Activity 19

1. $x_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $x_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $w = \begin{bmatrix} -1 \\ 0.5 \end{bmatrix}$

$d_1 = 1$, $d_2 = 1$ $\quad X = \begin{bmatrix} x_1^T \\ x_2^T \end{bmatrix}$

b) a) $\|Xw - y\|^2 \Rightarrow \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$= \begin{bmatrix} -0.5 \\ 1.5 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$

$\Rightarrow (0.5)^2 + (0.5)^2 = 0.5$

b) $\sum_i (1 - yXw)$

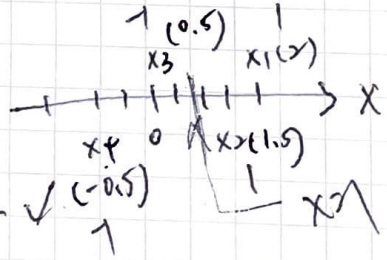
c) $= 1 - \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} \quad \end{bmatrix}$

$= 1 - (-1)\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 0.5 \end{bmatrix}$

$+ 1 - (1) \begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 0.5 \end{bmatrix}$

$= 1 - (+1)(+0.5) + 1 - (1.5)$

$= 1 - 0.5 + 1 - 1.5 = 1$

2. 

-5 ✓

2. a) 5

a) $1.5 - 0.5 = 1$

b) yes

x, c) $x = 1$, $\rightarrow x - 1 = 0$

$\begin{bmatrix} x & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ -\alpha \end{bmatrix} = 0$

w

no error

d) yes $\quad \Rightarrow \alpha x - \alpha = 0$

e) yes

$x = 1$

3. a) $1 \geq 13$ b) $\neq 9.5$

c) squared error

d) nothing happens, same: $1 \geq 13$

e) hinge loss, worse: ≥ 66

In [31]:
```
import numpy as np
from scipy.io import loadmat
import matplotlib.pyplot as plt
from sklearn.svm import LinearSVC
```

2b)

In [32]:
```
A = np.array([[2, 1.5, 0.5, -0.5]]).reshape(-1,1)
# print(A)

y = np.array([[1], [1], [-1], [-1]])
# print(y)

w = np.linalg.inv(A.T @ A) @ A.T @ y
print(w)

sqaured_error = np.linalg.norm(A@w-y) **(2)
print('sqaured error:', sqaured_error)
```

```
[[0.51851852]]
sqaured error: 2.185185185185185
```

In [33]:
```
A = np.array([[2, 1.5, 0.5, -5]]).reshape(-1,1)
# print(A)

y = np.array([[1], [1], [-1], [-1]])
# print(y)

w = np.linalg.inv(A.T @ A) @ A.T @ y
print(w)

sqaured_error = np.linalg.norm(A@w-y) **(2)
print('sqaured error:', sqaured_error)
```

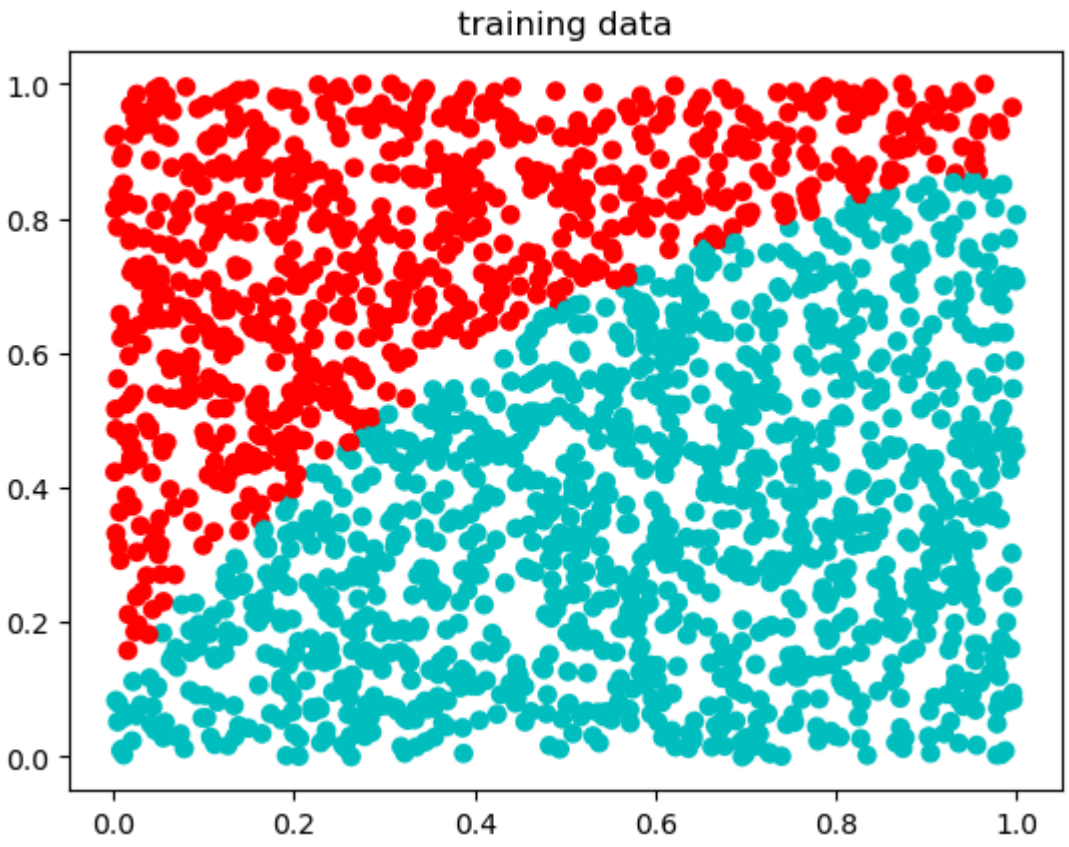
```
[[0.25396825]]
sqaured error: 1.9682539682539681
```

In [ ]:
```
```

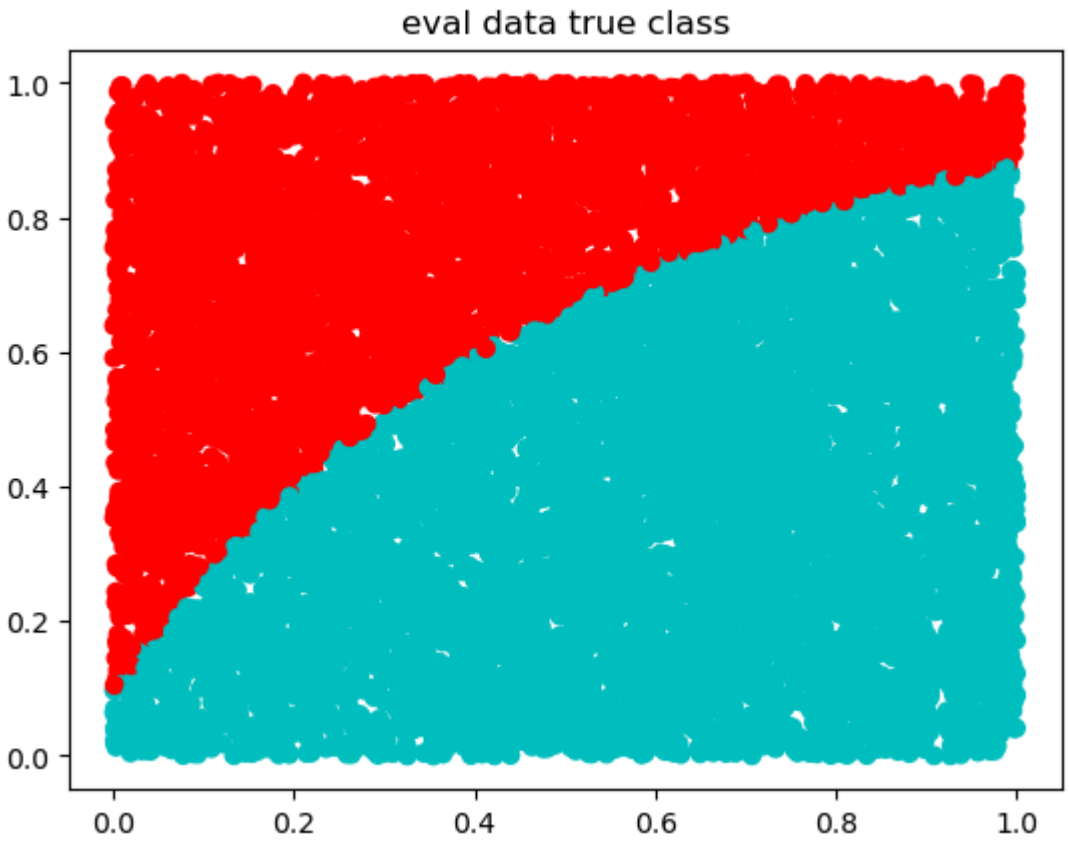
Problem 3

In [34]:
```
in_data = loadmat('classifier_data.mat')

x_train = in_data['x_train']
x_eval = in_data['x_eval']
y_train = in_data['y_train']
y_eval = in_data['y_eval']

n_eval = np.size(y_eval)
n_train = np.size(y_train)
print(n_train)


plt.scatter(x_train[:,0],x_train[:,1], color=['c' if i==-1 else 'r' for i in y_t
plt.title('training data')
plt.show()
```

```
2000
```

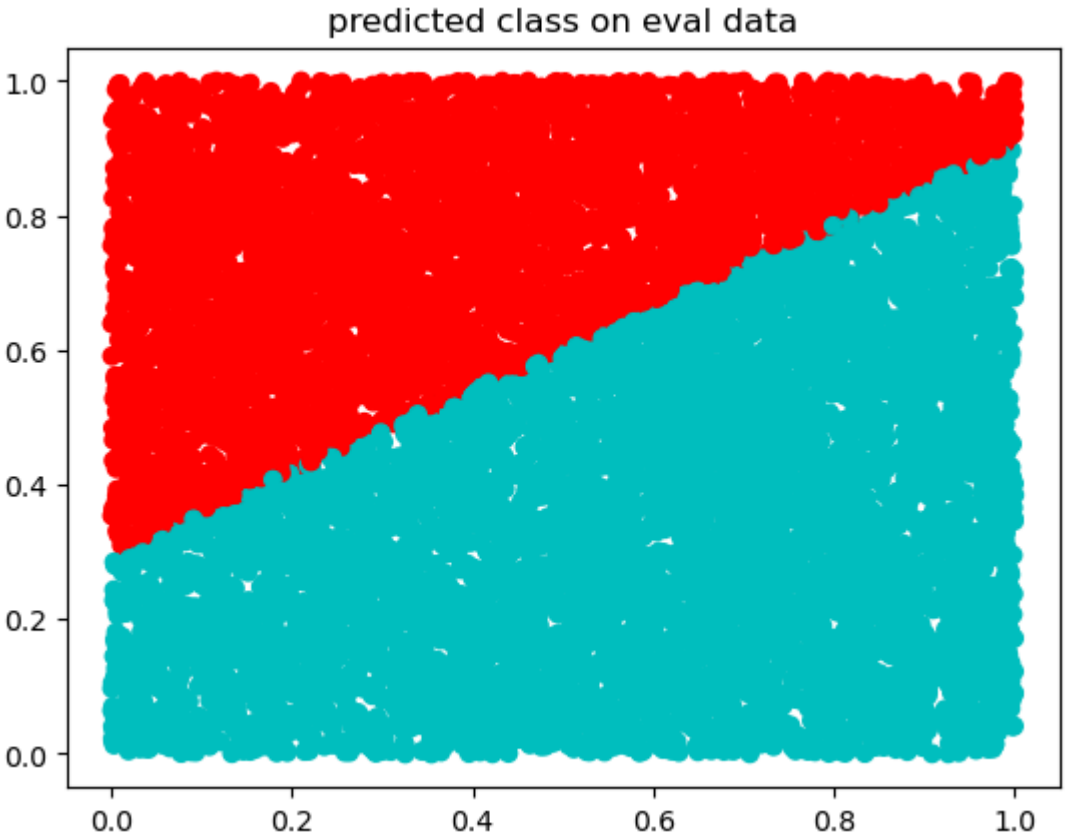


In [35]:
```
plt.scatter(x_eval[:,0],x_eval[:,1], color=['c' if i==-1 else 'r' for i in y_eva
plt.title('eval data true class')
plt.show()
```



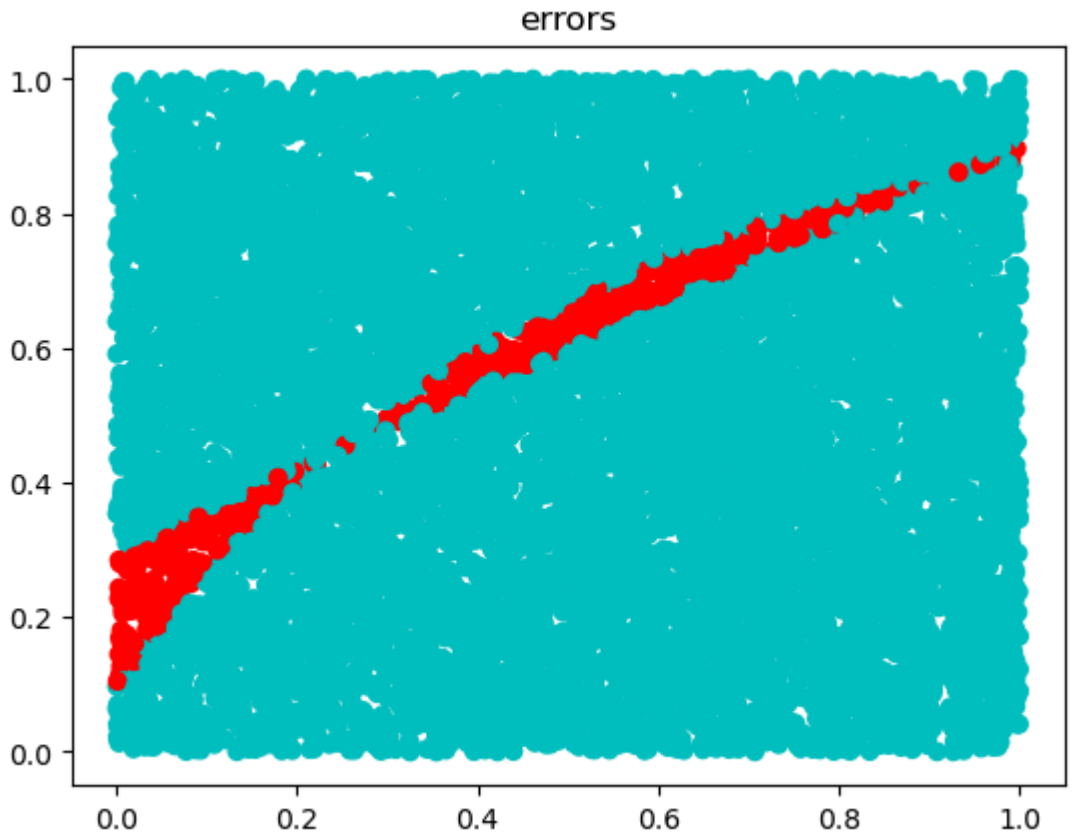
In [36]:
```
## Classifier 1
x_train_1 = np.hstack(( x_train, np.ones((n_train,1)) ))
```

```
x_eval_1 = np.hstack(( x_eval, np.ones((n_eval,1)) ))

# Train classifier using linear SVM from SK Learn library
# clf = LinearSVC(random_state=0, tol=1e-8)
# clf.fit(x_train_1, np.squeeze(y_train))
# w_opt = clf.coef_.transpose()

#uncomment this line to use least squares classifier
#w_opt = np.linalg.inv(x_train_1.T@x_train_1)@x_train_1.T@y_train

y_hat_outlier = np.sign(x_eval_1@w_opt)
plt.scatter(x_eval[:,0],x_eval[:,1], color=['c' if i==-1 else 'r' for i in y_hat
plt.title('predicted class on eval data')
plt.show()
```



In [37]:

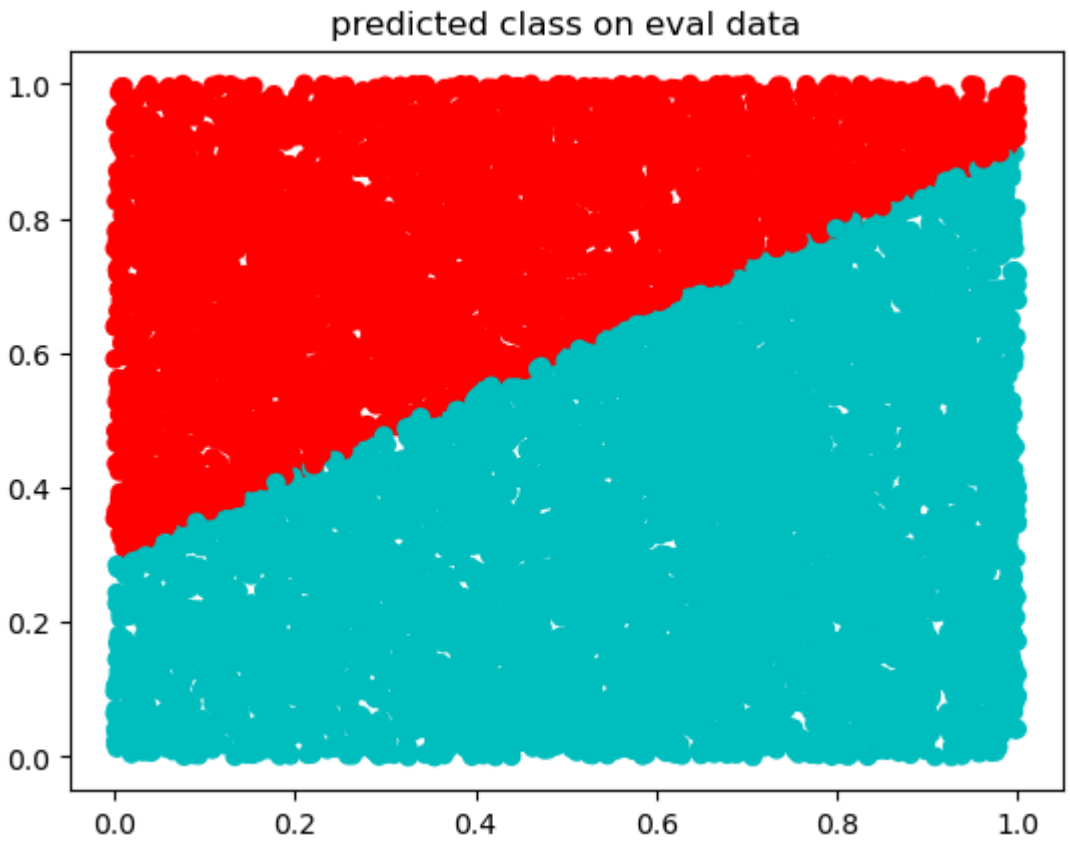
```
error_vec = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat_outlier, y_eval))]
plt.scatter(x_eval[:,0],x_eval[:,1], color=['c' if i==0 else 'r' for i in error_
plt.title('errors')
plt.show()

print('Errors: '+ str(sum(error_vec)))
```



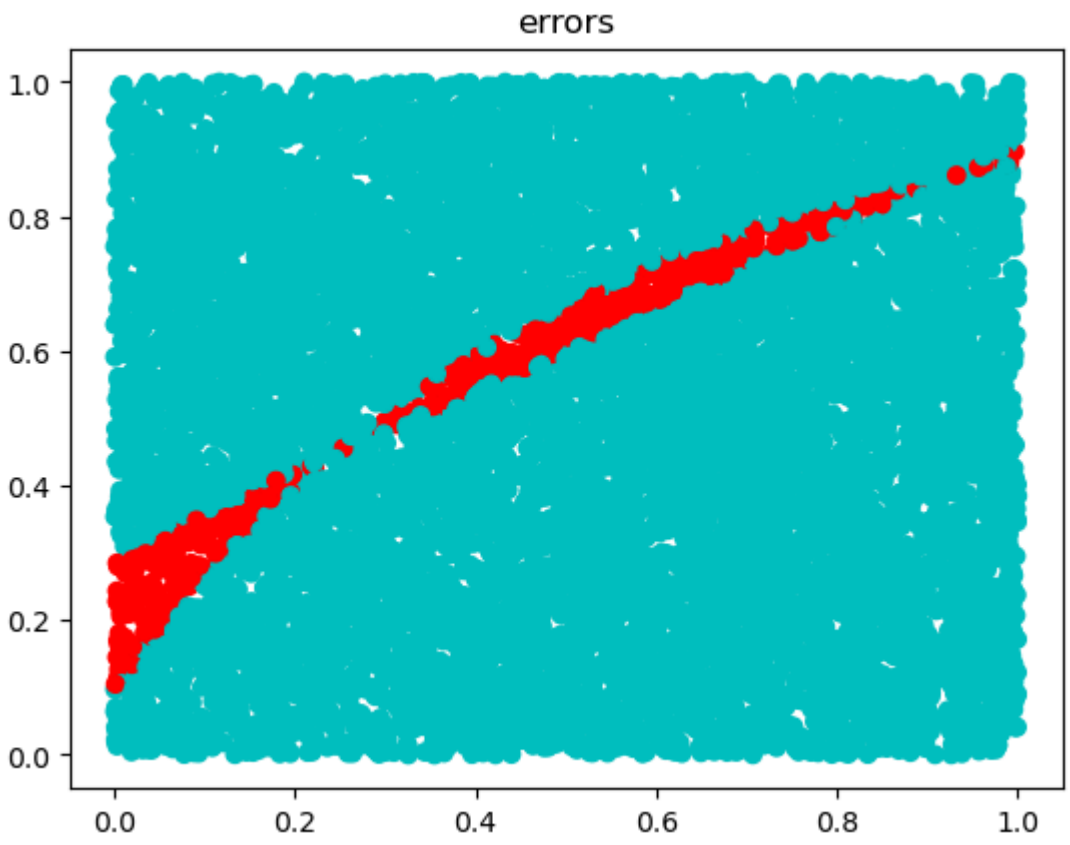

```
Errors: 495
```

In [38]:

```
## Classifier 2
x_train_1 = np.hstack(( x_train, np.ones((n_train,1)) ))
x_eval_1 = np.hstack(( x_eval, np.ones((n_eval,1)) ))

# Train classifier using linear SVM from SK Learn library
# clf = LinearSVC(random_state=0, tol=1e-8)
# clf.fit(x_train_1, np.squeeze(y_train))
# w_opt = clf.coef_.transpose()

#uncomment this line to use least squares classifier
w_opt = np.linalg.inv(x_train_1.T@x_train_1)@x_train_1.T@y_train

y_hat_outlier = np.sign(x_eval_1@w_opt)
plt.scatter(x_eval[:,0],x_eval[:,1], color=['c' if i==-1 else 'r' for i in y_hat
plt.title('predicted class on eval data')
plt.show()
```



In [39]:
```
error_vec = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat_outlier, y_eval))]
plt.scatter(x_eval[:,0],x_eval[:,1], color=['c' if i==0 else 'r' for i in error_
plt.title('errors')
plt.show()

print('Errors: '+ str(sum(error_vec)))
```



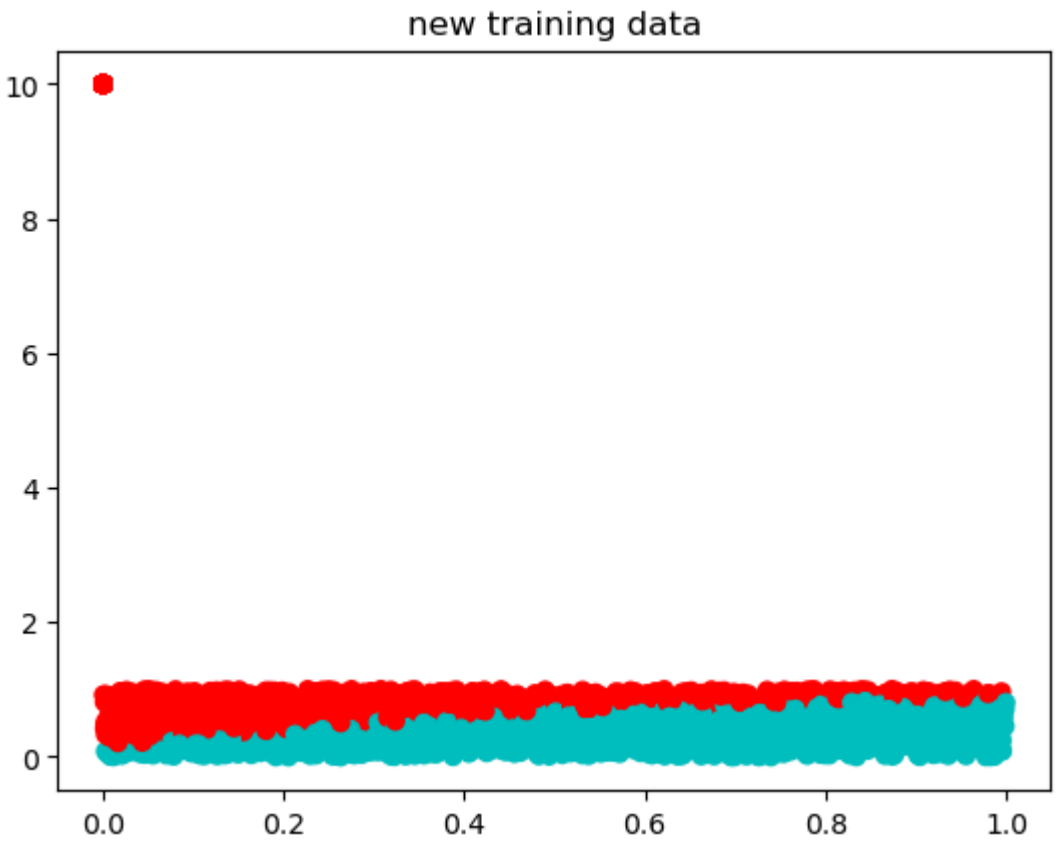

```
Errors: 495
```

In [ ]:

Add correct points far from boundary

In [40]:
```
## create new, correctly labeled points
n_new = 1000 #number of new datapoints
x_train_new = np.hstack((np.zeros((n_new,1)), 10*np.ones((n_new,1))))
y_train_new = np.ones((n_new,1))

## add these to the training data
x_train_outlier = np.vstack((x_train,x_train_new))
y_train_outlier = np.vstack((y_train,y_train_new))
n_outlier = np.size(y_train_outlier)
print(n_outlier)
plt.scatter(x_train_outlier[:,0],x_train_outlier[:,1], color=['c' if i==-1 else
plt.title('new training data')
plt.show()
```

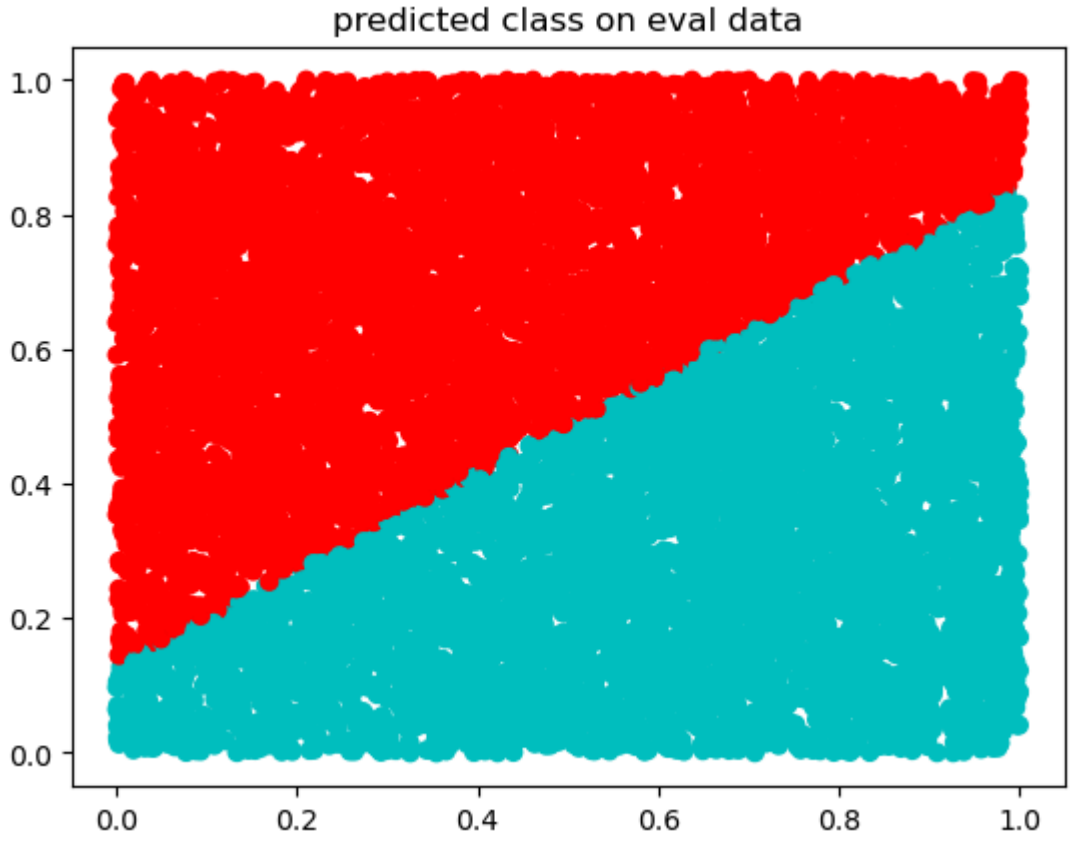
```
3000
```



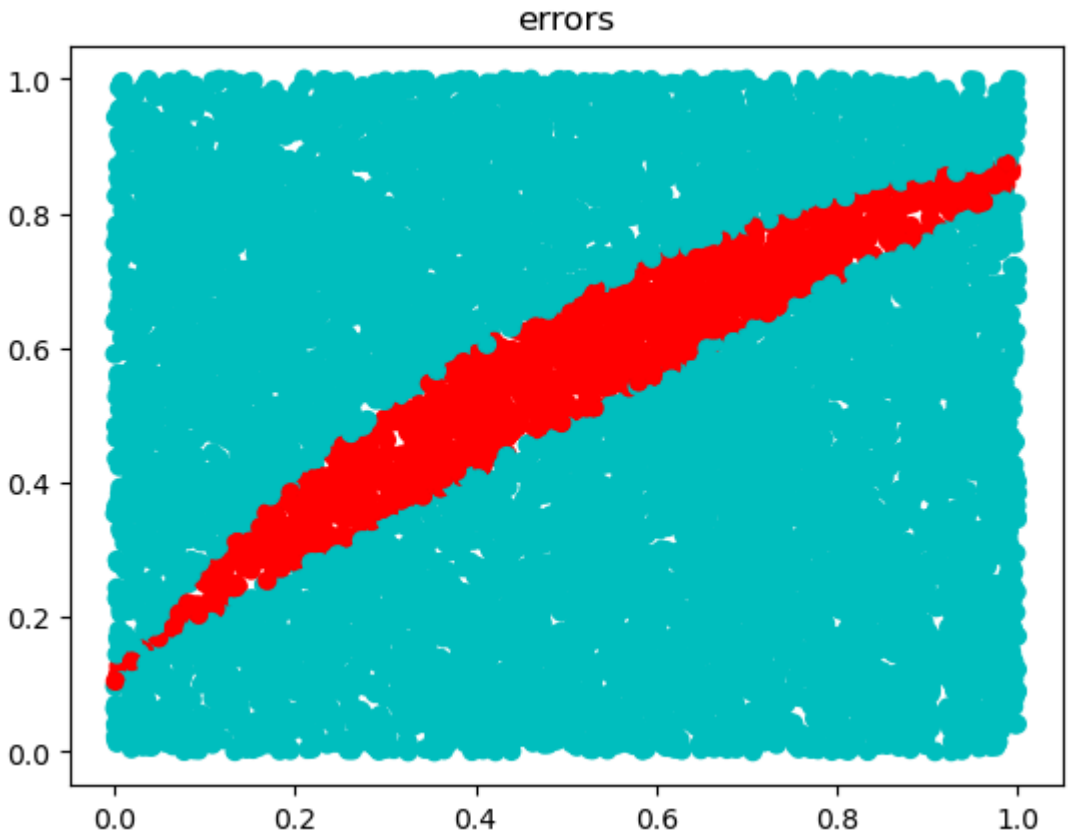
In [41]:
```
x_train_outlier_1 = np.hstack((x_train_outlier, np.ones((n_train+n_new,1)) ))
x_eval_1 = np.hstack((x_eval, np.ones((n_eval,1)) ))

#Train classifier using off the shelf SVM from sklearn
clf = LinearSVC(random_state=0, tol=1e-5)
clf.fit(x_train_outlier_1, np.squeeze(y_train_outlier))
w_opt_outlier = clf.coef_.transpose()

#uncomment this line to use least squares classifier
#w_opt_outlier = np.linalg.inv(x_train_outlier_1.T@x_train_outlier_1)@x_train_ou

y_hat_outlier = np.sign(x_eval_1@w_opt_outlier)
```

```
plt.scatter(x_eval[:,0],x_eval[:,1], color=['c' if i==-1 else 'r' for i in y_hat
plt.title('predicted class on eval data')
plt.show()
```

```
C:\Users\ftstc\anaconda3\envs\lis\lib\site-packages\sklearn\svm\_classes.py:32: F
utureWarning: The default value of `dual` will change from `True` to ``'auto'`` in
1.5. Set the value of `dual` explicitly to suppress the warning.
  warnings.warn(
```



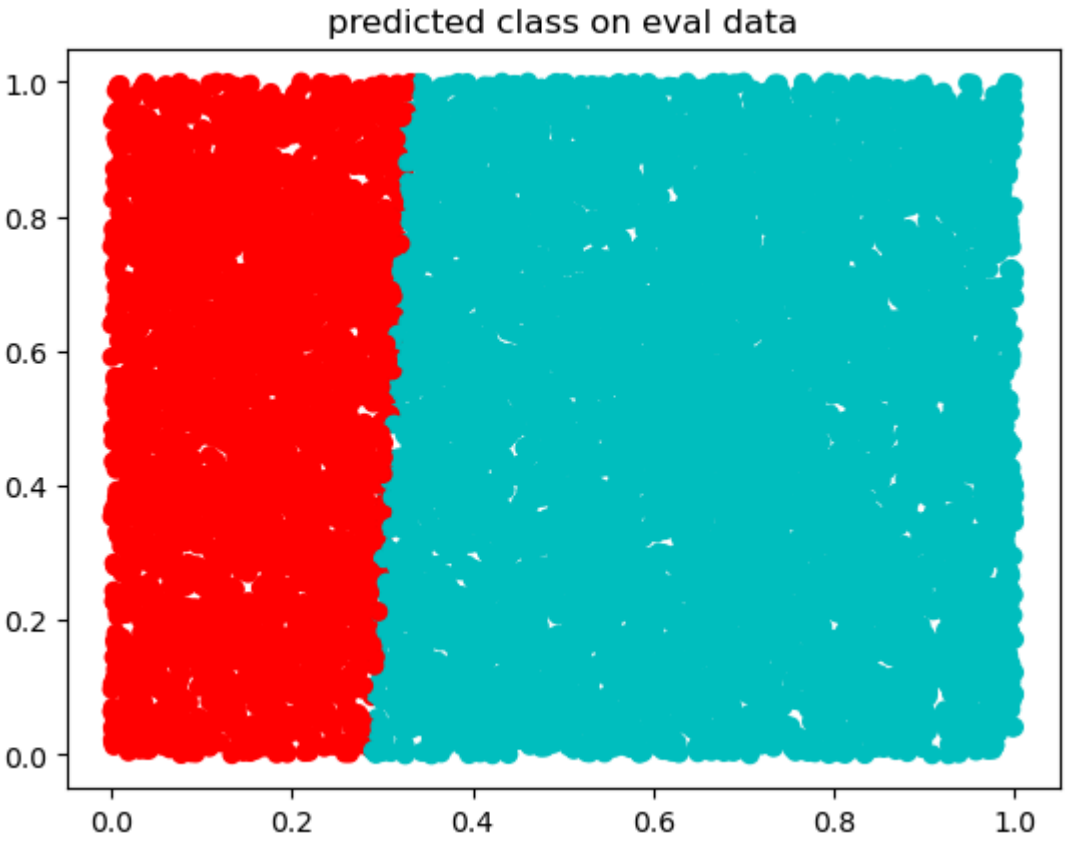
In [42]:
```
error_vec = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat_outlier, y_eval))]
plt.scatter(x_eval[:,0],x_eval[:,1], color=['c' if i==0 else 'r' for i in error_
plt.title('errors')
plt.show()

print('Errors: '+ str(sum(error_vec)))
print('Errors: '+ str(sum(error_vec)*100/n_outlier) + '%')
```



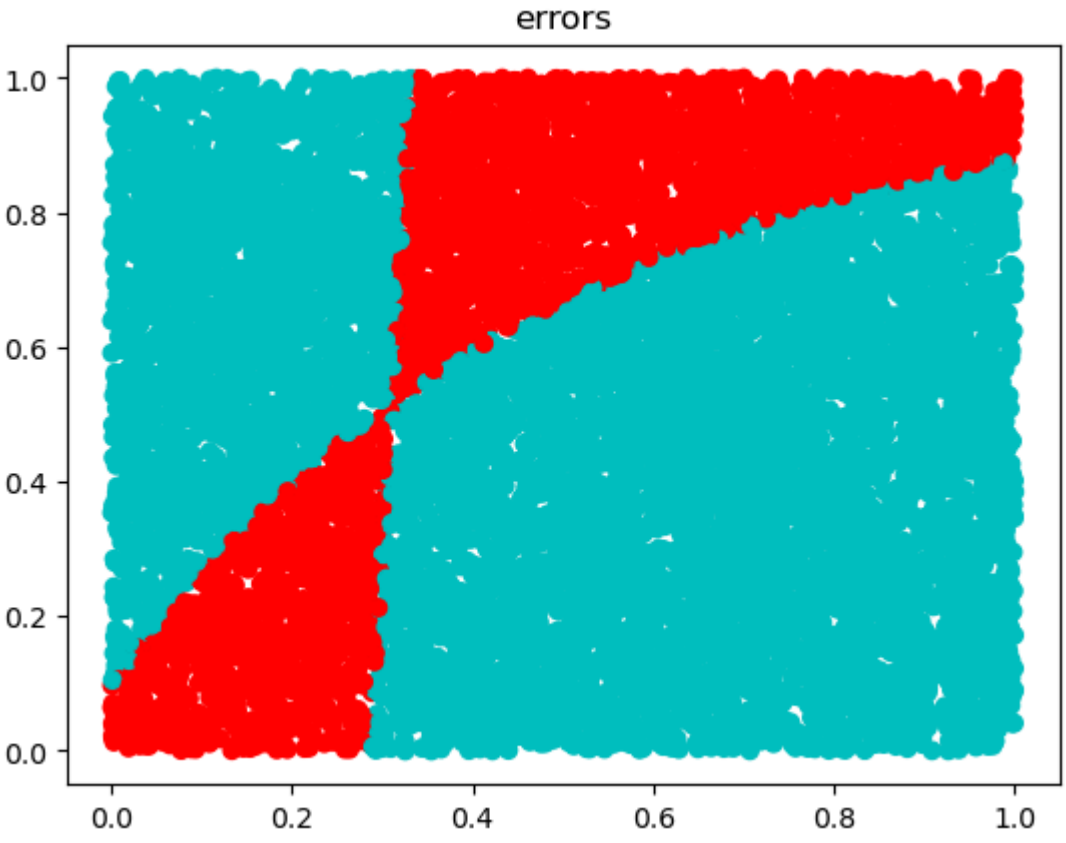

```
Errors: 1213
Errors: 40.43333333333333%
```

In [43]:

```
x_train_outlier_1 = np.hstack((x_train_outlier, np.ones((n_train+n_new,1)) ))
x_eval_1 = np.hstack((x_eval, np.ones((n_eval,1)) ))

#Train classifier using off the shelf SVM from sklearn
# clf = LinearSVC(random_state=0, tol=1e-5)
# clf.fit(x_train_outlier_1, np.squeeze(y_train_outlier))
# w_opt_outlier = clf.coef_.transpose()

#uncomment this line to use least squares classifier
w_opt_outlier = np.linalg.inv(x_train_outlier_1.T@x_train_outlier_1)@x_train_out

y_hat_outlier = np.sign(x_eval_1@w_opt_outlier)
plt.scatter(x_eval[:,0],x_eval[:,1], color=['c' if i==-1 else 'r' for i in y_hat
plt.title('predicted class on eval data')
plt.show()
```



In [44]:
```
error_vec = [0 if i[0]==i[1] else 1 for i in np.hstack((y_hat_outlier, y_eval))]
plt.scatter(x_eval[:,0],x_eval[:,1], color=['c' if i==0 else 'r' for i in error_
plt.title('errors')
plt.show()

print('Errors: '+ str(sum(error_vec)))
print('Errors: '+ str(sum(error_vec)*100/n_outlier) + '%')
```



```
Errors: 2668
Errors: 88.93333333333334%
```

In [ ]: