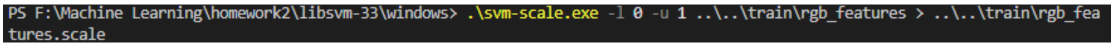
Michine Learning-Homework2

資訊碩一 11177016 董承宥

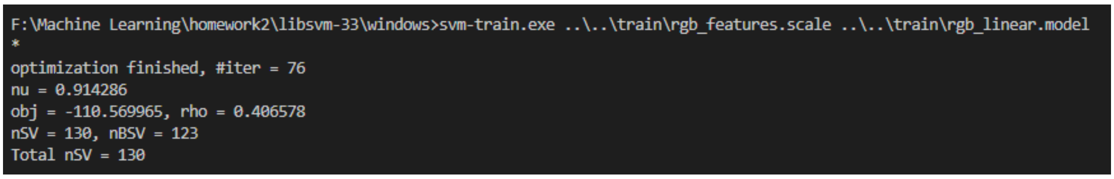
a. Using linear kernel and RGB color feature (400 × 3 dims for each image)

1. use to_svm.py to convert the raw data into svm file format.
2. Use svm-scale.exe to scale the converted files and data to [0,1].



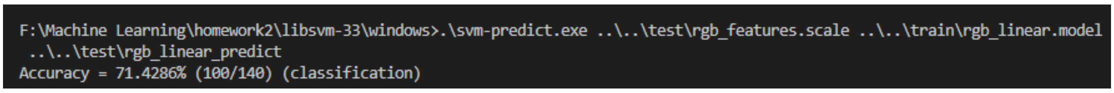
```
PS F:\Machine Learning\homework2\libsvm-33\windows> .\svm-scale.exe -l 0 -u 1 ..\..\train\rgb_features > ..\..\train\rgb_fea
tures.scale
```

3. Training with tuned data to generate models.



```
F:\Machine Learning\homework2\libsvm-33\windows>svm-train.exe ..\..\train\rgb_features.scale ..\..\train\rgb_linear.model
*
optimization finished, #iter = 76
nu = 0.914286
obj = -110.569965, rho = 0.406578
nSV = 130, nBSV = 123
Total nSV = 130
```

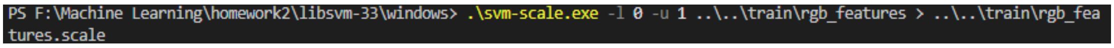
4. The accuracy of the prediction using the trained model is 71.4286%.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-predict.exe ..\..\test\rgb_features.scale ..\..\train\rgb_linear.model
 ..\..\test\rgb_linear_predict
Accuracy = 71.4286% (100/140) (classification)
```

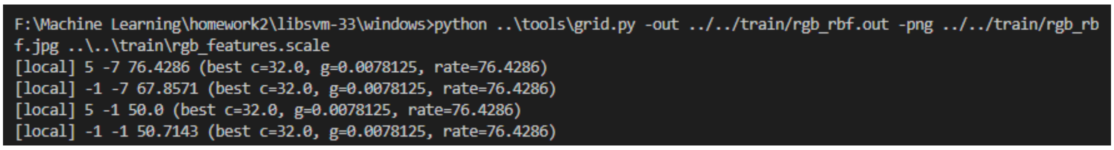
b. Using RBF kernel and RGB color feature

1. Use to_svm.py to convert the raw data into svm file format.
2. Use svm-scale.exe to scale the converted files and data to [0,1].

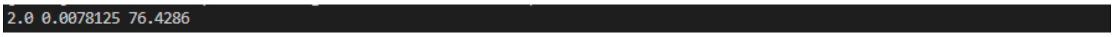


```
PS F:\Machine Learning\homework2\libsvm-33\windows> .\svm-scale.exe -l 0 -u 1 ..\..\train\rgb_features > ..\..\train\rgb_fea
tures.scale
```

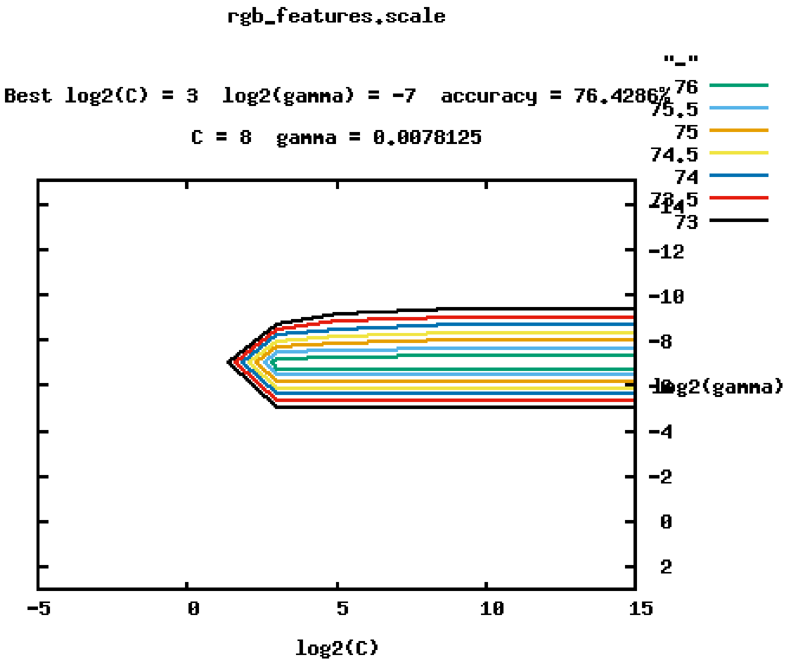
3. Use grip.py to find the best arguments.



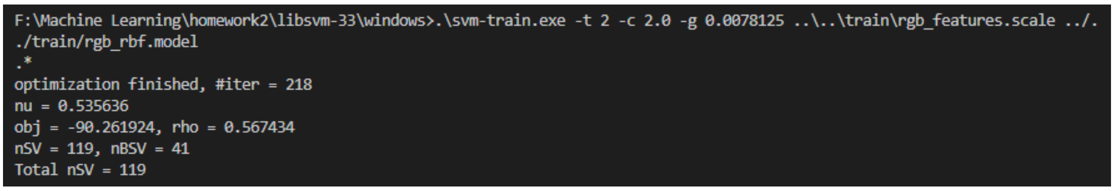
```
F:\Machine Learning\homework2\libsvm-33\windows>python ..\tools\grid.py -out ../../train/rgb_rbf.out -png ../../train/rgb_rb
f.jpg ..\..\train\rgb_features.scale
[local] 5 -7 76.4286 (best c=32.0, g=0.0078125, rate=76.4286)
[local] -1 -7 67.8571 (best c=32.0, g=0.0078125, rate=76.4286)
[local] 5 -1 50.0 (best c=32.0, g=0.0078125, rate=76.4286)
[local] -1 -1 50.7143 (best c=32.0, g=0.0078125, rate=76.4286)
```



```
2.0 0.0078125 76.4286
```

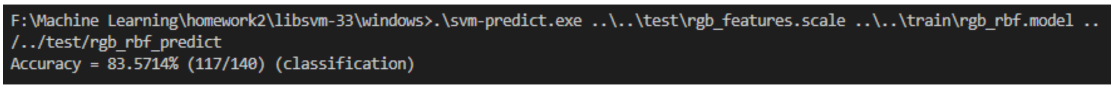


4. Model training using parameters obtained from grip.py.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-train.exe -t 2 -c 2.0 -g 0.0078125 ..\..\train\rgb_features.scale ../.
./train/rgb_rbf.model
.*
optimization finished, #iter = 218
nu = 0.535636
obj = -90.261924, rho = 0.567434
nSV = 119, nBSV = 41
Total nSV = 119
```

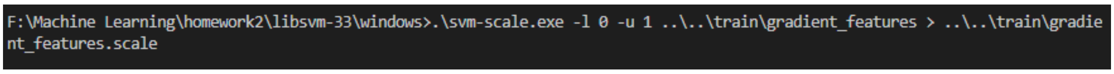
5. The accuracy of the prediction using the trained model is 83.5714%.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-predict.exe ..\..\test\rgb_features.scale ..\..\train\rgb_rbf.model ..
/../test/rgb_rbf_predict
Accuracy = 83.5714% (117/140) (classification)
```

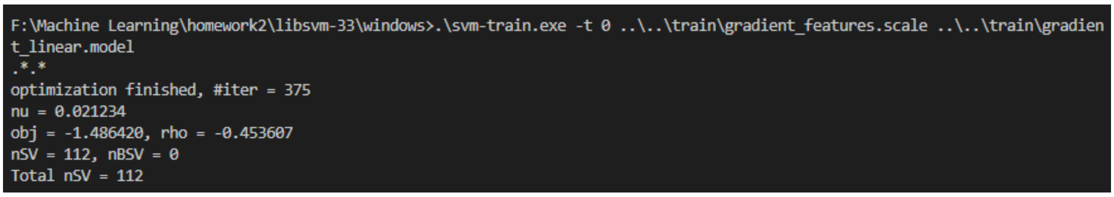
c. Using linear kernel and gradient feature (400 × 2 dims for each image)

1. Use to_svm.py to convert the raw data into svm file format.
2. Use svm-scale.exe to scale the converted files and data to [0,1].



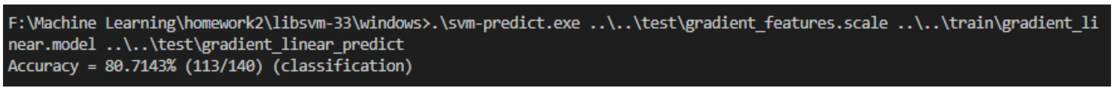
```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-scale.exe -l 0 -u 1 ..\..\train\gradient_features > ..\..\train\gradie
nt_features.scale
```

3. Training with tuned data to generate models.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-train.exe -t 0 ..\..\train\gradient_features.scale ..\..\train\gradien
t_linear.model
.*.*
optimization finished, #iter = 375
nu = 0.021234
obj = -1.486420, rho = -0.453607
nSV = 112, nBSV = 0
Total nSV = 112
```

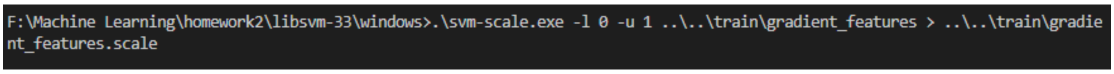
4. The accuracy of the prediction using the trained model is 80.7143%.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-predict.exe ..\..\test\gradient_features.scale ..\..\train\gradient_li
near.model ..\..\test\gradient_linear_predict
Accuracy = 80.7143% (113/140) (classification)
```

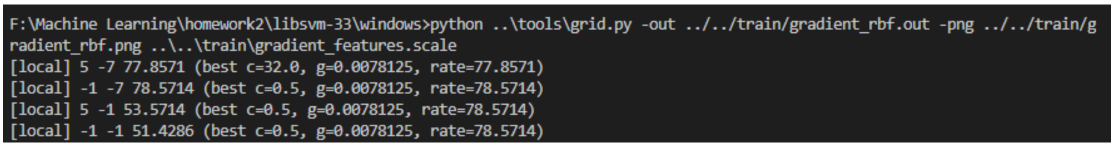
d. Using RBF kernel and gradient feature

1. Use to_svm.py to convert the raw data into svm file format.
2. Use svm-scale.exe to scale the converted files and data to [0,1].

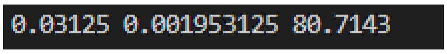


```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-scale.exe -l 0 -u 1 ..\..\train\gradient_features > ..\..\train\gradie
nt_features.scale
```

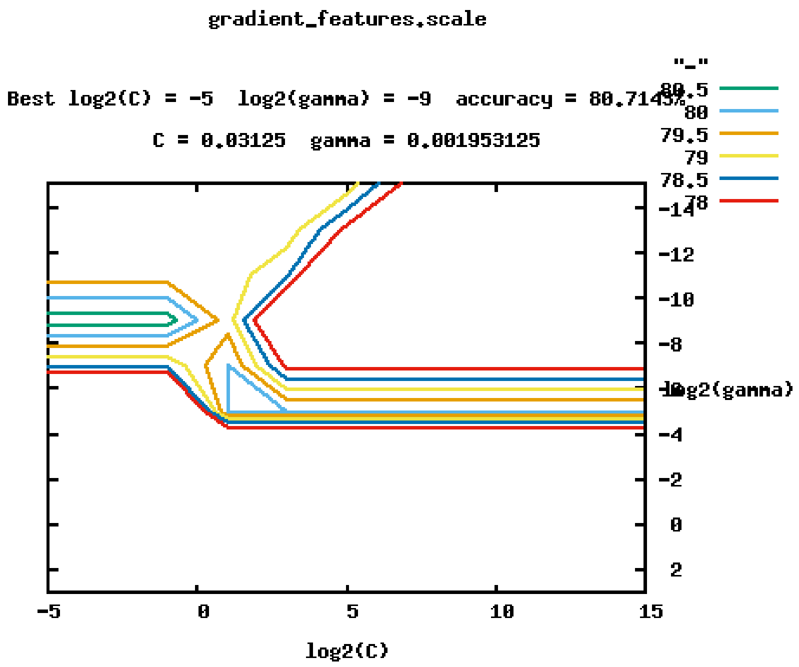
3. Use grip.py to find the best arguments.



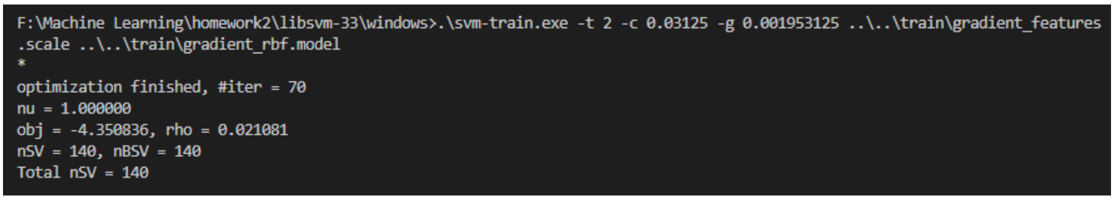
```
F:\Machine Learning\homework2\libsvm-33\windows>python ..\tools\grid.py -out ../../train/gradient_rbf.out -png ../../train/g
radient_rbf.png ..\..\train\gradient_features.scale
[local] 5 -7 77.8571 (best c=32.0, g=0.0078125, rate=77.8571)
[local] -1 -7 78.5714 (best c=0.5, g=0.0078125, rate=78.5714)
[local] 5 -1 53.5714 (best c=0.5, g=0.0078125, rate=78.5714)
[local] -1 -1 51.4286 (best c=0.5, g=0.0078125, rate=78.5714)
```



```
0.03125 0.001953125 80.7143
```

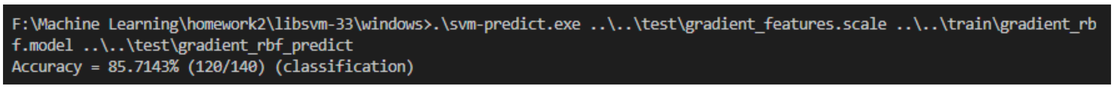


4. Model training using parameters obtained from grip.py.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-train.exe -t 2 -c 0.03125 -g 0.001953125 ..\..\train\gradient_features
.scale ..\..\train\gradient_rbf.model
*
optimization finished, #iter = 70
nu = 1.000000
obj = -4.350836, rho = 0.021081
nSV = 140, nBSV = 140
Total nSV = 140
```

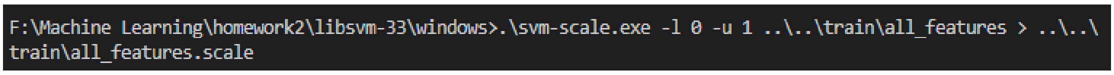
5. The accuracy of the prediction using the trained model is 85.7143%.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-predict.exe ..\..\test\gradient_features.scale ..\..\train\gradient_rb
f.model ..\..\test\gradient_rbf_predict
Accuracy = 85.7143% (120/140) (classification)
```

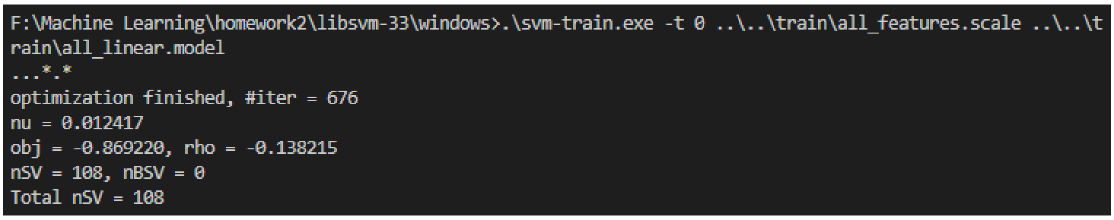
e. Using linear kernel and color+gradient feature (400 × 5 dims for each image)

1. Use to_svm.py to convert the raw data into svm file format.
2. Use svm-scale.exe to scale the converted files and data to [0,1].



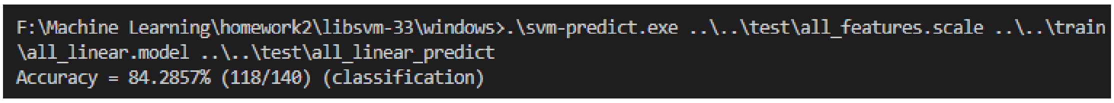
```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-scale.exe -l 0 -u 1 ..\..\train\all_features > ..\..\
train\all_features.scale
```

3. Training with tuned data to generate models.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-train.exe -t 0 ..\..\train\all_features.scale ..\..\t
rain\all_linear.model
...*.*
optimization finished, #iter = 676
nu = 0.012417
obj = -0.869220, rho = -0.138215
nSV = 108, nBSV = 0
Total nSV = 108
```

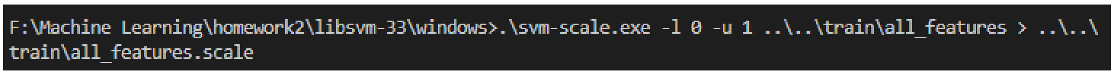
4. The accuracy of the prediction using the trained model is 84.2857%.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-predict.exe ..\..\test\all_features.scale ..\..\train
\all_linear.model ..\..\test\all_linear_predict
Accuracy = 84.2857% (118/140) (classification)
```

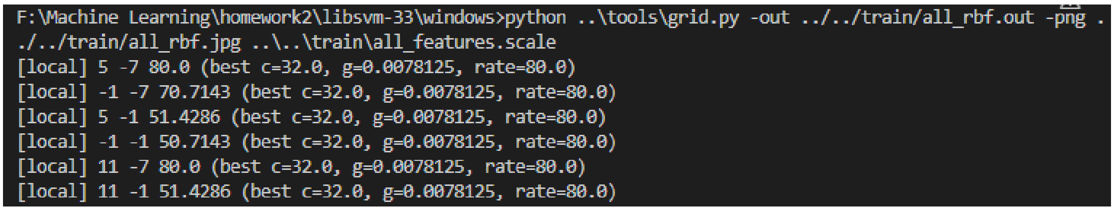
f. Using RBF kernel and color+gradient feature

1. Use to_svm.py to convert the raw data into svm file format.
2. Use svm-scale.exe to scale the converted files and data to [0,1].

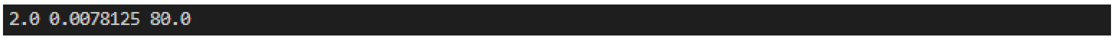


```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-scale.exe -l 0 -u 1 ..\..\train\all_features > ..\..\
train\all_features.scale
```

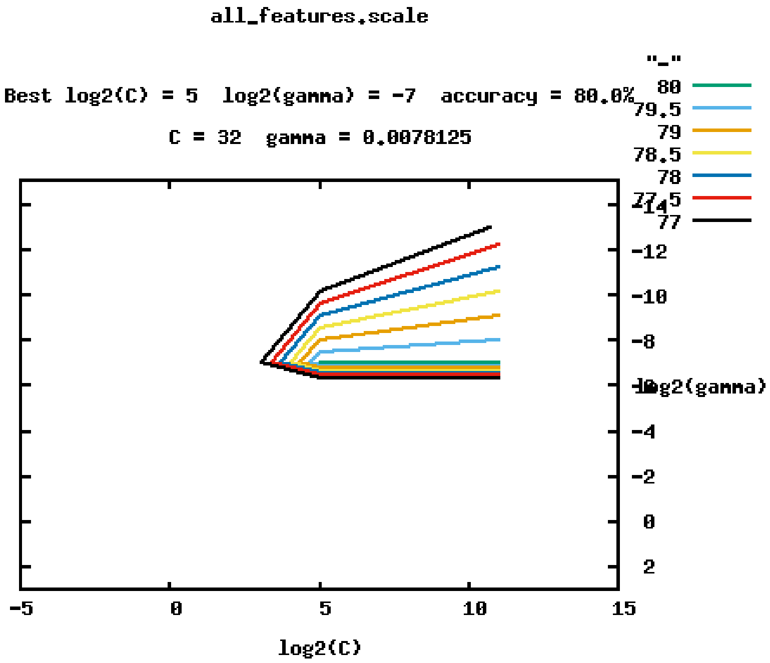
3. Use grip.py to find the best arguments.



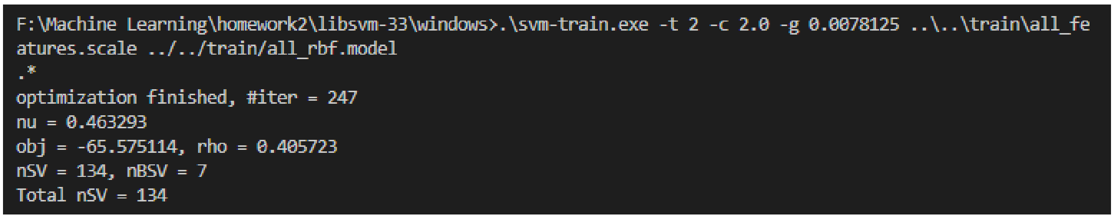
```
F:\Machine Learning\homework2\libsvm-33\windows>python ..\tools\grid.py -out ../../train/all_rbf.out -png .
./../train/all_rbf.jpg ..\..\train\all_features.scale
[local] 5 -7 80.0 (best c=32.0, g=0.0078125, rate=80.0)
[local] -1 -7 70.7143 (best c=32.0, g=0.0078125, rate=80.0)
[local] 5 -1 51.4286 (best c=32.0, g=0.0078125, rate=80.0)
[local] -1 -1 50.7143 (best c=32.0, g=0.0078125, rate=80.0)
[local] 11 -7 80.0 (best c=32.0, g=0.0078125, rate=80.0)
[local] 11 -1 51.4286 (best c=32.0, g=0.0078125, rate=80.0)
```



```
2.0 0.0078125 80.0
```

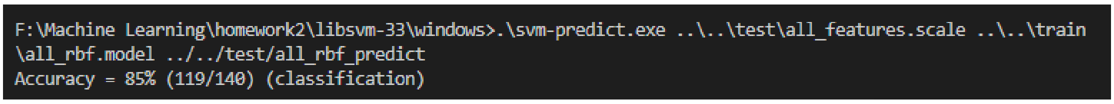


4. Model training using parameters obtained from grip.py.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-train.exe -t 2 -c 2.0 -g 0.0078125 ..\..\train\all_fe
atures.scale ../../train/all_rbf.model
.*
optimization finished, #iter = 247
nu = 0.463293
obj = -65.575114, rho = 0.405723
nSV = 134, nBSV = 7
Total nSV = 134
```

5. The accuracy of the prediction using the trained model is 85%.



```
F:\Machine Learning\homework2\libsvm-33\windows>.\svm-predict.exe ..\..\test\all_features.scale ..\..\train
\all_rbf.model ../../test/all_rbf_predict
Accuracy = 85% (119/140) (classification)
```

Conclusion

For both training schemes, RBF kernel is more accurate than linear kernel.
In terms of feature selection, the color+gradient feature trained model has the highest accuracy when using the linear kernel, but the color+gradient feature trained model does not have the best accuracy when using the RBF kernel.