Data Science Challenge: Inferring Gender from Transaction Data

Warren Mackay-Smith

05/05/2019

## Executive Summary

The purpose of this paper is to explore potential statistical models that **infer gender** of our customers in order to **power meaningful personalisation** on and off site for our customers. Male and Female shoppers exhibit different shopping behaviors and attitudes, this is reflected in both their path to purchase but also ultimately in the product(s) they choose to buy. By being able to better craft our communication and content based on these attributes and key moments, we will be able to drive superior brand equity through stronger relevance, minimize customer frustration and most importantly improve conversion on site. Assuming an annual revenue of 250M Aud and an average conversion rate of 3%, with all others things being equal, a 0.5% percent point improvement in conversion repersents a 40M Aud opportunity for **TheIconic**.

In this paper, we explore some of the transaction data from the TheIconic, and how to prepare this data in order to create a **‘gender inference’** model that can drive personalisation. As we do not have a ‘Gender Flag’ in the data, we used the ‘categories’ of the items purchased (female items, male items, unisex items) to act as a proxy for Male or Female. Two different models were then been created to predict, with a high level of accuracy (>90%), the inferred gender of the customer. A **Logistic Regression** and **Deep Learning AI model** were built with machine learning techniques to train and test the data. The selected final models had a similar predication accuracy, and should be applied to the wider organisation data set to understand thier ability to successfully predict out of sample. This paper captures the techniques and code used.

It is important to note that the benefit can only be realized through co-ordinated **business commercialization efforts**. This would require leveraging some of the predicted models’ outcomes to conduct A/B testing with strong test and control mechanisms, to understand the potenital business impact. Some of the areas where the gender inference model could make an impact on performance would be around tailored email communication, content on the site, product placement and offers, as well as media off site. A fast iterative test and learn culture and approach will then validate the value of each % point improvement from the modeling to the point where we are able to **maximize the revenue** and **life time value** of our customers through such efforts.

## The Challenge

The original challenge can be found here <https://github.com/theiconic/datascientist>. There are four sections to the Data Science challenge, namely

1. SQL – use SQL to answer a few key questions about a given data set
2. CLEAN – Prepare the data for modelling
3. BUILD – Build a data model to infer gender
4. DELIVER – Share results in a business-friendly report

## Stage 1: SQL

*This stage deals with running queries on a database (SQLite) provided. The ask is to unhash the SQL data and answer the following questions (note: it is OK to ignore the underlying errors in the data)*

**1. What was the total revenue to the nearest dollar for customers who have paid by credit card?**

SELECT round(SUM (revenue)) total FROM customers WHERE cc\_payments = 1

Total revenue for credit card users is = $503,772,282

**2. What percentage of customers who have purchased female items have paid by credit card?**

SELECT count(female\_items) FROM customers WHERE female\_items>0 GROUP BY cc\_payments

Results show the number of females items that **were** paid by card was 11,935, whilst the total number of females items **not** paid by card was 22,588. Therefore, 65% of all female items **were** paid for by credit card.

**3. What was the average revenue for customers who used either iOS, Android or Desktop?**

SELECT round(AVG (revenue),2) FROM customers   
WHERE desktop\_orders>0 OR msite\_orders> 0 OR ios\_orders>0

The average revenue by customer who used iOS, Android or Desktop was $1305.93

**4. We want to run an email campaign promoting a new men’s luxury brand. Can you provide a list of customers we should send to?**

A **simple mailing list** can been created based on the customer being part of the email list (has given consent) and has purchased at least one male item (content relevance) in the last 6 months (some level of recency).

SELECT customer\_id FROM customers  
WHERE is\_newsletter\_subscriber = "Y" AND male\_items > 0 AND days\_since\_last\_order <180

Displaying records 1 - 10

|  |
| --- |
| customer\_id |
| 18923c9361f27583d2320951435e4888 |
| af2a73b69cd9d10facb3018df8d5a17a |
| 55ab0a4916f94eeb2b0d588d46265708 |
| b485c1df91627cbea977008eb126a4b0 |
| 3bff0c40e43263dde3db43d0304434f0 |
| 52be990db43844a01b5141661519db13 |
| 58b983187e368d872083a23471c11c6a |
| 71436a46c49a576cb9045484edb911a1 |
| 1638696d6430178085b96c59bc87d95a |
| cd16a36c4184646e5f403219a1c5eedb |

This approach has created a list of **293 customers**, a sample of the first ten customers has been shared in the above table. Whilst this is a simple list, further refinement could be applied by looking at aspects such as segmenting spend levels, previous campaign responses and identifying purchase affinities. This would improve the quality of the list and the overall effectiveness of the campaign whilst minimizing annoyance for our customers through over communication.

## Stage 2: CLEAN

*Unhash the data (test\_data.zip) using the secret key, extract it and most importantly clean it and put it in a form you can use. We have also “intentionally” corrupted two columns in this file - two columns that might look correct but are not correct. They need “some correction” to be useful.*

First task is to unhash and load the data (which has been provided in a json format) and create a working table called “fashion”.

fashion <- fromJSON("data 2.json")

An example of the data and some useful statistics from “fashion” can be created using the below code, as well as plotting each of the variables…

head(fashion)  
summary(fashion)

This immediately highlights several variables that need adjustments as well as some issues in the underlying data. 9 areas have been addressed in this section

* ‘is\_newsletter\_subscriber’
* handling of missing Values in data set
* ‘Days\_since\_last\_order’
* ‘average\_discount\_used’
* creating time lapse since first and last order
* handling of ‘Null Values’ in the data
* handling of extreme outlier for orders
* no gender flag
* normalization

**1. ‘is\_newsletter\_subscriber’**

This variable is currently in a character format, namely “Y” for yes or “N” for no. This needs to be converted into a numeric format in order to be useful in our models. We do this by creating a new **dummy variable** that contains values, replacing 0 for **“No” and 1 for “Yes”**. This new variable is then loaded into our “fashion” data set under the name subscriber.

fashion$subscriber = ifelse(fashion$is\_newsletter\_subscriber == 'Y',1,0)

**2. Missing values in data set**

There is a significant number of missing data values in the **‘coupon\_discount\_applied’** variable. These missing values could be explored further and replaced with proxy data (such as the mean value), however we have chosen to omit these data points from the model and thereby have created a new data set “cleanfashion”.

cleanfashion <- na.omit(fashion)

**3. ‘days\_since\_last\_order’**

Exploring the raw data shows that the maximum number of days since the last order is **51840** which is equivalent to **142 years**. Given that **TheIconic** was founded in 2011, this is an obvious error. Further examination implies that the data has been multiplied by 24. This is fixed using the following code

cleanfashion$last\_order\_days <- cleanfashion$days\_since\_last\_order / 24

**4. ‘average\_discount\_used’**

Through exploration it also seems that the average discount has been multiplied by 10000 and therefore is at a different scale to the other discount variables. Adjusting this, as per below therefore to make it useful for modeling.

cleanfashion$av\_update\_discount\_used <- cleanfashion$average\_discount\_used / 10000

**5. Creating time since first and last order**

A variable that might be of interest is the difference between first and last order to symbolize how long a customer has been purchasing with **TheIconic**.

cleanfashion$diff\_days <- cleanfashion$days\_since\_last\_order - cleanfashion$days\_since\_first\_order

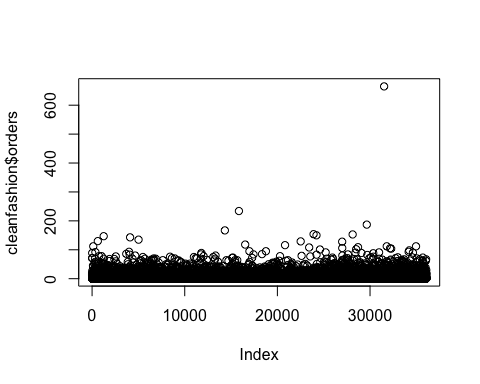
**6. Null values in the Data**

Further exploration also identified several NULL (not valid) values in the data. Further exploration should be done in the data to understand why these exist. By deleting these NULL variables from macc\_items, av\_update\_discount\_used and diff\_days it will enable us to run our chosen models.

cleanfashion$diff\_days <- NULL  
cleanfashion$macc\_items <- NULL  
cleanfashion$av\_update\_discount\_used <- NULL

**7. Extreme outlier value for orders**

By plotting the variables we also notice that there is one instance of an extreme outlier in the number of orders. This would warrant further investigation however for this task it has been removed from the data set. A plot of the data can be seen here.



We then can create an adjusted data set for modeling by identifying and removing the outlier in this case

cln\_fashion <- cleanfashion [-31519,]

**8. No Gender Flag**

Within the data there is **no gender flag** and as such we are ‘flying blind’. The closest proxy we have to gender is Female item or Male Item indicating if a customer has made a purchase in that category. It should be noted however that this is not a perfect representation as Females **could** also buy Male items for their partners and vice versa. To understand this more we can explore the overlap of Female, Male and Unisex items to see their interactions. This is done analyzing a range of **dummy variables** designed to capture each mix of purchases (male, female or unisex):

dfmale <- ifelse(cln\_fashion$male\_items > 0, 1, 0)  
dffemale <- ifelse(cln\_fashion$female\_items > 0, 10, 0)  
dfunisex <- ifelse(cln\_fashion$unisex\_items>0, 100, 0)  
dfmixed <- dffemale + dfmale + dfunisex

Identifying the amount of customers in each group can then be found and can be repeated for each permutation (female only, male + female etc):

maleonly <- length(which(cln\_fashion$dfmixed == 1))

Using this knowledge and coupled with the writer’s own experience in his household; we can then build a hypothesis which stipulates the following to be the basis of the models to be created:

* Male only purchases are most likely made by males
* Male and Unisex purchase are most likely made by males
* All other cases are most likely to be female (female only or female buying on behalf of partner etc)

We then create a **proxy variable** which we will call **“genderprox”**. This is created by using the dfmixed variable and identifying the male signals (male = 1, male + unisex = 101) and making all those values = 0 and then by default all female values = 1:

genderprox <- ifelse(dfmixed == 1, 0, ifelse(dfmixed==101,0,1))  
cln\_fashion$genderprox <- genderprox

**9. Normalization**

With the chosen models (explained in next section) we will also need to normalize our data (by creating similar orders of magnitude for our variables that will produce better output). This can be done via several methods, in this case we have used a normalize function that takes the min and max of the data variable to standardize the values between 0 and 1. This function is shown below and then run on the relevant data points to create our final database for modeling:

normalize <- function(x) { return ((x - min(x)) / (max(x) - min(x)))}

N\_Fashion <- as.data.frame(lapply(cln\_fashion[5:45], normalize))

Our **clean data set** is now ready for modeling and has 36073 data points.

## Stage 3 - BUILD

*Build a deep learning model (preferably) or any other model that suitably answers this question and predict the inferred gender using the features provided and deriving more features at your end. Remember, there is no gender flag, so you are flying blind here.*

The problem we are trying to solve is to create a **prediction** for gender, i.e is the customer Male or Female. This is a binary classification problem where we aim to develop a model that can predict one of the two possible outcomes.

Before we start we need to establish a baseline for comparison and to ensure that our model improves our ability to predict. Let’s assume that we merely classified all customers **as female**; what therefore is the probability we would get that right? This should form the minimum that our model needs to then improve on. This baseline can be calculated by using the number of instances of **male and female** in the genderprox variable (0=male & 1=female):

table(cln\_fashion$genderprox)

##   
## 0 1   
## 7325 28748

**Therefore, based on this data, by just assuming that all of the customers were ‘female’, we would be correct 79% (28748 / 36073) of the time. This is our benchmark and our model needs to better this result at a minimum**.

Turning to the modeling approach we have used two different models to infer gender:

* Logistic Regression - where our goal is to infer **Gender** based on Machine Learning approach
* Deep Learning - where our goal is to infer **Gender** based on a Neural Network approach

**1. Machine Learning via Logistic Regression**

Logistic Regression is a common model used where the output represents a probability. Logit models are used to help identify when the ‘odds’ favor a positive (value closer to 1) or a negative outcome (value closer to 0) and therefore can be used in our classification example to identify Male or Female.

In order to run our Machine Learning approach we will need to create a *TRAINING* and a *TEST* data set. This helps us to be able to predict and assess our model with the data we have. The predictive power can then be tested further on the full database. In order to create these sets, we start with a random number (set.seed) that is then used to split the data between test and train rows and stores them into two data sets “FashionTrain” and “FashionTest”…

set.seed(67)  
split = sample.split(N\_Fashion$genderprox, SplitRatio = 0.75)  
FashionTrain = subset(N\_Fashion, split == TRUE)  
FashionTest = subset(N\_Fashion, split == FALSE)

nrow(FashionTrain)

## [1] 27055

nrow(FashionTest)

## [1] 9018

As shown in the above tables this code splits the data into two sets (based on the assigned 75% split)

* FashionTrain which has 27055 data points - this will be used to train the data.
* FashionTest which then has 9018 data points - this will be used to predict our outcomes.

Using the **FashionTrain** data set we can then run the Logistic Regression model (glm function in R). After several iterations of including / excluding certain variables and reviewing the levels of significance for each key variables a final model is obtained. The model and results are shown below:

FSHLOG = glm (formula = genderprox ~ msite\_orders + returns + home\_orders + items + wapp\_items + wftw\_items + sacc\_items + mspt\_items + mftw\_items + orders + mapp\_items + average\_discount\_used + paypal\_payments + cc\_payments, family = binomial, data = FashionTrain)  
summary(FSHLOG)

##   
## Call:  
## glm(formula = genderprox ~ msite\_orders + returns + home\_orders +   
## items + wapp\_items + wftw\_items + sacc\_items + mspt\_items +   
## mftw\_items + orders + mapp\_items + average\_discount\_used +   
## paypal\_payments + cc\_payments, family = binomial, data = FashionTrain)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -8.4904 0.0000 0.0002 0.1671 5.5829   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.2115 0.1104 1.916 0.0553 .   
## msite\_orders 11.2029 5.9768 1.874 0.0609 .   
## returns 76.6859 11.0660 6.930 4.21e-12 \*\*\*  
## home\_orders 21.9636 5.5660 3.946 7.95e-05 \*\*\*  
## items 203.5992 11.0279 18.462 < 2e-16 \*\*\*  
## wapp\_items 1771.8218 69.1009 25.641 < 2e-16 \*\*\*  
## wftw\_items 580.0829 17.0785 33.966 < 2e-16 \*\*\*  
## sacc\_items 6.3391 2.0083 3.156 0.0016 \*\*   
## mspt\_items -20.6594 1.2861 -16.063 < 2e-16 \*\*\*  
## mftw\_items -74.5503 2.6697 -27.925 < 2e-16 \*\*\*  
## orders 35.4220 7.8759 4.498 6.87e-06 \*\*\*  
## mapp\_items -97.5551 3.4436 -28.329 < 2e-16 \*\*\*  
## average\_discount\_used -0.5132 0.1131 -4.538 5.67e-06 \*\*\*  
## paypal\_payments -0.2335 0.1044 -2.236 0.0253 \*   
## cc\_payments -0.2212 0.1052 -2.104 0.0354 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 27305 on 27054 degrees of freedom  
## Residual deviance: 10401 on 27040 degrees of freedom  
## AIC: 10431  
##   
## Number of Fisher Scoring iterations: 13

This model indicates that factors such as specific women’s items, the total number of items, home orders, returns and totaol orders are **likely to have a positive impact on the probability it is a female purchaser**. Women’s apparel and footwear are two of the strongest indicators. Device and payment type can also have a smaller impact on infering gender.

This model had the lowest AIC value (similar measure to Rsquared but where the lowest value is best) when compared across different iterations and therefore is considered the best approach. We would then want to see how well this model can predict our proxy gender measure. A **confusion matrix** helps to interpret the accuracy and sensitivity of the model.

The confusion matrix can be recreated through the following code:

predictTrain = predict(FSHLOG, type="response")  
table(FashionTrain$genderprox, predictTrain >0.5)

##   
## FALSE TRUE  
## 0 4694 800  
## 1 1647 19914

It has 4 possible outcomes:

+**True positives (TP)**: We predicted Female when the customer actually is Female **(19914 instances)**

+**True negatives (TN)**: We predicted Male when the customer actually is Male **(4694 instances)**

+**False positives (FP)**: We predicted Female however the customer is actually Male **(800 instances)**

+**False negatives (FN)**: We predicted Male however the customer is actually Female **(1647 instances)**

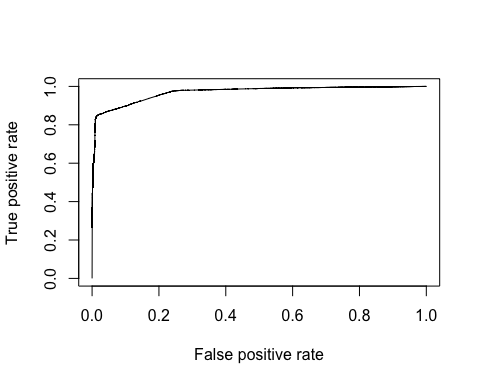
Using this we can calculate the **accuracy of the model** by taking the sum of correctly predicted cells (FALSE & 0 = Correct Male prediction) and (TRUE & 1 = correct Female prediction) and then dividing by all predictions:

(4694 + 19914) / (4694 + 19914 `+ 1647 + 800) = 90.9% accuracy *Please note that the numbers may change from the example given due to the random number starting point and re-running the code - however the result should stand true*

Overall this model is already a strong improvement to our original baseline of 79%. We can do further work to understand the performance of this model in predicting specifically female (right vs wrong) in the data or specifically male - this is done by testing the sensitivity and specificity of the model.

Another way to improve the model is to adjust the **threshold level**. This impacts the level at which a prediction is considered TRUE or FALSE, i.e at what level should a prediction be classified as 0 (Male) or 1 (Female). The default is set a 0.5 indicating that below the 0.5 level the prediction should be classified as = 0 and above 0.5 level it should be classified as = 1. Exploring the ROC (Receiver Operating Characteristic) curve can help identify at what level the threshold should be set.

plot(ROCpref)



Exploring the chart in R shows that a threshold of between **0.3 -0.4** may be a better choice. Using this and reprinting the classification model and evaluating the accuracy gives the following:

table(FashionTrain$genderprox, predictTrain >0.4)

##   
## FALSE TRUE  
## 0 4115 1379  
## 1 490 21071

This now has an accuracy of 93% and is the optimal Logistic Regression model based on the genderprox variable. This model has a tendency to have a higher accurarcy for predicting women than for men. This should be explored further.

Finally we can predict our data using our **FashionTest data set** and then re-creating the confusion table to test for accuracy

predictTest = predict(FSHLOG, type = "response", newdata = FashionTest)  
table(FashionTest$genderprox,predictTest >= 0.4)

##   
## FALSE TRUE  
## 0 1398 433  
## 1 166 7021

The test set also has an accuracy of 93%. Looking at the split between Male and Female shows the following

* Female is **correctly predicted 97%** of the time 7021 / (7021+166)
* Male is **correctly predicted 76%** of the time 1398 / (1398+433)

**2. Deep Learning via a Neural Network to infer Gender**

Given that significant data preparation has been done (cleansed, normalized and split to train / test) we can also explore an alternative model. This time we will look at a **Deep Learning AI Neural Network** approach. Similar to our brain, a neural network leverages a range of ‘neurons’ that communicate to each other and find patterns in the data that traditional methods some times can not find.

In this example, we can leverage the previously modelled data to create a fairly **basic neural network** with the default setting of (2,1) hidden layers. In order to achieve optimization we had to increase the iteration to **0.1** from the intial 0.01 level.

Running the Neural Network leads to the following model, *please note given the model and computation required results may take some time to load…*

nn <- neuralnet(genderprox ~ items + returns + wapp\_items + cc\_payments + wftw\_items + sacc\_items + mspt\_items + mftw\_items + orders + mapp\_items, data=FashionTrain, hidden=c(2,1), linear.output=FALSE, threshold=0.1)  
nn$result.matrix

## [,1]  
## error 6.533919e+02  
## reached.threshold 9.139605e-02  
## steps 2.250200e+04  
## Intercept.to.1layhid1 -1.891416e+00  
## items.to.1layhid1 -7.338354e+01  
## returns.to.1layhid1 3.982273e+00  
## wapp\_items.to.1layhid1 -9.280634e+02  
## cc\_payments.to.1layhid1 -9.362749e-02  
## wftw\_items.to.1layhid1 -2.477357e+02  
## sacc\_items.to.1layhid1 1.361121e+00  
## mspt\_items.to.1layhid1 4.243914e+00  
## mftw\_items.to.1layhid1 1.557629e+01  
## orders.to.1layhid1 -2.023401e+01  
## mapp\_items.to.1layhid1 3.047529e+01  
## Intercept.to.1layhid2 1.536817e+00  
## items.to.1layhid2 1.657224e+02  
## returns.to.1layhid2 -6.180171e+01  
## wapp\_items.to.1layhid2 -8.499155e+02  
## cc\_payments.to.1layhid2 1.529077e-01  
## wftw\_items.to.1layhid2 -2.878617e+02  
## sacc\_items.to.1layhid2 -6.907838e+00  
## mspt\_items.to.1layhid2 2.869068e+01  
## mftw\_items.to.1layhid2 1.472630e+02  
## orders.to.1layhid2 -1.259116e+00  
## mapp\_items.to.1layhid2 1.792560e+02  
## Intercept.to.2layhid1 2.817277e+00  
## 1layhid1.to.2layhid1 -4.982972e+00  
## 1layhid2.to.2layhid1 -2.367140e+00  
## Intercept.to.genderprox -1.361059e+01  
## 2layhid1.to.genderprox 2.618913e+01

plot(nn)

Once the data is plotted the left hand ‘nodes’ show the input data whilst the lines and arrows show the weights assigned to these inputs. The middle nodes represent the model learning the impact of these variables (and ultimately assigning a bias level) which determines the output displayed on the right hand side. The network learns by passing the data back and forth through the nodes until the model determines the right values. In some ways this is a black box approach to modelling with the ultimate test of performacne being the final choosen models ability to predict gender correctly.

We can now create prediction values from the model:

temp\_test <- subset(FashionTest, select = c("items", "returns", "cc\_payments", "wapp\_items", "wftw\_items", "sacc\_items", "mspt\_items", "mftw\_items", "orders", "mapp\_items"))  
nn2.results <- compute(nn, temp\_test)  
results <- data.frame(actual = FashionTest$genderprox, prediction = nn2.results$net.result)

Using these predicted values we can now calculate the **confusion matrix** where we can again see if we have improved our accuracy further:

roundedresults1<-sapply(results,round,digits=0)  
roundedresults1df=data.frame(roundedresults1)  
attach(roundedresults1df)  
table(actual,prediction)

## prediction  
## actual 0 1  
## 0 570 1261  
## 1 131 7056

For this model, it shows us that our Neural Network model approach has an accuracy of (1428 + 7011) / (1428 + 7011 + 403 + 176) = 93% and is therefore similar in this instance, to our original Logistic Model approach.Given the simplicity of running and interpreting the logistic model it recommended to use that model as the preferred approach given this data set.

## For Further Exploration

Both models showed a strong ability to infer the gender of our customers based on the proxy variable created. Personalisation then **based on gender**, of both the web platform and in our marketing communication, has the potential to represent a significant uplift in revenue when applied well. Alternate models could be explored to understand if they can improve on accuracy, these could include models such as Bayesian, Random Forest or SVM approaches.

Further, it is highly likely that additional variables would be beneficial in understanding our customer base and/or improving our ability to predict gender. Such variables could include

* Pages / Type of Content engaged with on Website by customer
* Specific Items placed in basket (with high probability of being linked to specific gender)
* Further Demographics (Income, Location, Culture)
* Frequency / Spend per Visit per customer
* Lifetime Value of our customer base
* Cost per Acquistion of Male / Female Customer
* Abandoned Baskets
* Day / Time of Day
* Key Events / Promotions (Women Shoe Sales online, Valentines Day, New Range offer)
* Email Campaign Responsiveness
* Media engagement
* Attitudinal data where available

It would also be important to understand the **benefit delivered by improving the gender inference for TheIconic**, for instance what is the incremental business impact of a x% improvement in the user experience or engagement based on their gender. Using the output and starting with a small A/B test on the web platform or email campaign, can help to review the business impact from personalisation on the customer behaviors (engagement) and ultimately in what they purchase (revenue). Once the benefit is estimated then the model can be applied to more areas over time to achieve a scale impact from the data.