Letter Recognition

Philip George, Seif Nagi, Marian Kromel, Ahmed Osama, Youssif Assem

Abstract:

On our data set, we want to generate various classifiers in order to build a machine learning model that can predict the outcome. We use numerous classifiers (KNN, Naive Bayes, SVM, Random Forest) and a deep learning model (MLP classifier) to achieve an accuracy of 0.94 in knn, 0.64 in naive Bayes, 0.99 in SVM, 0.96 in Random Forest, and 0.97 in MLP classifier. We want to improve our model by experimenting with different classifiers and testing our models on massive data.

1 Introduction

We aim to apply some multi-classifiers on the data set and get the best accuracy to make a model that can predict output according to our fetchers after applying (MLP, Random Forest, SVM) we found that the best accuracy we get in SVM and Random Forest we get in the SVM and get in Random Forest. On the other hand, we aim to apply MLP with different activation functions and optimization techniques.

2 Dataset

Dataset: https://www.kaggle.com/nishan192/letterrecognition-using-svm

Our dataset is related to letter recognition for letters from A to Z for 26 letter our dataset contains of 20,000 instances based on 16 features where x-box horizontal position of box means the horizontal of letter , y-box vertical position of box means the vertical of the letter, width is width of the box (integer),high is the height of box (integer), onpix means the total number on pixels (integer),x-bar mean x of on pixels in box (integer),y-bar mean y of on pixels in box (integer),x2bar mean x variance (integer),y2bar mean y variance (integer), xybar mean x y correlation (integer), x2ybr mean of x * x * y (integer),xy2br mean of x * y * y (integer), x-ege mean edge count left to right (integer), xegvy correlation of x-ege with y (integer), y-ege mean edge count bottom to top (integer), yegvx correlation of y-ege with x (integer)w divide our dataset into test and train where the test was 70% and train was 30%.

More info about our dataset: https://archive.ics.uci.edu/ml/datasets/Letter+Recognition

Workload Management Template
Project Title

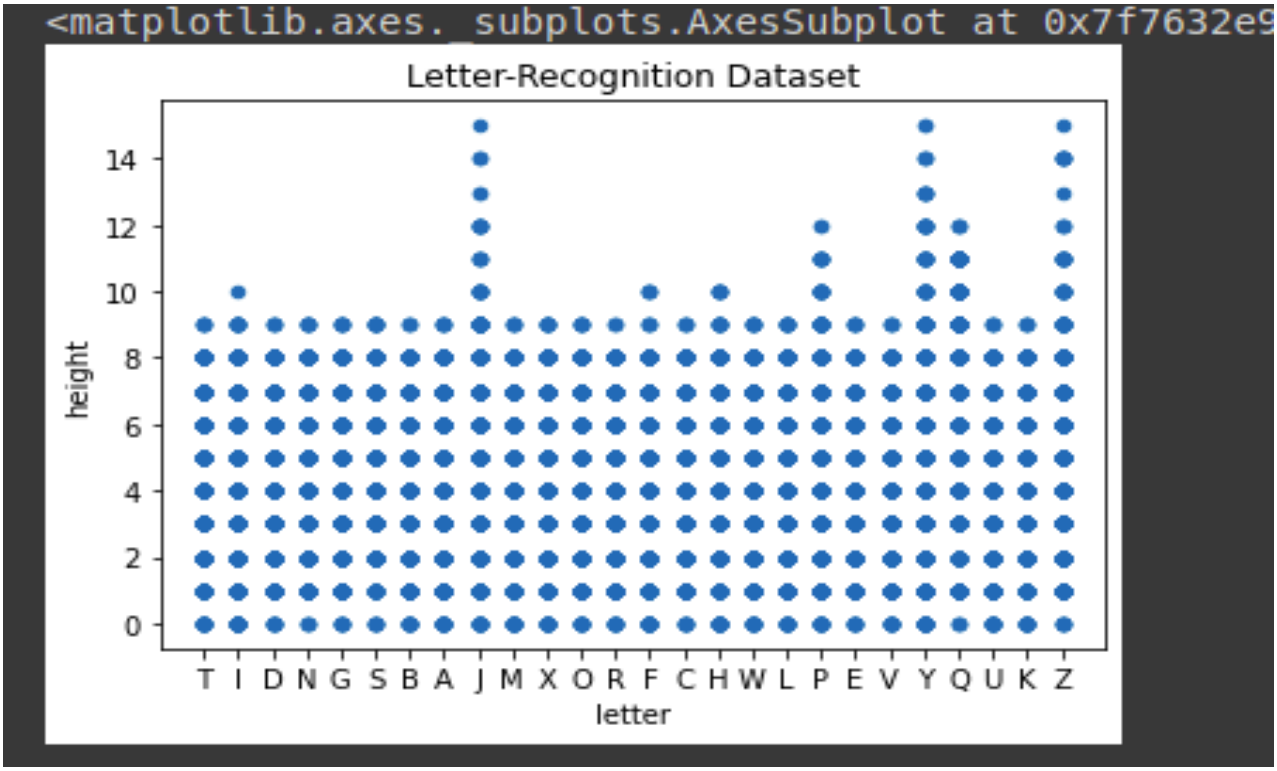
Assignee	Ahmed Osama	Youssif Assem Mondy	Marian Kromel	Seif Nagi	Philip George
Task title	Implementing KNN	Implementing MLP	Implementing Naive Bayes	Implementing SVM	Implementing Random Forest
Task Effort	1 Day	1 Day	1 Day	1 Day	2 Days
Task Status	Finished 8/1/2022	Finished 5/1/2022	Finished 8/1/2022	Finished 4/1/2022	Finished 4/1/2022, 8/1/2022
Per week time	2 hrs	2 hrs	3 hrs	2 hrs	1:30 hr
Task title	Documentation Experimental results	Documentation Introduction, Experimental results, references	Document Experimental results	Documentation DataSet, Experimental result	Documentation Experimental results
Task Effort	1 Day	1 Day	1 Day	1 Day	1 Day
Task Status	Finished 8/1/2022	Finished 5/1/2022	Finished 8/1/2022	Finished 8/1/2022	Finished 8/1/2022
Per week time	1 hr	2 hrs	1 hr	1 hr	1 hr
Task title		Data Visualization, Presentation, abstract			
Task Effort		1 Day			
Task Status		Finished 8/1/2022			
Per week time		2 hrs			

3 Experimental Results

MLP

Classifier	Activation Function	Optimization	No Of Iterations	No Of Hidden layers	No Of preceptrons	Accuracy	Precision	Recall
MLP	Relu	sgd	2000	3	100	0.95	0.95	0.95
MLP	Relu	adam	2000	3	250	0.96	0.96	0.96
MLP	logistic	adam	2000	3	200	0.96	0.96	0.96

Visualization of the data



```
1 modelSgd = MLPClassifier(hidden_layer_sizes=(100,100,100), activation='relu', solver='sgd', max_iter=2000)
2 modelSgd.fit(X_train, y_train)
3 print(classification_report(y_test, modelSgd.predict(X_test)))
```

	precision	recall	f1-score	support
A	0.97	0.98	0.97	217
B	0.91	0.88	0.90	241
C	0.95	0.96	0.96	217
D	0.90	0.95	0.93	230
E	0.94	0.92	0.93	225
F	0.94	0.94	0.94	251
G	0.91	0.95	0.93	235
H	0.92	0.93	0.92	217
I	0.95	0.94	0.94	217
J	0.96	0.95	0.95	215
K	0.94	0.96	0.95	219
L	0.98	0.95	0.97	252
M	0.97	0.99	0.98	214
N	0.97	0.95	0.96	234
O	0.91	0.94	0.92	211
P	0.96	0.96	0.96	258
Q	0.98	0.95	0.97	234
R	0.89	0.88	0.88	235
S	0.95	0.95	0.95	193
T	0.95	0.97	0.96	238
U	0.98	0.96	0.97	269
V	0.95	0.95	0.95	243
W	0.99	0.96	0.97	220
X	0.97	0.97	0.97	244
Y	0.97	0.96	0.96	240
Z	0.97	0.99	0.98	231
accuracy			0.95	6000
macro avg	0.95	0.95	0.95	6000
weighted avg	0.95	0.95	0.95	6000

```
1 modelAdam = MLPClassifier(hidden_layer_sizes=(200,200, 200), activation='logistic', solver='adam', max_iter=2000)
2 modelAdam.fit(X_train, y_train)
3 print(classification_report(y_test, modelAdam.predict(X_test)))
```

	precision	recall	f1-score	support
A	0.97	0.98	0.97	217
B	0.93	0.91	0.92	241
C	0.95	0.98	0.96	217
D	0.92	0.96	0.94	230
E	0.95	0.96	0.96	225
F	0.94	0.96	0.95	251
G	0.94	0.93	0.94	235
H	0.93	0.94	0.93	217
I	0.95	0.92	0.93	217
J	0.94	0.94	0.94	215
K	0.93	0.99	0.96	219
L	1.00	0.97	0.98	252
M	0.97	0.99	0.98	214
N	0.97	0.98	0.97	234
O	0.97	0.92	0.95	211
P	0.99	0.90	0.95	258
Q	0.99	0.93	0.96	234
R	0.88	0.95	0.91	235
S	0.95	0.99	0.97	193
T	0.95	0.98	0.96	238
U	0.99	0.96	0.97	269
V	0.96	0.98	0.97	243
W	1.00	0.96	0.98	220
X	0.97	0.98	0.98	244
Y	1.00	0.97	0.98	240
Z	0.99	0.98	0.98	231
accuracy			0.96	6000
macro avg	0.96	0.96	0.96	6000
weighted avg	0.96	0.96	0.96	6000

```
modelAdam = MLPClassifier(hidden_layer_sizes=(250,250, 250), activation='relu', solver='adam', max_iter=2000)
modelAdam.fit(X_train, y_train)
print(classification_report(y_test, modelAdam.predict(X_test)))
```

```
              precision    recall  f1-score   support

           A       0.99      0.98      0.99       217
           B       0.93      0.93      0.93       241
           C       0.95      0.98      0.96       217
           D       0.98      0.91      0.94       230
           E       0.91      0.97      0.94       225
           F       0.97      0.97      0.97       251
           G       0.95      0.95      0.95       235
           H       0.90      0.94      0.92       217
           I       0.95      0.94      0.94       217
           J       0.94      0.94      0.94       215
           K       0.96      0.96      0.96       219
           L       0.99      0.96      0.98       252
           M       0.97      0.99      0.98       214
           N       1.00      0.96      0.98       234
           O       0.91      0.98      0.94       211
           P       0.94      0.97      0.95       258
           Q       0.98      0.95      0.97       234
           R       0.92      0.91      0.91       235
           S       0.96      0.95      0.96       193
           T       0.97      0.97      0.97       238
           U       0.99      0.97      0.98       269
           V       0.94      0.98      0.96       243
           W       0.99      0.97      0.98       220
           X       0.99      0.97      0.98       244
           Y       0.98      0.97      0.98       240
           Z       0.99      0.99      0.99       231

    accuracy                           0.96      6000
   macro avg       0.96      0.96      0.96      6000
weighted avg       0.96      0.96      0.96      6000
```

KNN

Classifier	No of neighbors	Accuracy	Precision	Recall	F1- score
KNN	10	0.94%	0.94	0.94	0.94
KNN	4	0.95%	0.95	0.95	0.95
KNN	2	0.95%	0.95	0.95	0.95
KNN	7	0.95%	0.95	0.95	0.95

```
In [33]: 1 print(classification_report(Y_pred,Y_test))
         2
```

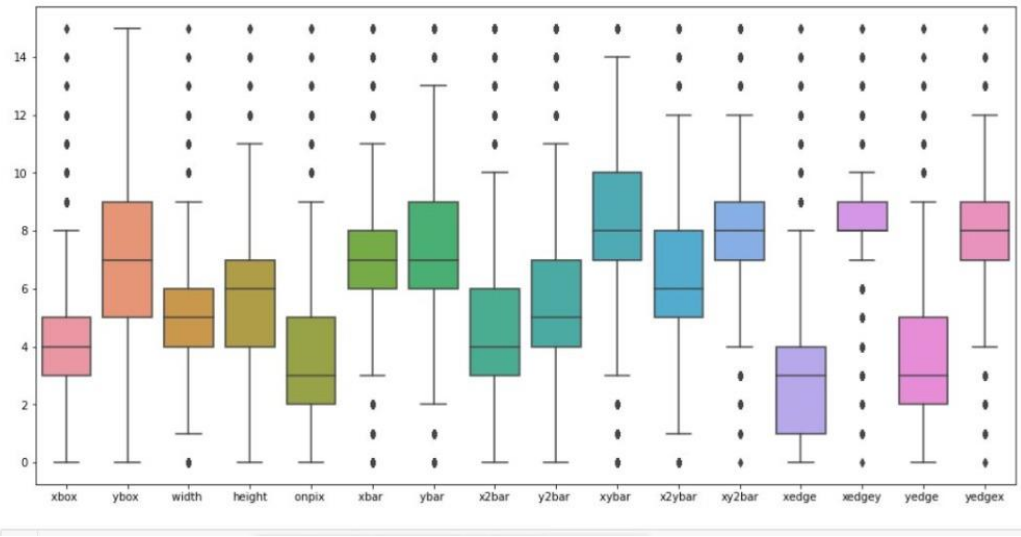
```
              precision    recall  f1-score   support

           A       0.99      0.99      0.99       217
           B       0.96      0.88      0.92       210
           C       0.94      0.96      0.95       190
           D       0.97      0.86      0.91       236
           E       0.96      0.91      0.93       191
           F       0.92      0.93      0.93       193
           G       0.93      0.94      0.93       186
           H       0.85      0.84      0.84       177
           I       0.95      0.97      0.96       180
           J       0.95      0.96      0.96       176
           K       0.83      0.93      0.87       164
           L       0.98      0.98      0.98       169
           M       0.97      0.97      0.97       196
           N       0.92      0.98      0.95       180
           O       0.94      0.86      0.90       206
           P       0.93      0.98      0.96       197
           Q       0.92      0.96      0.94       192
           R       0.92      0.90      0.91       207
           S       0.94      0.96      0.95       182
           T       0.97      0.93      0.95       179
           U       0.98      0.98      0.98       214
           V       0.95      0.96      0.95       183
           W       0.95      0.98      0.97       162
           X       0.93      0.95      0.94       201
           Y       0.96      0.99      0.97       213
           Z       0.98      0.97      0.98       199

    accuracy                           0.94      5000
   macro avg       0.94      0.94      0.94      5000
weighted avg       0.94      0.94      0.94      5000
```

Visualization of the data

```
In [11]:
# Checking for Outliers in numerical data
plt.figure(figsize=[16,8])
sb.boxplot(data = df)
plt.show()
```



Random Forest Classifier

Classifier	criterion	n_estimators	Accuracy	Precision	Recall	F1- score
Random Forest	entropy	200	96.37%	0.96	0.96	0.964
Random Forest	entropy	250	95.75%	0.96	0.96	0.958
Random Forest	gini	200	95.67%	0.96	0.96	0.957
Random Forest	gini	250	95.95%	0.96	0.96	0.96

References: https://www.kaggle.com/rahulvv/nb-and-rf-models-99-accuracy

&

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Screenshots:

In [11]:
```
# Model evaluation
rfc=RandomForestClassifier(n_estimators=200,criterion='entropy',random_state=0,min_samples_split=2)
print(classification_report(y_test, predictions))
print(confusion_matrix(y_test, predictions))
```

```
              precision    recall  f1-score   support

           A       0.99      0.99      0.99       239
           B       0.94      0.96      0.95       237
           C       0.99      0.95      0.97       234
           D       0.95      0.97      0.96       250
           E       0.96      0.95      0.96       232
           F       0.96      0.97      0.96       233
           G       0.93      0.96      0.95       245
           H       0.90      0.91      0.90       200
           I       0.95      0.93      0.94       226
           J       0.96      0.92      0.94       201
           K       0.96      0.92      0.94       245
           L       1.00      0.96      0.98       230
           M       1.00      0.98      0.99       241
           N       0.96      0.98      0.97       227
           O       0.93      0.95      0.94       210
           P       0.97      0.96      0.97       234
           Q       0.97      0.97      0.97       238
           R       0.94      0.96      0.95       233
           S       0.97      0.97      0.97       227
           T       0.98      0.98      0.98       215
           U       0.98      0.98      0.98       247
           V       0.96      0.96      0.96       219
           W       0.97      1.00      0.98       203
           X       0.97      0.99      0.98       236
           Y       0.98      0.98      0.98       252
           Z       0.99      0.98      0.99       246

    accuracy                           0.96      6000
   macro avg       0.96      0.96      0.96      6000
weighted avg       0.96      0.96      0.96      6000
```

In [12]:
```
rfc_f1 = round(f1_score(y_test, predictions, average= 'weighted'), 3)
rfc_accuracy = round((accuracy_score(y_test, predictions) * 100), 2)

print("Accuracy : " , rfc_accuracy , " %")
print("f1_score : " , rfc_f1)
```

```
Accuracy :  96.37  %
f1_score :  0.964
```

In [11]:
```
# Model evaluation
rfc=RandomForestClassifier(n_estimators=250,criterion='entropy',random_state=0,min_samples_split=2)
print(classification_report(y_test, predictions))
print(confusion_matrix(y_test, predictions))
```

```
              precision    recall  f1-score   support

           A       0.98      1.00      0.99       226
           B       0.87      0.95      0.91       233
           C       0.98      0.98      0.98       211
           D       0.94      0.97      0.95       247
           E       0.93      0.95      0.94       232
           F       0.95      0.94      0.95       238
           G       0.93      0.96      0.95       217
           H       0.97      0.94      0.96       204
           I       0.96      0.93      0.95       230
           J       0.97      0.92      0.94       234
           K       0.96      0.97      0.96       214
           L       0.99      0.96      0.98       248
           M       0.97      1.00      0.98       229
           N       0.97      0.95      0.96       226
           O       0.94      0.96      0.95       233
           P       0.96      0.94      0.95       227
           Q       0.97      0.96      0.97       249
           R       0.94      0.94      0.94       229
           S       0.98      0.96      0.97       217
           T       0.99      0.97      0.98       249
           U       0.96      0.99      0.97       223
           V       0.97      0.95      0.96       235
           W       0.99      0.96      0.97       252
           X       0.96      0.98      0.97       243
           Y       1.00      0.98      0.99       234
           Z       1.00      0.98      0.99       220

    accuracy                           0.96      6000
   macro avg       0.96      0.96      0.96      6000
weighted avg       0.96      0.96      0.96      6000
```

In [12]:
```
rfc_f1 = round(f1_score(y_test, predictions, average= 'weighted'), 3)
rfc_accuracy = round((accuracy_score(y_test, predictions) * 100), 2)

print("Accuracy : " , rfc_accuracy , " %")
print("f1_score : " , rfc_f1)
```

```
Accuracy :  95.75  %
f1_score :  0.958
```

In [11]:
```
# Model evaluation
rfc=RandomForestClassifier(n_estimators=200,criterion='gini',random_state=0,min_samples_split=2)
print(classification_report(y_test, predictions))
print(confusion_matrix(y_test, predictions))
```

```
              precision    recall  f1-score   support

           A       0.98      1.00      0.99       238
           B       0.87      0.96      0.92       227
           C       0.97      0.97      0.97       229
           D       0.89      0.95      0.92       246
           E       0.94      0.95      0.95       218
           F       0.96      0.93      0.94       241
           G       0.94      0.95      0.95       220
           H       0.98      0.89      0.93       229
           I       0.97      0.94      0.95       240
           J       0.95      0.93      0.94       220
           K       0.94      0.96      0.95       230
           L       0.99      0.95      0.97       209
           M       0.96      0.98      0.97       223
           N       0.99      0.95      0.97       251
           O       0.95      0.91      0.93       248
           P       0.95      0.96      0.95       207
           Q       0.95      0.97      0.96       252
           R       0.90      0.95      0.93       218
           S       0.97      0.98      0.97       232
           T       0.96      0.99      0.97       254
           U       0.97      0.97      0.97       236
           V       0.98      0.91      0.95       234
           W       0.96      0.99      0.98       191
           X       0.99      0.96      0.98       236
           Y       0.99      0.98      0.99       238
           Z       0.98      0.98      0.98       233

    accuracy                           0.96      6000
   macro avg       0.96      0.96      0.96      6000
weighted avg       0.96      0.96      0.96      6000
```

In [12]:
```
rfc_f1 = round(f1_score(y_test, predictions, average= 'weighted'), 3)
rfc_accuracy = round((accuracy_score(y_test, predictions) * 100), 2)

print("Accuracy : " , rfc_accuracy , " %")
print("f1_score : " , rfc_f1)
```

```
Accuracy :  95.67  %
f1_score :  0.957
```

In [11]:
```
# Model evaluation
rfc=RandomForestClassifier(n_estimators=250,criterion='gini',random_state=0,min_samples_split=2)
print(classification_report(y_test, predictions))
print(confusion_matrix(y_test, predictions))
```

```
              precision    recall  f1-score   support

           A       0.99      0.99      0.99       247
           B       0.88      0.95      0.92       217
           C       0.99      0.96      0.97       207
           D       0.90      0.95      0.92       232
           E       0.98      0.96      0.97       250
           F       0.96      0.96      0.96       228
           G       0.94      0.96      0.95       237
           H       0.93      0.89      0.91       207
           I       0.95      0.96      0.95       215
           J       0.98      0.94      0.96       238
           K       0.94      0.93      0.93       199
           L       1.00      0.97      0.98       230
           M       0.95      0.97      0.96       235
           N       0.99      0.94      0.96       234
           O       0.94      0.96      0.95       251
           P       0.97      0.96      0.96       238
           Q       0.96      0.93      0.95       227
           R       0.92      0.96      0.94       224
           S       0.97      0.97      0.97       212
           T       0.97      0.98      0.97       253
           U       0.97      0.99      0.98       240
           V       0.97      0.94      0.95       225
           W       0.96      0.99      0.98       241
           X       0.97      0.98      0.98       263
           Y       0.99      0.95      0.97       240
           Z       1.00      0.98      0.99       210

    accuracy                           0.96      6000
   macro avg       0.96      0.96      0.96      6000
weighted avg       0.96      0.96      0.96      6000
```

In [12]:
```
rfc_f1 = round(f1_score(y_test, predictions, average= 'weighted'), 3)
rfc_accuracy = round((accuracy_score(y_test, predictions) * 100), 2)

print("Accuracy : " , rfc_accuracy , " %")
print("f1_score : " , rfc_f1)
```

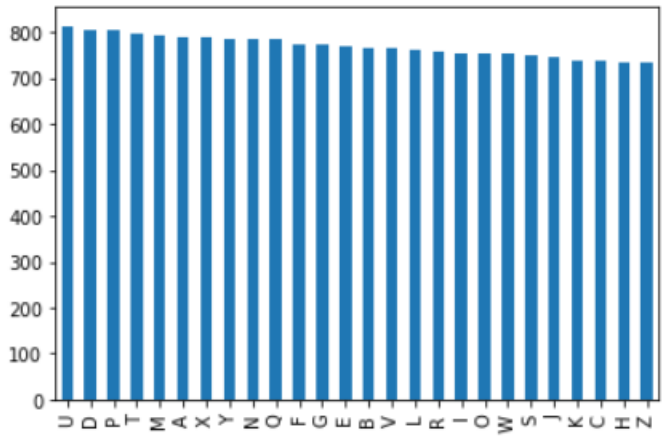
```
Accuracy :  95.95  %
f1_score :  0.96
```

Visualization

Visualization

In [7]:
```
letter['letter'].value_counts().plot.bar()
plt.plot()
```

Out[7]: []

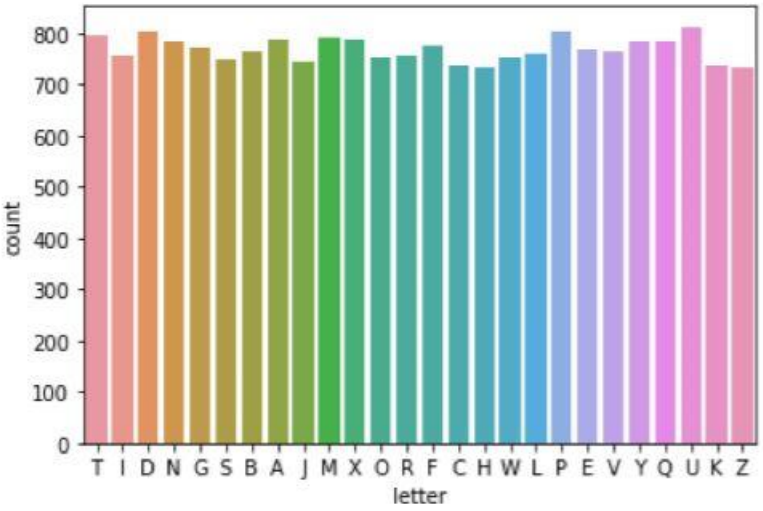


SVM

Classifier	Kernel	Sensitivity	Accuracy	Precision	Recall	Specificity
SVM	Sigmoid	0.1051	93.12%	-	0.1051	0.9643
SVM	Linear	0.8102	98.54%	0.8184	0.8102	0.9924
SVM	polynomial	0.9457	99.58%	0.9461	0.9457	0.9978
SVM	rbf	0.929	99.45%	0.9305	0.929	0.9972

Screenshots:

Visualization of the dataset



Linear Kernel

```
Overall Performance Prediction for Linear kernel
Sensitivity: 0.8102
Specificity: 0.9924
Accuracy: 0.9854
Balanced Accuracy: 0.9013
Recall :0.8102
Precision: 0.8184
```

Sigmoid Kernel Output

```
Overall Performance Prediction for Sigmoid Kernel
Sensitivity: 0.1051
Specificity: 0.9643
Accuracy: 0.9312
Balanced Accuracy: 0.5346
Recall :0.1051
Precision: nan
```

Rbf Kernel Output

```
Overall Performance Prediction of rbf kernel
Sensitivity: 0.929
Specificity: 0.9972
Accuracy: 0.9945
Balanced Accuracy: 0.9631
Recall :0.929
Precision: 0.9305
```

Polynomial Kernel Output

```
Overall Performance Prediction for polynomial kernal
Sensitivity: 0.9457
Specificity: 0.9978
Accuracy: 0.9958
Balanced Accuracy: 0.9718
Recall :0.9457
Precision: 0.9461
```

NAÏVE BAYES

Classifier	criterion	N_estimators	Accuracy	Precision	Recall	F1- score
NB	GaussianNB	2000	65 %	0.66	0.65	0.65
NB	multinomial	2000	55%	0.55	0.55	0.53
NB	Bernoulli	2000	11%	0.17	0.11	0.09

	precision	recall	f1-score	support
A	0.82	0.87	0.85	180
B	0.46	0.65	0.54	206
C	0.74	0.78	0.76	196
D	0.61	0.70	0.65	212
E	0.60	0.45	0.52	186
F	0.69	0.72	0.71	201
G	0.61	0.51	0.56	221
H	0.52	0.35	0.42	156
I	0.52	0.76	0.62	181
J	0.78	0.73	0.76	196
K	0.46	0.49	0.48	174
L	0.99	0.77	0.87	189
M	0.69	0.90	0.78	210
N	0.81	0.68	0.74	171
O	0.60	0.72	0.65	191
P	0.86	0.70	0.77	208
Q	0.51	0.57	0.53	175
R	0.56	0.63	0.60	193
S	0.35	0.28	0.31	193
T	0.75	0.69	0.72	209
U	0.90	0.72	0.80	201
V	0.73	0.83	0.78	200
W	0.69	0.80	0.74	182
X	0.48	0.54	0.51	202
Y	0.75	0.38	0.51	188
Z	0.73	0.60	0.66	179
accuracy			0.65	5000
macro avg	0.66	0.65	0.65	5000
weighted avg	0.66	0.65	0.65	5000

```
y_pred  =  classifier.predict(x_test)
from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(y_test, y_pred)
ac = accuracy_score(y_test,y_pred)
print("Accuracy:" , ac)
```

```
Accuracy: 0.651
```

```
y_pred =clf.predict(X_test)
acc_score = accuracy_score(y_test, y_pred)
acc_score = round((accuracy_score(y_test, y_pred) * 100))
from sklearn.metrics import confusion_matrix,accuracy_score
print("Accuracy : " , acc_score , "%")
```

```
Accuracy :  55 %
```

```
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

	precision	recall	f1-score	support
A	0.67	0.89	0.76	180
B	0.35	0.51	0.41	206
C	0.40	0.73	0.52	196
D	0.60	0.58	0.59	212
E	0.50	0.29	0.37	186
F	0.60	0.69	0.64	201
G	0.36	0.23	0.28	221
H	0.18	0.03	0.05	156
I	0.71	0.75	0.73	181
J	0.75	0.61	0.67	196
K	0.22	0.26	0.24	174
L	0.85	0.63	0.73	189
M	0.70	0.70	0.70	210
N	0.53	0.64	0.58	171
O	0.57	0.63	0.60	191
P	0.80	0.60	0.68	208
Q	0.49	0.63	0.55	175
R	0.34	0.44	0.38	193
S	0.32	0.22	0.26	193
T	0.64	0.63	0.63	209
U	0.78	0.70	0.74	201
V	0.69	0.84	0.76	200
W	0.82	0.80	0.81	182
X	0.31	0.41	0.35	202
Y	0.50	0.08	0.14	188
Z	0.65	0.66	0.66	179
accuracy			0.55	5000
macro avg	0.55	0.55	0.53	5000
weighted avg	0.55	0.55	0.54	5000

```
y_pred =clf.predict(X_test)
acc_score = accuracy_score(y_test, y_pred)
acc_score = round((accuracy_score(y_test, y_pred) * 100))
from sklearn.metrics import confusion_matrix,accuracy_score
print("Accuracy : " , acc_score , "%")
```

```
Accuracy :  11 %
```

```
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
```

	precision	recall	f1-score	support
A	0.77	0.20	0.32	180
B	0.00	0.00	0.00	206
C	0.00	0.00	0.00	196
D	0.05	0.97	0.10	212
E	0.00	0.00	0.00	186
F	0.00	0.00	0.00	201
G	0.00	0.00	0.00	221
H	0.00	0.00	0.00	156
I	0.55	0.60	0.57	181
J	0.59	0.18	0.27	196
K	0.00	0.00	0.00	174
L	0.89	0.26	0.41	189
M	0.67	0.01	0.02	210
N	0.16	0.35	0.22	171
O	0.00	0.00	0.00	191
P	0.00	0.00	0.00	208
Q	0.00	0.00	0.00	175
R	0.32	0.04	0.07	193
S	0.00	0.00	0.00	193
T	0.00	0.00	0.00	209
U	0.00	0.00	0.00	201
V	0.00	0.00	0.00	200
W	0.21	0.03	0.06	182
X	0.00	0.00	0.00	202
Y	0.19	0.07	0.11	188
Z	0.13	0.21	0.16	179
accuracy			0.11	5000
macro avg	0.17	0.11	0.09	5000
weighted avg	0.17	0.11	0.09	5000

References

https://www.kaggle.com/nishan192/letterrecognition-using-svm