Predicting stone artefact classification: A case study from the Ban Rai Area, Thailand.

Laura Vodden

Abstract

Stone artefacts comprise a large proportion of objects recovered from archaeological sites, owing to their durability and resistance to deterioration (Clarkson and O'Connor, 2007). They are valuable to archaeologists, because they provide an excellent record of human behaviour throughout prehistory, and can reveal information about diet, creativity, and past human interactions with the environment (Clarkson and O'Connor, 2007; Marwick, 2008).

The purpose of this report is to investigate whether machine learning algorithms can use archaeological data to classify a stone artefact assemblage into two broad categories. The report makes use of observational data relating to artefacts excavated from a rock shelter at Ban Rai, Northwest Thailand, between 2001 and 2007.

In performing the investigation, three machine learning algorithms were applied to the data: A Naïve-Bayes Classifier first assessed the accuracy of predictions, a Logistic Regression then assessed the variables most likely to affect classification either way, and a K-Means Cluster Analysis was used to visualise the data, to see if the parameters for each artefact category were distinct enough to form two clusters.

The report finds that machine learning algorithms can be used to classify the Ban Rai stone artefacts into two categories. This has implications for archaeological investigation, because it suggests that machine learning algorithms can be applied to other archaeological assemblages, and perhaps can even further differentiate between sub-categories of artefacts, leading to new and exciting discoveries.

Introduction

Stone artefact, or 'lithic' analysis is well-documented in archaeological investigation (Clarkson and O'Connor, 2007). Based on the distinct morphologies and techniques used, archaeologists are able to derive information relating to human behaviour and the evolution of human ingenuity throughout prehistory (Clarkson and O'Connor, 2007; Marwick, 2008).

This report aims to identify whether machine learning models can be applied to a lithic assemblage in order to classify stone artefacts into two categories: Cores and flakes. Using data from excavations at Ban Rai, Northwest Thailand, this report seeks to determine whether existing observational lithic analysis data can be used to predict artefact type. The null and alternative hypotheses are as follows:

> ***H_0: Observational lithic analysis data cannot be used to predict artefact type***
> ***H_A: Observational lithic analysis data can be used to predict artefact type***

These hypotheses will be tested by applying a **Naïve-Bayes classification model** using numeric and categorical parameters, a **logistic regression** using the numeric parameters, and a **k-means cluster analysis** to visualise the clustered data.

Data

The data used in this report are publicly available as companion to Marwick's (2008) PhD thesis, submitted to the Australian National University (ANU). The data were accessed from Harvard Dataverse in November 2020. The data were originally collected by the Highland Archaeology Project in Pang Mapha (HAPP), with excavations taking place at Ban Rai, Thailand, between 2001 and 2007.

The Ban Rai data originally consisted of 2,419 observations and 138 variables, however, removal of records containing missing data, as well as columns lacking useful information resulted in a useable dataset of 1,564 observations and 15 variables. A summary of the data is shown in *Table 1*, and a snapshot of the dataframe is presented in *Table 2.* The first ten (numeric) variables describe the various measurements taken from each stone artefact, in accordance with universal procedure in the field of lithic archaeology (Clarkson and O'Connor, 2007). The latter six variables relate to the morphology of each artefact, determined by visual analysis and also in line with standard procedure within the discipline.

Table 1: A summary of the lithic assemblage data from the Ban Rai excavation site.

Variable Name	Description	Variable Type	Unit	Levels
MASS__G	Mass	Continuous, numeric	grams	
PROXIMAL_WIDTH	Width of stone artefact at widest point	Continuous, numeric	mm	
MEDIAL_WIDTH	Width at mid-point (width)	Continuous, numeric	mm	
DISTAL_WIDTH	Length at mid-point (length)	Continuous, numeric	mm	
PROXIMAL_THICKNESS	Thickness at thickest point	Continuous, numeric	mm	
MEDIAL_THICKNESS	Thickness at mid-point	Continuous, numeric	mm	
DISTAL_THICKNESS	Thickness at mid-point	Continuous, numeric	mm	
PLATFORM_WIDTH	Width of platform	Continuous, numeric	mm	
PLATFORM_THICKNESS	Thickness of platform	Continuous, numeric	mm	
OVERHANG_R	Overhang reduction, or removal of overhanging material between creation of separate flakes.	Categorical, factor		Absent, Present
INITIATION	Fracture shape at point of force, also known as 'cone of force'.	Categorical, factor		Bending, Hertzian (conical fracture), Absent
BULB_OF_PE	Bulb of percussion, also known as bulb of force, formed as the fracture migrates through the material.	Categorical, factor		Absent, Diffuse, Pronounced
TERMINATION	Step or feather	Categorical, factor		Feather, Step, Hinge, Outrepasse, Absent
RAW_MATERIAL	Main material composition of artefact	Categorical, factor		Quartzite, Yellow_Quartzite, Black_Quartzite, Red_Quartzite, Shale, Mudstone, Andesite, Sandstone
ARTEFACT_C	Classification of artefact	Categorical, factor		Core, Flake

Table 2: The Ban Rai dataframe in R.

```
head(data)
  MASS__G_ PROX_WIDTH MED_WIDTH_ DIST_WIDTH PROX_THICK MED_THICKN DIST_THICK PLAT_WIDTH
      4.45      23.43      17.96      11.40       8.17       6.39       4.98      23.12
      1.12      16.60      20.21      14.07       3.45       3.70       1.36      12.88
     86.46      52.38      63.07      56.34      15.23      20.32       7.42      50.85
      3.00      19.89      27.56      16.12       4.14       4.81       2.02      18.61
    131.49      64.94      70.38      59.50      56.63      52.51      44.52      63.11
      9.76      23.61      42.97      32.84       7.18       7.86       2.17      19.65
  PLAT_THICK OVERHANG_R INITIATION BULB_OF_PE TERMINATIO        RAW_MATERI ARTEFACT_C
        7.03     Absent   Herztian    Diffuse    Feather         Quartzite      Flake
        3.25     Absent   Herztian    Diffuse    Feather         Quartzite      Flake
       10.78    Present   Herztian Pronounced    Feather Yellow Quartzite      Flake
        3.87     Absent   Herztian    Diffuse    Feather         Quartzite      Flake
       55.90     Absent     Absent     Absent     Absent         Quartzite       Core
        8.66     Absent   Herztian Pronounced    Feather  Black Quartzite      Flake
```

It is important to clarify a few terms that are specific to archaeological lithics analysis and therefore may not be familiar to the reader. *Table 3* is a glossary of the most pertinent terms relating to this dataset:

Table 3: Glossary of terms used in this report, derived from explanations in Clarkson and O'Connor, 2017.

Terminology	**Definition/Example**
Assemblage	A collection of artefacts belonging to a particular site, linked spatially, temporally and/or culturally.
Core	The main part of a rock from which pieces are strategically removed via a process known as 'knapping', or striking a rock to remove pieces. The removed pieces are known as 'flakes'.
Flake	Stone fragments that have been removed from a core. These can be discarded, used as they are, or modified into more sophisticated tools, such as knives or arrowheads.
Platform	A flat surface found on a core, this surface usually faces upwards and flakes are struck off perpendicular to this plane.
Overhang	Material left behind after a flake has been struck from a core. This can be left as is or removed before creating the next flake.
Initiation/cone of force	The shape of the initial fracture created by striking the core. Present on the ventral (under) side of flakes, and on the core. Shape varies depending on material used.
Bulb of percussion/force	As the fracture moves through the rock, the stress creates a protrusion which is observed immediately beneath the cone of force, on the ventral side of the flake.
Termination	The shape of the fracture at the point at which the force of striking exits the material (observed at the pointed end of a flake). The two most common forms are feathered (a smooth exit) and stepped (a step shape). Dependent on the material.

See *Appendix 1* for a labelled image of an example stone artefact.

Methods

The following machine-learning algorithms were performed using the R statistical software, version 4.0.2. A full list of packages used can be found in the reference list.

1. Naïve-Bayes Classifier (NBC)

Naïve-Bates Classifiers use the Bayes Theorem to classify the probability of X occurring given that Y has already occurred. The NBC is ideal for this situation, since it performs well with a mix of numeric and categorical variables. Because the Naïve Bayes Classifier assumes conditional independence; it is most accurate when all attributes are assumed to be conditionally independent of each other (Han, Kamber & Pei, 2011). For this analysis, the data are presumed to satisfy this assumption. All **14** variables are used to predict the artefact class (ARTEFACT_C).

Because the Naïve-Bayes classifier assumes a Gaussian distribution in numeric data (James et al. 2013), it was necessary to investigate the distribution of the numeric Ban Rai data. The density plots in *Figure 1* show that, for the 'Flake' category, the nine numeric variables are skewed to the left, indicating a mean lower than the median. This is likely due to the far higher variation in flake size (and number of very small fragments) as opposed to larger cores, for which the distribution is markedly less skewed.). It should be noted that the proportion of cores to flakes is approximately 8 cores to 92 flakes, so this is an unbalanced dataset. This is a natural effect, since one core can yield multiple flakes, and even the smallest, discarded flake fragments have archaeological value and are recorded as artefacts (Marwick, 2008). Since this effect cannot be avoided, it is necessary to use a non-parametric kernel distribution for the Naïve-Bayes classifier in this instance.

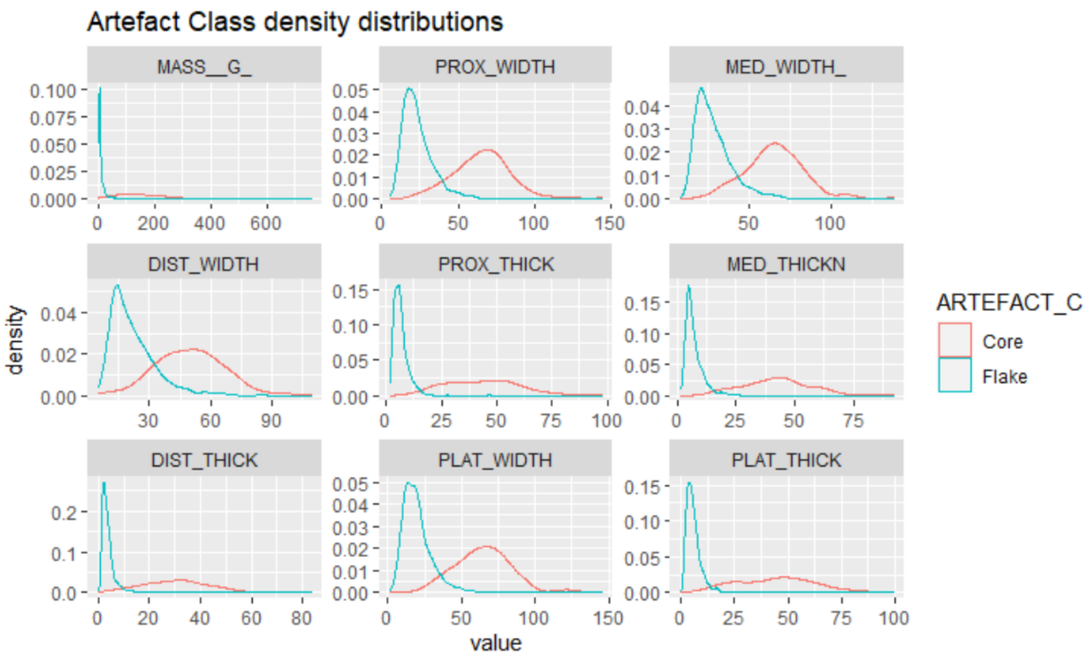

Figure 1: Density plots for numeric variables in the Ban Rai dataframe.

Also to be taken into consideration is the fact that the Ban Rai dataframe (*Table 2*) contains both numeric and categorical data. It was therefore a requirement that Laplace smoothing was used, in order to account for these categorical variables.

The following code shows the application of the Naïve-Bayes classifier to the Ban Rai data.

```
# Ensure reproducibility
set.seed(111)

# Split data into test and train
split <- createDataPartition(data$ARTEFACT_C, p = 0.8, list = FALSE)
train <- data[split, ]
test <- data[-split, ]

# Train model on train data
```

```
nB_model <- naive_bayes(ARTEFACT_C ~ ., data = train, laplace = 1, kernel = TRUE)

# View predictions and accuracy on test data
pred <- predict(nB_model, test, type = "class")
confusionMatrix(pred, test$ARTEFACT_C)

# View predictions and accuracy on train data
pred <- predict(nB_model, train, type = "class")
confusionMatrix(pred, train$ARTEFACT_C)
```

The results of the Naïve-Bayes classification can be found in the *Results* section.

2. Logistic Regression

A logistic model provides information as to which variables are most important in producing a given outcome (James et al, 2013). For the Ban Rai data, a logistic model requires use of the numeric predictors only, to determine the artefact class and the variables that have the greatest impact on this classification.

Logistic regression does not require variables to be statistically independent from each other, however, the model does assume that variables are not collinear, since collinearity introduces complications when determining the individual impact of multiple variables (Midi et al., 2010). *Table 4* shows that this assumption has been met in most cases (PROX_THICK and PLAT_THICK do approach collinearity).

Table 4: Table of correlation showing, in general, variables are not collinear.

	MASS__G_	PROX_WIDTH	MED_WIDTH_	DIST_WIDTH	PROX_THICK	MED_THICKN	DIST_THICK	PLAT_WIDTH	PLAT_THICK
MASS__G_	1.0000000	0.7861545	0.7625795	0.6538550	0.8169184	0.8477081	0.8102758	0.7781210	0.8000076
PROX_WIDTH	0.7861545	1.0000000	0.8713244	0.7063473	0.7823621	0.7839668	0.7367732	0.9411582	0.7613651
MED_WIDTH_	0.7625795	0.8713244	1.0000000	0.8584730	0.6945719	0.7489014	0.6953178	0.8014425	0.6658846
DIST_WIDTH	0.6538550	0.7063473	0.8584730	1.0000000	0.5970961	0.6519855	0.6256154	0.6499102	0.5597469
PROX_THICK	0.8169184	0.7823621	0.6945719	0.5970961	1.0000000	0.9562966	0.9175047	0.7988801	0.9739558
MED_THICKN	0.8477081	0.7839668	0.7489014	0.6519855	0.9562966	1.0000000	0.9526678	0.7816227	0.9370189
DIST_THICK	0.8102758	0.7367732	0.6953178	0.6256154	0.9175047	0.9526678	1.0000000	0.7411379	0.8998309
PLAT_WIDTH	0.7781210	0.9411582	0.8014425	0.6499102	0.7988801	0.7816227	0.7411379	1.0000000	0.7941236
PLAT_THICK	0.8000076	0.7613651	0.6658846	0.5597469	0.9739558	0.9370189	0.8998309	0.7941236	1.0000000

The following code was used to perform the logistic regression on the numeric variables:

```
# Isolate dataframe numeric variables
log_data <- data[-c(10:14)]

# Revalue Core and Flake to binary factor where Flake = 1, Core = 0
log_data$ARTEFACT_C <- revalue(log_data$ARTEFACT_C, c("Flake"="1", "Core"="0"))

# Set numeric predictors
names(log_data)
numeric_predictors <- c("MASS__G_", "PROX_WIDTH", "MED_WIDTH_", "DIST_WIDTH",
"PROX_THICK", "MED_THICKN", "DIST_THICK", "PLAT_WIDTH", "PLAT_THICK")

set.seed(111)

#split into training (80%) and test (20%)
split <- createDataPartition(log_data$ARTEFACT_C, p = 0.8, list = F)
train <- log_data[split, c(numeric_predictors, "ARTEFACT_C")]
test <- log_data[-split, c(numeric_predictors, "ARTEFACT_C")]

# Train the model
log_model <- glm(
  ARTEFACT_C ~ MASS__G_ + PROX_WIDTH + MED_WIDTH_ + DIST_WIDTH
```

```
  + PROX_THICK + MED_THICKN + DIST_THICK +
    PLAT_WIDTH + PLAT_THICK,
  family = "binomial",
  data = train
)
log_model
summary(log_model)

# Accuracy of train data
lodds <- predict(log_model, type = "link")
preds_lodds <- ifelse(lodds > 0, "1", "0")
# Accuracy
confusionMatrix(as.factor(preds_lodds), train$ARTEFACT_C)

# Make predictions on test data
test_lodds <- predict(log_model, newdata = test, type = "link")
test_preds_lodds <- ifelse(test_lodds > 0, "1", "0")

# Accuracy
confusionMatrix(as.factor(test_preds_lodds), test$ARTEFACT_C)
```

The results of this linear regression can be found in the *Results* section.

3. K-means Cluster Analysis

K-means cluster analysis is an unsupervised learning model, which makes use of a predetermined value (k) to form k centroids, from which that number of clusters is built (James et al, 2013). Each observation belongs to the cluster with the nearest mean. As such, the value of k must be known prior to the analysis. This analysis seeks to investigate whether the two artefact categories will form two separate clusters, so k = 2.

The following code was used to perform the analysis:

```
# Create cluster dataframe
clust_data <- data[-c(10:14)]

# Scale data
dat_scaled <- scale(clust_data[1:9])

set.seed(111)

# Train model
kmeans_res <- kmeans(dat_scaled, centers = 2, nstart = 25) #centers? nst
str(kmeans_res)
fviz_cluster(kmeans_res, data = dat_scaled)
# Plot model
fviz_cluster(kmeans_res,
             data = dat_scaled,
             geom = "point",
             shape = 19,
             alpha = 0)+
  geom_point(aes(colour = as.factor(kmeans_res$cluster),
                 shape = clust_data$ARTEFACT_C))+
  ggtitle("Comparing Clusters and Artefact Class")

# Perform kmeans & calculate ss
total_sum_squares <- function(k){
  kmeans(dat_scaled, centers = k, nstart = 25)$tot.withinss
}
# Define a sequence of values for k
all_ks <- seq(1,20,1)
choose_k <- sapply(seq_along(all_ks), function(i){ #apply to all values
  total_sum_squares(all_ks[i])
})
choose_k_plot <- data.frame(k = all_ks, # dataframe for plotting
                            within_cluster_variation = choose_k)
```

```
head(choose_k_plot)
ggplot(choose_k_plot, aes(x = k, # plot
                          y = within_cluster_variation))+
  geom_point()+
  geom_line()+
  xlab("Number of Clusters (K)")+
  ylab("Within Cluster Variation")

# Evaluate performance
SWC <- function(clusterLabels, dataPoints){
  require(cluster)
  sil <- silhouette(x = clusterLabels, dist = dist(dataPoints))
  return(mean(sil[,3]))
}
```

Results and discussion

1. Naïve-Bayes Classifier (NBC)

The Naïve-Bayes Classifier, run on the test data, returned a **97.7%** accuracy in classifying stone artefacts as either Core or Flake. This is only marginally lower than the **98.5%** accuracy in the training data. The R output for the Naïve-Bayes Classifier are shown in *Figure 2.*

```
Confusion Matrix and Statistics

          Reference
Prediction Core Flake
     Core    25     7
     Flake    0   280

               Accuracy : 0.9776
                 95% CI : (0.9543, 0.9909)
    No Information Rate : 0.9199
    P-Value [Acc > NIR] : 1.306e-05

                  Kappa : 0.8651

 Mcnemar's Test P-Value : 0.02334

            Sensitivity : 1.00000
            Specificity : 0.97561
         Pos Pred Value : 0.78125
         Neg Pred Value : 1.00000
             Prevalence : 0.08013
         Detection Rate : 0.08013
   Detection Prevalence : 0.10256
      Balanced Accuracy : 0.98780

       'Positive' Class : Core
```

```
Confusion Matrix and Statistics

          Reference
Prediction Core Flake
     Core   100    18
     Flake    0  1132

               Accuracy : 0.9856
                 95% CI : (0.9773, 0.9914)
    No Information Rate : 0.92
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9096

 Mcnemar's Test P-Value : 6.151e-05

            Sensitivity : 1.0000
            Specificity : 0.9843
         Pos Pred Value : 0.8475
         Neg Pred Value : 1.0000
             Prevalence : 0.0800
         Detection Rate : 0.0800
   Detection Prevalence : 0.0944
      Balanced Accuracy : 0.9922

       'Positive' Class : Core
```

Figure 2: Naive-Bayes Classifier output for test (left) and train (right) data.

The model was able to accurately classify 100% of the cores, while in both the test and train data, a relatively small proportion of flakes were misclassified as cores. The success in correctly classifying cores may be attributed to the categorical variables, since, while entries for RAW_MATERIAL are not specific to either flakes or cores, OVERHANG_R, INITIATION, BULB_OF_PE and TERMINATION are all 'Absent' for core artefacts and as such should push the classification towards 'core'. It is likely that one or a combination of the numeric variables is also significant in the classification of artefacts. The results of the logistic regression will further elaborate on this.

2. Logistic Regression

Figure 3 shows the output for the logistic regression. The odds of an artefact being classified as a flake or core can be derived from the 'Estimate' column in the output in *Figure 4*. For each estimate, a one-unit increase for each variable is associated with an x increase in the log odds of ARTEFACT_C being a core.

Figure 3 shows the accuracy of the test and train data to be **98.7%** and **99.1%**, respectively. The logistic regression shows that, with categorical variables removed, there is misclassification of both flakes and cores, although the overall accuracy is higher. Figure 4 shows the Receiver Operator Characteristic (ROC) and associated Area Under Curve (AUC). The sensitivity and specificity are both very high, resulting in a high AUC and indicating the effectiveness of the model.

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0  22   1
         1   3 286

               Accuracy : 0.9872
                 95% CI : (0.9675, 0.9965)
    No Information Rate : 0.9199
    P-Value [Acc > NIR] : 1.253e-07

                  Kappa : 0.9097

 Mcnemar's Test P-Value : 0.6171

            Sensitivity : 0.88000
            Specificity : 0.99652
         Pos Pred Value : 0.95652
         Neg Pred Value : 0.98962
             Prevalence : 0.08013
         Detection Rate : 0.07051
   Detection Prevalence : 0.07372
      Balanced Accuracy : 0.93826

       'Positive' Class : 0
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0   94    5
         1    6 1145

               Accuracy : 0.9912
                 95% CI : (0.9843, 0.9956)
    No Information Rate : 0.92
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9399

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9400
            Specificity : 0.9957
         Pos Pred Value : 0.9495
         Neg Pred Value : 0.9948
             Prevalence : 0.0800
         Detection Rate : 0.0752
   Detection Prevalence : 0.0792
      Balanced Accuracy : 0.9678

       'Positive' Class : 0
```

Figure 3: Logistic regression output for test (left) and train (right) data. Core = 0, Flake = 1. Positive class = Core.

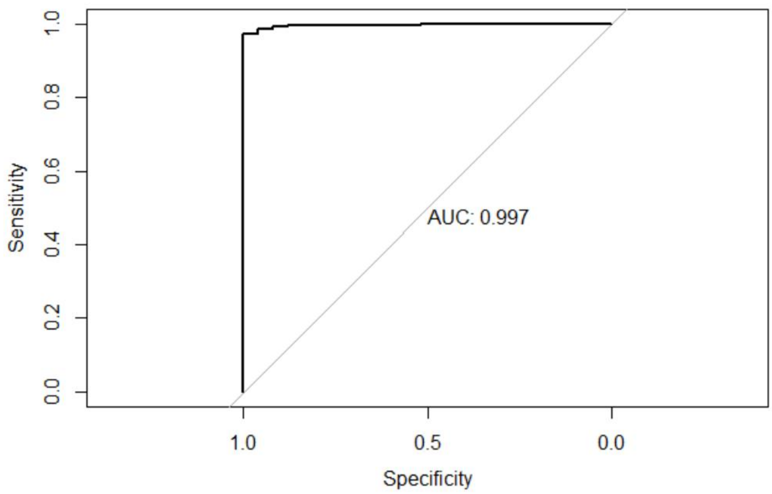

Figure 4: ROC curve showing the relationship between sensitivity and specificity for the test data.

Figure 5 shows the output for the logistic regression while *Table 5* shows all variables in descending order of their impact on the classification.

```
Call:
glm(formula = ARTEFACT_C ~ MASS__G_ + PROX_WIDTH + MED_WIDTH_ +
    DIST_WIDTH + PROX_THICK + MED_THICKN + DIST_THICK + PLAT_WIDTH +
    PLAT_THICK, family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4451   0.0207   0.0308   0.0466   3.6943

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  9.08254    1.41132   6.436 1.23e-10 ***
MASS__G_    -0.01963    0.01298  -1.512   0.1305
PROX_WIDTH   0.02953    0.07276   0.406   0.6849
MED_WIDTH_   0.04276    0.06302   0.679   0.4974
DIST_WIDTH   0.02723    0.04719   0.577   0.5639
PROX_THICK  -0.24626    0.10659  -2.310   0.0209 *
MED_THICKN  -0.05948    0.09792  -0.607   0.5436
DIST_THICK  -0.08848    0.07132  -1.241   0.2148
PLAT_WIDTH  -0.09449    0.05662  -1.669   0.0952 .
PLAT_THICK   0.02318    0.07227   0.321   0.7485
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 696.923  on 1249  degrees of freedom
Residual deviance:  59.988  on 1240  degrees of freedom
AIC: 79.988

Number of Fisher Scoring iterations: 9
```

Figure 3: Logistic regression output, showing coefficients to be used for further analysis.

Proximal thickness (the thickness at the thickest point) is by far the most significant variable in determining the odds of an artefact being classified either a flake or a core. PROX_THICK has p-value of **0.0209** and an Estimate of **-0.24626**. For a one-unit increase in proximal thickness, we can expect a **24%** increase in the odds of an artefact being classified as a core, given than the exponent of Estimate is **1.24**. This is a logical result, since cores will naturally be thicker than the flakes that are removed. However, this is not always the case. The k-means cluster analysis will visualise the data in terms of two artefact type clusters.

Table 5: Ban Rai data variables in order of impact on classification (descending).

	overall	names
5	2.3103973	PROX_THICK
8	1.6688319	PLAT_WIDTH
1	1.5122369	MASS__G_
7	1.2405555	DIST_THICK
3	0.6785990	MED_WIDTH_
6	0.6074062	MED_THICKN
4	0.5770652	DIST_WIDTH
2	0.4057926	PROX_WIDTH
9	0.3206713	PLAT_THICK

3. K-means Cluster Analysis

Figure 6 shows the Ban Rai data plotted using k = 2, since it was expected that the ARTEFACT_C data would form two reasonably distinct clusters. *Figure 7* shows that this is in fact the optimal number of clusters, with the ‘elbow’ located at k = 2. The elbow is where the line becomes shallower and indicates the most appropriate value for k.

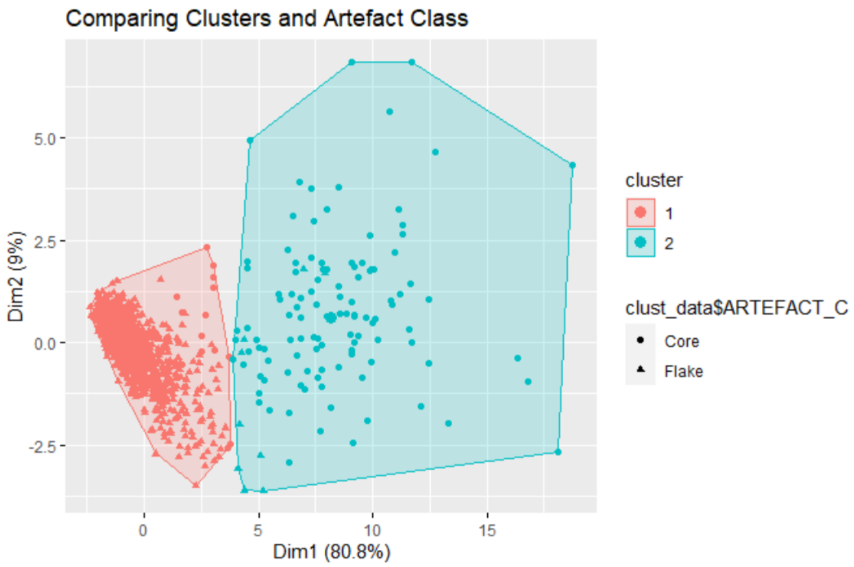

Figure 4: K-means cluster visualisation where k=2.

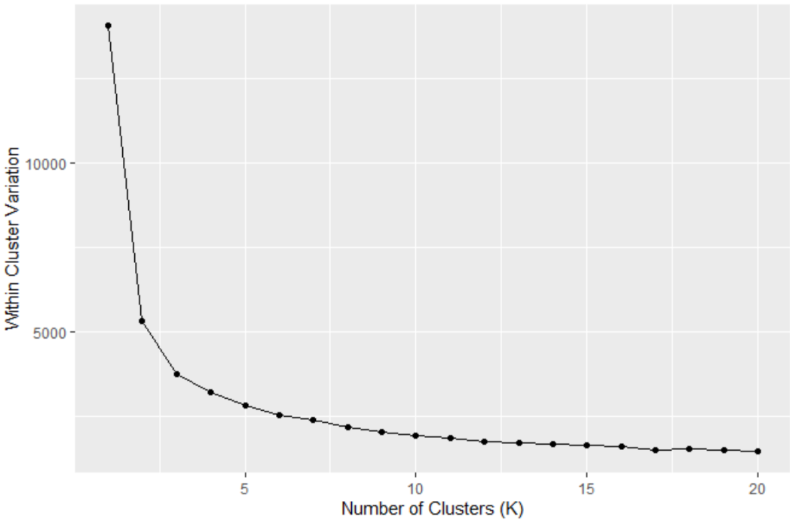

Figure 5: Within-cluster variation, plot shows 'elbow' at k = 2.

Figure 8 shows that the Silhouette Width Criterion (SWC) value is **0.80**. The silhouette value indicates the similarity of an object is to its assigned cluster, compared with any other clusters in the model. The value can range between −1 to +1; a value close to 1, as is observed, indicates that the object is well-suited to the cluster it has been assigned to (Machado and Santos, 2007).

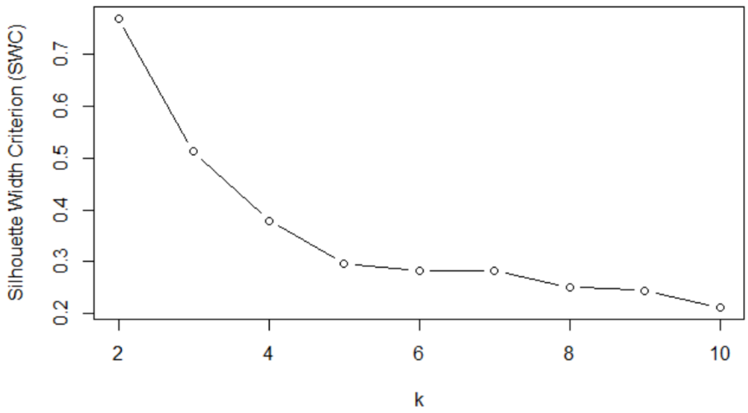

Figure 6: Silhouette Width Criterion showing high SWC at k=2

Finally, *Figure 9* shows the Within Cluster Sum of Squares value at a reasonable **62%**. This value is a measure of the total variance in the data that is explained by the clustering, or how tightly packed the points are within the clusters (Krider and Putler, 2012).

```
Within cluster sum of squares by cluster:
[1] 3257.823 2077.321
 (between_SS / total_SS =  62.0 %)
```

Figure 7: Output of k-means.

Taken together, these results demonstrate that the two artefact classes (core and flake) do form two separate clusters, although some flakes do fall into the core cluster and vice versa. An explanation for this is that, since the logistic regression showed that overall size is the main determinant of whether an artefact is classified as either a flake or core, unusually large flakes may fall into the 'core' cluster, and unusually small cores may fall into the 'flake' cluster.

The alternative hypothesis, that observational lithic analysis data can be used to predict artefact type, can be accepted. This outcome is useful, because it demonstrates that machine learning algorithms are able to correctly classify stone artefacts into flakes and cores with at least **97%** accuracy. It is therefore reasonable to deduce that further predictions can be made, if the data are available, to identify different types of artefact within these categories. For example, flakes made by different people or using different techniques and technologies. This may lead to the realisation that artefacts previously classified as the same type are actually different, and the implication of this is that archaeology can benefit greatly from the application of machine learning algorithms to already available data.

Conclusions

This report has demonstrated that in a dataset of lithic artefacts, it is possible to use machine learning algorithms to differentiate between flake and core type artefacts. This aligns with the alternative hypothesis presented earlier. Because overall size (especially thickness) is such a strong determinant of stone artefact classification, it must be remembered that particularly large flakes may be misclassified as cores. Despite this, the Naïve-Bayes model returned a **97.7%** prediction accuracy on test data, and the linear regression retuned **99.1%**. Furthermore, the core and flake data were statistically different enough to be able to form two separate clusters.

These results are useful because they have implications for the broader use of machine learning in archaeology; machine learning algorithms can potentially speed up or verify the observational classification process. Further research is recommended, since these same principles could be applied to other types of artefact, which may shed light on heretofore unanswered research questions. The potential of applying machine learning algorithms to archaeological assemblages, however, is limited to the amount of data that are readily available, so archaeologists who wish to apply machine learning techniques may benefit from identifying new aspects of artefacts to measure and take note of, in order to maximise the data available for analysis.

References

Clarkson, C., & O'Connor, S. (2006). An introduction to stone artefact analysis. In A. Paterson & J. Balme (Eds.), *Archaeology in practice: A student guide to archaeological analyses*, 159-206. Wiley-Blackwell.

Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning: With applications in R*. Springer.

Krider, R. E., & Putler, D. S. (2012). *Customer and business analytics: Applied data mining for business decision making using R*. CRC Press.

Machado, J, & Santos, M. (2007, December 3-7). *Progress in Artificial Intelligence* [Paper presentation]. 13th Portuguese Conference on Artificial Intelligence. Berlin, Germany.

Marwick, B. (2008). *Stone artefacts and human ecology at two rockshelters in Northwest Thailand* [PhD Thesis, Australian National University]. Harvard Dataverse. https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/UHIFDK

Marwick, B., 2008, Stone artefacts and human ecology at two rockshelters in Northwest Thailand [Dataset]. https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/UHIFDK

Midi, H., Sarkar, S., & Rana, S. (2010). Collinearity diagnostics of binary logistic regression model. *Journal of Interdisciplinary Mathematics*, *13*(3), 253-267.

RStudio Team (2020). RStudio: Integrated Development for R. RStudio, PBC, Boston, MA URL http://www.rstudio.com/.

R Packages

caret
cluster
dplyr
factoextra
ggplot2
ISLR
naivebayes
naniar
plyr
pROC
tidyr

Appendices

Appendix 1

From Clarkson and O’Connor, 2007

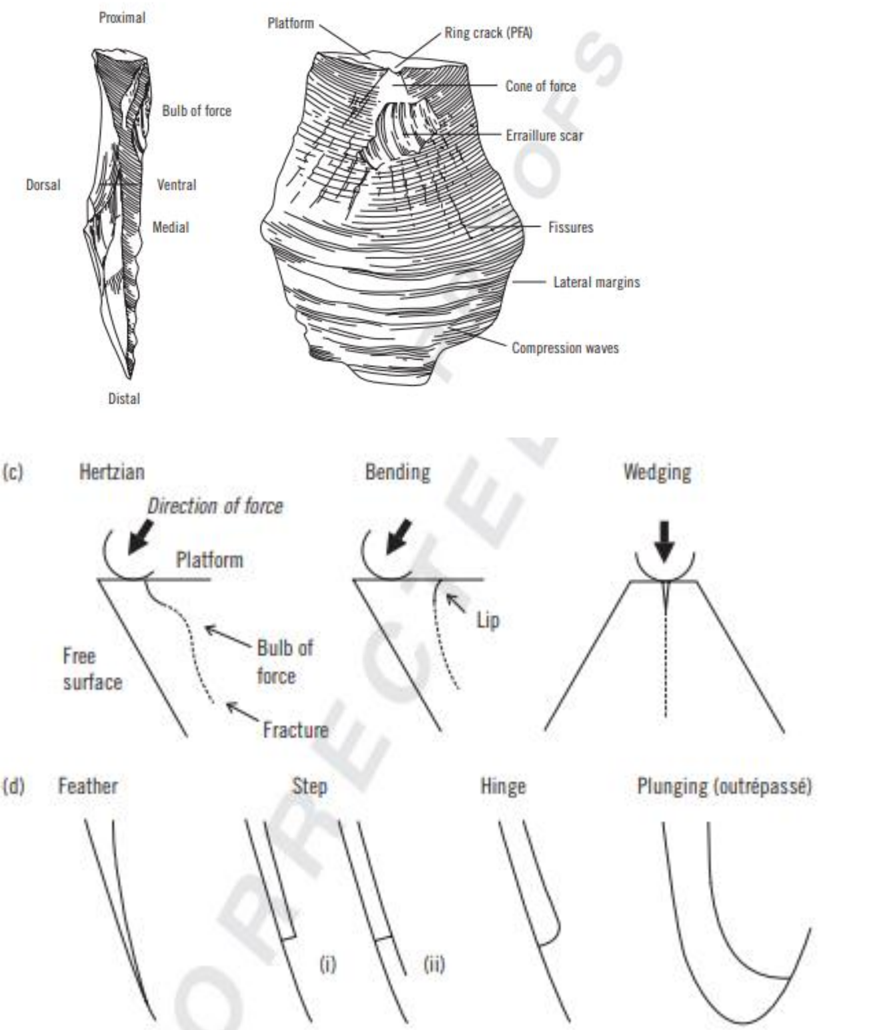

Figure 6.2. Types and features of initiation and termination: (a) fracture forces; (b) Hertzian cones; (c) fracture initiations; (d) termination types. With kind permission from Chris Clarkson. Adapted from Cotterell and Kamminga (1987); Andrefsky and Bindon (1995).

Appendix 2

R.version

```
library(ISLR, warn.conflicts = F, quietly = T)
library(caret, warn.conflicts = F, quietly = T)
library(dplyr, warn.conflicts = F, quietly = T)
```

```
library(cluster, warn.conflicts = F, quietly = T)
library(factoextra, warn.conflicts = F, quietly = T)
library(ggplot2)
library(naniar)
library(naivebayes)
library(tidyr)
library(plyr)
library(pROC)

########################################################################################
########################################
#                                          LOAD DATA                                       #
########################################################################################
########################################

# Read csv file
data <- read.csv("Ban_Rai_Area_3_lithics.csv")


########################################################################################
########################################
#                                        DATA CLEANING                                       #
########################################################################################
########################################

# View column names as list for removal
names <- names(data)
names <- as.data.frame(names)

# Remove superfluous columns
data <- data[-c(1:5,7:9,15,17,20:22,26,29:71,73:164)]

# Remove invalid data
data <- na.omit(data)

# Fill empty cells with 'Absent'
data$OVERHANG_R <- sub("^$", "Absent", data$OVERHANG_R)
data$INITIATION <- sub("^$", "Absent", data$INITIATION)
data$BULB_OF_PE <- sub("^$", "Absent", data$BULB_OF_PE)
data$TERMINATIO <- sub("^$", "Absent", data$TERMINATIO)


# Remove the two 'Broken Cbl' entries as they equate to 'retouched flake' and there is no distinction
in
# the data between this and 'flake'
data <- data[- grep("Broken Cbl", data$ARTEFACT_C),]


# Set as factor/Standardise abbreviations within catgorical variables
data$OVERHANG_R <- as.factor(data$OVERHANG_R)
levels(data$OVERHANG_R)

data$INITIATION <- as.factor(data$INITIATION)
```

```
levels(data$INITIATION)

data$BULB_OF_PE <- as.factor(data$BULB_OF_PE)
levels(data$BULB_OF_PE)

data$TERMINATIO <- as.factor(data$TERMINATIO)
levels(data$TERMINATIO)

data$RAW_MATERI <- as.factor(data$RAW_MATERI)
data$RAW_MATERI <- revalue(data$RAW_MATERI, c("Blk qtzt"="Black Quartzite", "Rd qtzt"="Red
Quartzite", "Yell qtzt"="Yellow Quartzite", "hi-q qtzt"="Quartzite", "vfg Qtzt"="Quartzite", "Fe
qtzt"="Quartzite", "Chert blk"= "Chert", "Blk chert"="Chert", "Blk Chrt"="Chert", "chert"="Chert",
"Ind mudst"="Shale", "Diorite 1"="Diorite", "Chalc"="Chalk", "Limest"="Limestone"))
levels(data$RAW_MATERI)

# Set response variable as factor
data$ARTEFACT_C <- as.factor(data$ARTEFACT_C)
levels(data$ARTEFACT_C)
#########################################################################################
#######################################
#                                   NAIVE-BAYES CLASSIFICATION                                   #
#########################################################################################
#######################################

# View proportions of categories within response variables
table(data$ARTEFACT_C) %>% prop.table()



# Preliminary visualisation: Density plot
numeric_data <- data[-c(10:14)]
numeric_data %>%
  reshape2::melt(id.vars = "ARTEFACT_C") %>%
  ggplot(aes(value, colour = ARTEFACT_C))+
  labs(title="Artefact Class density distributions")+
  geom_density(show.legend = TRUE)+
  facet_wrap(~variable, scales = "free")



# Ensure reproducibility
set.seed(111)

# Split data into test and train
split <- createDataPartition(data$ARTEFACT_C, p = 0.8, list = FALSE)
train <- data[split, ]
test <- data[-split, ]

# Train model on train data
nB_model <- naive_bayes(ARTEFACT_C ~ ., data = train, laplace = 1, kernel = TRUE)

# View predictions and accuracy on test data
pred <- predict(nB_model, test, type = "class")
```

```
confusionMatrix(pred, test$ARTEFACT_C)

# View predictions and accuracy on train data
pred <- predict(nB_model, train, type = "class")
confusionMatrix(pred, train$ARTEFACT_C)


##########################################################################################
########################################
#                                    LOGISTIC REGRESSION                                    #
##########################################################################################
########################################

# Isolate dataframe numeric variables
log_data <- data[-c(10:14)]

# Investigate collinearity
cor_tab <- cor(log_data[1:9])

# Revalue Core and Flake to binary factor where Flake = 1, Core = 0
log_data$ARTEFACT_C <- revalue(log_data$ARTEFACT_C, c("Flake"="1", "Core"="0"))

# Set numeric predictors
names(log_data)
numeric_predictors <- c("MASS__G_", "PROX_WIDTH", "MED_WIDTH_", "DIST_WIDTH",
"PROX_THICK", "MED_THICKN", "DIST_THICK", "PLAT_WIDTH", "PLAT_THICK")

set.seed(111)

#split into training (80%) and test (20%)
split <- createDataPartition(log_data$ARTEFACT_C, p = 0.8, list = F)
train <- log_data[split, c(numeric_predictors, "ARTEFACT_C")]
test <- log_data[-split, c(numeric_predictors, "ARTEFACT_C")]

# print number of observations in test vs. train
c(nrow(train), nrow(test))

# Proportions of core vs flake in train data
table(train$ARTEFACT_C) %>% prop.table()

# Proportions of core vs flake in test data
table(test$ARTEFACT_C) %>% prop.table()

# Train the model on the train data
log_model <- glm(
  ARTEFACT_C ~ MASS__G_ + PROX_WIDTH + MED_WIDTH_ + DIST_WIDTH
  + PROX_THICK + MED_THICKN + DIST_THICK +
    PLAT_WIDTH + PLAT_THICK,
  family = "binomial",
  data = train
)
log_model
summary(log_model)
```

```
# Accuracy of train data
lodds <- predict(log_model, type = "link")
preds_lodds <- ifelse(lodds > 0, "1", "0")
# Accuracy
confusionMatrix(as.factor(preds_lodds), train$ARTEFACT_C)

# Make predictions on test data
test_lodds <- predict(log_model, newdata = test, type = "link")
test_preds_lodds <- ifelse(test_lodds > 0, "1", "0")
# Accuracy
confusionMatrix(as.factor(test_preds_lodds), test$ARTEFACT_C)


# Order variables in order of their impact on classification
summary(log_model)

mod_fit <- glm(ARTEFACT_C ~ MASS__G_ + PROX_WIDTH + MED_WIDTH_ + DIST_WIDTH
          + PROX_THICK + MED_THICKN + DIST_THICK +
            PLAT_WIDTH + PLAT_THICK, data=train, family=binomial(link = 'logit'))

imp <- as.data.frame(varImp(mod_fit))
imp <- data.frame(overall = imp$Overall,
            names   = rownames(imp))
importance <- imp[order(imp$overall,decreasing = T),]
importance

# Plot ROC curve
test_prob = predict(mod_fit, newdata = test, type = "response")
test_roc = roc(test$ARTEFACT_C ~ test_prob, plot = TRUE, print.auc = TRUE)
###########################################################################################
#########################################
#                          K-MEANS CLUSTER ANALYSIS                         #
###########################################################################################
#########################################

# Create cluster dataframe
clust_data <- data[-c(10:14)]

# Scale data
dat_scaled <- scale(clust_data[1:9])

set.seed(111)

# Train model
kmeans_res <- kmeans(dat_scaled, centers = 2, nstart = 25)
str(kmeans_res)
fviz_cluster(kmeans_res, data = dat_scaled)

# Accuracy
kmeans_res

# Plot model
```