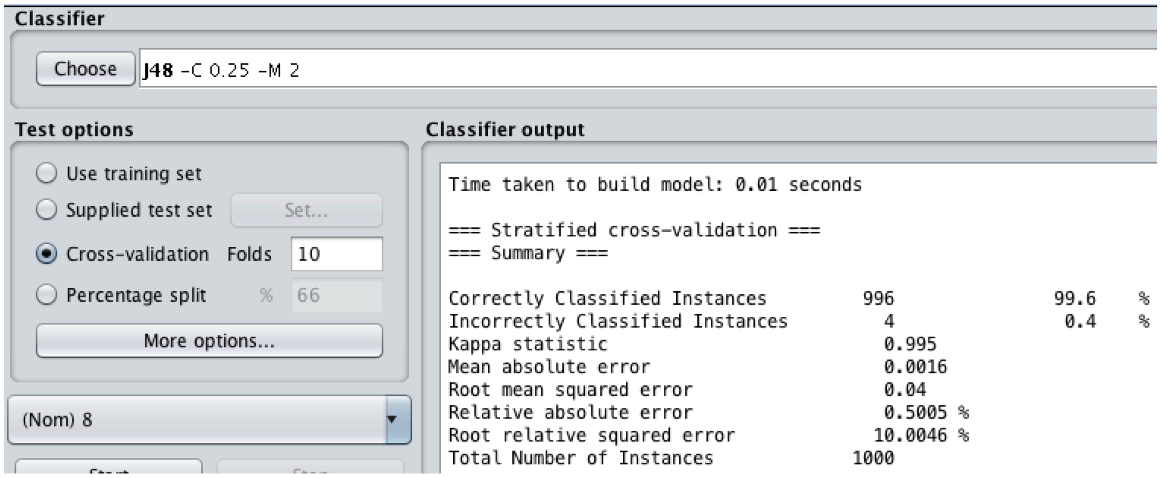
EECS 349

Problem Set 4

William Wei

Part 1

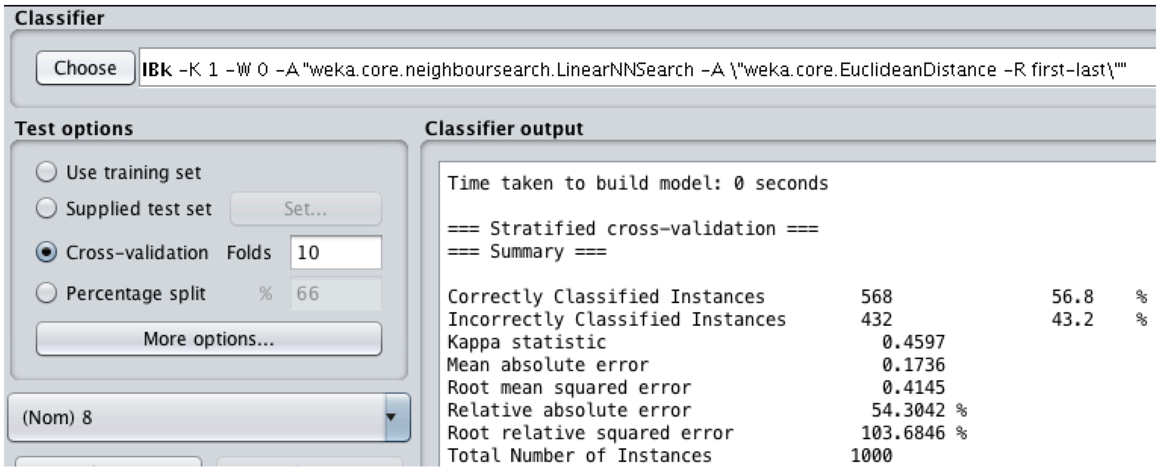
1



J48 -C 0.25 -M 2

[✓ Cross-validation] Folds 10

```
Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         996               99.6    %
Incorrectly Classified Instances         4                0.4    %
Kappa statistic                          0.995
Mean absolute error                      0.0016
Root mean squared error                  0.04
Relative absolute error                  0.5005 %
Root relative squared error             10.0046 %
Total Number of Instances             1000
```



IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

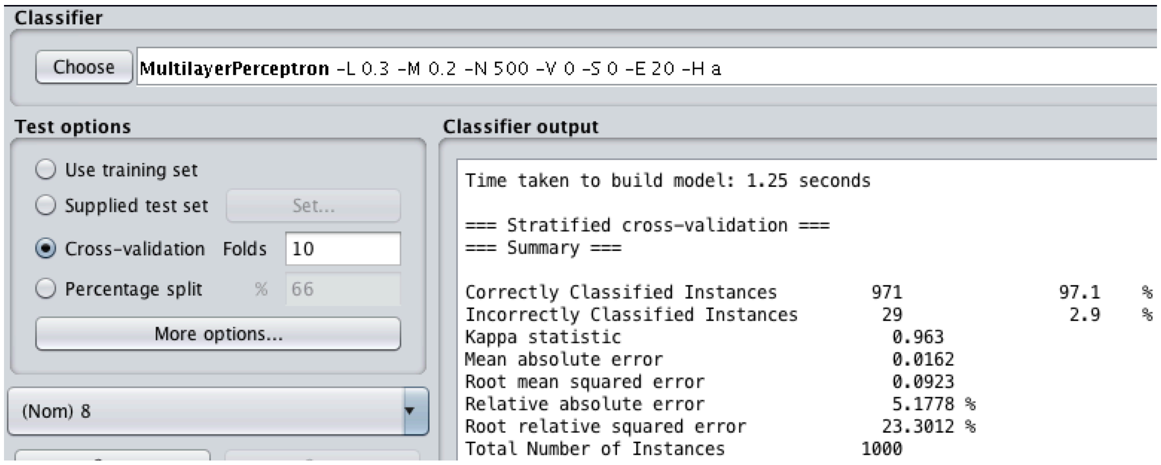
[✓ Cross-validation] Folds 10

```
Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         568               56.8    %
Incorrectly Classified Instances       432               43.2    %
Kappa statistic                          0.4597
Mean absolute error                      0.1736
Root mean squared error                  0.4145
Relative absolute error                 54.3042 %
Root relative squared error            103.6846 %
Total Number of Instances             1000
```

Absolute difference is 99.6%-56.8%=42.8%

I create a dataset, which has 8 attributes, and only the 3rd attribute influence the classification. other numeric attributes has nothing to do with the classification. Since KNN will calculate the distance through all the attributes, no matter whether this attribute matters classification or not, therefore, KNN will have a low accuracy for this data set.

2

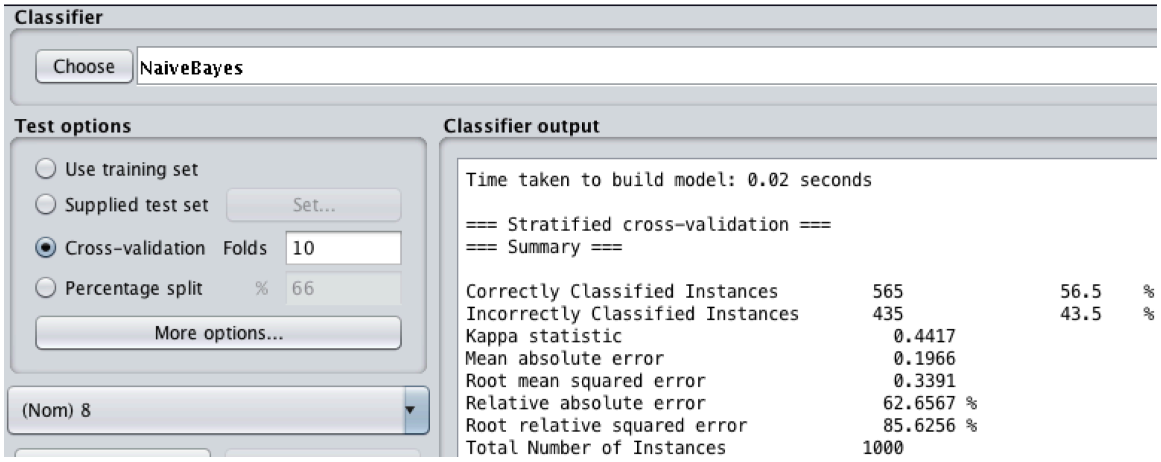


MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a

Cross-validation Folds 10

% 66

```
Time taken to build model: 1.25 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         971               97.1    %
Incorrectly Classified Instances        29                2.9    %
Kappa statistic                          0.963
Mean absolute error                      0.0162
Root mean squared error                  0.0923
Relative absolute error                  5.1778 %
Root relative squared error             23.3012 %
Total Number of Instances             1000
```



NaiveBayes

Cross-validation Folds 10

% 66

```
Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         565               56.5    %
Incorrectly Classified Instances       435               43.5    %
Kappa statistic                          0.4417
Mean absolute error                      0.1966
Root mean squared error                  0.3391
Relative absolute error                 62.6567 %
Root relative squared error             85.6256 %
Total Number of Instances             1000
```

Absolute difference is 97.1%-56.5%=40.6%

The naive bases pretend that all attributes are independent. So I made the attributes not independent. So the accuracy of the naive bayes will be very low.

3

If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as "Zero Frequency". To solve this, we can use the smoothing technique, adding 1 to every count so it's never zero. To balance this, we add the number of possible words to the divisor, so the division will never be greater than 1.

4

(a) The joint distribution over these 4 variables needs 9*3*3*3-1=242 independent parameters.

(b) This conditional distribution will contain (3-1)*9*3*3=162 independent parameters

(c) By using the Naive Bayes assumption, it will have 38 independent parameters

Because P(ExamScore | HoursOfSleep, Studied, LikesMaterial) =P(HoursOfSleep, Studied, LikesMaterial | ExamScore)*P(ExamScore)/P(HoursOfSleep, Studied, LikesMaterial)

(According the the Naive Bayes Rule)

And assuming HoursOfSleep, Studied, LikesMaterial conditionally independent of ExamScore

We can further write P(HoursOfSleep, Studied, LikesMaterial | ExamScore) as P(HoursOfSleep|ExamScore)*P(Studied|ExamScore)*P(LikesMaterial|ExamScore)

(According to the rule of conditional probability)

This part needs 3*8+3*2+3*2=36 independent parameters

P(ExamScore) needs 2 independent variables

So using the naives we only need to have 36+2=38 independent parameters

Part 2

5

See the code:

```
import numpy as np

def cosine_similarity(v1,v2):
        return (np.dot(v1,v2.T))/(np.linalg.norm(v1)*np.linalg.norm(v2))
```

6

The similarity is:

Pixel:
Cosine_Similarity(mj1, mj2) = 0.37086195
Cosine_Similarity(mj2, cat) = 0.61974399
Cosine_Similarity(mj1, cat) = 0.47308852

Vgg
Cosine_Similarity(mj1, mj2) = 0.96001103
Cosine_Similarity(mj2, cat) = 0.14183421
Cosine_Similarity(mj1, cat) = 0.15253511

The pair of mj1 and mj2 is the most similar in VGG representation and the pair of mj2 and cat is the most similar in pixel representation.

7

The problem that pixel representation has is that this approach only can recognize the value of each pixel and takes it as the feature, thus it will result in regarding two images with similar pixel values but different details as similar images.
The CNN is fixing this since the CNN approach takes the structure of an image as the feature, which can correctly describe the details of an image.

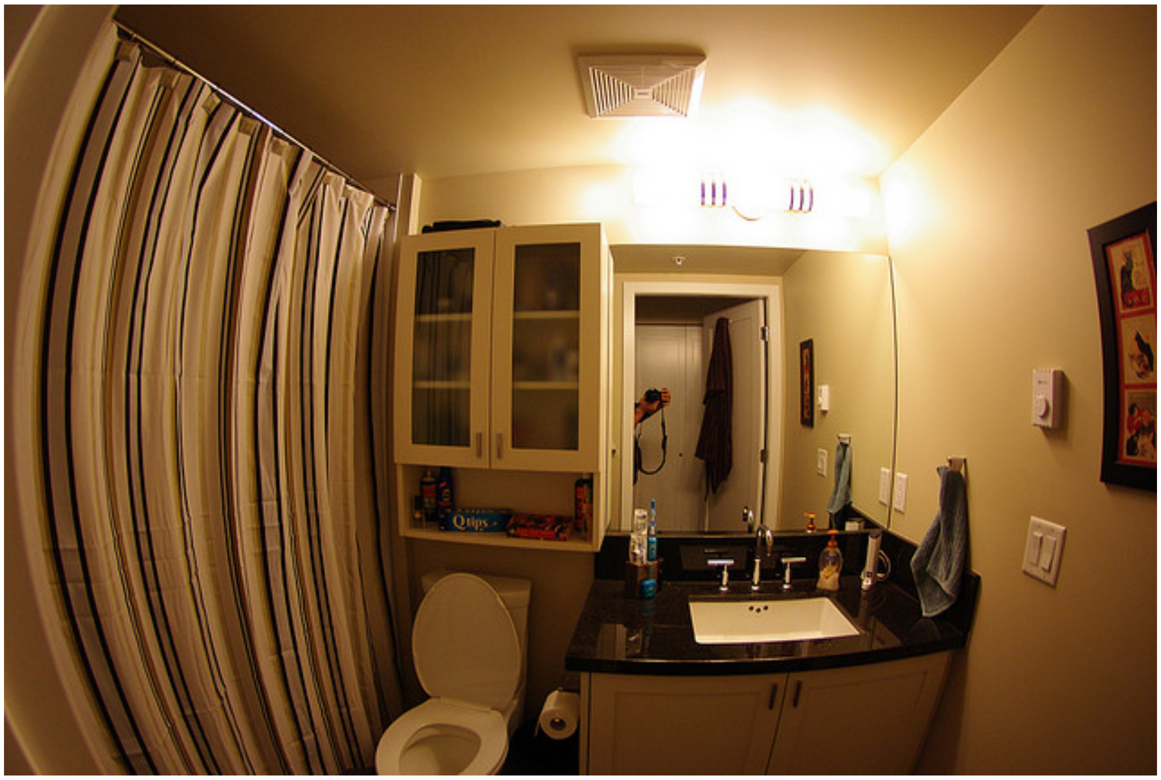
8

The result from VGG representation is better than the one from pixel representation. For VGG method, almost all pair of the generated caption and the true caption contains at least one same key word. However, for pixel method, most of the generated captions are totally irrelevant with correspond test picture.
Thus, VGG got the better result than the one got by Pixel.

9

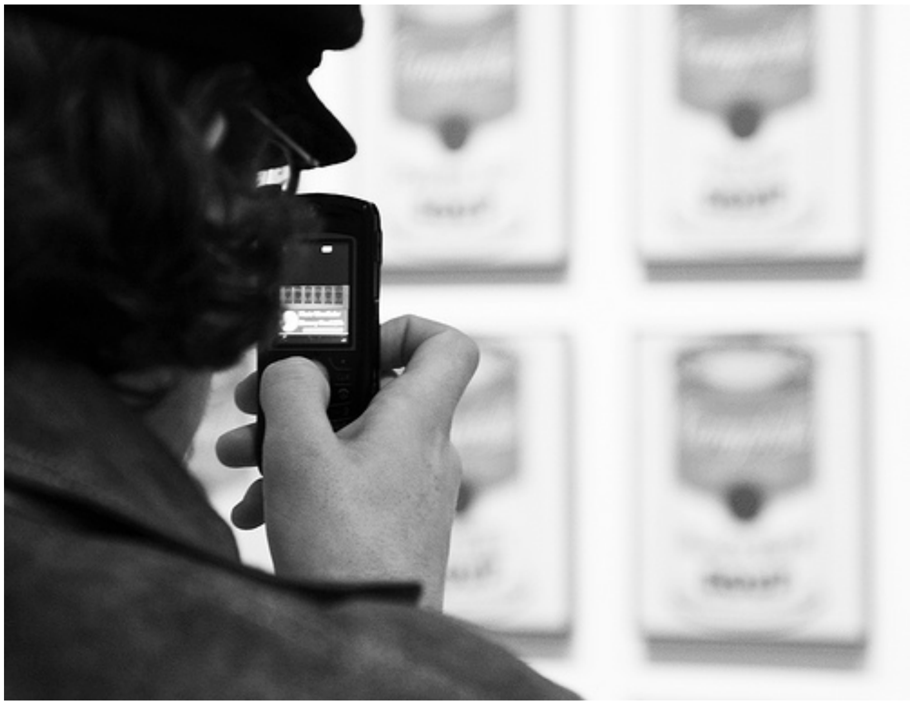
For VGG representation
Test picture: COCO_val2014_000000242934.jpg
Caption: A large living room with a black sectional and a recliner

Output picture: COCO_val2014_000000347740.jpg
Caption: A bathroom with a shower, toilet, and sink

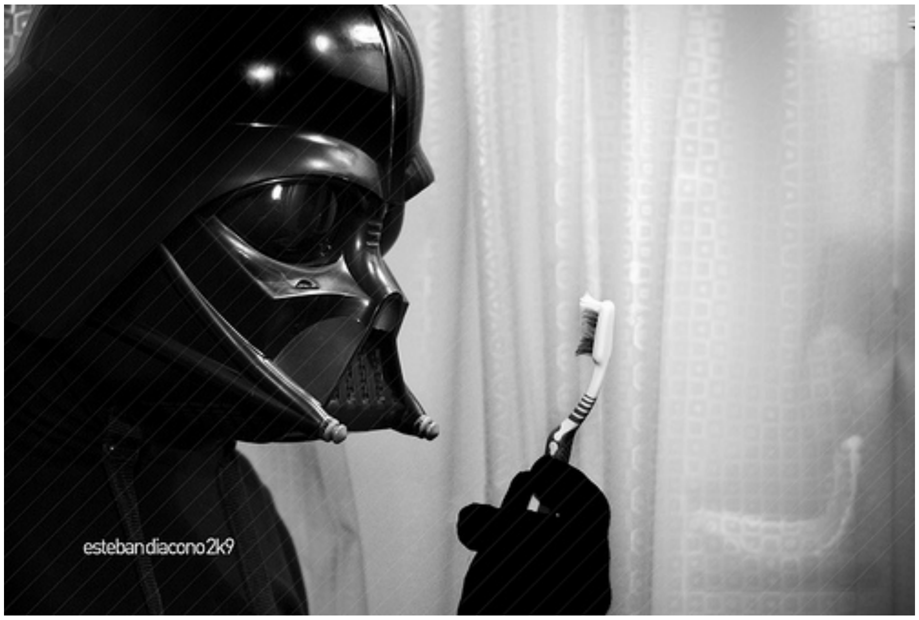


Although the generated caption described the test picture badly, there are still some similarities between the test picture and the output picture. Both picture are a room with some furniture and house stuff. The structures of the two pictures are quiet similar, which is the reason why VGG method gets the result. However, the content in detail of the two pictures are so different, and thus leads to the poor description.

For Pixel representation,
Test picture: ImageCOCO_val2014_000000183889.jpg
Caption: a man with a phone lifting it up for use



Output picture: ImageCOCO_val2014_000000411596.jpg
Caption: Darth Vader in a bathroom holding a toothbrush in one hand and staring at it.



For pixel presentation, we can rarely find any relation between the true caption and the test caption. However, when we look into the two pictures, we can find an obvious similarity on pixel levels that both picture are black-and-white pictures, which should also be the reason that produce the poor result.