# Predicting Individual Income Using US Census Data

MATH 2319 Machine Learning Applied Project Phase II

*Yong Kai Wong (s9999999) & Vural Aksakalli (s0000000)*

*2 January 1900*

## Contents

## 1 Introduction

The objective of this project was to build classifiers to predict whether an individual earns more than USD 50,000 or less in a year from the 1994 US Census Data sourced from the UCI Machine Learning Repository. In Phase I, we cleaned the data and re-categorised some descriptive features to be less granular. In Phase II, we built three binary-classifiers on the cleaned data. The rest of this report is organised as follow. Section 2 describes an overview of our methodology. Section 3 discusses the classifiers' fine-tuning process and detailed performance analysis of each classifier. Section 4 compares the performance of the classifiers using the same resampling method. Section 5 critiques our methodology. The last section concludes with a summary.

## 2 Methodology

We considered three classifiers - Naive Bayes (NB), Random Forest (RF), and $K$-Nearest Neighbour (KNN). The NB was the baseline classifier. Each classifier was trainned to make probability predictions so that we were able to adjust prediction threshold to refine the performance. We split the full data set into 70 % training set and 30 % test set. Each set resembled the full data by having the same proportion of target classes i.e. approximately 76 % of individuals earning less than USD 50,000 and 24 % earning higher. For fine-tuning process, we ran a five-folded cross-validation stratified sampling on each classifier. Stratified sampling was used to cater the slight imbalance class of the target feature.

Next, for each classsifer, we determined the optimal probability threshold. Using the tuned hyperparameters and the optimal thresholds, we made predictions on the test data. During model training (hyperparameter tuning and threshold adjustment), we relied on mean misclassification error rate (mmce). In addition to mmce, we also used the confusion matrix on the test data to evaluate classifiers' performance. The modelling was implemented in `R` with the `mlr` package (Bischl et al. 2016).

## 3 Hyperparameter Tune-Fining

### 3.1 Naive Bayes

Since the training set might have unwittingly excluded rare instances, the NB classifier might produce some fitted zero probabilities as predictions. To mitigate this, we ran a grid search to determine the optimal value of the Laplacian smoothing parameter. Using the stratified sampling discussed in the previous section, we experimented values ranging from 0 to 30. The optimal Laplacian parameter was 3.33 with a mean test error of 0.167.

### 3.2 Random Forest

We tune-fined the number of variables randomly sampled as candidates at each split (i.e. `mtry`). For a classification problem, Breiman (2001) suggests `mtry` $= \sqrt{p}$ where $p$ is the number of descriptive features. In our case, $\sqrt{p} = \sqrt{11} = 3.31$. Therefore, we experimented `mtry` $= 2$, 3, and 4. We left other hyperparameters, such as the number of trees to grow at the default value. The result was 3 with a mean test error of 0.139.

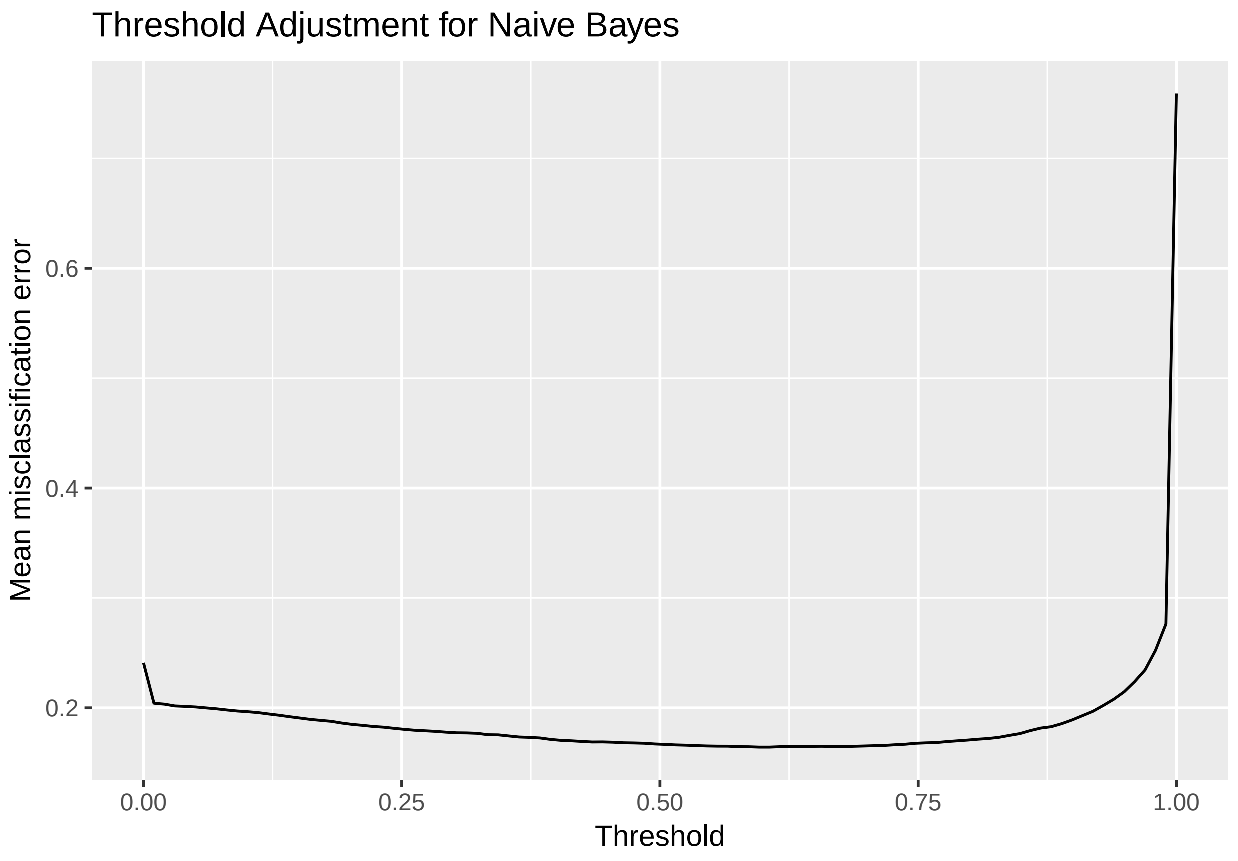
### 3.3 $K$-Nearest Neighbour

By using the optimal kernel, we ran a grid search on $k = 2, 3, ...20$. The outcome was 20 with a mean test error of 0.165.

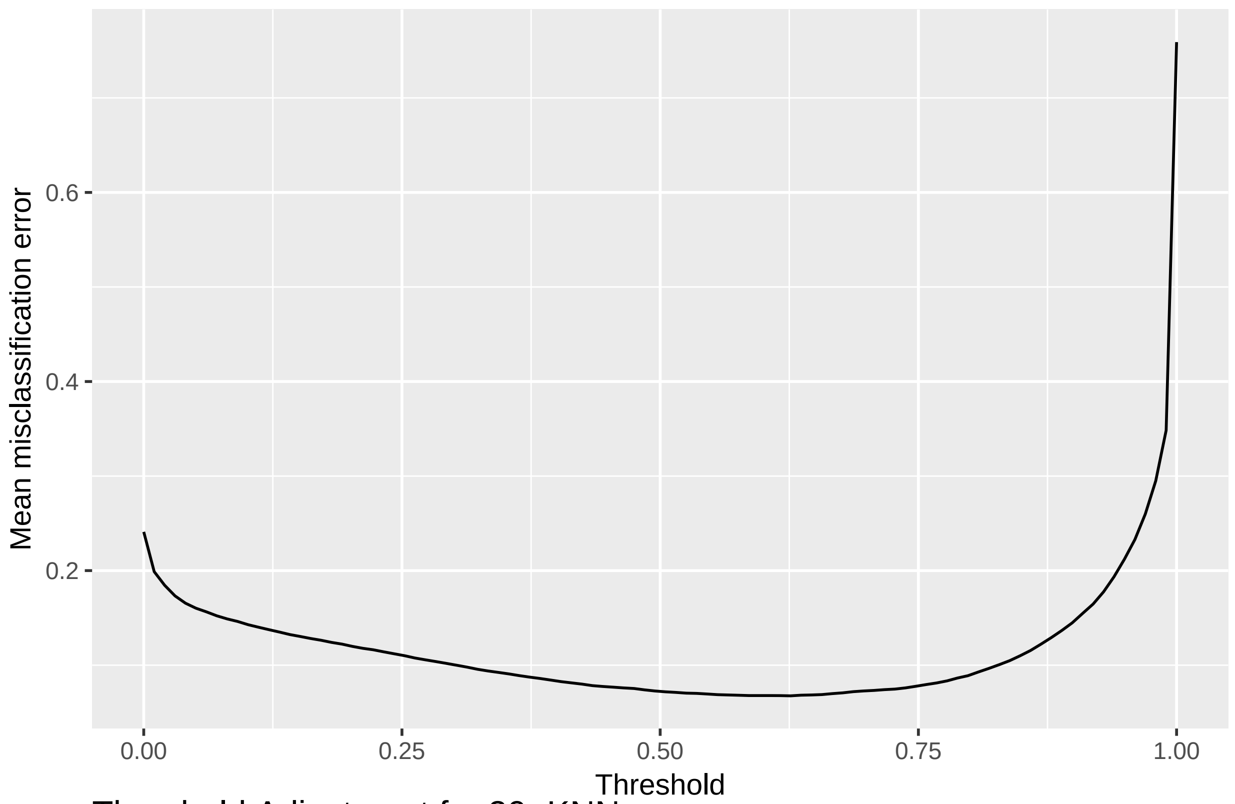
## 3.4 Threshold Adjustment

The following plots depict the value of mmce vs. the range of probability thresholds. The thresholds were approximately 0.60, 0.63, and 0.51 for NB, RF, and 20-KNN classifiers respectively. These thresholds were used to determine the probability of an individual earning more than USD 50,000.

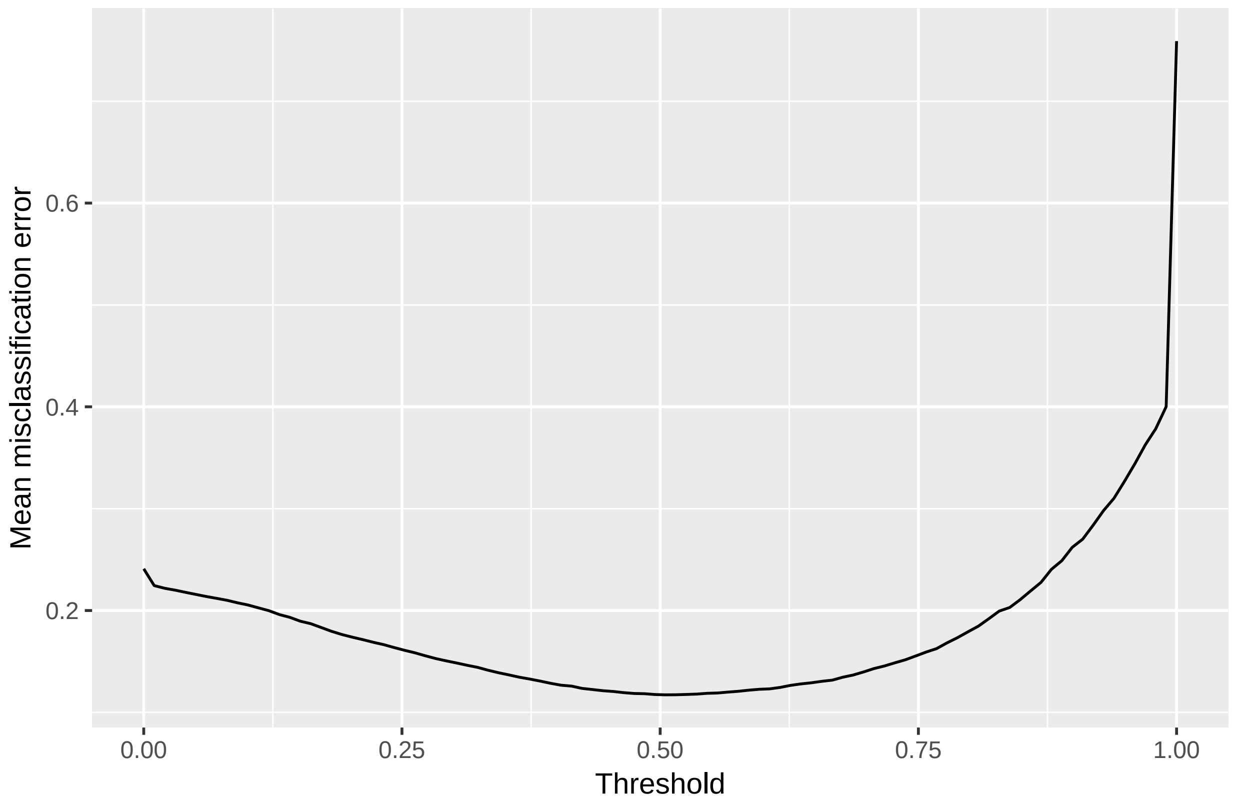
```
## Warning: replacing previous import 'BBmisc::isFALSE' by
## 'backports::isFALSE' when loading 'mlr'
```



Threshold Adjustment for Random Forest



Threshold Adjustment for 20−KNN



## 4 Evaluation

Using the parameters and threshold levels, we calculated the confusion matrix for each classifier. The confusion matrix of NB classifer is as follow:

```
## Relative confusion matrix (normalized by row/column):
##         predicted
## true     <=50K     >50K      -err.-
##   <=50K  0.94/0.86 0.06/0.29 0.06
##   >50K   0.48/0.14 0.52/0.71 0.48
##   -err.-      0.14      0.29 0.16
##
##
## Absolute confusion matrix:
##         predicted
## true     <=50K >50K -err.-
##   <=50K  10493  712    712
##   >50K    1663 1785   1663
##   -err.-  1663  712   2375
```

The confusion matrix of RF classifer is as follow:

```
## Relative confusion matrix (normalized by row/column):
##         predicted
## true     <=50K     >50K      -err.-
##   <=50K  0.90/0.91 0.10/0.31 0.10
##   >50K   0.30/0.09 0.70/0.69 0.30
##   -err.-      0.09      0.31 0.15
##
##
## Absolute confusion matrix:
##         predicted
## true     <=50K >50K -err.-
##   <=50K  10096 1109   1109
##   >50K    1025 2423   1025
##   -err.-  1025 1109   2134
```

The confusion matrix of 20-KNN classifer is as follow:

```
## Relative confusion matrix (normalized by row/column):
##         predicted
## true     <=50K     >50K      -err.-
##   <=50K  0.92/0.87 0.08/0.32 0.08
##   >50K   0.43/0.13 0.57/0.68 0.43
##   -err.-      0.13      0.32 0.17
##
##
## Absolute confusion matrix:
##         predicted
## true     <=50K >50K -err.-
##   <=50K  10253  952    952
##   >50K    1470 1978   1470
##   -err.-  1470  952   2422
```

All classifiers accurately predicted individual earning less than USD 50,000, but not high-income earners. The class accuracy difference was substantial. Based on class accuracy and mmce, we concluded that the RF classifer was the best model.

## 5 Discussion

The previous section showed that all classifiers did not perform accurately in predicting the high-income earners despite the stratified sampling. This implies the imbalance class problem was prevalent. A better approach would be a cost-sensitive classification where we could have allocated more cost to true positive groups i.e. the correctly predicted high-income class. Another alternative would be under- or oversampling to adjust the class balance, despite the risk of inducing biases.

The NB model assumes the descriptive features to follow normality that are not necessarily true. The solution would be a transformation on numeric features. Based mmce, the KNN classifer underperformed the RF and NB classifier. This highlights the KNN classifier might not be appropriate given there were many categorical features in the data. The RF outperformed other models because it had the bagging mechanism (i.e. 500 trees) to improve its accuracy. Having said this, it was "unfair" to the NB and KNN classifiers because the RF was able to run multiple bagged models at each iteration during the resampling.

## 6 Conclusion

Among three classifiers, the Random Forest produces the best performance in predicting individuals earning more than USD 50,000. We split the data into training and test sets. Via a stratified sampling, we determined the optimal value of the selected hyperparameter of each classifier and the probability threshold. Despite this, the imbalance class issue still persisted and therefore reduced the class accuracy of the high-income earners. For future works, we proposed to consider cost-sensitive classification and under/over-sampling methods to mitigate the class imbalance.

## References

Bischl, Bernd, Michel Lang, Lars Kotthoff, Julia Schiffner, Jakob Richter, Erich Studerus, Giuseppe Casalicchio, and Zachary M. Jones. 2016. "`mlr`: Machine Learning in R." Journal of Machine Learning Research.

Breiman, L. 2001. "Random Forests." Machine Learning.