In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import model_selection
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn import preprocessing
from yellowbrick.classifier import ConfusionMatrix
```

In [2]:
```
dataset = pd.read_csv("letter-recognition.data", sep = ",")
```

In [3]:
```
names = ['Class',
         'x-box',
         'y-box',
         'width',
         'high',
         'onpix',
         'x-bar',
         'y-bar',
         'x2bar',
         'y2bar',
         'xybar',
         'x2ybr',
         'xy2br',
         'x-ege',
         'xegvy',
         'y-ege',
         'yegvx']
```

In [4]:
```
X = dataset.iloc[:, 1 : 17]
Y = dataset.select_dtypes(include = [object])
```

In [5]:
```
X_train, X_validation, Y_train, Y_validation = model_selection.train_test_split(X,
```

In [6]:
```
scaler = StandardScaler()
scaler.fit(X_train)
```

Out[6]:
```
StandardScaler()
```

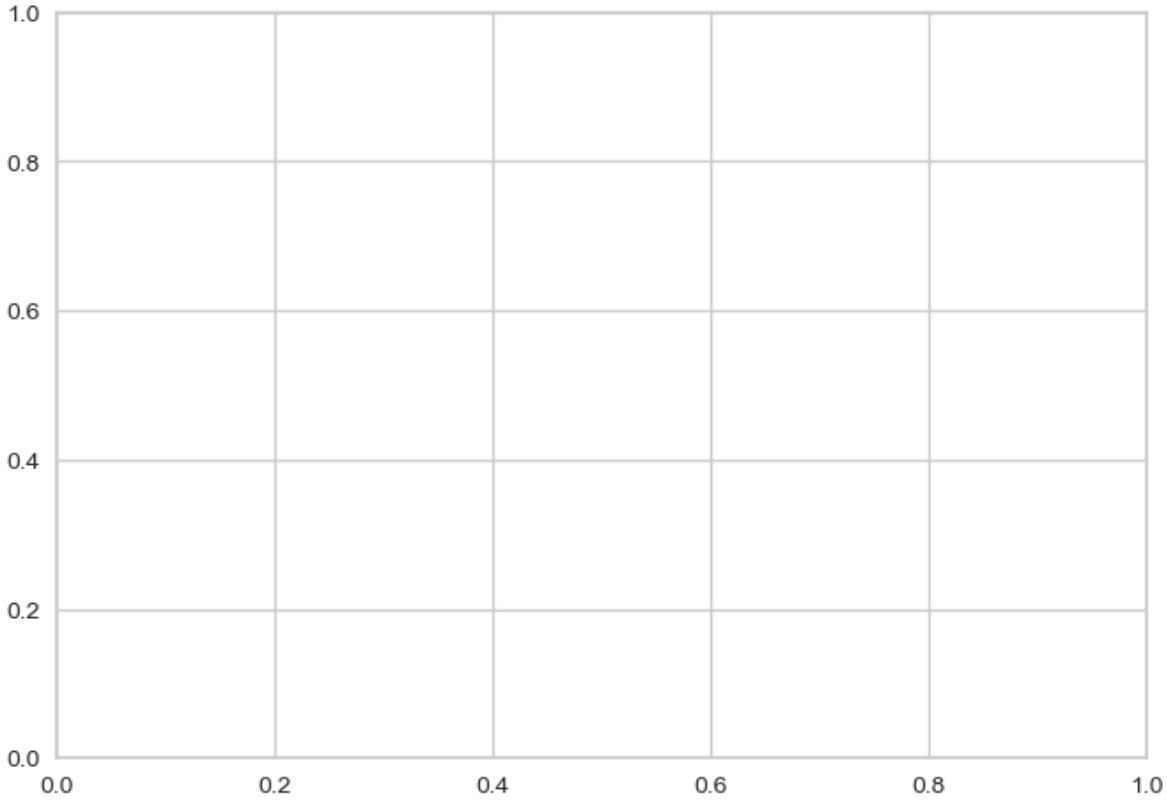
In [7]:
```
X_train = scaler.transform(X_train)
X_validation = scaler.transform(X_validation)
```

In [8]:
```
mlp = MLPClassifier(hidden_layer_sizes = (250, 300), max_iter = 1000000, activatio
```

In [9]:
```
cm = ConfusionMatrix(mlp, classes="A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,
```

In [10]:
```
cm.fit(X_train, Y_train.values.ravel())
```

Out[10]:
```
ConfusionMatrix(ax=<AxesSubplot:>,
                classes=['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K',
                         'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V',
                         'W', 'X', 'Y', 'Z'],
                cmap=<matplotlib.colors.ListedColormap object at 0x00000207A6D93C4
0>,
                estimator=MLPClassifier(activation='logistic',
                                        hidden_layer_sizes=(250, 300),
                                        max_iter=1000000))
```



In [11]:
```
cm.score(X_validation, Y_validation)
```

```
D:\CodingSetup\Anaconda\lib\site-packages\sklearn\preprocessing\_label.py:115: Dat
aConversionWarning: A column-vector y was passed when a 1d array was expected. Ple
ase change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[11]:
```
0.975
```

In [12]:
```
predictions = cm.predict(X_validation)
```

In [13]:
```
print("Accuracy: ", accuracy_score(Y_validation, predictions))
```

```
Accuracy:  0.975
```

In [14]:
```
print(confusion_matrix(Y_validation, predictions))
```

```
[[169   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0
    0   0   0   0   0   0   0   0]
 [  0 138   0   0   2   0   0   1   1   0   0   0   0   0   0   0   0   2
    0   0   0   0   0   0   0   0]
 [  0   0 158   0   1   0   0   0   0   0   0   1   0   0   1   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   1   0 170   0   0   0   1   0   0   0   0   0   2   0   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   2   0 140   0   0   0   0   0   0   2   0   0   0   0   0   0
    1   0   0   0   0   0   0   3]
 [  0   0   0   0   0 134   0   0   1   0   0   0   0   0   0   1   0   0
    0   1   0   0   0   0   0   0]
 [  0   2   0   0   2   0 134   0   0   0   0   1   1   0   1   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0 129   0   0   2   0   0   2   0   0   1   2
    0   0   1   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0 142   5   0   0   0   0   0   0   0   0
    0   1   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   3 137   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   2   0   0 133   1   0   0   0   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   2   0   0   0   0 162   0   0   0   0   0   0
    1   0   0   0   0   2   0   0]
 [  0   0   0   0   0   0   1   0   0   0   0   0 165   0   0   0   0   0
    0   0   1   0   1   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   1   0   0 145   2   0   0   1
    0   0   0   0   0   0   0   0]
 [  0   0   0   2   0   0   0   0   0   0   0   0   0   0 150   0   1   0
    1   0   0   0   0   0   0   0]
 [  1   0   0   0   0   3   0   0   0   0   0   0   0   0   0 152   1   0
    0   0   0   1   0   0   1   0]
 [  0   0   0   0   0   0   0   0   0   0   0   1   0   0   1   1 139   0
    0   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   4   0   0   0   0   0   1   0   0   0 155
    0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0
  144   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   1   0   0   0   0   0   0   0   0   0   0
    0 157   0   0   0   0   0   1]
 [  0   0   1   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0
    0   0 159   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0
    0   0   0 158   2   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   1   0   2   0   0   0   0   0
    0   0   0   0 155   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0 163   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   1   0   0   0   0 166   0]
 [  0   0   0   0   2   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0 146]]
```

In [15]:
```
print(classification_report(Y_validation, predictions, digits=5))
```

	precision	recall	f1-score	support
A	0.99412	0.99412	0.99412	170
B	0.97872	0.95833	0.96842	144
C	0.98137	0.98137	0.98137	161
D	0.97143	0.97701	0.97421	174
E	0.95238	0.94595	0.94915	148
F	0.97810	0.97810	0.97810	137
G	0.97810	0.95035	0.96403	141
H	0.93478	0.93478	0.93478	138
I	0.96599	0.95946	0.96271	148
J	0.96479	0.97857	0.97163	140
K	0.97080	0.97794	0.97436	136
L	0.96429	0.97006	0.96716	167
M	0.98214	0.98214	0.98214	168
N	0.96026	0.97315	0.96667	149
O	0.96774	0.97403	0.97087	154
P	0.98065	0.95597	0.96815	159
Q	0.96528	0.97887	0.97203	142
R	0.96875	0.96273	0.96573	161
S	0.97959	0.99310	0.98630	145
T	0.98125	0.98125	0.98125	160
U	0.98758	0.98758	0.98758	161
V	0.99371	0.98137	0.98750	161
W	0.98101	0.98101	0.98101	158
X	0.98788	1.00000	0.99390	163
Y	0.99401	0.99401	0.99401	167
Z	0.97333	0.98649	0.97987	148
accuracy			0.97500	4000
macro avg	0.97454	0.97453	0.97450	4000
weighted avg	0.97503	0.97500	0.97498	4000

In [16]:
```
cm.poof()
```

Out[16]:
```
<Figure size 800x550 with 0 Axes>
<AxesSubplot:title={'center':'MLPClassifier Confusion Matrix'}, xlabel='Predicted
Class', ylabel='True Class'>
```

In [ ]:
```

```