In [1]:
```
import numpy as np
import pandas as pd
import os
from Utilities.evaluation_utils import *
from Utilities.ensemble_utils import *
from DataPreparation.dataset_preparation import get_catsvsdogs_dataset

%matplotlib inline
```

Load Dataset

In [2]:
```
data_dir = 'Dataset/'
validation_split = 0.2
split_seed = 6135
```

In [3]:
```
X_train, y_train, X_val, y_val, X_test, X_train_moments = get_catsvsdogs_dataset(
                                                validation_split, split_seed, normalize_t
mean_img, std_img = X_train_moments
print('Train data size: ', X_train.shape)
print('Train labels size: ', y_train.shape)
print('Val data size: ', X_val.shape)
print('Val labels size: ', y_val.shape)
print('Test data size: ', X_test.shape)
```

```
Train data size:  (15999, 64, 64, 3)
Train labels size:  (15999,)
Val data size:  (3999, 64, 64, 3)
Val labels size:  (3999,)
Test data size:  (4999, 64, 64, 3)
```

Validation Logits

Load Validation Predictions

In [4]:
```
PATH = 'CSV/'
file_names = os.listdir(f'{PATH}')
file_names = [file_name for file_name in file_names if 'val' in file_name]
model_val_logits = load_model_logits(PATH, file_names)
model_val_predictions = logits2predictions(model_val_logits)
print('%d validation prediction CSVs loaded.' % len(model_val_predictions))
```

```
17 validation prediction CSVs loaded.
```

Search For Best Ensemble on Validation Set

In [5]:
```
num_ensemble_models = 10
num_weight_search = 50
best_ensemble = ensemble_search(model_val_predictions, y_val,
                                num_ensemble_models, num_weight_search)
```

```
Searching all possible 10 model combinations.
Trying 50 random weights for each combination.
-----------
(05:08:30 PM) 0/19448 combinations searched...
(05:08:57 PM) 3889/19448 combinations searched...
(05:09:23 PM) 7778/19448 combinations searched...
(05:09:49 PM) 11667/19448 combinations searched...
(05:10:15 PM) 15556/19448 combinations searched...
(05:10:41 PM) 19445/19448 combinations searched...
Search done! Best accuracy achieved: 95.999%
```

Print Best Ensemble Found from Search

In [6]:
```
print('Best Result: Accuracy  %.3f%%' % (100*best_ensemble['acc']))
print('Combination:')
print(best_ensemble['combination'])
print('Weights:')
print(best_ensemble['weights'])
```

```
Best Result: Accuracy  95.999%
Combination:
['B_VGG19_probs_val_40000.csv' 'B_VGG19_probs_val_45000.csv'
 'B_VGG19_probs_val_55000.csv' 'C_VGG19_probs_val_45000.csv'
 'C_VGG19_probs_val_55000.csv' 'D_Wide28_10_probs_val_35000.csv'
 'E_Wide28_10_probs_val_40000.csv' 'E_Wide28_10_probs_val_45000.csv'
 'E_Wide28_10_probs_val_50000.csv' 'E_Wide28_10_probs_val_55000.csv']
Weights:
[0.09580687 0.01754878 0.20608302 0.05645623 0.11940935 0.08831574
 0.22459627 0.11709245 0.04051778 0.03417352]
```

The Best Ensemble (for Validation Set) We Found

In [7]:
```
# best_combination = ['A_VGG19_probs_val_70000.csv', 'A_VGG19_probs_val_80000.csv
#                     'B_VGG19_probs_val_55000.csv', 'C_VGG19_probs_val_55000.csv
#                     'D_Wide28_10_probs_val_15000.csv', 'D_Wide28_10_probs_val_2
#                     'E_Wide28_10_probs_val_40000.csv', 'E_Wide28_10_probs_val_4
#                     'E_Wide28_10_probs_val_50000.csv', 'E_Wide28_10_probs_val_6
# best_weights = np.array([0.08185934, 0.02822021,
#                          0.20549974, 0.10928467,
#                          0.00154461, 0.13861998,
#                          0.21291829, 0.02852817,
#                          0.07212696, 0.12139803])

best_combination = ['A_VGG19_probs_val_70000.csv', 'B_VGG19_probs_val_45000.csv',
                    'B_VGG19_probs_val_55000.csv', 'C_VGG19_probs_val_55000.csv',
                    'C_VGG19_probs_val_60000.csv', 'D_Wide28_10_probs_val_35000.c
                    'E_Wide28_10_probs_val_40000.csv', 'E_Wide28_10_probs_val_450
                    'E_Wide28_10_probs_val_50000.csv', 'E_Wide28_10_probs_val_600
best_weights = np.array([0.20819787, 0.02486325,
                         0.17434291, 0.,
                         0.07626797, 0.05308012,
                         0.19968367, 0.03093464,
                         0.08773788, 0.14489168])

ensemble_result = ensemble_models(model_val_predictions,
                                  best_combination,
                                  best_weights)
ensemble_acc = accuracy(ensemble_result, y_val)
print('Accuracy: %.3f%%' % (ensemble_acc * 100))
```

```
Accuracy: 96.024%
```

Ensembling for Test Set

Ensembe Test Predictions

In [8]:
```
best_combination = ['A_VGG19_probs_70000.csv', 'B_VGG19_probs_45000.csv',
                    'B_VGG19_probs_55000.csv', 'C_VGG19_probs_55000.csv',
                    'C_VGG19_probs_60000.csv', 'D_Wide28_10_probs_35000.csv',
                    'E_Wide28_10_probs_40000.csv', 'E_Wide28_10_probs_45000.csv',
                    'E_Wide28_10_probs_50000.csv', 'E_Wide28_10_probs_60000.csv']
test_logits = load_model_logits(PATH, best_combination)
test_predictions = logits2predictions(test_logits)
ensemble_result = ensemble_models(test_predictions, best_combination, best_weight
```

Save Test Predictions as CSV

In [9]:
```
predictions = np.argmax(ensemble_result, axis=1)
labels = ['Cat', 'Dog']
save_predictions(predictions, labels, None, 'Ensemble_Test_Final')
```

```
Ensemble_Test_Final.csv saved.
```

In [ ]:

In [ ]: