Income Prediction based on 1994 United States Census Data: Determining Whether a Person Earns over $50k using Classification Methods

Weiheng Hong (s3643760)
Jia Jun Yong (s3688090)

Practical Data Science
School of Science
RMIT University

Vincent: s3643760@student.rmit.edu.au
Jia Jun: s3688090@student.rmit.edu.au

31 May 2019

Table of Content:

Abstract:

The aim of this report is to determine whether a person earns over $50k using classification methods based on 1994 United States Census Data which is also commonly known as the “Adult” dataset. To do so, we started off with sanitising the data, generating visualisations to look for relationships between attributes, then performing data modelling was performed with K-nearest Neighbour Classifier and Decision Tree Classifier using *scikit-learn* library. Finally, feature selection was performed to help answering the research question. As a result, it has been proven that for dealing with data modelling, Decision Tree Classifier is the better choice for this particular dataset as it produces a higher accuracy score and a lower error rate than using K-nearest Neighbour Classifier. Therefore, it is recommended that Decision Tree Classifier should be used when performing classification with the “Adult” dataset.

Introduction:

The “1994 United States Census Income” dataset, also known as the “Adult” dataset contains 14 attributes including one class attribute and 32561 instances. Thanks to its donors, Ronny and Barry who both graciously donated the dataset in 1996, it has been one of the most popular datasets that is used for predicting whether a person’s income exceeds US$50k per year based on the census data collected in 1994 in the United States of America **[1]**. It has been chosen due to its high success rate to perform data modelling with classification methods as recommended by the UCI Machine learning Repository website **[1]**. The aim of the report is to explore the dataset using visualisations, model the given data using classification methods with K-nearest Neighbour Classifier and Decision Tree Classifier, analyse the performance of each classifier and provide recommendation for which classifier should be used.

Attributes of the dataset are as follow:

- **age**: an individual’s age
  - Continuous. Min: 17, Max: 90
- **workclass**: an individual’s employment status
  - Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
- **fnlwgt**: final weight, the sampling weight of each entry
  - Continuous. Min: 12285, Max: 1484705
- **education**: an individual’s highest level of education achieved
  - 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:** an individual’s highest level of education achieved represented in numerical values
  - Ordinal numbers: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
- **marital-status** : an individual’s marital status
  - Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
- **occupation**: an individual’s occupation field
  - 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**: a representation of what the individual is relative to others.
  - Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.
- **race**: an individual’s race
  - White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
- **sex**: an individual’s sex
  - Female, Male.
- **capital-gain**: an individual’s capital gain
  - Continuous. Min: 0, Max: 99999
- **capital-loss**: an individual’s capital loss
  - Continuous. Min: 0, Max: 4356
- **hours-per-week**: an individual’s working hours per week
  - Continuous. Min: 1, Max: 99
- **native-country**: an individual’s native country
  - United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.
- **income**: label/target for if an individual makes over $50k or under
  - >50K, <=50K.

The dataset contains missing values and errors as stated on the UCI Machine Learning Repository website, which was handled appropriately and will be discussed in the next section of the report.

Methodology:

Jupyter Notebook running on Python 2.7 along with other libraries was used to produce the findings for the report. First of all, the dataset that was downloaded from UCI Machine Learning Repository was imported into our Jupyter Notebook environment for pre-processing and was given a set of column names as the header.

We began pre-processing the data by checking the existence of null and NaN values in the dataset but there were none. Following that, each column that contains numerical values was checked to ensure that there are no out of range values in the dataset and the range of each numerical column was recorded. Then, the checking of nominal/ordinal values in other columns was carried out by listing each distinct value and its count to ensure that there are no missing values and the values are free of typos and misplaced whitespace. Interestingly, we found a small number of missing values labelled as "?" in three of the columns, namely *'workclass'*, *'occupation'* and *'native-country'* and the missing values were replaced by the most frequent value in each of their respective columns as they did not occupy much of the columns.

We then moved on to data exploration and the first part of it was to generate visualisations for each of the selected columns in the dataset. Therefore, 11 visualisations were generated for the **Results section** of the report from different columns mainly to show each of their distribution across the dataset and they are listed as follows:

- ***Figure 1.1**: a scatter plot for 'age'
- **Figure 1.2**: a pie chart for 'workclass'
- ***Figure 1.3**: a pie chart for 'education'
- **Figure 1.4**: a bar chart for 'marital-status'
- **Figure 1.5**: a bar chart for 'occupation'
- **Figure 1.6**: a pie chart for 'relationship'
- **Figure 1.7**: a pie chart for 'race'
- **Figure 1.8**: a pie chart for 'sex'
- ***Figure 1.9**: a density plot for 'hours-per-week'
- **Figure 1.10**: a pie chart for 'native-country'
- ***Figure 1.11**: a bar chart for 'income'

*** Visualisations selected to be included in the report.**

Then for the second part of data exploration, we have also generated 11 visualisations for the **Results section** of the report but each visualisation was generated from a pair of attributes to show their relationship while also addressing a plausible hypothesis for the data concerned and they are the following:

- ***Figure 2.1**: a bar chart for 'hours-per-week' vs 'income'
- **Figure 2.2**: a bar chart for 'workclass' vs 'income'
- ***Figure 2.3**: a bar chart for 'age group' vs 'income ['age' was grouped appropriately]
- ***Figure 2.4**: a box plot for 'age' vs 'income'
- **Figure 2.5**: a box plot for 'race' vs 'education-num'
- ***Figure 2.6**: a box plot for 'education-num' vs 'income'
- **Figure 2.7**: a bar chart for 'sex' vs 'income'
- **Figure 2.8**: a bar chart for 'marital-status' vs income
- **Figure 2.9**: a bar chart for 'race' vs 'income'
- **Figure 2.10**: a box plot for 'workclass' vs 'hours-per-week'
- **Figure 2.11**: a bar chart for 'occupation' vs 'income'

Scikit-Learn library was used for the data modelling section and we have chosen to model the data by treating it as a Classification task using two methods which are K-nearest Neighbour Classifier and Decision Tree Classifier to predict whether an individual earns over $50k per year based on their attributes. Initially, we labelled the nominal values that were in String format and replaced them with integers numbered sequentially as the function loop encounters each distinct value to make the

dataset compatible with data modelling. After that, 3 classification tasks were carried out but each with different fractions of data used for training and testing as part of the requirements.

We started off by splitting the data accordingly to the requirements:

- **Task 1**: 50% of the data for training, the other 50% for testing
- **Task 2**: 60% of the data for training, the other 40% for testing
- **Task 3**: 80% of the data for training, the other 20% for testing

For each task, we first had 4 subtasks and each of them trained the data using K-nearest Neighbour Classifier but each subtask used a different set of parameters for K-nearest Neighbour Classifier.

The subtasks are as follows:

- **Subtask 1.1**: k = 3, (default) weights='minkowski'
- **Subtask 1.2**: k = 5, (default) weights='minkowski'
- **Subtask 1.3**: k = 5, weights='distance', p=1
- **Subtask 1.4**: k = 5, weights='distance', (default) p=2

Each subtask generated a classification report and we compared all 4 reports to select the best set of parameters that provided the most favourable scores. In each classification report, *'precision'*, *'recall'* and *'f1-score'* scores were provided for each label [0 = <=50k, 1 = >=50k], micro average, macro average and weighted average. From the scores, the best set of parameters was selected to generate a confusion matrix and perform k-fold Cross-validation [k = 10] for testing purposes. For each 10-fold Cross-validation, a testing accuracy score and an error rate were returned for our analysis. Observations from these test accuracy scores can tell us whether the distribution of this sample is stable. The smaller the difference between these test accuracy scores, the more stable the statistic sample is, which means we can use it for our further research.

Then when we were done with K-nearest Neighbour Classifier, we moved on to the second part of each task, which was training the data using Decision Tree Classifier. But before that, the tuning of parameters was required again. Therefore, we had 4 subtasks for tuning purposes and just like before, it was for us to select the best set of parameters:

The subtasks are as follows:

- **Subtask 2.1:** default parameters
- **Subtask 2.2:** criterion='entropy'
- **Subtask 2.3:** criterion='entropy', max_depth=12
- **Subtask 2.4:** criterion='entropy', max_depth=12, min_samples_split=6

Like what we did with K-nearest Neighbour Classifier, we selected the best set of parameters that produced the best scores. After that, a confusion matrix was generated along with a classification report. Following that, a 10-fold Cross-validation was performed to obtain testing accuracy scores and an error rate for further evaluations and analysis.

Moving on, we then compared the scores and error rates from using both classifiers in the following sections of the report to provide a recommendation for choosing the best classification method for this specific dataset.

Finally, we have 2 methods for running feature selection which is hill climbing and genetic algorithm respectively.

Data Exploration Results:

Data exploration for each selected column:

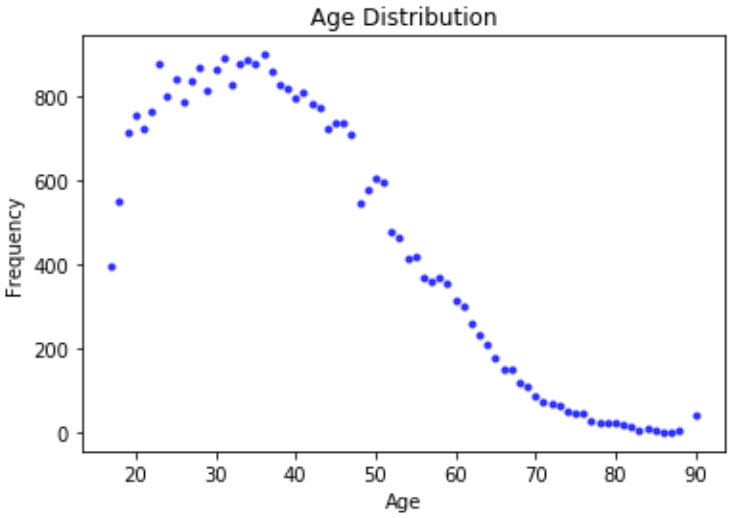

Figure 1.1: a scatter plot for ‘age’

Figure 1.1 shows that the individuals in our dataset are mostly people in adulthood ranging from the age from 17 - 60.

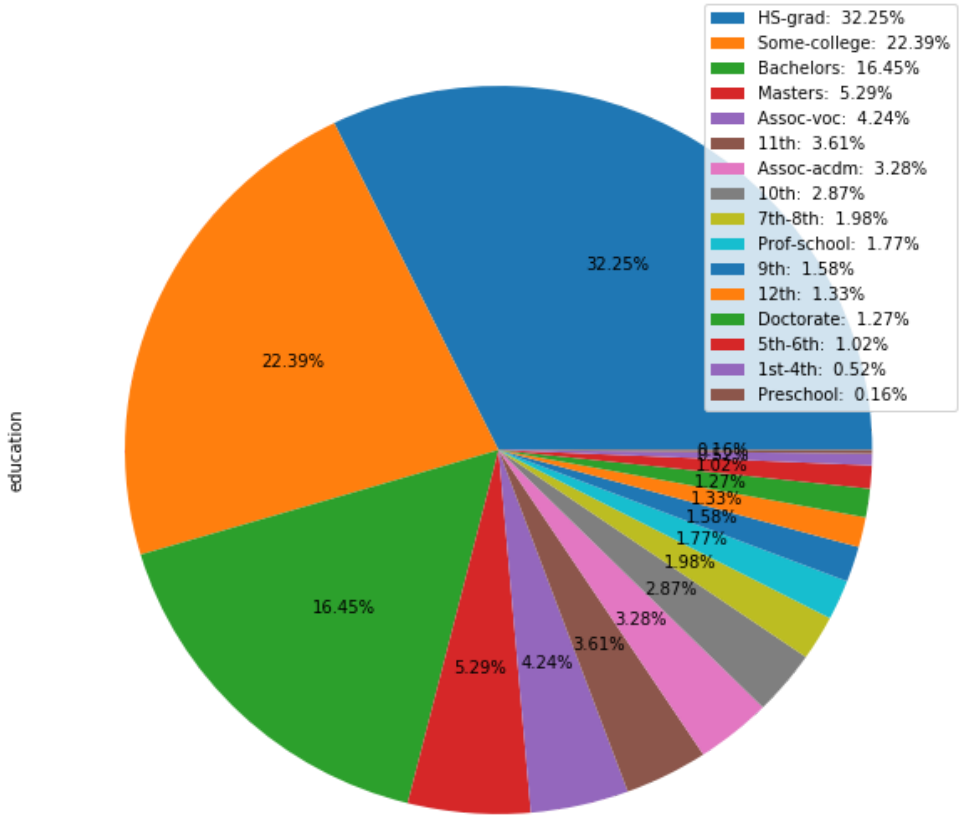

Figure 1.3: a pie chart for ‘education’

In terms of highest education achieved, Figure 1.3 shows that majority of the individuals in the dataset are high-school, college or bachelor graduates, which account for 32.25%, 22.39% and 16.45% respectively.

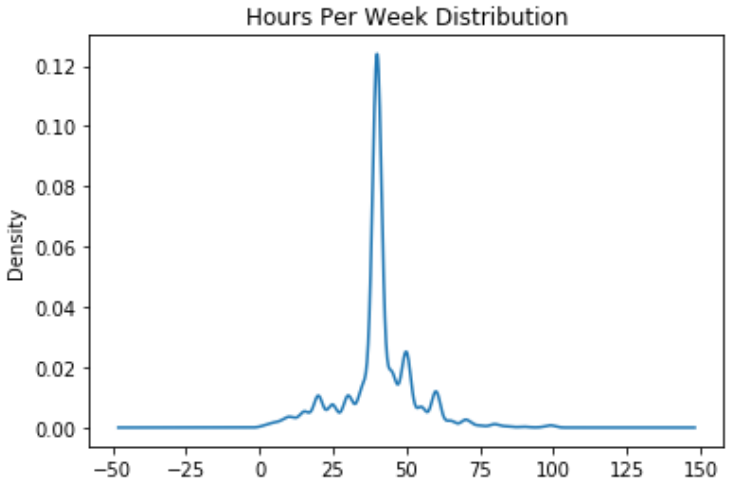

Figure 1.9: a density plot for 'hours-per-week'

Moving on, Figure 1.9 shows the distribution of the individuals' working hours per week across the dataset and here we notice that its peak is at 40 hours, meaning that most individuals work 40 hours per week.

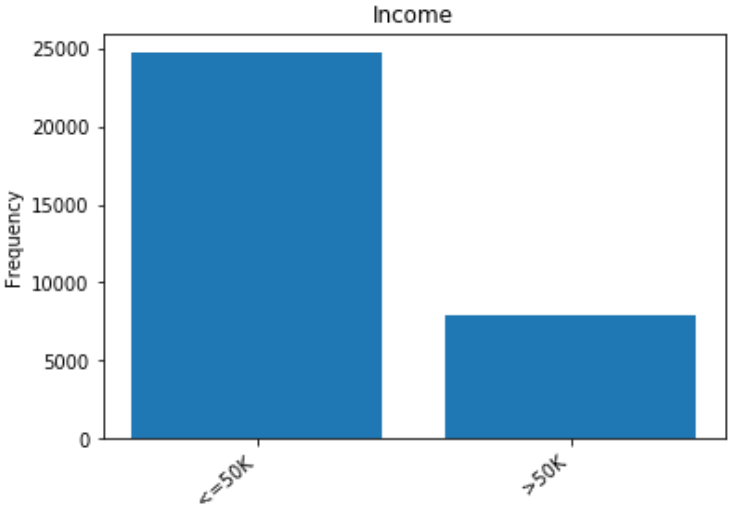

Figure 1.11: a bar chart for 'income'

Last but not least, Figure 1.11 shows that most of the individuals in the dataset earn less than $50k.

Data exploration for pairs of selected columns:

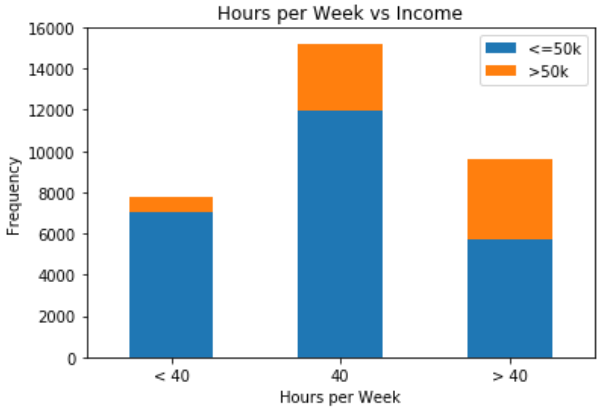

Figure 2.1: a bar chart for 'hours-per-week' vs 'income'

First and foremost, we would like to explore the relationship between working hours per week and income. We hypothesise that the longer the working hours per week, the greater one's income would be. Then, we generated a bar chart to visualise individuals' income frequency in terms of less than 40 hours per week, equal to 40 hours per week and more than 40 hours per week. As shown in Figure 2.1, we observe that the greater the working hours per week, the higher the fraction of the individuals earning more than $50k, hence matching our hypothesis.

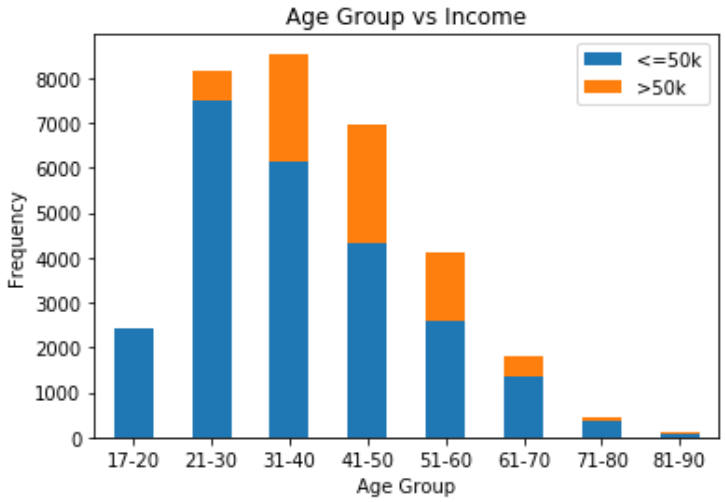

Figure 2.3: a bar chart for 'age group' vs 'income

Moving on, we hypothesise that as the older the individual is, the greater their income. In Figure 2.3, we have divided the age of each individual into appropriate age groups to provide a clearer visualisation on how an individual's age affects their income. Here we observe a very interesting trend. Starting from the age group of 17-20, everyone in that particular age group earns less than $50k per year, then more and more people start to earn more than $50k annually as the age number goes higher until the age group of 51-60 where the number of individuals earning more than $50k per year starts going down. We suspect that the rise of income as one grows older could be due to job promotions as one works longer at a company or could be due to accumulated experience that has led to higher pay, then when they get older, they retire. With that being said, our hypothesis does match the results to a certain extent.

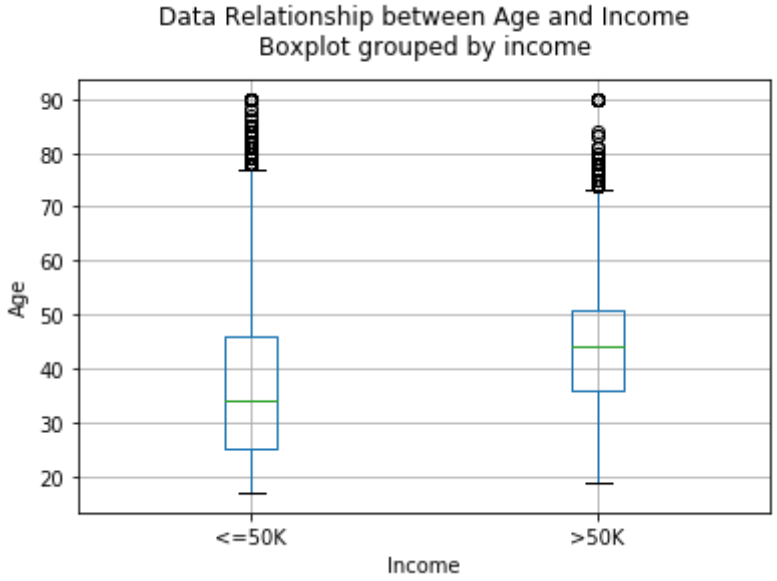

Figure 2.4: a box plot for 'age' vs 'income'

Relating to our previous hypothesis that was about age and income, Figure 2.4 shows the income distribution of individual in different age groups. In the plot, we see that the majority of individuals who earn less than $50k per year are people in their 20s to 40s while individuals who earn more than $50k per year are mostly people who are in their 30s to early 50s. Like what we have seen in Figure 2.3m this means that people generally earn more as they grow older until they are in their 50s. Again, We suspect that the rise of income as one grows older could be due to job promotions as one works longer at a company or could be due to accumulated experience that has led to higher pay, then when they get older, they retire.

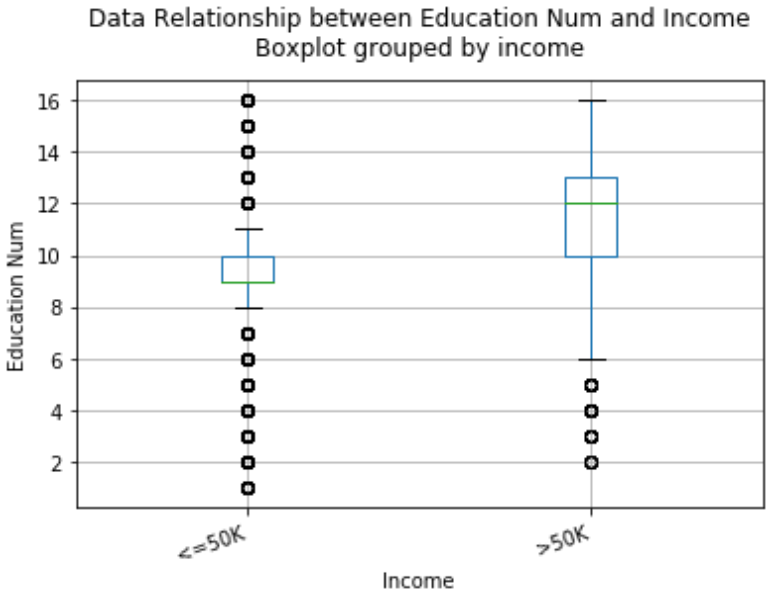

Figure 2.6: a box plot for 'education-num' vs 'income'

Finally, we hypothesise that the higher the highest level of education achieved by one, the greater their income would be. Figure 2.6 explores the relationship between education numbers how it affects an individual's income. As we can see, it has been shown that the majority of the individuals who achieved education numbers from 10 to 13 tend to earn more than people who achieved education numbers between 9 to 10. Therefore, it can be said that the outcome matches our hypothesis.

Data Modeling Analysis:

Task 1: Splitting Data: 50% for training and 50% for testing

Task 1.1: Comparing Results Between Different Parameters for K-Nearest-Neighbor Classifier

For the comparison, we were interested in using the set of parameters that provided the highest weighted average of f1-score. Then, we compared with different k values and found that when the k=5 provided a better f1-score. Next, we changed the value of parameter *'weights'* to *'distance'*, and it gave a better accuracy score than using 'minkowski'. After that, we modified the value of *'p*' to 1 as the default was 2 and a better result was given. Therefore, we conclude with the best parameter combination of K-nearest-neighbor classifier in this Task 1 dataset is *k=5, weights='distance' and p=1.* Here is the classification report from using the best set of parameters:

	precision	recall	f1-score	support
0	0.82	0.90	0.86	12323
1	0.54	0.37	0.44	3958
micro avg	0.77	0.77	0.77	16281
macro avg	0.68	0.64	0.65	16281
weighted avg	0.75	0.77	0.76	16281

Figure 3.1: KNN with k=5, weights='distance', p=1

Task 1.2: Selecting *k=5, weights='distance', p=1* for K-Nearest-Neighbor Classifier to run 10-cross-validation Tests

Confusion Matrix Report:

[[11098 1225]
[2492 1466]]

Calculating error rate from 10-Cross-Validation Scores:

Error Rate: 0.2192

Task 1.3: Comparing Results Between Different Parameters for Decision Tree Classifier

For the comparison, we were interested in using the set of parameters that provided the highest weighted average of f1-score. Then, we compared with different values of the '*criterion'* parameter and using *'entropy'* as a value provided a better result. Secondly, we tested with different values *'max_depth'*. When max_depth=12, the f1-score produced the highest score achieved. After that, we modified with the values for *'min_samples_split'* to a different value and the best result was produced when the *min_samples_split*=6. So, we conclude with the best parameter combination of Decision Tree classifier for this dataset for Task 1 is *criterion="entropy", max_depth=12, min_samples_split=6*.
Here is the classification report from using the best set of parameters:

	precision	recall	f1-score	support
0	0.89	0.93	0.91	2452
1	0.74	0.63	0.68	804
micro avg	0.85	0.85	0.85	3256
macro avg	0.81	0.78	0.79	3256
weighted avg	0.85	0.85	0.85	3256

Figure 3.2: decision tree with criterion="entropy", max_depth=12,min_samples_split=6

Task 1.4: Selecting *criterion="entropy", max_depth=12, min_samples_split=6* for Decision Tree Classifier to run 10-cross-validation Tests

Confusion Matrix Report:

[[2272 180]
[293 511]]

Calculating error rate from 10-Cross-Validation Scores:

Error Rate: 0.1439

Task 2: Splitting Data: 60% for training and 40% for testing

Task 2.1: Comparing Results Between Different Parameters for K-Nearest-Neighbor Classifier

For the comparison, we were interested in using the set of parameters that provided the highest weighted average of f1-score. The comparison is similar to Task 1 and we got the best result by using the parameter combination of K-nearest-neighbor classifier for this dataset for Task 2 is *k=5, weights='distance' and p=1*.
Here is the classification report from using the best set of parameters:

	precision	recall	f1-score	support
0	0.82	0.90	0.86	9873
1	0.55	0.37	0.45	3152
micro avg	0.78	0.78	0.78	13025
macro avg	0.69	0.64	0.65	13025
weighted avg	0.75	0.78	0.76	13025

Figure 3.3: KNN with k=5, weights='distance',p=1

Task 2.2: Selecting *k=5, weights='distance', p=1* for K-Nearest-Neighbor Classifier to run 10-cross-validation Tests

Confusion Matrix Report:

[[8928 945]
[1975 1177]]

Calculating error rate from 10-Cross-Validation Scores:

Error Rate: 0.2192

Task 2.3: Comparing Results Between Different Parameters for Decision Tree Classifier

For the comparison, we were interested in using the set of parameters that provided the highest weighted average of f1-score. The comparison is similar to Task 1 and we got the best result by using the parameter combination of Decision Tree classifier for this dataset for Task 2 is *criterion="entropy", max_depth=12, min_samples_split=6*.

Here is the classification report from using the best set of parameters:

	precision	recall	f1-score	support
0	0.88	0.93	0.90	2452
1	0.74	0.63	0.68	804
micro avg	0.85	0.85	0.85	3256
macro avg	0.81	0.78	0.79	3256
weighted avg	0.85	0.85	0.85	3256

Figure 3.4: decision tree with criterion="entropy", max_depth=12,min_samples_split=6

Task 2.4: Selecting *criterion="entropy", max_depth=12, min_samples_split=6* for Decision Tree Classifier to run 10-cross-validation Tests

Confusion Matrix Report:

[[2269 183]
[296 508]]

Calculating error rate from 10-Cross-Validation Scores:

Error Rate: 0.1440

Task 3: Splitting Data: 80% for training and 20% for testing

Task 3.1: Comparing Results Between Different Parameters for K-Nearest-Neighbor Classifier

For the comparison, we were interested in using the set of parameters that provided the highest weighted average of f1-score. The compare is similar to Task 1 and we got the best result by using the parameter combination of K-nearest-neighbor classifier for this dataset for Task 3 is k=*5, weights='distance' and p=1*.

Here is the classification report from using the best set of parameters:

	precision	recall	f1-score	support
0	0.82	0.90	0.86	4918
1	0.55	0.39	0.45	1595
micro avg	0.77	0.77	0.77	6513
macro avg	0.68	0.64	0.66	6513
weighted avg	0.75	0.77	0.76	6513

Figure 3.5: KNN with k=5, weights='distance',p=1

Task 3.2: Selecting k=5, weights='distance', p=1 for K-Nearest-Neighbor Classifier to run 10-cross-validation Tests

Confusion Matrix Report:

[[4417 501]
[980 615]]

Calculating error rate from 10-Cross-Validation Scores:

Error Rate: 0.2192

Task 3.3: Comparing Results Between Different Parameters for Decision Tree Classifier

For the comparison, we were interested in using the set of parameters that provided the highest weighted average of f1-score. The compare is similar to Task 1 and we got the best result by using the parameter combination of Decision Tree classifier for this dataset for Task 3 is *criterion="entropy", max_depth=12, min_samples_split=6*.

Here is the classification report from using the best set of parameters:

	precision	recall	f1-score	support
0	0.88	0.93	0.90	2452
1	0.74	0.63	0.68	804
micro avg	0.85	0.85	0.85	3256
macro avg	0.81	0.78	0.79	3256
weighted avg	0.85	0.85	0.85	3256

Figure 3.6: decision tree with criterion="entropy", max_depth=12,min_samples_split=6

Task 3.4: Selecting *criterion="entropy", max_depth=12, min_samples_split=6* for Decision Tree Classifier to run 10-cross-validation Tests

Confusion Matrix Report:

[[2268 184]
[295 509]]

Calculating error rate from 10-Cross-Validation Scores:

Error Rate: 0.1440

Discussions of Data Modelling Results:

Classification Methods	f1-score Weighted Average Score	Error Rate (10-Cross-Validation)
K-Nearest-Neighbour Classifier (*k=5, weights='distance', p=1)*	0.76	0.2192
Decision Tree Classifier (*criterion="entropy", max_depth=12, min_samples_split=6)*	**0.85**	**0.1439**

Table 1: Comparison of Results from using different classifiers for 50% training / 50% testing

Classification Methods	f1-score Weighted Average Score	Error Rate (10-Cross-Validation)
K-Nearest-Neighbour Classifier (*k=5, weights='distance', p=1)*	0.76	0.2192
Decision Tree Classifier (*criterion="entropy", max_depth=12, min_samples_split=6)*	**0.85**	**0.1440**

Table 2: Comparison of Results from using different classifiers for 60% training / 40% testing

Classification Methods	f1-score Weighted Average Score	Error Rate (10-Cross-Validation)
K-Nearest-Neighbour Classifier (*k=5, weights='distance', p=1)*	0.76	0.2192
Decision Tree Classifier (*criterion="entropy", max_depth=12, min_samples_split=6)*	**0.85**	**0.1440**

Table 3: Comparison of Results from using different classifiers for 80% training / 20% testing

For data modeling, we recommend using *Decision Tree Classifier* with the following parameters:
DecisionTreeClassifier(criterion="entropy", max_depth=12, min_samples_split=6)

Decision Tree Classifier provides a **higher *f1-score Weighted Average Score*** and a **lower *Error Rate*** than using *K-Nearest-Neighbour Classifier* as seen in the tables. By using 10-cross-validation, we see that the score from each test is very similar, which means the distribution of this sample is quite stable. The average error rate from these 10 validations ranging from 50% train, 50% test to 80% train, 20% test is between 0.1349 and 0.1440, which means the sample of this data is very consistent. In addition, we have noticed something interesting and that is the f1-score Weighted Average Scores and error rates from each task do not differ much from one task to another. We suspect that this is due to a large number of instances in this data sample and they have a very high level of consistency.

Feature selection:

While performing feature selection with the hill climbing method did not give us a very reasonable feature subset, we tried to design a feature selection model based on Genetic Algorithm (GA). After running adult data with feature selection model based on GA with 30 population and 100 generations by using Decision tree classifier with the best set of parameters set as the fitness function, we got the most relevant feature subset [*education-num, occupation, relationship, capital-gain, capital-loss*] with an accuracy score of 0.8582, which is higher when compared to the accuracy score with all features which is 0.8552. By removing the redundant features, feature selection model did improve the accuracy of dataset.

Conclusion:

Experiments done have evidently proved that by using Decision Tree Classifier, we are able to obtain a higher accuracy score and a lower error rate than using K-Nearest-Neighbor Classifier for this particular dataset. The similarity of results in different split percentages indicates this data sample has a very high level of consistency.

Due to space constraint, we are unable to include the Decision Tree graphs generated for each task, but the links to the graphs are available below for your reference:

Decision Tree Graph for Task 1: Splitting Data into 50% for Training / 50 % for Testing
Decision Tree Graph for Task 2: Splitting Data into 60% for Training / 40 % for Testing
Decision Tree Graph for Task 3: Splitting Data into 80% for Training / 20 % for Testing

From the Decision Tree graphs, we can see that the top 5 relevant features include *education-num*, *capital-gain*, *relationship*, *age*, and *capital-loss*, which is very similar to the feature subset using feature selection model based on GA. So, in order to improve f1-score and reduce the error rate, we can remove the redundant features.

References:

[1] *Adult Data Set* on UCI Machine Repository: https://archive.ics.uci.edu/ml/datasets/adult