Kiana Mills

904754661

HW 5

Neural Net Classification

**Representation of Ship Features (f vector)**

1. Categorical features

For the categorical features (‘name’ and ‘surface color’ properties), I first decided to create some features based on patterns of the names.

I used the following general patterns:

for name in train\_df['name']:

# Check if name has digits [Antarean]

if has\_number(name):

numeric\_name.append(1)

else:

numeric\_name.append(0)

# Check if name starts with A or E [Antarean]

if re.match(r'[AE].+', name):

firstAE.append(1)

else:

firstAE.append(0)

# Check if name starts with two consonants [Klingon]

if re.match(r'^[^aeiouAEIOU]{2}.+', name):

first\_consonants.append(1)

else:

first\_consonants.append(0)

# Check if the name as double 'l' [Romulan]

if re.match(r'.+[l]{2}.+', name):

double\_l.append(1)

else:

double\_l.append(0)

# Check if last letter is a ‘k’ [Klingon]

if re.match(r'.+[k]$', name):

last\_k.append(1)

else:

last\_k.append(0)

Then I appended the above arrays to my data frame object as their own features.

For the surface colors, I first added features depending on if the color name had ‘Dark’ or ‘Light Color’ to more heavily weight this feature. Then, I chose to **One-Hot Encode** the color names, so that I could increase the number of features while also keeping the encoding binary. The result is that each color generated a new feature, and the corresponding ship in this color would have a ‘1’ for this feature and a ‘0’ for every other color feature.

By One Hot Encoding, I also avoided the problem of assuming that different category values assumed higher values than other, such as in the case of Label Encoding.

1. Numeric features

For the numeric features, I chose to simply normalize the values using the simple normalization equation:

X(n) = (X - Xmin)/(Xmax - Xmin)

After encoding all categorical and numeric features, I then centered and scaled my feature vectors:

for key,value in train\_df.iterrows():

f = value.to\_numpy()

f = f.reshape((dim,1))

# Center and Scale

f = f - f.mean()

f = f / np.linalg.norm(f)

train\_X.append(f)

**Representation of Target (g vector)**

1. My initial thought was to simply use Label Encoder to encode the target values, so that the g vector would resemble:

G\_set = [2 2 2 2 2 3 3 3 3 3 0 0 0 0 0 1 1 1 1 1]

where 2 = Klingon, 3 = Romulan, 0 = Antarean, 1 = Federation ( Numbers assigned alphabetically)

However, after realizing I couldn’t construct an Associative matrix from this dimension, I decided to instead use a binary encoding so that my g vectors would resemble this:

If Klingon: g1 = [1 0 0 0]

If Romulan: g2 = [0 1 0 0]

If Antarean: g3 = [0 0 1 0]

If Federation: g4 = [0 0 0 1]

My resulting f vector set had dimensions (20x21) and g vector set (20x4)

Constructing A = gf’ resulted in dimensions (4 x 21) because each f vector was (21x1) and each g was (4x1).

Filling in Missing Data:

1. For missing (completely corrupted) surface colors, I simply filled in the data with NaN because I already created a column in my initial training set called ‘surface\_nan’ under One Hot Encoding. For vague colors ie ‘Dark Color’ or ‘Light Color’ I simply added the corresponding ‘Dark’ or ‘Light’ features and then dropped the color column.
2. For completely corrupted names, I added zeros for all my ‘first A or E’, ‘first two consonants’, etc vectors. For partially corrupted data, I again ran my conditionals for determining if it had a digit, last ‘k’, first two consonants, etc.
3. For missing numeric values, I looked at other fully filled values in that row and used the average of that group to fill in the corrupted data. The averages were computed like this:

AVG\_WARP\_F = (6.7 + 6.8 + 6.5 + 6.4 + 6.5)/5

AVG\_WARP\_A = (6.7 + 6.8 + 6.5 + 6.8 + 6.7)/5

AVG\_WARP\_R = (7.3 + 7.4 + 7.3 + 7.2 + 7.4)/5

AVG\_WARP\_K = (6.9 + 7.0 + 7.3 + 7.1 + 7.1)/5

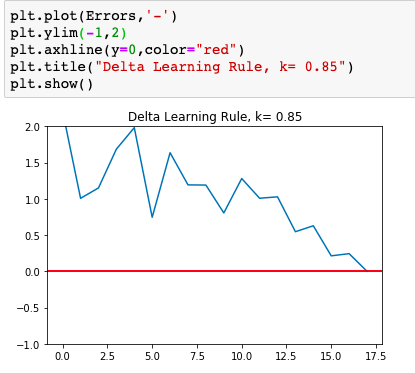
… and the same was done for transponder and axis ratio.

After filling in missing numeric values, I again normalized each value as before.

**Learning Process**

For the delta learning rule, I used an initial k value of 0.85 and slowly decreased it. I set an error threshold of 0.0001 and reached this threshold in 17 trials for my training data.

The following graph shows the absolute value of the error lengths slowly decreasing over the course of 17 trials .



My training set predictions were all correct and predictions on corrupted data were nearly all correct:

**TRAINING SET PREDICTIONS**

Klingon detected: HOSTILE Stay Alert

Klingon detected: HOSTILE Stay Alert

Klingon detected: HOSTILE Stay Alert

Klingon detected: HOSTILE Stay Alert

Klingon detected: HOSTILE Stay Alert

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Antarean detected: PEACEFUL No action required, but be careful

Antarean detected: PEACEFUL No action required, but be careful

Antarean detected: PEACEFUL No action required, but be careful

Antarean detected: PEACEFUL No action required, but be careful

Antarean detected: PEACEFUL No action required, but be careful

Federation detected: PEACEFUL No action required

Federation detected: PEACEFUL No action required

Federation detected: PEACEFUL No action required

Federation detected: PEACEFUL No action required

**CORRUPTED SET PREDICTIONS:**

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Federation detected: PEACEFUL No action required

Federation detected: PEACEFUL No action required

Federation detected: PEACEFUL No action required

Klingon detected: HOSTILE Stay Alert

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Klingon detected: HOSTILE Stay Alert

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Klingon detected: HOSTILE Stay Alert

Antarean detected: PEACEFUL No action required, but be careful

Klingon detected: HOSTILE Stay Alert

Klingon detected: HOSTILE Stay Alert

Antarean detected: PEACEFUL No action required, but be careful

Antarean detected: PEACEFUL No action required, but be careful

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Romulan detected: POTENTIALLY HOSTILE Stay Alert

Antarean detected: PEACEFUL No action required, but be careful

Antarean detected: PEACEFUL No action required, but be careful

Federation detected: PEACEFUL No action required

Federation detected: PEACEFUL No action required