jia Sheng	19ACB02018	P6	33.33%
Total:				100%

Contents

1.0 Introduction	1
2.0 Methods to Process Data	2
2.1 Data Description	2
2.2 Data Exploration	4
2.2.1 Data Overview	4
2.2.2 Missing Data	5
2.2.3 Statistical Data Analysis	6
2.3 Data Visualization	7
2.4 Overall Framework and Implementation Details	15
2.4.1 Data Preprocessing	16
2.4.2 Model Training and Validation	24
2.4.3 Fine Tuning.....	28
2.4.4 Testing	29
3.0 Results and Discussion.....	32
3.1 Experimental Results	32
3.3 Comparison of the algorithms	41
4.0 Conclusion.....	43
References.....	44

1.0 Introduction

Income gap refers to the gap expressed by the difference between high and low income levels or the difference in the proportion of income. It is a concept corresponding to income equality. However, income inequality is currently a serious problem faced by a lot of countries. Income inequality can be defined as the difference in how income is being distributed among individuals or throughout the populations. When personal opportunities are ostensibly restricted, various factors may exacerbate the problem. This can come in the form of lack of access to education, preferential treatment, prejudice, etc. These problems can only be solved through the use of appropriate policies formulated by relevant parties such as the government and politicians.

Correctly formulate unbiased policies as the guiding principle for the government to formulate new and better policies. The United State Census Bureau must conduct a census at least once every ten years in order to better understand their people and economic census statistics through national surveys. Variety of qualitative and quantitative data that describes the populations are being collected within this period.

So, in order to process mass information in a fast and efficient manner, there are techniques found in the field of Artificial Intelligence (AI) that can be implemented by modeling the populations in a Binary Classification Problem. Census data collected by statisticians can be used to train various models that represent the current population. Once the model has been trained to simulate the current population characteristics, after that it can be used to predict which person's salary is in the high or low range.

The objective is to train a Binary Classification Model which can be used to determine based on the person's profile information (Eg: Occupation, Race, Marital and Relationship status) and other related indicators such as investment, number of hours worked, etc. to determine whether that person salary is in the high or low range.

2.0 Methods to Process Data

In this chapter will discuss about what content does the dataset have by following steps:

1. Data Describe
2. Data Explore
3. Data Visualize
4. Data Preprocessing
5. Methodology Used
6. Flowchart of Training Model
7. Ways to Implement Training Model

2.1 Data Description

Since the dataset is given, hence there are no data collection steps in this assignment. This dataset is about the salary information of individuals from several countries. Moreover, it also consists of various attributes and information of the person's profile information. Below are the summarized lists of the dataset attributes and its corresponding values.

Numerical Attributes:

No.	Attributes	Values
1	age	Int64 {17-90yeas old}
2	weight	Int64
3	investment_gain	Int64 {0.00-99999.00}
4	investment_loss	Int64 {0.00-4356.00}
5	company_size	Int64 {5-1000 employees}
6	weekly_hours	Int64 {1-99hours}

Categorical Attributes:

No.	Attributes	Values
1	work_type	Object { Private,Self-emp-not-inc, Local-gov, State-gov, Self-emp-inc, Federal-gov, Without-pay, }
2	education	Object { HS-grad,Some-college,Bachelors,Masters,Assoc-voc,11th,Assoc-acdm,10th,7th-8th,Prof-school,9th,12th,Doctorate,5th-6th,1st-4th,Preschool }
3	education_years	Object { 1-16 }
4	marital_status	Object { Married-civ-spouse, Never-married, Divorced, Separated, Widowed, Married-spouse-absent,Married-AF-spouse }
5	occupation	Object { Prof-specialty ,Craft-repair,Exec-managerial ,Adm-clerical,Sales,Other-service,Machine-op-inspct,Transport-moving ,Handlers-cleaners,Farming-fishing,Tech-support,Protective-serv ,Priv-house-serv ,Armed-Forces }
6	relationship	Object { Husband, Not-in-family ,Own-child, Unmarried, Wife, Other-relative }
7	race	Object { White, Black, Asian-Pac-Islander, Amer-Indian-Eskimo,Other }
8	sex	Object { Male, Female }
9	country	Object { 42 country }
10	salary_range	Object { high, low }

2.2 Data Exploration

2.2.1 Data Overview

```
salary.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 16 columns):
 #   Column              Non-Null Count  Dtype  
---  -
 0   work_type           30725 non-null  object  
 1   age                 32561 non-null  int64   
 2   weight              32561 non-null  int64   
 3   education            32561 non-null  object  
 4   education_years      32552 non-null  float64  
 5   marital_status      32561 non-null  object  
 6   occupation           30718 non-null  object  
 7   relationship         32561 non-null  object  
 8   race                 32561 non-null  object  
 9   sex                  32561 non-null  object  
10  investment_gain      32561 non-null  int64   
11  investment_loss      32561 non-null  int64   
12  company_size         32561 non-null  int64   
13  weekly_hours         32561 non-null  int64   
14  country              31978 non-null  object  
15  salary_range         32561 non-null  object  
dtypes: float64(1), int64(6), object(9)
memory usage: 4.0+ MB
```

In this dataset, it has 32561 samples record which consists of 16 attributes/features. For the attribute it has numerical and also categorical types of values. There are 6 numerical attributes which are age, weight, investment_gain, investment_loss, company_size, weekly_hours and 10 categorical attributes which are work_type, education, education_years, marital_status, occupation, relationship, race, sex, country, salary_range. For our target variable in this dataset which is the “salary_range” attribute, it is a string object type attribute which we can turn into binary classification (since it only has two outputs which are “high” and “low”).

2.2.2 Missing Data

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	work_type	age	weight	education	education_years	marital_status	occupation	relationship	race	sex	investment_gain	investment_loss	company_size	weekly_hours	country	salary_range
2	Private	19	216804	7th-8th	4.0	Never-married	Other-service	Own-child	White	Male	0	0	825	33	United-States	low
3	Private	23	207546	11th	7.0	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	0	0	865	40	United-States	low
4	Private	41	253759	HS-grad	9.0	Never-married	Sales	Unmarried	Black	Female	0	0	727	40	United-States	low
5	Private	28	180928	Some-college	10.0	Married-civ-spouse	Handlers-cleaners	Husband	White	Male	5013	0	328	55	United-States	low
6	Private	17	208463	HS-grad	9.0	Never-married	Sales	Own-child	White	Female	0	0	977	20	United-States	low
7	Private	73	333676	HS-grad	9.0	Married-civ-spouse	Farming-fishing	Husband	White	Male	0	0	118	50	United-States	low
8	Local-gov	65	24824	HS-grad	9.0	Married-civ-spouse	Craft-repair	Husband	White	Male	0	0	793	40	United-States	low
9	Private	37	96330	Some-college	10.0	Never-married	Executive-managerial	Unmarried	Black	Female	0	0	193	40	United-States	low
10	Private	53	47396	HS-grad	9.0	Married-civ-spouse	Craft-repair	Husband	White	Male	0	0	427	40	United-States	low
11	?	61	166855	Bachelors	13.0	Married-civ-spouse	?	Husband	White	Male	0	0	164	10	United-States	low

The above figure shows the original dataset using Microsoft excel. There are some columns having missing or invalid values, in the beginning the dataset contains the “?” symbol as the missing value and this may affect our prediction model. So in this case, we have to handle these invalid data inputs so that we can detect those missing values by replacing them to “NaN” while we read the data in the jupyter notebook. (as the figure shown below)

```
salary = pd.read_csv("Salary.csv", na_values = "?")
```

```
salary.head(10)
```

	work_type	age	weight	education	education_years	marital_status	occupation	relationship	race	sex	investment_gain	investment_loss	company_size	weekly_hours	country	salary_range
0	Private	19	216804	7th-8th	4.0	Never-married	Other-service	Own-child	White	Male	0	0	825	33	United-States	low
1	Private	23	207546	11th	7.0	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	0	0	865	40	United-States	low
2	Private	41	253759	HS-grad	9.0	Never-married	Sales	Unmarried	Black	Female	0	0	727	40	United-States	low
3	Private	28	180928	Some-college	10.0	Married-civ-spouse	Handlers-cleaners	Husband	White	Male	5013	0	328	55	United-States	low
4	Private	17	208463	HS-grad	9.0	Never-married	Sales	Own-child	White	Female	0	0	977	20	United-States	low
5	Private	73	333676	HS-grad	9.0	Married-civ-spouse	Farming-fishing	Husband	White	Male	0	0	118	50	United-States	low
6	Local-gov	65	24824	HS-grad	9.0	Married-civ-spouse	Craft-repair	Husband	White	Male	0	0	793	40	United-States	low
7	Private	37	96330	Some-college	10.0	Never-married	Executive-managerial	Unmarried	Black	Female	0	0	193	40	United-States	low
8	Private	53	47396	HS-grad	9.0	Married-civ-spouse	Craft-repair	Husband	White	Male	0	0	427	40	United-States	low
9	NaN	61	166855	Bachelors	13.0	Married-civ-spouse	NaN	Husband	White	Male	0	0	164	10	United-States	low

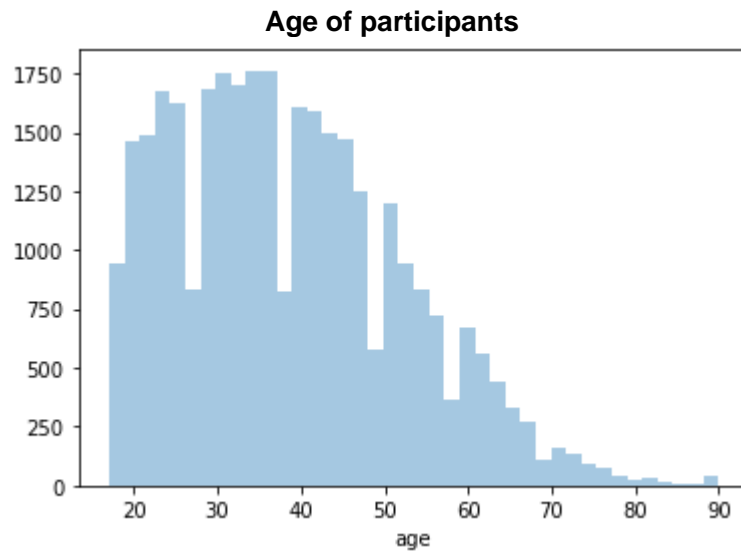
2.2.3 Statistical Data Analysis

	age	weight	education_years	investment_gain	investment_loss	company_size	weekly_hours
count	32561.000000	3.256100e+04	32552.000000	32561.000000	32561.000000	32561.000000	32561.000000
mean	38.581647	1.897784e+05	10.080609	1077.648844	87.303830	500.930838	40.437456
std	13.640433	1.055500e+05	2.572762	7385.292085	402.960219	285.784842	12.347429
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	5.000000	1.000000
25%	28.000000	1.178270e+05	9.000000	0.000000	0.000000	254.000000	40.000000
50%	37.000000	1.783560e+05	10.000000	0.000000	0.000000	502.000000	40.000000
75%	48.000000	2.370510e+05	12.000000	0.000000	0.000000	745.000000	45.000000
max	90.000000	1.484705e+06	16.000000	99999.000000	4356.000000	1000.000000	99.000000

The figure above shows that the data describe function results, from these statistical values we can know the maximum, minimum, standard division, mean, and also the total sample for each attribute.

Smallest mean value is the “weight” attribute which consists of only 1.8978, whereas the largest mean value belongs to the “investment_gain” attribute which has a 1077.6488. Moreover, for the smallest standard deviation it still belongs to the “weight” attribute which only has 1.0555. While for the largest standard deviation it belongs to the “investment_gain” attribute which has 7385.2920. The figure above also shows that there is varying scale in the dataset, so standardization must be performed.

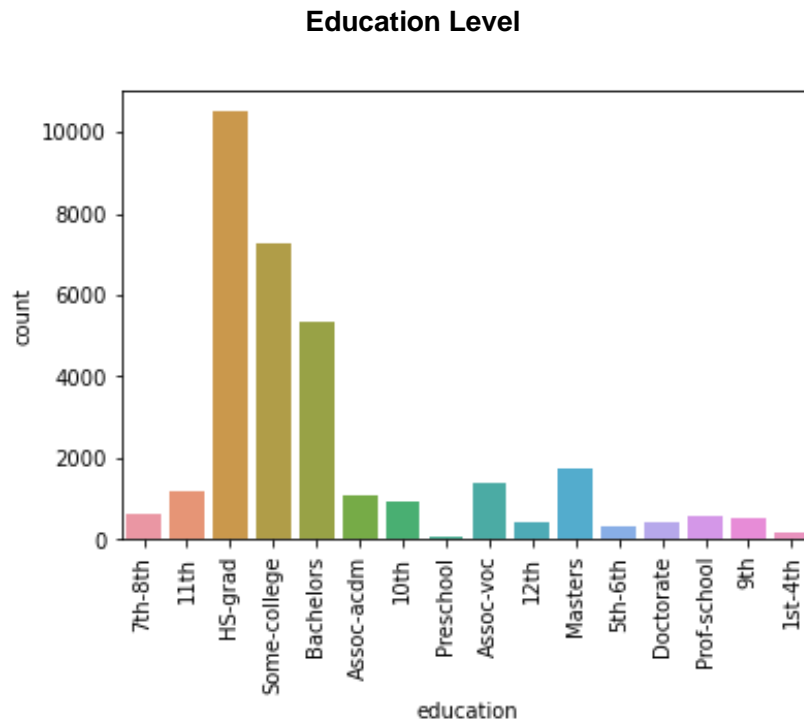
2.3 Data Visualization



There are the most number of individuals within the 28 - 36 age range.

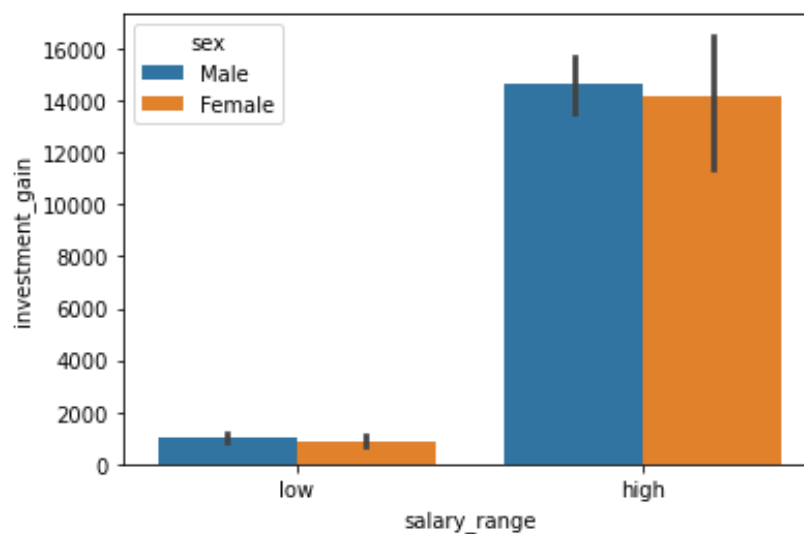


We can observe that there are way more people in the low salary range than the high salary range. There are approximately 25000 individuals in the low salary range, while there are only approximately 9000 individuals in the high salary range.



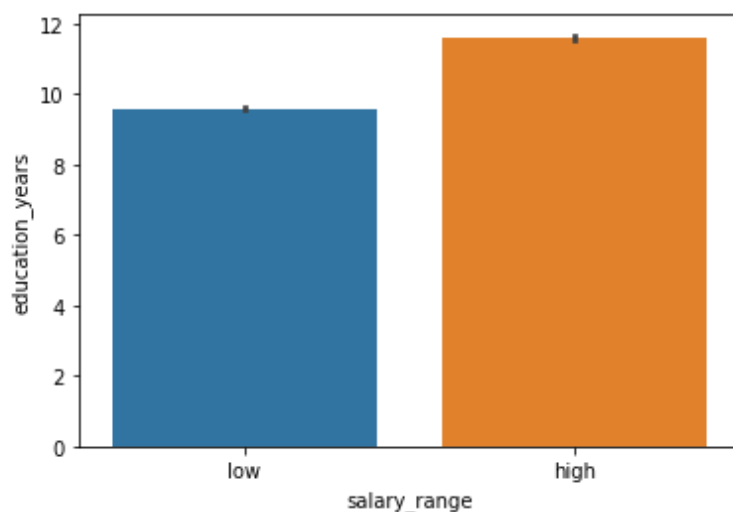
Most of the individuals are high school graduates followed by college. The least amount of people have the highest education of preschool.

Relationship between investment gain and salary range between male and females



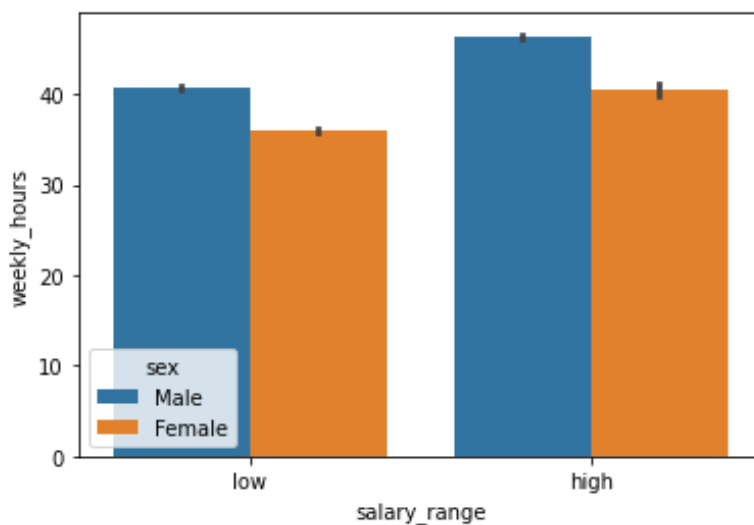
Both men and women that are in the high salary range have significantly higher investment gains.

Relationship between years of education and salary range



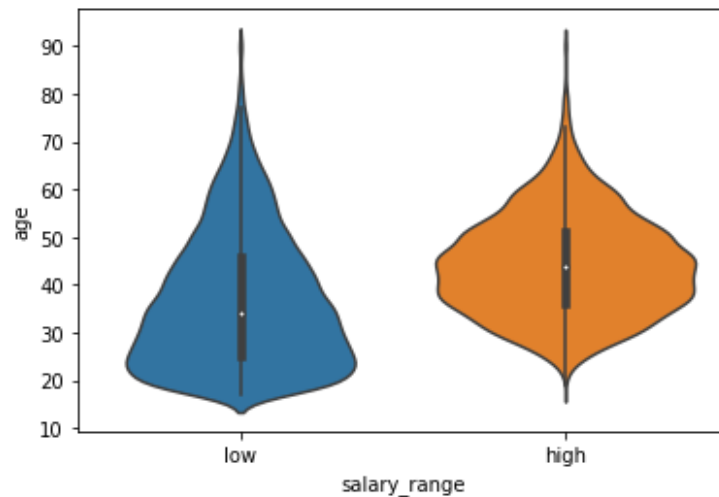
Individuals that have more years of education are more likely to be in the high salary range than individuals with less years of education.

Relationship between weekly hours worked and salary range



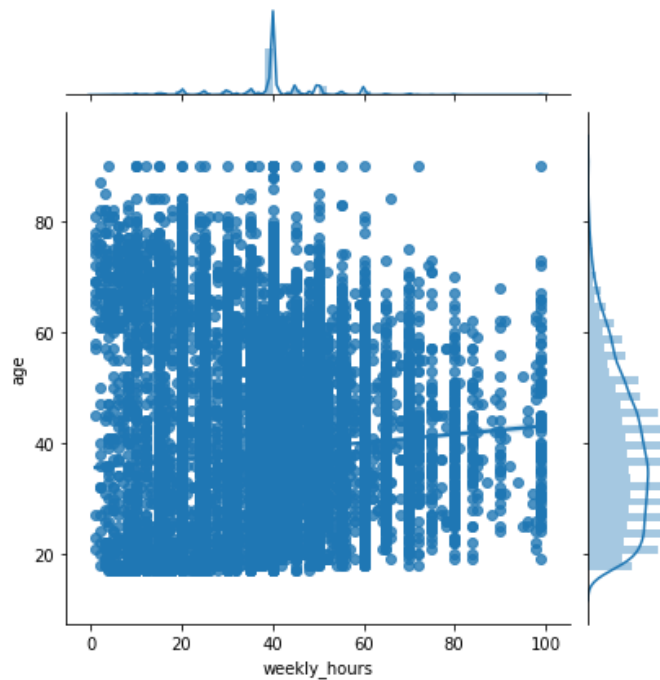
The people that are in the high salary range work slightly more weekly hours than those in the low salary range. Additionally, males tend to work more hours than females in both high and low salary ranges.

Relationship between age and salary range

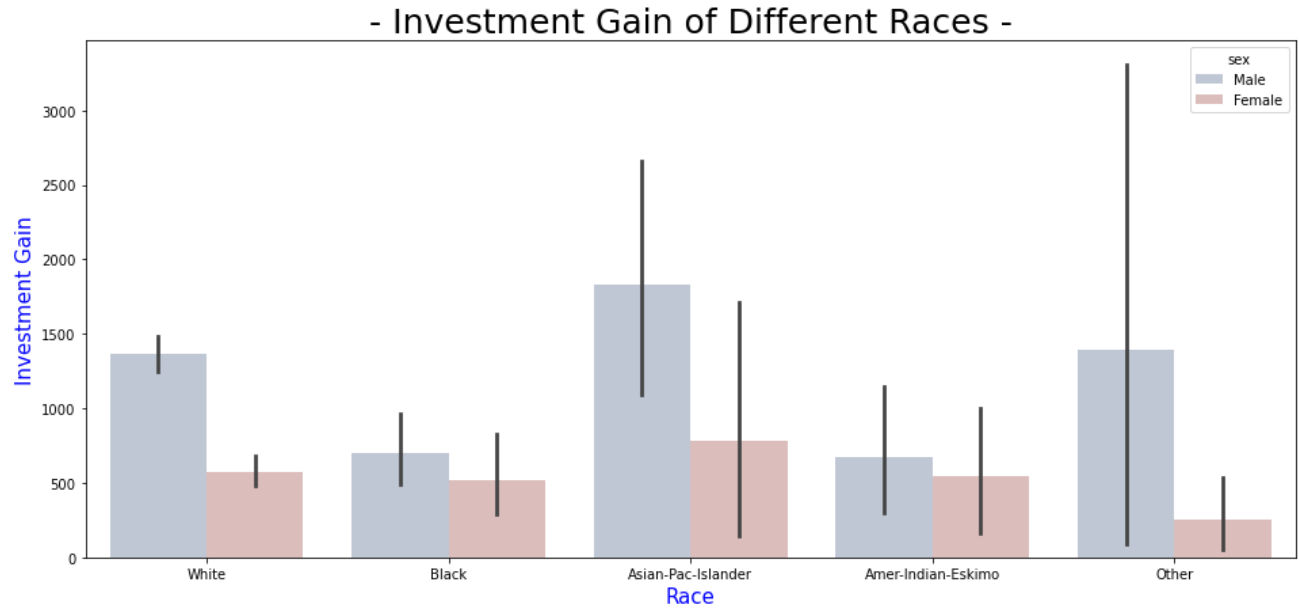


The peak number of individuals that are in the high salary range are at an age range of 40-50. Most individuals that are in the low salary range are in the age range of 20-30. From what we observe from the chart, we can conclude that the salary of individuals is low in their 20s but gradually increase into their 30's and reach a peak at the age of 40-50 years old.

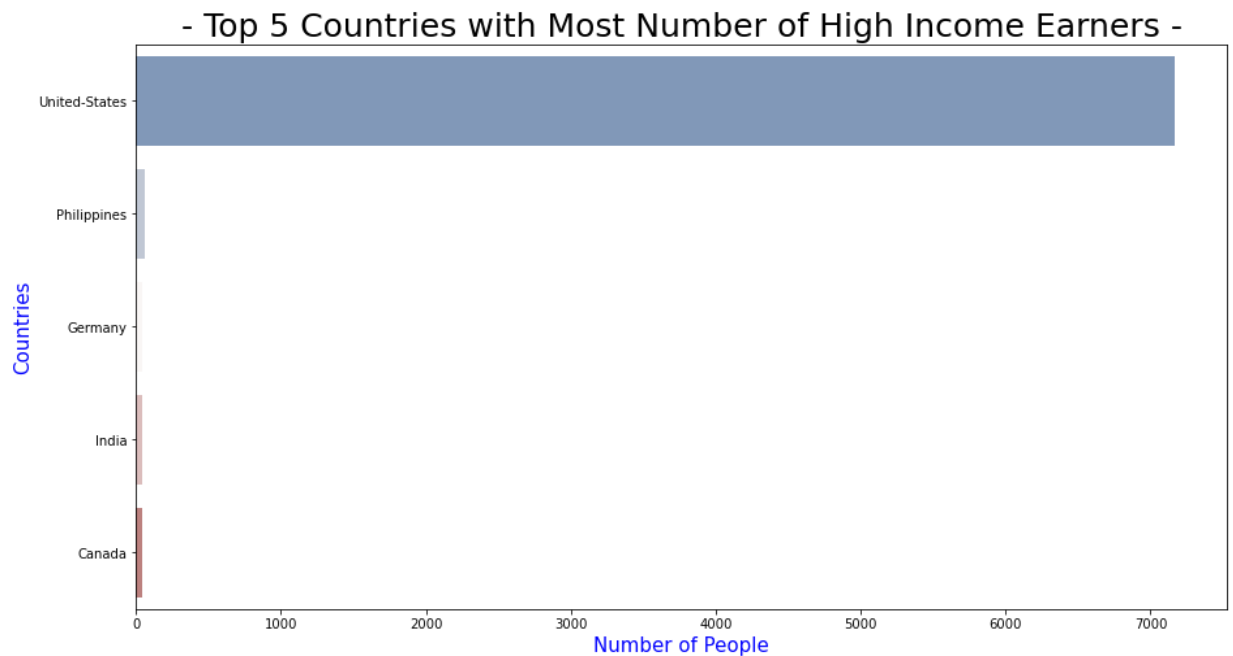
Relationship between weekly hours and age



Most people work 40 hours per week and people that are of younger ages tend to work more hours than that of older people.

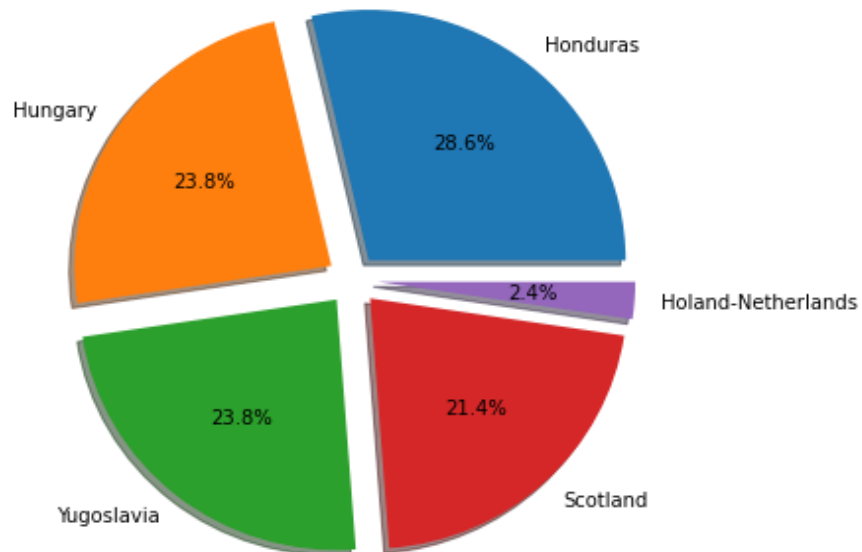


Asian Pacific Islanders have the highest investment gain among all races, while American Indian Eskimo has the lowest. Males have higher investment gain than females regardless of race.



The United States has the highest number of high salary individuals. The United States having so many more high-income individuals than other countries is because the dataset has more individuals from that country.

- Top 5 Countries with Most Low Income Earners -

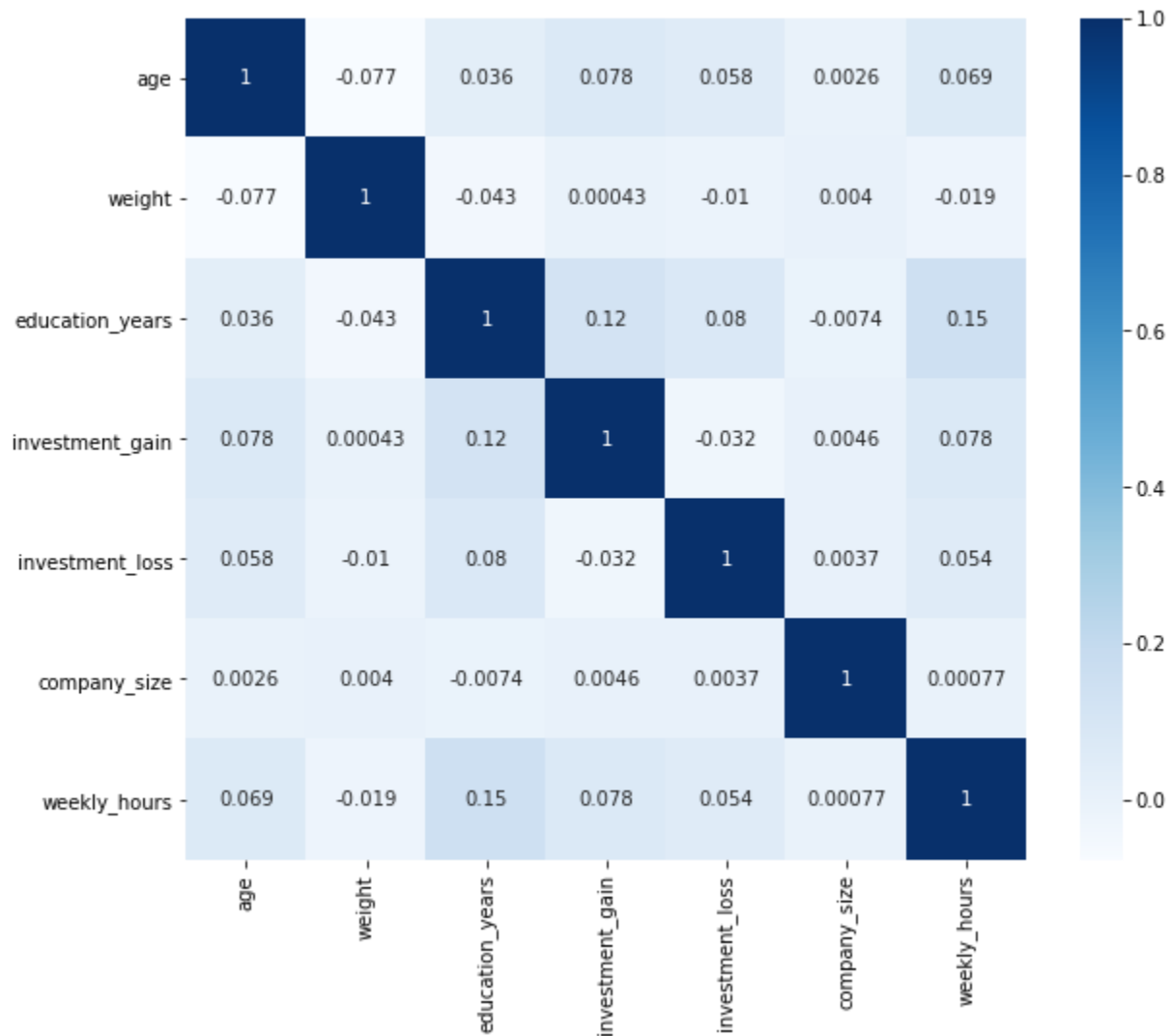


Honduras is the country with the most low-income earners which has a percentage of 28.6%. A large number of people in Honduras are considered to have a low salary range when compared with other countries.

Correlation

Data Correlation is a way to understand the relationship between the variables and attributes in the dataset. Using Correlation, we can gain extra insights on:

- One or multiple attributes depend on another attribute or a cause for another attribute.
- One or multiple attributes are associated with other attributes.



From the heatmap above, there are certain features with high correlation:

Positive Correlation:

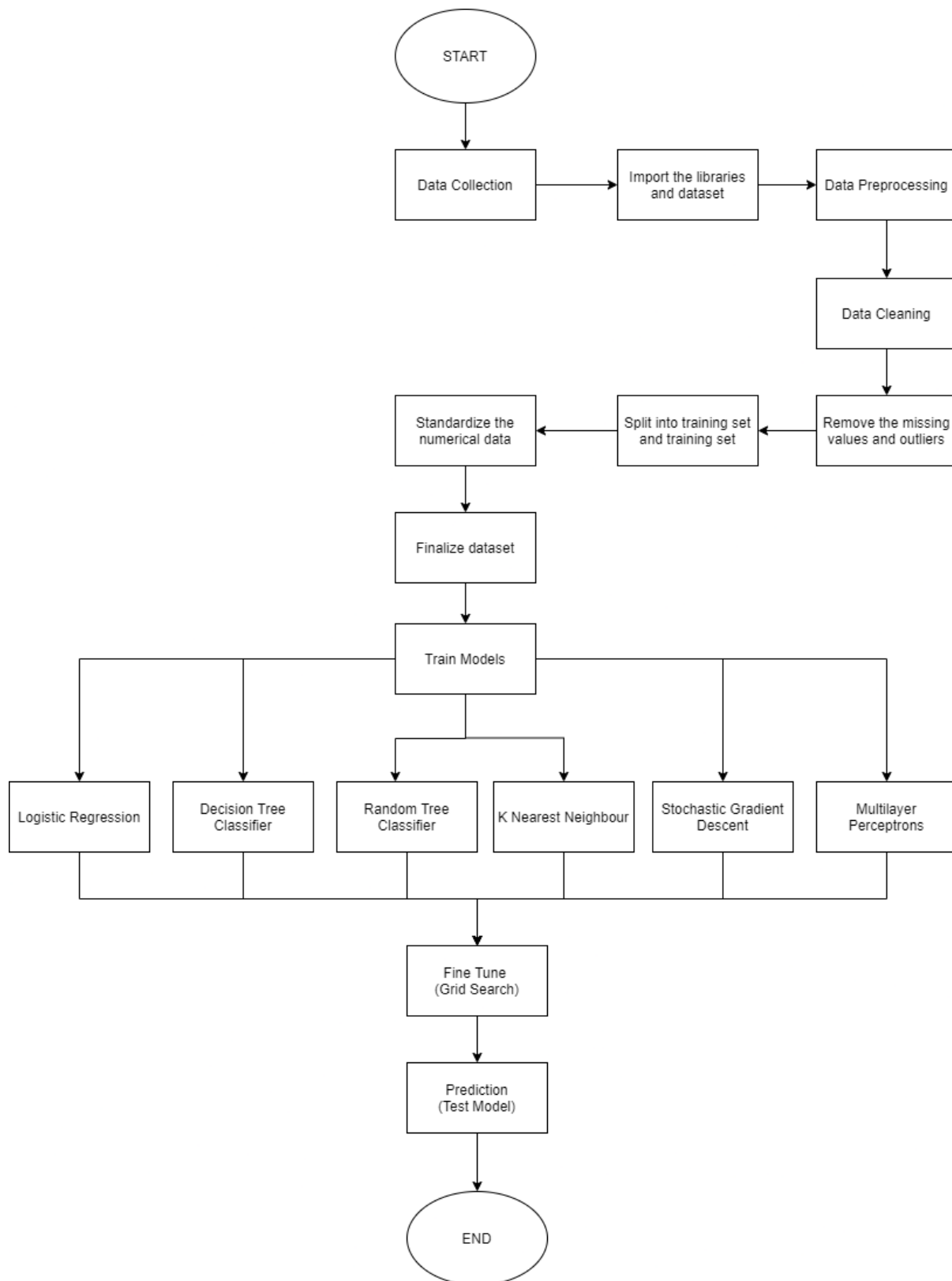
- Age and Investment Gain
- Age and Investment Loss
- Weekly Hours and Age
- Weekly Hours and Education Years
- Education Years and Investment Gain
- Education Years and Investment Lost

Negative Correlation:

- Age and Weight
- Weight and Investment Loss
- Weight and Weekly Hours
- Investment Gain and Investment Loss
- Education Years and Company Size

Data Correlation is important as it will help us determine which features have positive correlation or negative correlation. This would help us tremendously in feature selection to identify which features are needed. For example, when the age is older/higher, the investment gain is higher.

2.4 Overall Framework and Implementation Details



2.4.1 Data Preprocessing

a. Handling Missing Data

```
work_type      1836
age            0
weight         0
education      0
education_years 9
marital_status 0
occupation     1843
relationship    0
race           0
sex            0
investment_gain 0
investment_loss 0
company_size   0
weekly_hours   0
country        583
salary_range   0
dtype: int64
```

Before Remove

```
work_type      0
age            0
weight         0
education      0
education_years 0
marital_status 0
occupation     0
relationship    0
race           0
sex            0
investment_gain 0
investment_loss 0
company_size   0
weekly_hours   0
country        0
salary_range   0
dtype: int64
```

After Remove

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 30153 entries, 0 to 32558
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   work_type             30153 non-null  object
 1   age                   30153 non-null  int64
 2   weight                30153 non-null  int64
 3   education              30153 non-null  object
 4   education_years        30153 non-null  float64
 5   marital_status         30153 non-null  object
 6   occupation              30153 non-null  object
 7   relationship            30153 non-null  object
 8   race                   30153 non-null  object
 9   sex                    30153 non-null  object
10   investment_gain         30153 non-null  int64
11   investment_loss         30153 non-null  int64
12   company_size            30153 non-null  int64
13   weekly_hours            30153 non-null  int64
14   country                 30153 non-null  object
15   salary_range            30153 non-null  object
dtypes: float64(1), int64(6), object(9)
memory usage: 3.9+ MB
```

Summary after handle missing values

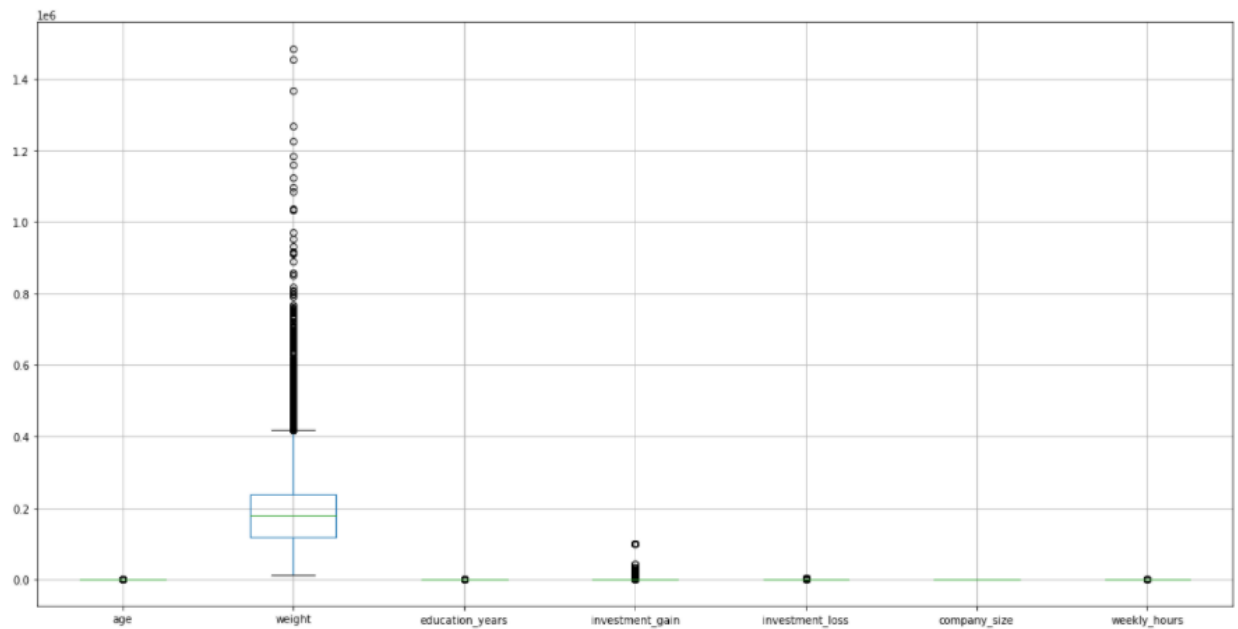
Based on the data exploration section mentioned that there are some missing values from the dataset, in this section we will try to deal with this issue so that the output prediction won't be affected too much. The figures above shows that how many missing values for each feature. As observed that only 3 features having the missing values and in this case since, we have 32561 sample records then we can just directly drop those missing values.

Original Attribute Shape: (32561, 16)
Clean Attribute Shape: (30153, 16)

- Result: $32561 - 30153 = 2408$ missing values (have been removed)

The above figure shows that the before and after the missing values has been removed, which there are in total of 2408 missing values in this dataset, and hence the sample records of cleaned dataset will be left 30153 which is still enough for the model training.

b. Remove Outlier



Another cleansing technique will be removing the outliers, this is because the outliers is something like a noise which means the data not in the expected range or abnormal compare with other data, and it may also affect the accuracy of the model train. As the

observed of the graph from the above figure, we can see that the attribute “weight” that have many outliers, so in this case we need to remove it.

For the figure below shows the result after remove the outliers for this dataset, which the remaining sample records left will be 27613.

	age	company_size	education_years	investment_gain	investment_loss	weekly_hours	weight
count	3.015300e+04	3.015300e+04	3.015300e+04	3.015300e+04	3.015300e+04	3.015300e+04	3.015300e+04
mean	-3.581816e-17	-7.069374e-17	6.150355e-17	-9.425831e-19	1.555262e-17	-1.042143e-16	-1.358498e-16
std	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00
min	-1.632126e+00	-1.738309e+00	-3.576911e+00	-1.474647e-01	-2.185333e-01	-3.332833e+00	-1.665962e+00
25%	-7.946425e-01	-8.629447e-01	-4.396992e-01	-1.474647e-01	-2.185333e-01	-7.762613e-02	-6.830671e-01
50%	-1.094289e-01	5.416791e-03	-4.754771e-02	-1.474647e-01	-2.185333e-01	-7.762613e-02	-1.076257e-01
75%	6.519197e-01	8.562710e-01	1.128907e+00	-1.474647e-01	-2.185333e-01	3.397081e-01	4.527480e-01
max	3.925718e+00	1.745641e+00	2.305361e+00	1.335236e+01	1.055805e+01	4.846918e+00	1.225594e+01

Number of rows before discarding outliers = 30153
Number of rows after discarding outliers = 27613

c. Remove Duplicate Data

Number of duplicate rows = 0

The duplicate data will also affect our model’s prediction, which means that this duplicate data may become a misleading to the model evaluation. For example, while perform the cross validation or the model evaluation using the testing set, but if the testing set also have the similar records, then it may not so accurate, so it is needed to be remove. As the figure shows that there are no duplicate data in this dataset so no need to remove the duplicate data.

d. Separate output vector from input matrix

```
#Drop targeted variable from the matrix X as the input matrix Y, and also the unused feature "weight"
X_drop = ['salary_range', 'weight']
X = new_salary1.drop(X_drop, axis = 1)

#Convert the target attribute's value into boolean type (easy when process, since it only has "high" and "low")
new_salary1['salary_range'] = [1 if x=='high' else 0 for x in new_salary1['salary_range']]
y = new_salary1['salary_range']

#Show the shape and type class of the dataframe for matrix X and Y
print("Shape original dataframe", new_salary1.shape)
print("X shape", X.shape, "type = ", type(X))
print("Y shape", y.shape, "type = ", type(y))

Shape original dataframe (27613, 16)
X shape (27613, 14) type = <class 'pandas.core.frame.DataFrame'>
Y shape (27613,) type = <class 'pandas.core.series.Series'>
```

After perform cleaning of the dataset, the next step is to split them out for the X as our prediction input, and y as our prediction label which our target. As the figures show above the X has drop the feature of “salary_range” and “weight”, for the “salary_range” is for the y while the “weight” has drop is because it can be representing all the other demographic features this because it became a duplicate feature. After splitting, X gets 14 features, while y gets the only one feature which is our target attribute.

e. Split Training and Testing data

```
Original Dataset = (27613, 14)
Training Dataset = (22090, 14)
Testing Dataset = (5523, 14)
```

It is needed to split again the X into training and testing dataset so that we also use for the evaluation purpose at the end of model training. As the figure above shows that the training set get 22090 sample records while for the testing set get 5523, which this distribution is in the ratio of 80% (training set) and 20% (testing set). In addition, since our target attribute has an imbalance class issue so in this case, we need to apply the stratify to solve this issue.

f. Split Numerical and Categorical data

```
cat_attribute = ['work_type', 'education', 'marital_status', 'occupation', 'relationship', 'race', 'sex', 'country']
X_train_num = X_train.drop(cat_attribute, axis = 1)
X_train_cat = X_train[cat_attribute]
```

```
print(X_train_num.shape)
print(X_train_cat.shape)
```

```
(22090, 6)
(22090, 8)
```

Since we have 2 types of attributes, in this case we need to split them out to X_train_num to represent the numerical attributes, while for the X_train_cat to represent the categorical attributes. As a result, there are 6 numerical attributes and 8 categorical attributes.

g. Standardize Numerical Data

Standardize numeric data

```
from sklearn.preprocessing import StandardScaler

#Standardisation of the Numerical Attribute
X_train_num_tr = StandardScaler().fit_transform(X_train_num)
```

```
print('Mean:\n', X_train_num_tr.mean (axis=0))
print('\nStandard Deviation:\n', X_train_num_tr.std (axis=0))
```

```
Mean:
[-4.63187659e-17 -1.85275064e-16 -1.80128534e-17 -9.97140100e-18
 -3.40957583e-17  1.01643959e-16]
```

```
Standard Deviation:
[1.  1.  1.  1.  1.  1.]
```

```
X_train_num_tr #transformed into ndarray
```

```
array([[ -1.24690729, -0.45794306, -0.24826905, -0.03799699,  0.63229247,
        -0.03557418],
       [-1.40255447, -1.27997803, -0.24826905, -0.03799699, -1.57134843,
         1.82526817],
       [-0.85778932, -0.45794306, -0.24826905, -0.03799699,  0.05864627,
        -0.03557418],
       ...,
       [ 2.17733079, -0.45794306, -0.24826905, -0.03799699,  1.12199044,
        -0.03557418],
       [-1.0912601 , -1.69099551, -0.24826905, -0.03799699, -1.37896708,
         2.29047876],
       [-1.0912601 ,  1.18612686, -0.24826905, -0.03799699,  0.50986797,
        -0.03557418]])
```

Standardization is a process that rescale the attributes so that they have the mean value of 0 and standard deviation of 1. It is needed to perform standardization to the numerical data, this is because the data won't get a different scale of range and after perform this may help to improve the effectiveness in the model training phase.

h. Perform One Hot Encoding

Since we have categorical data in this dataset, so it is needed to perform labelling for this kind of data which helps to improve the accuracy of the training. And for the reason we choose this one hot encoding method instead of label encoding is because our data is majority in the nominal data type while the label encoding is preferable for the ordinal data types. The figures below shows that the number of different values for each category, after performing the encoding it is needed to combine the numerical and categorical data into `X_train_tr`.

```
These are the number of different values per category
(22090, 7)
(22090, 14)
(22090, 7)
(22090, 14)
(22090, 6)
(22090, 5)
(22090, 1)
(22090, 40)
```

```
#Combine all One Hot Encoded Category Table in to one Numpy Array  
combine=np.hstack((X_train_cat_1,X_train_cat_2,X_train_cat_3,X_train_cat_4,X_train_cat_5,X_train_cat_6,X_train_cat_7,X_train_cat_8))  
combine.shape
```

(22090, 94)

Finalize Train Set

```
#Combine Categorical and Numerical Data Together for Training Set
#Original Training Set
X_train_tr=np.hstack((X_train_num_tr,combine))

print(X_train_tr.shape)
print(y_train.shape)
```

(22090, 100)
(22090,)

i. Preprocessing on Test Set

In this part it is similar with the training process as mentioned above, which the process of perform standardization on numerical data, one hot encoding on categorical data, and then combine all of them become “X_test_tr”.

These are the number of different values per category

```
(5523, 7)
(5523, 14)
(5523, 7)
(5523, 14)
(5523, 6)
(5523, 5)
(5523, 1)
(5523, 40)
```

```
#Combine all One Hot Encoded Category Lable in to one Numpy Array
combine_test=np.hstack((X_test_cat_1,X_test_cat_2,X_test_cat_3,X_test_cat_4,X_test_cat_5,X_test_cat_6,X_test_cat_7,X_test_cat_8),
combine_test.shape
```

```
(5523, 94)
```

Finalize Test Set

```
#Combine Categorical and Numerical Data Together for Testing Dataset
X_test_tr=np.hstack((X_test_num_tr,combine_test))
```

```
print(X_test_tr.shape)
print(y_test.shape)
```

```
(5523, 100)
(5523,)
```


Summary of the preprocessing part:

As the figure below shows that there are 22090 sample records for the training set and for the 5523 is for testing set, which this is split from the original dataset of 27613 sample records.

```
Original Dataset
Salary: (27613, 16)

Preprocessed and Combined Training Set
X_train_tr: (22090, 100)
y_train    : (22090,)

Preprocessed and Combined Testing Set
X_test_tr: (5523, 100)
y_test    : (5523,)
```

Data Preprocessing End

2.4.2 Model Training and Validation

After the data pre-processing, the clean dataset is used to build 6 different classification models to classify the salary range of individuals. Listed below are the overview of the 6 models that were built.

a. Logistic Regression

Logistic Regression is a type of classification algorithm used when the dependent variable is a categorical value. Logistic regression is most commonly used when the dependent variable is dichotomous or a binary output. Logistic Regression was used in Biological Sciences in the early twentieth century. It was then used in many social science applications.

b. Decision Tree Classifier

Decision Tree Classifier is a supervised algorithm that uses a set of rules to make decisions, similarly to how humans make decisions. Decision trees can perform both classification and also regression tasks. Decision tree uses dataset features to create yes or no questions and continuously split the dataset until all data points belong to each class. Decision trees is a simple algorithm but it has several advantages such as easily interpreted as it can be visualized. Also, no preprocessing required as data doesn't need to be prepared before building the model. Finally, the algorithm is great at handling all types of data. (Bento, 2018).

c. Random Forest Classifier

Random forest consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes become the model's prediction. A large number of relatively uncorrelated trees operating as a committee will outperform any of the individual constituent trees. The low correlation between models can produce ensemble predictions that are more accurate than any of the individual predictions because the trees serve as a protection from each individual error.

d. K Nearest Neighbour Classifier

The K Nearest Neighbour (KNN) is a simple, yet coherent and versatile supervised learning algorithm. It can be used to solve both regression and classification problems. KNN is an algorithm that makes predictions based on the nature of other data points that are present close to the training dataset. This algorithm is commonly used when there is little to no information about the distribution of data. Finally, the algorithm is non-parametric in nature which means that it does not make any underlying assumption about the data.

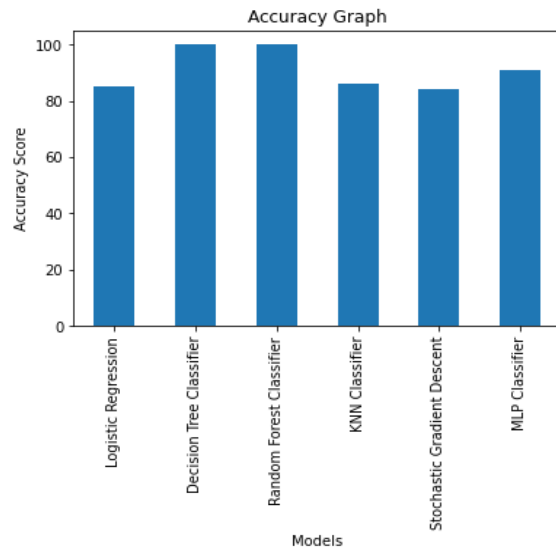
e. Stochastic Gradient Descent (SGD)

Stochastic gradient descent (SGD) is a very commonly used algorithm. This algorithm forms the basis of Neural Networks. SGD is an iterative algorithm where it starts from a random point on a function and travels down its slope in steps until it reaches the lowest point of that function. This algorithm is useful in scenarios where the optimal points cannot be found by equating the slope of the function to 0. (Srinivasan, 2019).

f. Multilayer Perceptron (MLP)

Multilayer Perceptron (MLP) is a supplement of feed forward neural network. It consists of three types of layers which are the input layer, out layer and hidden layer. The input layer receives the input signal to be processed. The required tasks including prediction and classification are performed by the output layer. The hidden layers that are in between the input and output layer serve as the computational engine of the MLP.

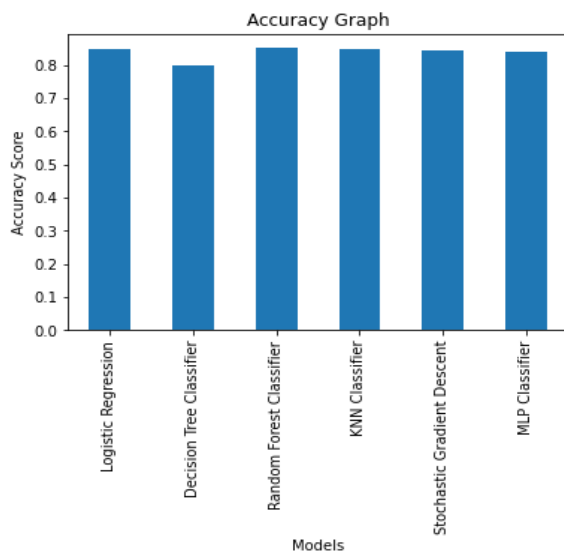
Before Cross Validation



	Accuracy
Logistic Regression	85.0973
Decision Tree Classifier	99.9864
Random Forest Classifier	99.9864
KNN Classifier	86.1702
Stochastic Gradient Descent	84.1603
MLP Classifier	90.7288

Based on the figures shown above, we can observe that the data has overfitting issues in two out of the six algorithms. Decision Tree Classifier and Random Forest Classifier are the algorithms with overfitting issues where the model fits against the training data too well. Hence, we have to perform k-fold cross validation on all algorithms to resolve overfitting issues.

After Cross Validation



	Accuracy
Logistic Regression	0.8494
Decision Tree Classifier	0.8004
Random Forest Classifier	0.8513
KNN Classifier	0.8495
Stochastic Gradient Descent	0.8443
MLP Classifier	0.8388

Based on the figures, we can observe that the accuracy has decreased after performing cross validation. The models that were previously overfitted has now been resolved. The overall accuracy has been decreased, so we have to perform fine tuning to improve the model.

	Fit Time	Score Time	Accuracy	Precision	Recall	F1 score
Logistic Regression	0.4624	0.0045	0.8494	0.7201	0.5694	0.7705
Decision Tree Classifier	0.1022	0.0052	0.8004	0.5653	0.5880	0.7228
Random Forest Classifier	1.2779	0.0723	0.8513	0.7113	0.6002	0.7783
KNN Classifier	0.0047	1.1128	0.8495	0.7116	0.5855	0.7735
Stochastic Gradient Descent	0.1140	0.0046	0.8443	0.7148	0.5458	0.7597
MLP Classifier	11.9735	0.0082	0.8388	0.6591	0.6262	0.7687

According to the figure above, we can see that random forest classifier produces the model with the best accuracy, and F1 score when compared with the other algorithms. Hence, we choose the random forest classifier model as our best performing model.

2.4.3 Fine Tuning

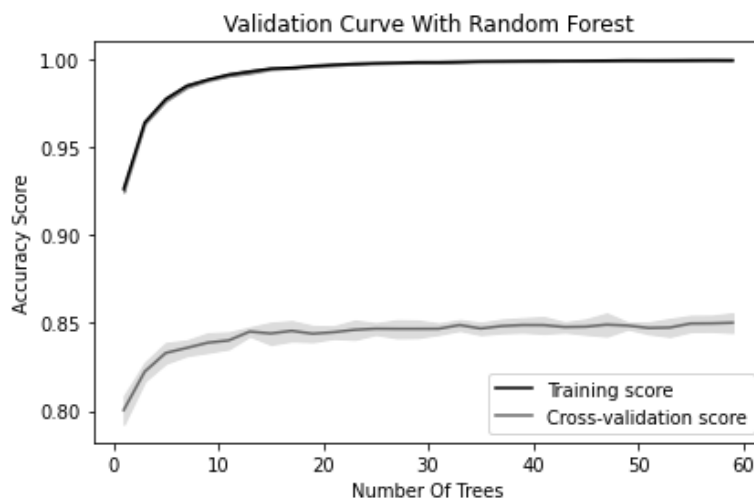
After training the model and performing cross validation, we will select the algorithm with the best performing algorithm, which is Random Forest Classifier. We will then perform Hyperparameter Tuning to further improve the performance of the model and enhance the score. Hyperparameters are settings of a machine learning algorithm that needs to be set prior to training. To find the best hyperparameter values for these settings, we shall use Grid Search Cross Validation.

```
'bootstrap': [True, False],  
'max_features': [5, 8, 9, 10, 11],  
'n_estimators': [80, 100, 110, 120]
```

We have chosen three hyperparameters which are bootstrap, max_features and n_estimators. There are 40 candidates in total by multiplying the number of values we have for each hyperparameter. The cross-validation count is set to 3-fold which is 120 fits in total. The Grid Search needs to run 120 times to find the best hyperparameters to be used.

```
Best Hyperparameter Settings: {'bootstrap': True, 'max_features': 10, 'n_estimators': 110}  
Best Hyperparameter Settings: 0.8506565607421598
```

After grid search is complete, we get the best hyperparameters for the model. And the best hyperparameter produces a score of 85.1%



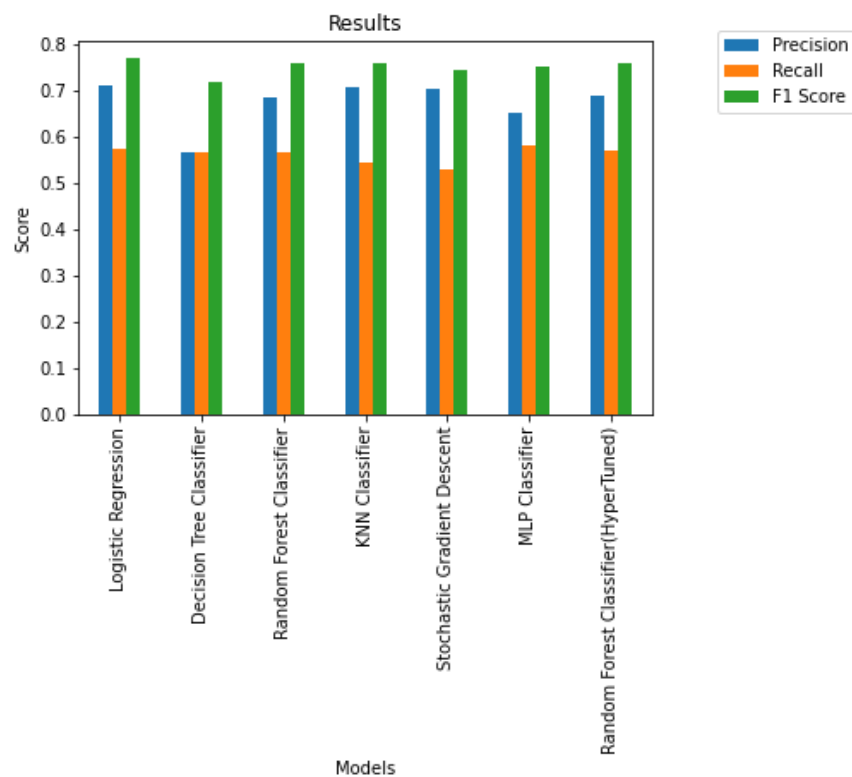
Based on the chart above, we can see that the number of trees reach a highest point at around 10 to 20. Then, it slowly starts to plateau and only rises very slightly when the number of trees increase.

2.4.4 Testing

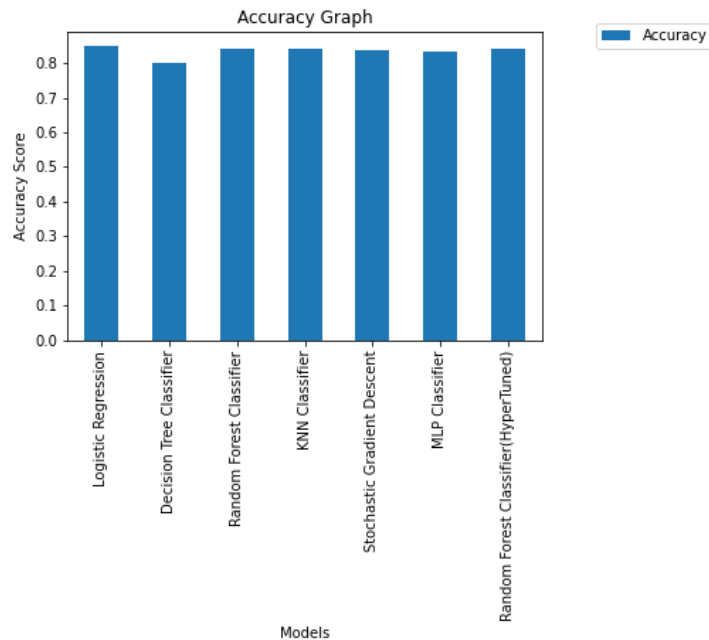
After the best hyperparameters are determined through grid search cross validation, we will test the model on the test set to evaluate its performance.

	Fit time	Score time	Accuracy	Precision	Recall	F1 score
Logistic Regression	0.0510	0.0024	0.8475	0.7106	0.5733	0.7689
Decision Tree Classifier	0.0216	0.0025	0.7996	0.5658	0.5671	0.7179
Random Forest Classifier	0.3316	0.0212	0.8403	0.6858	0.5678	0.7599
KNN Classifier	0.0018	0.1048	0.8430	0.7085	0.5451	0.7585
Stochastic Gradient Descent	0.0340	0.0025	0.8376	0.7055	0.5294	0.7459
MLP Classifier	3.0302	0.0034	0.8313	0.6522	0.5827	0.7530
Random Forest Classifier(HyperTuned)	0.3578	0.0225	0.8412	0.6884	0.5694	0.7611

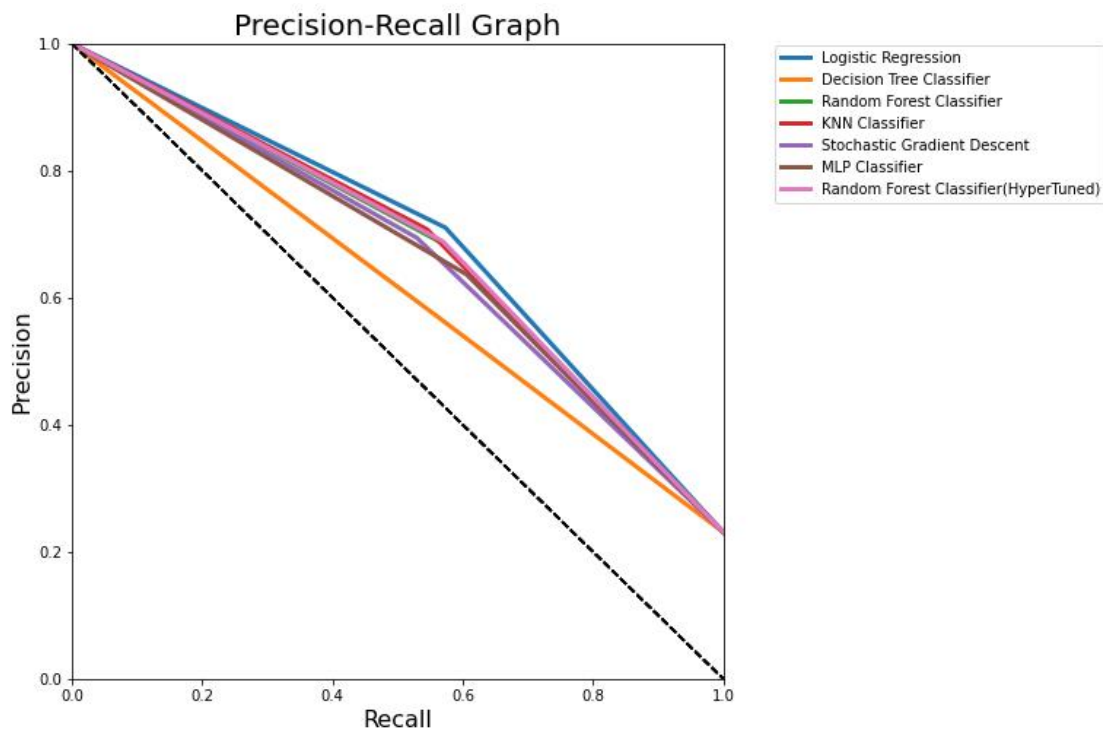
According to the table above, we can observe that the accuracy, precision, recall and f1 score of the Hyperparameter Tuned Random Forest Classifier is higher than the Random Forest Classifier before Hyperparameter Tuning.



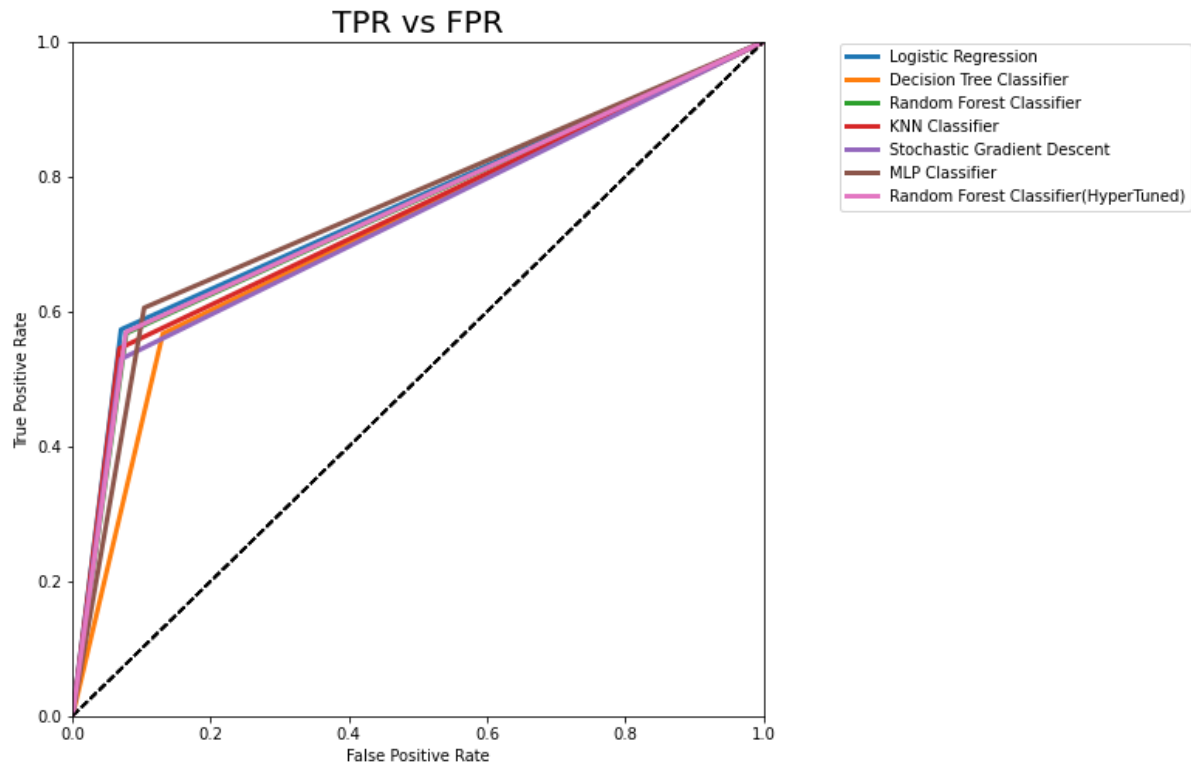
Based on this figure, the visualization can be used to observe the difference between the precision, recall and f1 score of each algorithms.



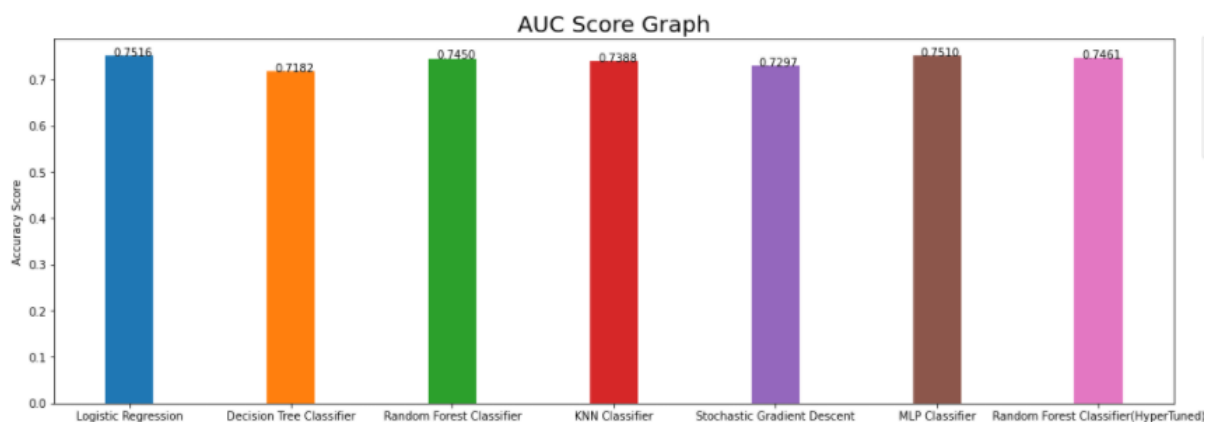
When comparing the accuracy, we can observe that quite a few algorithms have very similar accuracy except decision tree classifier.



From the precision recall graph, we can see that logistic regression is the furthest away from the dotted line closely followed by the Tuned Random Forest Classifier. The decision tree classifier performs the worst as its line is closest to the dotted line.



From the comparison between TPR vs FPR, we can see that the Tuned Random Forest Classifier has the best performance and it is very similar to the performance of Logistic Regression.

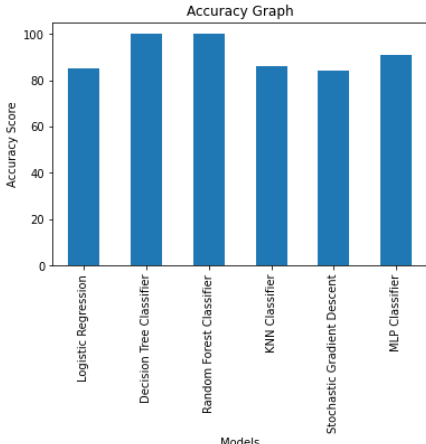
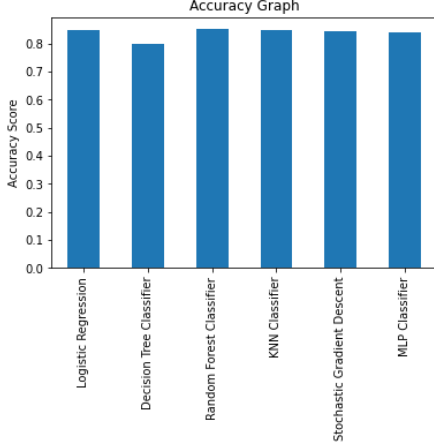


Finally, from the AUC score graph, we can see that the Tuned Random Forest Classifier(pink) performs slightly poorer than MLP Classifier and Logistic Regression. It performs better than the Random Forest Classifier before tuning and also decision tree classifier, KNN classifier and Stochastic Gradient Descent.

3.0 Results and Discussion

This section will discuss about the results of the model trained which includes the performance analysis, and model comparison.

3.1 Experimental Results

Algorithms	Accuracy	
	Without Cross Validation	With Cross Validation
Logistic Regression	85.0973	84.94
Decision Tree Classifier	99.9864 (Overfit)	80.04
Random Forest Classifier	99.9864 (Overfit)	85.13
KNN Classifier	86.1702	84.95
Stochastic Gradient Descent	84.1603	84.43
MLP Classifier	90.7288	84.06
		

The table above shows that the summary for both accuracy which with and without using cross validation. Firstly, for the accuracy without perform the cross validation which having 2 algorithms that under the overfitting status which it consists of 99.99% of accuracy score. In order to deal with this issue there are many ways to do that, but in this assignment we used the k-fold cross validation to solve this issue. This k-fold cross validation is used to split the training set into k folds, and for each of the fold which has been split out will act as a testing data to evaluate the model performance. Then this process will keep repeating the process until the end of K value. In

addition, it having the equal opportunity for each of the fold sample which will be used in the hold out set 1 time and used to train the model $k-1$ times.

By comparing both accuracy from the table above, the accuracy which perform the cross validation is more promising compare with the previous accuracy which does not perform the cross validation. However, the overfitting issue has been solved but we still need to evaluate the performance by using other methods.

Metrics for Evaluation

	Fit Time	Score Time	Accuracy	Precision	Recall	F1 score
Logistic Regression	0.4363	0.0045	0.8494	0.7201	0.5694	0.7705
Decision Tree Classifier	0.1017	0.0052	0.8004	0.5653	0.5880	0.7228
Random Forest Classifier	1.2700	0.0697	0.8513	0.7113	0.6002	0.7783
KNN Classifier	0.0044	1.1306	0.8495	0.7116	0.5855	0.7735
Stochastic Gradient Descent	0.1148	0.0046	0.8443	0.7148	0.5458	0.7597
MLP Classifier	11.6742	0.0078	0.8406	0.6613	0.6398	0.7731

Based on the figure above, it shows that an overall performance for each of the model trained which include their fit time, score time, accuracy, precision, recall, and f1score. By having these score, we can analyze their performance more effectively. The below table explain that each of the evaluate methods:

Formula	Explanation
$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$	Also known as positive predictive value which use to find out the accuracy of positive prediction.
$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$	Also known as true positive rate which used to find out the ratio of true positive sample from the total of correct prediction.
$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$	<p>F1 score, it is a harmonic mean of the precision and recall score, which the measured value in the range of 0 and 1.</p> <p><i>Note: 0 represent worse score, 1 represent best score.</i></p>
$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{FP} + \text{TN} + \text{FN})}$	Accuracy score, although in this classification it is not so importance score to determine whether a model is good or not, it also able to tell us that the overall correct output.

Summary of evaluation metrics:

Based on the accuracy table shows above the longest fit time algorithm will be the MLP classifier, while the KNN classifier takes the shortest time to complete the model fitting. Moreover, the highest scoring of algorithms is belonging to the Random Forest Classifier which takes the highest accuracy score, recall score and also the f1 score.

3.2 Performance Analysis

There are several ways to perform analysis for each algorithm, in this assignment we use 4 methods to analyze the model trained:

1. Classification Report
2. Confusion Matrix
3. Precision-Recall Graph
4. ROC Curve

Classification Report

The classification report is to show the overall performance in a report text format which include the precision, recall, f1-score and support. Below figures are the classification report for each of the algorithms.

Logistic Regression					Decision Tree Classifier				
Classification Report:					Classification Report:				
	precision	recall	f1-score	support		precision	recall	f1-score	support
0	0.88	0.93	0.91	16988	0	0.87	0.86	0.87	16988
1	0.72	0.57	0.64	5102	1	0.57	0.59	0.58	5102
accuracy			0.85	22090	accuracy			0.80	22090
macro avg	0.80	0.75	0.77	22090	macro avg	0.72	0.73	0.72	22090
weighted avg	0.84	0.85	0.84	22090	weighted avg	0.80	0.80	0.80	22090
Random Forest Classifier					KNN Classifier				
Classification Report:					Classification Report:				
	precision	recall	f1-score	support		precision	recall	f1-score	support
0	0.89	0.93	0.91	16988	0	0.88	0.93	0.90	16988
1	0.71	0.60	0.65	5102	1	0.71	0.59	0.64	5102
accuracy			0.85	22090	accuracy			0.85	22090
macro avg	0.80	0.76	0.78	22090	macro avg	0.80	0.76	0.77	22090
weighted avg	0.85	0.85	0.85	22090	weighted avg	0.84	0.85	0.84	22090
Stochastic Gradient Descent					MLP Classifier				
Classification Report:					Classification Report:				
	precision	recall	f1-score	support		precision	recall	f1-score	support
0	0.87	0.93	0.90	16988	0	0.88	0.91	0.90	16988
1	0.71	0.55	0.62	5102	1	0.68	0.60	0.64	5102
accuracy			0.84	22090	accuracy			0.84	22090
macro avg	0.79	0.74	0.76	22090	macro avg	0.78	0.76	0.77	22090
weighted avg	0.84	0.84	0.84	22090	weighted avg	0.84	0.84	0.84	22090

Confusion Matrix

This confusion matrix can provide us how the overall prediction looks like, it consists of True Negative, True Positive, False Negative, False Positive. Where these values can let us know more about the precision and recall score.

		True Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

1. **True Negative (TN):** It measure how many false items are predicted as negative.
2. **True Positive (TP):** It measure how many true items are predicted as positive.
3. **False Negative (FN):** It measure how many true items are predicted as negative.
4. **False Positive (FP):** It measure how many false items are predicted as positive.

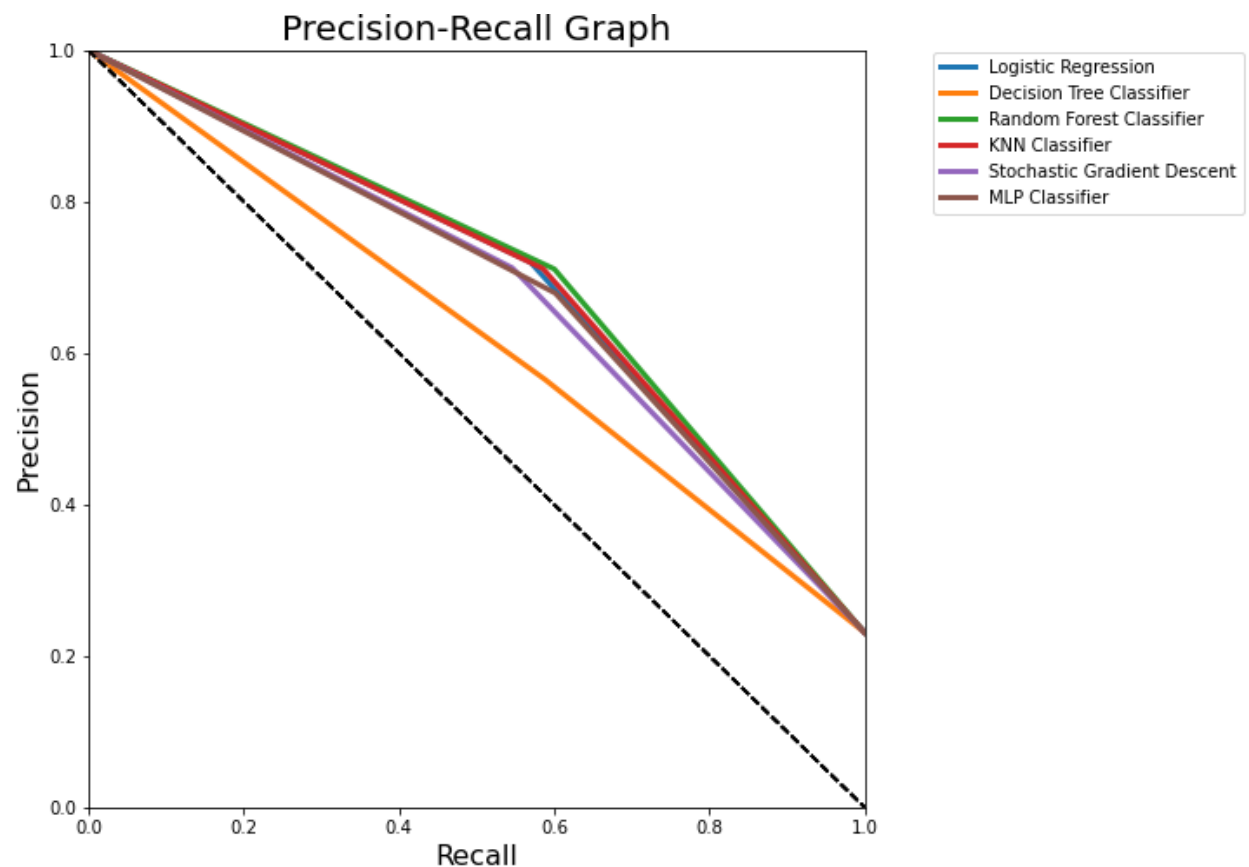
Logistic Regression Confusion Matrix: <pre>[[15859 1129] [2197 2905]]</pre> Accuracy: 0.8494341330918967 TN, FP, FN, TP : 15859 1129 2197 2905	Decision Tree Classifier Confusion Matrix: <pre>[[14681 2307] [2102 3000]]</pre> Accuracy: 0.8004074241738343 TN, FP, FN, TP : 14681 2307 2102 3000
Random Forest Classifier Confusion Matrix: <pre>[[15744 1244] [2040 3062]]</pre> Accuracy: 0.8513354459031236 TN, FP, FN, TP : 15744 1244 2040 3062	KNN Classifier Confusion Matrix: <pre>[[15778 1210] [2115 2987]]</pre> Accuracy: 0.849479402444545 TN, FP, FN, TP : 15778 1210 2115 2987
Stochastic Gradient Descent Confusion Matrix: <pre>[[15865 1123] [2317 2785]]</pre> Accuracy: 0.8442734268899955 TN, FP, FN, TP : 15865 1123 2317 2785	MLP Classifier Confusion Matrix: <pre>[[15540 1448] [2034 3068]]</pre> Accuracy: 0.8423721140787687 TN, FP, FN, TP : 15540 1448 2034 3068

The table above shows that the confusion matrix for each algorithm. As a result, the majority predicted of true positive is the MLP classifier, but for the best overall performance is still belongs to the Random Forest Classifier.

Precision-Recall Graph

Precision-Recall graph, it is also called P-R curve which it is plot by using the precision and recall score for each algorithm. Whereas it can be used to show the relationship for both values when changing the threshold values. In order to know which algorithms is consider a good classifier on this dataset, we just have to see which classifier having a curve that close to the top right corner.

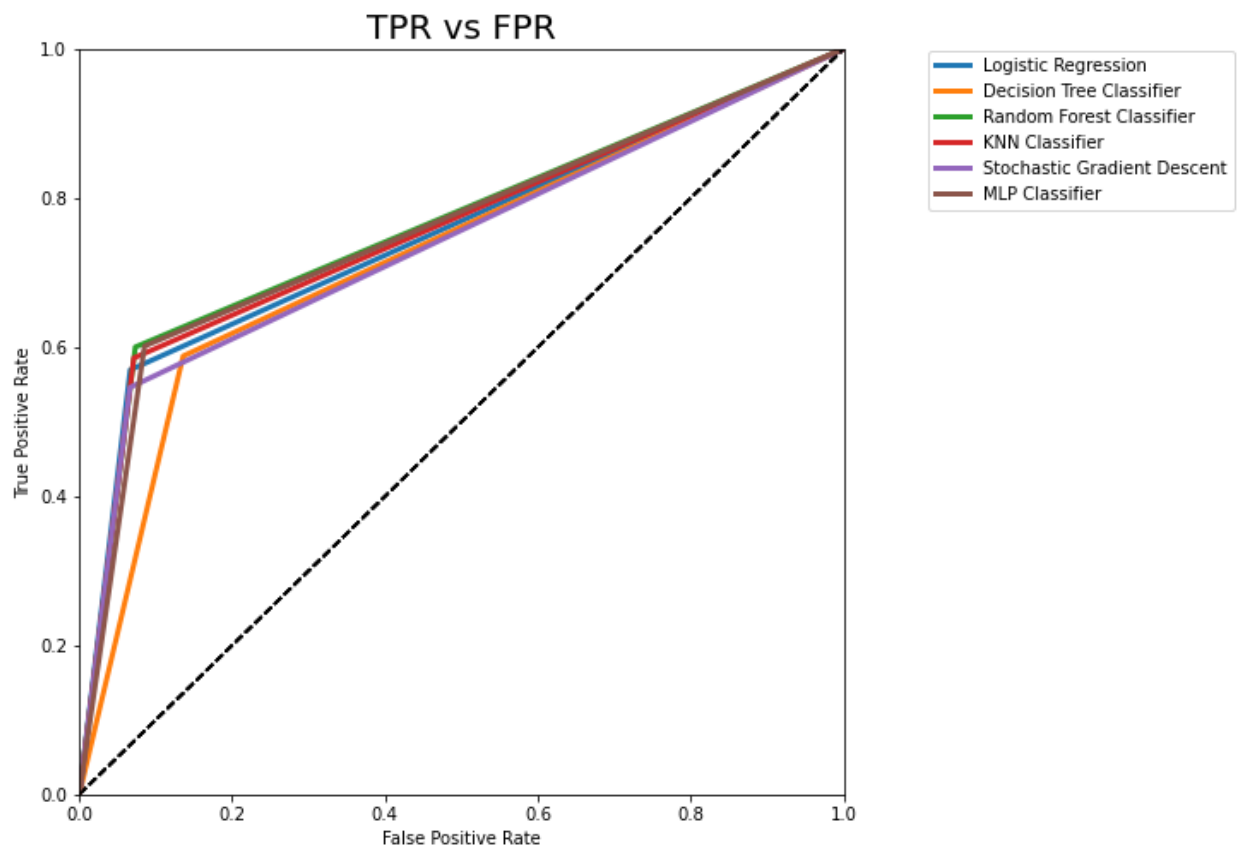
Based on the figure below, it shows that the Random Forest Classifier is the closest to the top right corner, while the Decision Tree Classifier is far away from the top right corner. Therefore, we can consider that the Random Forest Classifier is a good model for this dataset.



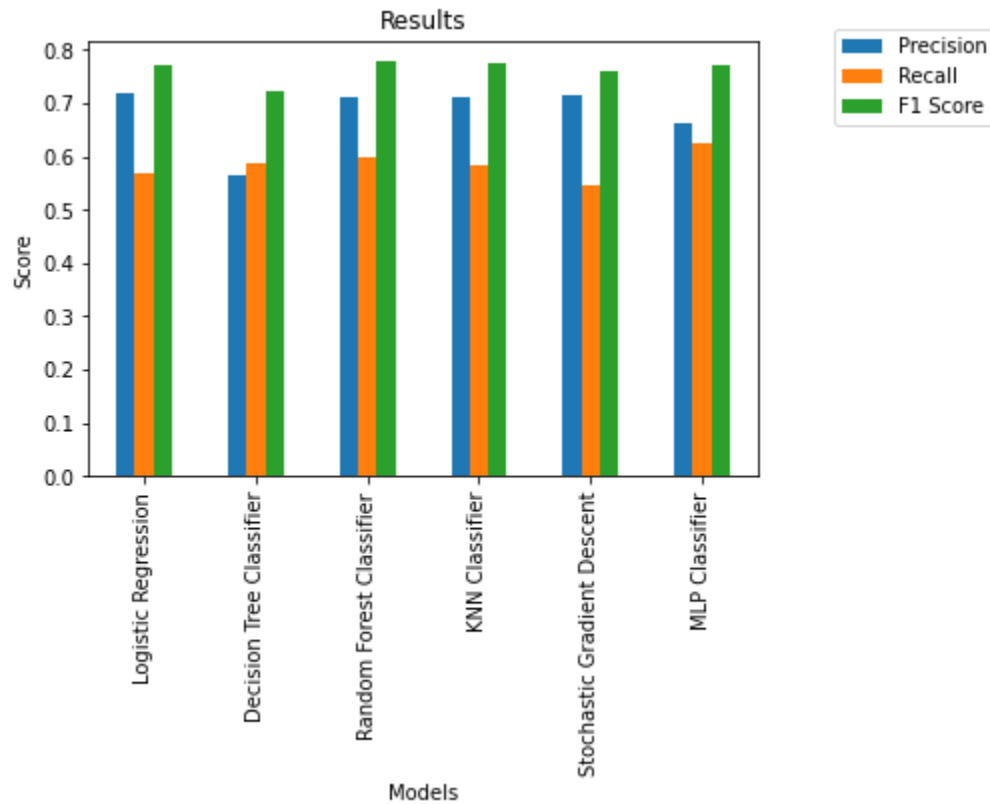
ROC Curve

ROC Curve also known as Receiver Operating Characteristic curve; it is also one of the common tools for the binary classification evaluation. It is plot by using the recall score and also the false positive rate (FPR). As opposite to the Precision-Recall graph, the consideration of a good classifier in this curve is close to the top-left corner.

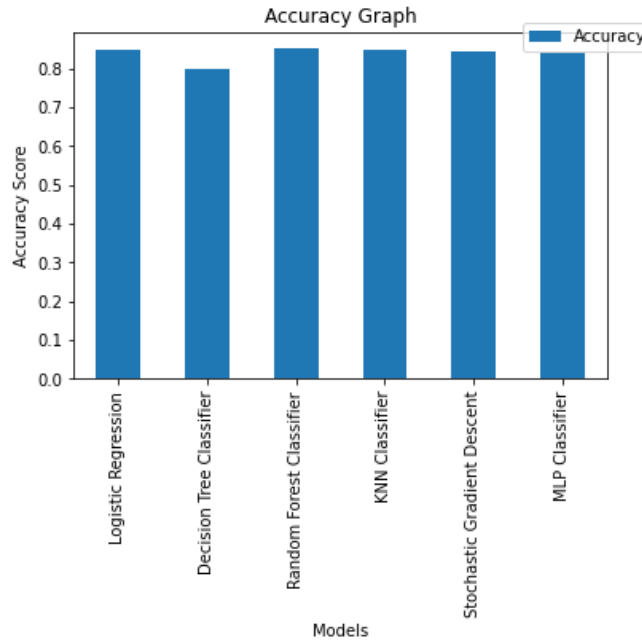
Based on the figure below, it shows that the result for each algorithm, which the closest to the top left corner is belongs to the Random Forest Classifier again, while the Decision tree classifier is also the same as the bad classifier in this roc curve.



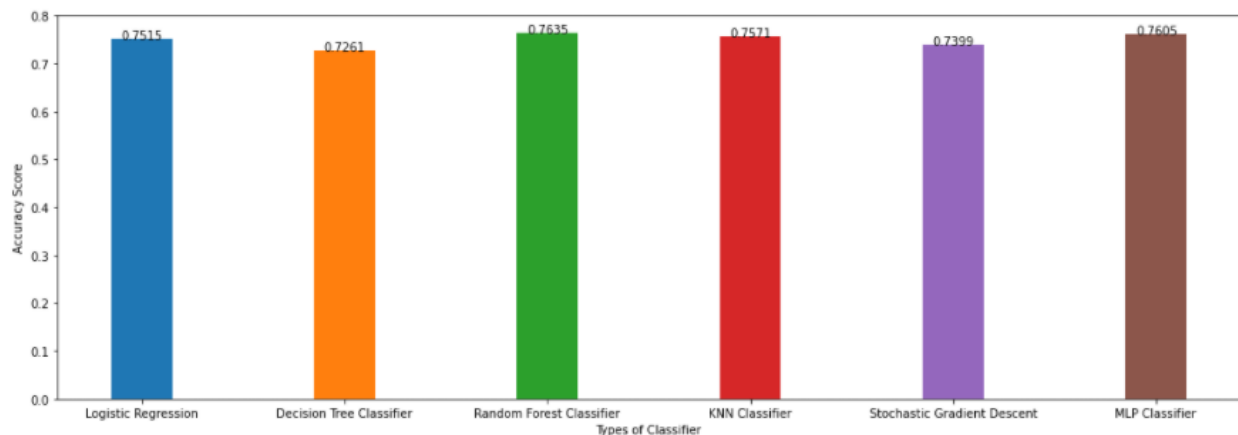
3.3 Comparison of the algorithms



Based on the results shown in the figure above, we can use it to evaluate the performance of different machine learning algorithms. The evaluation method uses the precision, recall and f1 score to determine the performance of each model. As can be observed from the graph, Random Forest Classifier has the best overall performance when compared with the other algorithms. Random Forest Classifier achieves an accuracy of 85% and f1 score of 77.8% which is the highest among all algorithms. However, its precision is lower than KNN Classifier, Stochastic Gradient Descent and also Logistic Regression. The recall of Random Forest Classifier of 60% only loses to MLP Classifier which has the highest recall of 62.4%



From this visualization, we can see that the accuracy of Random Forest Classifier of 85% is slightly higher than Logistic Regression and MLP Classifier which are trailing closely. Decision Tree Classifier is the worst performing algorithm as it produces the model with 80% accuracy which is comparably lower than the other algorithms which have an average accuracy of 84%.



Listed above is the AUC score of each algorithms which can also be used to compare the performance of each classifier. As can be observed from the graph, Random Forest Classifier has AUC score of 0.7635 which is the highest among all algorithms. MLP Classifier's AUC score of 0.7605 comes in at a close second slightly lower than Random Forest Classifier. Hence, we can conclude that out of all the algorithms, Random Forest Classifier can most accurately distinguish between classes as it has the highest AUC score.

4.0 Conclusion

The purpose of this project is to make the most accurate prediction of the salary range of individuals based on the dataset provided. We used six classification algorithms to train the models and compare the performance of each model and chose the model that performed the best when compared with the other models. After training, cross validation is performed and the results are again compared and the overall results is much lower than before cross validation but overfitting issues are resolved. The best model is produced by Random Forest Classifier. After performing k-fold cross validation, Random Forest Classifier had an accuracy of 85% and F1 score of 0.7783 which is the highest among all algorithms. But after hyperparameter tuning the Random Forest Classifier model, the accuracy, precision, recall and f1 score are all higher than before tuning. However, when the model is tested using the test sets, Logistic Regression ended up producing better results than the Random Forest Classifier model even after hyperparameter tuning. This came as a surprised to us, as Logistic Regression did not undergo hyperparameter tuning but produced accuracy, precision, recall and f1 score that are slightly higher than the Random Forest Classification Model. Hence, we conclude that Logistic Regression and Random Forest Classifier can be used to predict the salary range of individuals given the dataset with an accuracy of about 74.6%.

References

Bento, C. (2020). Decision Tree Classifier explained in real-life: picking a vacation destination. Retrieved from: <https://towardsdatascience.com/decision-tree-classifier-explained-in-real-life-picking-a-vacation-destination-6226b2b60575#:~:text=Decision%20trees%20are%20a%20rule,same%20class%20are%20grouped%20together.>

Srivivasan, A. (2019). Stochastic Gradient Descent - Clearly Explained. Retrieved from: <https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31>

S. Abirami. (2020). The Digital Twin Paradigm for Smarter Systems and Environments: The Industry Use Cases. Retrieved from: <https://doi.org/10.1016/bs.adcom.2019.09.007>.