STAT 542 Final Project

Fashion MNIST Classification

Contributor: Yijun Zhao yijunz3

Qixuan Xing qixuanx2

Xiao Zhang xiaoz16

Date: 2022/05/05

1. Project Description and Summary

1.1 Objective

In this project, our goal is to build models to classify 10 types of clothing in the Fashion MNIST dataset. We will establish and evaluate different models, then come up with the most efficient and accurate one.

1.2 Approach

First of all, we did cluster for the dataset, and we applied PCA to data first then clustered with k-means using PCA's result. It helped us identify the similar group of data.

In order to evaluate the speed and performance of other models fit either in our compressed dataset, the first classification model we chose is the random forest. This algorithm only considers a random subset of features instead of all the features of the model, which helps to decorrelate the trees. Also, random forest is convenient and fast with high-dimensional datasets since it works with subsets of data. However, since Fashion-MNIST is a large dataset, and the random forest can tend to overfit, we need to tune some of the model's parameters for better accuracy and computational efficiency. And after tuning, the accuracy we get is 0.8852.

We selected the support vector machine to be the second model. First, we rescale the data and then experiment with linear and non-linear SVMs with various hyperparameters. The accuracy of linear kernel SVM is about 0.8111 and the accuracy of radial basis function SVM without tuning is 0.8686, and the model after tuning hyperparameters C and gamma is about 0.8791.

Many models we learned are only for binary classifications, so in this project, we also tried to extend one of them to handle multi-class problems. The one we picked was the binary logistic regression model, which originally can only deal with the output variable that is discrete in two classes. We applied the one-vs-all technique by splitting the multi-classification problem into several binary-classification ones.

In the ensemble modeling, we began with PCA, performed four models (KNN, random forest, logistic, SVM), and then made predictions by assigning a hard vote and a soft vote for the observation that was predicted. The accuracy we get is 0.8726 and 0.8785, respectively.

1.3 Conclusion

Starting with 10 clusters, we performed K-means and PCA clustering and found that the main categories such as shoes and cloth still have clearly borderline, but more detailed categorial such as cloth with leaves and coats are still hard to classify. We finally decided to apply 36 for cluster numbers then help a lot to separate labels. Clustering can help us identify the similar group of data.

From all the models we performed, the random forest has the highest accuracy, the second is SVM, and the third is the ensemble model. In our ensemble model, we used PCA to reduce the

dimension first and then fitted the four models, this may cause information loss and lower the accuracy. Due to a little difference among the ensemble model and single multi-class classification models and a large reduction in computational cost, the soft voting ensemble model performs better.

2. Literature Review

Fashion-MNIST the dataset serves as a replacement for the original MNIST dataset. Many research and experiments have been conducted on the Fashion MNIST, which leads to the production of more than 250 academic papers. The best accuracy on this dataset so far is 96.91%[1] with fine-tuning differential Architecture Search (DARTS).

2.1 An improved method of identifying mislabeled data

Xinbin Zhang [2] from the Sydney Machine Learning Study Group modified the traditional Brodley's algorithm. In Brodley's algorithm, the dataset is split into n parts, for each of which the filter algorithms are trained on the other n-1 parts, and then used to identify correctly or incorrectly labeled instances.

Instead of using n-1 parts to train the filter, Xinbin divided the training and testing data into 7 groups, each of which contains 10000 instances. Next, he trained CNN as a filter. Besides the base filters, Xinbin also introduced ensemble filters that detect mislabeled instances by constructing a set of filtering detectors and then using their classification errors to identify mislabeled instances, two of which are the majority filter and the consensus filter. The accuracy of each filter is in the range of 0.925-0.955.

Finally, Xinbin summarized the type of errors of mislabeled data in the Fashion-MNIST dataset. There are input errors with wrong labels, input errors with wrong pictures, subjectivity errors with wrong labels, two objects errors, and insufficient data errors.

2.2 HOG and SVM model

Greeshma K V and Sreekumar K [3] presented the classification of the Fashion-MNIST dataset using HOG (Histogram of Oriented Gradient) feature descriptor and multiclass SVM.

First, they extracted various features of the images. In advance of training a classifier and evaluating the test, they introduced a preprocessing task to decrease noise artifacts produced while collecting samples of images-HOG feature descriptor. They used the descriptor to divide the image into 2-by-2, 4-by-4, and 8-by-8 small cells, which is used in this work and computes the edge directions. After comparing, they chose the [4 4] cell size. By using this size the numbers of dimensions are limited and this helps to speed up the training process. Also, it contains enough information to visualize the fashion image shape.

Then, they used these extracted HOG features to train the SVM classifier. The results are evaluated using the testing dataset images, and for measuring the accuracy of the classifier they

also produce a confusion matrix. And then these features of 60000 images are given into multiclass SVM for training.

Finally, testing is conducted on 10000 images in the testing dataset. It achieves 86.53% accuracy on test images. Finally, they compared the accuracy results of F-MNIST dataset testing dataset results with various models in the literature, and it shows this accuracy is better than SGD Classifier and Linear SVC.

3. Summary Statistics

Fashion-MNIST is a dataset of Zalando's article images containing 60000 training examples and 10000 testing examples. In this dataset, each image is cut into 28 pixels in height and 28 pixels in width and is assigned to a class. Therefore, for each data, there are 784 pixels in total and a label column. There is no missing value in this data set.

3.1 Data description

Each class is associated with one label. The frequencies of 10 labels and classes in the training dataset and the testing dataset are shown below.

Table 1. Frequency of labels in dataset

Label	Class	Freq in training dataset	Freq in testing dataset
0	T-shirt/top	6000	1000
1	Trouser	6000	1000
2	Pullover	6000	1000
3	Dress	6000	1000
4	Coat	6000	1000
5	Sandal	6000	1000
6	Shirt	6000	1000
7	Sneaker	6000	1000
8	Bag	6000	1000
9	Ankle boot	6000	1000

To more intuitively understand our data, we convert the data into matrices and plot some samples to see the classes.

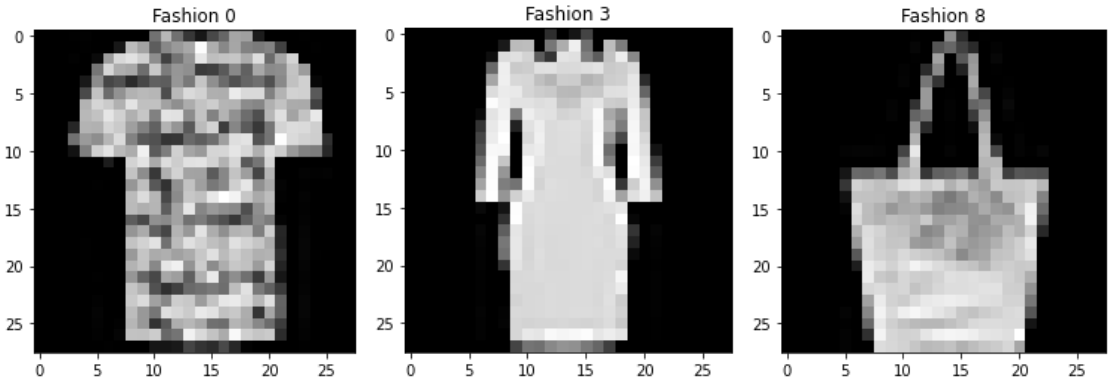

Figure 1. Sample images

3.2 PCA (Principal component Analysis)

When we visualize the mean and standard deviation for each pixel on the train data, for many of the pixels in the image, the mean value is around 100 to 150 but the standard deviation is relatively high. For center regions, the pixels have consistently high pixel values. For other parts, the extreme low/high value and low/high standard derivation have low information content and do not contribute much to models for image classification. In order to remove that low information from the dataset. We can use PCA as a tool to reduce the dimensional to get a maximum variance with fewer components.

We can use the results of PCA to perform a type of information compression on the original data by decreasing the number of original redundancy components. For analysis, we first compare the performance of PCA on our data and use a criterion of 95% variance explained for 784 components.

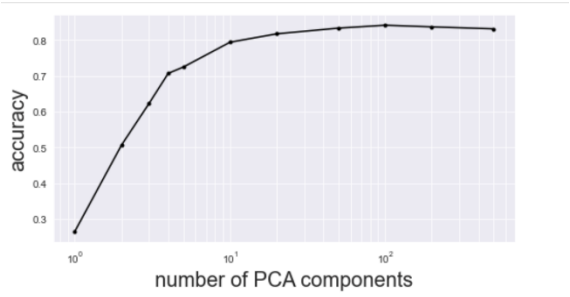

Figure 2. Cumulative Explained Variance

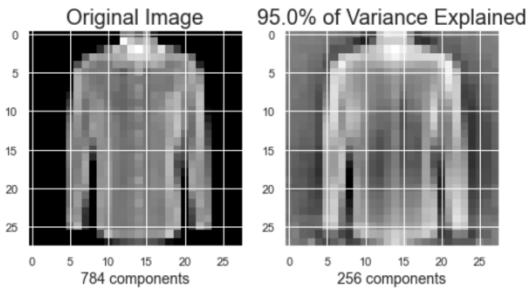

Figure 3. Image before and after PCA

Compressing the data to fewer components does result in information loss, however, as we can see from the figure above, the images retain much of the detail from the original images and only use a feature space of 32.65% of the original.

The figure above shows the representations in the first 2 dimensional PC1 and PC2 feature space. Each clothing item category is represented by a different color. We can find that there is separation across the categories. For example, the shoes group together towards the top left, the cloth group together in the bottom, such that we can easily find the borderline between the big

main categorial. However, 1 has a noticeable distance from 0 with sleeves but is vague with the 3 categories. 1 and 6 are also shared in unclear part.

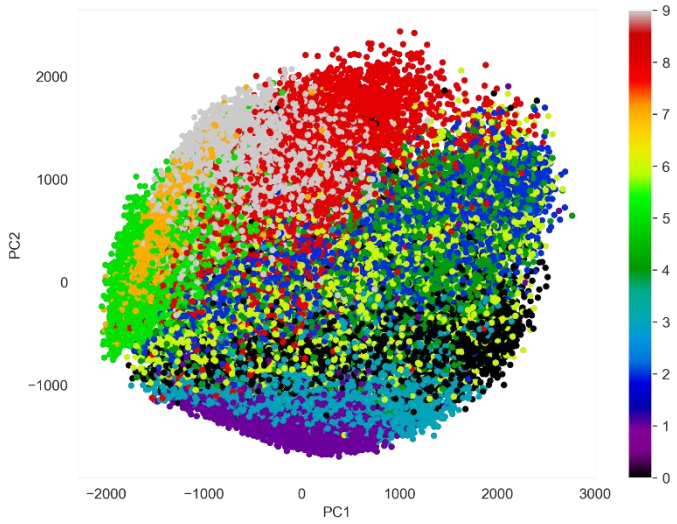

Figure 4. Data projection onto PC1 and PC2

3.3 K-means clustering

At the beginning, we made the assumption that K = 10 was the appropriate number of clusters, then we fit the K-means clustering algorithm with different values of K, then evaluate the performance. The algorithm requires a distance measure to be defined in the data space, and the Euclidean distance is used. Although accuracy is ultimately the most important metric in our clustering algorithm for classification, homogeneity is also an important metric that is directly applied to the clusters themselves. Then we increase our cluster number the result shows as in table: as the increase of cluster number, homogeneity will also increase. Therefore, the cluster number at 36 is not the best result but simple logic and reasonable parameters in our project.

Table 2. Cluster accuracy table

Cluster number	Homogeneity	Accuracy
10	0.58	0.46
36	0.64	0.64
64	0.66	0.687
144	0.71	0.697
256	0.73	0.76

We found that the k-means algorithm can distinguish its clusters based on a reasonable pattern. Thus we can draw a rough conclusion that the K-means clustering method after performing a Principal Component Analysis can get a decent result in classifying images without labels.

4. Multi-class Classification Model

4.1 Random Forest

4.1.1 Approach

As we can see, the outcome variable ‘label’ is incorrectly treated as a number when in reality it represents different types of clothing. So we transformed it to be a factor.

If we, again, print a summary of our data, we get the following. We also checked for the missing values but there weren’t any.

We built a random forest using the `ranger` package. We were not using randomForest which had a very slow implementation in R. We limited the number of trees to speed the training process to 300. To improve prediction accuracy, we tuned the `max.depth` parameter in order to find the optimal depth which gave the highest accuracy (in other words, the lowest prediction error).

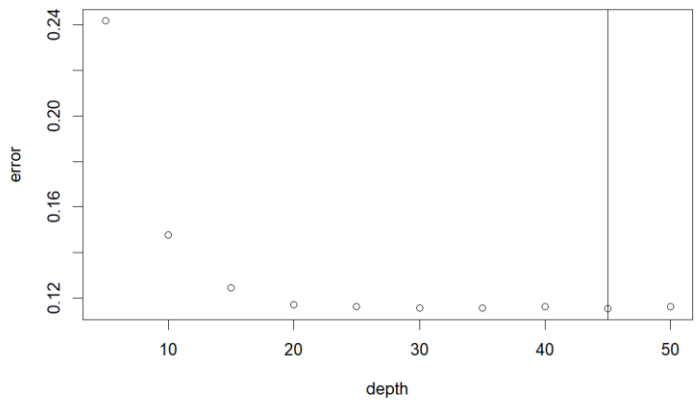

Figure 5. Prediction error vs max.depth

Although it’s not performed in this project due to the memory issue, one possible approach to avoid overfitting is to use a subset of columns by setting `mtry` parameter to build less correlated trees.

4.1.2 Result

After building our random forest, we can start interpreting the result. We created a confusion matrix regarding the random forest and calculated the prediction accuracy, which is 0.8852.

```
Confusion Matrix and Statistics

   pred1
      0   1   2   3   4   5   6   7   8   9
  0 861   0  13  28   0   1  85   0  12   0
  1   1 971   7  15   1   1   4   0   0   0
  2   8   1 803  10 117   0  52   0   9   0
  3  19   7   9 928  20   0  17   0   0   0
  4   0   0  54  30 867   0  46   0   3   0
  5   0   0   0   0   0 948   0  36   5  11
  6 164   1  97  28  75   0 618   0  17   0
  7   0   0   0   0   0  15   0 928   0  57
  8   1   1   8   0   3   2   7   2 975   1
  9   0   0   0   0   0   6   1  38   2 953
```

```
Overall Statistics

               Accuracy : 0.8852
                 95% CI : (0.8788, 0.8914)
    No Information Rate : 0.1083
    P-Value [Acc > NIR] : < 2.2e-16
```

Figure 6. Result of random forest

4.2 Support Vector Machine

4.2.1 Approach

The second multi-class classification model we used is the support vector machine. We'll first scale the data and then experiment with linear and non-linear SVMs with various hyperparameters.

From the perspective of parameter estimation and prediction formulas of SVM, there are some products of the support vector and the prediction vector. What's more, after normalization, the speed of gradient descent to find the optimal solution will be accelerated and the accuracy may be improved. We can check the average of all the 784 features to see whether there is a big difference in the scales. The range of average is 161.81, which is very large, so it is better to rescale them.

After normalization, first, we perform an SVM model with a linear kernel. We don't need to tune parameters for the linear kernel. The result is that the accuracy of the linear SVM model is about 81.11%.

Second, we perform an SVM model with a Gaussian RBF kernel with default hyperparameters, the accuracy is about 86.86%.

Third, we perform an SVM model with a Gaussian RBF kernel and tuning hyperparameters. The parameters that need to be tuned are C and gamma. The C parameter in SVM is the penalty parameter of the error term. If we have low C means low error and if we have large C means large error. Gamma decides how much curvature we want in a decision boundary. If we have high gamma means more curvature and if we have low gamma then less curvature.

We use grid search cross-validation to tune the parameters. For greater values of C, we consider the values of C to be from 5 to 10, and for gamma, we try to 0.01, 0.001, and 0.0001. The figure below (Figure 2.) shows the result more intuitively. From higher to lower gamma (left to right). At very high gamma (0.01), the model is achieving 100% accuracy on the training data, though the testing accuracy is quite low (0.75). Thus, the model is overfitting. When gamma is 0.001, the testing accuracy is much higher. When gamma is 0.0001, the accuracies are lower than the second. Thus, the best combination is gamma=0.001 and C=10 while avoiding overfitting.

4.2.2 Result

The confusion matrix of the third SVM model is shown below. The lighter the color of the diagonal pattern is, the more accurate the classification of the class is. We can see that class 6 has the lowest classification accuracy.

The accuracy of this RBF kernel SVM model is 0.8791, which is much higher than that of a linear kernel.

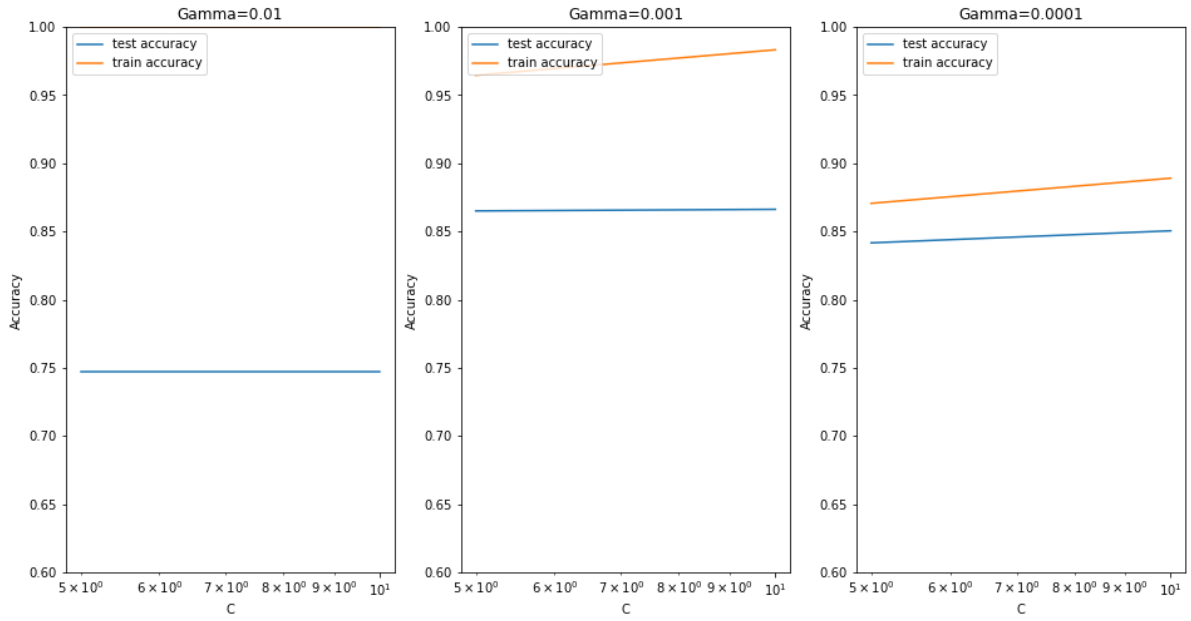

Figure 7. SVM hyperparameters tuning

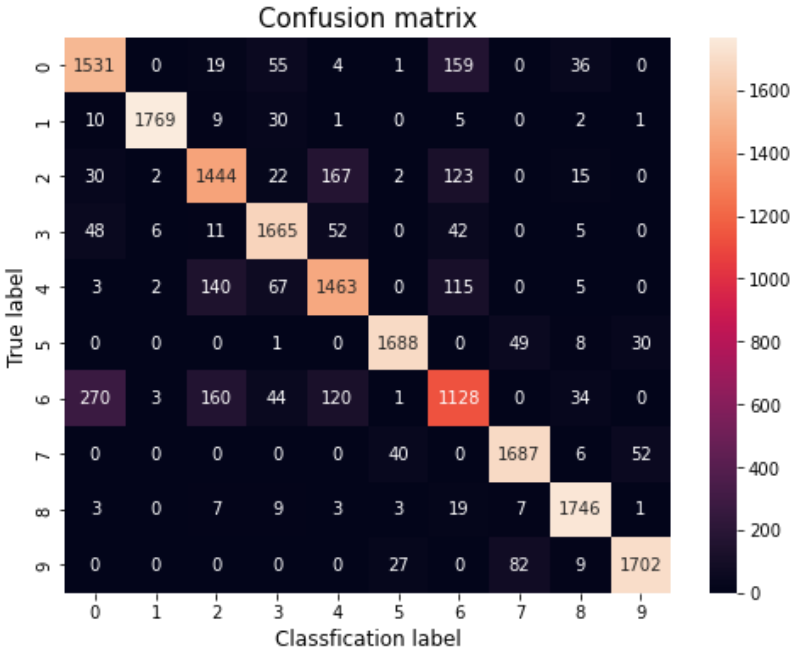

Figure 8. Heatmap of classification confusion matrix

4.3 Binary classifier extension: one-vs-all technique

4.3.1 Motivation

My motivation comes from the image given below.

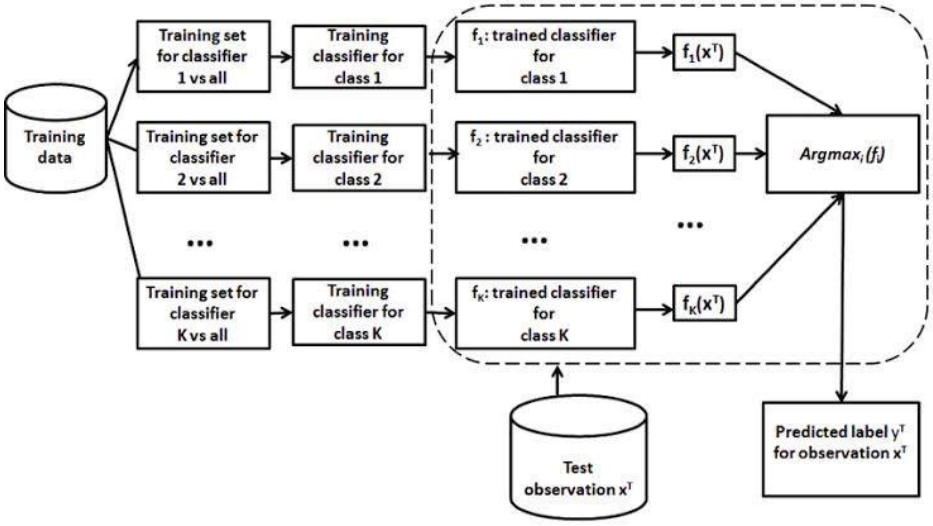

Figure 9. One-vs-all technique

4.3.2 Approach

In one-vs-all classification, in order to classify the dataset with N-class instances, we have to generate an N binary classifier, each corresponding to one specific class.

The Fashion-MNIST dataset contains 10 labels, so we created ten classifiers in manner given below:

- Classifier 1: [0] vs [1,2,3…9]
- Classifier 2: [1] vs [0,2,3,...,9]
- Classifier 3: [2] vs [0,1,3,4,...,9] and so on.

We modified our original dataset before training these classifiers: for instance, for classifier 1, we put 1 in the label column for the feature value (label = 0); for the remaining feature values (label = 1-9) we put 0.

Next, we used these new datasets as the training dataset for each classifier. We applied them to the binary logistic regression model.

After training the models, we predict each testing instance to the class having the greatest probability score.

4.3.3 Result

However, due to numerical error and computational complexity, this method didn't perform as well as the previous two. We got the following confusion matrix and accuracy:

```
Confusion Matrix and Statistics

   pred
      0   1   2   3   4   5   6   7   8   9
  0 847   8  23   3   1   5  95   1  17   0
  1  10 970   6   6   0   2   4   0   2   0
  2  21   4 808   5  48   4 102   0   8   0
  3 147  29  39 647   5   6 113   1  12   1
  4  22   3 259   5 331   2 367   2   9   0
  5  13   7   8   3   0 871  25  26  34  13
  6 179   4 119  13  21   8 637   1  18   0
  7   4   2  17   0   0 126  44 728  67  12
  8   7   2   8  10   5  32  23   5 906   2
  9   3   2  13   0   1  35  57  15  68 806

Overall Statistics

                Accuracy : 0.7551
                  95% CI : (0.7465, 0.7635)
     No Information Rate : 0.1467
     P-Value [Acc > NIR] : < 2.2e-16

                   Kappa : 0.7279

  Mcnemar's Test P-Value : < 2.2e-16
```

Figure 10. Result of binary classifier extension

4.4 Summary

We applied the random forest, support vector machine, and robust multi-class logistic regression as an extension of binary. The model which had the best performance is the random forest, followed by the support vector machine. One possible explanation is that the random forests are inherently multiclass while Support Vector Machines need workarounds to handle multiclass classification tasks.

Table 3. Classification error of models

Model	Classification Accuracy
Random Forest	0.8852
SVM with linear kernel	0.8111
SVM with RBF kernel	0.8686
SVM with RBF kernel and tuned hyperparameters	0.8791
Binary classifier extension	0.7551

5. Ensemble Model and Feature Engineering

5.1 Model building

We have shown above that there is a bunch of low information data in the Fashion-MNIST dataset. Here we will use PCA to reduce the number of features first while retaining as much of the variance possible. In this part, we will evaluate the speed and performance of more models that fit either the original 784 feature image set or the PCA compressed 256 dataset. The compressed train dataset will fit faster, but we still need to consider the prediction accuracy. Then, we fit four models: KNN (k-nearest neighbors), logistic regression, random forest, and SVM (support vector machine).

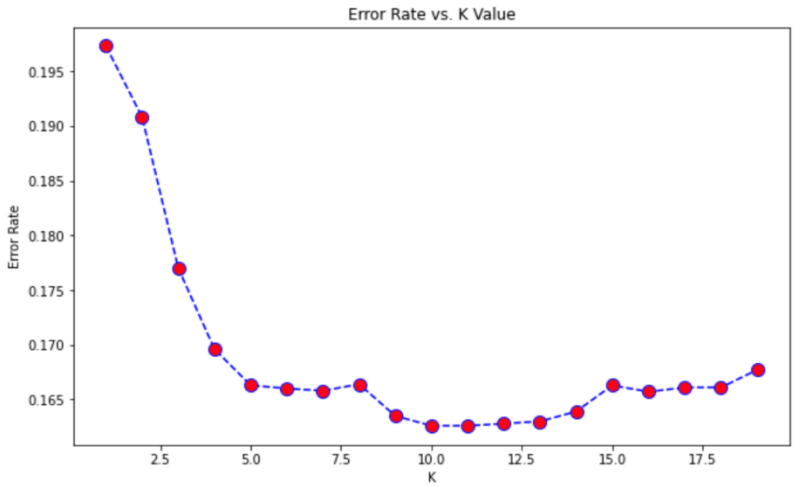

Figure 11. Prediction error vs. K

It is robust to noisy training data and is effective in case of a large number of training examples.

But for this algorithm, the value of parameter K (number of nearest neighbors) is 10 by using the elbow method, the error rate increases after 10. Random Forest algorithms can handle our high dimensional spaces as well as a large number of training examples. In our case, It is measured by the Gini index error rate and set n_estimators = 100. The main reason to use an SVM instead is that the problem might not be linearly separable. In that case, we will have to use an SVM with a Radial Basis Function (RBF) kernel, and the C parameter trades off correct classification of training examples against maximization of the decision function's margin is 10. Because the larger values of C caused a smaller margin, but A lower C will encourage a larger margin. Therefore, the decision function is simpler and has low training accuracy. For logistic regression, we decided to use liblinear since it is a simple class for solving large-scale regularized linear classification.

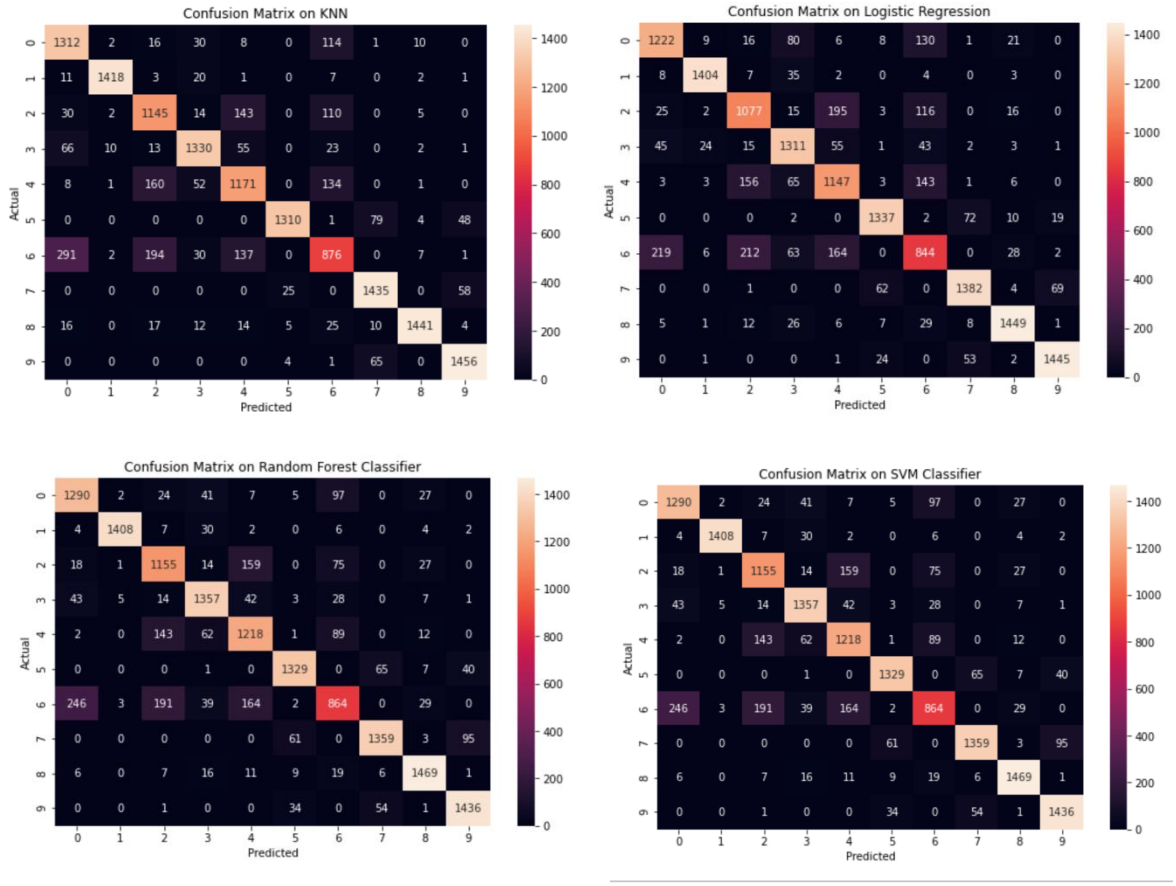

Figure 12. Confusion matrix of four models after PCA

Apparently, SVM is the most time-consuming algorithm in our model even if we apply the PCA. Other methods are a lot more efficient. We think the most time-consuming part of the SVM method is the optimization of the objective function. In SVM, we are trying to maximize the margin of the hyperplane that separates the data points, then down to minimizing an objective function. The accuracy for KNN is higher than others but not a lot. So, we decided to combine the performances of multiple models to make predictions. Since model voting relies on the performance of our all models, they will not be affected by large errors or misclassifications.

Table 4. Classification accuracy of four models after PCA

Model	Classification Accuracy	Time Consuming
KNN	0.8866	82.34 seconds
Logistic	0.8412	112.323 seconds
Random Forest	0.8756	73.24 seconds
SVM	0.8794	1125 seconds

From these four models, we extract the predicted labels and combine them for the next stage.

5.2 Hard Voting Classifiers vs Soft Voting Classifiers

In the second stage, we first use both hard voting classifier and soft voting classifier to estimate the classification based on the results above. To classify input data, the hard voting classifier focuses on the mode of all the predictions made by different classifiers, while the soft voting classifier focuses on the probabilities. Weights associated with the different classifiers will affect the performance of majority voting. In this model, we do the majority voting based on equal weights.

We define a function to combine these models and input the data after PCA, then get their predictive labels and use them to vote for a majority as the final predictive label. The results are shown below.

Table 5. Accuracy of ensemble model

Model	Hard voting classifier	Soft Voting classifier
Ensemble model	0.8727	0.8785

It seems that the results of the hard voting classifier and soft voting classifier are very similar, though the accuracy of the soft voting model is a little higher.

The accuracy of this ensemble model is not better than that of the previous random forest and nonlinear SVM models. This may be because when we use PCA first, we lose some information, then the results may not be as good as before. However, the data after PCA can greatly reduce the computational cost, and the difference of accuracy is small, so we can choose this soft voting ensemble model.

Reference:

[1] Tanveer M S, Khan M U K, Kyung C M. Fine-tuning darts for image classification[C]//2020 25th International Conference on Pattern Recognition (ICPR). IEEE, 2021: 4789-4796.
[2] Zhang X. An improved method of identifying mislabeled data and the mislabeled data in MNIST and CIFAR-10 appendix Findings in fashion-MNIST[J]. Available at SSRN 3097307, 2018.
[3] Greeshma K V, Sreekumar K. Fashion-MNIST classification based on HOG feature descriptor using SVM[J]. International Journal of Innovative Technology and Exploring Engineering, 2019, 8(5): 960-962.
[4] Band A. Multi-class Classification — One-vs-All & One-vs-One. Towards Data Science. 9 May 2020.
https://towardsdatascience.com/multi-class-classification-one-vs-all-one-vs-one-94daed32a87b.
[5] MNIST Digit recognition using SVM
https://www.kaggle.com/code/nishan192/mnist-digit-recognition-using-svm/notebook