visualisation

September 19, 2024

1 Lab 1

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

1.1 Attribute Information

1. lettr capital letter (26 values from A to Z)
2. x-box horizontal position of box (integer)
3. y-box vertical position of box (integer)
4. width width of box (integer)
5. high height of box (integer)
6. onpix total # on pixels (integer)
7. x-bar mean x of on pixels in box (integer)
8. y-bar mean y of on pixels in box (integer)
9. x2bar mean x variance (integer)
10. y2bar mean y variance (integer)
11. xybar mean x y correlation (integer)
12. x2ybr mean of x * x * y (integer)
13. xy2br mean of x * y * y (integer)
14. x-ege mean edge count left to right (integer)
15. xegvy correlation of x-ege with y (integer)
16. y-ege mean edge count bottom to top (integer)
17. yegvx correlation of y-ege with x (integer)

1.2 Load the data

[2]:
```
df = pd.read_csv('letter+recognition/letter-recognition.data')
```

[3]:
```
df.head()
```

[3]:

	lettr	x-box	y-box	width	high	onpix	x-bar	y-bar	x2bar	y2bar	xybar	\
0	T	2	8	3	5	1	8	13	0	6	6	
1	I	5	12	3	7	2	10	5	5	4	13	
2	D	4	11	6	8	6	10	6	2	6	10	
3	N	7	11	6	6	3	5	9	4	6	4	

```
4     G      2      1      3      1      1      8      6      6      6      6

   x2ybr  xy2br  x-ege  xegvy  y-ege  yegvx
0     10      8      0      8      0      8
1      3      9      2      8      4     10
2      3      7      3      7      3      9
3      4     10      6     10      2      8
4      5      9      1      7      5     10
```

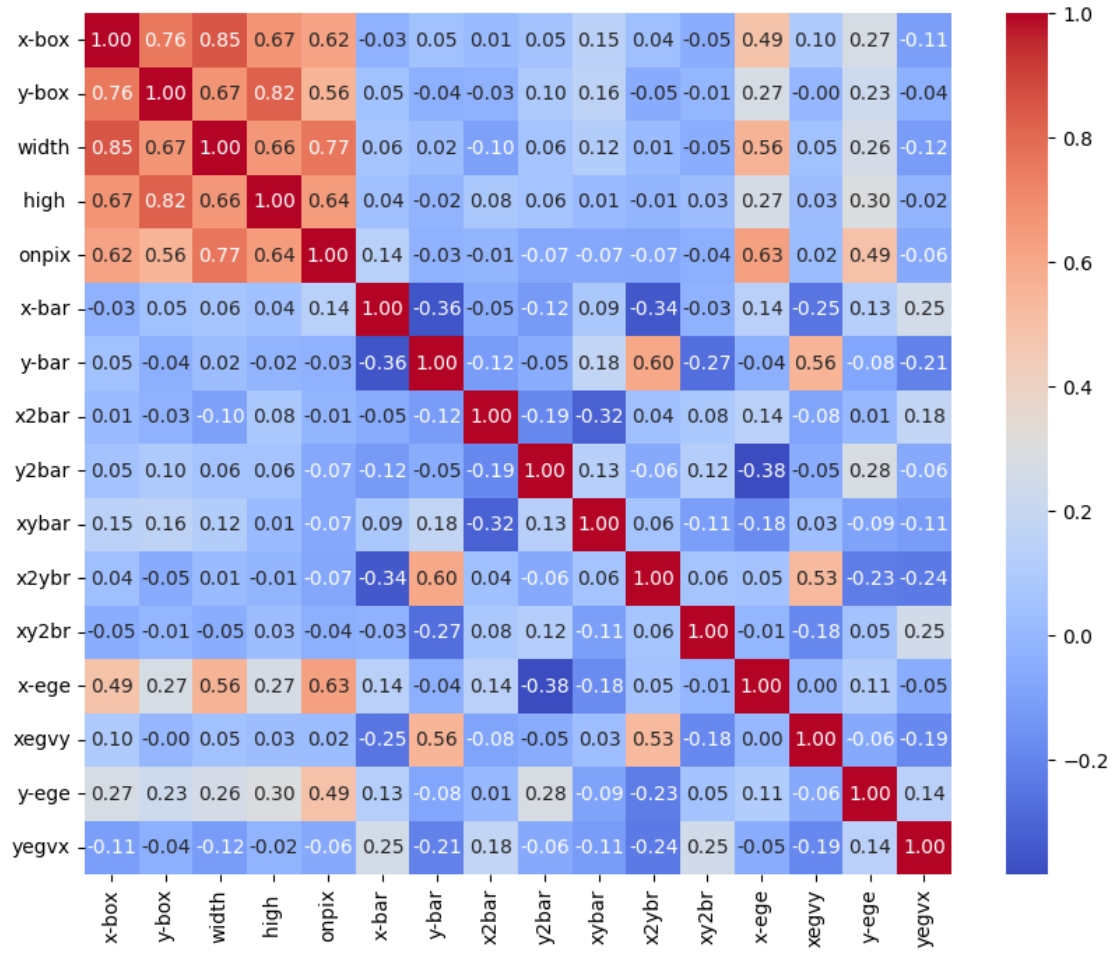
[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 17 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   lettr   20000 non-null  object
 1   x-box   20000 non-null  int64
 2   y-box   20000 non-null  int64
 3   width   20000 non-null  int64
 4   high    20000 non-null  int64
 5   onpix   20000 non-null  int64
 6   x-bar   20000 non-null  int64
 7   y-bar   20000 non-null  int64
 8   x2bar   20000 non-null  int64
 9   y2bar   20000 non-null  int64
 10  xybar   20000 non-null  int64
 11  x2ybr   20000 non-null  int64
 12  xy2br   20000 non-null  int64
 13  x-ege   20000 non-null  int64
 14  xegvy   20000 non-null  int64
 15  y-ege   20000 non-null  int64
 16  yegvx   20000 non-null  int64
dtypes: int64(16), object(1)
memory usage: 2.6+ MB
```

[5]:
```
target = "lettr"
```

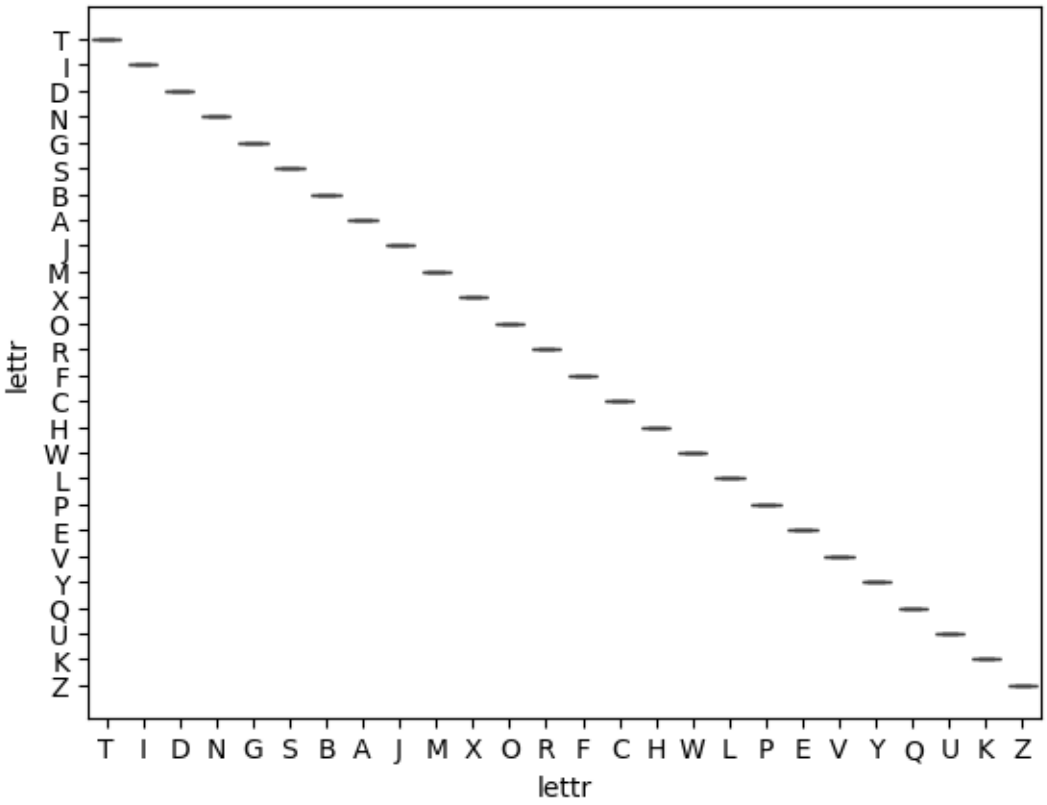
1.3 Pairwise correlation of columns

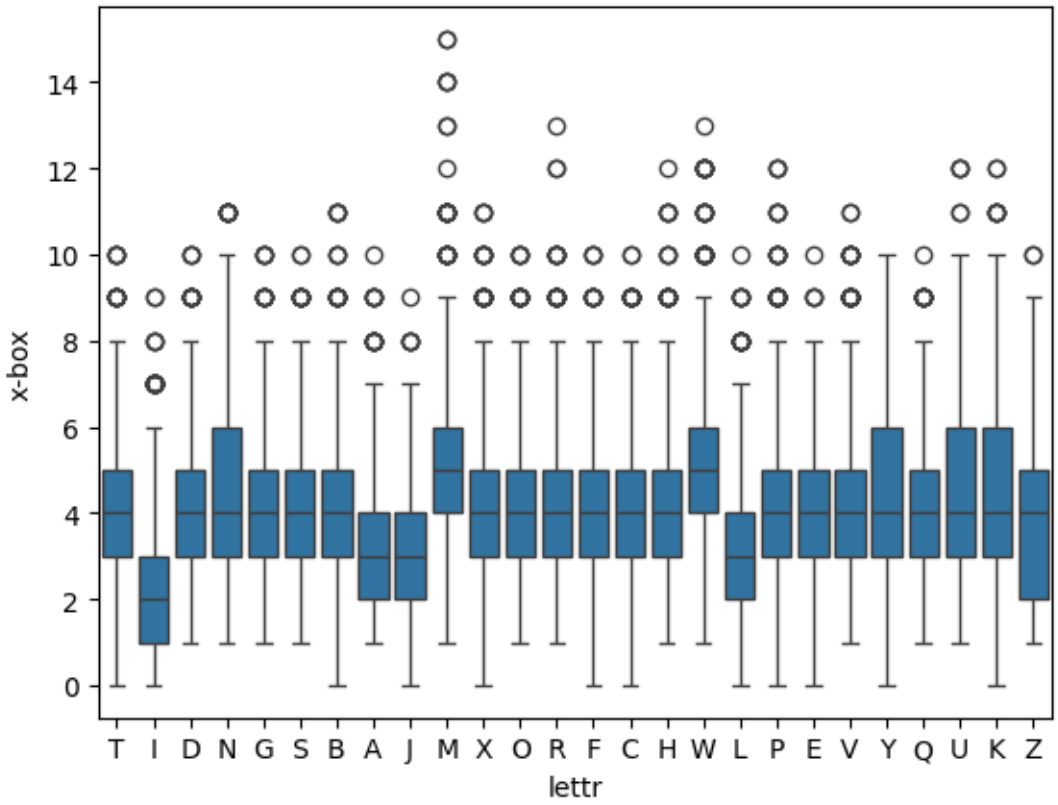
[6]:
```
corr = df.corr()
fig, ax = plt.subplots(figsize=(10, 8))
sns.heatmap(corr,  cmap='coolwarm', annot=True, fmt=".2f")
plt.show()
```

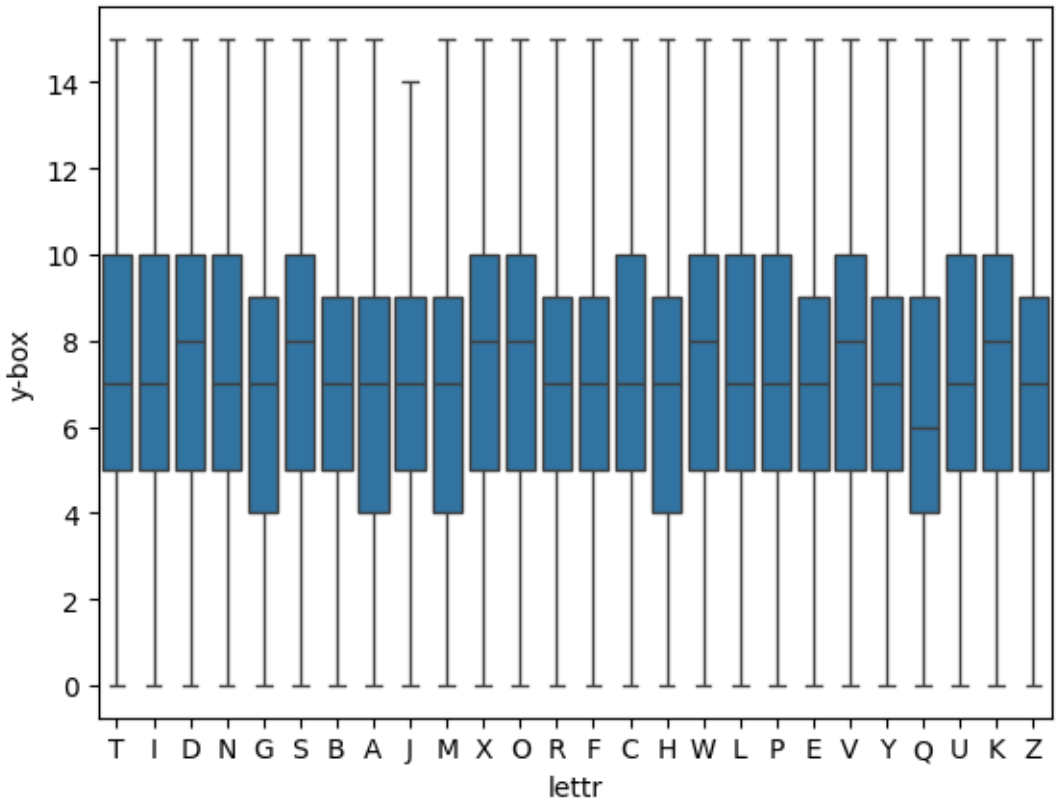


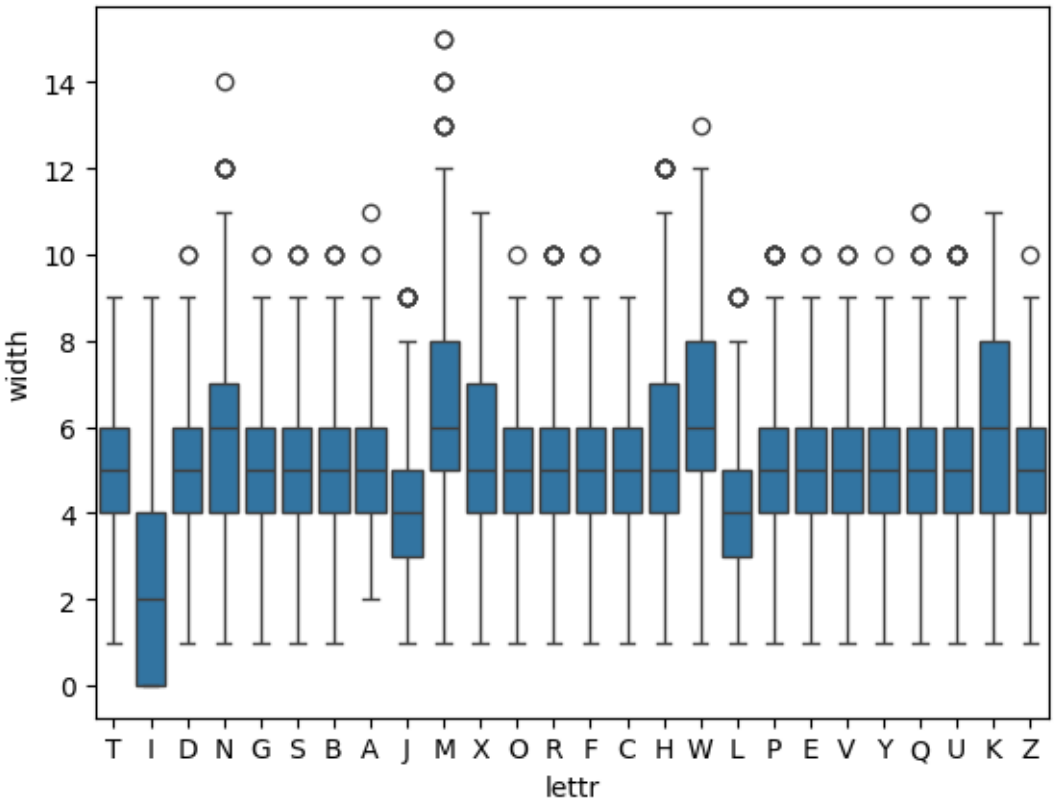
1.4 Boxplots

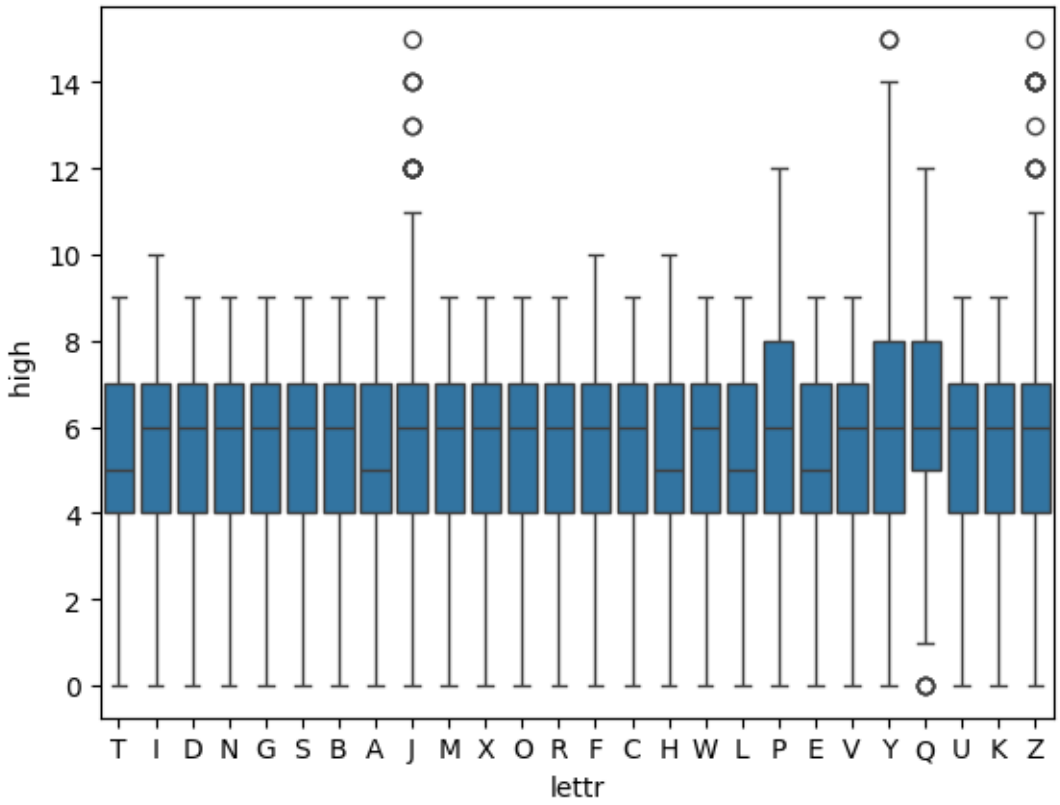
[7]:
```
feature_names = df.columns

for i in range(len(feature_names)):
    figure = plt.figure()
    ax = sns.boxplot(x=target, y=feature_names[i], data=df)
```

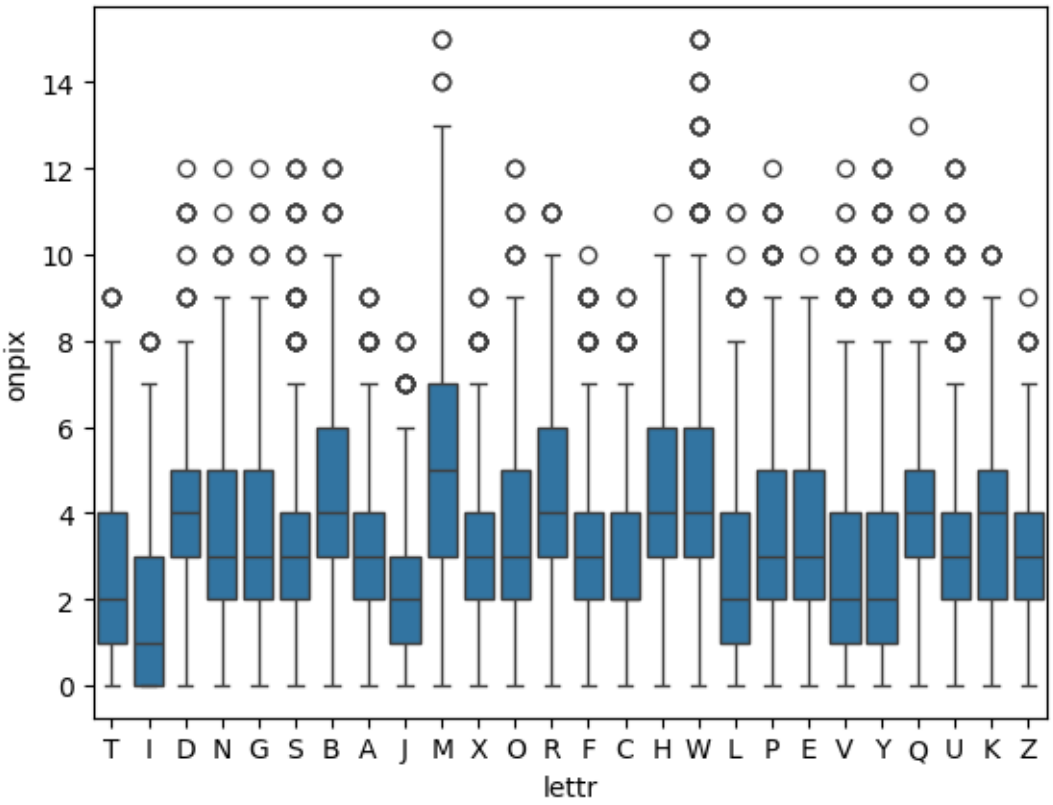

lettr
T
I
D
N
G
S
B
A
J
M
X
O
R
F
C
H
W
L
P
E
V
Y
Q
U
K
Z
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

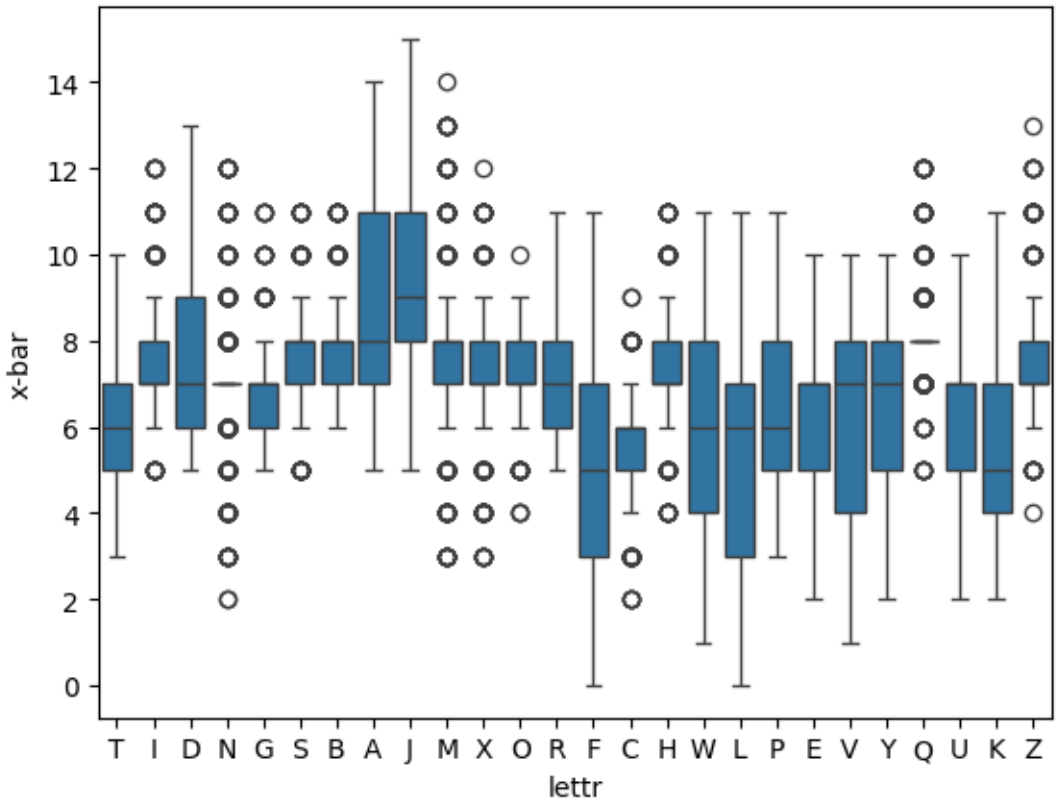

x-box
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

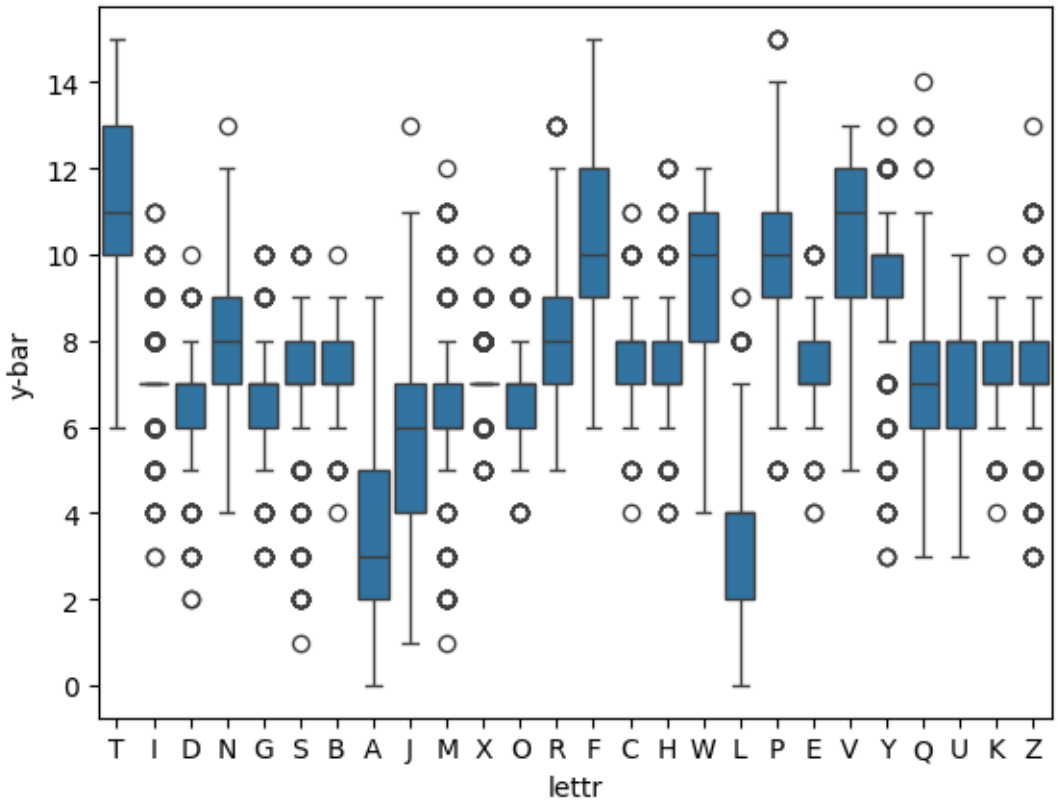

y-box
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

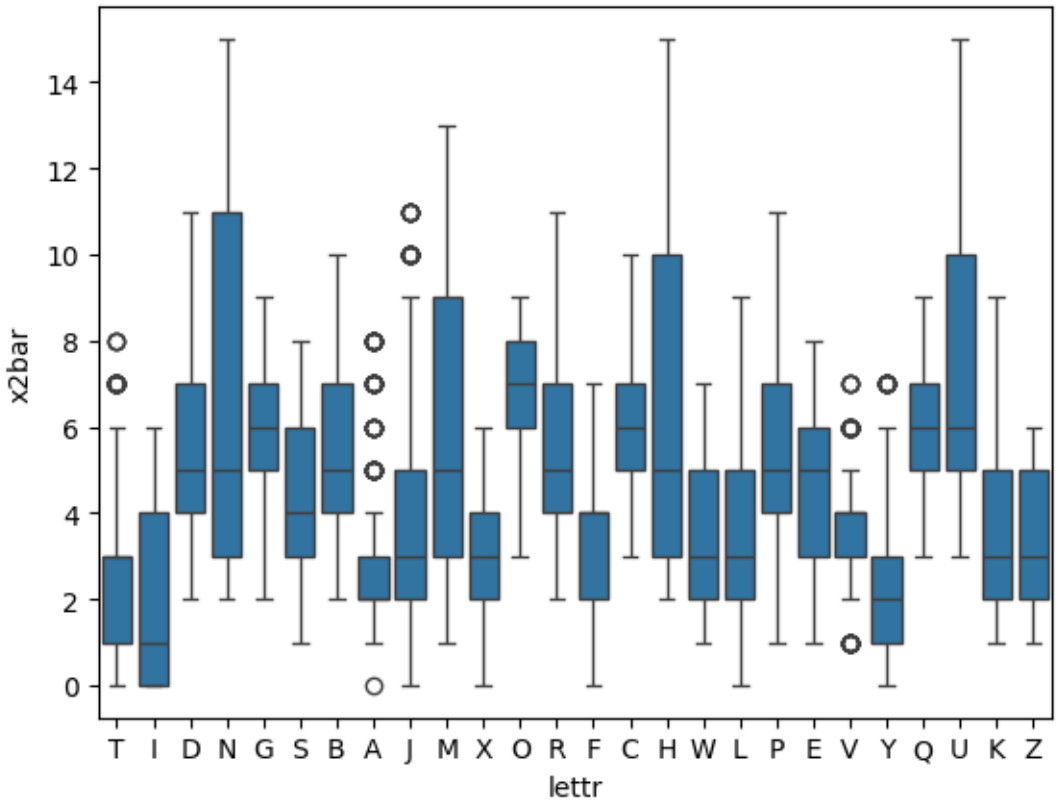

width
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

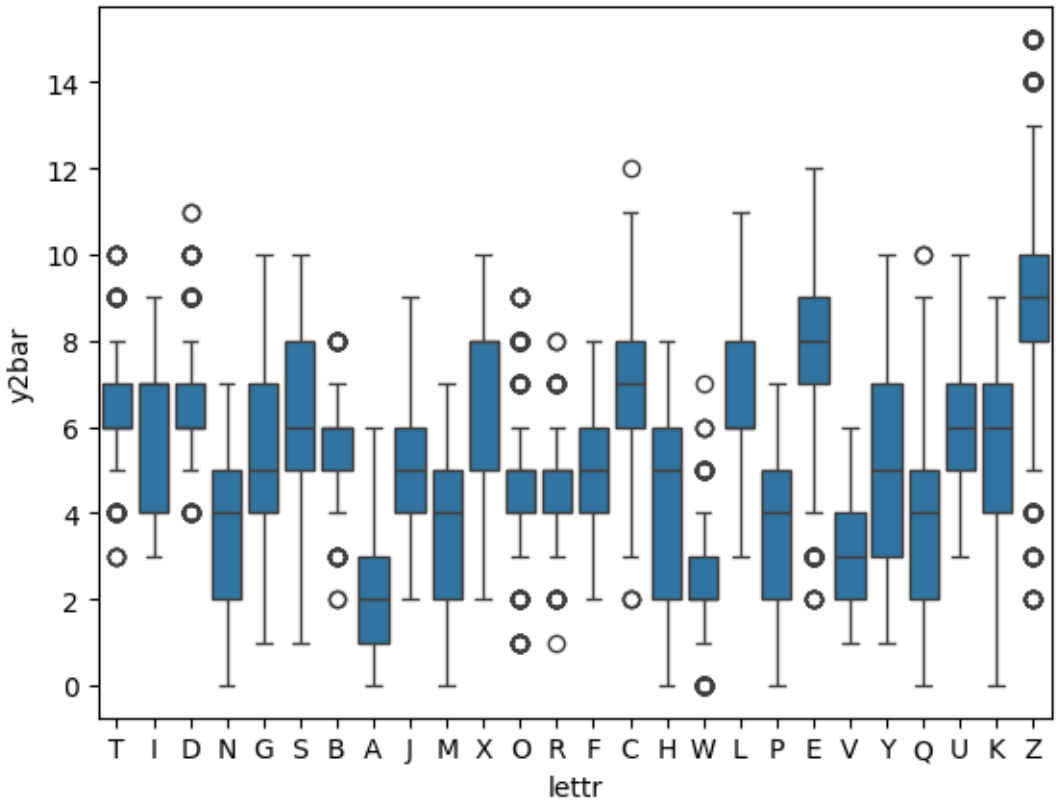

high
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

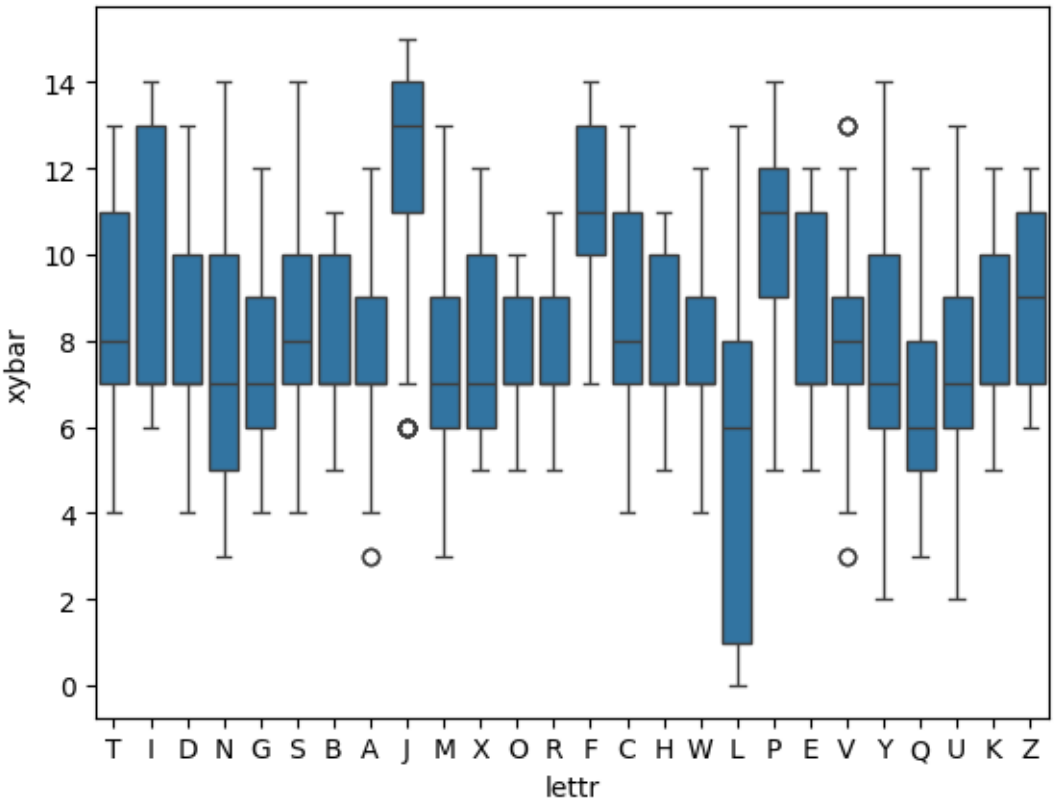

onpix
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

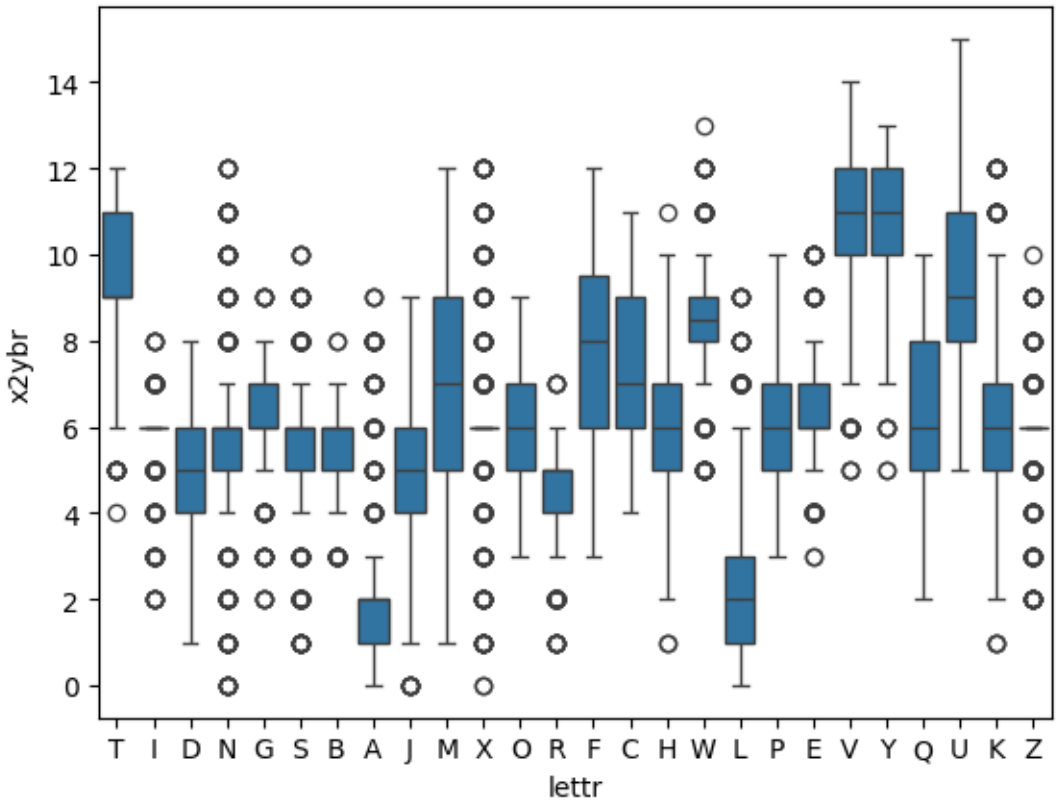

x-bar
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

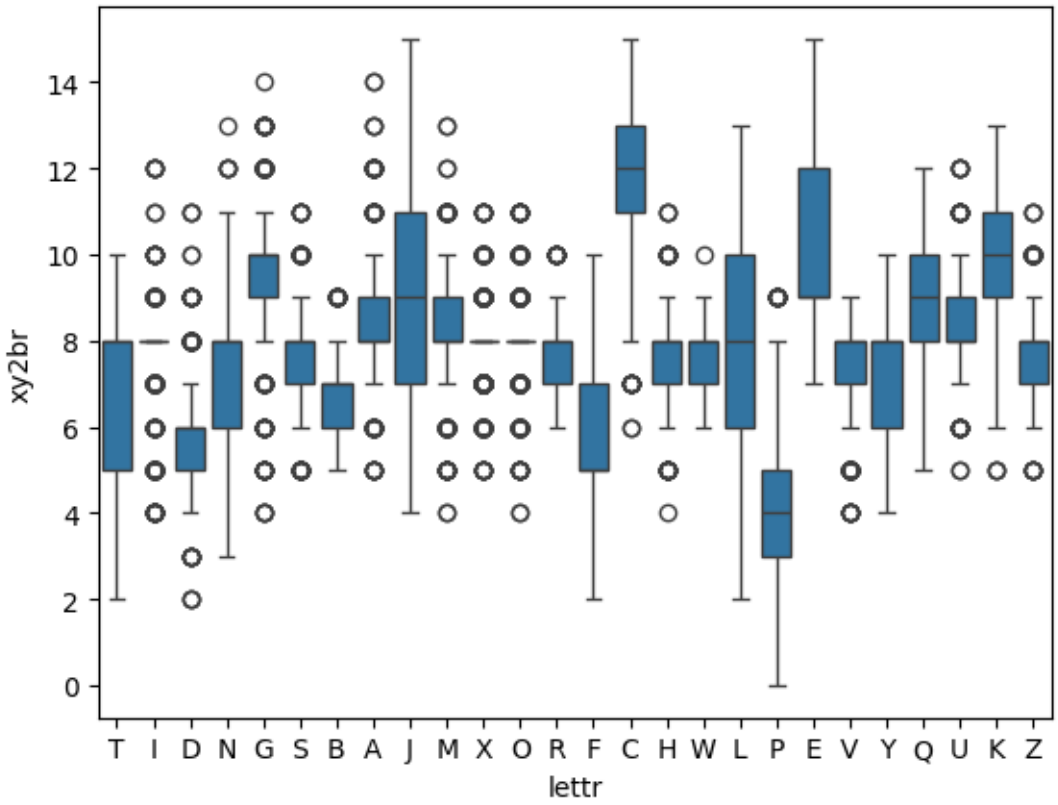

y-bar
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

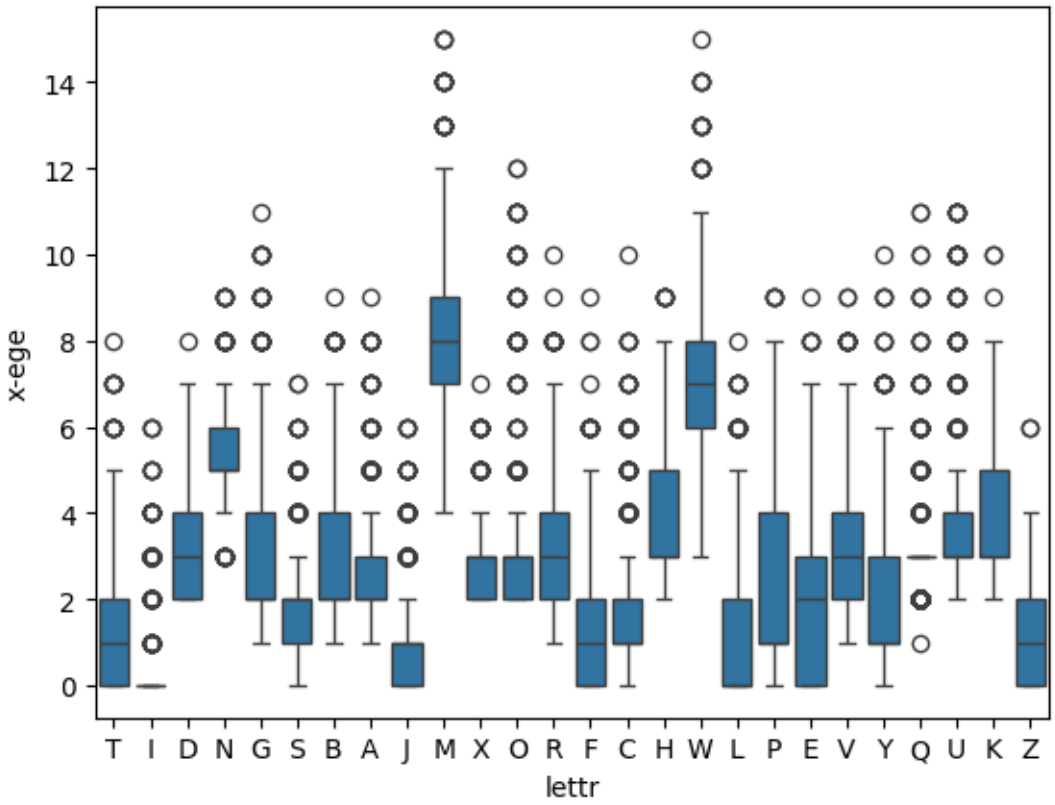

x2bar
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

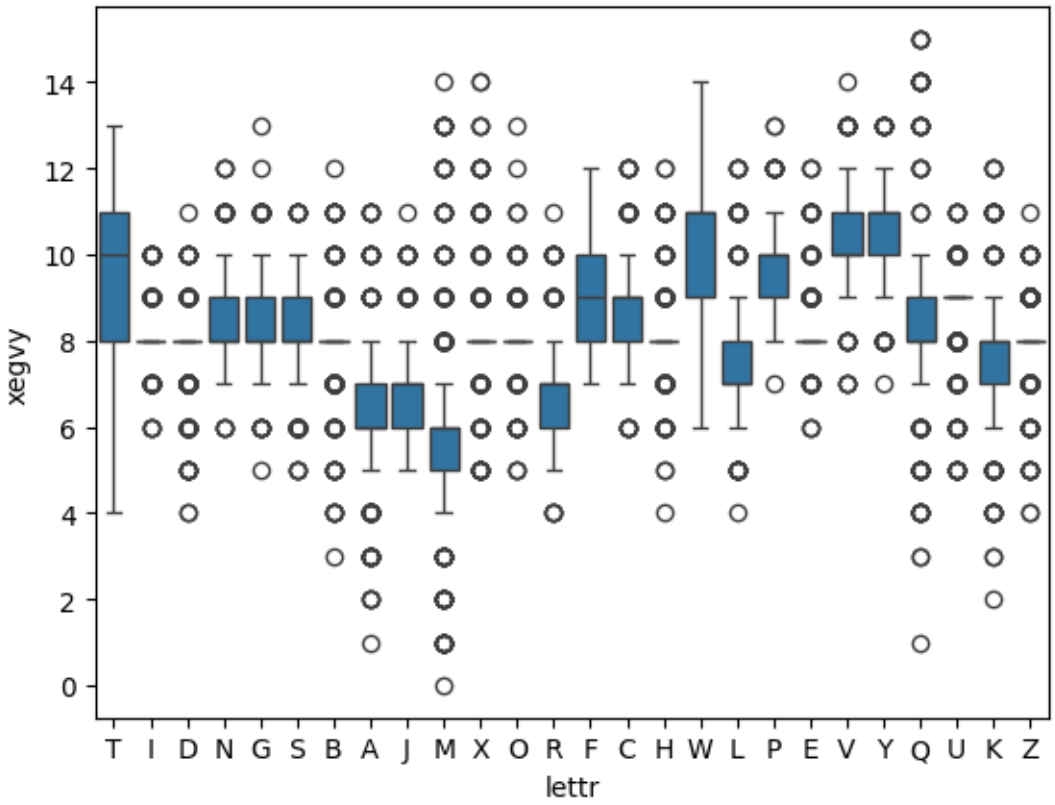

y2bar
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

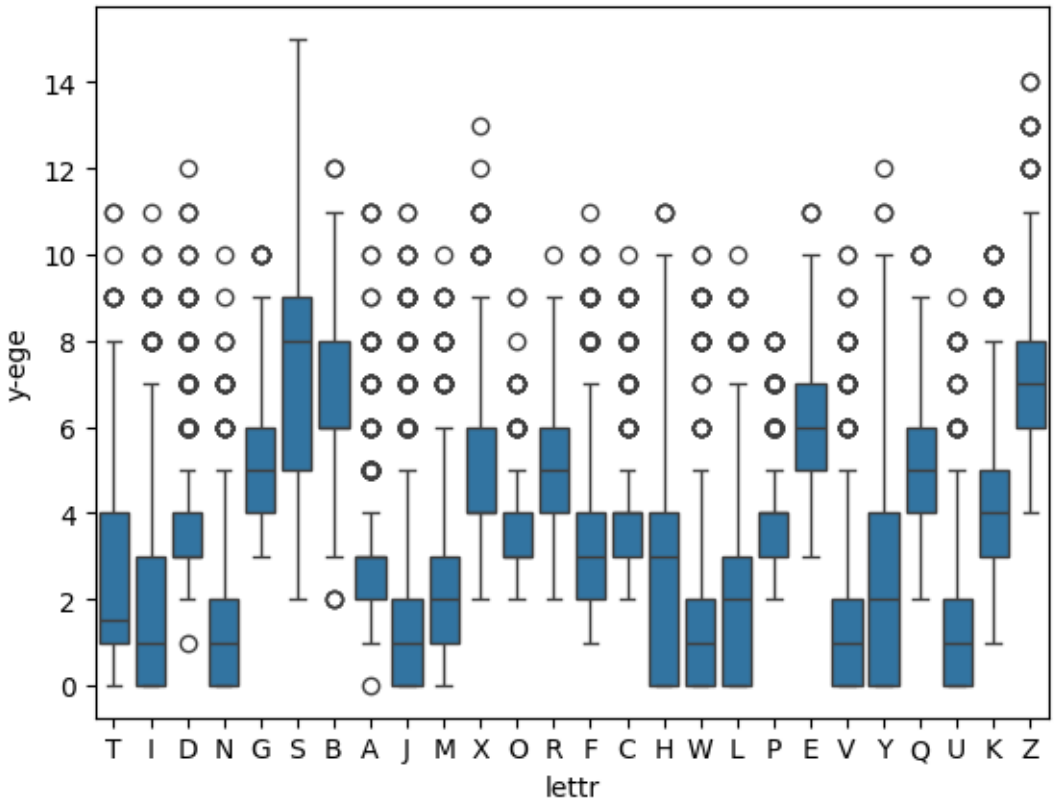

xybar
14
12
10
8
6
4
2
0
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

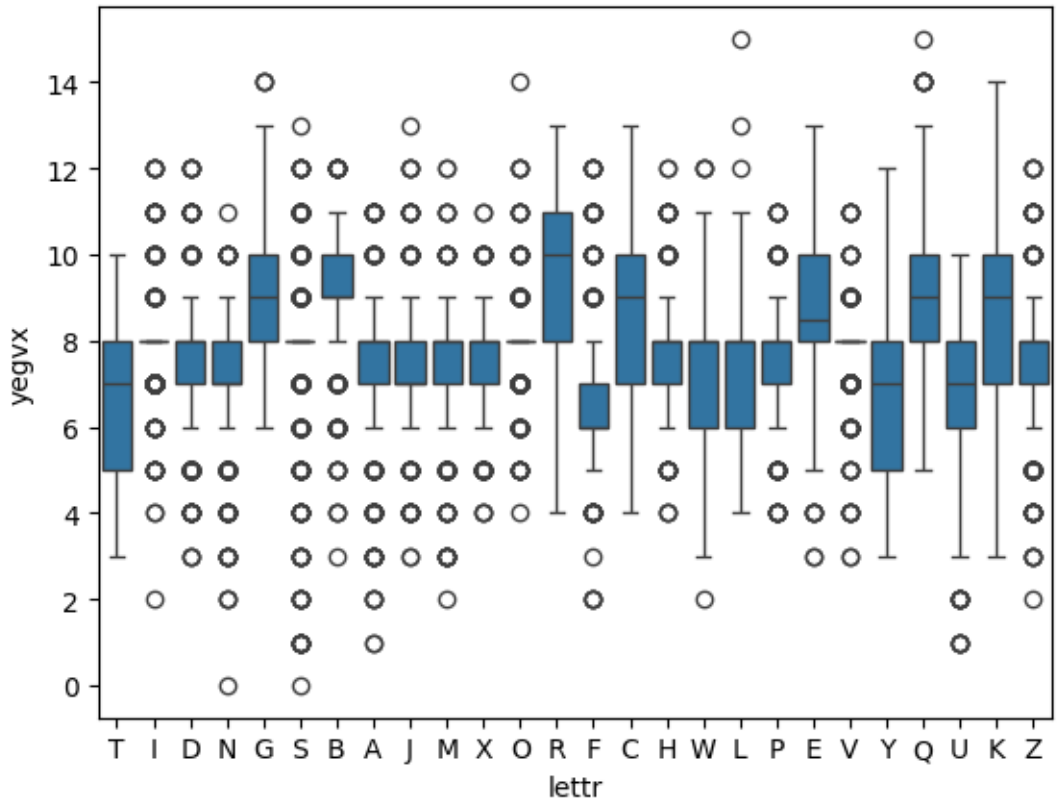

x2ybr
14
12
10
8
6
4
2
0
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr


xy2br
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr


x-ege
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr


xegvy
14
12
10
8
6
4
2
0
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr


y-ege
0
2
4
6
8
10
12
14
T I D N G S B A J M X O R F C H W L P E V Y Q U K Z
lettr

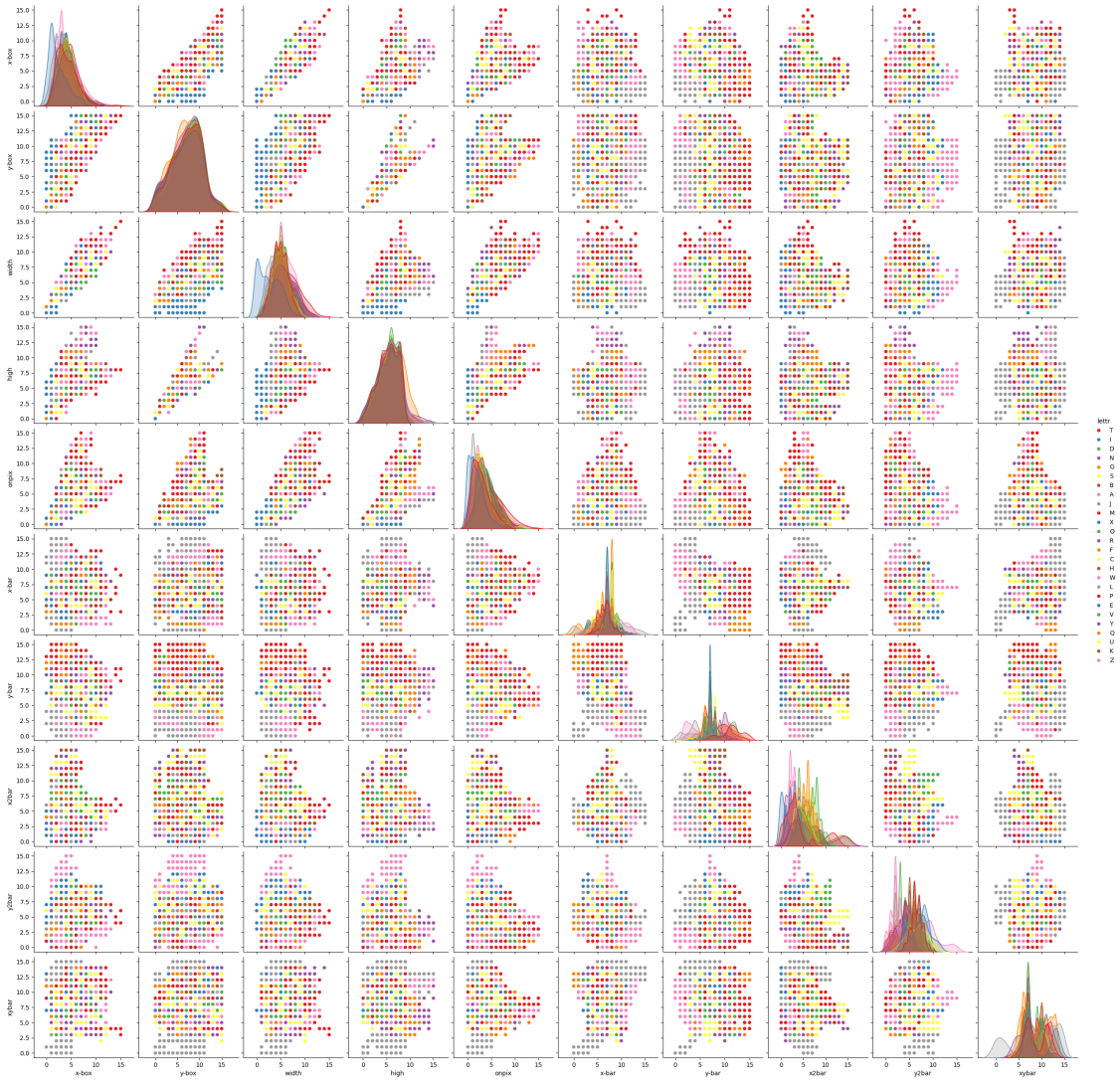


1.5 Pairplot

[8]:
```
# part = df[["lettr","x-box","y-box","width","high␣
  ↪","onpix","x-bar","y-bar","x2bar","y2bar","xybar","x2ybr","xy2br","x-ege","xegvy","y-ege","
part = df[["lettr","x-box","y-box","width","high␣
  ↪","onpix","x-bar","y-bar","x2bar","y2bar","xybar"]]
sns.pairplot(part, hue=target, palette = 'Set1')
```

```
C:\Users\Lenovo\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5
n2kfra8p0\LocalCache\local-packages\Python311\site-
packages\seaborn\axisgrid.py:123: UserWarning: The figure layout has changed to
tight
  self._figure.tight_layout(*args, **kwargs)
```

[8]:
```
<seaborn.axisgrid.PairGrid at 0x138725111d0>
```



[9]:
```
part2 = df[["lettr","x2ybr","xy2br","x-ege","xegvy","y-ege","yegvx"]]
sns.pairplot(part2, hue=target, palette = 'Set1')
```

```
C:\Users\Lenovo\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5
n2kfra8p0\LocalCache\local-packages\Python311\site-
packages\seaborn\axisgrid.py:123: UserWarning: The figure layout has changed to
tight
  self._figure.tight_layout(*args, **kwargs)
```

[9]:
```
<seaborn.axisgrid.PairGrid at 0x1386f782b10>
```

1.6 Normalisation

1.7 Learning

[10]:
```
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
```

Divide data

[11]:
```
train, test = train_test_split(df, test_size=0.2)

train_classes = train[target]
train = train.drop([target], axis=1)
test_classes = test[target]
test = test.drop([target], axis=1)
```

1.7.1 KNN

[12]:
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=4, weights="distance", algorithm="brute")
knn.fit(train, train_classes)
prediction = knn.predict(test)

print(classification_report(test_classes, prediction))
confusion_matrix(test_classes, prediction)
```

```
              precision    recall  f1-score   support

           A       0.99      0.99      0.99       168
           B       0.91      0.90      0.91       164
           C       0.99      0.97      0.98       157
           D       0.98      0.98      0.98       160
           E       0.95      0.92      0.94       149
           F       0.93      0.94      0.93       158
           G       0.96      0.98      0.97       159
           H       0.89      0.91      0.90       138
           I       0.95      0.96      0.96       129
           J       0.97      0.96      0.96       150
           K       0.94      0.86      0.90       159
           L       0.99      0.99      0.99       149
           M       0.99      0.99      0.99       168
           N       0.97      0.95      0.96       145
           O       0.94      0.99      0.97       148
           P       0.96      0.90      0.93       145
           Q       0.97      0.94      0.96       151
           R       0.85      0.94      0.89       157
           S       0.96      0.98      0.97       162
           T       0.98      0.98      0.98       161
           U       0.99      0.98      0.98       159
           V       0.96      0.97      0.97       148
           W       0.99      0.99      0.99       150
           X       0.95      0.96      0.95       166
           Y       0.99      0.97      0.98       156
           Z       0.99      0.99      0.99       144

    accuracy                           0.96      4000
   macro avg       0.96      0.96      0.96      4000
weighted avg       0.96      0.96      0.96      4000
```

[12]:
```
array([[167,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   1,   0],
       [  0, 148,   0,   0,   2,   0,   0,   2,   0,   0,   2,   0,   0,
          0,   0,   0,   0,   7,   2,   0,   0,   1,   0,   0,   0,   0],
```

```
[  0,   0, 152,   0,   2,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   1,   0,   0,   0,   0,   0,   1,   0,   1,   0,   0,   0],
[  0,   0,   0, 157,   0,   0,   0,   2,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   1,   0,   0,   0,   0,   0,   0,   0],
[  0,   1,   1,   0, 137,   1,   2,   0,   0,   0,   1,   1,   0,
   0,   0,   0,   1,   0,   1,   0,   0,   0,   0,   2,   0,   1],
[  0,   0,   0,   0,   0, 149,   0,   0,   0,   0,   0,   0,   0,
   2,   0,   4,   0,   0,   0,   2,   0,   0,   0,   0,   0,   1],
[  0,   0,   0,   1,   0,   0, 156,   0,   0,   0,   0,   0,   0,
   0,   2,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   2,   0,   0,   0,   0,   2, 126,   0,   0,   1,   0,   1,
   1,   1,   0,   0,   4,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0, 124,   5,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   6, 144,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   1,   0,   0,   1,   0,   0,   6,   0,   0, 137,   0,   0,
   0,   0,   0,   0,   8,   0,   0,   0,   0,   0,   6,   0,   0],
[  0,   0,   0,   0,   0,   0,   1,   0,   0,   0,   1, 147,   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, 166,
   0,   0,   0,   0,   0,   0,   0,   0,   1,   1,   0,   0,   0],
[  0,   0,   0,   2,   0,   0,   0,   2,   0,   0,   0,   0,   0,
 138,   1,   0,   0,   1,   0,   0,   0,   1,   0,   0,   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0, 147,   0,   1,   0,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   1,   0,   0,   0,  11,   0,   1,   0,   0,   0,   1,   1,
   0,   0, 130,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   0,   0,   0,   1,   0,   2,   0,   0,   0,   0,   0,   0,
   0,   3,   1, 142,   2,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   3,   0,   1,   0,   0,   0,   1,   0,   0,   2,   0,   0,
   2,   0,   0,   0, 148,   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,   4, 158,   0,   0,   0,   0,   0,   0,   0],
[  1,   0,   0,   0,   0,   0,   0,   1,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0, 157,   0,   1,   0,   0,   1,   0],
[  1,   0,   0,   0,   0,   0,   0,   1,   0,   0,   0,   0,   0,
   0,   1,   0,   0,   0,   0,   0, 156,   0,   0,   0,   0,   0],
[  0,   4,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0, 144,   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, 149,   0,   0,   0],
[  0,   2,   0,   0,   1,   0,   0,   0,   0,   0,   2,   0,   0,
   0,   0,   0,   0,   0,   2,   0,   0,   0,   0, 159,   0,   0],
[  0,   1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   1,   1,   1,   0,   1, 151,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
```

```
       0,   0,   0,   2,   0,   0,   0,   0,   0,   0,   0,   0, 142]],
      dtype=int64)
```

1.7.2 Choise tree

[13]:
```
from sklearn.tree import DecisionTreeClassifier

tree = DecisionTreeClassifier(criterion="entropy", splitter="best",␣
 ↪max_depth=20, min_samples_split=2)
tree.fit(train,train_classes)
prediction = tree.predict(test)

print(classification_report(test_classes, prediction))
confusion_matrix(test_classes, prediction)
```

```
              precision    recall  f1-score   support

           A       0.94      0.95      0.94       168
           B       0.85      0.85      0.85       164
           C       0.89      0.90      0.90       157
           D       0.86      0.89      0.87       160
           E       0.91      0.86      0.89       149
           F       0.89      0.82      0.86       158
           G       0.84      0.90      0.87       159
           H       0.78      0.79      0.78       138
           I       0.89      0.94      0.91       129
           J       0.89      0.90      0.90       150
           K       0.86      0.82      0.84       159
           L       0.88      0.91      0.90       149
           M       0.94      0.95      0.94       168
           N       0.93      0.88      0.90       145
           O       0.84      0.89      0.87       148
           P       0.85      0.88      0.86       145
           Q       0.90      0.86      0.88       151
           R       0.83      0.83      0.83       157
           S       0.90      0.88      0.89       162
           T       0.88      0.93      0.90       161
           U       0.92      0.92      0.92       159
           V       0.89      0.91      0.90       148
           W       0.96      0.91      0.94       150
           X       0.91      0.92      0.91       166
           Y       0.90      0.87      0.88       156
           Z       0.96      0.92      0.94       144

    accuracy                           0.89      4000
   macro avg       0.89      0.89      0.89      4000
weighted avg       0.89      0.89      0.89      4000
```

```
[13]: array([[160,   0,   2,   0,   0,   0,   0,   2,   0,   1,   0,   0,   1,
                0,   0,   0,   0,   0,   0,   1,   0,   0,   0,   0,   1,   0],
             [  0, 140,   0,   6,   0,   2,   0,   4,   1,   1,   0,   0,   0,
                0,   1,   1,   0,   3,   0,   0,   0,   4,   0,   1,   0,   0],
             [  0,   0, 142,   0,   2,   0,   3,   0,   0,   0,   3,   2,   0,
                0,   1,   0,   0,   0,   0,   0,   2,   0,   1,   0,   1,   0],
             [  0,   2,   0, 143,   0,   1,   1,   5,   0,   0,   0,   0,   0,
                2,   3,   0,   0,   3,   0,   0,   0,   0,   0,   0,   0,   0],
             [  0,   0,   3,   0, 128,   0,   5,   0,   1,   0,   2,   3,   0,
                0,   0,   0,   0,   1,   1,   0,   0,   1,   0,   1,   2,   1],
             [  0,   1,   1,   1,   1, 130,   0,   1,   1,   1,   0,   0,   1,
                0,   0,   8,   0,   0,   4,   5,   0,   0,   0,   1,   2,   0],
             [  0,   1,   4,   0,   2,   1, 143,   1,   0,   0,   0,   1,   0,
                0,   1,   1,   2,   0,   0,   1,   0,   0,   1,   0,   0,   0],
             [  1,   1,   0,   2,   0,   1,   2, 109,   0,   0,   4,   3,   1,
                0,   1,   1,   2,   5,   0,   1,   2,   0,   0,   1,   1,   0],
             [  0,   1,   0,   0,   1,   0,   0,   0, 121,   5,   0,   0,   0,
                0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   1,   0],
             [  0,   0,   0,   2,   0,   2,   0,   0,   4, 135,   2,   1,   0,
                0,   0,   1,   0,   0,   0,   0,   1,   0,   0,   1,   1,   0],
             [  0,   1,   1,   3,   1,   0,   2,   3,   1,   0, 131,   1,   1,
                0,   0,   0,   0,   6,   0,   1,   1,   0,   0,   5,   0,   1],
             [  0,   1,   0,   0,   1,   0,   1,   0,   0,   1,   0, 136,   0,
                0,   2,   2,   1,   2,   1,   1,   0,   0,   0,   0,   0,   0],
             [  2,   0,   0,   0,   0,   0,   1,   1,   0,   0,   0,   0, 159,
                4,   0,   0,   0,   0,   0,   0,   0,   0,   1,   0,   0,   0],
             [  1,   0,   0,   3,   0,   0,   1,   2,   0,   0,   0,   0,   1,
              127,   2,   0,   0,   0,   0,   0,   3,   1,   0,   0,   4,   0],
             [  0,   0,   1,   0,   0,   0,   0,   1,   0,   0,   1,   0,   1,
                1, 132,   1,   6,   1,   0,   0,   3,   0,   0,   0,   0,   0],
             [  0,   2,   0,   0,   0,   3,   4,   1,   1,   0,   0,   1,   1,
                0,   0, 128,   0,   0,   1,   1,   0,   1,   0,   1,   0,   0],
             [  1,   0,   2,   2,   0,   0,   2,   2,   0,   0,   0,   3,   0,
                0,   2,   1, 130,   1,   1,   0,   0,   0,   0,   1,   2,   1],
             [  1,   5,   0,   3,   1,   0,   2,   1,   1,   1,   4,   0,   1,
                0,   3,   1,   0, 130,   1,   1,   0,   0,   1,   0,   0,   0],
             [  0,   2,   1,   0,   1,   1,   1,   1,   2,   0,   2,   1,   0,
                0,   1,   1,   1,   2, 142,   0,   0,   0,   0,   1,   0,   2],
             [  0,   0,   1,   0,   0,   2,   0,   1,   1,   0,   0,   0,   0,
                0,   1,   2,   0,   0,   2, 150,   0,   1,   0,   0,   0,   0],
             [  1,   1,   0,   1,   0,   0,   0,   1,   0,   2,   0,   0,   1,
                1,   3,   0,   0,   0,   1,   0, 147,   0,   0,   0,   0,   0],
             [  0,   2,   0,   0,   0,   0,   1,   3,   0,   0,   0,   2,   0,
                0,   1,   3,   0,   0,   0,   1,   0, 135,   0,   0,   0,   0],
             [  1,   0,   0,   0,   0,   1,   0,   0,   0,   0,   0,   0,   2,
                0,   2,   0,   1,   0,   0,   1,   0,   5, 137,   0,   0,   0],
             [  0,   3,   0,   1,   1,   0,   0,   0,   0,   1,   3,   0,   0,
```

```
   0,   0,   0,   0,   2,   3,   0,   0,   0,   0, 152,   0,   0],
 [  2,   2,   1,   0,   0,   2,   0,   1,   1,   1,   0,   0,   0,
   0,   1,   0,   0,   0,   0,   6,   0,   3,   1,   0, 135,   0],
 [  1,   0,   0,   0,   1,   0,   1,   0,   1,   2,   0,   0,   0,
   1,   0,   0,   1,   0,   1,   1,   0,   0,   0,   2,   0, 132]],
dtype=int64)
```

1.7.3 SVM

[14]:
```
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV

parameters = {'gamma':('scale', 'auto'), 'C':[10, 15, 20, 25, 30]}

svc = SVC()
clf = GridSearchCV(svc, parameters)
clf.fit(train, train_classes)
```

[14]:
```
GridSearchCV(estimator=SVC(),
             param_grid={'C': [10, 15, 20, 25, 30], 'gamma': ('scale', 'auto')})
```

[18]:
```
svc = SVC(C=20, gamma='auto')
svc.fit(train, train_classes)
prediction = svc.predict(test)

print(classification_report(test_classes, prediction))
confusion_matrix(test_classes, prediction)
```

```
              precision    recall  f1-score   support

           A       0.99      0.99      0.99       168
           B       0.95      0.96      0.96       164
           C       0.99      0.99      0.99       157
           D       0.98      0.99      0.98       160
           E       0.98      0.96      0.97       149
           F       0.98      0.97      0.98       158
           G       0.95      0.98      0.96       159
           H       0.92      0.91      0.92       138
           I       0.97      0.97      0.97       129
           J       0.98      0.97      0.98       150
           K       0.97      0.93      0.95       159
           L       0.99      0.99      0.99       149
           M       0.98      0.99      0.99       168
           N       0.98      0.97      0.97       145
           O       0.98      0.99      0.98       148
           P       0.98      0.95      0.97       145
           Q       0.99      0.98      0.99       151
           R       0.93      0.95      0.94       157
```

```
           S       0.99      0.99      0.99       162
           T       1.00      0.99      1.00       161
           U       0.99      0.99      0.99       159
           V       1.00      0.99      1.00       148
           W       0.99      0.99      0.99       150
           X       0.98      0.99      0.99       166
           Y       0.98      0.99      0.98       156
           Z       0.99      1.00      1.00       144

    accuracy                           0.98      4000
   macro avg       0.98      0.98      0.98      4000
weighted avg       0.98      0.98      0.98      4000
```

[18]:
```
array([[166,   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, 158,   0,   0,   1,   0,   0,   1,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   3,   1,   0,   0,   0,   0,   0,   0,   0],
       [  0,   0, 156,   0,   0,   0,   1,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
       [  0,   0,   0, 158,   0,   0,   1,   1,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
       [  0,   1,   0,   0, 143,   1,   2,   0,   0,   0,   0,   1,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   1,   0],
       [  0,   0,   0,   1,   0, 154,   0,   1,   0,   0,   0,   0,   0,
          0,   0,   2,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
       [  0,   0,   2,   0,   0,   0, 156,   0,   0,   0,   0,   0,   0,
          0,   1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
       [  0,   2,   0,   1,   0,   0,   3, 126,   0,   0,   0,   0,   2,
          0,   1,   0,   0,   2,   0,   0,   1,   0,   0,   0,   0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0, 125,   3,   0,   0,   0,
          1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   4, 146,   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, 148,   0,   0,
          0,   0,   0,   0,   6,   0,   0,   0,   0,   0,   3,   0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   1, 148,   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, 167,
          1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
       [  0,   0,   0,   2,   0,   0,   0,   2,   0,   0,   0,   1,   0,
        140,   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, 146,   0,   1,   0,   0,   0,   1,   0,   0,   0,   0,   0],
       [  0,   1,   0,   0,   2,   2,   1,   0,   0,   0,   0,   0,   0,
          0,   0, 138,   0,   0,   0,   0,   0,   0,   1,   0,   0,   0],
       [  0,   0,   0,   0,   0,   0,   1,   0,   0,   0,   0,   0,   0,
```

```
   0,   1,   1, 148,   0,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   2,   0,   0,   0,   0,   0,   2,   0,   0,   3,   0,   0,
   1,   0,   0,   0, 149,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0, 161,   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, 160,   0,   0,   0,   0,   0,   0],
[  1,   0,   0,   0,   0,   0,   0,   1,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0, 157,   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, 147,   0,   0,   1,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   1,
   0,   0,   0,   0,   0,   0,   0,   0,   0, 149,   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, 165,   0,   0],
[  0,   1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   1, 154,   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, 144]],
dtype=int64)
```

1.7.4 Random Forest

[19]:
```
from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier(n_estimators=130, criterion="entropy",␣
 ↪max_depth=20, min_samples_split=2)
rfc.fit(train,train_classes)
prediction = rfc.predict(test)

print(classification_report(test_classes, prediction))
confusion_matrix(test_classes, prediction)
```

```
              precision    recall  f1-score   support

           A       0.99      1.00      0.99       168
           B       0.89      0.96      0.92       164
           C       0.99      0.97      0.98       157
           D       0.95      0.98      0.97       160
           E       0.95      0.93      0.94       149
           F       0.94      0.92      0.93       158
           G       0.95      0.97      0.96       159
           H       0.91      0.91      0.91       138
           I       0.94      0.97      0.95       129
           J       0.97      0.93      0.95       150
           K       0.95      0.91      0.93       159
           L       0.99      0.97      0.98       149
           M       0.99      0.98      0.98       168
```

```
           N       0.96      0.95      0.96       145
           O       0.96      0.98      0.97       148
           P       0.95      0.92      0.94       145
           Q       0.94      0.97      0.95       151
           R       0.92      0.95      0.93       157
           S       0.99      0.96      0.97       162
           T       0.98      0.98      0.98       161
           U       0.98      1.00      0.99       159
           V       0.98      0.97      0.98       148
           W       0.99      0.99      0.99       150
           X       0.96      0.98      0.97       166
           Y       0.99      0.99      0.99       156
           Z       0.99      0.98      0.99       144

    accuracy                           0.96      4000
   macro avg       0.96      0.96      0.96      4000
weighted avg       0.96      0.96      0.96      4000
```

[19]:
```
array([[168,   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, 157,   0,   0,   1,   0,   0,   2,   0,   0,   1,   0,   0,
          1,   0,   0,   0,   0,   0,   0,   0,   1,   0,   1,   0],
       [  0,   0, 152,   0,   2,   0,   1,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   1,   0,   0,   0,   1,   0,   0,   0,   0],
       [  0,   0,   0, 157,   0,   0,   0,   1,   0,   0,   0,   0,   0,
          2,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
       [  0,   1,   1,   0, 138,   1,   3,   0,   0,   0,   0,   2,   0,
          0,   0,   0,   1,   0,   0,   0,   0,   0,   1,   0,   1],
       [  0,   2,   0,   1,   0, 146,   0,   1,   0,   0,   1,   0,   0,
          0,   0,   6,   0,   0,   0,   1,   0,   0,   0,   0,   0],
       [  0,   1,   0,   1,   1,   0, 154,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   2,   0,   0,   0,   0,   0,   0,   0,   0],
       [  0,   2,   0,   0,   0,   1,   1, 126,   0,   0,   2,   0,   0,
          0,   0,   0,   1,   2,   0,   0,   1,   0,   0,   1,   1],
       [  0,   0,   0,   0,   0,   1,   0,   0, 125,   2,   0,   0,   0,
          0,   0,   1,   0,   0,   0,   0,   0,   0,   0,   0,   0],
       [  1,   0,   0,   0,   0,   0,   0,   1,   8, 139,   0,   0,   0,
          0,   0,   0,   0,   0,   1,   0,   0,   0,   0,   0,   0],
       [  0,   0,   0,   1,   1,   0,   0,   2,   0,   0, 144,   0,   0,
          0,   0,   0,   0,   7,   0,   1,   0,   0,   0,   3,   0,   0],
       [  0,   2,   0,   0,   0,   0,   1,   0,   0,   1,   0, 145,   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, 164,
          2,   0,   0,   0,   0,   0,   0,   0,   0,   1,   0,   0,   0],
       [  0,   0,   0,   2,   0,   0,   0,   2,   0,   0,   0,   0,   0,
        138,   1,   0,   0,   1,   0,   0,   0,   1,   0,   0,   0,   0],
```

```
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0, 145,   0,   2,   0,   0,   0,   1,   0,   0,   0,   0,   0],
[  0,   1,   0,   0,   1,   6,   1,   1,   0,   0,   0,   0,   0,
   0,   1, 134,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0],
[  1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   3,   0, 146,   1,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   3,   0,   1,   0,   0,   0,   1,   0,   0,   2,   0,   0,
   1,   0,   0,   0, 149,   0,   0,   0,   0,   0,   0,   0,   0],
[  0,   1,   0,   0,   0,   0,   0,   2,   0,   1,   0,   0,   0,
   0,   0,   0,   1,   2, 155,   0,   0,   0,   0,   0,   0,   0],
[  0,   0,   0,   1,   0,   1,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0, 158,   0,   1,   0,   0,   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0, 159,   0,   0,   0,   0,   0],
[  0,   4,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0, 144,   0,   0,   0,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   1,
   0,   1,   0,   0,   0,   0,   0,   0,   0, 148,   0,   0,   0],
[  0,   1,   0,   1,   0,   0,   0,   0,   0,   0,   1,   0,   0,
   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 163,   0,   0],
[  0,   1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   0,   0,   1,   0,   0,   0,   0,   0, 154,   0],
[  0,   0,   0,   0,   1,   0,   0,   0,   0,   0,   0,   0,   0,
   0,   0,   0,   2,   0,   0,   0,   0,   0,   0,   0,   0, 141]],
dtype=int64)
```

1.7.5 AdaBoost

[20]:
```
from sklearn.ensemble import AdaBoostClassifier

abc = AdaBoostClassifier(n_estimators=70, learning_rate=0.25)
abc.fit(train,train_classes)
prediction = abc.predict(test)

print(classification_report(test_classes, prediction))
confusion_matrix(test_classes, prediction)
```

```
C:\Users\Lenovo\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5
n2kfra8p0\LocalCache\local-packages\Python311\site-
packages\sklearn\ensemble\_weight_boosting.py:527: FutureWarning: The SAMME.R
algorithm (the default) is deprecated and will be removed in 1.6. Use the SAMME
algorithm to circumvent this warning.
  warnings.warn(
              precision    recall  f1-score   support

           A       0.71      0.07      0.13       168
           B       0.44      0.61      0.51       164
```

```
           C       0.80      0.48      0.60       157
           D       0.63      0.29      0.40       160
           E       0.31      0.35      0.33       149
           F       0.40      0.54      0.46       158
           G       0.38      0.83      0.52       159
           H       0.07      0.40      0.13       138
           I       0.42      0.06      0.11       129
           J       0.27      0.51      0.36       150
           K       0.30      0.09      0.14       159
           L       0.00      0.00      0.00       149
           M       0.52      0.37      0.43       168
           N       0.40      0.21      0.28       145
           O       0.43      0.81      0.56       148
           P       0.76      0.51      0.61       145
           Q       0.43      0.17      0.25       151
           R       0.45      0.66      0.53       157
           S       0.28      0.38      0.32       162
           T       0.36      0.21      0.27       161
           U       0.71      0.57      0.63       159
           V       0.06      0.05      0.05       148
           W       0.40      0.03      0.05       150
           X       0.44      0.78      0.57       166
           Y       0.77      0.06      0.12       156
           Z       0.30      0.07      0.11       144

    accuracy                           0.35      4000
   macro avg       0.42      0.35      0.33      4000
weighted avg       0.43      0.35      0.33      4000
```

[20]:
```
array([[ 12,   1,   0,   0,   0,   0,   4, 114,   0,   8,   1,   0,   0,
         0,   1,   0,  12,  10,   4,   0,   0,   0,   0,   1,   0,   0],
       [  0, 100,   0,   0,   0,   0,   1,   3,   1,   1,   5,   0,   0,
         0,   0,   0,   0,  38,   6,   0,   0,   3,   0,   6,   0,   0],
       [  0,   0,  75,   0,  26,   0,  29,   3,   0,   0,   3,   1,   0,
         0,   5,   0,   0,   0,   1,   2,   9,   0,   0,   3,   0,   0],
       [  0,  18,   0,  47,   0,   3,   1,  38,   0,   3,   1,   0,   0,
         1,  36,   0,   0,   6,   5,   1,   0,   0,   0,   0,   0,   0],
       [  0,   2,   3,   0,  52,   0,  28,   1,   0,   0,   7,   5,   0,
         0,   0,   0,   0,   0,   8,   6,   0,   0,   0,  35,   0,   2],
       [  0,   2,   0,   1,   0,  85,   5,  16,   0,  22,   0,   0,   0,
         0,   0,   9,   0,   2,   1,  12,   0,   1,   0,   2,   0,   0],
       [  0,   4,   1,   0,   3,   0, 132,   2,   0,   0,   3,   0,   2,
         0,   1,   0,   0,   5,   5,   0,   0,   1,   0,   0,   0,   0],
       [  0,   6,   0,   0,   0,   1,   4,  55,   0,   0,   0,   0,   1,
         0,  49,   0,   0,  15,   6,   0,   0,   1,   0,   0,   0,   0],
       [  0,   3,   1,   0,   3,   3,   1,   0,   8,  76,   0,   0,   0,
```

```
    0,   3,   0,   1,   0,  30,   0,   0,   0,   0,   0,   0,   0],
 [  0,   7,   0,   1,   0,   3,   0,  31,   4,  77,   0,   0,   0,
    0,  13,   0,   3,   4,   4,   0,   0,   0,   0,   3,   0,   0],
 [  0,   5,   4,   1,  12,   0,  15,  49,   0,   0,  14,   0,   0,
    0,  10,   0,  10,  21,   1,   2,   1,   0,   0,  14,   0,   0],
 [  2,   2,   0,  14,   2,   0,   8,   7,   2,  62,   2,   0,   0,
    0,   0,   0,   4,   1,  39,   2,   0,   0,   0,   2,   0,   0],
 [  0,   5,   1,   0,   0,   0,   0,  66,   0,   0,   3,   0,  62,
    9,   0,   0,   0,   1,   0,   0,  13,   2,   6,   0,   0,   0],
 [  1,   2,   0,   1,   0,   2,   0,  53,   0,   1,   1,   0,  27,
   31,  14,   0,   0,   2,   0,   0,   4,   6,   0,   0,   0,   0],
 [  0,   4,   1,   0,   0,   0,   8,   0,   0,   0,   0,   0,   3,
    0, 120,   0,   1,   7,   0,   0,   1,   3,   0,   0,   0,   0],
 [  0,  11,   0,   0,   0,  33,   6,  12,   2,   0,   0,   0,   0,
    0,   1,  74,   0,   0,   0,   0,   0,   3,   0,   1,   2,   0],
 [  0,   5,   1,   6,   1,   0,  76,   3,   0,   1,   1,   0,   4,
    0,  15,   0,  26,   0,  10,   0,   0,   2,   0,   0,   0,   0],
 [  0,  21,   0,   2,   0,   0,  16,  11,   0,   0,   1,   0,   0,
    0,   1,   0,   0, 103,   0,   0,   0,   2,   0,   0,   0,   0],
 [  1,  13,   0,   0,   9,   1,   1,   8,   2,   7,   2,   0,   0,
    0,   0,   0,   3,   3,  61,   1,   0,   0,   0,  32,   0,  18],
 [  0,   1,   0,   1,   0,  53,   6,  30,   0,  19,   0,   0,   0,
    0,   1,   0,   0,   1,   2,  34,   1,   0,   0,  11,   0,   1],
 [  0,   3,   6,   1,   0,   0,   3,  40,   0,   0,   0,   0,   2,
    0,   7,   0,   0,   1,   1,   3,  91,   1,   0,   0,   0,   0],
 [  0,   8,   0,   0,   0,   2,   1, 114,   0,   0,   0,   0,   0,
    1,   0,  12,   0,   0,   0,   0,   2,   7,   0,   0,   1,   0],
 [  1,   2,   0,   0,   0,   0,   0,  14,   0,   0,   0,   0,  18,
   35,   0,   0,   0,   1,   0,   0,   4,  71,   4,   0,   0,   0],
 [  0,   3,   0,   0,   2,   0,   0,   3,   0,   0,   2,   0,   0,
    0,   1,   0,   0,   5,   6,  11,   1,   0,   0, 130,   0,   2],
 [  0,   1,   1,   0,   1,  27,   2,  65,   0,   0,   1,   0,   0,
    0,   2,   3,   0,   0,   9,  19,   2,   8,   0,   5,  10,   0],
 [  0,   0,   0,   0,  57,   0,   1,   0,   0,   4,   0,   0,   0,
    0,   0,   0,   1,   5,  17,   1,   0,   0,   0,  48,   0,  10]],
dtype=int64)
```

, SVC.