Citizens Income Prediction

Venkata Sriram Rachapoodi
ID: 01897282
Department of Computer Science
Kennedy College of Sciences
University of Massachusetts Lowell
venkatasriram_rachapoodi@student.uml.edu

Kamal Yeshodhar Shastry Gattu
ID: 02007505
Department of Computer Science
Kennedy College of Sciences
University of Massachusetts Lowell
kamalyeshodharshastry_gattu@student.uml.edu

Aditya Santhoshkumar Karnam
ID: 02003014
Department of Computer Science
Kennedy College of Sciences
University of Massachusetts Lowell
aditya_karnam@student.uml.edu

***Abstract* - The classification problem in the field of Data Science is not new and there are many Machine Learning algorithms available to solve classification problems depending on the situation. But there is always a debate on what algorithm to choose or which is the best Machine Learning algorithm for the classification problem because of the diversity of the available algorithms.**

We make use of the Adult Dataset to predict the income of citizens and classify people into two categories based on various dependent properties of a person collected during a Census. We use different Classification algorithms like Naïve-Bayes Classifier, Logistic Regression, Support Vector Machine, K Nearest Neighbors Classifier, Decision Tree, and Random Forest algorithms to perform this task. Furthermore, we compare the results obtained from the above approaches using evaluation metrics like Confusion Matrix, F1 score, Recall, Precision, and Accuracy and find which algorithm is more suitable for the current scenario to produce the most accurate prediction.

Keywords – Machine Learning, Classification, Naïve-Bayes Classifier, Logistic Regression, Support Vector Machine, K Nearest Neighbors Classifier, Decision Tree Classifier, Random Forest Classifier, Confusion Matrix, F1 score, Recall, Precision, Accuracy, Adult dataset

I. INTRODUCTION

It is essential for any government or an organization to have a rough idea about the benefit plans for the citizens. In recent years, the issue of economic disparity has been a major source of worry. Making the poor's lives better does not appear to be the main criterion for solving this problem. People in the United States feel that rising economic inequality is unacceptably high, and they desire a fair distribution of wealth in society. Some of those plans include the mode of living and how much income, citizens make in the country. This helps in devising plans for the people based on their requirements.

In this report, we estimate the income of citizens and classify whether they have income greater than 50,000$ per year or not, based on various dependent factors of a person. Then, we process the samples by removing some data inconsistencies and handling the missing values. Further, we study the features by correlating them and filtering them out. We visualize the key features in the dataset. Finally, We use algorithms like Logistic Regression, K Nearest Neighbours, Naïve-Bayes Classifier, Support Vector Machines, Decision Trees, and Random Forest to train the dataset to perform classification.

We use this problem as the basis for the problem of comparison of how different machine learning models perform on the same dataset. We train multiple systems on six different algorithms and compare the models using some evaluation metrics. We analyze the performance of each algorithm and come to a conclusion on which algorithm works best for the current dataset.

II. RELATED WORK

- Junda Chen[1] has implemented logistic regression, random forest, and LASSO algorithms on the Adult dataset to analyse the potential factors contributing to income bias.
- Navoneel and Sanket [2] analyzed the adult data set and used Ensemble learning with Boosting Algorithm known as the Gradient boosting classifier
- Jamal Khanam[3] has analyzed how different machine learning algorithms like Decision Tree, Random Forest, K-Nearest Neighbour(With K = 7), Support Vector Machine, and Logistic regression work on the same dataset(Diabetes Dataset).
- Boran Sekergolu[4] has implemented three different machine learning algorithms namely Backpropagation Neural Network, Radial Bias Function Neural Network, and Support Vector Machine to analyze the best algorithm for classification problems.
- Anwar Al HUssan [5] has compared the performance of different Machine Learning classifiers namely Logistic Regression, Decision Tree, Naïve Bayes, K-Nearest Neighbour, Support Vector Machine, and Random Forest, on Hepatitis disease dataset and Heart Disease to identify the best algorithm for classification problem.

III. METHODOLOGY

A. The Dataset:

1) Dataset Description:

The data we use for this project is taken from the *Adult Income* Dataset which is published by the Machine Learning Repository at the University of California, Irvine (UCI). This dataset comprised 48,842 samples from 42 countries, each with 14 characteristics. It has eight categorical and six continuous variables that comprise information on age,

education, nationality, marital status, relationship status, occupation, job classification, gender, race, weekly working hours, capital loss, and capital gain. The income level is the binomial label in the data set, which predicts whether a person makes more than $50,000 per year or not, based on a set of variables.

TABLE I. DATASET DESCRIPTION

Features	*Description*
age	(continuous, positive integer) The age of person.
work-class	(categorical, 9 distinct values) Employment status of person
fnlwgt	(continuous, positive integer) Number of people represented by this row.
education-num	(categorical, 13 distinct values) The education level, in numeric form.
education	(categorical, 13 distinct values) The education level of person.
marital-status	(categorical, 7 distinct values) Marital status of a person.
occupation	(categorical, 15 distinct values) Occupation of Person.
relationship	(categorical, 6 distinct values) Relationship of person in terms of the family.
race	(categorical, 5 distinct values) Race of the person.
sex	(boolean) Gender of person
capital-gain	(continuous) Gain of capital in dollars.
capital-loss	(continuous) Loss of capital in dollars.
hours-per-week	(continuous, positive integer) Working hours per week.
native-country	(categorical, 41 distinct values) Native Country of person.
income	(boolean) income of person per year given in brackets >50K & <= 50K

2) Dataset Compilation:

The dataset provided in the UCI repository contains two different .data files – adult.data and adult.test. For our work, we have combined both datasets to make a large new dataset of 48842 data entries. This is done so that the preprocessing and feature selection in the data will be easier.

B. Data Preprocessing:

1) Handling Missing Values:

When an exploratory data analysis is performed, we have found that there are few entries with a value as '?' in three columns namely *work-class*, *occupation*, and *native-country*. We considered these values to be missing value entries and when checked there are about 5882 data entries with missing values. All of these are of type string and these features have minimal effect on the outcome of *income*. Hence, we have removed the data entries having missing values.

2) Removing inconsistencies in data:

When the data in the *income* is analyzed, we found that instead of having two classes i.e., >50K and <=50K, there are four classes >50K, >50K., <=50K. and <=50K. It is obvious that the data entered is having discrepancies – having an extra '.' in the data. Hence, we modify the data by removing the abovementioned inconsistency.

3) Encoding Categorical Data:

As mentioned in the dataset description, the data of most of the Categorical features are non-numeric, we have Label-encoded the data where all features are encoded in alphabetical order starting from 0.

C. Feature Study and Selection:

1) Filtering Features based on Dataset Analysis:

When the dataset is analyzed, we found there are 14 features in the dataset on which the outcome of the income class is said to be directly dependent. But when analyzed more thoroughly we found that few of the features are not relevant to the problem. The feature *fnlgwt* is the id referencing the survey through which the data entry is collected. This does not affect the training model. Hence, we have removed it from the feature set. And few features are indirectly pointing out a few other features. Hence, using these features will just increase the learning time but does not affect the outcome of the models. We found that *education* gives the same information as that of *education-num*, and it is easier to use *education-num* for model training as this is a numeric feature. Hence, we have removed *education* from the feature set. Similarly, *relationship* and *marital-status give* almost the same information. Hence, we have decided to remove one feature – *relationship*. And from the literature survey for the dataset is done, we found that capital-gain and capital-loss are gains and loss a person had on investing and concluded irrelevant to the *income* of a person. Therefore, this feature is removed from the data.

2) Feature-to-Feature Correlation Analysis:

After removing features based on literature analysis, we performed Feature-to-Feature and Feature-to-Label Correlation analysis on the data to get a picture of which features are affecting the outcome of income the least – to remove the least useful features. We found *marital-status, workclass, occupation, race, native-country* are having a correlation of -0.19, 0.016, 0.05, 0.07, 0.02 respectively with *income*. Therefore, we have not considered the above features for the model training. The final Feature set now consists of *age, education-num, sex, hours-per-week* for

predicting *income*. The Correlation Matrix in the form of a heatmap between the final features is shown in Fig 1.

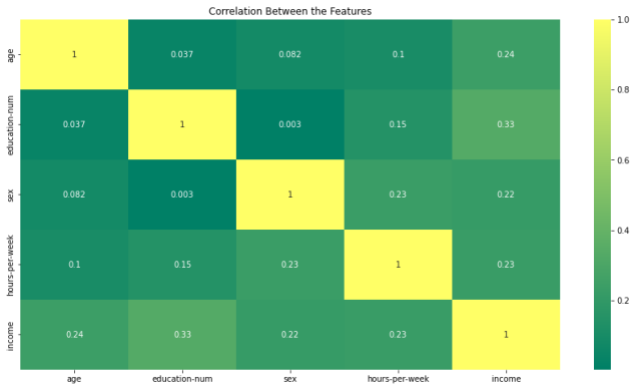

Fig 1. Heat-Map showing Feature-to-Feature and Feature-to-Label's Pearson Correlation Coefficients

E. Data Visualization:

All continuous features were visualized using Box and Whisker Plots to readily comprehend the measures of their central inclinations, as shown in Fig 2, Fig 3, Fig 4, Fig 5, Fig 6

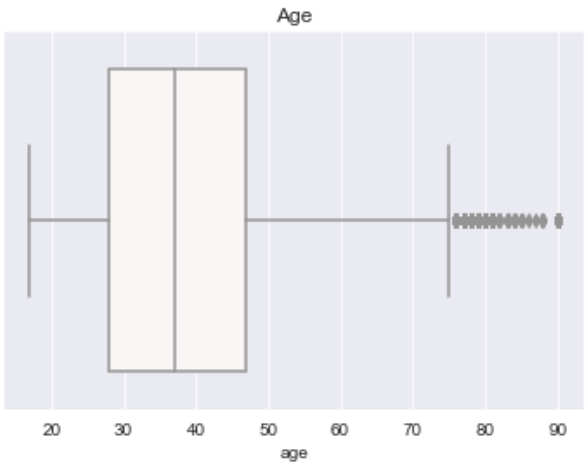

Fig 2. Box and Whisker plot for age

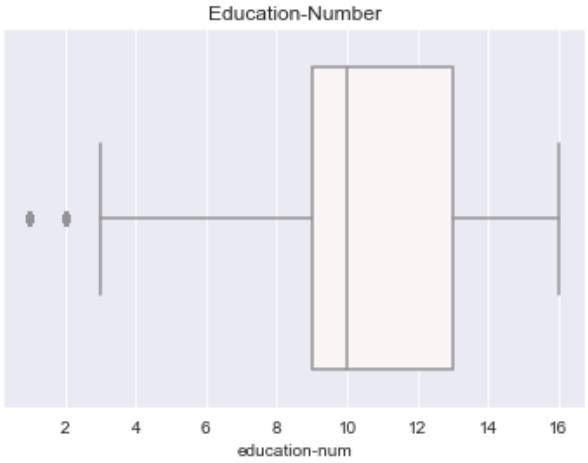

Fig3. Box and Whisker plot for education-num

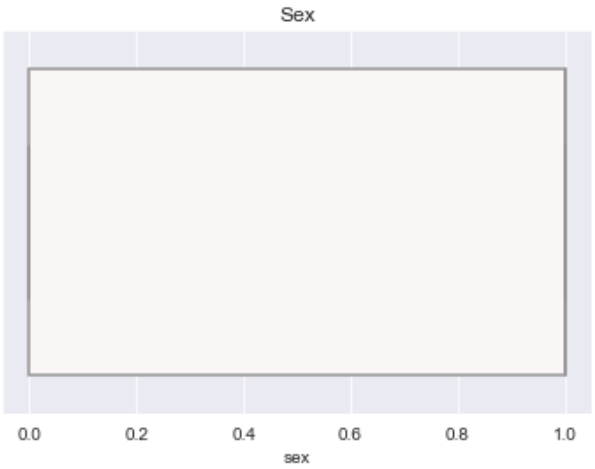

Fig 4. Box and Whisker plot for sex

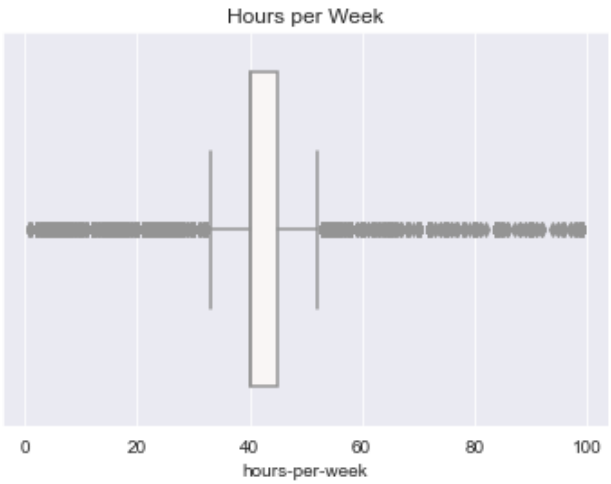

Fig 5. Box and Whisker plot for hours-per-week

D. Preparing Train and Test Datasets:

The entire dataset is shuffled consistently, ensuring that all the distinct attribute categories were included in the Training Set and Testing Set. The dataset has now been divided into training and testing sets with seventy percent of the data being made available for training, while the remaining thirty percent is used for testing.

E. Training the Models

1. Logistic Regression:

Logistic Regression is a classifier that works based on the learning function of P(Y/X). In this, a relationship is established between features and class which is used to detect the class of test data. We create a Logistic Regression model to train the system by minimizing the cost/loss function by using Gradient Descent. Our Model mainly consists of a Sigmoid Function that takes features and weights as input and gives 0 or 1 as the outputs.

$$Sigmoid\ Function\ \hat{y} = g(z)\frac{1}{1+e^{-z}}$$

After the model makes a prediction, we evaluate the result using the loss function. In this process, we calculate derivatives of the loss function concerning their weights. Derivatives can tell us which way to modify the weight and by how much to reduce the model's loss.

$$Loss\ Function = -\frac{1}{m}\sum_{i=1}^{m} yi.\log(\hat{y}i) + (1-yi).\log(1-\hat{y}i)$$

And We update weight until a local minimum is found and the model doesn't improve any further. Now we multiply each derived value with the learning parameter in the gradient descent function and subtract it from the weight. Now we fit the model based on the results of gradient descent. We use 1000 iterations with a learning rate of 0.5 to fit the model. We use this model to predict the income of persons in the testing data.

2. K Nearest Neighbors Classifier:

K Nearest Neighbor Classifier uses the distance between the data points to predict the class. In our model, we use Euclidean Distance to find out the distance from the data points to the target point and then we arrange the data points in the increasing order of their distances.

$$Euclidian\ Distance = \sqrt{(q1-p1)^2 + (q2-p2)^2}$$

Then, we count the n number of nearest data points from the target point. Then the number of data points in each class is counted and the class which has a maximum number of data points is assigned to the target class. We have tested the model with multiple values of the number of nearest neighbors and found around k value 20, we are getting an optimal result of accuracy. We train the model with the number of nearest neighbors as 20 and train the model. We use this model to predict the income of persons in the testing data.

3. Naïve Bayes Classifier:

A naïve Bayes classifier uses the Bayes Theorem of probability to define the probability of a data value to be mapped to a particular class. Bayes Theorem states if there are two events A and B then the probability of A given B is the probability of B given A time probability of A divided by the probability of B.

$$P(A|B) = P(B|A) * P(A) / P(B)$$

In this project, we calculate the probability of features in the training set to be part of classes in the target. All the features are independent of each other, and we are interested only in the class of outcome. Since all the probabilities lie between 0 and 1, multiplying them will result in a small value and we might run into overflow problems, therefore, to prevent this we apply a log function. Then multiplication becomes an addition. Hence, the class can be found by

$$y = argmax_y\ log(P(x_1|y)) + log(P(x_2|y)) + \ldots . + log(P(x_n|y)) + log(P(y))$$

So, to calculate the posterior probability(P(y|X)) i.e., is to predict our class label we need to calculate the prior probability (P(y)) which is nothing but the frequency of each class in the training data set. for calculating the conditional probability i.e., $p(x_i|y)$ we use Gaussian distribution formula.

$$p(xi|y)\frac{1}{\sigma\sqrt{2\pi}}\ \exp\left(-\frac{1}{2}\frac{(x-\mu)2}{\sigma 2}\right)$$

We calculate prior probability, then calculate the conditional probability for each feature and add them all to get the posterior probability for that respective class label. Then finally we chose the class label which has the highest value posterior probability. We fit the model using the above method using all data entries in the training set and use this model to predict the income of persons in the testing data.

4. Support Vector Machine Classifier:

Support Vector Machines classify the data by locating a dividing line (or hyperplane) between two classes of data. SVM is an algorithm that takes data as input and, if possible, generates a line that separates the classes. For the current system, we use predefined methods in the scikit-learn library of python to train the system. In this project, we trained multiple models using different kernels of the Support Vector Classifier method defined in SVM. We train the system using 'linear', 'poly', 'rbf', 'sigmoid' kernels separately and calculate train and test scores. Then based on these scores we finalize the kernel which has the best train and test scores and train the SVM Model using that kernel. We found that the polynomial kernel is giving the best scores and we used the 'poly' kernel to train the model we use this model to predict the income of persons in the testing data.

5. Decision Tree Classifier:

Decision Trees uses the CART algorithm to perform the classification on a dataset. The tree starts from a particular feature and based on the value of the feature it decides which feature to be checked next and this goes on until all the features are used (or a specified number of features are used). The important step in the decision tree classifier is to decide which feature is to be selected in each step. For this,

we use the DecisionTreeClassifier method defined in scikit-learn of python. A decision tree can be trained, or the best split can be decided by either using the Gini Index or Entropy of Information Gain on the dataset. The DecisionTreeClassifier in scikit-learn takes two input arrays consisting of data features and target space and input and gives a complete decision tree on the features. In our project, we initially train multiple systems by using both the Gini index and entropy criteria and choosing different approaches for maximum features like 'auto', 'sqrt', and 'log2'. We calculate the train and test scores of each decision tree developed and choose the tree with the best scores as our final model. By following this method, we found that the tree built using the Gini index and log2 maximum features gives the best scores. This tree is used to train the model and used to predict the income of persons in test data.

6. Random Forest Classifier:

Random Forest Classifier uses the Ensembling process in which each tree in the ensemble is a decision tree built from a sample of data drawn with replacement from the training data. The selection of these data samples is random and the best split for these trees is found from all input features or a random set of features selected from features in max_features. Using Random Forests rectifies problems of decision trees like overfitting and high variance. In our model, we use RandomForestClassifier defined in the scikit-learn library of python. Similarly, as we did in Decision Tree, we build multiple models using Gini index and entropy criteria and choosing different approaches for maximum features like 'auto', 'sqrt', and 'log2'. And for all cases, we used a number of trees as a hundred which is the default parameter defined in the scikit-learn method. We calculate the train and test scores of each decision tree developed and choose the tree with the best scores as our final model. By following this method, we found that the model built using Gini index and sqrt maximum features are giving the best scores. This model is used to train the model and used to predict the income of persons in test data.

IV. RESULTS – COMPARISON OF MODELS

We have trained Logistic Regression, K Nearest Neighbors Classifier, Naïve-Bayes Classifier, Support Vector Machine Classifier, Decision Tree Classifier, and Random Forest Classifier algorithms according to the above procedures using the training dataset. We have applied the trained models to the features of data in the test dataset to predict the outcome i.e., the income bracket of the person. And then compared the resultant prediction to the corresponding target feature in the testing set to map how accurate the trained models are. To compare the models, we use a few model evaluation metrics defined in the scikit-learn library of python.

A. Accuracy Score

This function computes subset accuracy i.e., the set of classes predicted for a sample that exactly match the associated set of class in the testing set. We calculated Accuracy Scores for each algorithm and plotted a graph which is as shown in Fig 6. We observe that Naïve Bayes and Support Vector Machine has almost equal accuracy score.

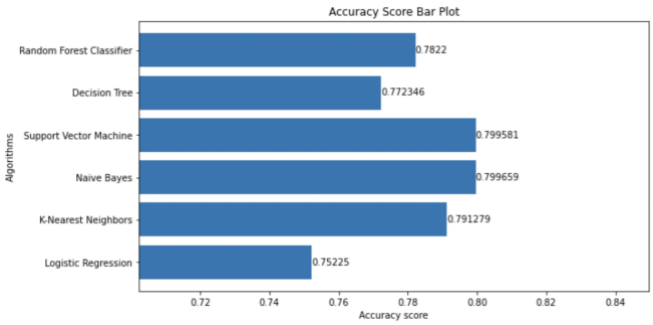

Fig 6. Accuracy Scores of Algorithms

B. Classification Report

The classification Report of a classification model gives a class-wise metrics value of the model. The Classification Reports of trained models are as follows

**********Logistic Regression**********

	precision	recall	f1-score	support
0	0.75	1.00	0.86	9695
1	0.00	0.00	0.00	3193
accuracy			0.75	12888
macro avg	0.38	0.50	0.43	12888
weighted avg	0.57	0.75	0.65	12888

Fig 7. Classification Report of Logistic Regression

**********K Nearest Neighbor**********

	precision	recall	f1-score	support
0	0.82	0.92	0.87	9695
1	0.62	0.40	0.49	3193
accuracy			0.79	12888
macro avg	0.72	0.66	0.68	12888
weighted avg	0.77	0.79	0.77	12888

Fig 8. Classification Report of K Nearest Neighbors

**********Naive Bayes**********

	precision	recall	f1-score	support
0	0.82	0.95	0.88	9695
1	0.68	0.36	0.47	3193
accuracy			0.80	12888
macro avg	0.75	0.65	0.67	12888
weighted avg	0.78	0.80	0.78	12888

Fig 9. Classification Report of Naïve Bayes

**********Support Vector Machine**********

	precision	recall	f1-score	support
0	0.81	0.95	0.88	9695
1	0.70	0.34	0.45	3193
accuracy			0.80	12888
macro avg	0.76	0.64	0.67	12888
weighted avg	0.78	0.80	0.77	12888

Fig 10. Classification Report of Support Vector Machine

**********Decision tree**********

	precision	recall	f1-score	support
0	0.82	0.90	0.86	9695
1	0.56	0.39	0.46	3193
accuracy			0.77	12888
macro avg	0.69	0.65	0.66	12888
weighted avg	0.75	0.77	0.76	12888

Fig 11. Classification Report of Decision Tree

**********Random Forest**********

	precision	recall	f1-score	support
0	0.83	0.90	0.86	9695
1	0.58	0.43	0.50	3193
accuracy			0.78	12888
macro avg	0.70	0.67	0.68	12888
weighted avg	0.77	0.78	0.77	12888

Fig 12. Classification Report of Random Forest

C. Confusion Matrix:

The Confusion Matrix of a Classification model gives the number of True Positives, False Positives, False Negatives, and True Negatives predicted by the model. We have plotted heatmaps for the confusion matrices of each model.

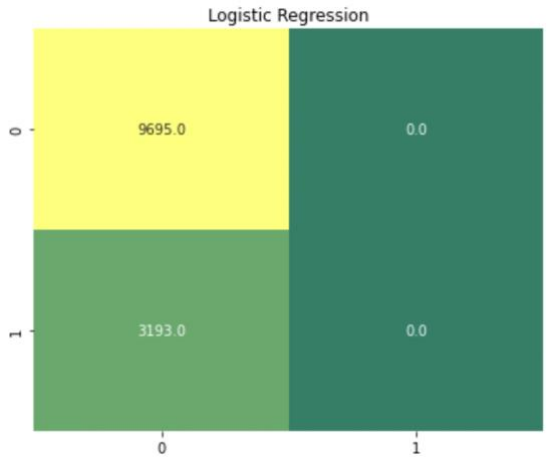

Fig 13. Confusion Matrix of Logistic Regression

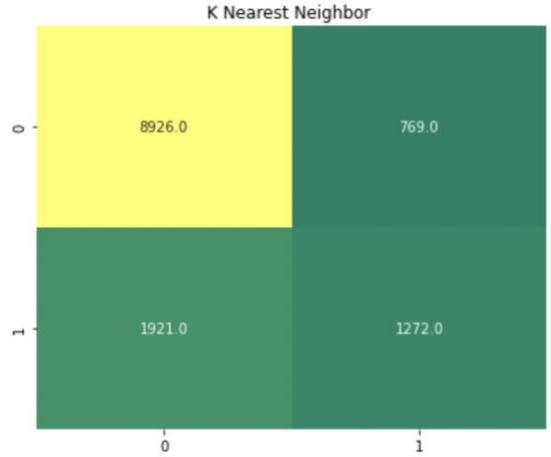

Fig 14. Confusion Matrix of K Nearest Neighbor

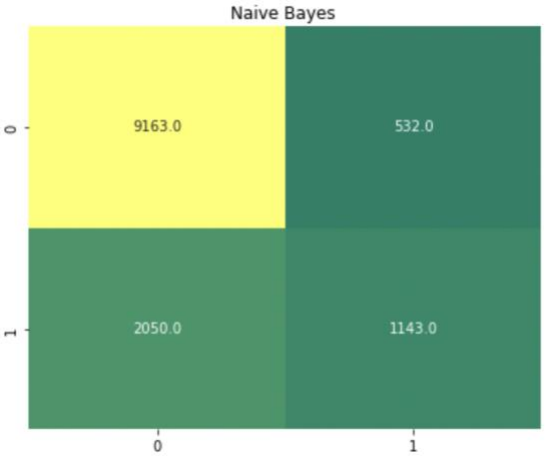

Fig 15. Confusion Matrix of Naïve Bayes

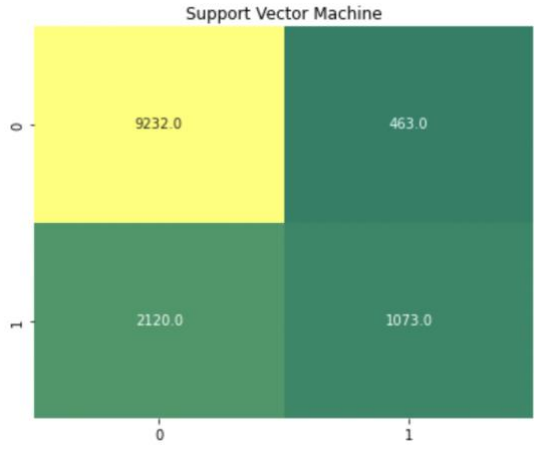

Fig 16. Confusion Matrix of Support Vector Machine

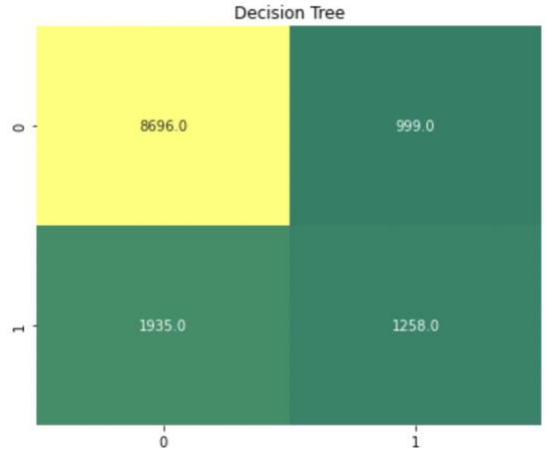

Fig 17. Confusion Matrix of Decision Tree

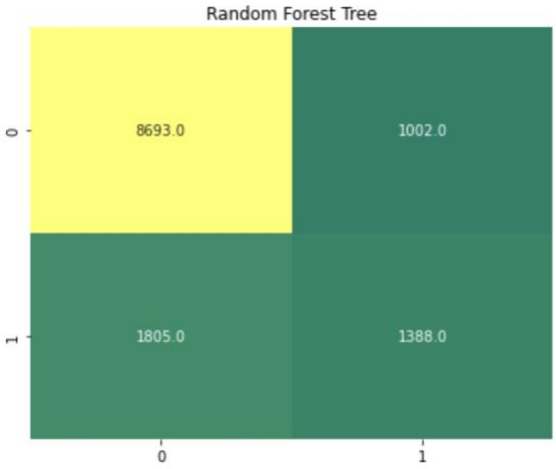

Fig 18. Confusion Matrix of Random Forest

D. Precision

The precision of a model is defined as the ratio between the True Positives and all the Positives. To calculate the precision, we first get the Classification Report of each algorithm and pull the macro average of precision of both classes as the precision. And we plot a graph of precisions shown in Fig 7. We observe that the Support Vector Machine has the best Precision Score.

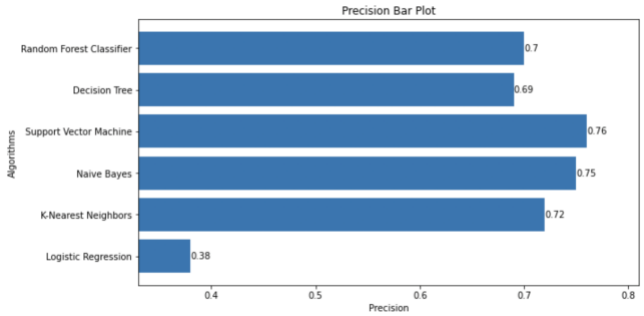

Fig 7. Precision of Algorithms

E. Recall

Recall of a model is defined as the measure of our model correctly identifying True Positives. To calculate the recall, we first get the Classification Report of each algorithm and pull the macro average of recall of both classes as the recall. And we plot a graph of recalls shown in Fig 8. We observe that Random Forest Classifier has the best Recall Score.

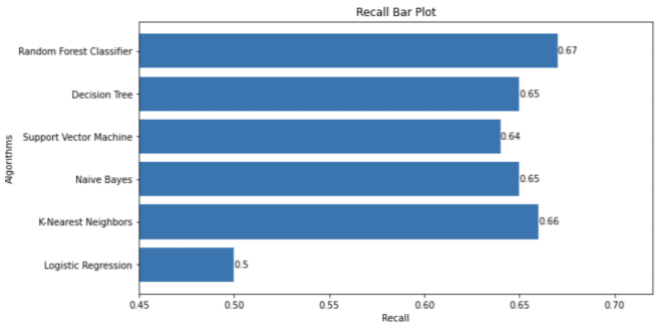

Fig 8. Recall of Algorithms

F. F1 Score

F1-score is the Harmonic mean of the Precision and Recall. F1-Score is a metric that indicates a good Precision and a good Recall value. To calculate the F1-Score, we first get the Classification Report of each algorithm and pull the macro average of F1-Scores of both classes as the F1-Score. And we plot a graph of F1-Scores shown in Fig 8. We observe that Random Forest Classifier and K Nearest Neighbor have the best F1-Score.

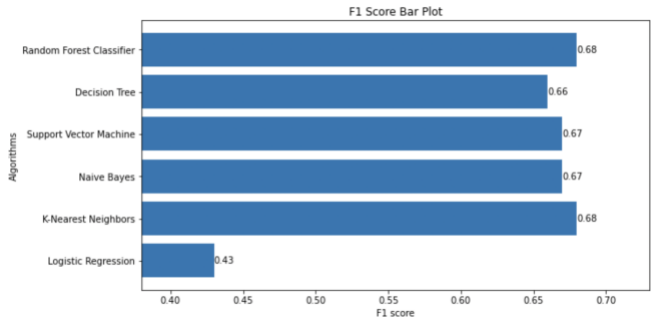

Fig 9. F1-Score of Algorithms

V. CONCLUSION

Observing the obtained results of the metric scores of all the trained algorithms we find that Naïve Bayes and Support Vector Machine has the best Accuracy Scores; Support Vector Machine have the best Precision on classes; Random Forest has better Recall than other and when analyzed the F1-Score all the algorithms except Logistic Regression have almost 66-68%. And Random Forest and K Nearest Neighbors have better F1-Score. Since the F1-Score is the Harmonic mean of Precision and Recall, we consider the F1-Score to be a more deciding factor in finding the most accurate model. And the data each class in the dataset is

imbalanced. Hence F1-Score is more feasible than Accuracy.

Hence, when comparing F1-Scores, Random Forest and K Nearest Neighbors have the best scores, but when both algorithms are compared, we find Precision is better for the K Nearest Neighbor. And Recall is better for Random Forest. But the precision difference is more for K Nearest Neighbor. Therefore, from all the above analyses, we conclude that K-Nearest Neighbor Classifier with 20 nearest neighbors is the best possible model for predicting a person's income.

REFERENCES

[1] Junda Chen: "Feature Significance Analysis of the US Adult Income Dataset", https://minds.wisconsin.edu/bitstream/handle/1793/82299/TR1869%20Junda%20Chen%203.pdf

[2] Navoneel Chakrabarty, Sanket Biswas : "A Statistical Approach to Adult Census Income Level Prediction" https://arxiv.org/pdf/1810.10076.pdf

[3] Khanam, J. & Foo, Simon Y.. (2021). A comparison of machine learning algorithms for diabetes prediction. Express, 7(4), 432-439. https://doi.org/10.1016/j.icte.2021.02.004

[4] Sekeroglu, B., Hasan, S.S., Abdullah, S.M. (2020). Comparison of Machine Learning Algorithms for Classification Problems. In: Arai, K., Kapoor, S. (eds) Advances in Computer Vision. CVC 2019. Advances in Intelligent Systems and Computing, vol 944. Springer, Cham. https://doi.org/10.1007/978-3-030-17798-0_39

[5] C. A. Ul Hassan, M. S. Khan and M. A. Shah, "Comparison of Machine Learning Algorithms in Data classification," 2018 24th International Conference on Automation and Computing (ICAC), 2018, pp. 1-6, doi: 10.23919/IConAC.2018.8748995.

[6] Ron Kohavi, "Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid", Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, 1996 http://robotics.stanford.edu/~ronnyk/nbtree.pdf

[7] https://archive.ics.uci.edu/ml/datasets/adult

[8] https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec

APPENDIX

Project Contribution:

TABLE 2. PROJECT CONTRIBUTIONS

Name	Roles
Kamal Yeshodhar Shastry Gattu	Data Preprocessing Logistic Regression Random Forest Model Comparison
Venkata Sriram Rachapoodi	Naïve Bayes Decision Tree Model Comparison
Aditya Santhoshkumar Karnam	K Nearest Neighbors Support Vector Machine Model Comparison