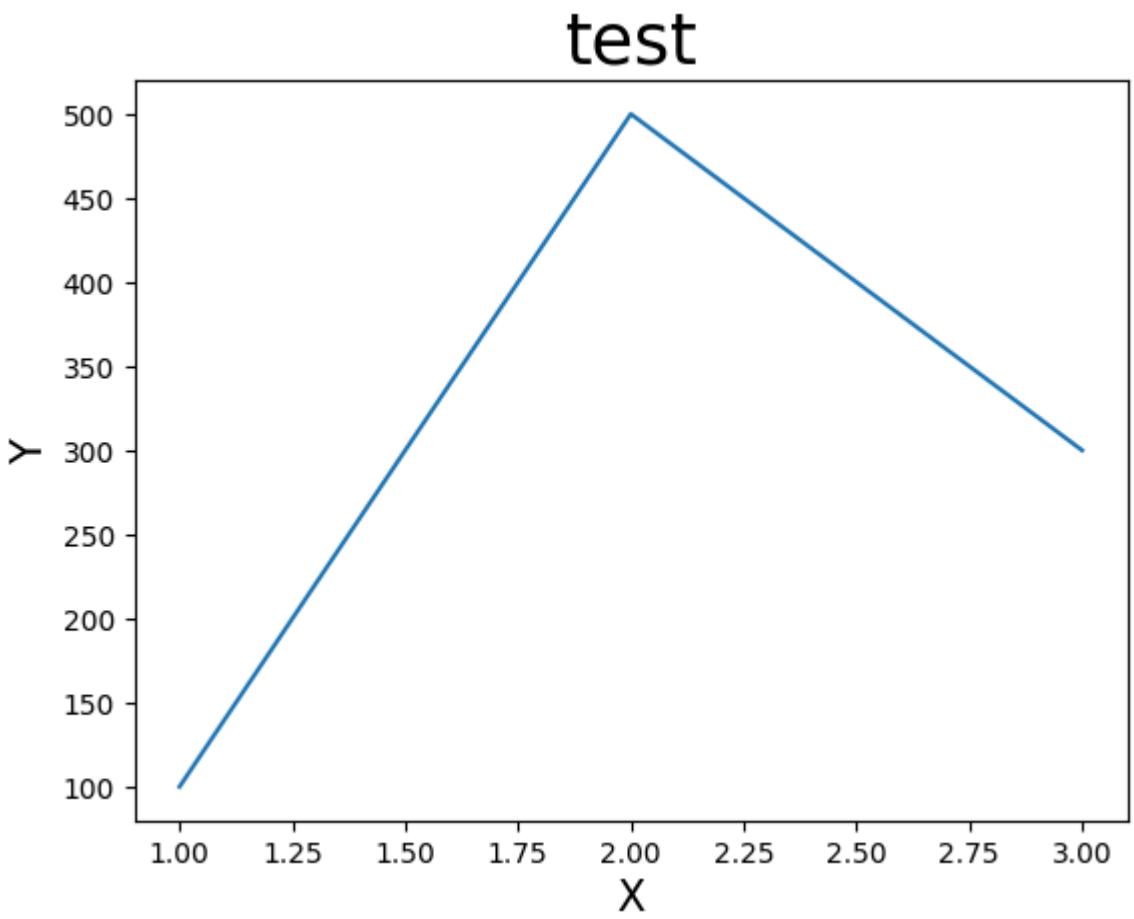
In [1]:
```
import pandas as pd

import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
plt.plot([1,2,3], [100,500,300])
plt.title('test', fontsize=25)
plt.xlabel('X', fontsize=15)
plt.ylabel('Y', fontsize=15)
plt.show()
```



Loading the training log form

In [3]:
```
df_train = pd.read_csv('/home/featurize/TrainginglogTrainingSets.csv')
df_test = pd.read_csv('/home/featurize/TrainingLogTestSet.csv')
```

In [4]:
```
df_train
```

Out[4]:

	epoch	batch	train_loss	train_accuracy
0	0	0	3.393817	0.03125
1	1	3	3.609225	0.09375
2	1	4	3.039786	0.18750
3	1	5	3.378135	0.03125
4	1	6	2.907747	0.21875
...	...	...	...	...
6897	30	6899	0.748141	0.81250
6898	30	6900	1.194558	0.59375
6899	30	6901	0.862398	0.75000
6900	30	6902	1.523913	0.50000
6901	30	6903	3.805648	0.00000

6902 rows × 4 columns

In [5]:
```
df_test
```

Out[5]:

	epoch	test_loss	test_accuracy	test_precision	test_recall	test_f1-score
0	0.0	3.697069	0.046771	0.022265	0.046261	0.019710
1	1.0	2.602122	0.280624	0.296360	0.287200	0.225205
2	2.0	2.543696	0.219376	0.197653	0.219646	0.169717
3	3.0	2.088492	0.366370	0.399844	0.369838	0.329300
4	4.0	2.002362	0.367483	0.390817	0.367168	0.337859
...	...	...	...	...	...	...
56	26.0	1.271067	0.636971	0.662936	0.637188	0.634950
57	27.0	1.117807	0.664811	0.676644	0.664701	0.663364
58	28.0	1.108036	0.662584	0.667760	0.660785	0.658596
59	29.0	1.164896	0.660356	0.676255	0.660759	0.658173
60	30.0	1.127610	0.661470	0.669976	0.658636	0.657960

61 rows × 6 columns

Training set loss function

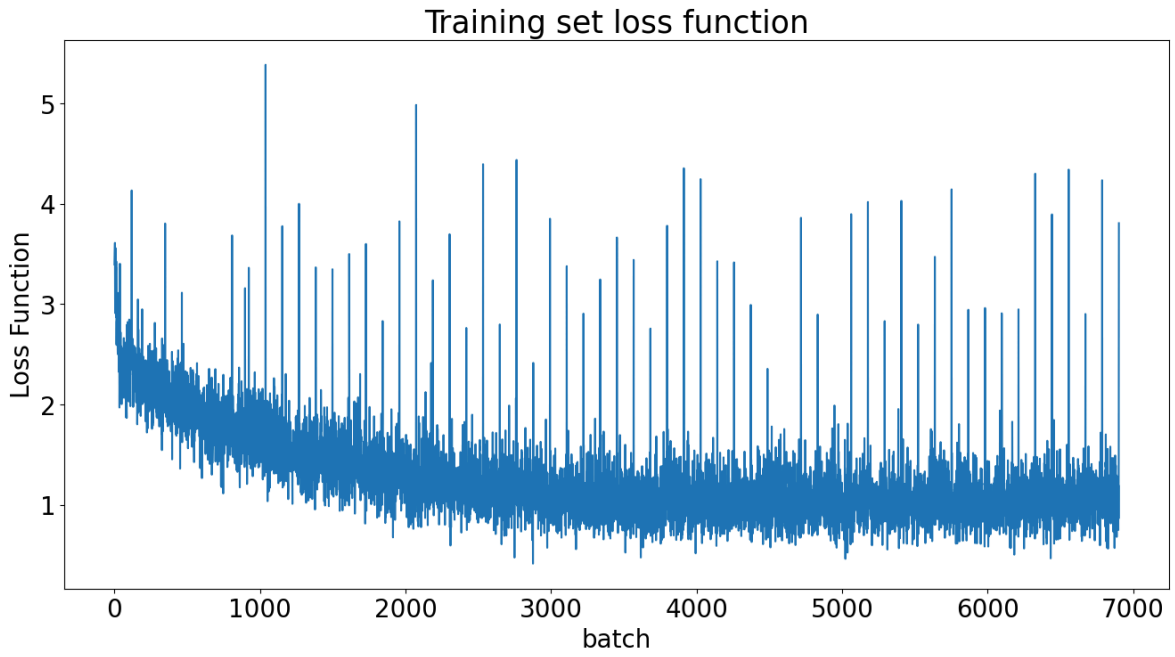
In [10]:
```
plt.figure(figsize=(16, 8))

x = df_train['batch']
y = df_train['train_loss']

plt.plot(x, y, label='traindataset')
```

```
plt.tick_params(labelsize=20)
plt.xlabel('batch', fontsize=20)
plt.ylabel('Loss Function', fontsize=20)
plt.title('Training set loss function', fontsize=25)
#plt.savefig('Graphs/Training Set Loss Function.pdf', dpi=120, bbox_inches='tigh

plt.show()
```



Training set accuracy

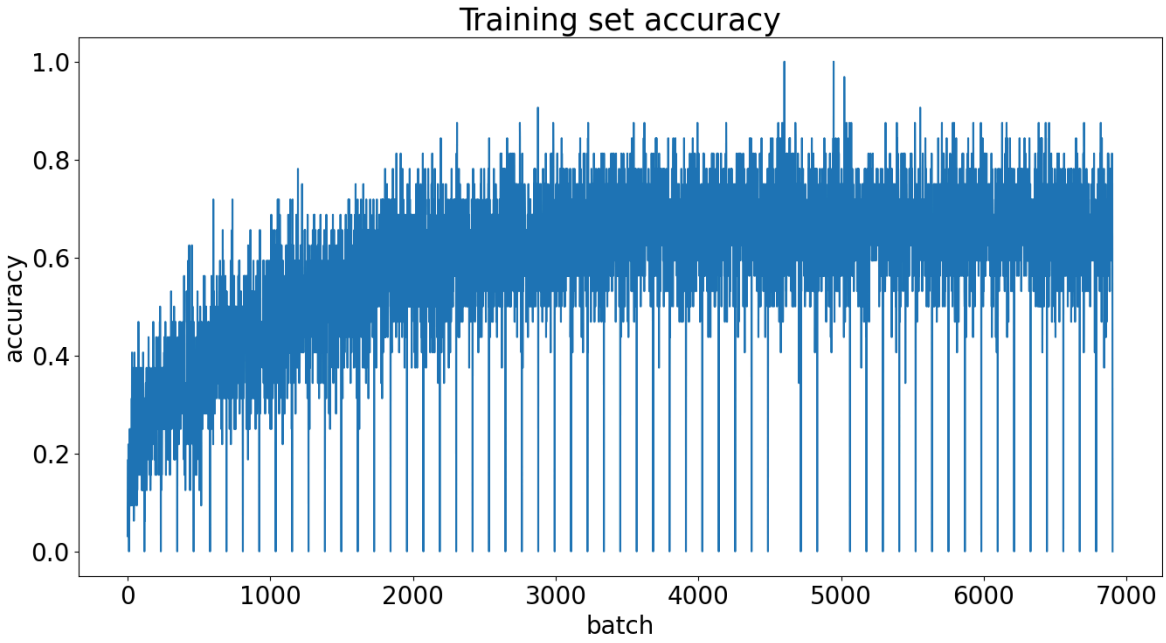
In [9]:
```
plt.figure(figsize=(16, 8))

x = df_train['batch']
y = df_train['train_accuracy']

plt.plot(x, y, label='traindataset')

plt.tick_params(labelsize=20)
plt.xlabel('batch', fontsize=20)
plt.ylabel('accuracy', fontsize=20)
plt.title('Training set accuracy', fontsize=25)
#plt.savefig('Graphs/Training Set accuracy.pdf', dpi=120, bbox_inches='tight')


plt.show()
```



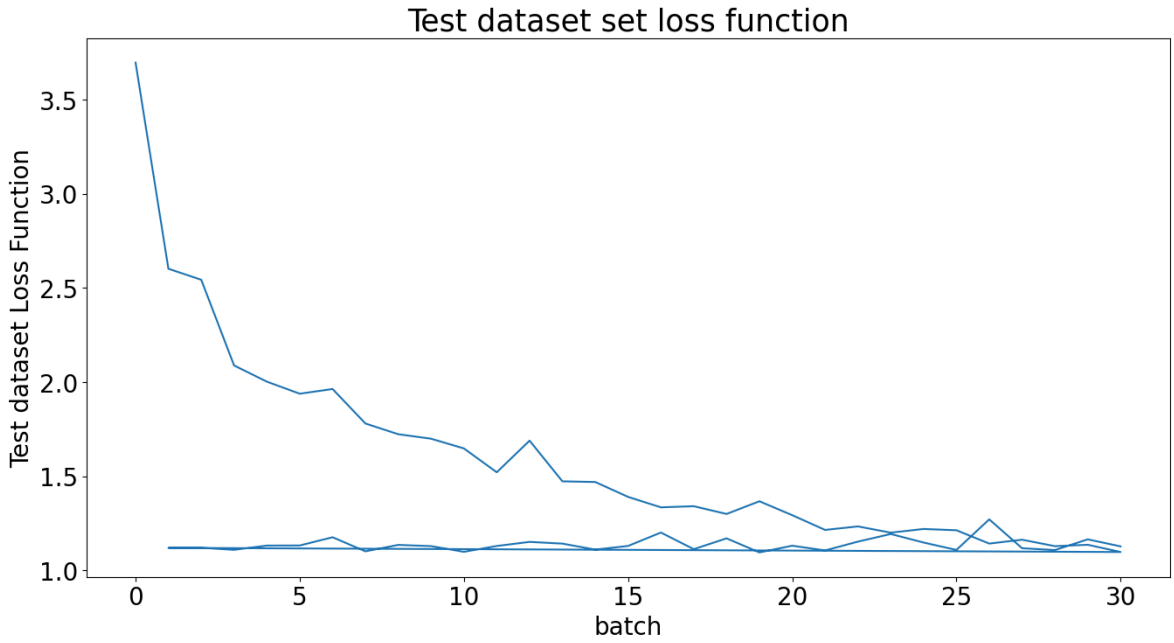
Test set loss function

In [12]:
```
plt.figure(figsize=(16, 8))

x = df_test['epoch']
y = df_test['test_loss']

plt.plot(x, y, label='testdataset')

plt.tick_params(labelsize=20)
plt.xlabel('epoch', fontsize=20)
plt.xlabel('batch', fontsize=20)
plt.ylabel('Test dataset Loss Function', fontsize=20)
plt.title('Test dataset set loss function', fontsize=25)
plt.show()
```



Test set evaluation metrics

In [13]:
```
from matplotlib import colors as mcolors
import random
random.seed(124)
colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k', 'tab:blue', 'tab:orange', 'tab:gree
markers = [".",",","o","v","^","<",">","1","2","3","4","8","s","p","P","*","h","
linestyle = ['--', '-.', '-']
def get_line_arg():
    '''
    Randomly generates a drawing line pattern
    '''
    line_arg = {}
    line_arg['color'] = random.choice(colors)
    # line_arg['marker'] = random.choice(markers)
    line_arg['linestyle'] = random.choice(linestyle)
    line_arg['linewidth'] = random.randint(1, 4)
    # line_arg['markersize'] = random.randint(3, 5)
    return line_arg
```

In [14]:
```
metrics = ['test_accuracy', 'test_precision', 'test_recall', 'test_f1-score']
```

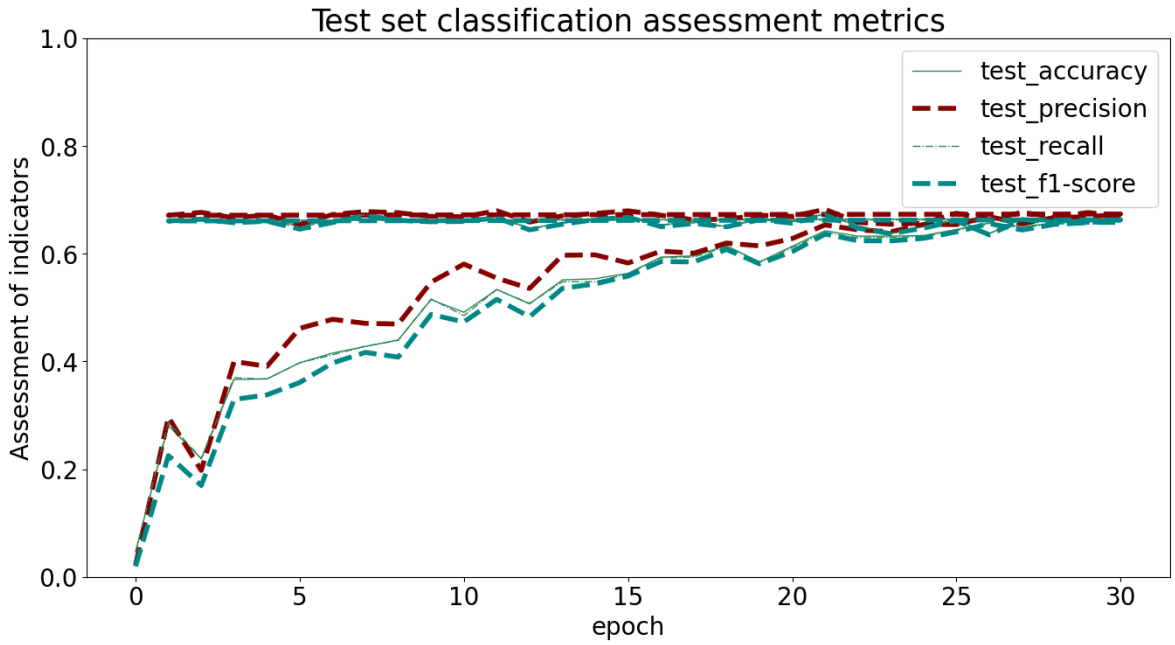
In [15]:
```
plt.figure(figsize=(16, 8))

x = df_test['epoch']
for y in metrics:
    plt.plot(x, df_test[y], label=y, **get_line_arg())

plt.tick_params(labelsize=20)
plt.ylim([0, 1])
plt.xlabel('epoch', fontsize=20)
plt.ylabel('Assessment of indicators', fontsize=20)
plt.title('Test set classification assessment metrics', fontsize=25)
#plt.savefig('1.pdf', dpi=120, bbox_inches='tight')

plt.legend(fontsize=20)

plt.show()
```



In [ ]: