Andrew Vu - CS156_HW6

March 31, 2022

1 CS156 (Introduction to AI), Spring 2022

2 Homework 6 submission

2.0.1 Roster Name: Andrew Vu

2.0.2 Student ID: 015055911

2.0.3 Email address: andrew.k.vu@sjsu.edu

Any special notes or anything you would like to communicate to me about this homework submission goes in here.

2.1 References and sources

List all your references and sources here. This includes all sites/discussion boards/blogs/posts/etc. where you grabbed some code examples.

- Perceptron.Breast file
- https://www.studytonight.com/python-howtos/how-to-round-floating-value-to-two-decimals-in-python

2.2 Solution

Load libraries and set random number generator seed

[60]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Perceptron
from sklearn.metrics import plot_confusion_matrix

from sklearn.model_selection import train_test_split
```

[61]:
```
np.random.seed(42)
```

Code the solution

2.2.1 Load the Dataset

[62]:
```
digits = datasets.load_digits()
X = digits.data
Y = digits.target
class_names = digits.target_names
X.shape, Y.shape, class_names
```

[62]:
```
((1797, 64), (1797,), array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]))
```

[63]:
```
digits_df = pd.DataFrame(X, columns=digits.feature_names)
digits_df['output_digit'] = Y
digits_df.head()
```

[63]:

	pixel_0_0	pixel_0_1	pixel_0_2	pixel_0_3	pixel_0_4	pixel_0_5	\
0	0.0	0.0	5.0	13.0	9.0	1.0	
1	0.0	0.0	0.0	12.0	13.0	5.0	
2	0.0	0.0	0.0	4.0	15.0	12.0	
3	0.0	0.0	7.0	15.0	13.0	1.0	
4	0.0	0.0	0.0	1.0	11.0	0.0	

	pixel_0_6	pixel_0_7	pixel_1_0	pixel_1_1	…	pixel_6_7	pixel_7_0	\
0	0.0	0.0	0.0	0.0	…	0.0	0.0	
1	0.0	0.0	0.0	0.0	…	0.0	0.0	
2	0.0	0.0	0.0	0.0	…	0.0	0.0	
3	0.0	0.0	0.0	8.0	…	0.0	0.0	
4	0.0	0.0	0.0	0.0	…	0.0	0.0	

	pixel_7_1	pixel_7_2	pixel_7_3	pixel_7_4	pixel_7_5	pixel_7_6	\
0	0.0	6.0	13.0	10.0	0.0	0.0	
1	0.0	0.0	11.0	16.0	10.0	0.0	
2	0.0	0.0	3.0	11.0	16.0	9.0	
3	0.0	7.0	13.0	13.0	9.0	0.0	
4	0.0	0.0	2.0	16.0	4.0	0.0	

	pixel_7_7	output_digit
0	0.0	0
1	0.0	1
2	0.0	2
3	0.0	3
4	0.0	4

[5 rows x 65 columns]

2.2.2 Convert the output digit to binary classifier (one-hot encoding)

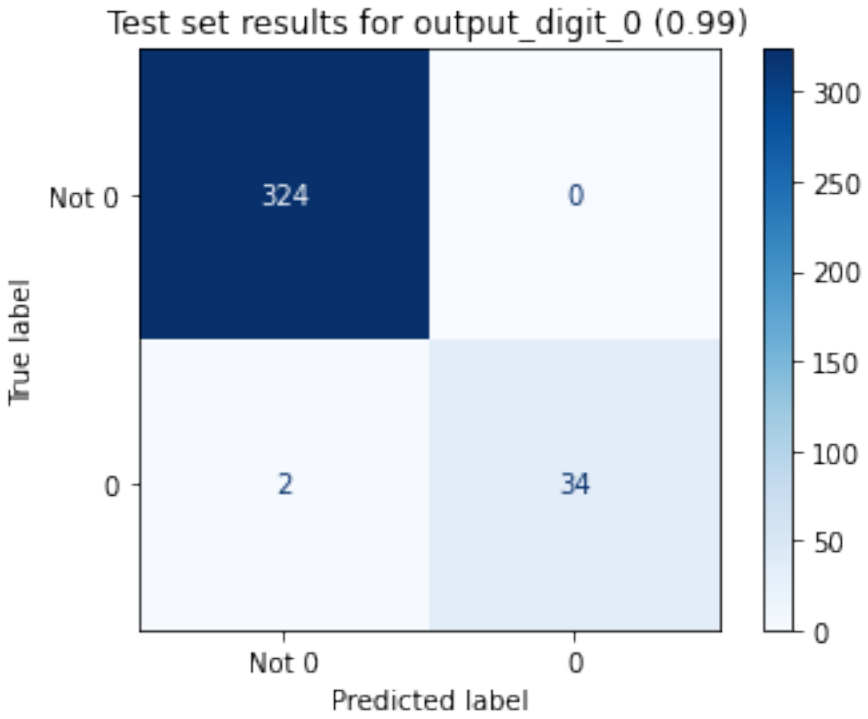
[64]:
```
df_numeric = pd.get_dummies(digits_df, columns=['output_digit'],␣
↪prefix=['output_digit'])
df_numeric

converted = ['output_digit_0', 'output_digit_1', 'output_digit_2',␣
↪'output_digit_3',
             'output_digit_4', 'output_digit_5', 'output_digit_6',␣
↪'output_digit_7',
             'output_digit_8', 'output_digit_9']
```

2.2.3 Split data into train/test –> train perceptron –> evaluate accuracy of perceptron –> plot confusion matrix

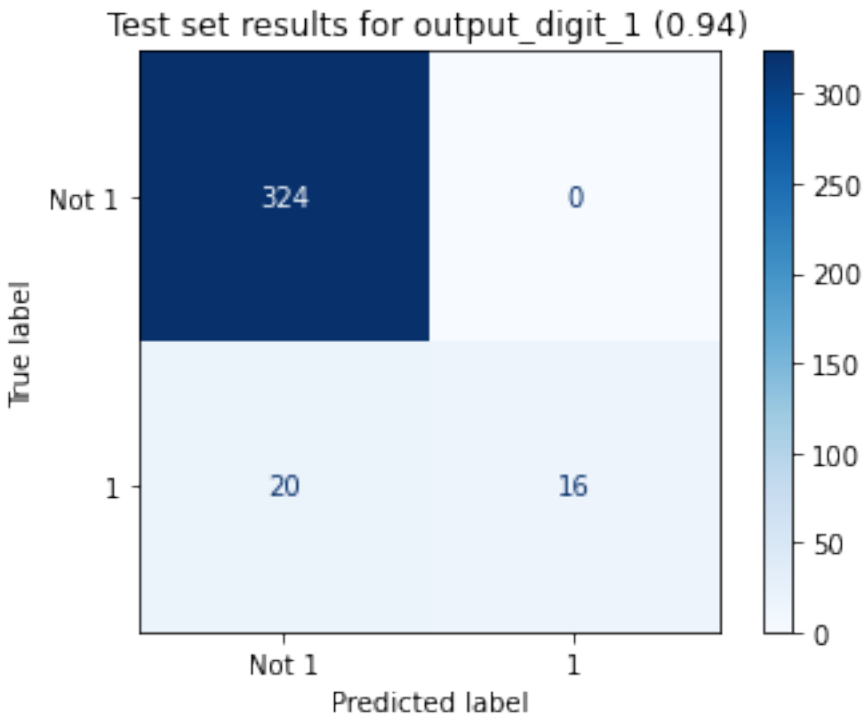
[65]:
```
X = X.astype("float32") / 255
for i in range(10):
    Y_new = df_numeric[converted[i]]
    X_train, X_test, Y_train, Y_test = train_test_split(X, Y_new, test_size=0.
↪2, random_state=0, stratify=Y)

    model = Perceptron(tol=1e-3, random_state=0).fit(X_train, Y_train)
    print('Accuracy of perceptron on training set: {:.2f}'.format(model.
↪score(X_train, Y_train)))
    print('Accuracy of perceptron on test set: {:.2f}'.format(model.
↪score(X_test, Y_test)))

    model_acc = "{:.2f}".format(model.score(X_test, Y_test))
    title = "Test set results for " + converted[i] + " (" + model_acc + ")"
    print('\n')


    disp = plot_confusion_matrix(model, X_test, Y_test,
                                 display_labels=['Not ' + str(i), str(i)],
                                 cmap=plt.cm.Blues,
                                 normalize=None)
    disp.ax_.set_title(title)

    print(title)
    print(disp.confusion_matrix)

    plt.show()
```

```
Accuracy of perceptron on training set: 0.99
Accuracy of perceptron on test set: 0.99


Test set results for output_digit_0 (0.99)
```

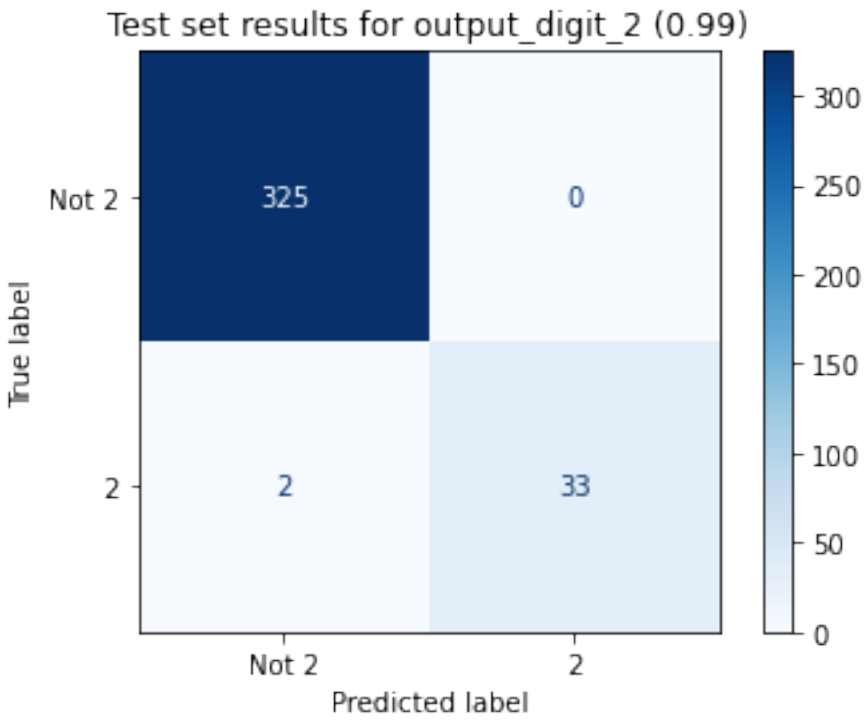
```
[[324   0]
 [  2  34]]
```



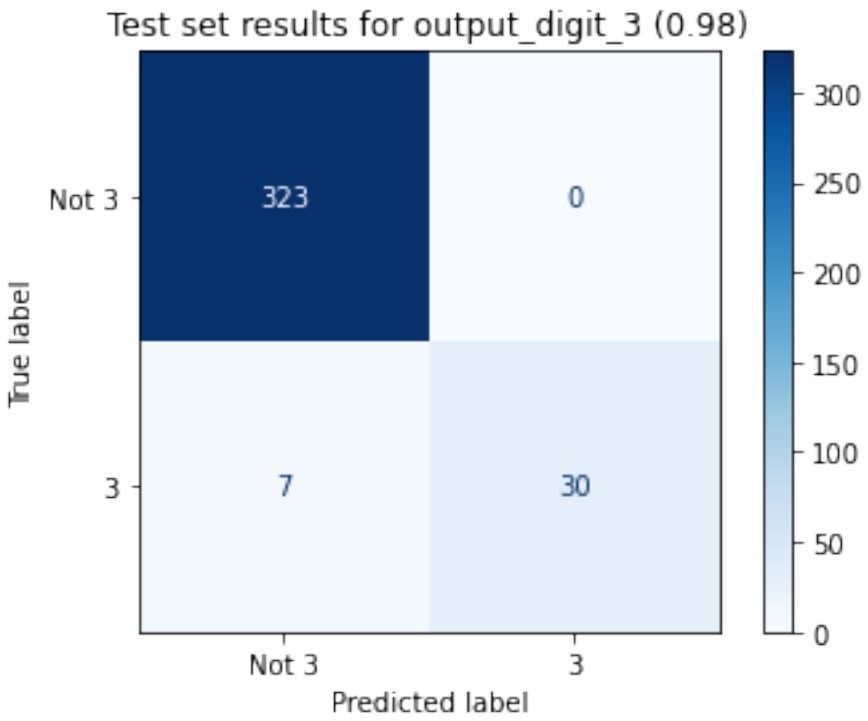

```
Accuracy of perceptron on training set: 0.95
Accuracy of perceptron on test set: 0.94


Test set results for output_digit_1 (0.94)
[[324   0]
 [ 20  16]]
```



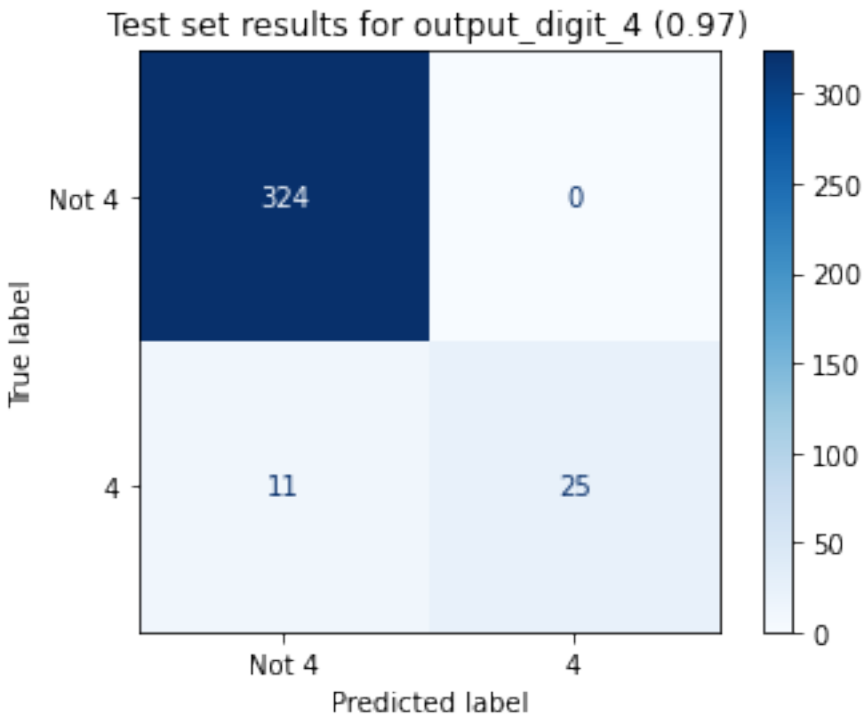

```
Accuracy of perceptron on training set: 0.99
Accuracy of perceptron on test set: 0.99


Test set results for output_digit_2 (0.99)
[[325   0]
 [  2  33]]
```



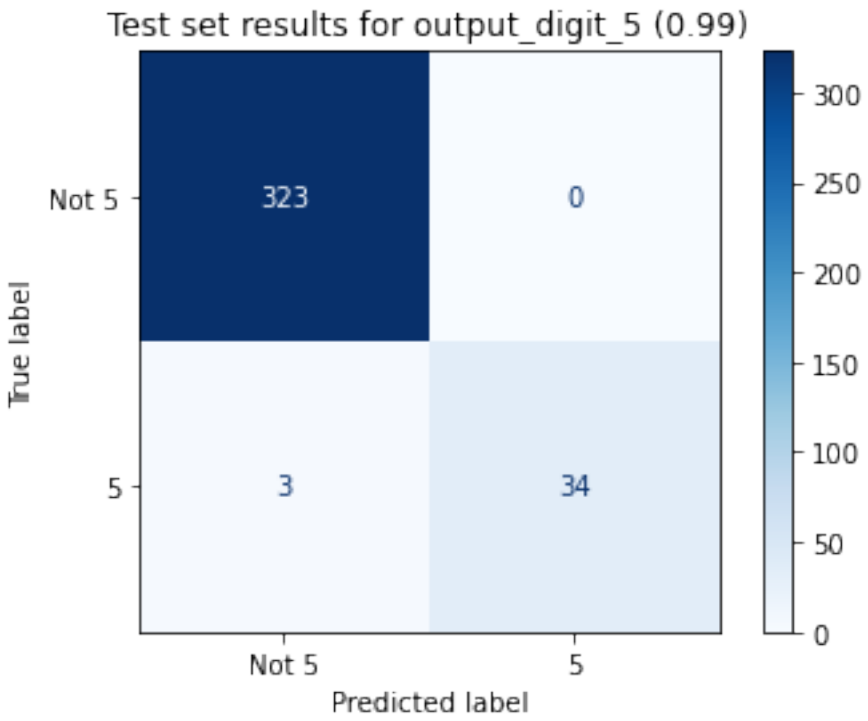

```
Accuracy of perceptron on training set: 0.98
Accuracy of perceptron on test set: 0.98


Test set results for output_digit_3 (0.98)
[[323   0]
 [  7  30]]
```



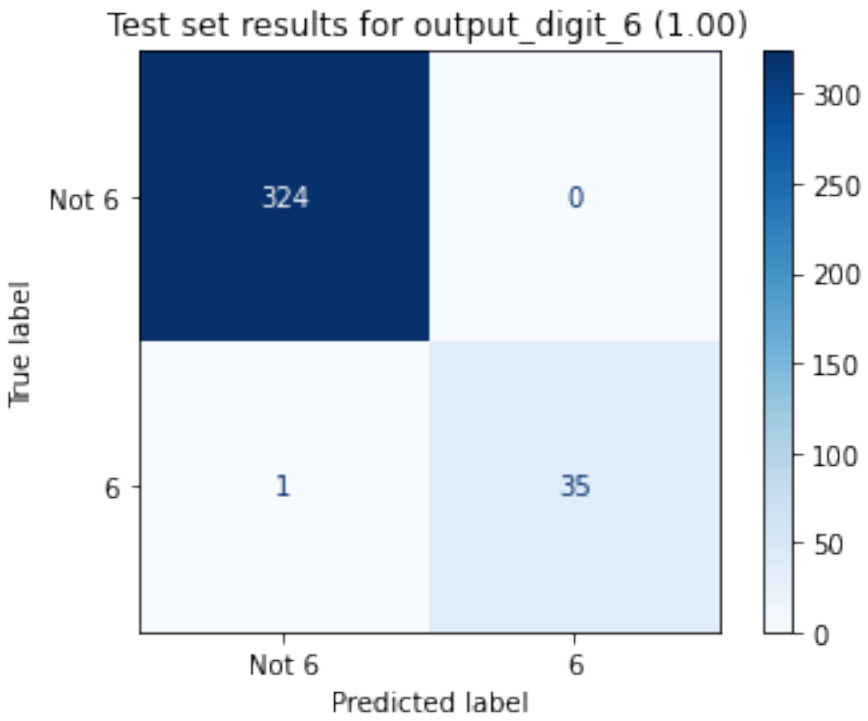

```
Accuracy of perceptron on training set: 0.98
Accuracy of perceptron on test set: 0.97


Test set results for output_digit_4 (0.97)
[[324   0]
 [ 11  25]]
```



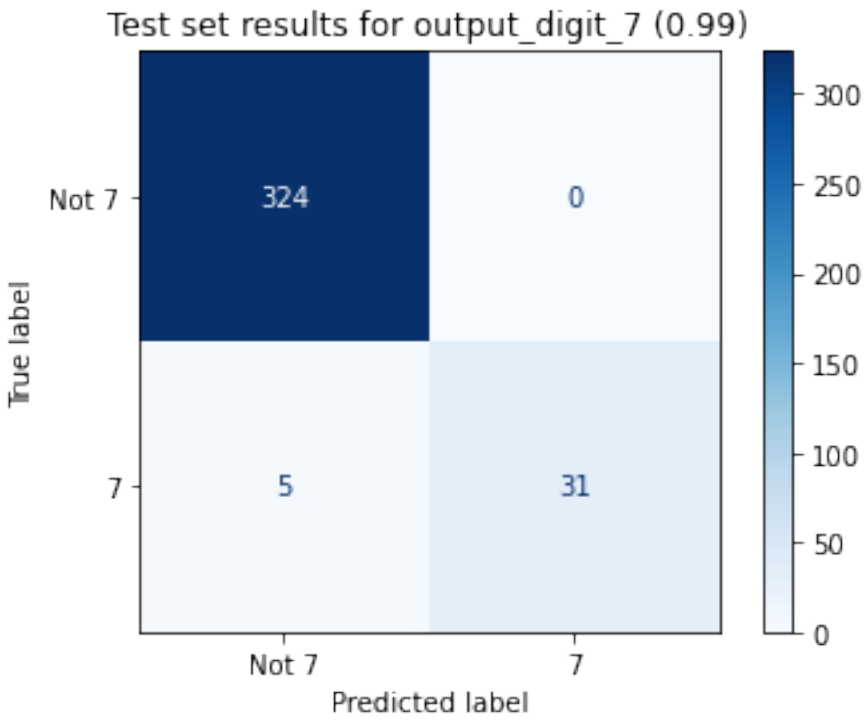

```
Accuracy of perceptron on training set: 0.99
Accuracy of perceptron on test set: 0.99


Test set results for output_digit_5 (0.99)
[[323   0]
 [  3  34]]
```



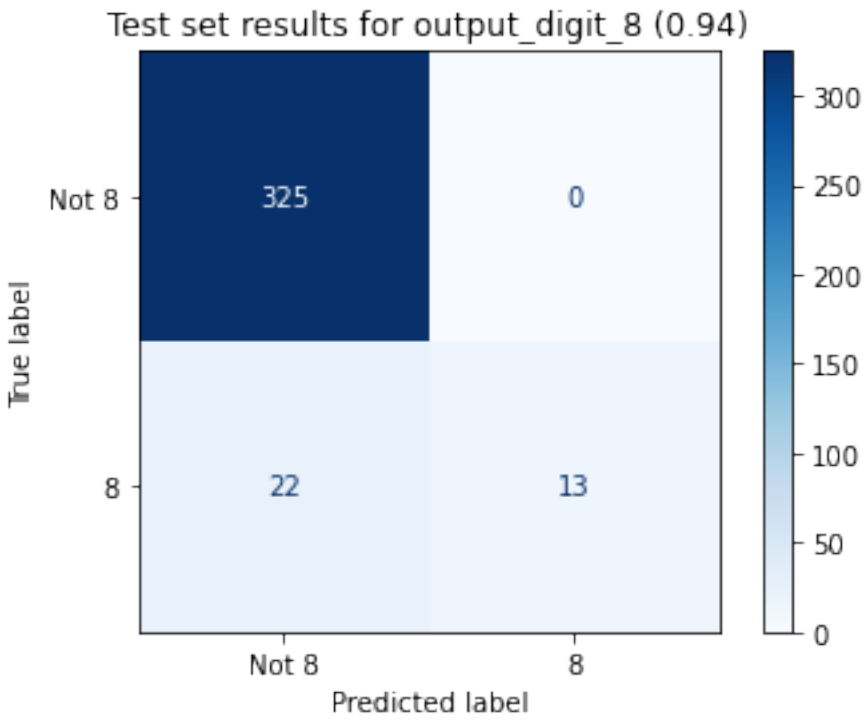

```
Accuracy of perceptron on training set: 0.99
Accuracy of perceptron on test set: 1.00


Test set results for output_digit_6 (1.00)
[[324   0]
 [  1  35]]
```



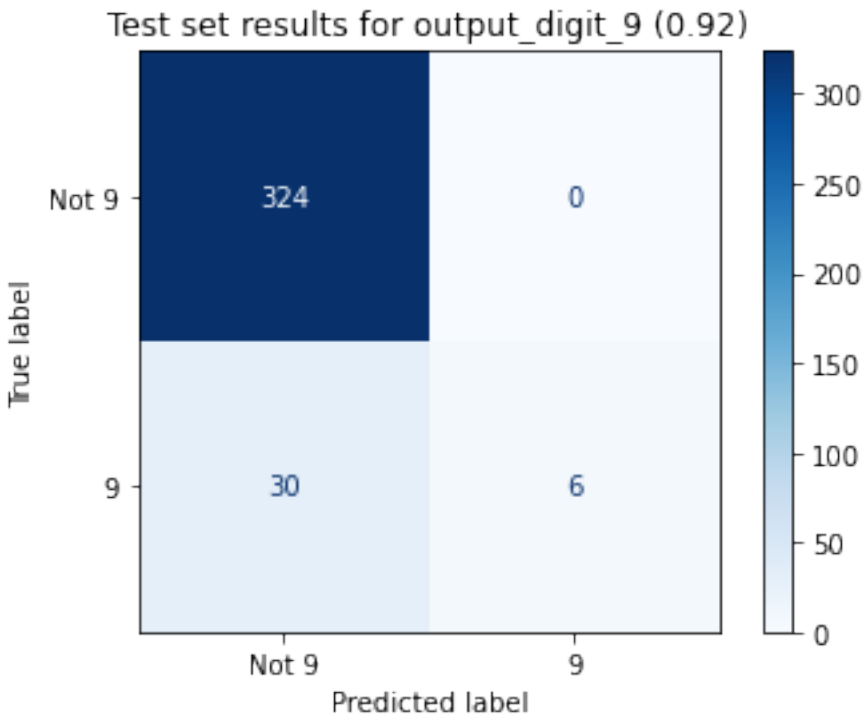

```
Accuracy of perceptron on training set: 0.98
Accuracy of perceptron on test set: 0.99


Test set results for output_digit_7 (0.99)
[[324   0]
 [  5  31]]
```




```
Accuracy of perceptron on training set: 0.95
Accuracy of perceptron on test set: 0.94


Test set results for output_digit_8 (0.94)
[[325   0]
 [ 22  13]]
```




```
Accuracy of perceptron on training set: 0.93
Accuracy of perceptron on test set: 0.92


Test set results for output_digit_9 (0.92)
[[324   0]
 [ 30   6]]
```


Test set results for output_digit_9 (0.92)
Not 9
9
True label
324
0
30
6
Not 9
9
Predicted label
300
250
200
150
100
50
0