IE 7300 : Statistical Learning Project Proposal

Classification and Recognition project

By Shucheng zhang

9793078609

Zhang.shuche@northeastern.edu

Percentage of effort:	100%
Signature of students:	Shucheng zhang
Submission date:	12/14/2023

1.Introduction

During the fast developed era, the ability of machines to understand and interpret written text has been a trend in the field of Data Science. The process of converting images of type or handwritten characters into machine-encoded text, known as the Optical Character Recognition, has found its utility in multitude applications. It belongs to the field of Pattern Recognition and Computer Vision as it can range from digitizing printed books, Virtual Reality, to Auto-Driven Vehicle Detection system. Nevertheless, the efficiency of Recognition has usually derived by the underlying algorithms that recognize individual characters. Thus, the Letter Recognition dataset from the UCI Machine Learning Repository provides an ideal platform to study, analyze, and improve such algorithms. As of starting of this project, the goal is to develop a proper classification machine learning algorithms with the assist of Neural Nets to drive the outcomes of this computer vision project and verify the possibility.

2.Problem Statement

Based on the original paper, there is a set of 20,000 unique letter images of the 26 uppercase letters from multiple various fonts including script, serif, Gothic, Italic, generated by randomly distorting pixel images. The primary objective of this project is to develop a classification model capable of accurately identifying each of the letters in the English alphabet based on specific features extracted from a collection of typewritten characters. The 16 numeric attributes are collected and summarized to statistical moments and edge counts which were then scaled to fit into a range of integer values from 0 through 15. Based on the successful model will not only demonstrate an understanding of the inherent patterns within the dataset but can also serve as a foundational block for larger OCR systems in real-world applications.

3.Dataset introduction

The Letter Recognition dataset has been extracting a set of 16 attribute-type features from each character. The dataset provides a comprehensive breakdown of these features, which applied a combination of statistical and geometrical properties. These features included:

- **x-box:** horizontal position of box
- **y-box:** vertical position of box
- **width:** width of box
- **high:** height of box
- **onpix:** total number on pixels
- **x-bar:** mean x of on pixels in box
- **y-bar:** mean y of on pixels in box
- **x2bar:** mean x variance
- **y2bar:** mean y variance
- **xybar:** mean x y correlation
- **x2ybar:** mean of x*x*y
- **xy2bar:** mean of x*y*y
- **x-ege:** mean edge count left to right
- **xegvy:** correlation of x-ege with y
- **y-ege:** mean edge count bottom to top
- **yegvx:** correlation of y-ege with x

Some of the key values are listed below that the **Number of Instances** are20000 samples, the **Number of Attributes** are 16 integer-valued features, the **Attribute Characteristics** are summarized to statistical and geometrical properties, no **Missing Values** are detected. The dataset is uniformly divided among the 26 letters of the alphabet, ensuring a balanced distribution and thereby eliminating potential biases in classification.

4.Analysis

- Descriptive statistics

  The distribution and description of the letter dataset has been generated. Below are one of the basic visualization of the features distribution of the dataset.

	x-box	y-box	width	high	onpix
count	20000.000000	20000.000000	20000.000000	20000.00000	20000.000000
mean	4.023550	7.035500	5.121850	5.37245	3.505850
std	1.913212	3.304555	2.014573	2.26139	2.190458
min	0.000000	0.000000	0.000000	0.00000	0.000000
25%	3.000000	5.000000	4.000000	4.00000	2.000000
50%	4.000000	7.000000	5.000000	6.00000	3.000000
75%	5.000000	9.000000	6.000000	7.00000	5.000000
max	15.000000	15.000000	15.000000	15.00000	15.000000

x-bar	y-bar	x2bar	y2bar	xybar	x2ybr
20000.000000	20000.000000	20000.000000	20000.000000	20000.000000	20000.00000
6.897600	7.500450	4.628600	5.178650	8.282050	6.45400
2.026035	2.325354	2.699968	2.380823	2.488475	2.63107
0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
6.000000	6.000000	3.000000	4.000000	7.000000	5.00000
7.000000	7.000000	4.000000	5.000000	8.000000	6.00000
8.000000	9.000000	6.000000	7.000000	10.000000	8.00000
15.000000	15.000000	15.000000	15.000000	15.000000	15.00000

xy2br	x-ege	xegvy	y-ege	yegvx
20000.000000	20000.000000	20000.000000	20000.000000	20000.00000
7.929000	3.046100	8.338850	3.691750	7.80120
2.080619	2.332541	1.546722	2.567073	1.61747
0.000000	0.000000	0.000000	0.000000	0.00000
7.000000	1.000000	8.000000	2.000000	7.00000
8.000000	3.000000	8.000000	3.000000	8.00000
9.000000	4.000000	9.000000	5.000000	9.00000
15.000000	15.000000	15.000000	15.000000	15.00000

- Explanatory Data Analysis

  Multiple functions and methods have been implemented in the part of EDA, including dataset preprocessing, dataset checking, analysis of missing values, distribution visualization and verification, data relationship visualization, dataset visualization in scatterplots and boxplots, descriptive data statistic, and correlation coefficients analysis with heatmap.

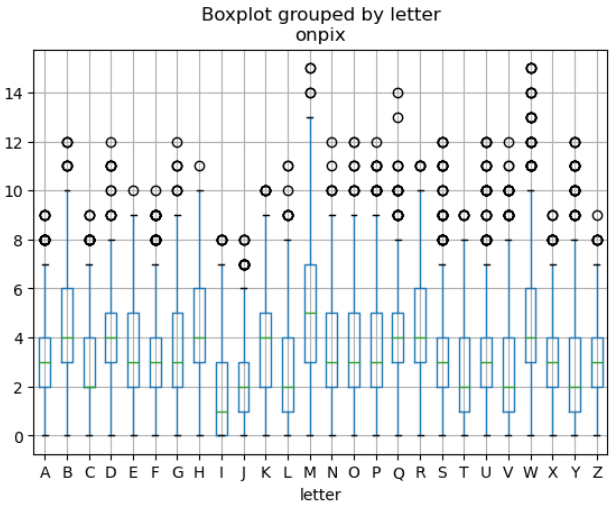
  For missing values, the quality of this letter dataset has been verified with no missing values detected. Below are some of the information addressing.

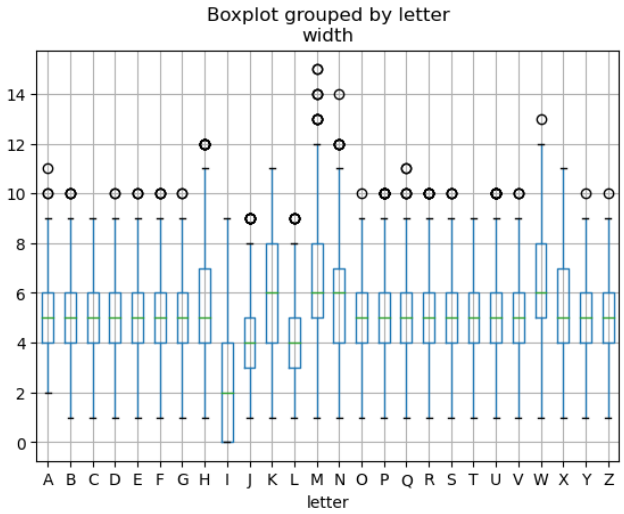
```
letter    0
x-box     0
y-box     0
width     0
high      0
onpix     0
x-bar     0
y-bar     0
x2bar     0
y2bar     0
xybar     0
x2ybr     0
xy2br     0
x-ege     0
xegvy     0
y-ege     0
yegvx     0
dtype: int64
```

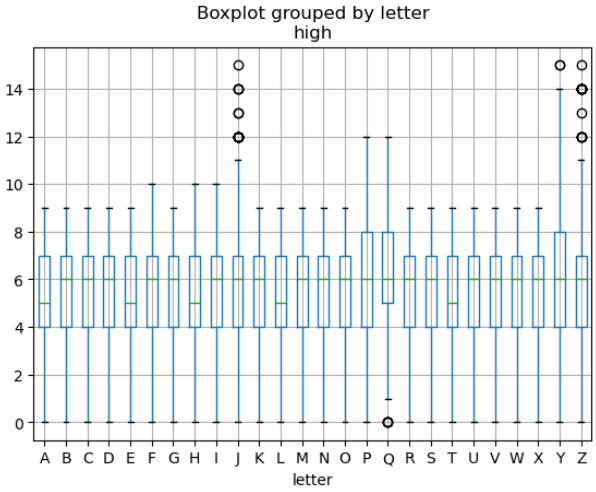
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 17 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   letter  20000 non-null  object
 1   x-box   20000 non-null  int64
 2   y-box   20000 non-null  int64
 3   width   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
```

- `letter`: capital letter. As the target variable. (26 values from A to Z)
- `x-box`: horizontal position of box. (integer)
- `y-box`: vertical position of box. (integer)
- `width`: width of box. (integer)
- `high`: height of box. (integer)
- `onpix`: total # 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)

Distribution of some of the important features are also generated and will be shown below.



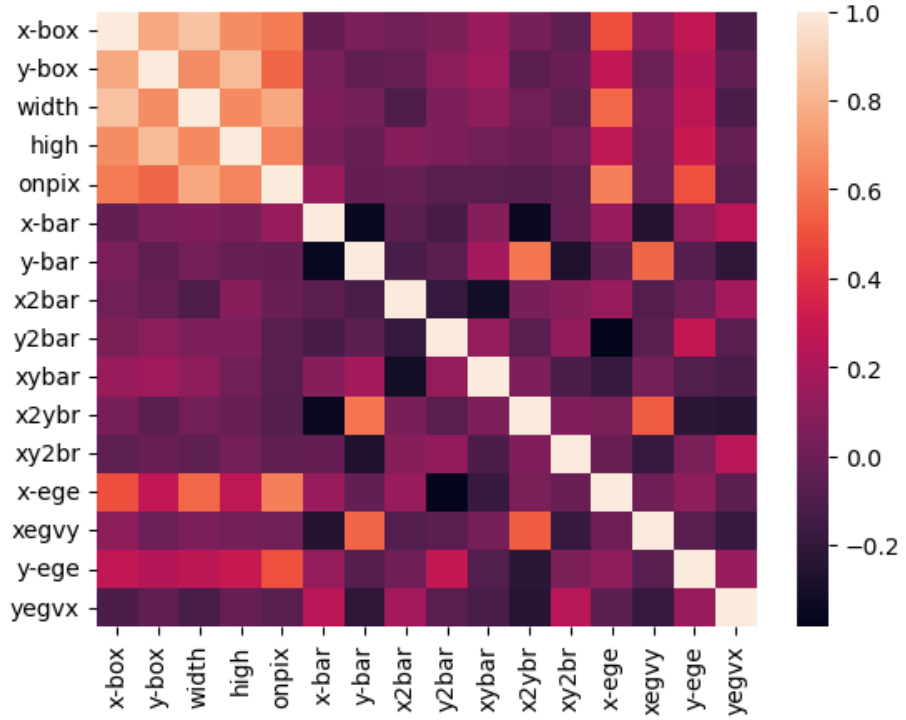




The three boxplots showed the basic data distribution, outliers, and the relationship between each letter's onpix values, width values, and height values.

Which provided a better view of the dataset when performing feature selection in the next step.

Moving to the correlation analysis part, it consists of both graphs and data explanations which are the heat map and corr values description. From the heat map, it is clear to find that for most of the features, there are no extreme correlations between each other with around 0 coefficient. Whereas some few features might appear with a little correlation with each other that concentrated on the top left side. Based on the heat map, it is safe to conclude that the dataset has small correlations with each other which is a high quality dataset for further classification.



	x-box	y-box	width	high	onpix	x-bar	y-bar	x2bar	y2bar	xybar	x2ybr	xy2br	x-ege	xegvy	y-ege	yegvx
x-box	1.000000	0.757793	0.851514	0.672764	0.619097	-0.032595	0.045545	0.014306	0.052086	0.148056	0.035464	-0.046333	0.489155	0.098180	0.273504	-0.105147
y-box	0.757793	1.000000	0.671912	0.823207	0.555067	0.045690	-0.040925	-0.025019	0.096478	0.159954	-0.054648	-0.007568	0.274431	-0.001336	0.230883	-0.042741
width	0.851514	0.671912	1.000000	0.660215	0.765716	0.061959	0.024832	-0.098611	0.057074	0.115018	0.011694	-0.045009	0.557251	0.045658	0.260285	-0.118273
high	0.672764	0.823207	0.660215	1.000000	0.644366	0.042844	-0.020072	0.082383	0.059032	0.012458	-0.011991	0.026386	0.265243	0.025359	0.297545	-0.018853
onpix	0.619097	0.555067	0.765716	0.644366	1.000000	0.139159	-0.028822	-0.011985	-0.065557	-0.069776	-0.072941	-0.038858	0.627507	0.017649	0.492653	-0.062969
x-bar	-0.032595	0.045690	0.061959	0.042844	0.139159	1.000000	-0.356580	-0.053306	-0.122851	0.085963	-0.341957	-0.032115	0.144325	-0.253339	0.127056	0.248816
y-bar	0.045545	-0.040925	0.024832	-0.020072	-0.028822	-0.356580	1.000000	-0.118625	-0.049658	0.178318	0.600397	-0.271649	-0.036722	0.555060	-0.078008	-0.207900
x2bar	0.014306	-0.025019	-0.098611	0.082383	-0.011985	-0.053306	-0.118625	1.000000	-0.188431	-0.317780	0.042545	0.082020	0.142132	-0.084820	0.006546	0.182902
y2bar	0.052086	0.096478	0.057074	0.059032	-0.065557	-0.122851	-0.049658	-0.188431	1.000000	0.132000	-0.060116	0.119048	-0.384018	-0.052545	0.277540	-0.061335
xybar	0.148056	0.159954	0.115018	0.012458	-0.069776	0.085963	0.178318	-0.317780	0.132000	1.000000	0.057988	-0.106759	-0.175676	0.029419	-0.087019	-0.114223
x2ybr	0.035464	-0.054648	0.011694	-0.011991	-0.072941	-0.341957	0.600397	0.042545	-0.060116	0.057988	1.000000	0.063214	0.053566	0.527239	-0.226251	-0.236518
xy2br	-0.046333	-0.007568	-0.045009	0.026386	-0.038858	-0.032115	-0.271649	0.082020	0.119048	-0.106759	0.063214	1.000000	-0.008753	-0.184927	0.049695	0.245808
x-ege	0.489155	0.274431	0.557251	0.265243	0.627507	0.144325	-0.036722	0.142132	-0.384018	-0.175676	0.053566	-0.008753	1.000000	0.002849	0.108411	-0.049789
xegvy	0.098180	-0.001336	0.045658	0.025359	0.017649	-0.253339	0.555060	-0.084820	-0.052545	0.029419	0.527239	-0.184927	0.002849	1.000000	-0.064402	-0.187591
y-ege	0.273504	0.230883	0.260285	0.297545	0.492653	0.127056	-0.078008	0.006546	0.277540	-0.087019	-0.226251	0.049695	0.108411	-0.064402	1.000000	0.143588
yegvx	-0.105147	-0.042741	-0.118273	-0.018853	-0.062969	0.248816	-0.207900	0.182902	-0.061335	-0.114223	-0.236518	0.245808	-0.049789	-0.187591	0.143588	1.000000

Another part of the EDA is the univariate numeric analysis. Which consists of the visualization of the distribution of all the 16 features. With the generation of

histogram showing their distribution, a clear relationship and description can be drawn from the dataset.

Based on the plots, We can find that

- `onpix`, `x2bar`, `y2bar`, `x-ege`, `y-ege` are right skewed distributed
- `x-box`, `width`, `x-bar`, `y-bar`, `xybar`, `x2ybar`, `xy2bar`, `xegvy`, `yegvx` are quite normally distributed
- `y-box`, `high` are left skewed distributed

5.Model Training and Evaluation

- **K-Nearest Neighbors**

  The overall accuracy of 0.94 means the KNN model correctly predicts the class 94% of the time for the test dataset. Which is a great score but sometimes can be threatened by overfitting.

  Throughout the training and prediction process, conclusion can be drawn that KNN is simple to implement, no extra assumption about data is required and very effective if the dataset has sufficient representative samples. However, it can be computationally intensive, especially with large datasets. The model performance might be heavily dependent on the choice of 'K' and the distance metric. So based on the strength of this model, it is good for baseline modeling and when the data might not have clear boundaries.

```
Accuracy: 0.94
Precision: 0.94
Recall: 0.94
F1 Score: 0.94
```

- **Multiclass Logistic Regression (default)**

  The overall accuracy of 0.726 means that among the multiple MLR models, the default MLR model is most suitable for this classification task, which is a decent score.

  Throughout the training and prediction process, conclusions can be drawn that the MLR is easy to implement, provides probabilistic results, and efficient to train. However, it can be overfit on complex or high-dimensional datasets. It assumes linearity between dependent and independent variables. The MLR can be useful when the relationship between features and classes is expected to be approximately linear.

```
Logistic Regression Results:
Accuracy: 0.7260
```

- **Multiclass Logistic Regression (lasso)**

  The overall accuracy of 0.6105 means that among the multiple MLR models, the lasso regularization MLR model might not be the most suitable for this classification task, with a quite poor score.

  Throughout the training and prediction process, conclusions can be drawn that the Multiclass Logistic Regression (Lasso Regularization) adds regularization to reduce overfitting, can perform feature selection by shrinking coefficients for less important features to zero. However, bias increases for higher levels of regularization. It is suitable when some features might be irrelevant and need feature selection.

```
L1 (Lasso) Regularized Logistic Regression Results:
Accuracy: 0.6105
```

- **Multiclass Logistic Regression (ridge)**

  The overall accuracy of 0.7037 means that among the multiple MLR models, the ridge regularization MLR model might not be the most suitable for this classification task, with a quite decent score.

  Throughout the training and prediction process, conclusions can be drawn that the Multiclass Logistic Regression (Ridge Regularization) reduces overfitting, keeps all features but reduces magnitude of coefficients. But it didn't perform feature selection since it never fully removes any feature. Therefore, it will be useful in scenarios where all features contribute to the output but overfitting is a concern.

```
L2 (Ridge) Regularized Logistic Regression Results:
Accuracy: 0.7037
```

- **Multiclass Logistic Regression (elastic net)**

  The overall accuracy of 0.6583 means that among the multiple MLR models, the elastic net regularization MLR model might not be the most suitable for this classification task, with a quite decent score.

  Throughout the training and prediction process, conclusions can be drawn that the Multiclass Logistic Regression (Elastic Net Regularization) combines Lasso and Ridge, offering a balance between feature selection and coefficient shrinkage. But it can be computationally intensive, requires tuning of an additional hyperparameter to balance between Lasso and Ridge. Thus, it will be beneficial when there's a need for a middle ground between Ridge and Lasso.

```
Elastic Net Regularized Logistic Regression Results:
Accuracy: 0.6583
```

- **Decision Tree**

  The overall accuracy of 0.87325 means the Decision Tree model correctly predicts 87.325% of the correct classes for the test dataset. Which is a great score but sometimes can be threatened by overfitting. The result can still be improved with more splits and larger tree depths but will be more computationally intensive if no optimization is done.

  Throughout the training and prediction process, conclusion can be drawn that Decision Tree is easy to interpret and understand, can handle both numerical and categorical data well, and require little data preprocessing. However, the tree can be prone to overfitting, especially with deep trees. It can also be unstable as small changes in data might lead to a different tree. Therefore, it offers a clear decision-making structure, which is helpful for understanding feature importance.

```
Accuracy: 0.87325
Top important features: Index(['onpix', 'x2bar', 'y-bar', 'x-box', 'letter'], dtype='object')
```

- **Random Forest**

  The overall accuracy of 0.90775 means the Decision Tree model correctly predicts 90.775% of the correct classes for the test dataset. Which is also a great score but sometimes can be threatened by overfitting. Similar to the decision tree model, the result can still be improved with more estimators, as more trees and larger tree depths but will be more computationally intensive if no optimization is done.

  Throughout the training and prediction process, conclusion can be drawn that the Random Forest model can handle overfitting better than decision trees, work well with a large range of data items, and give estimates of feature importance. Whereas it appeared to be less interpretable than a single decision tree and can be slow to predict. Overall, it is useful for improving accuracy and reducing overfitting risk compared to a single decision tree.

```
Accuracy: 0.90775
Top important features: Index(['y-bar', 'y-box', 'onpix', 'x-box', 'letter'], dtype='object')
```

- **Support Vector Machine**

  The overall accuracy of 0.76 means the SVM model correctly predicts 76% of the correct classes for the test dataset. Which is a decent score. The result can still be improved with more hyperparameters tuning and fine-tuning, algorithm optimization process.

  Throughout the training and prediction process, conclusion can be drawn that the SVM will be effective in high-dimensional spaces, works well with clear

margin of separation, and can be tuned through the kernel. However, it will have poor performance with overlapping classes, not suitable for large datasets, and requires feature scaling. Thus, the SVM is effective for datasets where classes are separable, and dimensionality is high.

```
Overall Accuracy: 0.76
```

- **Neural Networks**
  Based on the overall tuned hyperparameters of 16 input layers, 26 of output layers, with 25 hidden layers, the overall accuracy lies between 84%. Which demonstrates quite a strong performance with relatively high efficiency. It is not computationally intensive compared to random forest, which takes 30 mins.

```
Classification Report:
 {'average_precision': 0.86720880263427, 'average_recall': 0.7767912216498529, 'average_f1-score': 0.8048488033011642}
Model with [16, 25, 26] layers, relu activation, sgd optimizer
Accuracy: 0.83825
```

6.Model Selection

- In summary, the selection of these models for letter recognition project offered a comprehensive approach to the problem, allowing for a balance between simplicity, interpretability, computational efficiency, and predictive performance. Each model can potentially provide unique insights, making them a robust choice for this classification model. Therefore, KNN model and the Neural Nets model generally performed better with less time and high accuracy would be recommended based on its robustness to noise and stable prediction results without overfitting and underfitting.

7.Conclusion

To sum up the project, it focused on developing a classification model for letter recognition using various machine learning algorithms. Compared with the original experiment, improvements can be made with the modern machine learning techniques like K-Nearest Neighbors, Multiclass Logistic Regression, Decision Trees, Random Forest, Support Vector Machine, and Neural Networks. Each model had its strengths and limitations in terms of accuracy, computational efficiency, and suitability for the task. The final model decisions of the project are concluded that while all models provided valuable insights, the KNN model and Neural Networks generally performed better, offering a balance between simplicity, interpretability, computational efficiency, and predictive performance. Hence, these models will be recommended for deployment with more hyperparameters tuning and optimization.

Reference and Citation

Slate,David. (1991). Letter Recognition. UCI Machine Learning Repository. https://doi.org/10.24432/C5ZP40.