Machine Learning Contest #1

February 11, 2018

Can we tell what is in a painting by the colors of the pixels?

This contest is to analyze the happy paintings by Bob Ross. This was the subject of the 538 post, "A Statistical Analysis of the Work of Bob Ross".

All the painting images were taken from the sales site, read the images into R, and resized them all to be 20 by 20 pixels. Each painting has been classified into one of 8 classes based on the title of the painting. This is the data that you will work with.

Here are three of the original paintings in the collection, labelled as "scene", "dusk", "flowers":

Description of variables

The training data file consists of 178 rows and 1201 columns. 178 rows represent for 178 paintings. The first column is the class of image. The other columns are the predictors. There are $20 \times 20 = 400$ pixels for each image, and every pixel can be quantified by an additive RGB model (https://en.wikipedia.org/wiki/RGB_color_model) with three color measurements: r for red, g for green, and b for blue. Hence, the total number of predictors is $400 \times 3 = 1200$. The predictor name contains three pieces of information for the pixel: rgb, row, and column. For example, "r0102" means red for the pixel in row 1 and column 2, "g1005" means green for the pixel in row 10 and column 5, and "b3120" means blue for the pixel in row 31 and column 20.

```
train = read.csv("train.csv")
dim(train)
```

```
## [1]  178 1201
```

```
train[1:4,c(1:4,1199:1201)]
```

```
##     class      r0101     g0101     b0101     r2020     g2020     b2020
## 1    oval 0.61568627 0.5568627 0.4823529 0.7709804 0.7082353 0.6572549
## 2 flowers 0.92156863 0.9882353 1.0000000 0.1477451 0.1908824 0.1634314
## 3    cold 0.08627451 0.1490196 0.1450980 0.1519608 0.1794118 0.2186275
## 4   scene 0.32549020 0.6745098 0.8078431 0.2349020 0.2247059 0.1690196
```

```
table(train$class)
```

```
##
##        cold        dusk     flowers impressions        oval       scene
##          23          30          22          17           5          28
##       trees       water
##          18          35
```

In the test data file there are 63 rows and 1200 columns.

```
test = read.csv("test.csv")
dim(test)
```

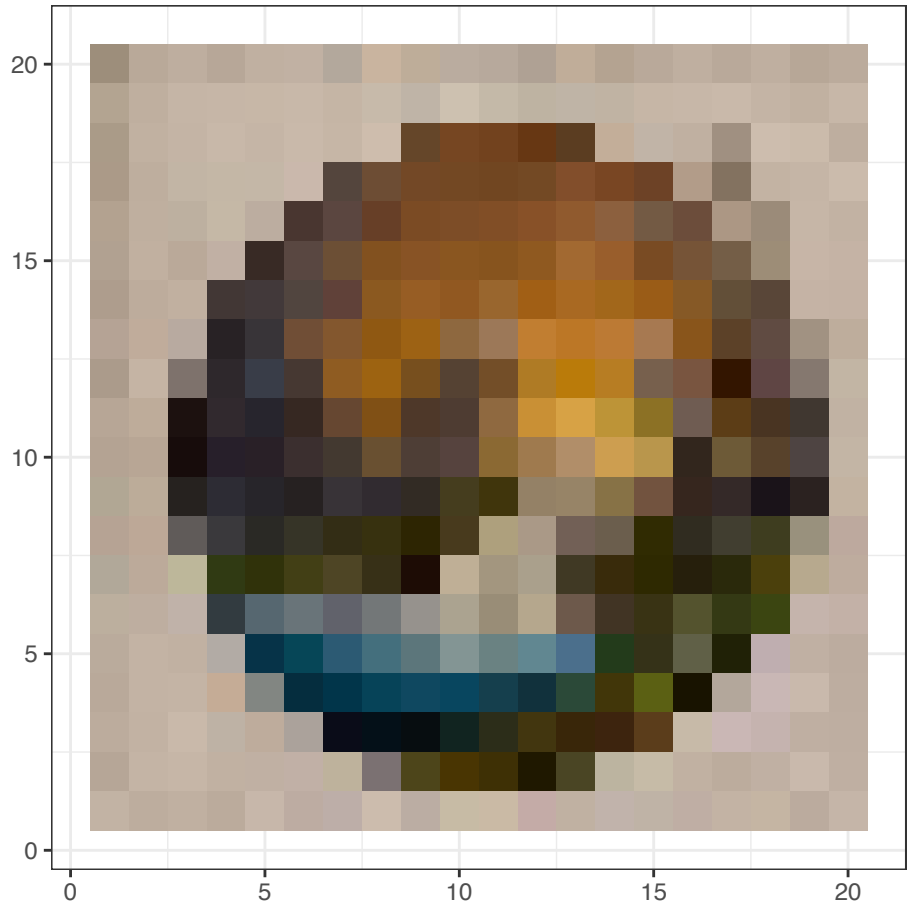
```
## [1]   63 1200
```

Can we draw the images and predict by eyes?

We can draw the images using R. The code below draw the first image in the training data. However, for this contest it is not eligible to predict the images by human being. You must submit your code after the contest for attestation.

```
image1 = train[1,-1,drop=FALSE]
color = rgb(image1[,(0:399)*3+1], image1[,(0:399)*3+2],
            image1[,(0:399)*3+3])
imgdat = data.frame(x=rep(1:20,20),y=rep(20:1,each=20),
                    col=color,stringsAsFactors=FALSE)

library(ggplot2)
qplot(x, y, data=imgdat, fill=col, geom="tile") +
  scale_fill_identity(labels=imgdat$col) + theme_bw() +
  theme(legend.position="none") + xlab("") + ylab("")
```



Task

1. Create the most accurate classifier that you can for the data, as measured by the test data.
2. Write a 3-5 page slides summarizing your approach to
   (a) formulating the model (design) matrix,
   (b) building the classifier, and
   (c) your findings from the data.

Format of submission

Your submission file should be in the csv format with two columns: id and class. Id must match the order of the test data. Example:

```
id,class
1,cold
2,dusk
3,water
...
63,trees
```

Comments

- Do some exploratory descriptive analysis on the data, that could illustrate how the classes of images differ, or are similar. Make strategies for the classes that are easily misclassified.
- It might be good to create new variables to help you build a better classifier; for example, the average of pixels in corners.
- You will need to submit at least one entry to the kaggle site (https://www.kaggle.com/c/unodatamining-2018-1) for testing the classifier.

Deadlines:

- February 28 (11:59 pm): Final prediction submission.
- March 5 (in class): Presentation. 6 minutes per two teams.
- March 7 (11:59 pm): Slides and code submission.

Grading:

- Total points: 20 (+1)
  - Accuracy of classifier: 6
    - Score = 6 * accuracy rate
  - Progress made from multiple submissions: 6
    - Number of good submissions (decreasing error rate): 4
    - Amount of decreasing of the error rate: 2
  - Presentation: 6 (+1)
    - Model matrix: 2
    - Model selection: 2
    - Model assessment: 2
    - Team battle: (+1)
  - Met the deadline: 2

Teams

- Undergrad 1: Phillip Baumberger,Bryan Vukorepa
- Undergrad 2: Kevin Emmel, Yao Todo
- Undergrad 3: Christopher Johnson, Brendan Schutte
- Undergrad 4: Jeremiah Casanova, Siddhi Munde
- Undergrad 5: Theophilus Amankwah, Kent Rainey Biler
- Undergrad 6: Russell Buffum, Jerrid Kimball
- Grad 1: Adithi Deborah Chakravarthy, Nicole Netsov
- Grad 2: Jiaqi Huang, Siddesh Southekal
- Grad 3: Amir Ebrahimifakhar, Tim Mastny
- Grad 4: Wei Chen, Catherine Rivier
- Grad 5: Jacques Anthony, Yifeng Hu
- Grad 6: Nirmal K C, Moctar Sadou Nouhou
- Grad 7: Sohan Gyawali, David Vincent
- Grad 8: Nagavardhini Avuthu, Carolina Goncalves
- Grad 9: Nathan Hotovy, Amanda Vander Wal
- Grad 10: Lale Madahali, Kieren Smith
- Grad 11: David Pease, Xinyu Zuo

Presentation

To save time for the presentations, two teams will be giving the presentation together in a battle mode. All the audience (including students who audit the class) will vote for the better presenter and one of the two teams who gets higher votes will receive an extra point. Teams will be grouped based on their final submission results – the battling teams will have very similar error rates. Teams in battle should make their slides together and present alternately in case they run out of time.