Final Project Report
STAT 542

Team members:

Kexin Zhu (kexinz8)
Erh Hsaun Wang (ewang36)
Wenjun Cao (wenjunc2)

PROJECT DESCRIPTION AND SUMMARY

In this project, we want to find the algorithm that can best classify labels from the Fashion MNIST dataset. To achieve the goal, we carry out both clustering and classification explorations on the Fashion MNIST dataset. In the unsupervised learning section, we select MiniBatch-Kmeans and Gaussian Mixed Model to represent the distance-based and density-based measures, respectively, for cluster analysis.

The performances of our unsupervised learnings are not as good as we expected, their accuracies are both around 50%. Hence, we try to use supervised learning models. In the supervised section, we extend the classical binary-classification methods like Linear Discrimination Analysis, multi-version Support Vector Machine, and implement LightGBM. We predicted their outcomes and compared the performance of these three models. Among them, LDA has the lowest accuracy (82.56%), and SVM's accuracy is in the middle (87.45%), while the highest accuracy level is obtained from LightGBM (90.73%). As for the time complexity, SVM needs the longest time to complete the calculation, while LightGBM runs the fastest.

As the implementary experiment, we take advantage of the strength of the above three models (LDA, SVM, and LightGBM) that we constructed as our base classifiers and utilize Logistic Regression as a meta-learner to make our final predictions.

In the ensemble stage, we conduct Majority Voting, propose a new method Voting++ and Stacking to integrate the models. The following table shows the outcome of each classifier, we can see the final test accuracy of our ensemble model stacking is 90.8%, which outperforms every single classifier.

	LightGBM	LDA	SVM	Voting	Voting++	Stacking
Accuracy	0.9073	0.8256	0.8745	0.89	0.894	0.9083

For questions to deeper explore, the first one is to explore relationships between indicators. By doing so, we can use the most informative ones to do prediction. Otherwise, we can try out different base classifier combinations, we now use the combination of LightGBM, LDA, and SVM classifier. We can try different combinations to construct the stacking model. Last but not least, resorting to the Bagging method to increase the variability of the models by different sampling. At the same time, disparating the computing.

LITERATURE REVIEW

Since being proposed in 2017, Fashion MNIST dataset has gained unquenchable discussion in the machine learning world. Topics centered on it vary from dimension deduction[i], autoencoder[ii], image classification and algorithm optimization, etc. This paper focuses on clustering and classification.

For traditional unsupervised learning methods, the largish features and samples of the dataset are putting up challenges for them. In response to this, scholars are diving into the high-dimensional clustering research. Allaoui(2020)[iii] use the dimensionality reduction technique UMAP strengthens traditional machine learning clustering methods(ask-means, HDBSCAN, GMM and Agglomerative Hierarchical Clustering) by improving 60% the accuracy.

There is a surge of applying classification methods on Fashion MNIST dataset. SVM[iv] , improved random forest, [v] and Robust Decision Trees[vi] are competitive methods. Among them, XGBoost and LightGBM[vii] are shown to be more efficient in multiclusters classification for the dataset.

Xiao (2017)[viii] compares the performance of Fashion-MNIST and MNIST. The author tries to use 13 different algorithms like decision tree, Gradient Boosting, Neighbors, SVC, and so on, with different tuning parameters to construct the model. Among all of the classifiers, SVC with 10 as regularization parameter and radial basis function kernel tends to have the highest predicting accuracy(89.7%); Gradient Boosting with 100 as the number of boosting stages, 10 as the maximum depth of each regression estimator and logistic as loss function also reports high accuracy(88%). Though the accuracy of these algorithms is pretty good, their efficiency and scalability of them are not quite well when dealing with high dimensional data or large data sets.

Greeshma (2019) [ix] aims at handling the high dimension features and increase the accuracy of classification. Therefore, they used SVM classification based on the HOG(Histogram of Oriented Gradient) feature descriptor. HOG is a method used to discriminate data and compare different feature sets. It can divide one image into several small subsets, and use these subsets to do the work. Compared to other feature descriptor methods such as SIFT and LBP, HOG has simpler computation and is more efficient.

By selecting the appropriate cell size, HOG can help to limit the number of dimensions and speed up the training process. They also compare the result from other methods, SGD Classifier and SVM with the linear kernel. The best result of accuracy came from the combination of HOG and SVM, which is 86.53%. SVM has been applied in many different kinds of pattern recognitions successfully, and it is also suitable for the Fashion MNIST data set.

To our knowledge, without deep learning methods, the highest test accuracy for the Fashion MNIST dataset classification is 89.7%.[x]

SUMMARY STATISTICS, DATA PROCESSING, AND UNSUPERVISED LEARNING.

Fashion MNIST dataset consists of 60000 training samples and 10000 testing samples. Each record takes 28*28 pixels, representing an image belongs to one specific label.

Table 1: Frequency tables

	Label 0	Label 1	Label 2	Label 3	Label 4	Label 5	Label 6	Label 7	Label 8	Label 9
	T-shirt	Trouse	Pullover	Dress	Coat	Sandal	Shirt	Sneake	Bag	Ankle
Train	6000	6000	6000	6000	6000	6000	6000	6000	6000	6000
Test	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000

From the frequency table, we could see that each category is quite balanced distributing in both training and test sets. Take a closer look at the dataset, we visualize the pixels into figures.

Fig 1: Glimpse of the dataset

On the left of Fig1, we randomly plot 10 records in each category. On the right of Fig1, it is a picture of an ankle boot with specific pixel values on it.

We summarize main characteristic of the source dataset as: high-dimensional in features (28*28) and sample size (60000) and comb out according expectations for following model establishment: being confident in front of largish dataset with robustness. Now moves on to the unsupervised learning section, we conduct distance-based (K-means) and distribution-based (Gaussian Mixed Model).

K-means Clustering

In the K-means model, we need to determine the number of classes to classify first. Using square error plots to select the optimal number of k, when k is larger than 10, the value of square error decreases slowly, so we use k = 10 to fit the final K-means model.

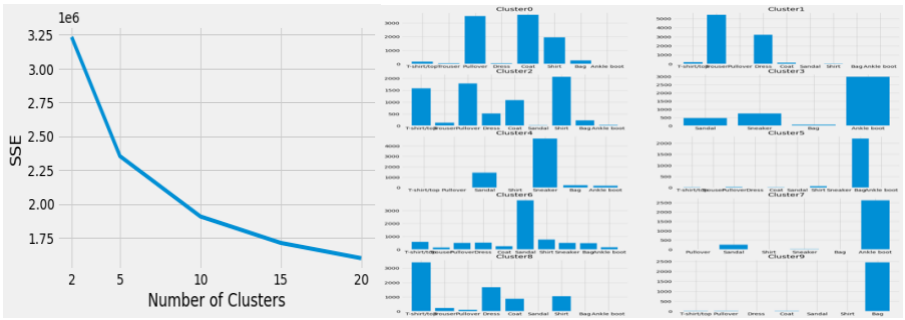

Fig 2: K-means output plots

If the initial selected cluster means are not suitable, the K-means will get the locally optimal results rather than global optimal results. Some clusters label the samples which have the same label, but some clusters such as cluster 0 and cluster 2 include several different kinds of labels.

Gaussian Mixed Model

GMM will calculate the probability of each kind of output first by assuming input variables and parameters are known, then treat input and output as known and solve the parameters by maximizing the likelihood.
GMM has many parameters to fit and it will need many iteration times to get the results. Only part of the samples is clustered into the class with these inputs with the same label.

Table 2: Confusion Table of Gaussian Mixed Model's result

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	1363	0	905	9	3328
1	4366	46	299	337	248	0	1156	0	295	0
2	721	516	417	3490	1183	6	542	0	381	3
3	127	2	3861	133	3746	0	2526	0	13	0
4	0	0	0	0	0	904	0	32	3	2216
5	3	0	0	0	0	3299	1	5048	327	425
6	55	5340	2	1802	33	0	19	0	33	0
7	74	5	56	18	6	384	112	1	66	7
8	296	7	134	22	57	34	412	14	3736	11
9	358	84	1231	198	727	10	1232	0	1137	10

No matter the MiniBatch-Kmeans or Gaussian Mixed Model, they are failing to separate the samples in a satisfying manner. So we discuss specialized machine learning in the following part.

MULTI-CLASS CLASSIFICATION MODEL

In the third part, we use LDA, SVM, and LightGBM to do multi-class classification.

Multiclass Classification Using Linear Discriminant Analysis (LDA)

A higher dimensionality can make the data become sparse, which also makes the model suffers from the curse of dimensionality and lower the accuracy. Since we have 784 features, LDA allows us to levitate that problem. The algorithm projects feature from higher-dimensional space into lower-dimensional space to avoid the curse of dimensionality and reduce dimensional costs.

LDA is a classification method that assumes data comes from multivariate Gaussian distribution, and the covariance matrix of every group is the same. We model the distribution of covariates in each class separately and use the Bayes theorem to flip things around the conditional probability. We then decide on the target group based on the group with the maximum value of the discriminant function.

	precision	recall	f1-score	support
0	0.801670	0.7680	0.784474	1000.0000
1	0.994709	0.9400	0.966581	1000.0000
2	0.755814	0.7150	0.734841	1000.0000
3	0.821698	0.8710	0.845631	1000.0000
4	0.746212	0.7880	0.766537	1000.0000
5	0.862403	0.8900	0.875984	1000.0000
6	0.585673	0.6050	0.595180	1000.0000
7	0.871560	0.8550	0.863200	1000.0000
8	0.937564	0.9160	0.926657	1000.0000
9	0.897233	0.9080	0.902584	1000.0000
accuracy	0.825600	0.8256	0.825600	0.8256
macro avg	0.827454	0.8256	0.826167	10000.0000
weighted avg	0.827454	0.8256	0.826167	10000.0000

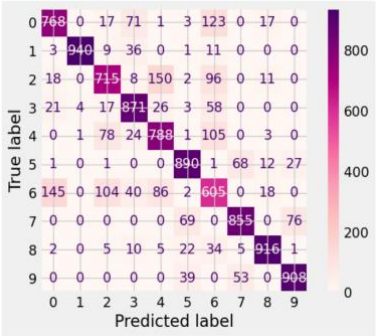

Fig 3: LDA model-fit result

However, LDA focuses on all data points and is not good at classifying points that are difficult to classify. Hence, we come up with SVM, which also focuses on the points that are difficult to classify.

Multiclass Classification Using Support Vector Machine

SVM classification allows us to deal with points that are hard to classify. However, the it can only support a binary classifier. Depending on the defined kernel function, the support vector machine does complex data transformation. It will try to maximize the separation boundaries between the data points according to the classes we defined.

There are two ways to gain mutual linear separation between every two classes to handle data points in high dimensions. One way is called a One-to-One approach, the same principle is utilized after breaking down the multi-classification problem into smaller subproblems, all of which are binary classification problems. Another method is One-to-Rest, divides the whole data set into two classes first, divides one of these two classes into two classes, then iterates to get more classes.
Here is an example of classifying three different colors of points using SVM. The left graph displays the One-to-One approach, there are separating lines between each pair of classes. The right graph displays the One-to-Rest approach, it will separate one kind of the point, such as blue points firstly, then separate the rest of the points, dividing yellow points and red points in the second step.

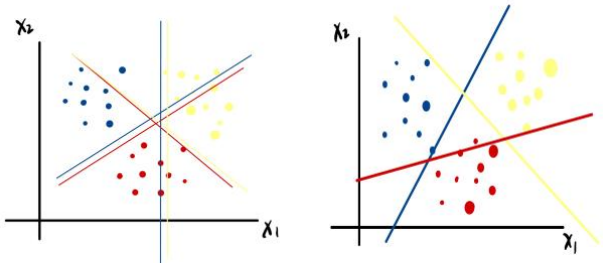

Fig 4: SVM extent to multiclassification

SVM is good at classifying the points which are close to the boundary between classes. Hence, we can see from the confusion matrix that SVM has an overall higher accuracy than the LDA classifier. We used the sum.SVC function from Scikitlean library to do the multiclass classifier. To select the kernel function and tuning parameter, we tried tuning regularization parameters equal to 1, 10, and 100, and selected 10 because of its highest accuracy. The selection kernel function is polynomial, while the linear kernel has lower accuracy.

	precision	recall	f1-score	support
0	0.803172	0.8610	0.831081	1000.0000
1	0.987830	0.9740	0.980866	1000.0000
2	0.831887	0.7670	0.798127	1000.0000
3	0.898696	0.8960	0.897346	1000.0000
4	0.853191	0.8020	0.826804	1000.0000
5	0.821552	0.9530	0.882407	1000.0000
6	0.709419	0.7080	0.708709	1000.0000
7	0.921026	0.8980	0.909367	1000.0000
8	0.976313	0.9480	0.961948	1000.0000
9	0.958121	0.9380	0.947954	1000.0000
accuracy	0.874500	0.8745	0.874500	0.8745
macro avg	0.876121	0.8745	0.874461	10000.0000
weighted avg	0.876121	0.8745	0.874461	10000.0000

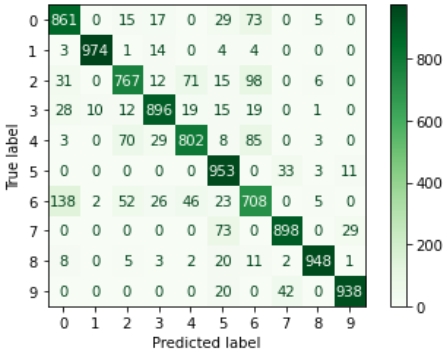

Fig 4: SVM model-fit result

Though SVM does well at classifying the points close to the boundary between classes, SVM is not suitable for large data sets since the training complexity of SVM is very high. The problem can be solved by the mechanism of LightGBM, which is efficient in dealing with large data sets.

Multiclass Classification Using LightGBM

LightGBM is a high-performance gradient boosting framework based on a decision tree algorithm, it splits the tree leaf-wise rather than depth-wise or level-wise, which can reduce loss and result in higher accuracy. However, it can also lead to overfitting. Hence, we need to tune the model wisely. There are some parameters like max depth, bagging fraction, min data in leaf, num leaves, and path smooth that we can tune to get our best model. Here we tune the path smooth over a grid that starts from 0 to 1 and is separated by 0.1 and set the min data in leaf to 2. Graph shows the result of the tuning, we get the minimum classification error when the path_smooth equals 0.5.

	precision	recall	f1-score	support
0	0.835735	0.8700	0.852523	1000.0000
1	0.991927	0.9830	0.987443	1000.0000
2	0.841046	0.8360	0.838516	1000.0000
3	0.912957	0.9230	0.917951	1000.0000
4	0.853491	0.8680	0.860684	1000.0000
5	0.991761	0.9630	0.977169	1000.0000
6	0.754497	0.7130	0.733162	1000.0000
7	0.950000	0.9690	0.959406	1000.0000
8	0.976096	0.9800	0.978044	1000.0000
9	0.962227	0.9680	0.965105	1000.0000
accuracy	0.907300	0.9073	0.907300	0.9073
macro avg	0.906974	0.9073	0.907000	10000.0000
weighted avg	0.906974	0.9073	0.907000	10000.0000

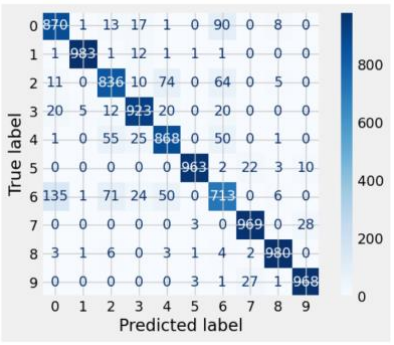

Fig 5: LightGBM model-fit result

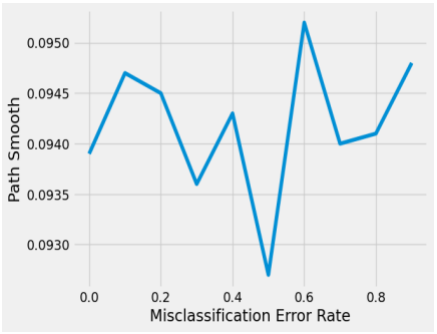

Fig 6: LightGBM tuning parameters

Fig 5 shows the misclassification rate of each model. The performance of these three models is pretty good, all of their error rates are lower than 20%. Especially for LightGBM, the accuracy is up to 89%. Table 3 shows the accuracy of each label of these three models. All these three algorithms can classify labels 1, 5, 8, and 9 really well. LightGBM has the highest accuracy in predicting label 0 and label 3 while SVM can predict label 4 and label 7 very well.

Table 3: Comparison of three multiclassification models result

	0	1	2	3	4	5	6	7	8	9	Error Rate
LightGBM	0.84	0.99	0.84	0.91	0.85	0.99	0.75	0.95	0.98	0.96	0.0927
LDA	0.80	0.99	0.76	0.82	0.75	0.86	0.59	0.87	0.94	0.90	0.1744
SVM	0.80	0.99	0.83	0.90	0.85	0.82	0.71	0.92	0.98	0.96	0.1255

ENSEMBLE MODEL AND FEATURE ENGINEERING

Model Establishment

SVM is effective in solving classification n problems with high-dimensional features, but sensitive to the choice of kernel functions. LightGBM has improved the speed and memory consumption in the XGboost algorithm, but still has problems with overfitting risk and noise sensitive. There is no one-fit-all model but the optimal choice under a trade-off.

Since every single model has its own applicability and limitations, it is natural to think about integrating strengths to complement each other to form more robust results, which is in line with the idea that two heads are better than one. Out of that, we resort to ensemble learning, of which the basic idea is to reduce the variance through the integration of multiple models.

The algorithmic strategy of ensemble learning can be summarized as follows：

- Train several individual models (usually regarded as a weak learners)
- Take a combination strategy to form a strong learner

Implementation

Developed from the multiclassification part, we take the Linear Discrimination Analysis, Support Vector Machine and LightGBM three models with their tuning parameters afterward as our base learners. Explore three different combination strategies to accomplish model integration work with open-source library scikit-learn.

Evaluation

There are many evaluation metrics in multiclassification prediction problems: precision, recall, and f1-score, to name a few. We focus on their prediction accuracy on the test set as a criterion when comparing models.

Majority Voting

The first method that comes into our mind is Voting, specifically in our project, we take Majority Voting as our integration procedure. Each of the individual classifiers votes for a label, and the majority wins.

	precision	recall	f1-score	support
0	0.808511	0.874	0.839981	1000.00
1	0.991870	0.976	0.983871	1000.00
2	0.836588	0.814	0.825139	1000.00
3	0.901381	0.914	0.907646	1000.00
4	0.849246	0.845	0.847118	1000.00
5	0.914505	0.952	0.932876	1000.00
6	0.739267	0.706	0.722251	1000.00
7	0.929878	0.915	0.922379	1000.00
8	0.974645	0.961	0.967774	1000.00
9	0.955420	0.943	0.949170	1000.00
accuracy	0.890000	0.890	0.890000	0.89
macro avg	0.890131	0.890	0.889821	10000.00
weighted avg	0.890131	0.890	0.889821	10000.00

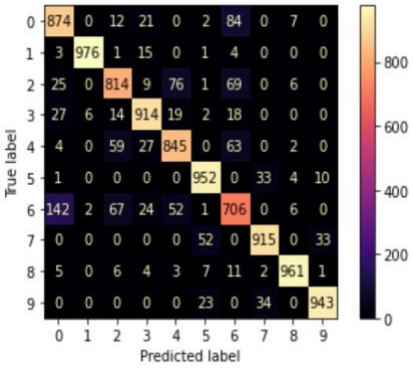

Fig 6: Majority Voting ensemble model result

The test accuracy of Majority Voting is 0.89, which is higher than SVM but lower than LightGBM. At this point, we could not have much confidence to say that this method brings significant improvement. And in some sense, the result is acting as an “averaging”.

Voting++ (User Defined Function)

Interestingly deserves to notice that, each base classifier performs variedly in predicting different fashion categories. For instance, LightGBM LGM seems to be particularly excellent at classifying scandals with a 99% precision (versus an 84% precision in the SVM model).

However, its prediction performance is inferior to that of SVM in classifying coats. In view of this, we are inspired to propose a method called Voting++ to improve the voting procedure. A high precision capacitates the model with a higher discourse power(weight) on the board. Concretely, for each sample, the vote with the highest weight is taken as the final prediction.

The algorithm flow is as follows.

- Train the base classifier and tune the parameters separately
- Construct a weight reference table based on the precisions of the base classifier in different fashion categories.
- Get the weight calculated from step2 for each model in each sample.
- For each sample, the vote with the highest weight is taken as the final result.

Experiment result and interpretation
The test accuracy of Voting++ is 0.884, improving the performance of the Majority Voting method. It can also be intuitively seen from the figure, with darker colors in the diagonals.

	precision	recall	f1-score	support
0	0.788732	0.896	0.838951	1000.000
1	0.982090	0.987	0.984539	1000.000
2	0.806420	0.829	0.817554	1000.000
3	0.867470	0.936	0.900433	1000.000
4	0.825000	0.891	0.856731	1000.000
5	0.963037	0.964	0.963518	1000.000
6	0.873333	0.524	0.655000	1000.000
7	0.952381	0.960	0.956175	1000.000
8	0.940556	0.981	0.960352	1000.000
9	0.952941	0.972	0.962376	1000.000
accuracy	0.894000	0.894	0.894000	0.894
macro avg	0.895196	0.894	0.889563	10000.000
weighted avg	0.895196	0.894	0.889563	10000.000

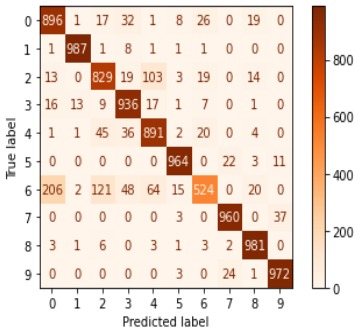

Fig 7: Voting++ ensemble model result

The test accuracy of Voting++ is 0.894, improving the performance of the Majority Voting method. It can also be intuitively seen from the figure, with darker colors in the diagonals.

What we have discussed above is based on the assumption that a series of individual learners are generated in parallel. Next, we try another direction by sequentially building multiple learners for multiple learning periods.

Stacking

The Stacking algorithm is like adding a layer after Bagging.

- A number of independent base learners (first-level learners) are generated by calling the base learning algorithm
- These independent base learners are then combined by a meta-learner (secondary learner) in such a way that the results of the primary learner are used as input and the labels are used as output for training

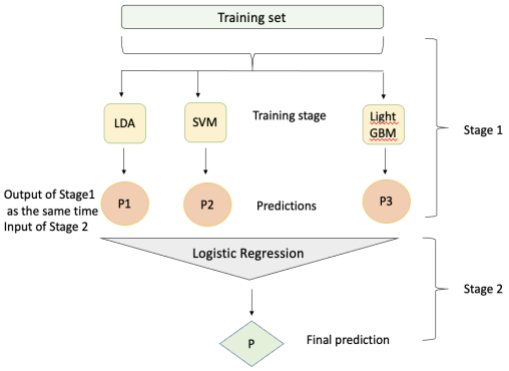

Fig 8: Stacking ensemble model workflow

Again, we take Linear Discrimination Analysis, Support Vector Machine and LightGBM three models as our base learners. And take Logistic regression as our meta-classifier. The stacking model performance on test datasets summarized as below:

Obviously, the stacking model outperforms all previous models with a highest prediction accuracy 0.9083.

Conclusion

Recap on the main work of this paper:

1. Conduct exploratory data analysis on Fashion MNIST dataset and summarize main characteristic of the source dataset as: high-dimensional in features(28*28) and sample size (60000).
2. Perform MiniBatch-Kmeans and Gaussian Mixture Model in clustering analysis
3. Establish Linear Discriminant analysis, Support vector machine and LightGBM three multiplication models. (including derivation from binary case)
4. Ensemble base classifiers (obtained in part3) with Majority Voting, Voting++ and Stacking. Improve the test accuracy to **(to be continued)**

[i] Espadoto, M., Martins, R. M., Kerren, A., Hirata, N. S., & Telea, A. C. (2019). Toward a quantitative survey of dimension reduction techniques. *IEEE transactions on visualization and computer graphics*, *27*(3), 2153-2173.

[ii] Yang, X., Deng, C., Zheng, F., Yan, J., & Liu, W. (2019). Deep spectral clustering using dual autoencoder network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 4066-4075).

[iii] Allaoui, M., Kherfi, M. L., & Cheriet, A. (2020, June). Considerably improving clustering algorithms using UMAP dimensionality reduction technique: a comparative study. In *International Conference on Image and Signal Processing* (pp. 317-325). Springer, Cham.

[iv] Agarap, A. F. (2017). An architecture combining convolutional neural network (CNN) and support vector machine (SVM) for image classification. *arXiv preprint arXiv:1712.03541*.

[v] Liu, J., Zheng, Y., Dong, K., Yu, H., Zhou, J., Jiang, Y., ... & Ding, R. (2020). Classification of fashion article images based on improved random forest and VGG-IE algorithm. International Journal of Pattern Recognition and Artificial Intelligence, 34(03), 2051004.]

[vi] Chen, H., Zhang, H., Boning, D., & Hsieh, C. J. (2019, May). *International Conference on Machine Learning* (pp. 1122-1131). PMLR.

[vii] Andriushchenko, M., & Hein, M. (2019). Provably robust boosted decision stumps and trees against adversarial attacks. *Advances in Neural Information Processing Systems*, *32*.

[viii] Xiao, Han, Kashif Rasul, and Roland Vollgraf. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms." arXiv preprint arXiv:1708.07747 (2017).

[ix] Greeshma, K. V., & Sreekumar, K. (2019). Fashion-MNIST classification based on HOG feature descriptor using SVM. *International Journal of Innovative Technology and Exploring Engineering*, *8*(5), 960-962.