Plots for Classifiers

Plots are rarely used in the context of classification, but they can aid understanding. I'll show three ways of thinking about plots for classification: bar graphs, heatmaps, and plotting the probability predictions from a logit model.

We'll continue working with the random acts of pizza dataset.

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.3
```

```
## -- Attaching packages ----------------------------------------------------------------------- tidyv
```

```
## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.1
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## Warning: package 'ggplot2' was built under R version 3.5.3
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
## Warning: package 'tidyr' was built under R version 3.5.3
```

```
## Warning: package 'purrr' was built under R version 3.5.3
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
```

```
## Warning: package 'stringr' was built under R version 3.5.3
```

```
## Warning: package 'forcats' was built under R version 3.5.3
```

```
## -- Conflicts -------------------------------------------------------------------------- tidyverse_c
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(modelr)
```

```
## Warning: package 'modelr' was built under R version 3.5.3
```

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 3.5.3
```

```
load("za.RData")
```

We always want to start with a cross tab of our dependent variable as a function of other variables. We structure cross tabs in a very particular way for the purposes of research: the independent variable goes on the rows, and the dependent variable goes on the columns. If proportions or percentages are going to be calculated, they should be calculated across rows.

Recalling our previous lesson, let's look at a crosstab of `got_pizza` with the independent variable of `student`

```
tab_student<-with(za,table(student,got_pizza))
```

If we want to make this a little better, we can change the row and column titles

```
colnames(tab_student)<-c("No Pizza","Received a Pizza")
kable(tab_student)
```

	No Pizza	Received a Pizza

	No Pizza	Received a Pizza
No student	3974	1267
Student	302	130

If we want to add proportions to this table, we can it like so:

```
tab_student_prop<-prop.table(tab_student,margin=1)
kable(tab_student_prop)
```

	No Pizza	Received a Pizza
No student	0.7582522	0.2417478
Student	0.6990741	0.3009259

Sometimes (okay, all the times) audiences prefer percentages. Easy enough to do:

```
kable(round(tab_student_prop*100,1))
```

	No Pizza	Received a Pizza
No student	75.8	24.2
Student	69.9	30.1

If you want to include a third variable in a cross tab, that requires splitting the dataset. For instance, if we want to know the proportion of posts that include "student" AND "grateful" that received pizza, we would do this:

```
tab_student_grateful<-with(filter(za,
                           as.character(grateful)=="Grateful in post"),
                           table(student,got_pizza))
```

Outcome by "Student" AND "Grateful"

```
prop.table(tab_student_grateful,margin=1)%>%kable()
```

	0	1
No student	0.7159763	0.2840237
Student	0.5263158	0.4736842

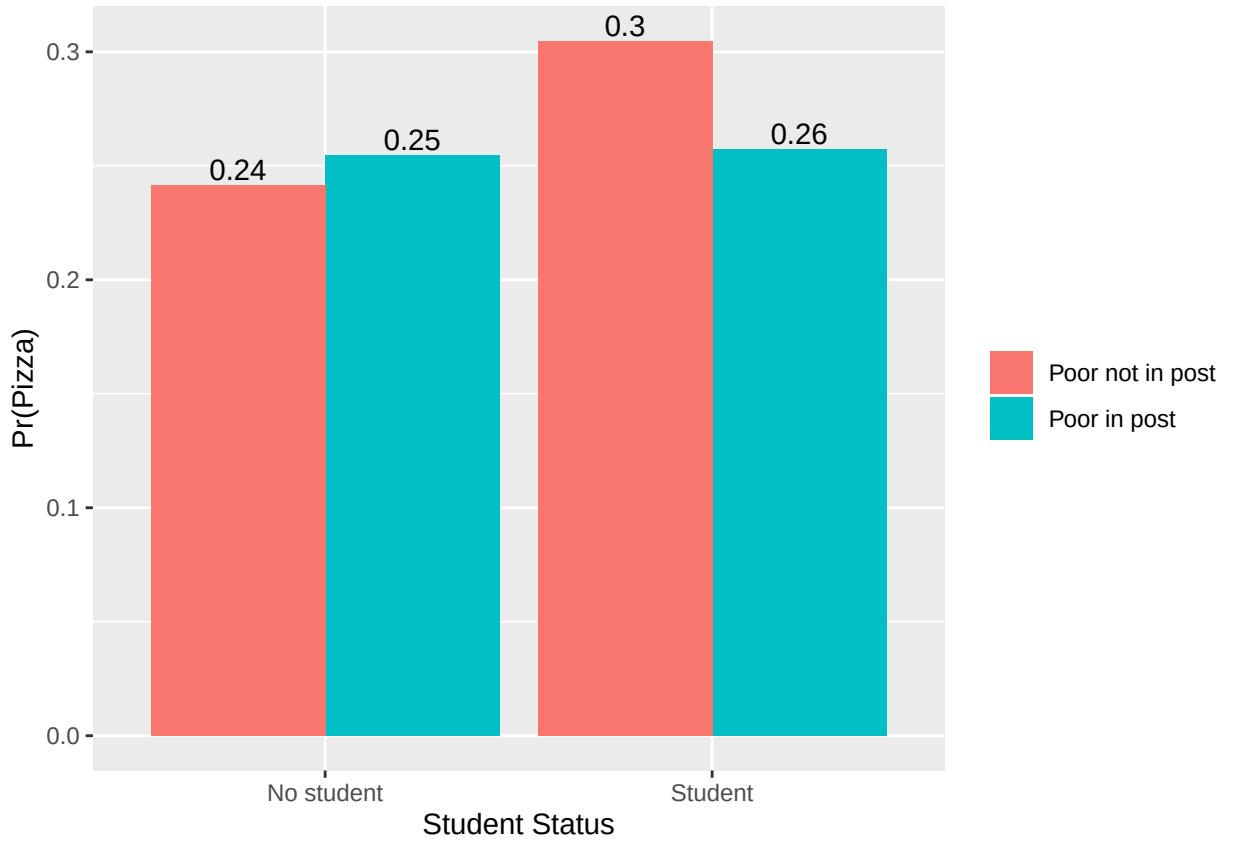
Bar Graphs from Cross Tabs

To format the data for barcharts, we make use of the concept of conditional means. Let's use two variables to calculate the probability of receiving a pizza: poor and student.

```
za_sum<-za%>%
  group_by(poor,student)%>%
  summarize(prob_pizza=mean(got_pizza,na.rm=TRUE))
```

Then we can plot this using our familiar ggplot commands:

```
gg1<-ggplot(za_sum,aes(y=prob_pizza,x=student,fill=poor))
gg1<-gg1+geom_bar(stat="identity",position="dodge")
gg1<-gg1+xlab("Student Status")+ylab("Pr(Pizza)")
gg1<-gg1+theme(legend.title=element_blank())

gg1<-gg1+geom_text(aes(label=round(prob_pizza,2)),
                   position=position_dodge(width=.9),
                   vjust=-.25)
gg1
```



Heat Maps

To generate a heat map, we'll first divide up the independent variables into quintiles:

```
za<-za%>%mutate(score_quintile=ntile(score,5),
                karma_quintile=ntile(karma,5))
```

Then we'll create a summary dataset that shows the probabilitie of the outcome across all of the combined categories of the two independent variables.

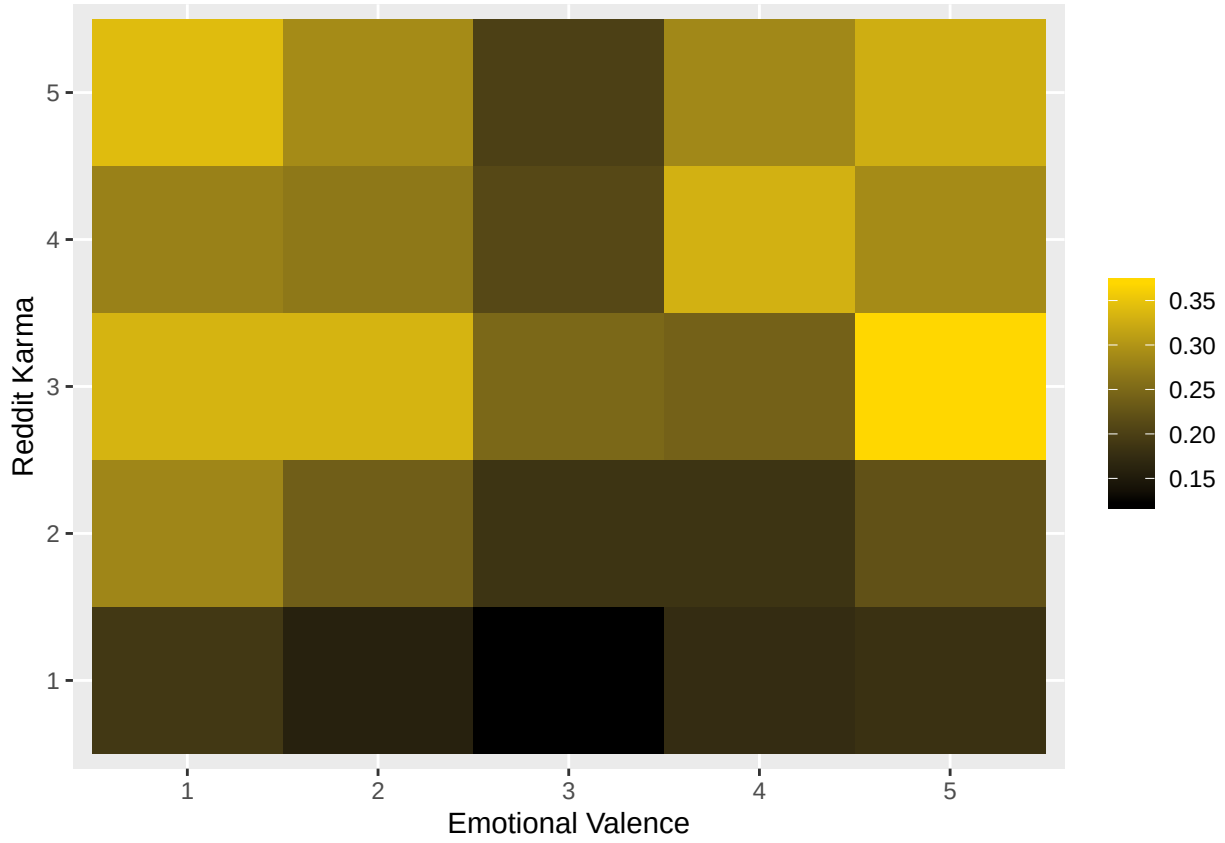
```
za_sum<-za%>%group_by(score_quintile,karma_quintile)%>%
  summarize(prob_pizza=mean(got_pizza,na.rm=TRUE))%>%
  arrange(-prob_pizza)
```

Missing data isn't important, so we'll drop it.

```
za_sum<-za_sum%>%filter(!(is.na(score_quintile)),!(is.na(karma_quintile)))
```

Now we're ready to plot!

```
gg<-ggplot(za_sum,
           aes(x=as.factor(score_quintile),
               y=as.factor(karma_quintile),fill=prob_pizza))
gg<-gg+geom_tile()
gg<-gg+scale_fill_gradient(low="black",high="gold")
gg<-gg+xlab("Emotional Valence")+ylab("Reddit Karma")
gg<-gg+theme(legend.title=element_blank())
gg
```



Plotting by probabilities from models

It can be difficult to plot the results of a logistic regresion. We're going to use the same solution that we used for linear regression, where we create simulations from a hpothetical dataset.

First we rerun our logisitc regression.

```
logit_mod<-glm(got_pizza~
               karma+
               total_posts+
               raop_posts+
               student+
               grateful,
```

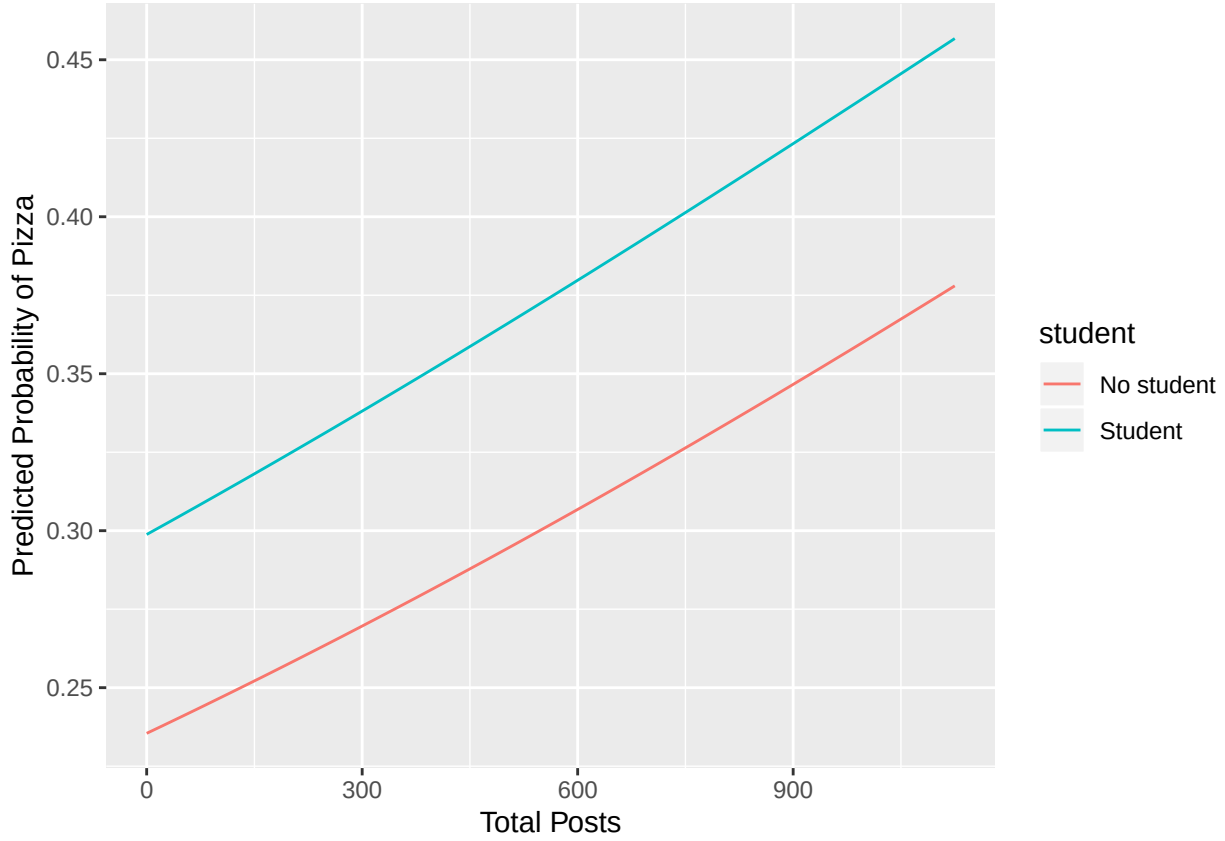
```
             data=za,
            na.action=na.exclude,
            family=binomial(link="logit"),
               y=TRUE)

#logit_mod<-glm(y~x1+x2,
#               family=binomial(link="logit"))
```

Then we create some hypothetical data.

```
hypo_data<-data_grid(za,
                     total_posts=seq_range(total_posts,n=100),
                     karma=mean(karma,na.rm=TRUE),
                     raop_posts=mean(raop_posts,na.rm=TRUE),
                     grateful=levels(grateful)[1],
                     student=levels(student))%>%
  mutate(pred=predict(logit_mod,newdata=.,type="response"))
```

And now we're ready to plot.

```
gg<-ggplot(hypo_data,
           aes(x=total_posts,y=pred,color=student))
gg<-gg+geom_line()
gg<-gg+xlab("Total Posts")+ylab("Predicted Probability of Pizza")
gg
```



How to do the same with Random Acts of Pizza posts (raop)

```
hypo_data<-data_grid(za,
                     total_posts=mean(total_posts,na.rm=TRUE),
                     karma=mean(karma,na.rm=TRUE),
                     raop_posts=seq_range(raop_posts,n=100),
                     grateful=levels(grateful)[2],
                     student=levels(student))%>%
  mutate(pred=predict(logit_mod,newdata=.,type="response"))
```

```
gg<-ggplot(hypo_data,
           aes(x=raop_posts,y=pred,color=student))
gg<-gg+geom_line()
gg<-gg+xlab("RAOP Posts")+ylab("Predicted Probability of Pizza")
gg
```