Investigate the performance of machine learning classifiers in high dimension dataset via applying principle component analysis

Kerui Du

The University of Auckland

June 5, 2023

Email: kdu407@aucklanduni.ac.nz

- Dataset: **MNIST dataset**
  - It is a popular dataset used for handwritten digit recognition. It consists of a large collection of 28x28 pixel grayscale images of handwritten digits from 0 to 9.
- Packages:
  - **dslabs:** To load the MNIST dataset
  - **tidyverse:** A unified and consistent framework for data manipulation, exploration, and visualization, promoting a tidy data workflow.
  - **rpart:** One of the packages for **Decision tree**
  - **ranger:** One of the packages for **Random Forest**
  - **e1071:** One of the packages for **Naive Bayes**
  - **kknn:** One of the packages for **K-Nearest Neighbor**

Running Time

Since the demension of hand writting dataset is $\mathbf{70000} \times \mathbf{785}$, it is time-consuming to process such a dataset when generate a model in R. Then, only **12000** of these records has been picked.

Train vs Test

Split 12k data randomly into **10k** training data and **2k** testing data.

Metrics

Comparsion of time spend on the model generation and score(misclassfication rate) of the model before and after pca.

EDA

Preview of the predictor and response variable

Distribution

Each value in predictor variable has already been scaled in [1,255].
The proportion of each digit is approximately same.

```
## Rows: 12,000
## Columns: 10
## $ v678 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
## $ v128 <int> 99, 155, 54, 0, 0, 240, 0, 0, 0, 103, 0, 0, 0, 0, 0, 0, 0, 0, 31,…
## $ v508 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
## $ v470 <int> 0, 0, 193, 0, 0, 0, 0, 0, 0, 0, 162, 0, 0, 0, 70, 149, 0, 0, 0, 2…
## $ v298 <int> 178, 188, 0, 51, 0, 0, 117, 0, 0, 0, 142, 43, 0, 0, 254, 0, 0, 0,…
## $ v269 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 51, 174, 0, 168, 0, 0, 68, 9, 31, 2…
## $ v186 <int> 253, 253, 181, 254, 191, 0, 0, 161, 231, 210, 253, 0, 85, 0, 140,…
## $ v306 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 18, 0, 0, 0, 0, 0, 0, 0, 0, 0…
## $ v596 <int> 253, 0, 197, 0, 0, 0, 0, 135, 252, 252, 0, 0, 253, 93, 0, 0, 0, 0…
## $ v276 <int> 16, 0, 0, 0, 0, 0, 244, 0, 0, 0, 0, 0, 116, 60, 0, 0, 0, 0, 0, 0,…
```

```
##    0    1    2    3    4    5    6    7    8    9
## 1246 1342 1138 1226 1152 1096 1180 1277 1144 1199
```

PCA

Concept

Principal component analysis (PCA) allows us to summarize a set of features with a smaller number of representative features that collectively explain most of the variability in the original data set. PCA projects the observations described by d features into orthogonal, and thus by definition uncorrelated, variables. The new set of synthetic variables is equal in number to the original set. However, the first synthetic variable represents as much of the common variation of the original variables as possible, the second variable represents as much of the residual variation as possible, and so forth.[1]

PCA is particularly powerful in dealing with multicollinearity and variables that outnumber the samples ($d > n$), and it is widely used for explanatory data analysis, outlier detection and as a data pre-processing technique for predictive modelling.[1]

---

[1] Hartmann, K., Krois, J., Waske, B. 2018, p.12

PCA

Mathematical formula and Logical

When performing PCA, if the data has standardized, the pca would expect correlation matrix as the input, otherwise covariance matrix is used.

The mathematical expression for covariance matrix is

$$\rho_{(x,y)} = \frac{1}{n-1}\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})$$

and the mathematical expression for correlation matrix is

$$r_{(X,Y)} = \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}}$$

The eigenvector can be computed as

$$A\vec{v} = \lambda\vec{v}$$

Where A is the matrix, $\vec{v}$ is an eigenvector and λ is a scalar(eigenvalue)[2]

---

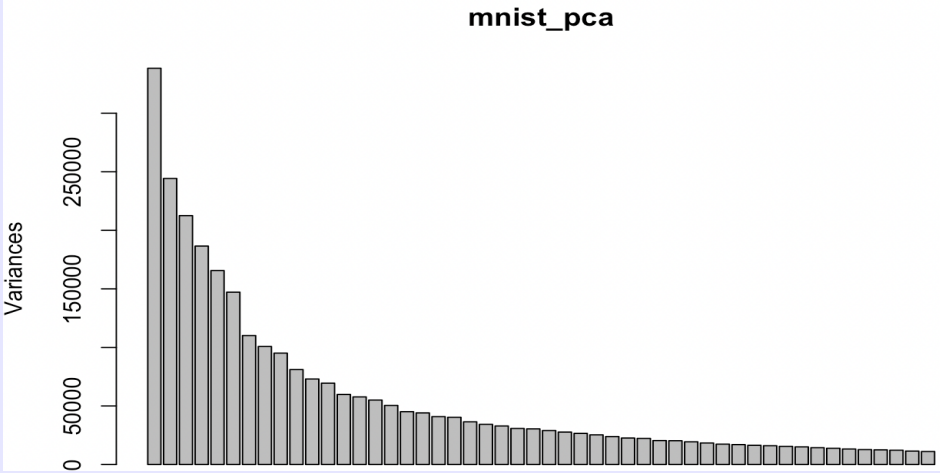
[2] Hartmann, K., Krois, J., Waske, B. 2018, p.14

Implementation

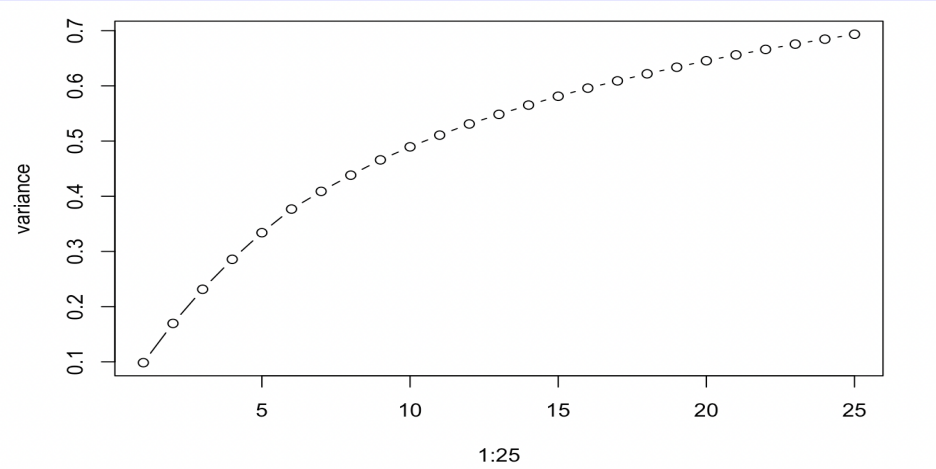
- In **prcomp**, the eigenvectors are defined by the **rotation**, also called principal components of the pca, which are the directions along which the data vary the most, note that, the sum of squared each principal component is 1.
- The eigenvalues or variances can be found in **sdev**, and used to determine the number of principal components.
- In order to gain the principal component scores, which correspond to the projection of the original data on the directions of the principal component, it has stored in **x** in prcomp.
- There are generally three ways for choosing the number of principal component:
  1. Elbow rule
  2. Kaiser rule
  3. Variance explained criteria rule

PCA

Number of principal component



Hard to determine the number of principal component via scree plot



The first 25 principal component explained around 70% original variability

PCA

Data transformation

As a consequence, the pca has refitted with **25** principal components. it basically means the $\mathbf{12000 \times 784}$ dataset has dimensional reduced to $\mathbf{12000 \times 25}$, the first principal component can roughly be expressed as

$$P_1 = \begin{pmatrix} \phi_{11} \\ \phi_{21} \\ \phi_{31} \\ \vdots \\ \phi_{n1} \end{pmatrix} = \begin{pmatrix} 3.75 \times 10^{-19} \\ 1.66 \times 10^{-16} \\ -5.55 \times 10^{-17} \\ \vdots \\ 0 \end{pmatrix}$$

Recall the definition that $\sum_{\mathbf{i=1}}^{\mathbf{n}} \phi_{\mathbf{n}}^{\mathbf{2}} = \mathbf{1}$.

To find out the first projected data, the approach is to apply the **'predict'** function, the formula behind that is

$$New_{11} = \phi_{11} \times V_{11} + \phi_{21} \times V_{12} + \phi_{31} \times V_{13} + \cdots + \phi_{n1} \times V_{1n}$$

Where n is number of variable and V_{1n} is first scaled(not in this case) and centered row of original data.

Machine Leaning Classifiers

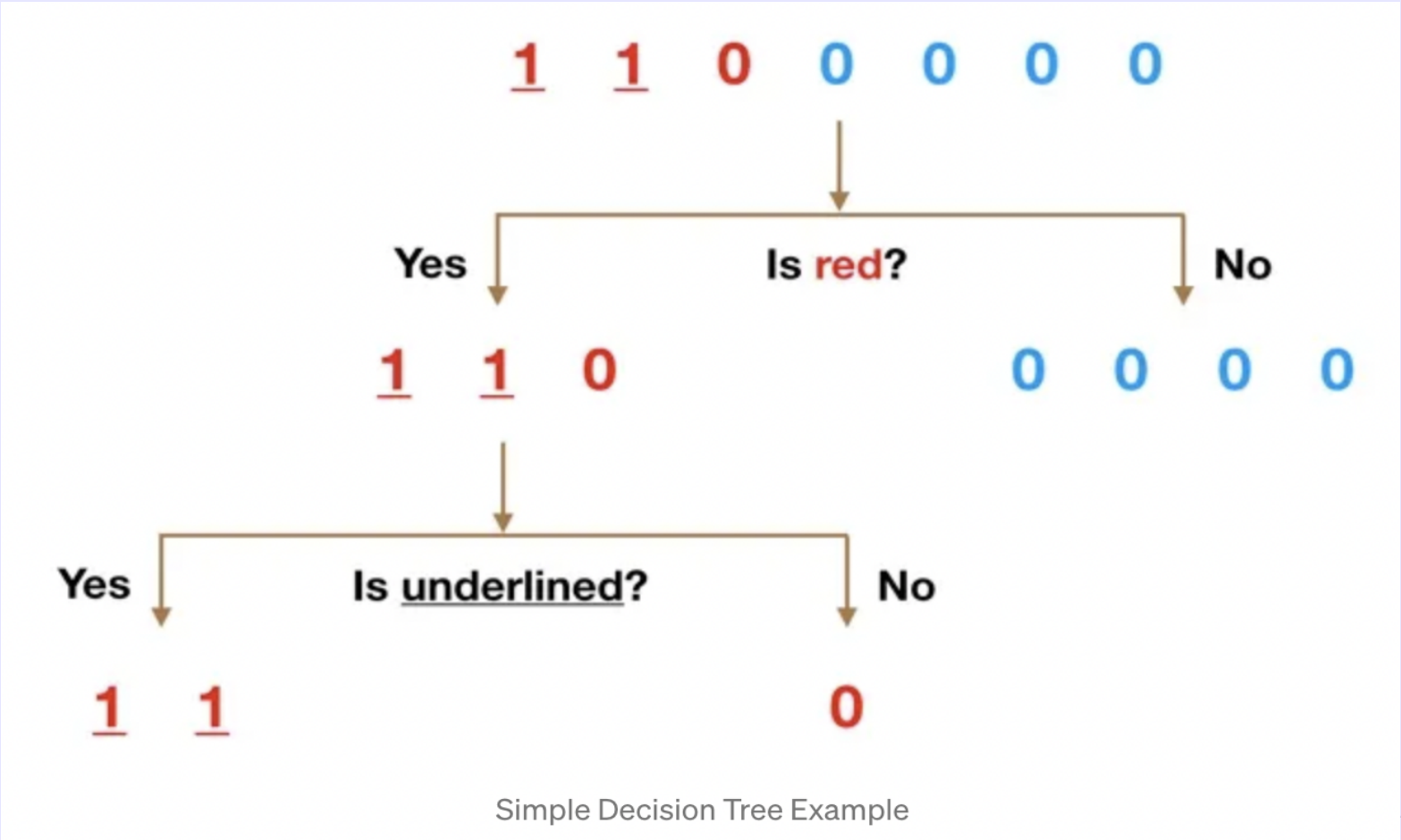
Decision Tree

Decision tree is a common machine learning algorithm that is commonly used for classification and regression analysis. It is a tree-like model used to represent a set of decisions and their possible consequences, including chance events and the likelihood of various outcomes.

The decision tree starts with a root node that represents the entire data set. The nodes of the tree are divided into multiple children based on specific characteristics and attributes, forming a series of branches. Each internal node contains a property test, and each leaf node represents a category or a numerical result. By moving around the tree, the input data set is classified to the correct leaf node based on its characteristic values. The goal of decision tree algorithm is to minimize the classification error rate or maximize the prediction accuracy by selecting the optimal attribute division.

Machine Leaning Classifiers

Decision Tree



Simple Decision Tree Example

[3]

[3] According to Yiu (2019), the graph titled "Simple Decision Tree Example" in the article

Machine Leaning Classifiers

Decision Tree

The package named **rpart** is one of the implementations for decision tree in R.

The function named **system.time** can be used to measure the elapsed time of model generation.

```
rpart(y~.,data=df)
```

```
system.time(rpart(y~.,data=df))
```

Contengency table for decision tree before pca

	predict									
actual	0	1	2	3	4	5	6	7	8	9
0	157	0	0	3	1	8	3	0	6	4
1	0	152	6	0	10	1	1	1	14	9
2	11	17	95	1	4	4	10	6	37	13
3	18	3	14	87	1	18	5	4	26	26
4	1	1	3	0	139	3	13	0	4	24
5	25	4	9	10	3	82	7	3	7	32
6	17	1	12	3	14	6	133	0	11	9
7	6	7	9	0	15	0	0	151	1	25
8	3	5	5	6	5	13	7	4	123	33
9	0	1	6	1	8	4	5	16	4	185

Contengency table for decision tree after pca

	predict									
actual	0	1	2	3	4	5	6	7	8	9
0	128	0	12	21	0	11	8	1	1	0
1	0	161	10	5	0	16	2	0	0	0
2	10	1	145	10	2	7	8	2	11	2
3	6	2	6	154	1	11	9	2	10	1
4	0	3	6	2	108	5	3	7	3	51
5	15	0	21	47	6	61	3	8	13	8
6	18	1	20	14	2	0	143	3	2	3
7	0	12	6	5	1	11	2	146	6	25
8	15	0	14	25	1	34	4	2	90	19
9	1	9	1	5	28	11	6	13	5	151

Machine Leaning Classifiers

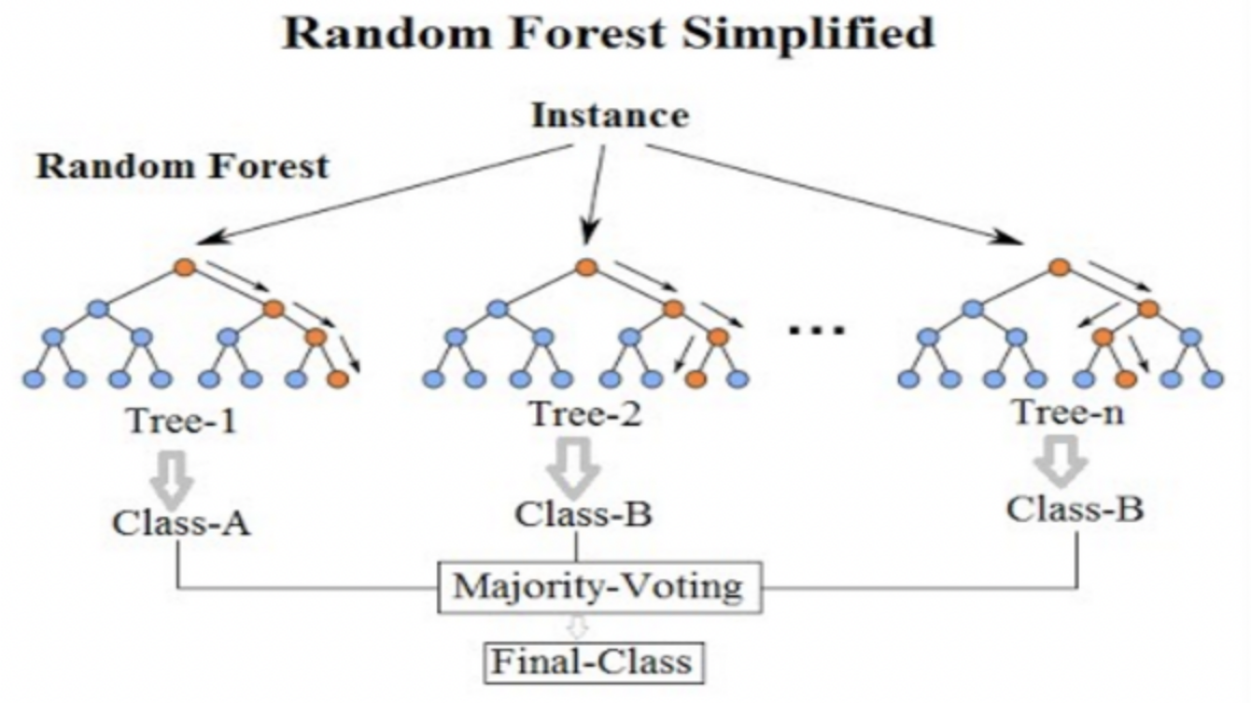
Random Forest

Random Forest is an Ensemble Learning method, which consists of several Decision trees. Each decision tree is trained independently, and its training data is obtained from the original data set by placing back sampling (i.e. bootstrap sampling). In addition, for the node splitting of each decision tree, random forest will randomly select a part of features for evaluation, thus avoiding the problems of collinearity and overfitting between features. Finally, the random forest average the prediction results of all decision trees to get the final classification or regression results.

Random forest has high accuracy, robustness(known for their ability to handle high-dimensional data and noisy or missing values), and is often used to solve classification, regression, clustering and other problems.

Machine Leaning Classifiers

Random Forest



[4]

The package named **ranger** is one of the implementations for Random Forest in R.(better to include set.seed())

```
ranger(y~.,data=df)
```

```
system.time(ranger(y~.,data=df))
```

---

[4] (Jagannath, 2017)

Machine Leaning Classifiers

Naïve Bayes

In machine learning, Naïve Bayes is a classification algorithm based on Bayes theorem. It is based on the assumption of independence among features, that is, the contribution of each feature to the classification results is independent of each other. The joint probability distribution of features and categories in the training set is used to estimate the posterior probability of the category to which the test samples belong, and the test samples are classified into the category with the highest probability. Because of its simple principle, efficient classification performance and good classification effect for small sample data, Naïve Bayes has been widely used in text classification, spam filtering, sentiment analysis and other fields.

The package named **e1071** is one of the implementations for Naïve Bayes in R.

```
naiveBayes(y~.,data=df)  system.time(naiveBayes(y~.,data=df))
```

Bayes Theorem

The theorem can be stated as follows:

$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$

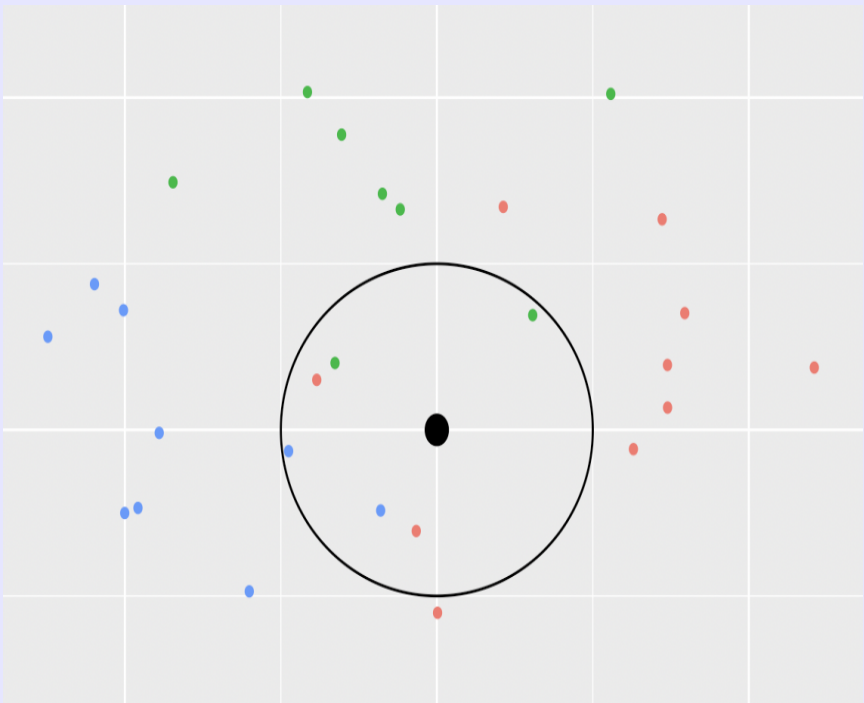
Where

- **P(A/B) is Posterior probability:** Probability of event A occurring given that event B has occurred.
- **P(B/A) is Likelihood:** Measures how likely the observed evidence B is, assuming that the hypothesis A is true.
- **P(A) is the Prior probability:** Represents the initial belief or probability of A before any new evidence is considered.
- **P(B) is Marginal Probability:** The overall probability of observing the evidence B, regardless of the hypothesis.

Machine Leaning Classifiers

KNN

KNN stands for k-nearest neighbors, a non-parametric classification algorithm in machine learning. It is a type of instance-based learning, where the algorithm makes predictions based on the closest (nearest) training examples in the feature space. KNN can be used for both classification and regression problems, depending on how the distance metric is defined and how the output is calculated.

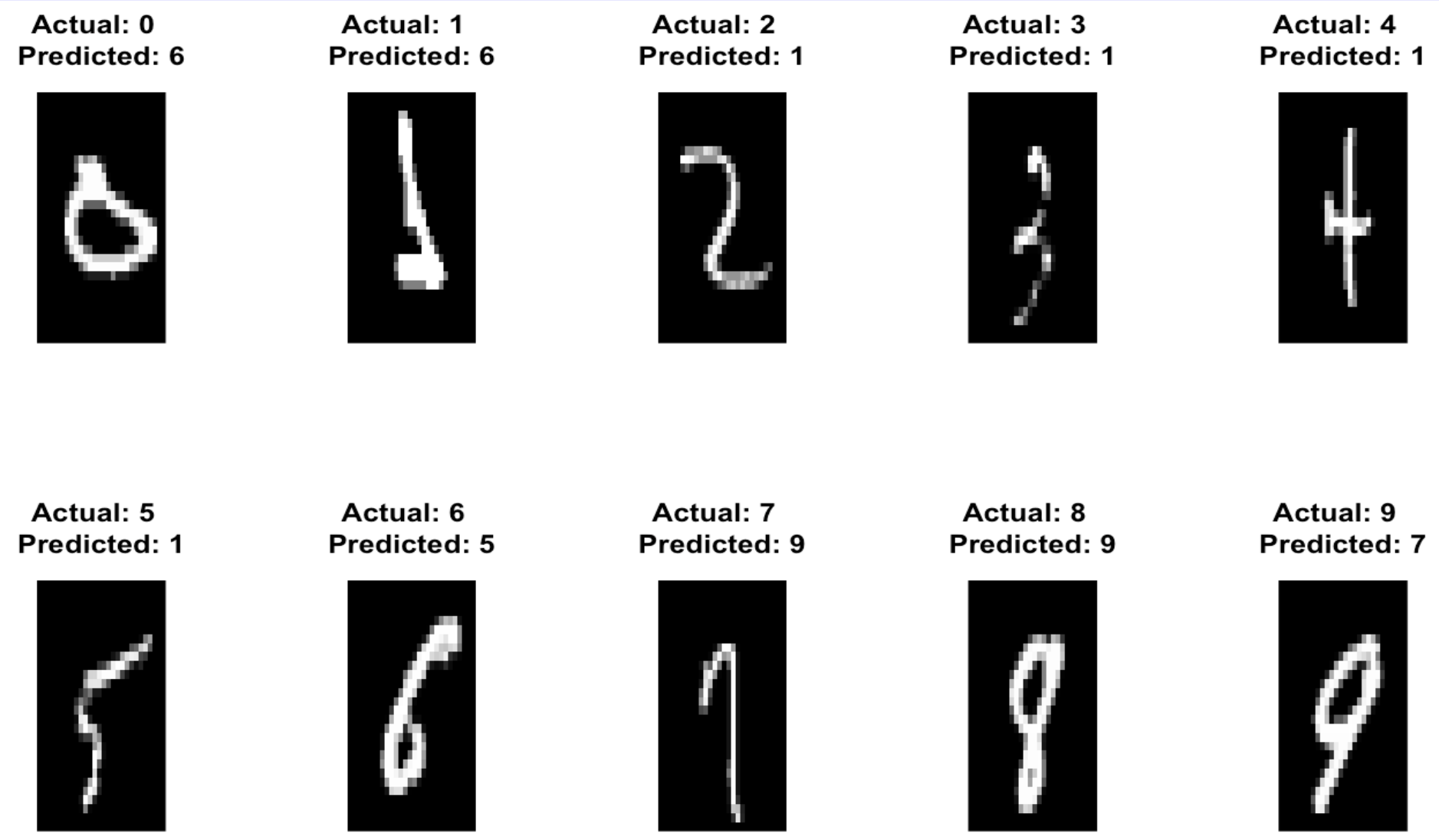


```
kknn(y~.,train_df,test_df,k=10)
```

```
system.time(
kknn(y~.,train_df,test_df,k=10)
)
```

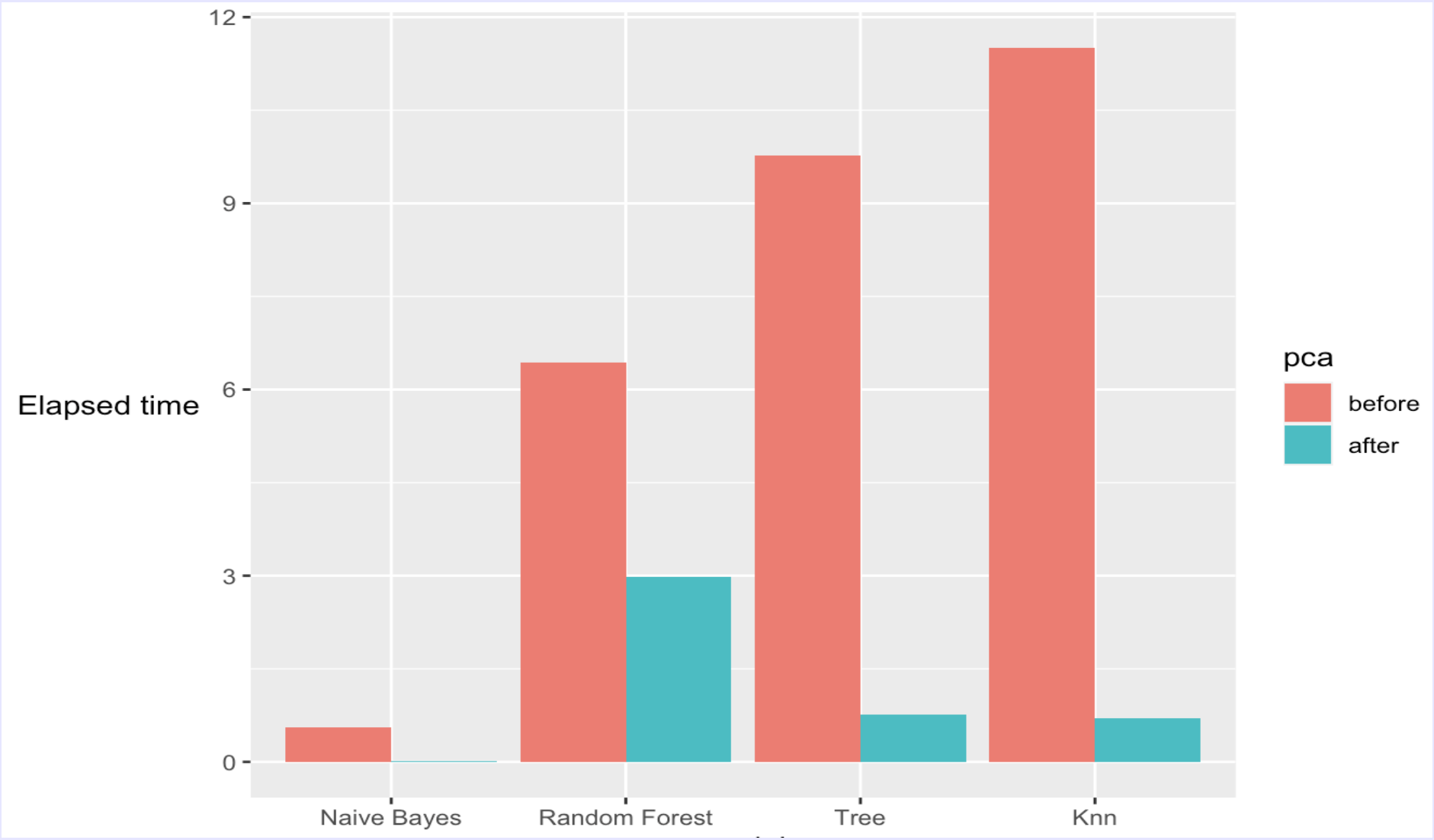
Evaluation

Visualization of some incorrect hand writing prediction in KNN



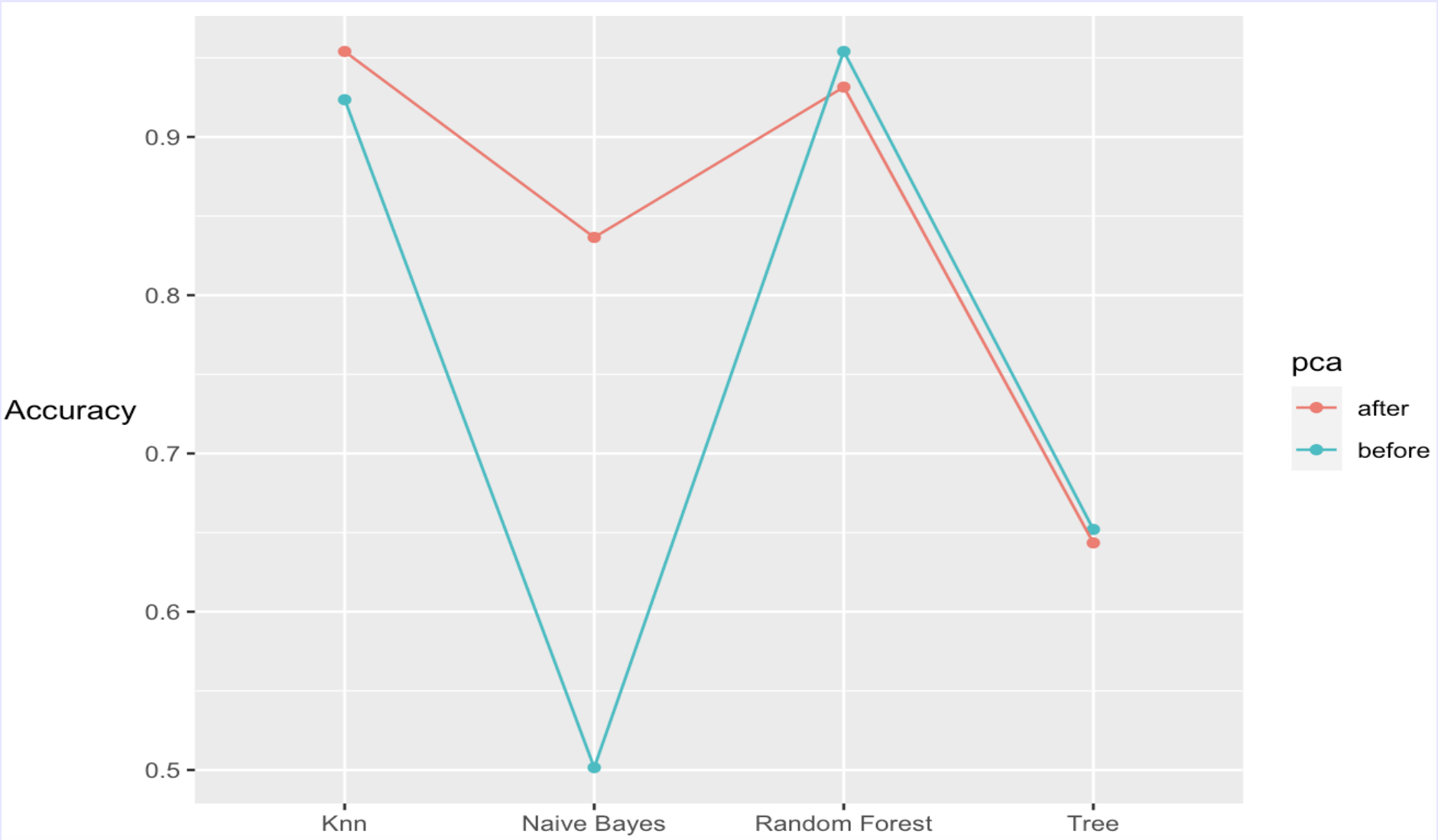
Evaluation

Visualization of comparsion between different models



Evaluation

Visualization of comparsion between different models



Summary

1. We can conclude that the implementation of Principal Component Analysis (PCA) provides a significant time-saving advantage during model generation. By reducing the dimensionality of the feature space, PCA allows for a more efficient computation and analysis of the data. With a smaller number of representative variables, the model training process becomes faster, enabling quicker iterations and experimentation.
2. Furthermore, despite the dimensionality reduction, PCA does not have a substantial negative impact on the accuracy of the models. While PCA condenses the original features into synthetic variables, it retains the most important patterns and variation in the data. The retained synthetic variables still capture a significant portion of the information present in the original data, allowing for accurate modeling and prediction.
3. Thus, the combination of time-saving benefits and maintained accuracy makes PCA an advantageous technique in model generation, enabling efficient analysis and effective predictions without sacrificing the overall quality of the results.

Bibliography

- *Hartmann, K., Krois, J., Waske, B. (2018): E-Learning Project SOGA: Statistics and Geospatial Data Analysis. Department of Earth Sciences, Freie Universitaet Berlin, pp.12-18.*
- Yiu, T. (2019, June 12). *Understanding Random Forest.* Medium; Towards Data Science. `https://towardsdatascience.com/understanding-random-forest-58381e0602d2`
- Jagannath, V. (2017, March 24). Bahasa Indonesia: Random Forest. Wikimedia Commons. `https://commons.wikimedia.org/wiki/File:Random_forest_diagram_complete.png`