Project 1 Report

Machine Learning for Data Science

Arman Durmus, Javad Kasravi, Gandlur Phani shankar Bharadwaj

December 2021

Exploratory data analysis

Adult dataset has 14 features and one target which is *class_label*. These feature are *age*, *work_class*, *fnlwgt*, *education*, *education_num*, *marital_status*, *occupation*, *relationship race*, *sex*, *capital_gain*, *capital_loss*, *hours_per_week*, and *native_country*. Our goal is estimating classlabel based on the input parameters.

(a) **Univariate analysis**

We used histogram plot to understand the data distribution. by looking at the target (class-variable), we can see that number of individuals with more than 50K income annually is about three times more than those who have less than 50K income. *Age* is the next feature that is distributed between 17 to 90. Most of the given data belongs to candidates between 20 to 50 years old. The other attributes are *work_class* and *race* which are imbalanced. Most of the mentioned these data can be categorized in *private* and *white*. Also, It is shown that a huge imbalance can be seen in the *capital_loss*, *capital_gain* and *native_country*.

Moreover, highly imbalanced data poses an added difficulty, as most learners will exhibit bias towards the majority class, and in extreme cases, may ignore the minority class altogether.

In this dataset, the decision has been made to omit features with huge imbalance. Therefore, *native_country*, *work_class*, *capital_loss*, and *capital_gain* has been removed from our analysis. The numerical features can be plotted as per following:

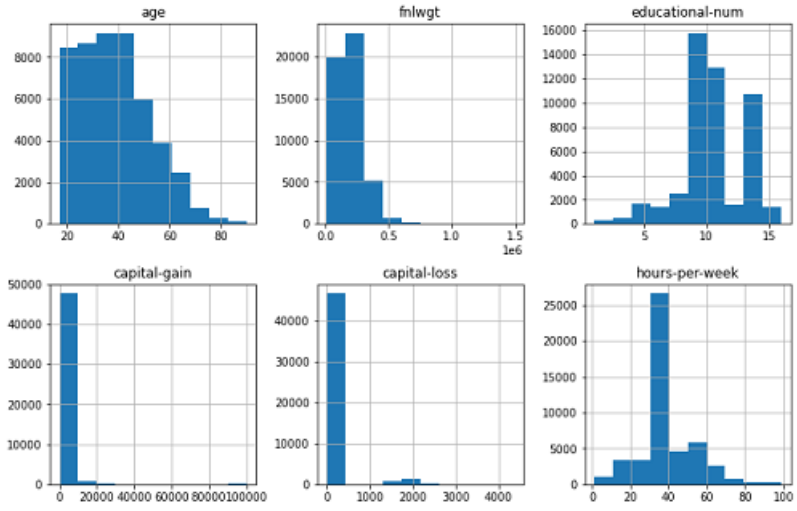

Figure 1: frequency of numerical features

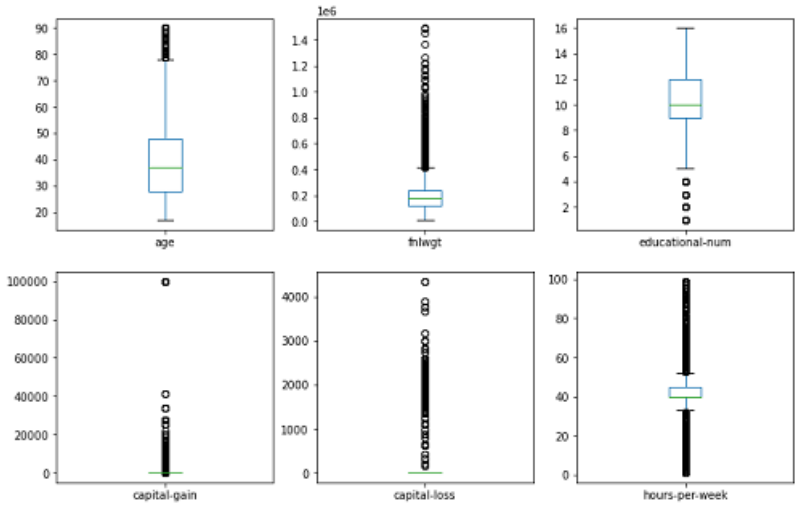

Figure 2: box-plot of numerical features

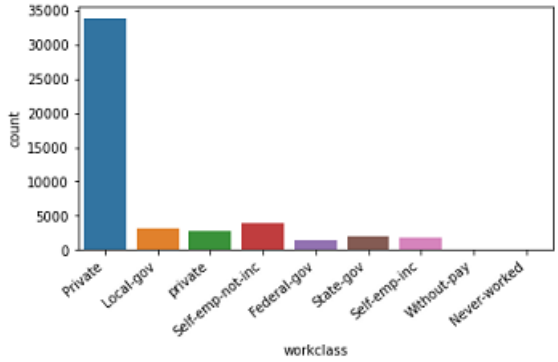

Figure 3: frequency of work class feature

According the above figures it is obvious that we can drop the imbalanced features like *work_class*, *native_country*, *capital_loss*, and *capital_gain*.
To increase the data quality, we can also do features engineering.

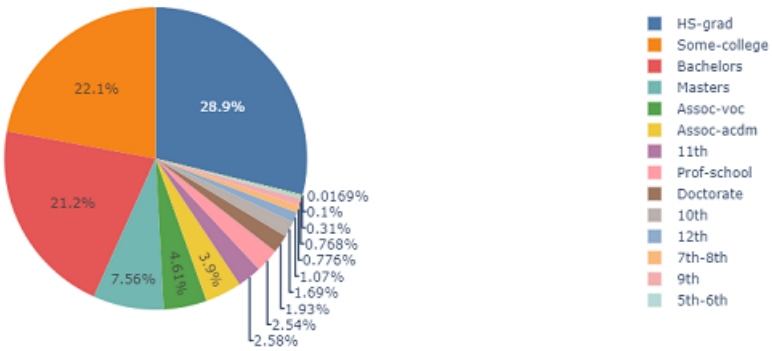

Figure 4: frequency of education feature

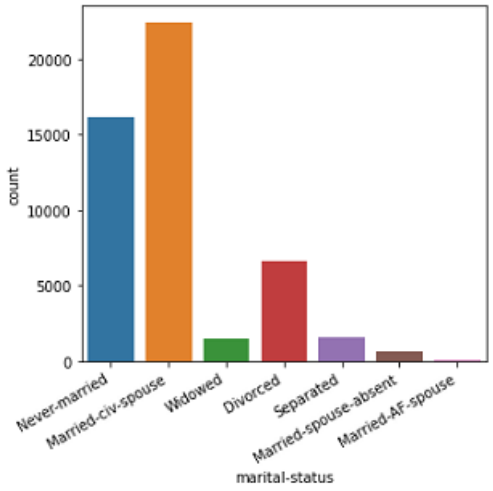

Figure 5: frequency of education feature

Based on the above graphs, these features show detailed information which is not helpful for modeling. These data can be divided effectively by the reduction of their information.

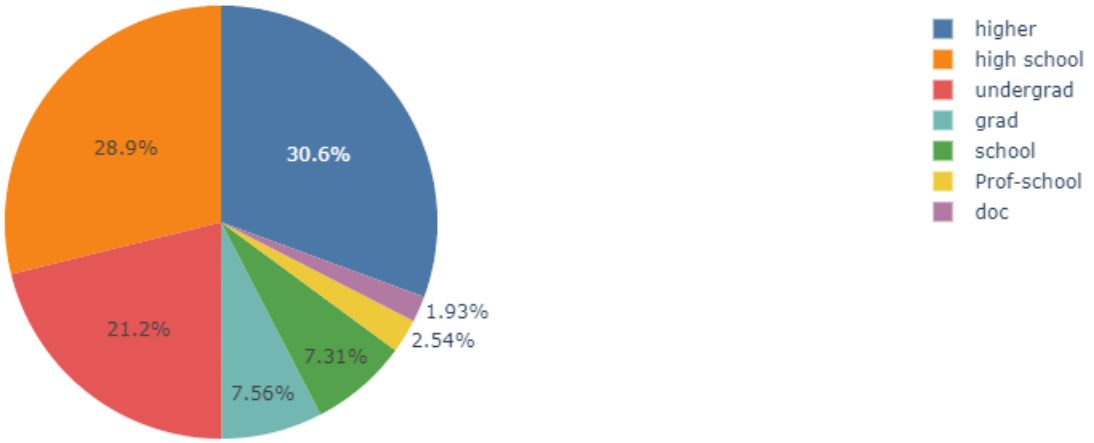

Figure 6: frequency of revised education feature

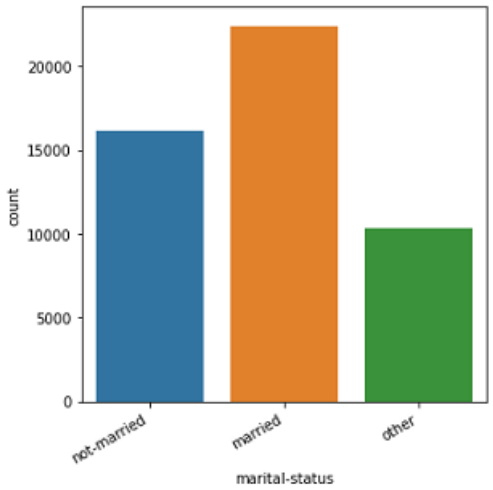

Figure 7: frequency of revised marital_status feature

As it is illustrated in the last two graphs, the stored information in these features decreased, resulting in dropping the model complexity

(b) **Bivariate analysis**

To perform bivariate analysis, we plotted a correlation matrix for all attributes and target. *education_num* and *marital_status* have a high correlation with the target with respect to other attributes, 0.34 and -0.38 respectively. Also, it can be seen that *age* has a moderate correlation coefficient of 0.24. However, it should be considered that there is a relation between *marital_status* / *sex* , *relationship* / *sex*, and *relationship* / *occupation*, resulting in increasing the complexity of our model. To decrease the dimension (which is not mentioned in our task), these features can be selected initially.

Feature selection - Dataset preparation

(a) **Dropping the data**

(i) There are a total of 48,842 rows of data , The dataset contains 3620 missing values that are marked with a question mark character (?). leaving 45,222 complete rows.Missing values that are marked with a ? character , these rows are deleted from the dataset.

(ii) Categories like " Native country" "work class" "capital loss" "capital gain" parameters have also been dropped , as the bar graph shows one particular Parameter in these categories has been completely dominating , which is not healthy for training the algorithm.

(b) **Feature transformation**
This feature transformation of merging various parameters into single parameter helps to train algorithm in better way by improving performance .

(i) Education section

- "preschool" , "1" are merged into one parameter called " School".
- " Assoc-voc" , "Assoc-acdm", "prof-school" , "some-college" are merged into "higher".

(ii) Martial status section

- "Married-civ-spouse" , "Married-AF-spouse" are merged into "married".
- "Divorced" , "separated", "widowed" , "married-spouse-absent" are merged into "other".

According to the above modification of features, the following bar charts indicate the result:

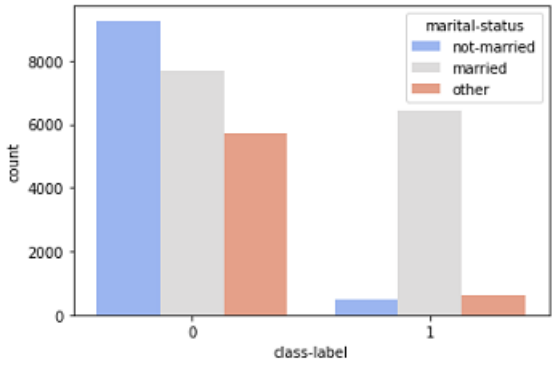

Figure 8: redistribution of earital-status

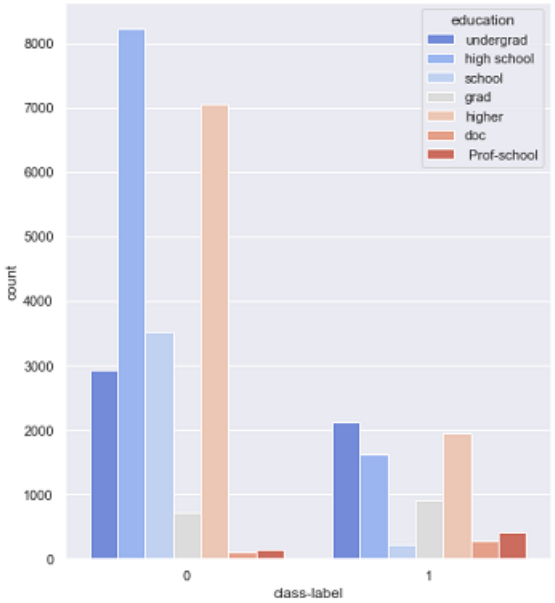

Figure 9: redistribution of education

Classification

(a) **Decision Trees**

we applied decision trees to classify the adult data set. Firstly, we used simple decision trees with *gini* index criteria. The scoring of this DTs can be examined by *accuracy*, *precision*, *recall*, and $f1\_score$.

	precision	recall	f1-score	support
0	0.85	0.84	0.84	11360
1	0.52	0.53	0.53	3700
accuracy			0.77	15060

According to the above information, This model can be classified as moderate-to-high with respect to accuracy. Although *precision*, *recall*, and $f1 - score$ of the class-label 0 are about 0.85, these score are decreased by 35% for the counterpart of class-table (1).Therefore, this model can predict the behavior of class-table 0 in both true positive and true negative. However, this model is not a good estimator with respect to true positive and true negative for class-table 1.

(i) Evaluation and Model parameter tuning
To tune the DTs parameters, we used $criterion$, max_depth, $min\_samples\_split$, $min\_samples\_leaf$, and max_leaf_nodes parameters with 10^{th} fold stratified cross-validation. In this step, the mentioned parameters have been changed to find the best parameters that the output score will be maximized. The Decision has been made to select *accuracy* as an objective score. The model's parameters have been initiated to tune with the following parameters:

- $criterion$: $['gini', 'entropy']$
- max_depth : $[7, 8, 9]$
- $min\_samples\_split$: $[4, 5]$
- $min\_samples\_leaf$: $[4, 5]$
- max_leaf_nodes : $[20, 25, 30]$

After tuning, the accuracy is increased from 0.77 to 0.83 with respect to initial model. The tuned parameters have been listed as following:

- $criterion$: $['gini']$
- max_depth : $[9]$
- $min\_samples\_split$: $[4]$
- $min\_samples\_leaf$: $[4]$
- max_leaf_nodes : $[30]$

Moreover, the evaluation with 10^{th} fold stratified has been performed and the Confusion Matrix result is:

$$\text{CM} = \begin{pmatrix} 20726 & 1928 \\ 3329 & 4179 \end{pmatrix}$$

According to the above confusion matrix, *accuracy*, *sensitivity*, $specificity$, and $f1score$ are 0.83, 0.91, 0.56, and 0.75 respectively.

Finally, we used test data set which is totally separated to test our model.

```
# DTs classifier

                precision    recall  f1-score   support
```

0	0.86	0.91	0.89	11360
1	0.68	0.56	0.61	3700
accuracy			0.83	15060

Based on the previous information, By comparing the measuring criteria before and after evaluation, it is obvious that all criteria boosted up and the model became more precise.
However, The model estimation for *class_label*(1) with respect to *precision*, *recall*, $f1\_score$, and *accuracy* is far behind the *class_label*(0).

(ii) Discriminative behaviour
To evaluate the discriminative behavior of the obtained model, we used all available data to predict the *class_label* of individuals. After prediction and plotting the correlation matrix, there is a positive correlation between *sex* and *class_label* which means this model can not avoid discrimination against individuals in the *sex* feature. However, this model can avoid discrimination against *race* because the correlation is negligible.

(b) **SVM Classification**
we applied SVM to classify adult data set. Firstly, we used a linear kernel. The scoring of this SVM can be examined by *accuracy*, *precision*, *recall*, and $f1\_score$.

SVM classifier

	precision	recall	f1-score	support
0	0.84	0.93	0.88	11360
1	0.68	0.46	0.55	3700
accuracy			0.81	15060

According to the above information, This model can be classified as moderate-to-high with respect to accuracy. Although *precision*, *recall*, and $f1 - score$ of the class-label 0 are fluctuated from 0.84 to 0.93, these score are much less than for counterpart of class-table (1).Therefore, this model can predict the behavior of class-table 0 in both true positive and true negative. However, this model is not a good estimator with respect to true positive and true negative for class-table 1.

(i) Evaluation and Model parameter tuning
To tune the SVM parameters, we used three different kernel function namely *liner*, *polynomial*, *sigmoid* and $Radial\_Basis\_Function$ with parameters with 10^{th} fold stratified cross-validation. In this step, the mentioned parameters have been changed to find the best parameters that the output score will be maximized. The Decision has been made to select *accuracy* as an objective score. The model's parameters have been initiated to tune with the following parameters:

- $svm\_linear = 'C' : [0.2, 0.3], 'kernel' : ['linear']$

- $svm\_poly = 'C' : [0.5, 1], 'gamma' : [0.001, 0.003], 'kernel' : ['poly'], 'degree' : [2, 3]$
- $svm\_others = 'C' : [0.3, 0.5], 'gamma' : [0.001, 0.003], 'kernel' : ['rbf', 'sigmoid']$

After tuning, the accuracy is increased from 0.81 to 0.82 with respect to the initial model which is not significant. The tuned parameters have been listed as follows:

- $'C' : 1, 'degree' : 3, 'gamma' : 0.003, 'kernel' :' poly'$

It is worth-mentioning that we tried to increase C parameters to decrease misclassification, but this algorithm didn't converge in the expected running time.
Moreover, the evaluation with 10^{th} fold stratified has been performed and the Confusion Matrix result is:

$$\text{CM} = \begin{pmatrix} 21381 & 1273 \\ 4218 & 3290 \end{pmatrix}$$

According to the above confusion matrix, *accuracy*, *sensitivity*, *specificity*, and $f1score$ are 0.82, 0.94, 0.44, and 0.72 respectively.

Finally, we used test data set which is totally separated to test our model.

	precision	recall	f1-score	support
0	0.84	0.94	0.89	11360
1	0.71	0.44	0.55	3700
accuracy			0.82	15060

Based on the previous information, By comparing the measuring criteria before and after evaluation, it is obvious that all criteria remained the same and the model didn't improve.
However, The model estimation for $class\_label(1)$ with respect to *precision*, *recall*, $f1\_score$, and *accuracy* is far behind the $class\_label(0)$.

(ii) Discriminative behaviour
To evaluate the discriminative behavior of the obtained model, we used all available data to predict the *class_label* of individuals. After prediction and plotting the correlation matrix, there is a positive correlation between *sex* and *class_label* which means this model can not avoid discrimination against individuals in the *sex* feature. However, this model can avoid discrimination against *race* because the correlation is negligible. Also, the SVM model is less discriminative than DTs with respect to *sex* features.

(c) **k Nearest Neighbors Classification**
We used the sklearn library to train our kNN classifier. Then, we optimized it's parameters using 10-fold stratified validation, and checked how good our classifier is by computing the accuracy, precision, recall, and f1 score. We use multi-threading to speed up the process.

(i) Evaluation and Model parameter tuning
The parameters which affect our model are: the neighbors (k), how the label is predicted in relation to the labels of the k nearest neighbours(how the 'weights' for the k nearest neighbors are chosen), and what the distance metric is. For example, acording to our parameter tuning above, the best metric for choosing the label is not the majority, but the weighted majority, considering the distances.

We tested some possible configurations for the parameters. For example, out of the following parameters:

- $n - neighbors = 3, 5, 10, 15, 20, 50$
- $weights = uniform, distance$
- $metric = euclidean, manhattan$

The tuning increases our accuracy by 0.01 w.r.t. the initial model.
The best parameters for evaluation are as per following:

- $'metric' :' manhattan',' n_neighbors' : 50,' weights' :' distance'$

Moreover, the evaluation with 10^{th} fold stratified has been performed and the Confusion Matrix result is:

$$\text{CM} = \begin{pmatrix} 21381 & 765 \\ 6584 & 924 \end{pmatrix}$$

According to the above confusion matrix, *accuracy*, *sensitivity*, $specificity$, and $f1score$ are 0.76, 0.97, 0.12, and 0.53 respectively.

Finally, we used test data set which is totally separated to test our model.

	precision	recall	f1-score	support
0	0.75	1.00	0.86	11360
accuracy			0.75	15060

To note is that the model estimation for $class\_label(1)$ with respect to *precision*, *recall*, $f1\_score$, and *accuracy* is far behind the $class\_label(0)$.

(ii) Discriminative behaviour
Next, we checked to see if our model discriminates based on gender or race. For this, we plotted the correlation matrix for all instances with their class label replaced with what was predicted by the classifier. We checked the correlation between the 'sex','race' features and the class label(income).
The correlation between 'sex' and 'income' is 0.1, while the correlation between 'sex' and 'income' is 0.054. The second value is insignificant, and we can sa that the model does not discriminate depending on race. The value is a bit higher for the correlation between 'sex' and income, but it can also be said that the model accounts for real-world discrimination which exists in regards to income distribution.

(d) **Native Bayes Classifier**
We used the sklearn library to train our NB classifier. Specifically, we used Gaussian native bayes classifier(this means, the assumption that our attributes follow the gauss-distribution), as it performed better for our data than others tested(Bernoulli, Multinomial).

	precision	recall	f1-score	support
0	0.76	1.00	0.86	11360
1	0.83	0.05	0.09	3700
accuracy			0.76	15060

(i) Evaluation and Model parameter tuning
Then, we optimized it's parameters using 10-fold stratified validation, and checked how good our classifier is by computing the accuracy, precision, recall, and f1 score. We use multi-threading to speed up the process.
For optimizing, we need to choose the parameters for the gaussian distribution. This means, we pick the middle point and the variance for the distribution. We do this using the sklearn library, telling it to tune the parameter '$var_smoothing'$.
(source: https://medium.com/analytics-vidhya/how-to-improve-naive-bayes-9fa698e14cba)

Moreover, the evaluation with 10^{th} fold stratified has been performed and the Confusion Matrix result is:

$$\text{CM} = \begin{pmatrix} 22555 & 99 \\ 7045 & 463 \end{pmatrix}$$

According to the above confusion matrix, *accuracy*, *sensitivity*, $specificity$, and $f1score$ are 0.76, 1.00, 0.06, and 0.49 respectively.

Finally, we used test data set which is totally separated to test our model.

	precision	recall	f1-score	support

0	0.76	1.00	0.86	11360
1	0.83	0.05	0.09	3700
accuracy			0.76	15060

(ii) Discriminative behaviour
Next, we checked to see if our model discriminates based on gender or race. For this, we plotted the correlation matrix for all instances with their class label replaced with what was predicted by the classifier. We checked the correlation between the 'sex','race' features and the class label(income).
The correlation between 'sex' and 'income' is 0.078, while the correlation between 'sex' and 'income' is 0.022.(See above)

(e) **Perceptron Classifier**
We used pytorch to implement and train the perceptron. We have a single hidden layer in our Network. We use rectified linear as our activation function. As the optimiser, we use the Adam optimiser from pytorch, which is an algorithm for gradient descent. Our loss function is the negative log likelyhood loss, though we have also tried binary cross-enthropy with similar results.

(i) Evaluation and Model parameter tuning
We tried optimizing our classifier by optimizing the learning rate and checked how good our classifier is by computing the accuracy, precision, recall, and f1 score. Another possible parameter which impacts the classifier's performance are the initial weights.

The best learning rate: tuning suggests that improvements are not significant, any learning rate between 0.1 and 0.01 delivers similar results.(See code)
Confusion Matrix result- on test set:

$$\text{CM} = \begin{pmatrix} 9233 & 2107 \\ 3002 & 718 \end{pmatrix}$$

We used test data set which is totally separated to test our model to find the $accuracy$, $sensitivity$, $specificity$, and $f1score$.

	precision	recall	f1-score
0	0.75	0.81	0.78
accuracy			0.66

Visualization

In order to visualize our models, we use an advanced technique , called dimension reduction, to have a better understanding of decision boundaries in different classifier. Dimensionality reduction, or dimension reduction, is the transformation of data from a high-dimensional space into a low-dimensional space so that the low-dimensional representation retains some meaningful properties of the original data,

ideally close to its intrinsic dimension. Working in high-dimensional spaces can be undesirable for many reasons; raw data are often sparse as a consequence of the curse of dimensionality, and analyzing the data is usually computationally intractable (https://en.wikipedia.org/wiki/Dimensionality_reduction).
Dimension reduction is computationally expensive, so we used just 2000 instances of test dataset. The following graphs are the ability of different classifiers to with respect to two dimensions.

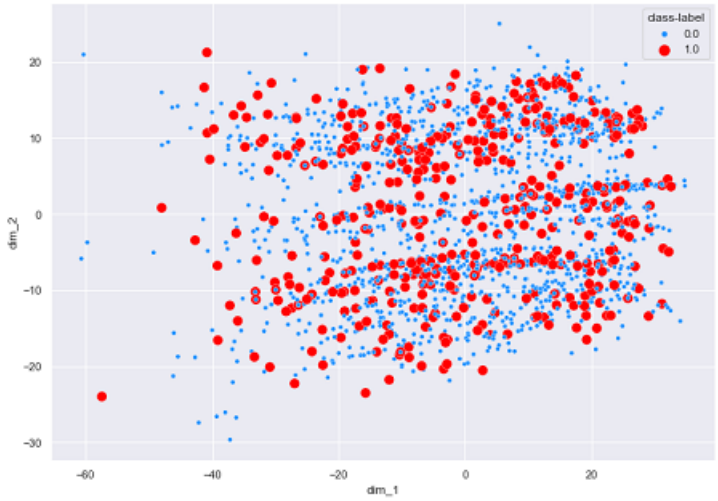

Figure 10: Decision boundaries of DTs

Figure 11: Decision boundaries of KNNs

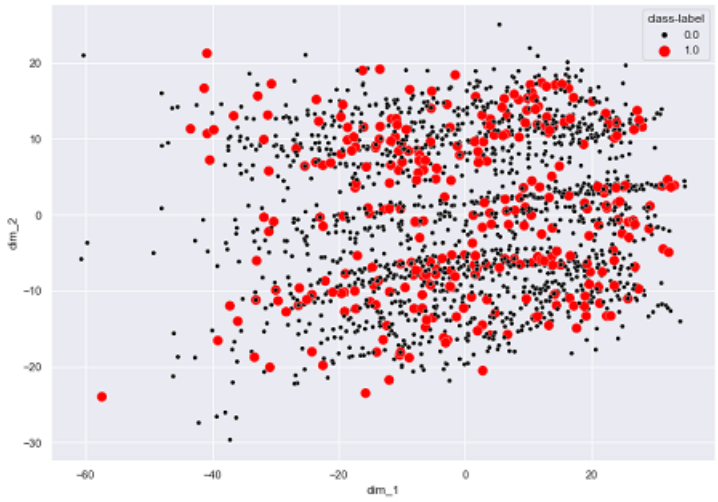

Figure 12: Decision boundaries of SVM

Figure 13: Decision boundaries of NBs

Result and Discussion

In this study, five different prediction models has been compared in their *accuracy*, *sensitivity*, *specificity*, and $f1score$:

- Decision Trees
- Support Vector Machine
- K Nearest Neighbors
- Naive Bayes
- perceptron

In summary, what has been achieved so far. The most important criteria to compare the result of different is *accuracy* that is shown below:

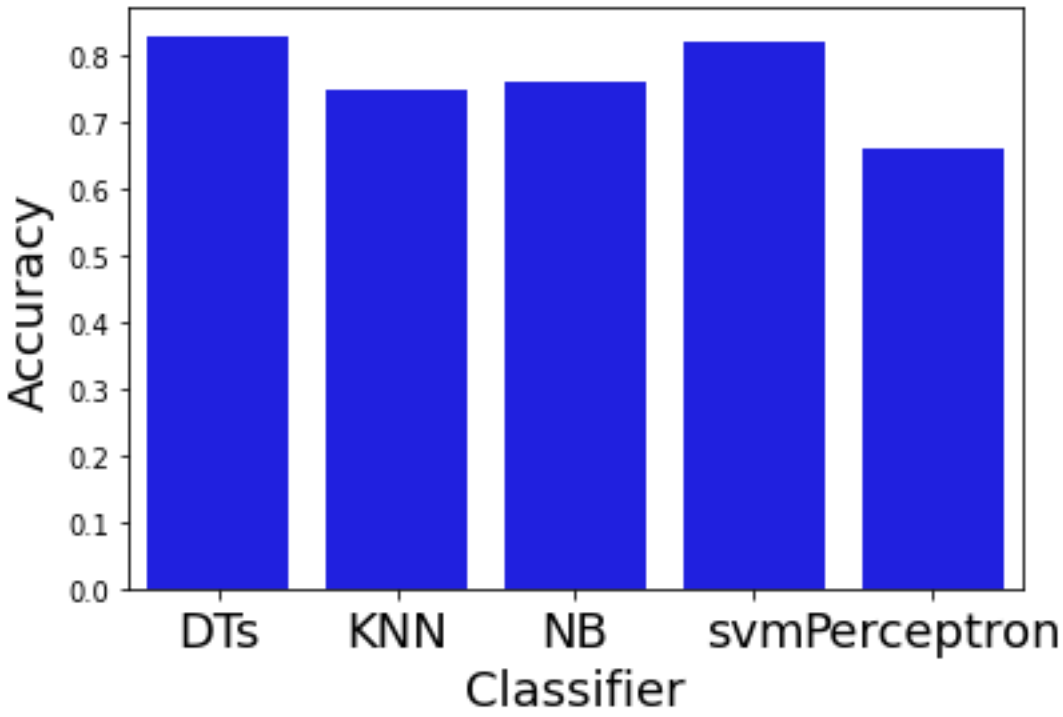

Figure 14: Accuracy of different classifiers

According to the above chart, it can be easily seen that DTs and SVM, have a same result with 0.83 *accuracy*, KNNs and NBs are somewhat less accurate then those two, but perceptron is far behid of these classifier with 0.66 *accuracy*.

Classifier	precision	recall	f1-score
DTs	0.86	0.91	0.89
KNNs	0.75	1.00	0.86
NBs	0.76	1.00	0.86
SVM	0.84	0.94	0.88
Perceptron	0.75	0.81	0.78

Table 1: Performance Result of Class_label 0

Classifier	precision	recall	f1-score
DTs	0.68	0.56	0.61
NBs	0.83	0.05	0.09
SVM	0.71	0.44	0.55
Perceptron	0.25	0.20	0.23

Table 2: Performance Result of Class_label 1

Overall, this dataset is suffering from imbalanced input data, resulting in a significant difference between $class-label$ 0 and 1. By comparing $precision$ and $recall$, $class-label$ 0 can classify better than $class-label$ 1. Although the performance of KNNs, NBs are the same and there is no preference to choose the model as a candidate with respect to performance. The best model belongs to DTs which is slightly more than KNNs and NBs. In the SVM model, $precision$ of $class-label$ 0 is decreased by 0.02 with compare to DTs, however this parameter for $class-label$ 1 is raised by 0.05. In contrast, $recall$ of $class-label$ 0 experiences a dramatic decline from 0.55 to 0.44. Combining $precision$ and $recall$ gives us $f1score$ which is same as the DTs classifier.
The simple perceptron classifier is not suitable for this dataset, due to the high-class

imbalance in multiple attributes, as well as attributes being at least half categorical. The performance of a more complex neural network, or a set of connected perceptrons, could improve on this.

In summary, DTs can be introduced as a good candidate (due to high score in terms of $accuracy$ and $f1score$) to classify instances with the given dataset.