STAT 598 Final Project

Linna Henry

April 28, 2016

Etsy.com is a website where artists and designers can list their handmade items for sale. Each item listed on Etsy must be categorized in some way. For example, items can be categorized as jewelry, clothing, kitchenware, etc. Each item usually has a description written by the artist as well, although it is not necessary for the creator to include a description. The goal of my project will be to correctly classify an item's category on etsy.com based on the words in the description.

I will use kNN to classify each item. During my analysis, I will search for and determine the best k to use. The target class label will be the listed category on Etsy. For the distance function, I will consider number of similar words in the description, price, and possibly number of words in the description. I plan on collecting my data via web scraping in R and I plan to have a data set of 600 observations that I will then split 75/25 into a training and test set. I will only use listings that have item descriptions. For this project, I will only use three categories: paintings, jewelry, and clothing patches.

Web Scraping Code for the painting category

```
library(xml2)
library(rvest)
library(ggplot2)

etsystring <- "https://www.etsy.com/"

epaint <- "https://www.etsy.com/c/art-and-collectibles/painting?page="
reffront <- ".position-relative:nth-child("
refend <- ") .card-title"

#200 painting descriptions
#webscraping
pdescrip <- rep(NA, 1)
pprice <- rep(NA, 1)
count <- 1
for(i in 1:18){
    print(i)
    epntcurr <- paste(epaint, i, sep = "")
    epaint.html <- read_html(epntcurr)
    epaint.sess <- html_session(epntcurr)
    for(j in 1:12){
        ref <- paste(reffront, j, refend, sep = "")
        pcurr <- epaint.html %>%
            html_node(ref) %>%
            html_text()
        pntlink <- try(epaint.sess %>% follow_link(pcurr))
        if ('try-error' %in% class(pntlink)) next #some links throw errors - I don't know why
        else{
            pntlink <- epaint.sess %>% follow_link(pcurr)
            pdes <- pntlink %>%
                html_node("#description-text") %>%
```

```
                html_text()
            ppce <- pntlink %>%
                html_node(".currency-value") %>%
                html_text()
            pdescrip[count] <- pdes
            pprice[count] <- ppce
            count <- count + 1
        }
    }
}

paints <- list(pprice, pdescrip)
p2 <- cbind(pprice, pdescrip)

write.csv(p2, "stat598paintings.csv")
```

Plot of the data

```
splot <- list(nwords=nwords, prce=prce, clab=clab)
splot <- as.data.frame(splot)
plt <- ggplot(splot) + geom_point(aes(x=prce, y=nwords, colour = factor(clab)))
plt + scale_x_continuous(limits=c(0,500)) + scale_y_continuous(limits=c(0,500))
```

This is a plot of number of words in the description vs price, with some outliers removed, for the entire data set. This plot shows that number of words is not a very good classifier for category. As number of words

increases, there is no pattern change in the etsy category. This makes sense because you wouldn't expect paintings, jewelry, or patches to differ for the number of words in the description. One interesting thing about this plot, however, is that there does appear to be a category distinction for price. Patches all have a low price, jewelry has a middle range price, and paintings have a high price. This means that price is a good classifier for category.

I cannot make a plot to represent number of similar words in the description, because that varies depending on what description I am comparing it to.

Split into training/test with a 75/25 split

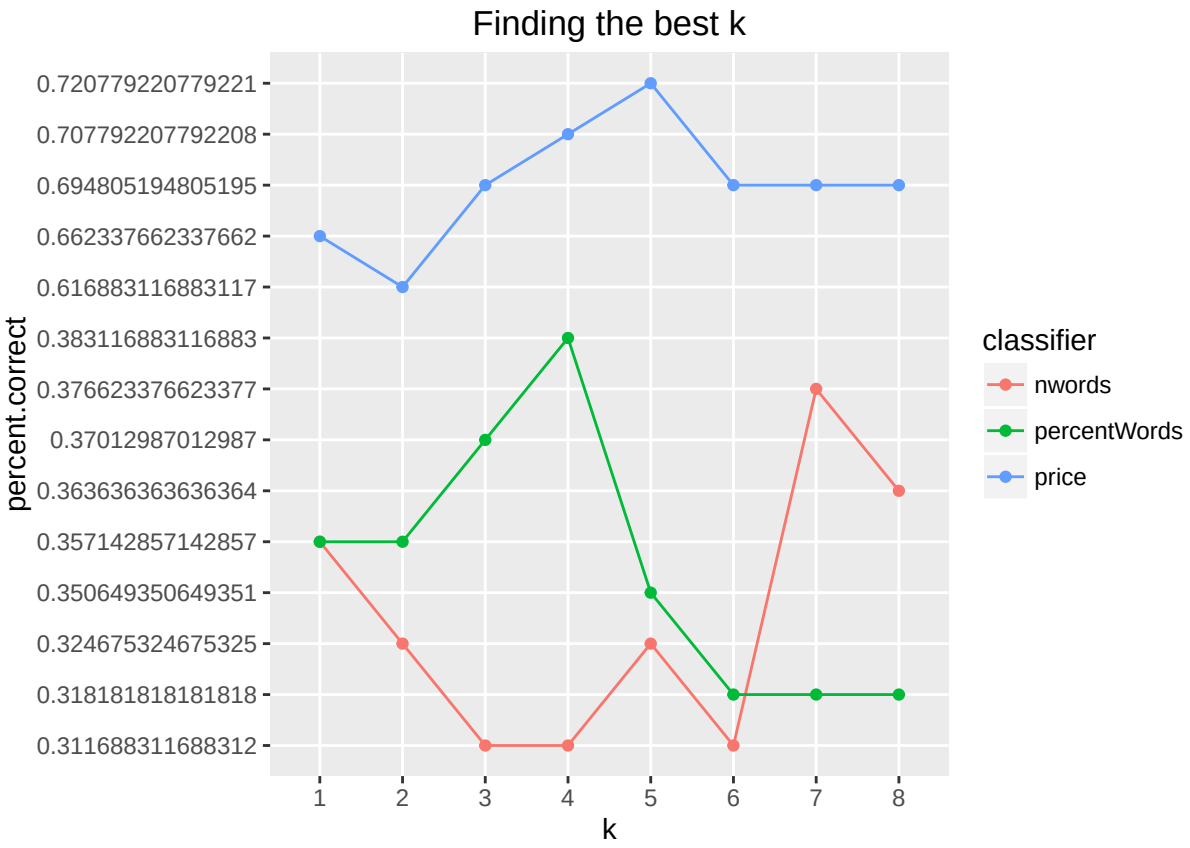
```
set.seed(100)
tlist <- sample(0:615, 462)
train <- dataset[tlist, ]
test <- dataset[-tlist,]
mtrain <- list()
mtest <- list()
trcount <- 1
tecount <- 1
tindex <- 1:615
teind <- tindex[-tlist]
for(i in 1:615){
  if(i %in% tlist){
    mtrain[[trcount]] <- m[i] #contains words in the description
    trcount <- trcount + 1
  }
  else{
    mtest[[tecount]] <- m[i]
    tecount <- tecount + 1
  }
}
```

Calculate the number of similar words in the description

```
distmat <- matrix(0, length(mtest), length(mtrain)) #number of similar words
ndistmat <- matrix(0, length(mtest), length(mtrain))
pmat <- matrix(0, length(mtest), length(mtrain))
for(i in 1:length(mtest)){
    for(j in 1:length(mtrain)){
        distmat[i,j] <- sum(mtest[[i]][[1]] %in% mtrain[[j]][[1]])
        pmat[i,j] <- distmat[i,j]/length(mtest[[i]][[1]])
        ndistmat[i,j] <- length(mtest[[i]][[1]]) - distmat[i,j]
    }
}
```

Run the kNN algorithm to find the best K

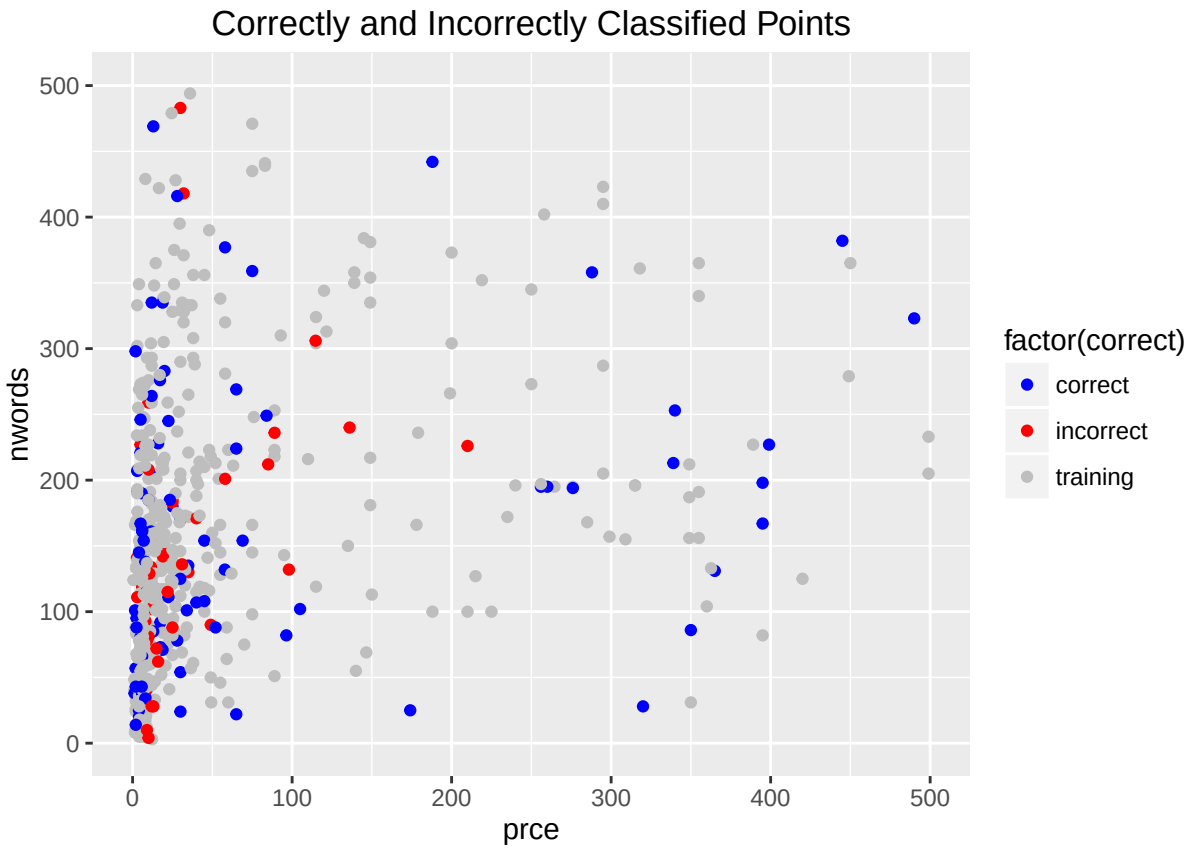
```
cpclass <- rep(NA, 5)
cdclass <- rep(NA, 5)
cdpclass <- rep(NA, 5)
for(k in 1:8){
  pclassification <- rep(NA, length(teind))
  dclassification <- rep(NA, length(teind))
  dpclassification <- rep(NA, length(teind))
  for(i in 1:length(teind)){
    #by price
    cpce <- abs(test$price[i] - train$price)
    psort <- sort(cpce, index.return = T)
    pclassif <- train$class[psort$ix[1:k]]
    pcl <- summary(as.factor(pclassif)) #find majority
    pclassification[i] <- names(pcl[pcl == max(pcl)][1])

    #by number of similar words
    dsort <- sort(distmat[i,], index.return = T, decreasing = T)
    dclassif <- train$class[dsort$ix[1:k]]
    dcl <- summary(as.factor(dclassif))
    dclassification[i] <- names(dcl[dcl==max(dcl)][1])

    #by percent of similar words
    dpsort <- sort(pmat[i,], index.return = T)
    dpclassif <- train$class[dpsort$ix[1:k]]
    dpcl <- summary(as.factor(dpclassif))
    dpclassification[i] <- names(dpcl[dpcl==max(dpcl)][1])
  }

  cpclass[k] <- sum(pclassification == test$class)/nrow(distmat)
  cdclass[k] <- sum(dclassification == test$class)/nrow(distmat)
  cdpclass[k] <- sum(dpclassification == test$class)/nrow(distmat)
}

total <- data.frame(cbind(rep(1:8,3), c(cpclass, cdclass, cdpclass), rep(c("price", "nwords", "percentW
names(total) <- c("k", "percent.correct", "classifier")
ggplot(data=total, aes(x=k, y=percent.correct, colour=classifier)) + geom_line(aes(group = classifier))
```



This plot shows that price is the best classifier, because on average price has the highest percent correctly classified. The highest percent correct is 72% when k=5. The next best classifier is the percent of similar words in the description. The highest percent correct for this classifier is 38% when k=4. The worst classifier is the number of similar words in the description. The highest percent correct for this classifier is 37.6% when k=7. For the number of similar words, the percent correct lingers around 33%. 33% is the percent correct I would expect to see due to random chance because I have 3 categories. Therefore, the number of similar words in the description as a classifier performs at the absolute worst. Based on this plot, I am going to use only price as my classifier and I am going to use k=5.

Results

```
correct <- rep("incorrect", nrow(dataset))
correct[tlist] <- "training"
corr <- rep("incorrect", length(teind))
corr[pclassification == test$class] <- "correct"
correct[teind] <- corr
dataset$correct <- correct
ggplot(dataset) + geom_point(aes(x=prce, y=nwords, colour = factor(correct))) + scale_x_continuous(limi
```



Previously, I showed that when k=5 and using price as the classifier, I observe 72% of points correctly classified. This plot once again shows number of words in the description vs. price. The gray points in this plot are points from the training set. The red points were points that were incorrectly classified and the blue points represent observations that were correctly classified in the test set. Most of the incorrectly classified points fall in the low price range. In the low price range, there was a lot of overlap between patches and jewelry. These incorrectly classified points are likely due to that overlap. Points in the high price range are almost all classified correctly.

Further Investigation

Since I am using k=5, there is a chance of a tie happening when I search for the majority vote. For example, I could observe 2 patches, 2 jewelry, and 1 painting for the 5 nearest neighbors. This actually happens quite often. When this happens, the test observation is assigned whichever appears first of the categories that receive 2 "votes". Now I am going to write a kNN algorithm that uses the percent of similar words when there is a tie for the price classifier.

```
pclagain <- rep(NA, length(teind))
k<- 5
find <- c(2,2)
for(i in 1:length(teind)){
    cpce <- abs(test$price[i] - train$price)
    psort <- sort(cpce, index.return = T)
    pclassif <- train$class[psort$ix[1:k]] #k = 1
    pcl <- summary(as.factor(pclassif)) #find majority
    if(sum(pcl %in% find) > 0){ #theres a tie
      dpsort <- sort(pmat[i,], index.return = T)
```

```
      dpclassif <- train$class[dpsort$ix[1:k]]
      dpcl <- summary(as.factor(dpclassif))
      pclagain[i] <- names(dpcl[dpcl==max(dpcl)][1])
    }
    else{
      pclagain[i] <- names(pcl[pcl == max(pcl)][1])
    }
}
sum(pclagain == test$class)/nrow(test)
```

```
## [1] 0.6493506
```

Here I observe 64.9% observations in the test set correctly classified. This is a decrease from the 72% I observe when I only use price as the classifier. This shows that even if there is a tie in the nearest neighbors, it is better to just pick one "winner" at random than to try to use another classifier that performs worse.

What I learned

I expected that number of similar words would be a good classifier. I thought that words to describe paintings would be a lot different from words that describe jewelry and patches. I learned that the number of similar words performs so bad that it is not any better than classifying points randomly. I did not expect price to be such a good classifier, but it ended up being the best classifier. I think if I had picked different categories to compare, price would not have done so well. I expected that number of total words in the description would be a bad classifier, and I ended up not even using this classifier because the first plot showed there is no distinction between categories for the number of total words.

What I couldn't do and why

When I was classifying points, I found that most points were correctly classified by price, however some points that were incorrectly classified by price were correctly classified by number of similar words. I was considering weighting the classifiers (i.e. price classification would have 70% weight and number of similar words would have a 30% weight) but I wasn't sure how I would determine the weights.

If I had more time to work on this project, I would have liked to try Naive Bayes Classification to classify the categories.

Also, there are some words that I would expect to see in almost every description like "the", "a", and "is". If I had a list of all common words, I would have liked to remove these words from every description. That way when I am comparing descriptions, I am comparing more unique words. I think this would have helped to correctly classify more points.