For this image classification HW, the SVM implementation as a shallow learning algorithm is done by using scikit-learn library. SVM is used to learn a binary classification task. SVC class which stand for C-Support Vector Classification is used.

Input Analysis

Original Dataset	
Size	25008 images
Resolution	415x418
Storage Size (per image)	300.65 KB
Storage Size (dataset)	917.3 MB

Resolution above is found by taking the average of representative 10 images from the dataset since the dimensions in the dataset varies from image to image.

There are 2 classes in the dataset which are cats and dogs. Below, you can see two representative pictures from each class. The training vs. test split ratio is 8:2 .




Two different versions of the dataset is created by means of resolutions which are 64x64 and 32x32.

In addition to the resolutions, 5 different size of datasets are created from the original dataset for training and the analysis of the model performance. These dataset experiments are to determine how the framework behaves when it is trained on different binary classifiers. In each experiment, 80% of the images are used for training set and 20% is used for test set.

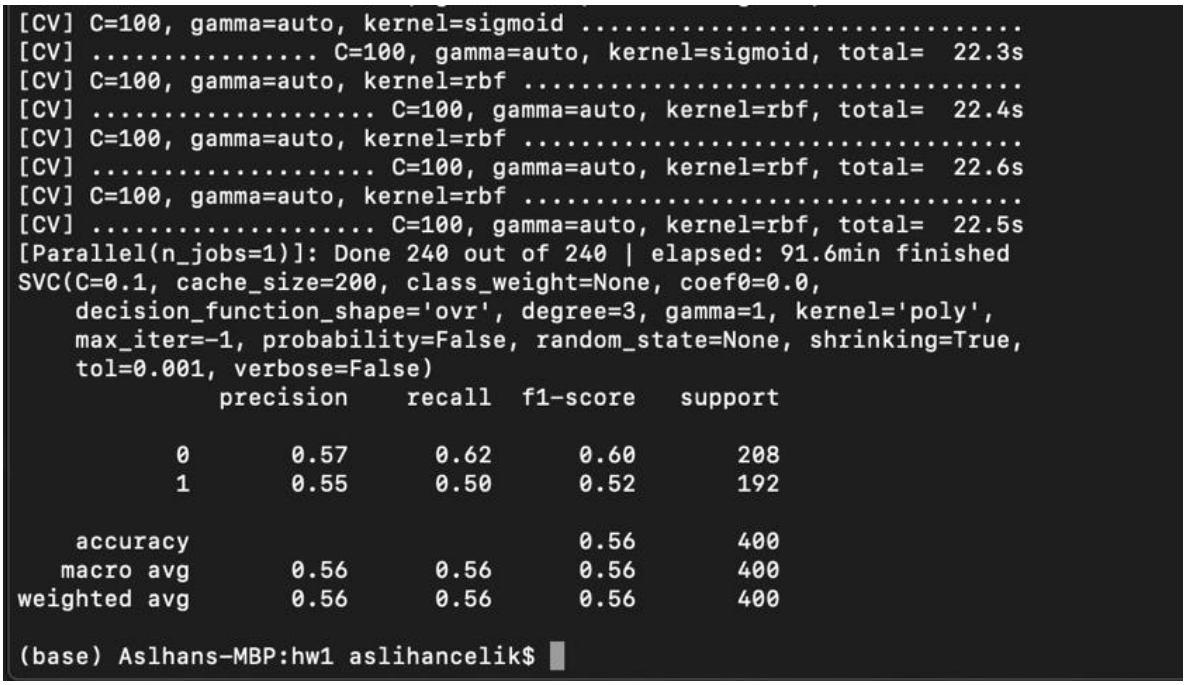
<u>64x64 Resolution Dataset</u>

Default Configuration of The Model (64x64 resolution dataset)

C	kernel	degree	gamma	tol	max_iter	Storage Size	Accuracy	Training Time
1.0	rbf	3	scale	0.001	-1	157.32MB	0.497	38.15 seconds

Max_iter -1 means no limit on the number of iterations.

I ran GridSearch in order to find the best combination of parameters for my model. I used GridSearch before both training any SVM model on my dataset and fitting it. Doing this on a dataset with 2000 pictures of cats and dogs in total, took 1 hour 32 minutes. The reason for this long run time is that the GridSearch started with a very big search space as the model is not initially trained with any parameters. In this case, GridSearch started its optimization process from default parameters. Below, you can see the screen shot of the classification report for the best estimator selected by GridSearch.



```
[CV] C=100, gamma=auto, kernel=sigmoid ..............................
[CV] ................ C=100, gamma=auto, kernel=sigmoid, total=  22.3s
[CV] C=100, gamma=auto, kernel=rbf ..................................
[CV] ..................... C=100, gamma=auto, kernel=rbf, total=  22.4s
[CV] C=100, gamma=auto, kernel=rbf ..................................
[CV] ..................... C=100, gamma=auto, kernel=rbf, total=  22.6s
[CV] C=100, gamma=auto, kernel=rbf ..................................
[CV] ..................... C=100, gamma=auto, kernel=rbf, total=  22.5s
[Parallel(n_jobs=1)]: Done 240 out of 240 | elapsed: 91.6min finished
SVC(C=0.1, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='poly',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
              precision    recall  f1-score   support

           0       0.57      0.62      0.60       208
           1       0.55      0.50      0.52       192

    accuracy                           0.56       400
   macro avg       0.56      0.56      0.56       400
weighted avg       0.56      0.56      0.56       400

(base) Aslhans-MBP:hw1 aslihancelik$
```

As can be seen from above, the best estimator according to GridSearch is by using a 3rd degree polynomial kernel, setting C=0.1 as the regularization parameter, gamma=1 as the kernel coefficient. The tol value is 0.001 which is already the default value. Tol value represent tolerance. It is the stopping criteria for optimization to end the training.

When the classification report above is observed, we can see that there is no bias in the dataset since the accuracy and the weighted accuracy are the same. Precision is the number of labels the model predicts correctly. Precision values are not very high but it is almost the same for both cat and dog classes. Recall is the fraction of relevant instances the model identified over the total relevant instances in the image. We can see that the recall value is higher for cat class. The f1-score is like the weighted average of the precision and recall. It is better if the f1 score has a value close to 1 and worse if it is close to zero.

Best Estimator Parameter Configuration (64x64 resolution)

C	kernel	degree	gamma	Tol	max_iter	Storage Size	Accuracy	Training Time
0.1	poly	3	1	0.001	-1	122.120 MB	0.56	35.65 seconds

Remark: We see that the accuracy of the model is improved (6% more) after tuning the parameters with GridSearch. Training time is very slightly improved.

32x32 Resolution Dataset

When GridSearch is ran for 32x32 resolution dataset with 2000 images it takes 41 minutes. Just like in the previous case the search space is big since the model is not trained prior to GridSearch. Even though the number of images in the training dataset were same as 64x64 resolution, the GridSearch for 32x32 resolution takes less time to find the best estimator.

```
[1] [CV] ..... C=100, gamma=0.001, kernel=sigmoid, total=  10.5s
    [CV] C=100, gamma=0.001, kernel=sigmoid .............................
    [CV] ..... C=100, gamma=0.001, kernel=sigmoid, total=  10.5s
    [CV] C=100, gamma=0.001, kernel=sigmoid .............................
    [CV] ..... C=100, gamma=0.001, kernel=sigmoid, total=  10.6s
    [Parallel(n_jobs=1)]: Done 240 out of 240 | elapsed: 41.4min finished
    SVC(C=0.1, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
        decision_function_shape='ovr', degree=3, gamma=1, kernel='poly',
        max_iter=-1, probability=False, random_state=None, shrinking=True,
        tol=0.001, verbose=False)
                  precision    recall  f1-score   support

               0       0.58      0.58      0.58       201
               1       0.58      0.58      0.58       199

        accuracy                           0.58       400
       macro avg       0.58      0.58      0.58       400
    weighted avg       0.58      0.58      0.58       400
```

While accuracy is similar to the model trained on 64x64 resolution dataset, values for precision, recall and f1-score are higher for the model trained on 32x32 resolution dataset. This might tell us that SVM performs better when there is less complexity(low resolution) in the dataset.

According to the values we have for precision, recall and f1score the model can perform the classification poorly because it only learns from pixels unlike the other classifiers such as Neural Network that can learn image features better. Therefore, SVM is not the best classifier to do image classification.

Best Estimator Parameter Configuration (32x32 resolution)

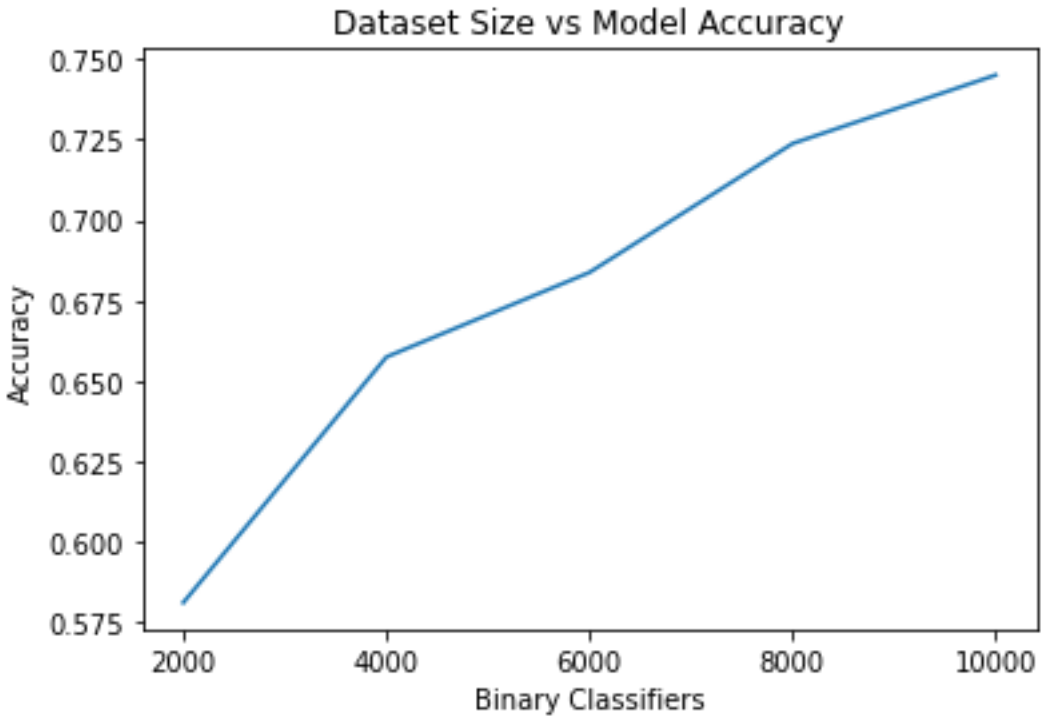
C	kernel	degree	gamma	Tol	max_iter	Storage Size	Accuracy
0.1	poly	3	1	0.001	-1	29.22 MB	0.58

Kernel Selection

I also specifically only evaluated different Kernel's for image classification with Support Vector Machine before running the GridSearch. To do this I used the 64x64 resolution dataset with 2000 images. According to the output below, linear and polynomial kernels provided the most promising results since the accuracy and the other model quality metrics are higher compared to the other two kernels. Both RBF and Sigmoid kernels did not guess any of the labels for cat class correctly and they performed poorly.

```
(base) Aslhans-MBP:hw1 aslihancelik$ python3 image_classify_deneme.py
Evaluation: Polynomial kernel
              precision    recall  f1-score   support

           0       0.57      0.62      0.60       208
           1       0.55      0.50      0.52       192

    accuracy                           0.56       400
   macro avg       0.56      0.56      0.56       400
weighted avg       0.56      0.56      0.56       400

Evaluation: RBF kernel
/Users/aslihancelik/opt/anaconda3/lib/python3.7/site-packages/sklearn/metrics/classification.py:1437: UndefinedMetricWarni
ng: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       208
           1       0.48      1.00      0.65       192

    accuracy                           0.48       400
   macro avg       0.24      0.50      0.32       400
weighted avg       0.23      0.48      0.31       400

Evaluation: Sigmoid kernel
/Users/aslihancelik/opt/anaconda3/lib/python3.7/site-packages/sklearn/metrics/classification.py:1437: UndefinedMetricWarni
ng: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       208
           1       0.48      1.00      0.65       192

    accuracy                           0.48       400
   macro avg       0.24      0.50      0.32       400
weighted avg       0.23      0.48      0.31       400

Evaluation: Linear kernel
              precision    recall  f1-score   support

           0       0.53      0.57      0.55       208
           1       0.49      0.44      0.46       192

    accuracy                           0.51       400
   macro avg       0.51      0.51      0.51       400
weighted avg       0.51      0.51      0.51       400
```

Dataset Size vs Model Accuracy



I obtained 5 different binary classifiers by preparing 5 different sizes of datasets retrieved from the original 32x32 resolution dataset. The SVM model for each classifier is trained by using the parameters obtained from GridSearch. SVM model is trained on datasets of sizes 2000, 4000, 6000, 8000 and 10000 in this order. Accuracy results of these datasets are plotted as you can see above. By training 10,000 images, we see that the accuracy of the model is increased to 74% while the accuracy was around 58% for the classifier that has 2000 size. According to this outcome the accuracy is increased 16% when the training data size is 5 times larger. According to the plot we see that the rate of increase in accuracy decreases as we increase the dataset size.

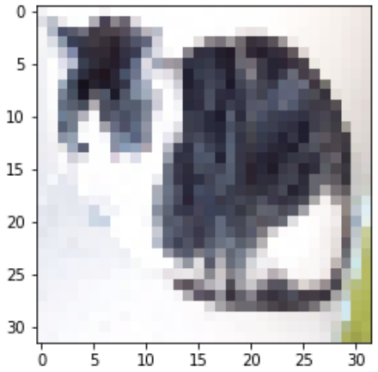
Models trained on these 5 different datasets can be found in “models_32x32Size” folder.

Let’s observe the performance of the models trained on some dataset examples with representative images from each. Each model uses the parameters suggested by GridSearch from the configuration tables you’ve seen previously.

This example is for 10,000 image dataset with 32x32 resolution. It can be hard for a person to identify the picture as a cat since the resolution is low. However, model performs okay since it is trained on 10,000 images and have a higher accuracy compared to models trained on smaller datasets.

	precision	recall	f1-score	support
0	0.72	0.74	0.73	103
1	0.72	0.70	0.71	97
accuracy			0.72	200
macro avg	0.72	0.72	0.72	200
weighted avg	0.72	0.72	0.72	200

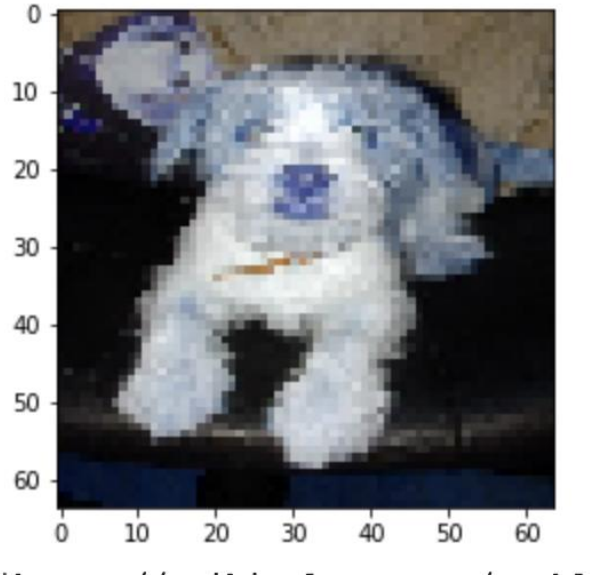
```
Accuracy: 0.72
Prediction is : Cat
```




```
'https://scikit-learn.org/stable/modules/generated/sklearn
```

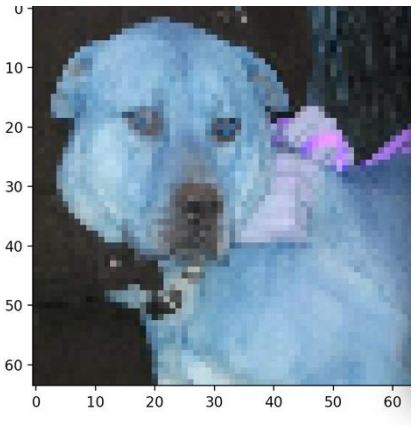
The example below is the result of a model trained on 64x64 resolution 2000 image dataset by using the parameters from GridSearch. I noticed when the data visualized again as an image, some colors are different than its original. I came across frequently that the beige color of the dog is visualized as blue when reshaping such as in the example on the right.

```
[LibSVM]Accuracy: 0.625
Prediction is : Dog
```



Here we can see an example where classifier doesn't work well since it labels the dog as a cat. This example is for the model trained on 1000 images that are converted to 64X64 resolution.

```
........*.*
optimization finished, #iter = 7789
obj = -0.000000, rho = 0.717966
nSV = 659, nBSV = 0
Total nSV = 659
[LibSVM]              precision    recall  f1-score   support

           0       0.58      0.66      0.62       100
           1       0.61      0.53      0.57       100

    accuracy                           0.59       200
   macro avg       0.60      0.59      0.59       200
weighted avg       0.60      0.59      0.59       200

Accuracy: 0.595
Prediction is : Cat
```

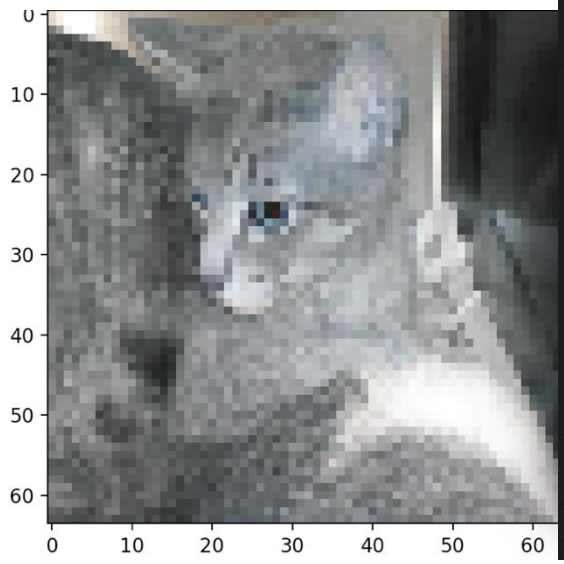


This example below, is for the model trained on 2000 images that are converted to 64X64 resolution. Number of iterations defaulted by the program is 12741. In the previous example with 1000 images, value for number of iterations was less which is 7789. As can be seen, the number of iterations are increased as the dataset increases.

```
........*....*
optimization finished, #iter = 12741
obj = -0.000000, rho = -0.201999
nSV = 1178, nBSV = 0
Total nSV = 1178
[LibSVM]              precision    recall  f1-score   support

           0       0.56      0.58      0.57       200
           1       0.56      0.54      0.55       200

    accuracy                           0.56       400
   macro avg       0.56      0.56      0.56       400
weighted avg       0.56      0.56      0.56       400

Accuracy: 0.56
Prediction is : Cat
(base) Aslhans-MBP:hw1 aslihancelik$
```

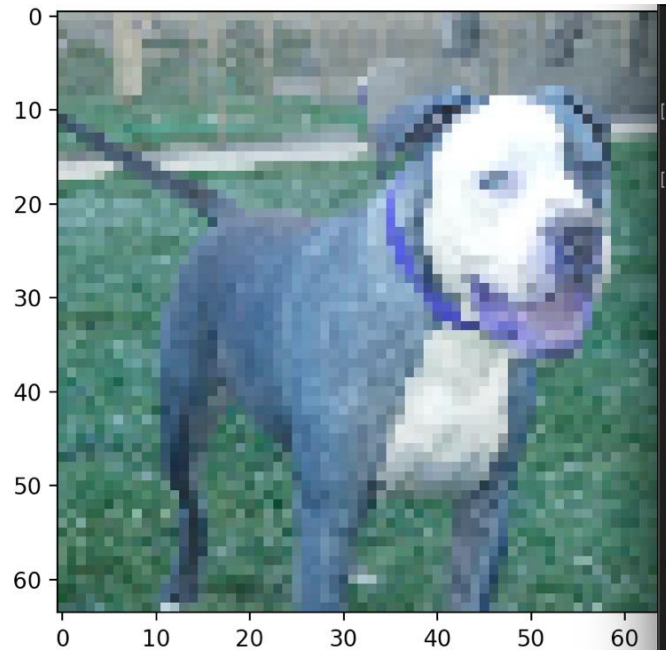
Training Process

Following three analysis are done for the models trained on 2000 image dataset with 64x64 resolution. When we set the max_iter number to 4000, we get the convergence warning. The model haven't reached the convergence and we trained it until the first intermediate stage of iterations we determined.



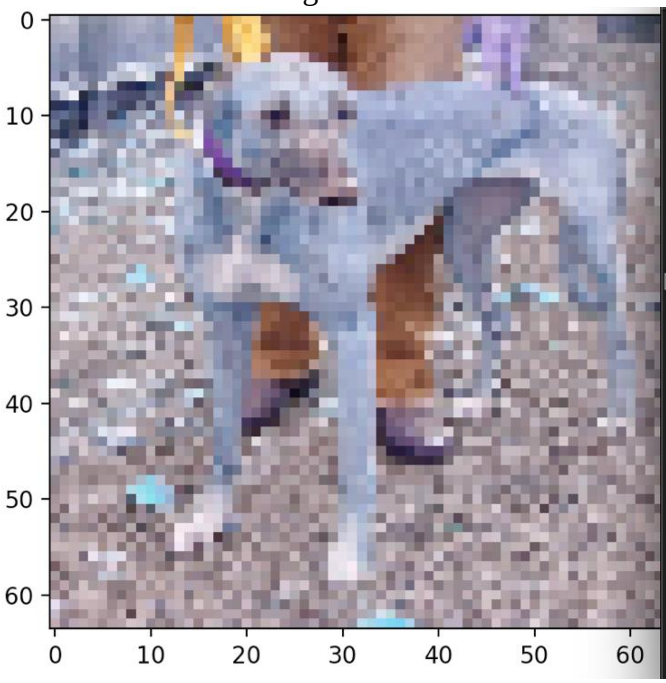
```
...WARN: libsvm Solver reached max_iter
optimization finished, #iter = 4000
obj = -0.000000, rho = 0.513773
nSV = 1250, nBSV = 0
Total nSV = 1250
/Users/aslihancelik/opt/anaconda3/lib/python3.7/site-packages/skl
earn/svm/base.py:241: ConvergenceWarning: Solver terminated early
 (max_iter=4000).  Consider pre-processing your data with Standar
dScaler or MinMaxScaler.
  % self.max_iter, ConvergenceWarning)
[LibSVM]              precision    recall  f1-score   support

           0       0.65      0.60      0.62       217
           1       0.56      0.61      0.59       183

    accuracy                           0.60       400
   macro avg       0.60      0.61      0.60       400
weighted avg       0.61      0.60      0.61       400

Accuracy: 0.605
Prediction is : Cat
```

When we set the max_iter number to 8000, we get the convergence warning as below. The model haven't reached the convergence and we trained it until some intermediate stages of iterations. However, the accuracy is slightly higher than the accuracy of the model when it reaches convergence.



```
Prediction is : Dog
^[[A
^Z
[1]+  Stopped                 python3 image_classify_deneme.py
(base) Aslhans-MBP:hw1 aslihancelik$ python3 image_classify_deneme.py
^Z
[2]+  Stopped                 python3 image_classify_deneme.py
(base) Aslhans-MBP:hw1 aslihancelik$ python3 image_classify_deneme.py
.......WARN: libsvm Solver reached max_iter
optimization finished, #iter = 8000
obj = -0.000000, rho = 0.484585
nSV = 1241, nBSV = 0
Total nSV = 1241
/Users/aslihancelik/opt/anaconda3/lib/python3.7/site-packages/sklearn/svm/base.py:241
rgenceWarning: Solver terminated early (max_iter=8000).  Consider pre-processing your
ith StandardScaler or MinMaxScaler.
  % self.max_iter, ConvergenceWarning)
[LibSVM]              precision    recall  f1-score   support

           0       0.62      0.65      0.64       205
           1       0.61      0.58      0.60       195

    accuracy                           0.62       400
   macro avg       0.62      0.62      0.62       400
weighted avg       0.62      0.62      0.62       400

Accuracy: 0.6175
Prediction is : Dog
```

For the example below, max_iter is not specified when training the Model 3. The accuracy of the first two models are slightly better than the accuracy of the model when it reaches convergence at the final stage.




```
   macro avg       0.62      0.62      0.62       400
weighted avg       0.62      0.62      0.62       400

Accuracy: 0.6175
Prediction is : Dog
^[[A
[1]-  Terminated: 15          python3 image_classify_deneme.py
(base) Aslhans-MBP:hw1 aslihancelik$ python3 image_classify_deneme.py
^[[A
^Z
[3]+  Stopped                 python3 image_classify_deneme.py
(base) Aslhans-MBP:hw1 aslihancelik$ python3 image_classify_deneme.py
.....*....*
optimization finished, #iter = 15880
obj = -0.000000, rho = 0.543647
nSV = 1235, nBSV = 0
Total nSV = 1235
[LibSVM]              precision    recall  f1-score   support

           0       0.59      0.63      0.61       203
           1       0.59      0.55      0.57       197

    accuracy                           0.59       400
   macro avg       0.59      0.59      0.59       400
weighted avg       0.59      0.59      0.59       400

Accuracy: 0.5875
Prediction is : Dog
```

	Number of Iterations	Accuracy	Model Size	Training Time
Model 1	4000	0.60	127.036 MB	31.9 seconds
Model 2	8000	0.62	125.168 MB	33.18 seconds
Model 3	15880	0.59	122.120 MB	39.36 seconds

The number of iterations to reach the convergence at final stage is selected by default and this default value is not necessarily the optimal number of iterations to get the best result for optimizing the model objective.

Conclusion

SVM is not the best model for image classification as it only tries to learn from the pixels. On the other hand, feature extraction methods such as in Neural Networks work better for image classification. Above experiments and results support the idea that SVM should not be the first preferred method for image classification.

Dataset Resource
https://www.kaggle.com/c/dogs-vs-cats

Environments used to run the code: both my local terminal and Google Collab.