STAT 542: Final Project

Karthik Vasu (kvasu2), Yining Lu (yining13)

Due: 05/05/2022

Contents

Project Summary

This project is on implementing classification algorithm on the Fashion-MNIST dataset. To begin with we pre process the data using PCA to extract enough useful information so that the data contains a signal but also reduce it enough for the algorithms to be fast.

First we try a K-means unsupervised algorithm to cluster the data hoping that it will be able to divide the labels. Unfortunately the clusters we get even with taking K=100 don't have a uniform label. There are a lot of mixed labels in each cluster. The ratio of majority vote in cluster to total cluster size we get is 73%, which is low.

Next we try supervised learning. We try 4 different methods, random forest, k-NN, linear SVM and kernel SVM. We use 5 fold cross validation to tune the parameter k in k-NN. The best k we achieve is k=10. The 10-NN algorithm gives us an accuracy of 85.6%. Random forest achieves the accuracy of 88.5%, which is better than that of 10-NN. The SVM classifier is a binary classifier which we adapt to our multiclass classification by using two approaches, One vs Rest and One vs One. We use the One vs One approach which achieves an accuracy of 81.2% in the linear version and 90.8% in the kernel version. This clearly outperforms the previous two algorithms.

Finally we create and ensemble model using multiple classifiers. We use kNN, random forest, kernel SVM and PCA-SVM(SVM after performing PCA). After fitting these models, we use the majority vote to predict the final label. The ensemble model performs marginally better than the SVM model to get an accuracy of 91.1%.

In conclusion, the ensemble model performs the best, but it has high computational cost as it has to train multiple models. The advantage of the ensemble model is that it is flexible.

Literature review

The Fashion-MNIST data set has been created by researchers at Zalando for the purposes of benchmarking ML algorithms. It consists of 70000 grayscale images of dimension 28*28. These are images of clothing articles like T-shirt, Trouser, Pullover etc with 60000 training samples and 10000 testing samples. The purpose of this data set is to provide a more challenging classifying compared to the original MNIST data. There are algorithms which 99% accuracy on this making it too easy for modern algorithms.

The best accuracy we found was by a GitHub user named *Andy Brock* who was able to achieve an accuracy of 96.7% using wide residual networks. A lot of people have implemented algorithms with high accuracy. They can be for on *Zalando Research's GitHub page*.

Xiao, Rasul, and Vollgraf (2017) test out a variety of classifiers including Decision Tree ,Gradient Boosting, K Neighbors, Linear SVC, Logistic Regression and many more. They achieve the best result using the SVC classifier with C=10 and the rbf kernel. The testing accuracy for this algorithm is 89.7% on the fashion data set and 97.3% on the original MNIST data. Gradient boosting performs well with testing accuracy at 88% and 96.9% respectively. This is achieved for n_estimators=100 and max_depth=10.

Meshkini, Platos, and Ghassemain (2019) perform classification on the Fashion-MNIST data set using convolutional neural networks. They compare the performance of several well-known deep learning frameworks, such as AlexNet, GoogleNet, VGG and ResNet, DenseNet and SqueezeNet. The authors also propose an additional step of batch normalization to enhance the training speed and accuracy of the model. The best results are achieved by ResNet44 and SqueezeNet with batch normalization with testing accuracy at 93.39% and 93.43% respectively.

Summary Statistics

Data table

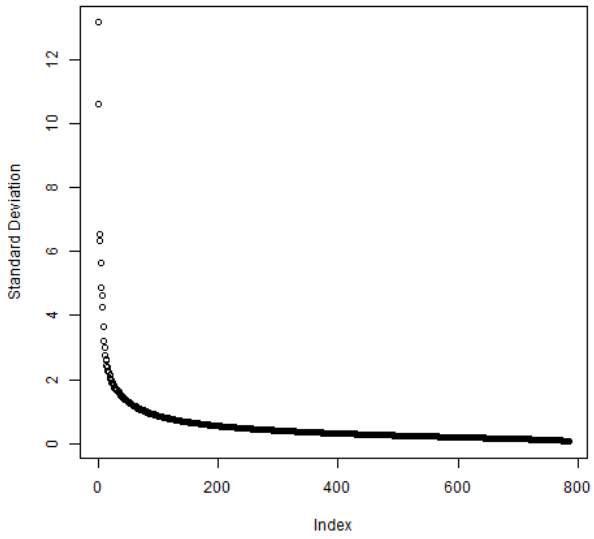
```
## Ytrain
##    0    1    2    3    4    5    6    7    8    9
## 6000 6000 6000 6000 6000 6000 6000 6000 6000 6000

## Ytest
##    0    1    2    3    4    5    6    7    8    9
## 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000
```

Pre processing

Before using PCA we have scaled and centered the data.

PCA

We perform PCA on the training data set and plot the standard deviation of each of the components in descending order. Using the elbow method the cutoff we choose is 1.75 and take all the components which have standard deviation above that value. It gives 28 components.



Clustering

Kmeans

To first determine which K to use we run the kmeans algorithm for k =10,20,. . . ,100 and measure for each cluster the ration # of votes the majority label got/ total number of elements in the clustes. We also sum over all the clusters by weighing the ratios according to the cluster size. The plot of this cluster confidence vs K is as follows.

From this plot we can see that as we increase K the confidence increases. We choose K=100 so that to get the best clustering while keeping computation time low.

The cluster confidence is 73.58% and the majority label in each cluster are as follows

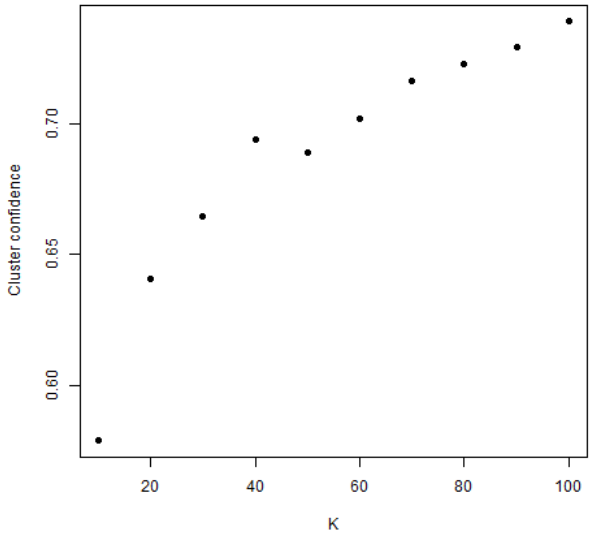


Figure 1: Cluster confidence vs K

```
##   [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14]
## x    1    4    1    9    0    8    8    8    6     0     8     6     8     3
##   [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25] [,26] [,27]
## x     7     8     8     6     2     9     8     5     3     3     6     9     0
##   [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35] [,36] [,37] [,38] [,39] [,40]
## x     4     8     2     9     3     8     9     8     5     6     6     9     3
##   [,41] [,42] [,43] [,44] [,45] [,46] [,47] [,48] [,49] [,50] [,51] [,52] [,53]
## x     0     2     0     7     8     9     4     3     3     8     2     4     5
##   [,54] [,55] [,56] [,57] [,58] [,59] [,60] [,61] [,62] [,63] [,64] [,65] [,66]
## x     8     8     8     2     8     0     0     4     1     8     1     4     5
##   [,67] [,68] [,69] [,70] [,71] [,72] [,73] [,74] [,75] [,76] [,77] [,78] [,79]
## x     7     8     0     9     4     2     4     0     0     2     9     2     4
##   [,80] [,81] [,82] [,83] [,84] [,85] [,86] [,87] [,88] [,89] [,90] [,91] [,92]
## x     6     8     4     7     5     3     6     9     5     1     1     2     8
##   [,93] [,94] [,95] [,96] [,97] [,98] [,99] [,100]
## x     4     7     9     8     6     5     4      9
```

Multi-class Classification Model

In this section, we implement multi-class classification models on the Fashion-MNIST dataset. We first use two algorithms, k-Nearest Neighbor (k-NN) and random forest, which can be directly used for multiclass classification. Then, we extend Support Vector Machine (SVM), which is a binary classifier, to handle the multiclass case.

k-Nearest Neighbors

We first implement k-NN classification model to classify the Fashion-MNIST data. The Euclidean distance is used to measure the distance between variables. The raw data are scaled in the data pre-processing step, so the distance is scale-invariant.

To determine the parameter k, we use 5-fold cross-validation and calculate the overall mis-classification rate for each k. The figure shows the relation between mis-classification rate and k. We find that the best mis-classification rate is reached at $k = 10$.

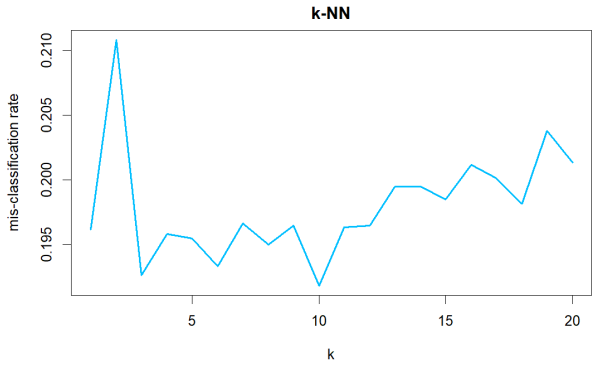


Figure 2: 5-fold CV Mis-classification Rate

Based on the result of cross-valiadation, we choose $k = 10$ to fit the k-NN model. The confusion matrix and mis-classification rate for each class are shown in the following two tables. For Fashion-MNIST data, the overall mis-classification rate of the 10-NN classifier is 0.144 and the testing accuracy is 0.856.

	Ytest_pred									
Ytest	0	1	2	3	4	5	6	7	8	9
0	856	1	18	17	4	0	95	1	8	0
1	3	964	9	13	1	0	10	0	0	0
2	11	0	794	11	97	0	86	0	1	0
3	30	8	13	888	37	0	24	0	0	0
4	2	0	74	22	810	0	90	0	2	0
5	0	0	0	1	0	788	10	121	6	74
6	191	1	106	19	71	0	601	0	11	0
7	0	0	0	0	0	2	0	936	0	62
8	2	1	15	3	5	0	11	6	955	2
9	0	0	0	0	0	1	1	31	0	967

(a) Confusion Matrix

class	mis−classification
0	0.218
1	0.011
2	0.228
3	0.088
4	0.21
5	0.004
6	0.352
7	0.145
8	0.028
9	0.125
overall	0.144

(b) Mis-classification Rate

Figure 3: 10-NN

Random Forest

We then classify the Fashion-MNIST data using the random forest. The confusion matrix and mis-classification rate for each class are shown in the following two tables. The overall mis-classification rate of random forest is 0.115 and the testing accuracy is 0.885. The results show that random forest has a better performance than 10-NN classifer.

	Ytest_pred_tree									
Ytest	0	1	2	3	4	5	6	7	8	9
0	859	0	12	31	0	1	84	0	13	0
1	1	972	5	16	1	1	4	0	0	0
2	8	1	804	11	114	0	52	0	10	0
3	18	7	7	931	20	0	17	0	0	0
4	1	0	58	31	862	0	45	0	3	0
5	0	0	0	0	0	947	0	36	5	12
6	167	1	96	27	73	0	619	0	17	0
7	0	0	0	0	0	17	0	928	0	55
8	1	1	10	0	3	1	6	2	975	1
9	0	0	0	0	0	7	1	40	3	949

(a) Confusion Matrix

class	mis−classification
0	0.186
1	0.01
2	0.19
3	0.111
4	0.197
5	0.028
6	0.252
7	0.078
8	0.05
9	0.067
overall	0.115

(b) Mis-classification Rate

Figure 4: Random Forest

Support Vector Machine

We have applied two multiclass classifiers k-NN and random forest for the Fashion-MNIST classification. Now, we extend SVM to the multiclass form to solve the classification problem. SVM is a binary classifier and it does not support multiclass classification natively. However, we can break the multiclass classification problem into several binary ones. There are two common methods to extend SVM for multiclass classification, One-vs-One approach and One-vs-Rest approach.

In One-vs-One approach, we fit SVM models for every two classes. Each classifier separates points of two different classes. Suppose we have k classes, we then fit $\frac{k(k-1)}{2}$ SVM models. In prediction stage, we input data into all binary classifiers. Each binary classifiers will decide a class that the input is belonged to. We let those binary classifiers vote for the class of input, and the prediction result is the class that most classifiers vote for.

In One-vs-Rest approach, we fit SVM models to distinguish points of one certain class from the other classes. Suppose we have k classes, we then fit k SVM models. In prediction stage, we input data into all One-vs-Rest classifiers. Each classifier will give a probability that the input is belonged to that class. The prediction result is the class with the largest probability.

One-vs-Rest approach is more computationally efficient than One-vs-One approach, since it only need to fit k SVM models while One-vs-One approach need to fit $\frac{k(k-1)}{2}$ models. However, in One-vs-Rest approach, the training data is unbalanced since the ratio of training data from each class is $1 : (k-1)$, which may cause biase.

We fit the multiclass SVM model using One-vs-One approach, with using radial basis kernel. The overall mis-classification rate of kernel SVM is 0.092 and the testing accuracy is 0.908. The kernel SVM classifier outperforms k-NN and random forest.

In addition, a linear SVM model is constructed to compare with the RBF kernel. The overall mis-classification rate of linear SVM is 0.188 and the testing accuracy is 0.812.

Ytest	Ytest_pred_svm 0	1	2	3	4	5	6	7	8	9
0	878	0	13	18	1	0	81	0	9	0
1	3	987	1	7	0	0	1	0	1	0
2	16	1	834	14	74	0	59	0	2	0
3	24	7	8	919	26	0	15	0	1	0
4	3	1	58	22	877	0	38	0	1	0
5	0	0	0	0	0	952	1	27	6	14
6	123	1	52	27	55	0	732	0	10	0
7	0	0	0	0	0	10	0	961	0	29
8	4	0	4	3	2	2	4	1	980	0
9	0	0	0	0	0	5	0	29	3	963

(a) Confusion Matrix

class	mis−classification
0	0.165
1	0.01
2	0.14
3	0.09
4	0.153
5	0.018
6	0.214
7	0.056
8	0.033
9	0.043
overall	0.092

(b) Mis-classification Rate

Figure 5: Kernel SVM (rbf)

Though linear SVM model does not perform better than k-NN and random forest, SVM with RBF kernel have a quite good performance on classification, which achieves an overall accuracy of 90.8%.

Ytest	Ytest_pred_lsvm 0	1	2	3	4	5	6	7	8	9
0	773	8	18	35	5	1	135	0	23	2
1	3	980	2	12	2	1	0	0	0	0
2	33	1	725	15	106	0	112	0	8	0
3	78	32	13	822	28	0	26	0	1	0
4	3	0	173	40	688	0	94	0	2	0
5	2	0	0	1	0	890	0	65	3	39
6	193	5	144	44	87	0	509	0	17	1
7	0	0	0	0	0	63	0	886	1	50
8	9	0	18	4	6	7	22	5	928	1
9	0	0	0	0	0	23	0	56	0	921

(a) Confusion Matrix

class	mis−classification
0	0.165
1	0.01
2	0.14
3	0.09
4	0.153
5	0.018
6	0.214
7	0.056
8	0.033
9	0.043
overall	0.188

(b) Mis-classification Rate

Figure 6: Linear SVM

Conclusion

In this part, we build k-NN, random forest, linear SVM and kernel SVM to classify Fashion-MNIST data. The testing accuracy of different algorithms are shown in the table. The testing accuracy of four methods: kernel SVM > random forest > k-NN > linear SVM.

Table 1: Model Summary

	kNN	Random.Forest	Linear.SVM	Kernel.SVM
Accuracy	0.8559	0.8846	0.8122	0.9083

Ensemble Model

In the last two sections, we have implemented unsupervised learning and supervised learning on Fashion-MNIST classification respectively. Now, we combine unsupervised learning and supervised learning together to build a joint model with a better accuracy of classification.

PCA-SVM

We first combine principal component analysis (PCA) and support vector machine (SVM) to create a classification pipeline. In the first stage, we apply PCA to reduce the dimensions of feature space as well as make features orthogonal to each other. In the second stage, we select components with great deviation among principal components, and use them as input to fit a SVM classifier.

We use the elbow method with the cutoff 1.75 and take all the components which have standard deviation above that value, which gives 28 components. We use those 28 components to fit a multiclass kernel SVM with RBF kernel, which is demonstrated in the classification part. Before performing PCA, we preprocess the data by subtracting the mean and scale it to variance 1.

The confusion matrix and mis-classification rate for each class are shown in the following two tables. The overall mis-classification rate of PCA-SVM classifier is 0.112 and the testing accuracy is 0.888, which is better than random forest but worse than the origin SVM.

Ytest	Ytest_pred_psvm									
	0	1	2	3	4	5	6	7	8	9
0	855	3	16	24	2	1	93	0	6	0
1	6	978	2	14	0	0	0	0	0	0
2	16	0	808	14	78	0	78	0	6	0
3	23	13	13	900	36	1	12	0	2	0
4	1	2	73	22	848	0	53	0	1	0
5	0	0	0	0	0	946	0	32	2	20
6	135	1	71	30	66	0	689	0	8	0
7	0	0	0	0	0	28	0	938	0	34
8	4	0	8	2	1	2	8	3	970	2
9	1	0	0	0	0	15	0	38	1	945

(a) Confusion Matrix

class	mis−classification
0	0.179
1	0.019
2	0.185
3	0.105
4	0.177
5	0.047
6	0.262
7	0.072
8	0.026
9	0.056
overall	0.112

(b) Mis-classification Rate

Figure 7: PCA-SVM

Though our PCA-SVM classifier does not improve the classification performance compared with the origin SVM, the classifier still explores some features of the data that the origin model may not have explored. Our next step is to combine PCA-SVM and SVM, in additional to random forest and k-NN model to build an ensemble classifer.

Ensemble Model

The classification models are limited to certain type of data structure. However, in real-world problems, it is difficult to find a model that well satisfied the data structure of the problem. Thus, ensemble multiple classification models and combine the information together may result in a more robust model which has a better performance than the single model.

We combine four different models, k-NN, random forest, kernel SVM and PCA-SVM, to build an ensemble classification model. The model tuning and performance of the single model is demonstrated in the classification section. The following figure shows the architecture of the ensemble model.

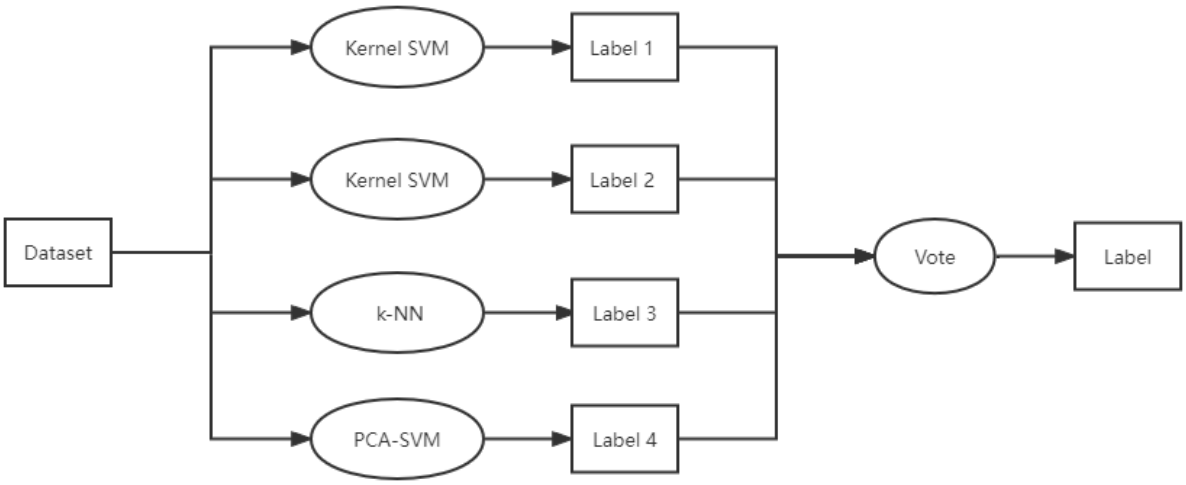


Figure 8: Ensemble Model Architecture

The classification pipeline has two stages. In the first stage, we let k-NN, random forest, kernel SVM and PCA-SVM classify input data respectively. Each classifier will give a predicted label for the input. In the second stage, we let those classifiers vote for the class of input, and the prediction result is the class that most classifiers vote for.

Now, we fit the ensemble model with Fashion-MNIST data. The confusion matrix and mis-classification rate for each class are shown in the following two tables. The overall mis-classification rate of the ensemble model is 0.089 and the testing accuracy is 0.911, which outperforms all of the single model.

	Ytest_pred_en									
Ytest	0	1	2	3	4	5	6	7	8	9
0	888	0	12	17	1	0	76	0	6	0
1	2	988	0	7	0	0	3	0	0	0
2	16	1	839	16	73	0	53	0	2	0
3	26	9	8	917	25	0	14	0	1	0
4	3	1	61	25	871	0	37	0	2	0
5	0	0	0	0	0	953	1	29	3	14
6	126	0	54	24	52	0	740	0	4	0
7	0	0	0	0	0	8	0	963	0	29
8	3	0	4	2	2	2	5	1	981	0
9	0	0	0	0	0	3	0	26	3	968

(a) Confusion Matrix

class	mis−classification
0	0.165
1	0.011
2	0.142
3	0.09
4	0.149
5	0.013
6	0.203
7	0.055
8	0.021
9	0.043
overall	0.089

(b) Mis-classification Rate

Figure 9: Ensemble Model

Conclusion

Finally, we take a comparison between the ensemble model and all other single models. The testing accuracy of different algorithms are shown in the table. We find that the ensemble model indeed utilizes the advantages from different single models and have the best performance among all algorithms.

Table 2: Model Summary

	kNN	Random.Forest	Linear.SVM	Kernel.SVM	PCA.SVM	Ensemble.Model
Accuracy	0.8559	0.8846	0.8122	0.9083	0.8877	0.9108

The computational cost for the ensemble model is relatively high since we need to fit each single model respectively. However, if we have already trained some classifiers and want to further improve the model's performance, the computational cost can be reduced.

Additionally, the ensemble model is very flexible since it is convenient to add new models or remove inappropriate ones to improve the performance of the ensemble model.

References

Meshkini, Khatereh, Jan Platos, and Hassan Ghassemain. 2019. "An Analysis of Convolutional Neural Network for Fashion Images Classification (Fashion-MNIST)." In *International Conference on Intelligent Information Technologies for Industry*, 85–95. Springer.

Xiao, Han, Kashif Rasul, and Roland Vollgraf. 2017. "Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms." August 28, 2017. https://arxiv.org/abs/cs.LG/1708.07747.