```
!pip install turicreate
```

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

Using deep features to train an image classifier

```
import turicreate
```

Load some data

```
image_train = turicreate.SFrame('/content/gdrive/My Drive/Turicreate/Week 6/image_train_data/')
image_test = turicreate.SFrame('/content/gdrive/My Drive/Turicreate/Week 6/image_test_data/')
```

Explore this image data

```
image_train['image'].explore()
```

```
/usr/local/lib/python3.7/dist-packages/turicreate/visualization/_plot.py:461: UserWarning: Di
  warnings.warn("Displaying only the first {} rows.".format(maximum_rows))
```

	SArray
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	

20 

21 

22 

23 

24 

25 

26 

27 

28 

29 

30 

31 

32 

33 

34 

35 

36 

37 

38 

39 

40 

41 

42 

43 

44 

45 

46 

47 

48 

49 

50 

51 

52 

53 

54 

55 

56 

57 

58 

59 

60 

61 

62 

63 

64 

65 

66 

67 

68 

69 

70 

71 

72 

73 

74 

75 

76 

77 

78 

79 

80 

81 

82 

83 

84 

85 

86 

87	
88	
89	
90	

Train an image classifier on raw image pixels

```
raw_pixel_model = turicreate.logistic_classifier.create(image_train,
                                                         target = 'label',
                                                         features = ['image_array'])
```

```
PROGRESS: Creating a validation set from 5 percent of training data. This may take a while.
          You can set ``validation_set=None`` to disable validation tracking.

Logistic regression:
--------------------------------------------------------
Number of examples          : 1904
Number of classes           : 4
Number of feature columns   : 1
Number of unpacked features : 3072
Number of coefficients      : 9219
Starting L-BFGS
--------------------------------------------------------
+-----------+----------+-----------+--------------+-------------------+---------------------+
| Iteration | Passes   | Step size | Elapsed Time | Training Accuracy | Validation Accuracy |
+-----------+----------+-----------+--------------+-------------------+---------------------+
| 0         | 5        | 0.018397  | 1.458064     | 0.272059          | 0.257426            |
| 1         | 10       | 1.563776  | 2.061635     | 0.363971          | 0.356436            |
| 2         | 13       | 0.781888  | 2.486026     | 0.402836          | 0.415842            |
| 3         | 14       | 0.977360  | 2.732055     | 0.377626          | 0.257426            |
| 4         | 20       | 2.509078  | 3.417364     | 0.462185          | 0.425743            |
| 9         | 30       | 1.095213  | 5.103655     | 0.529937          | 0.445545            |
+-----------+----------+-----------+--------------+-------------------+---------------------+
```

Make predictions using simple raw pixel model

```
image_test[0:3]['image'].explore()
```

SArray

```
image_test[0:3]['label']
```

```
dtype: str
Rows: 3
['cat', 'automobile', 'cat']
```

```
raw_pixel_model.predict(image_test[0:3])
```

```
dtype: str
Rows: 3
['bird', 'cat', 'bird']
```

Evaluate the raw pixel model on the test data

```
raw_pixel_model.evaluate(image_test)
```

```
{'accuracy': 0.4795, 'auc': 0.7275715000000003, 'confusion_matrix': Columns:
        target_label    str
        predicted_label str
        count   int

 Rows: 16

 Data:
 +--------------+-----------------+-------+
 | target_label | predicted_label | count |
 +--------------+-----------------+-------+
 |     bird     |       dog       |  155  |
 |     dog      |       cat       |  278  |
 |     cat      |       cat       |  395  |
 |     bird     |    automobile   |  151  |
 |  automobile  |    automobile   |  637  |
 |     dog      |    automobile   |  113  |
 |     dog      |       dog       |  398  |
 |     cat      |       dog       |  279  |
 |     bird     |       cat       |  206  |
 |  automobile  |       bird      |   89  |
 +--------------+-----------------+-------+
 [16 rows x 3 columns]
 Note: Only the head of the SFrame is printed.
 You can use print_rows(num_rows=m, num_columns=n) to print more rows and columns., 'f1_score
        threshold       float
        fpr     float
        tpr     float
        p       int
        n       int
        class   int

 Rows: 4004
```