Assignmet 3 – Image Classification & Dimensionality Reduction

Introduction

In this experiment, it is going to be used 5 different classification model (K-Neighbors, Perceptron, SVM, Decision Trees and Random Forests) to classify image data. Also, PCA algorithm is going to be used for dimension reduction.

K-Neighbors

KNN is a nonparametric model which cannot be characterized by a fixed set of parameters, and the number of parameters grows with the training data. In addition, KNN is an instance-based learning that memorize all training data instead of learning a discriminative function. As seen in" Figure-1" the algorithm chooses the number of k as a distance metric and finds k-nearest neighbors tot the instance and returns the majority class label.

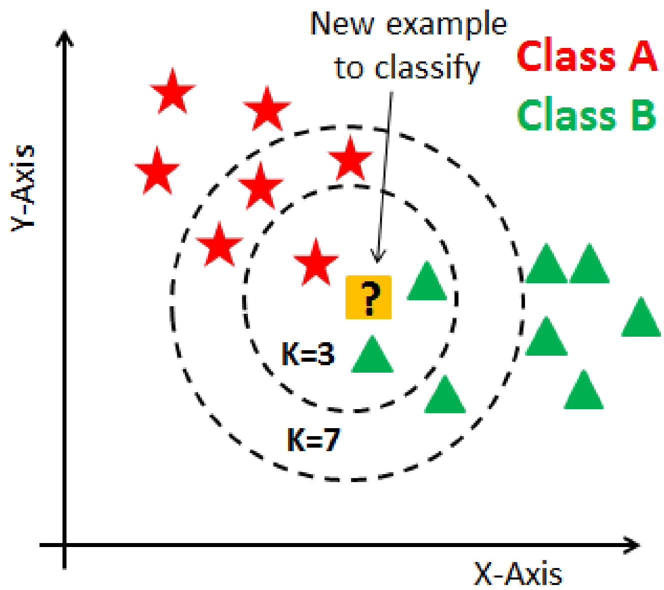

Figure 1 – KNN Example

Perceptron

Perceptron algorithm is a type of neural network model, it consists of a single node or neuron that takes a row of data as input and predict class label. This is achieved by calculating weighted sum of inputs and a bias. The weighted sum of the input of the model is called the activation ("Figure-2"). If the activation is above 0.0, the model will output 1.0; otherwise, it will output 0.0.

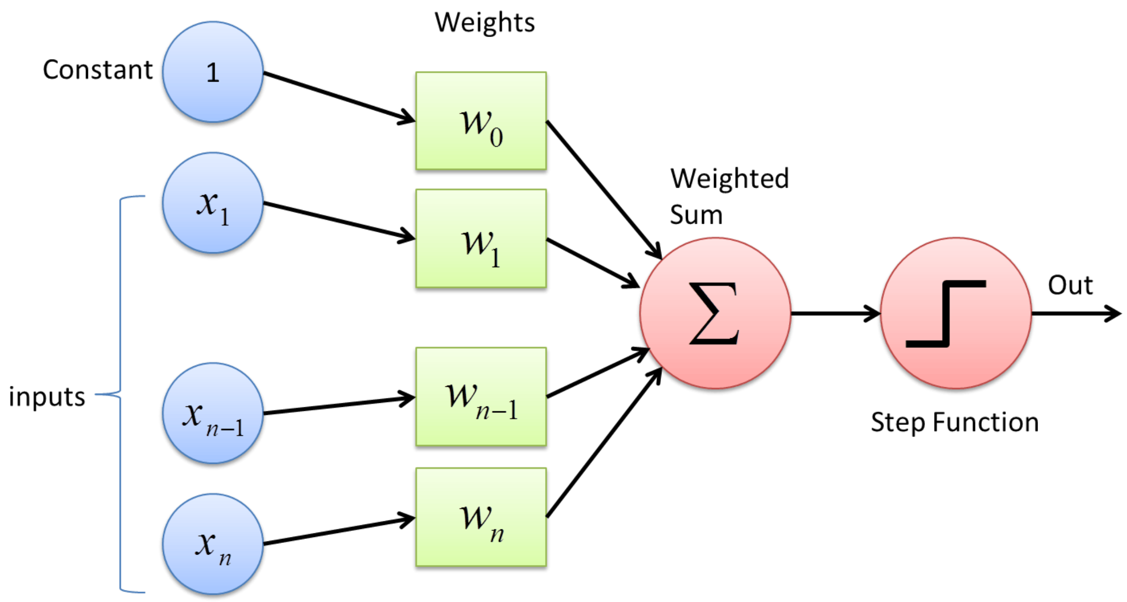

Figure 2 – Perceptron Example

SVM (Supporting Vector Machines)

SVM's objective is to maximize the margin – the distance between the separating hyperplane (decision boundary). The training samples are closest to this hyperplane, which are the so-called support vectors ("Figure-3").

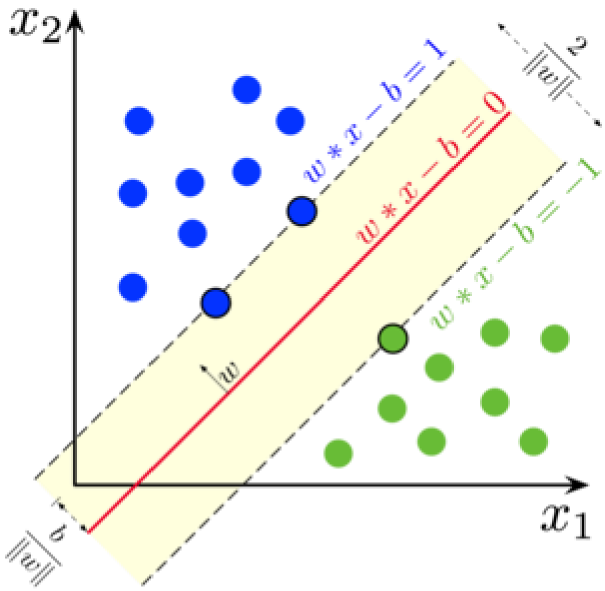

Figure 3 – SVM Examples

Decision Tree

A decision tree is a flowchart-like tree structure where an internal node represents feature (or attribute), the branch represents a decision rule, and each leaf node represents the outcome ("Figure-4"). The algorithm idea is simple, choose the most informative tree boundaries and split the data into two pieces.

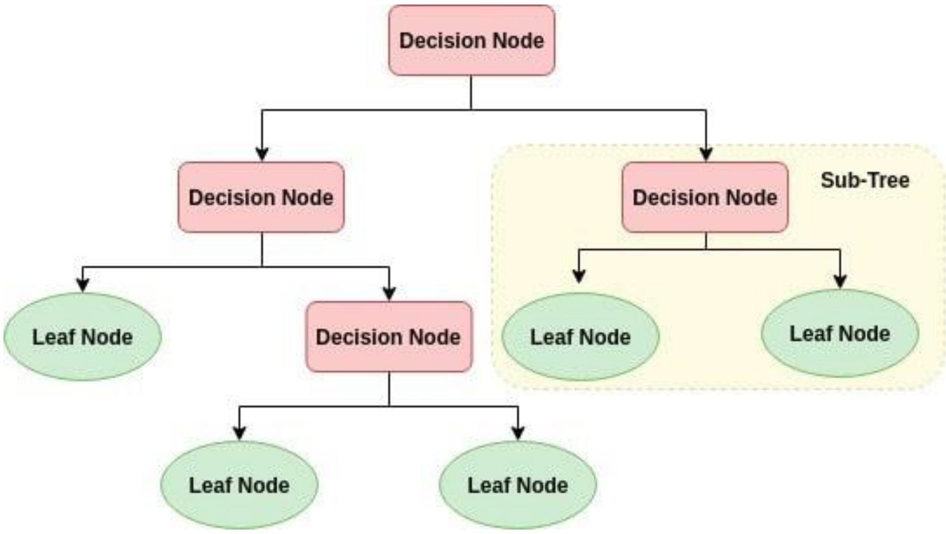

Figure 4 – Decision Trees Example

Random Forests

A random forest can be considered as an ensemble of decision trees. It works in four steps:

- Select random samples from a given dataset.
- Construct a decision tree for each sample and get a prediction result from each decision tree.
- Perform a vote for each predicted result.
- Select the prediction result with the most votes as the final prediction.

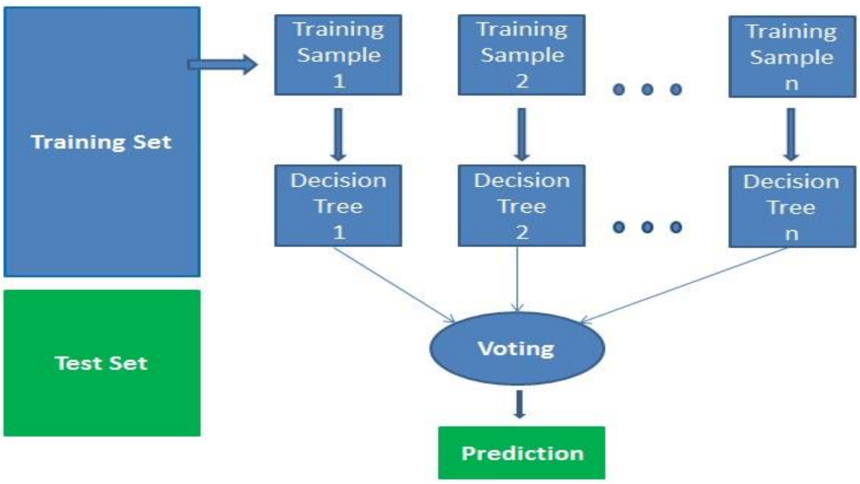

Figure 5 – Random Forest Working Mechanism

PCA

PCA is one of the most widely used dimension reduction techniques to transform the larger dataset into a smaller dataset identifying the correlations and patterns with preserving most of the valuable information

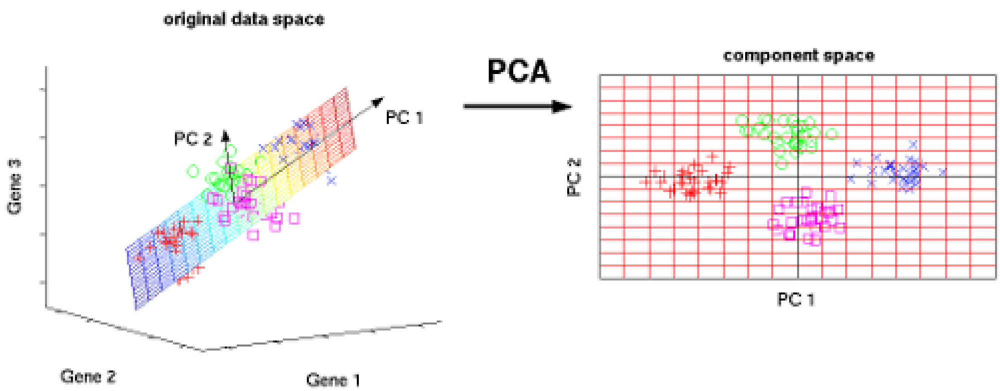

Figure 6 – PCA Example

Experiment Setup

In this experiment, Fashion MNIST data ,that includes 60,000 training samples and 10,000 test samples images and 10 image labels are going to be used. As seen in "Figure-7" images are grayscale and their size are 28px ,28px (width & height) . In order to get high performance, image data should be scaled which means image arrays should be divided by "255.0". Also, for processing data shape should be number of images in dataset, and width * height that means (60000,784) for train dataset and (10000,784) for test dataset.

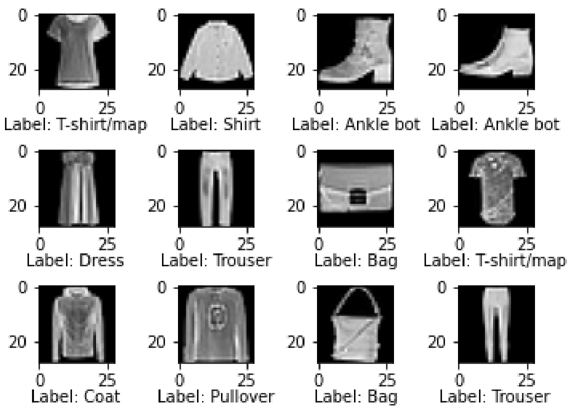

Figure 7 – Sample of MNIST Fashion Dataset

Implementation Details

After preprocessing the data, it should be implemented five different classification model, which is mentioned above, without any dimension reductions.

KNN Performance (Original Image)

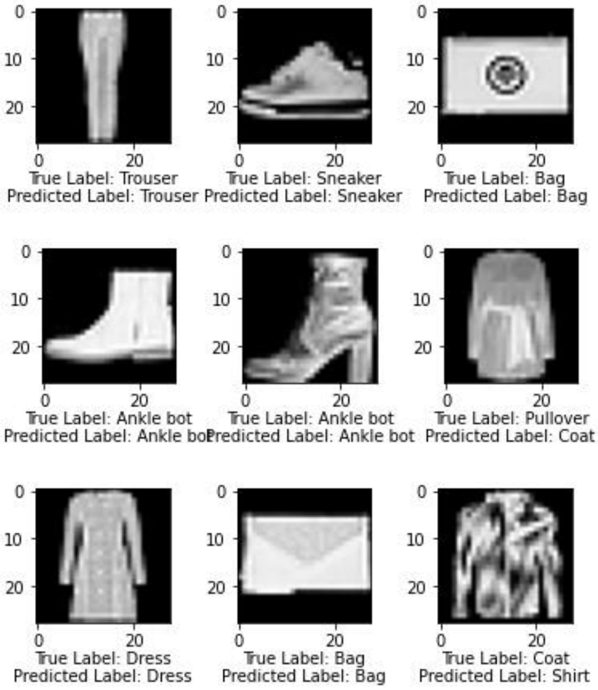

Figure 8 – Sample Image Results of KNN Model

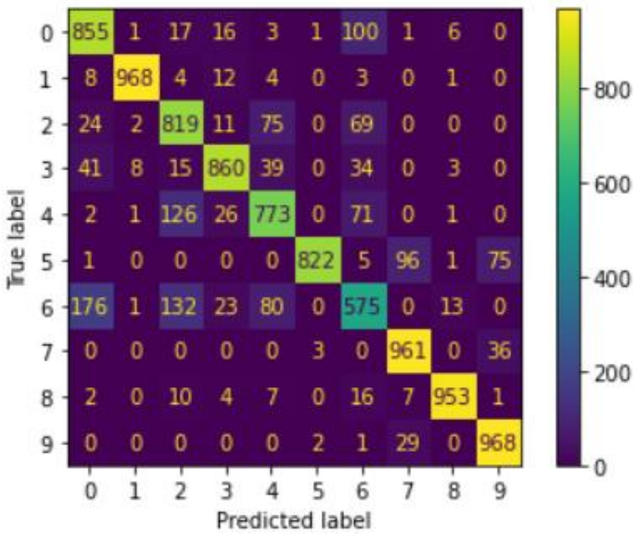

Figure 9 – KNN Confusion Matrix

Perceptron (Original Image)

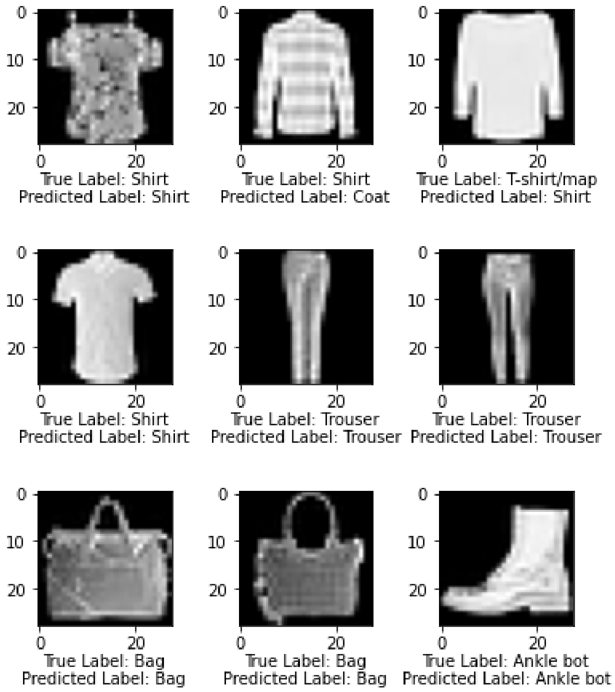

Figure 10 – Sample Image Results of Perceptron Model

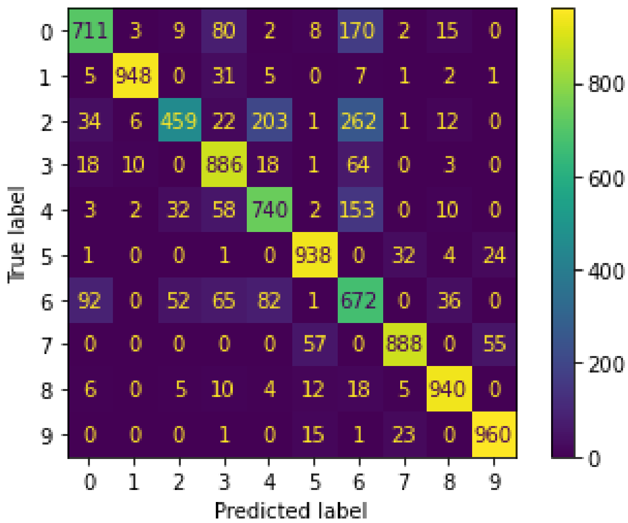

Figure 11 – Perceptron Confusion Matrix

SVM Performance (Original Image)

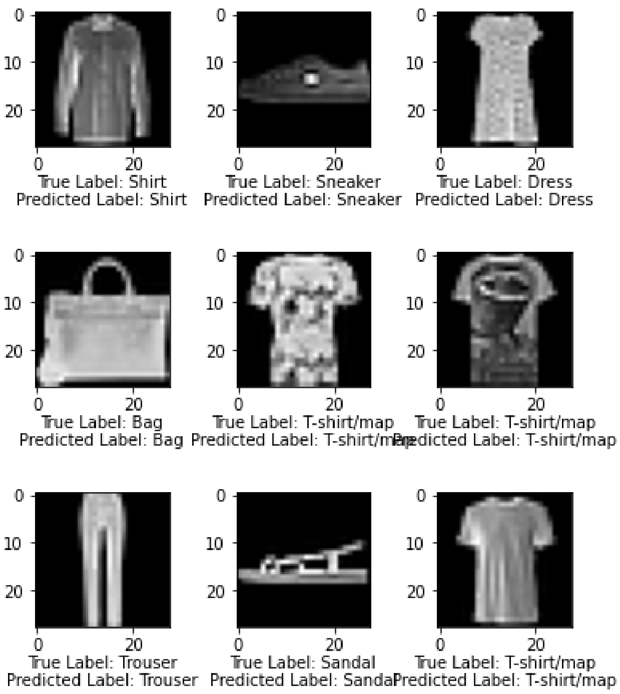

Figure 12 – Sample Image Results of SVM Model

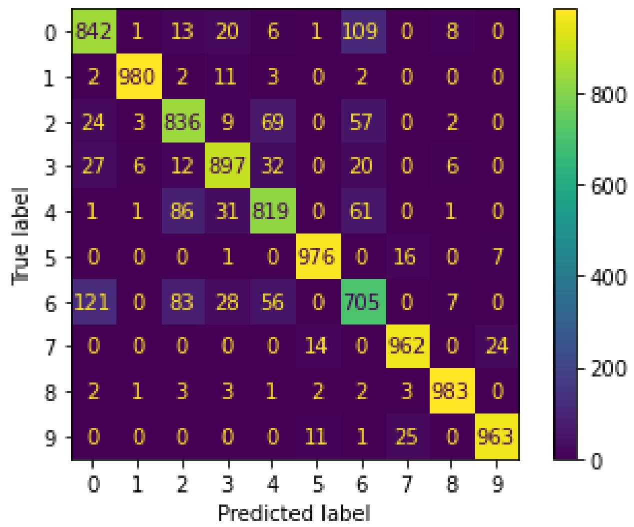

Figure 13 – SVM Confusion Matrix

Decision Trees (Original Image)

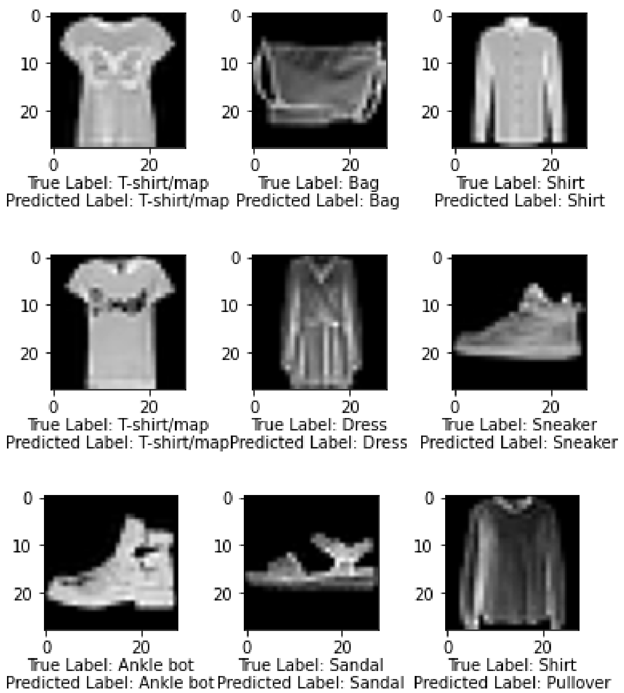

Figure 14 – Sample Image Results of Decision Trees Model

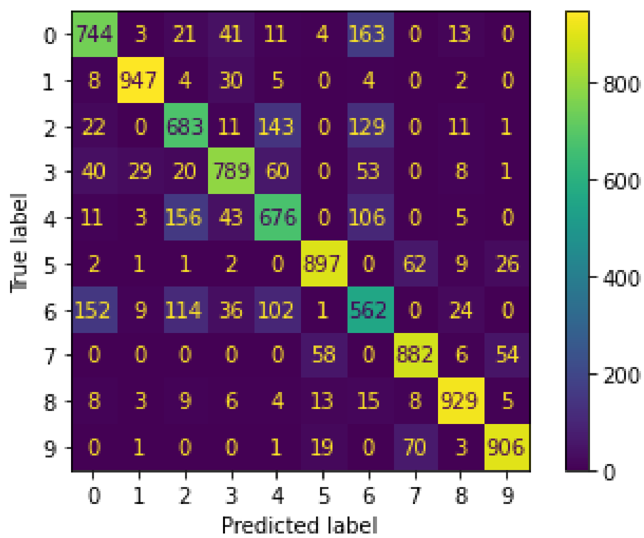

Figure 15 – Decision Trees Confusion Matrix

Random Forests (Original Image)

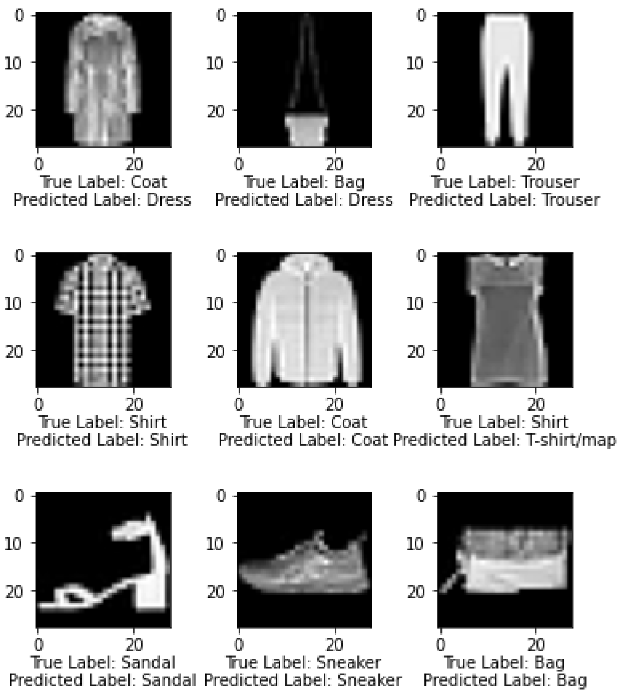

Figure 16 – Sample Image Results of Random Forests Model

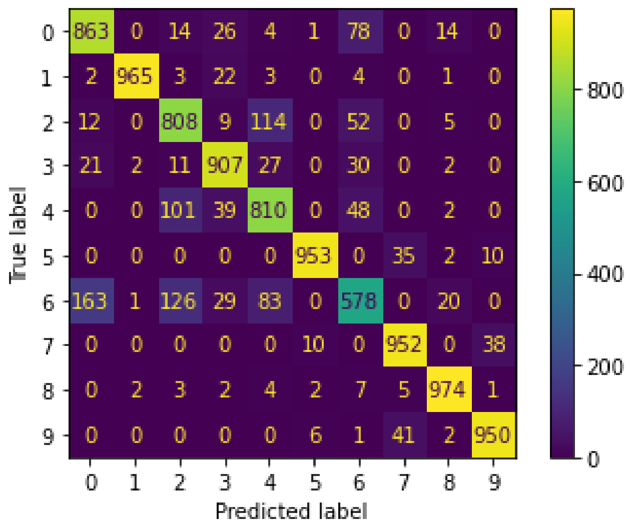

Figure 17 – Random Forests Confusion Matrix

By looking the Confusion Matrix of the 5 different classification model and the "Figure-18", it can be stated that the best accurate model is "SVM" (Supporting Vector Machine) and the worst accurate model is "Decision Tree" model.

Model Name	**Accuracy Score (%)**	**Misclassified Label in 10,000 labels**
K - Neighbors	85.54	1446
Perceptron	81.42	1858
SVM	89.63	1037
Decision Trees	80.15	1985
Random Forest	87.60	1240

Table 18 – Accuracy Score and Misclassified Label in 5 different classification model

After evaluating the 5 different classification model with original images, PCA technique as a dimension reduction should be implemented to the image data. Moreover, to find the number of components to explain P%, which should approximate 25,50,75,95, of the dataset it should be created explained variance ratio table. ("Figure-19" & "Figure-20")

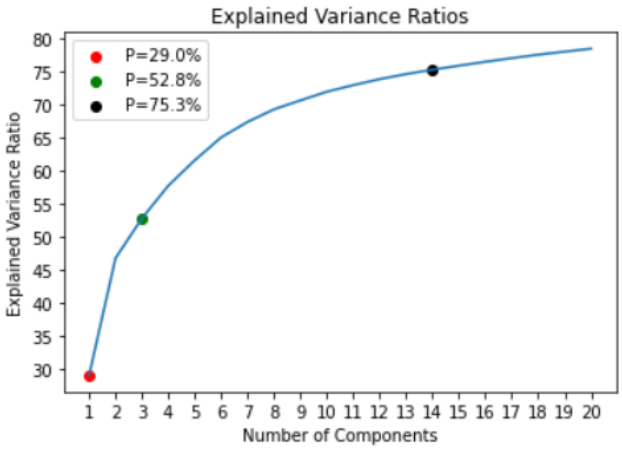

Figure 19 – Explained Variance Ratios Graph for P = 25,50,75 %

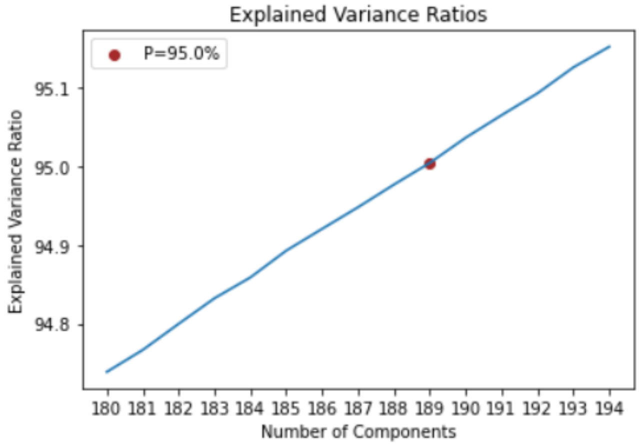

Figure 20 – Explained Variance Ratios Graph for P = 95 %

According to explained variance ratios graphs and “Figure-21”, in order to get specific P results number of components should be picked as seen in the table.

P	Explained Variance Results	Number of Components
25%	29.04%	1
50%	52.81%	3
75%	75.32%	14
95%	95.0%	189

Table 21 – Explained Variance Results and Number of Components according to P value

Number of Components = 1

To reach the P value approximately 25% dataset dimension should be reduced to 1 dimension. After the reduction, 5 different classification model should be implemented the same as the original image.

KNN Performance (Number of Components = 1)

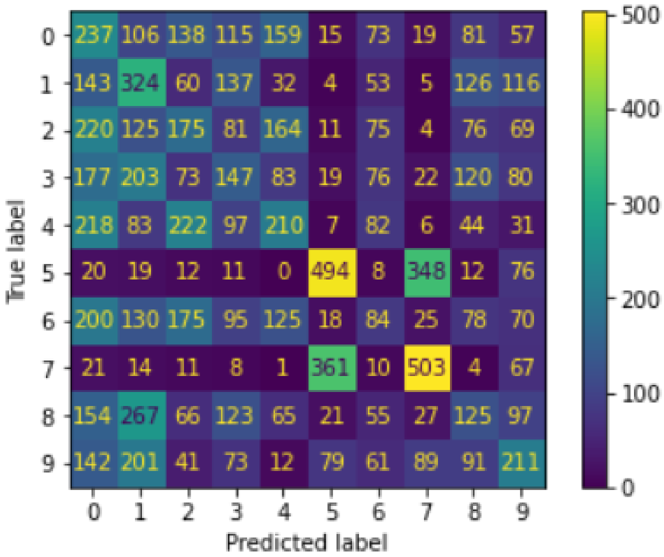

Figure 22 – KNN Confusion Matrix in 1D dataset

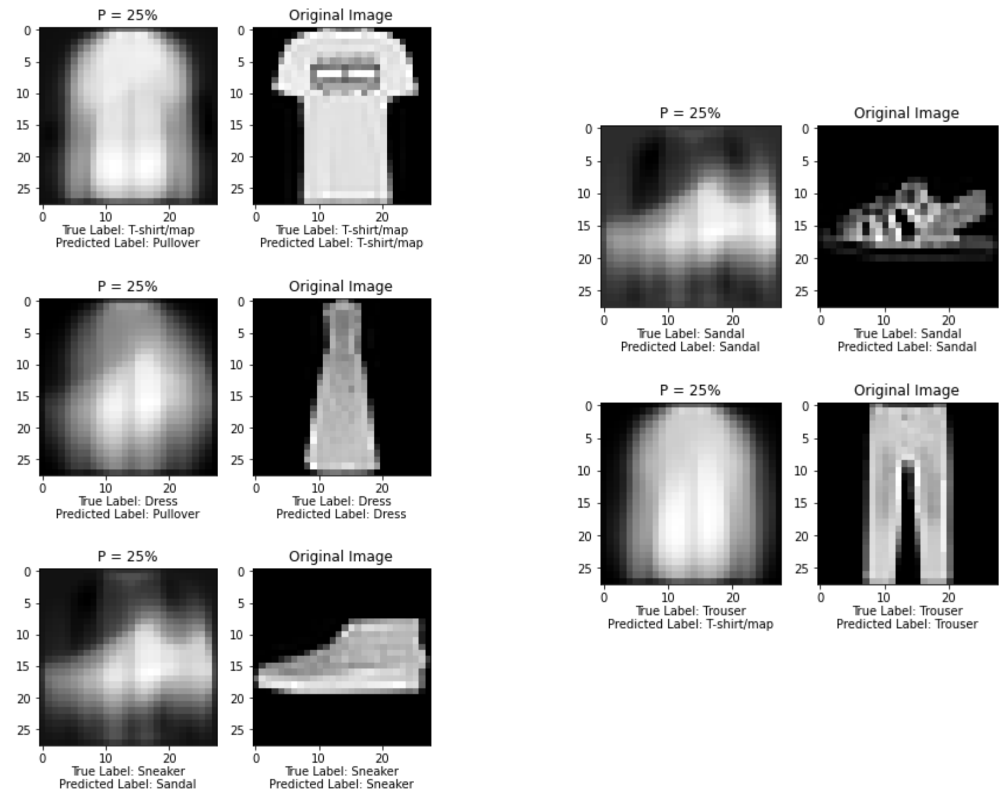

Figure 23 – Sample Results of Comparing Original Image Results and P = 25%

Perceptron Performance (Number of Components = 1)

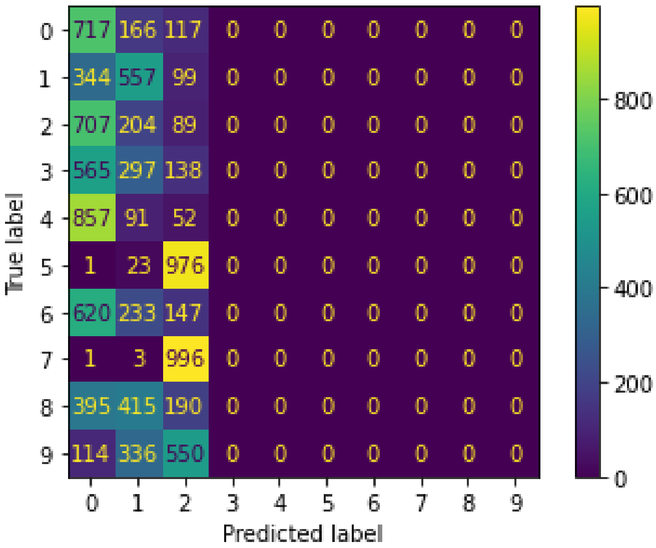

Figure 24 – Perceptron Confusion Matrix in 1D dataset

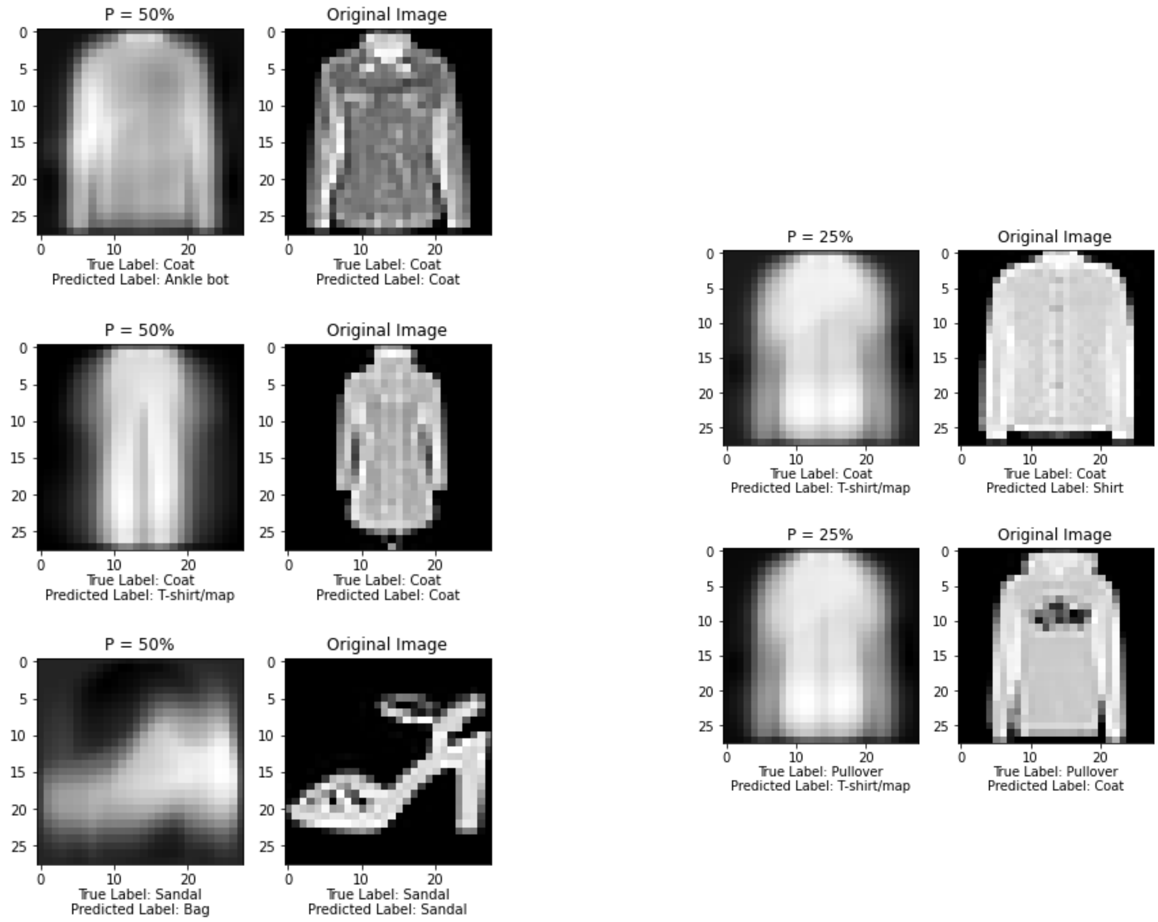

Figure 25 – Sample Results of Comparing Original Image Results and P = 25%

SVM Performance (Number of Components = 1)

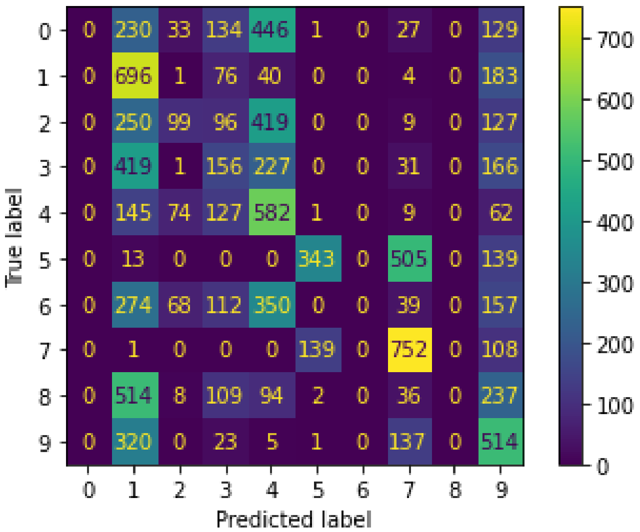

Figure 26 – SVM Confusion Matrix in 1D dataset

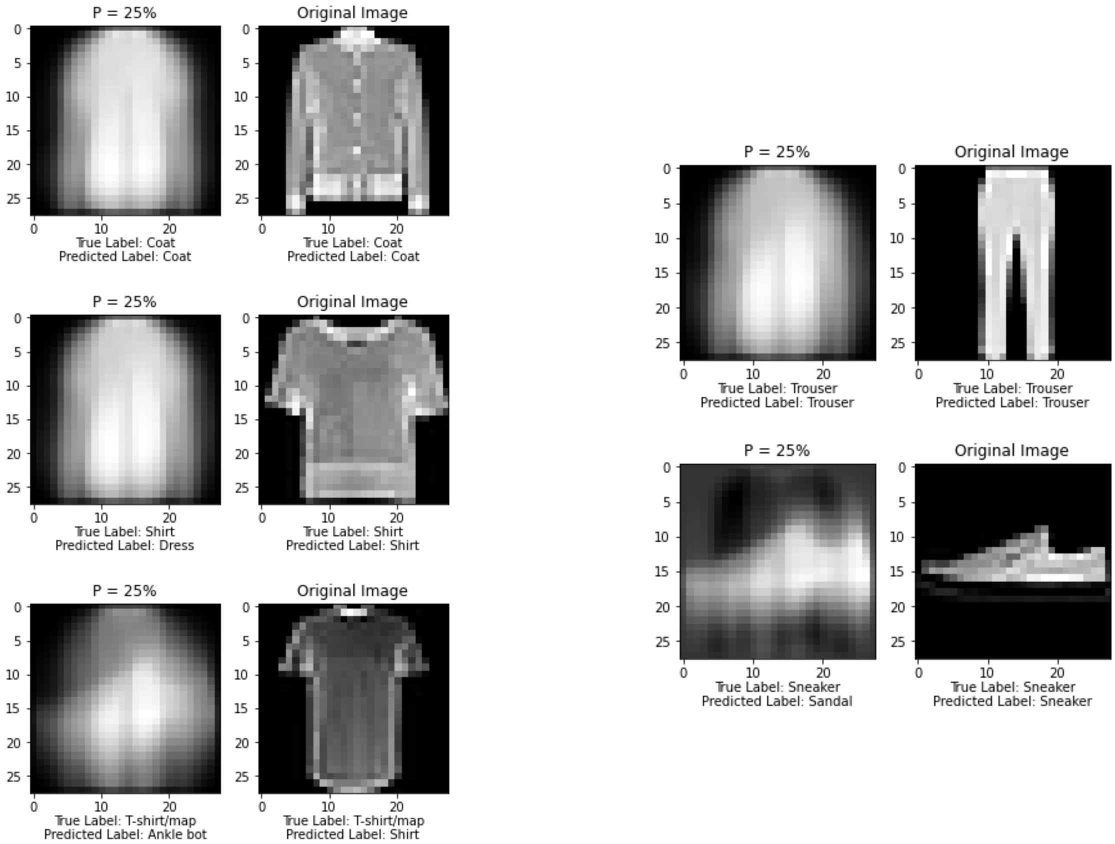

Figure 27 – Sample Results of Comparing Original Image Results and P = 25%

Decision Trees Performance (Number of Components = 1)

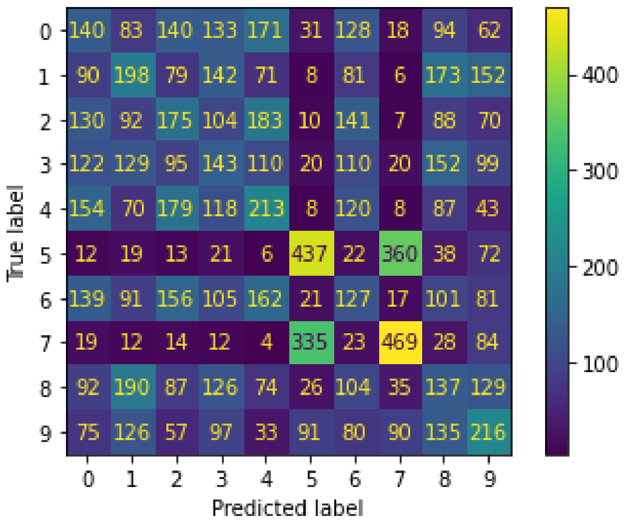

Figure 28 – Decision Trees Confusion in 1D dataset

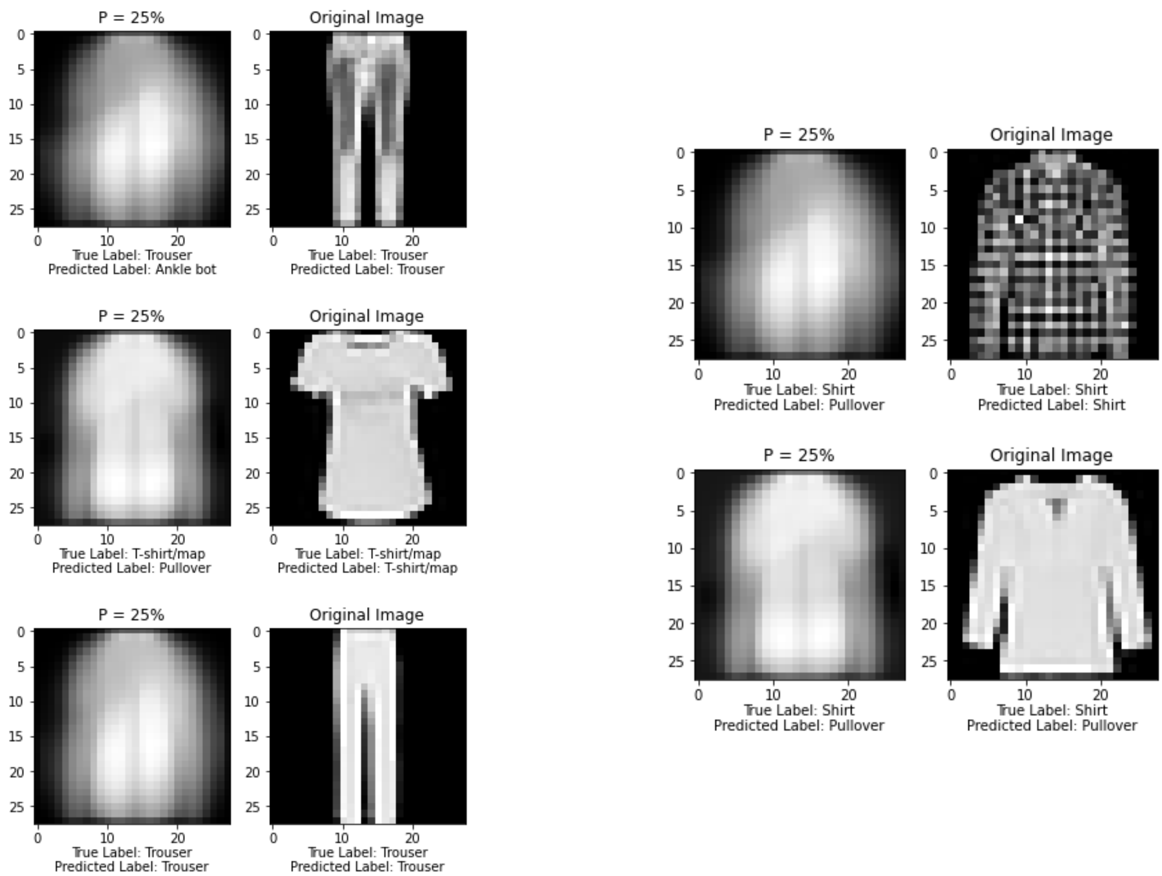

Figure 29 – Sample Results of Comparing Original Image Results and P = 25%

Random Forests Performance (Number of Components = 1)

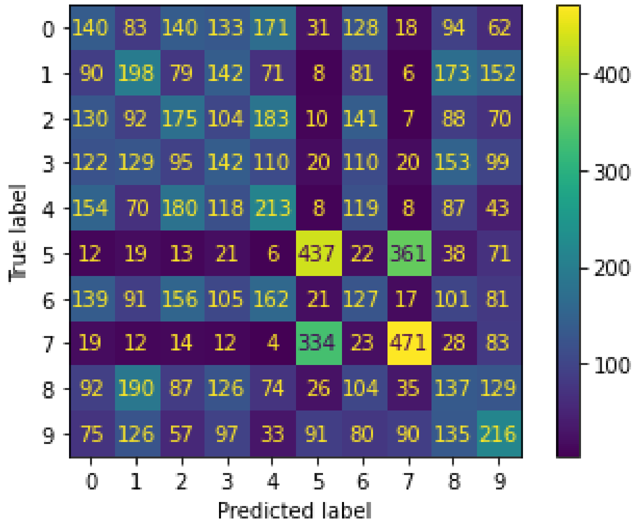

Figure 30 – Random Forests Confusion Matrix in 1D dataset

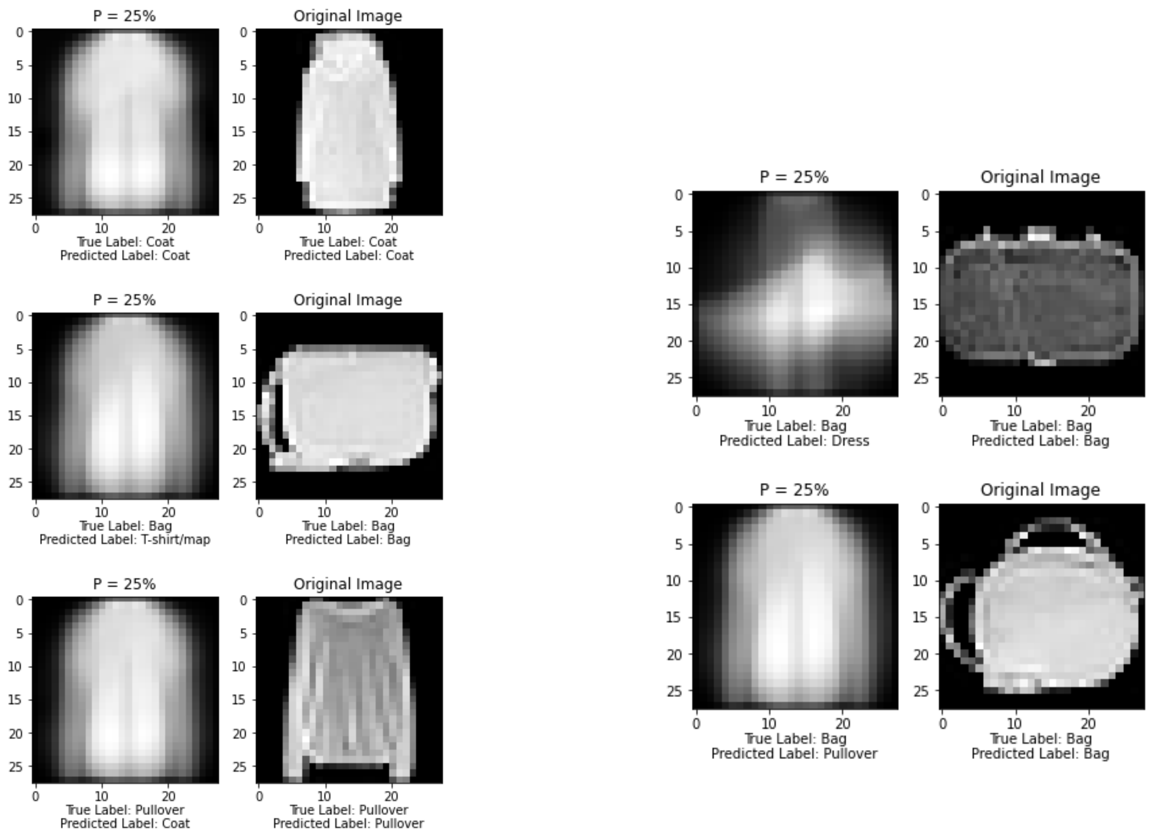

Figure 31 -Sample Results of Comparing Original Image Results and P = 25%

Number of Components = 3

To reach the P value approximately 50% dataset dimension should be reduced to 3 dimensions. After the reduction, 5 different classification model should be implemented the same as the original image.

KNN Performance (Number of Components = 3)

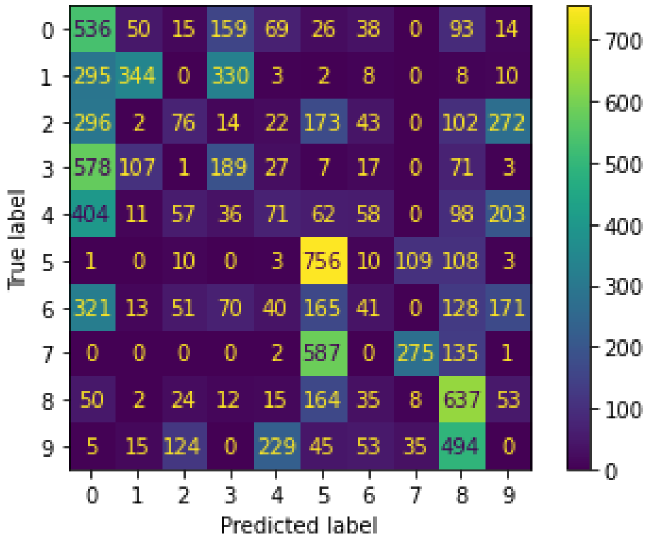

Figure 32 – KNN Confusion Matrix in 3D dataset

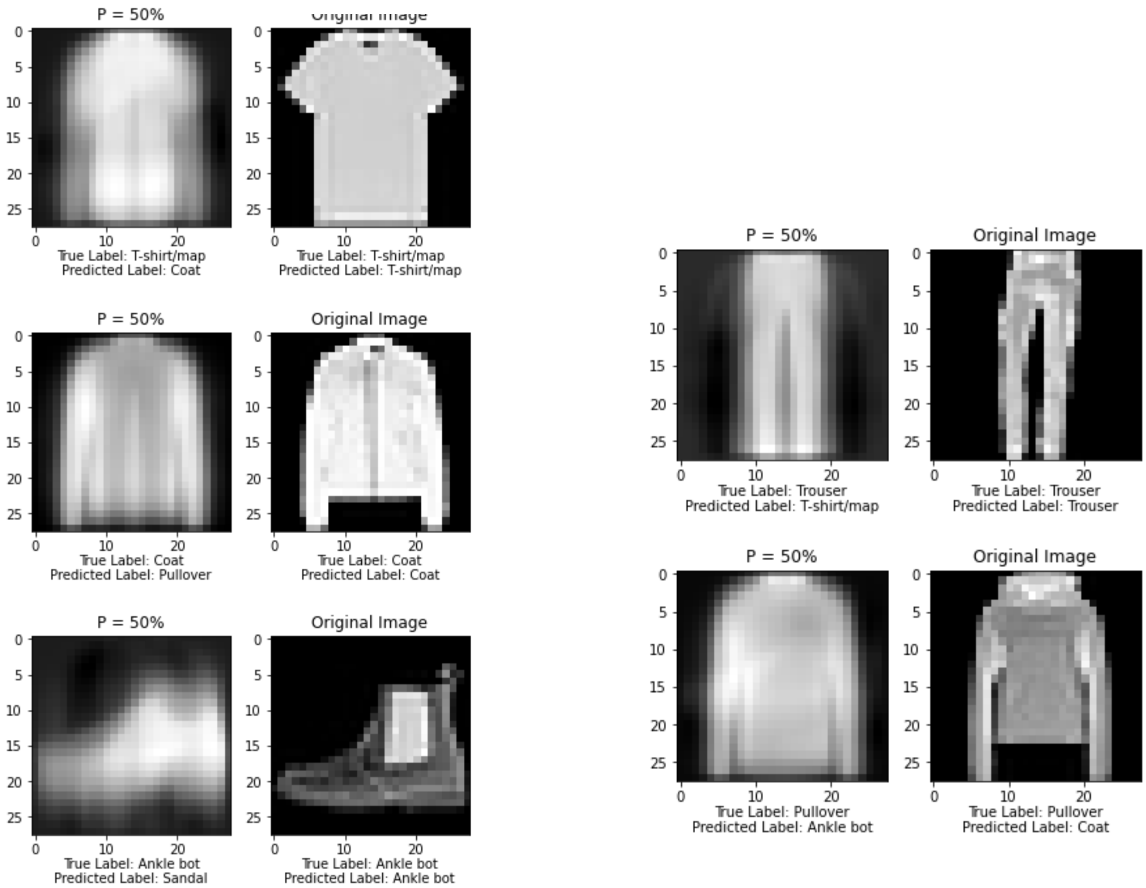

Figure 33 – Sample Results of Comparing Original Image Results and P = 50%

Perceptron Performance (Number of Components = 3)

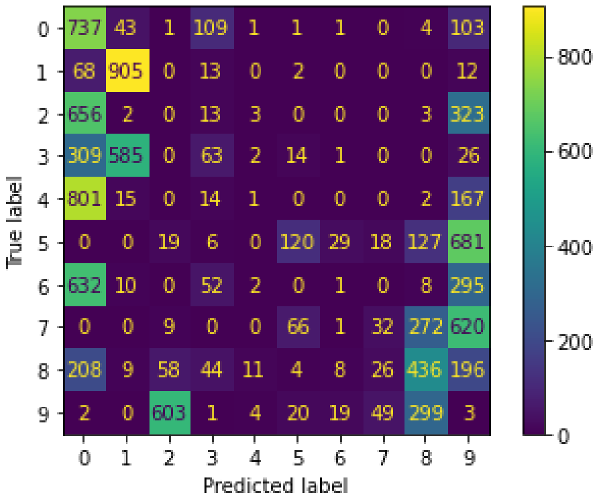

Figure 34 – Perceptron Confusion Matrix in 3D dataset

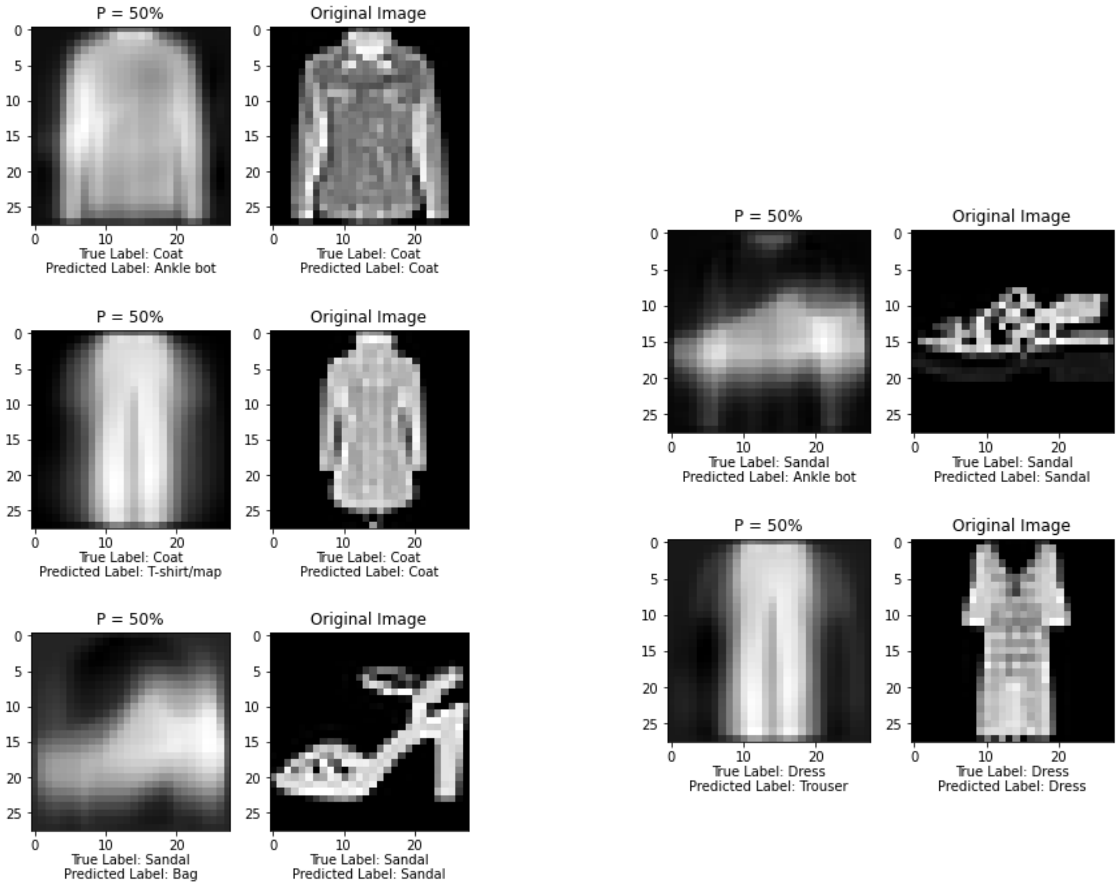

Figure 35 – Sample Results of Comparing Original Image Results and P = 50%

SVM Performance (Number of Components = 3)

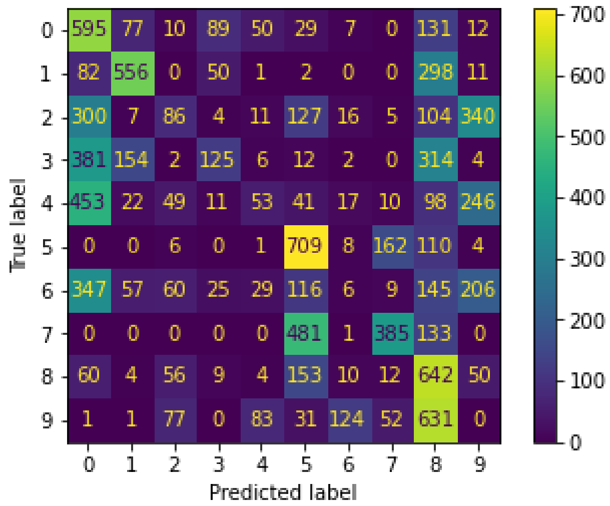

Figure 36 – SVM Confusion Matrix in 3D Dataset

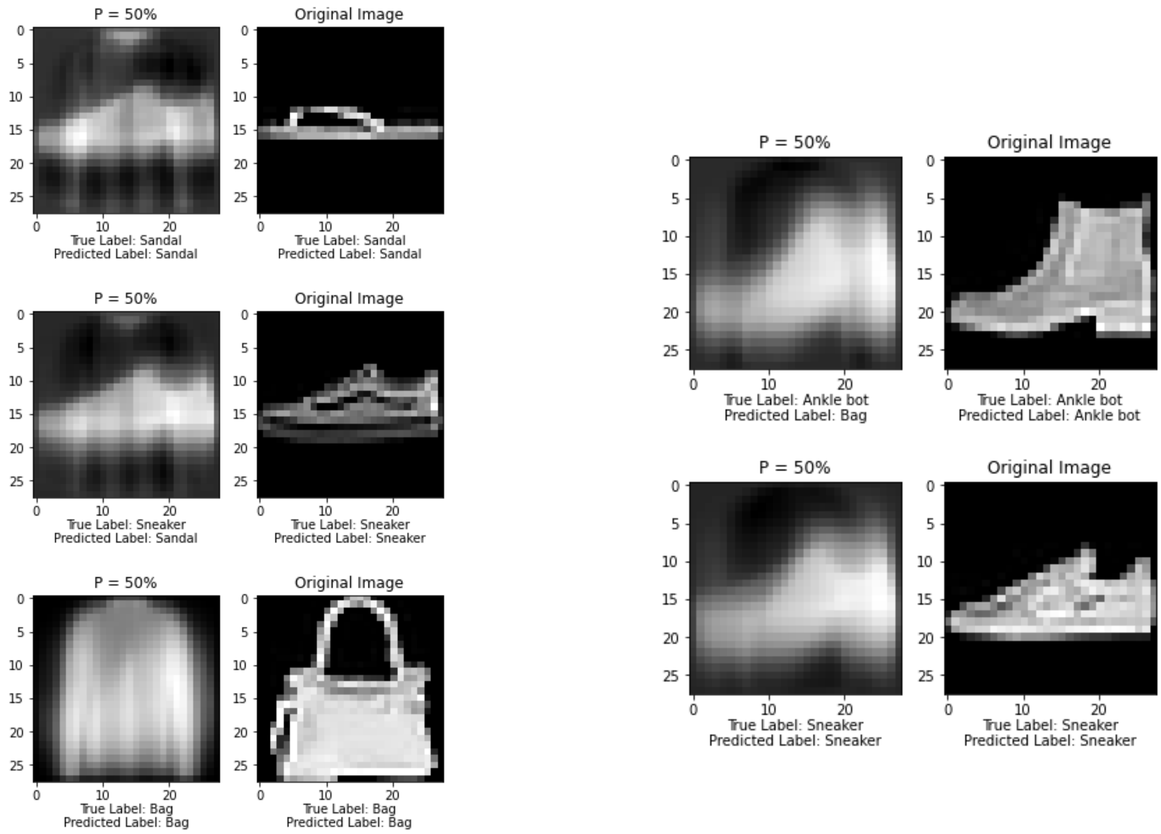

Figure 37 – Sample Results of Comparing Original Image Results and P = 50%

Decision Trees Performance (Number of Components = 3)

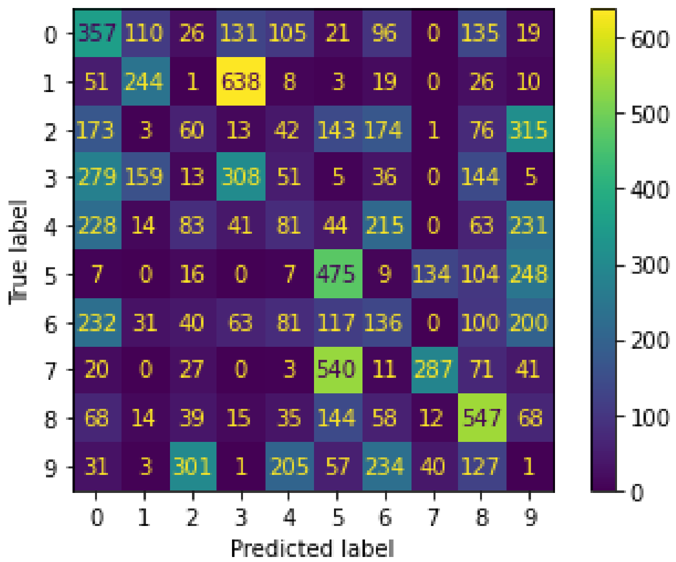

Figure 38 – Decision Trees Confusion Matrix in 3D dataset

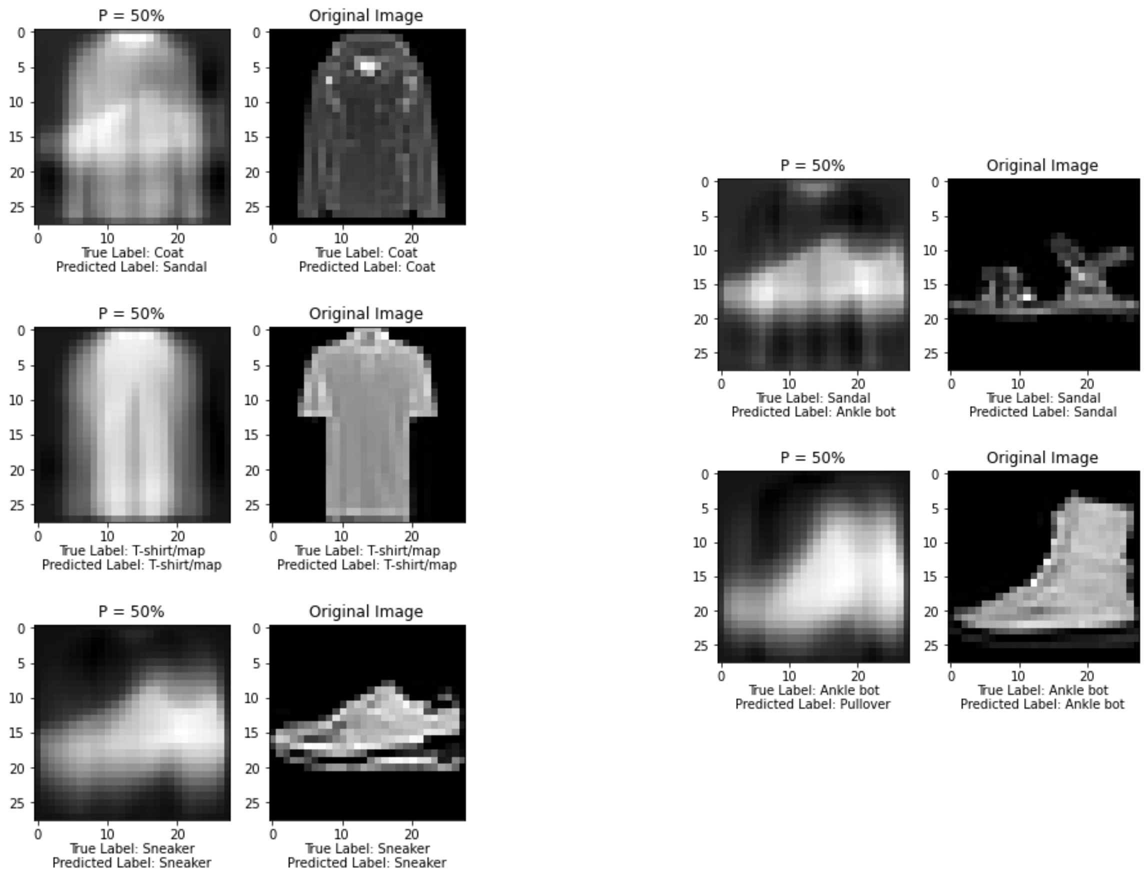

Figure 39 – Sample Results of Comparing Original Image Results and P = 50%

Random Forests Performance (Number of Components = 3)

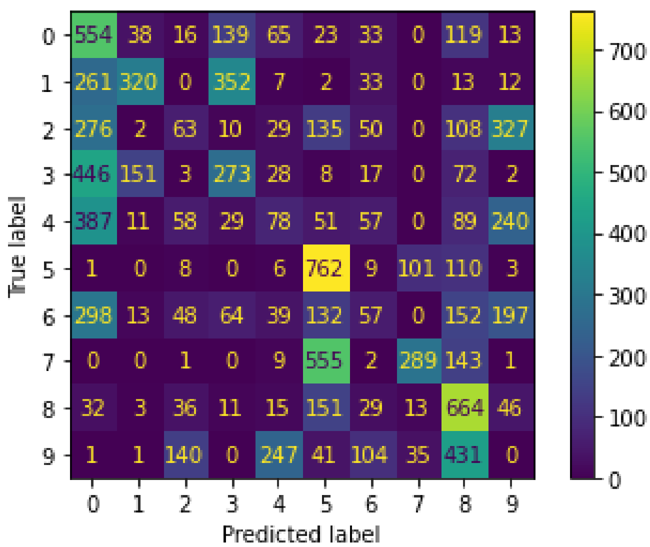

Figure 40 – Random Forests Confusion Matrix in 3D dataset

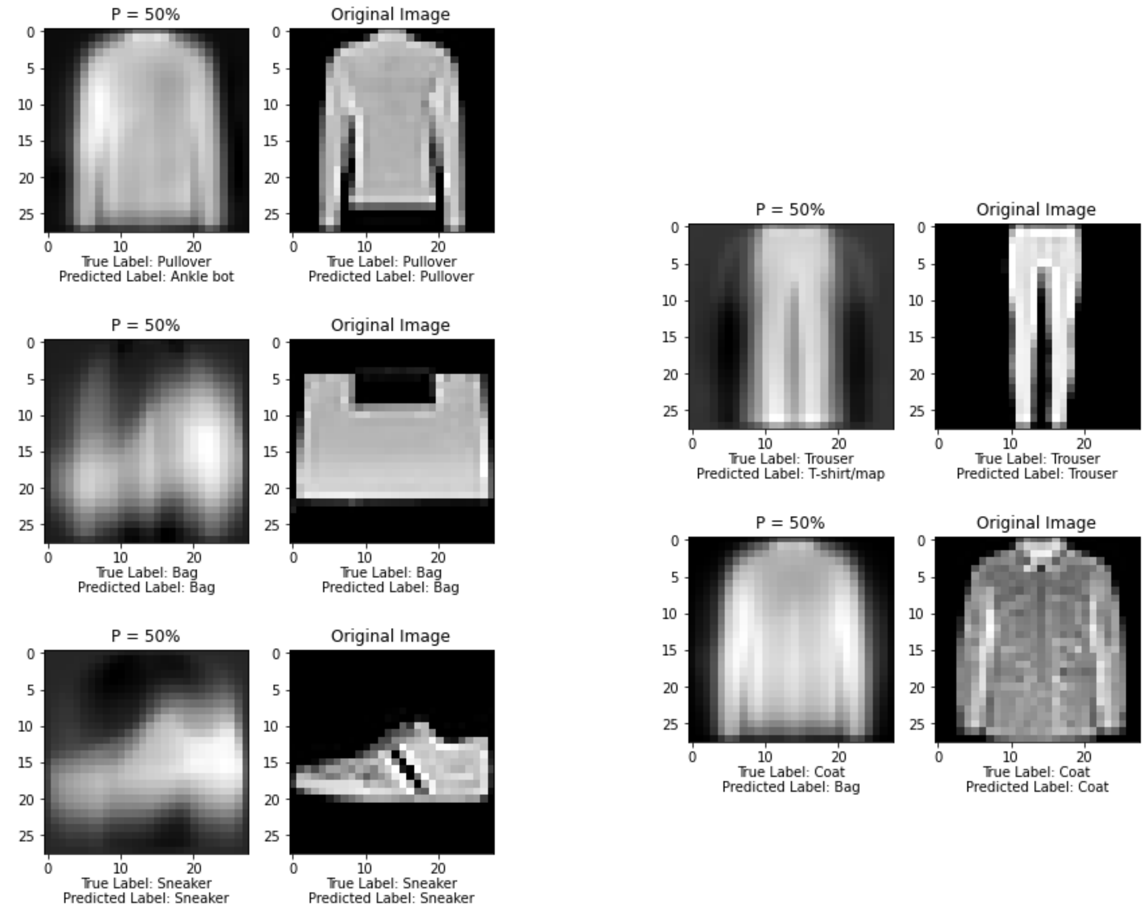

Figure 41 – Sample Results of Comparing Original Image Results and P = 50%

Number of Components = 14

To reach the P value approximately 75% dataset dimension should be reduced to 14 dimensions. After the reduction, 5 different classification model should be implemented the same as the original image.

KNN Performance (Number of Components = 14)

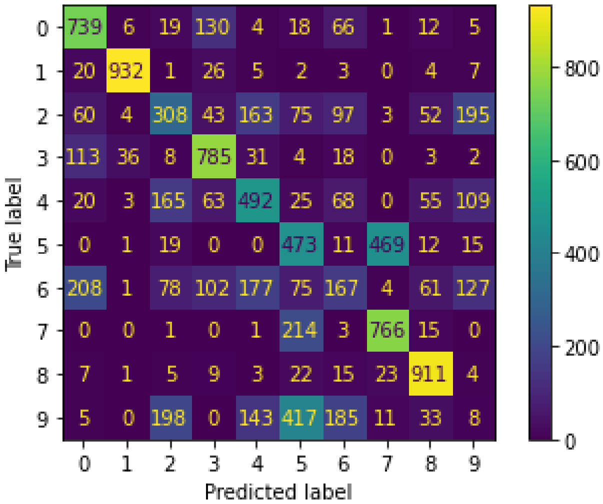

Figure 42 – KNN Confusion Matrix in 14D dataset

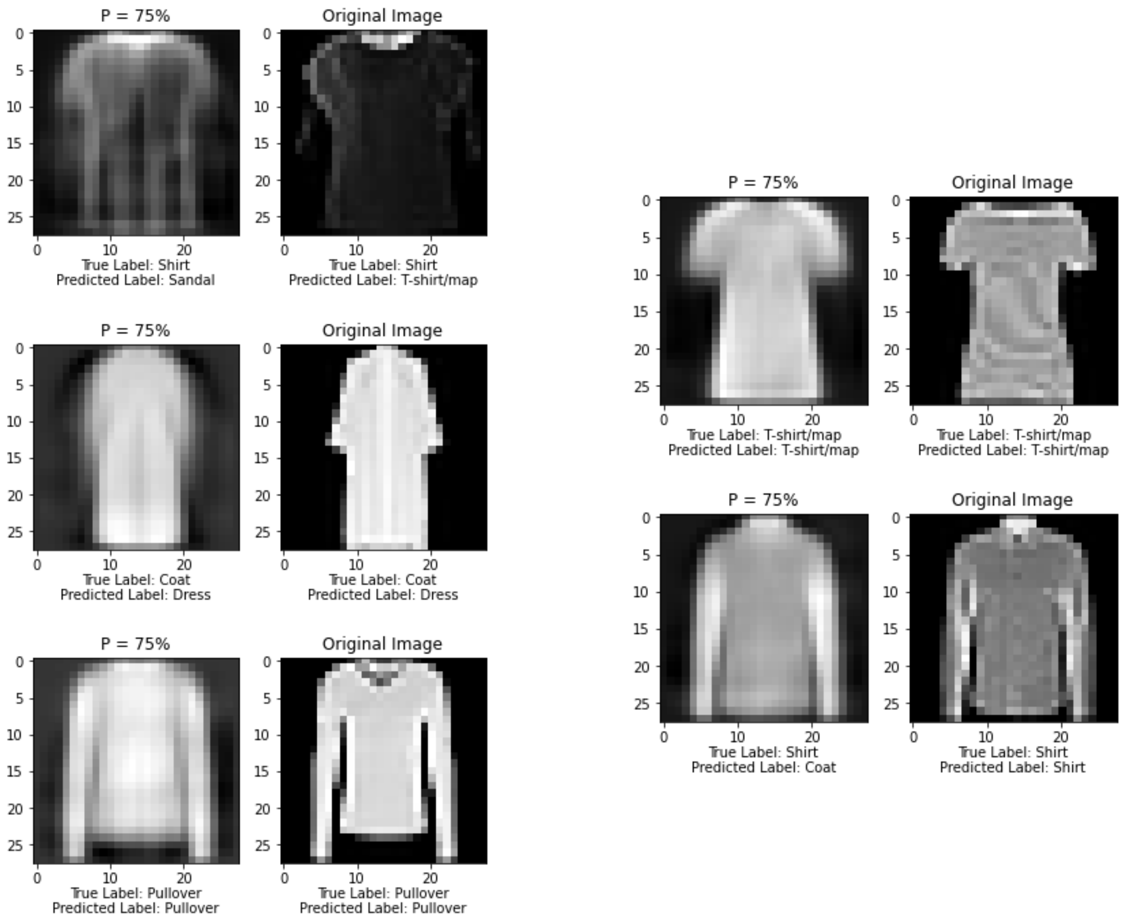

Figure 43 – Sample Results of Comparing Original Image Results and P = 75%

Perceptron Performance (Number of Components = 14)

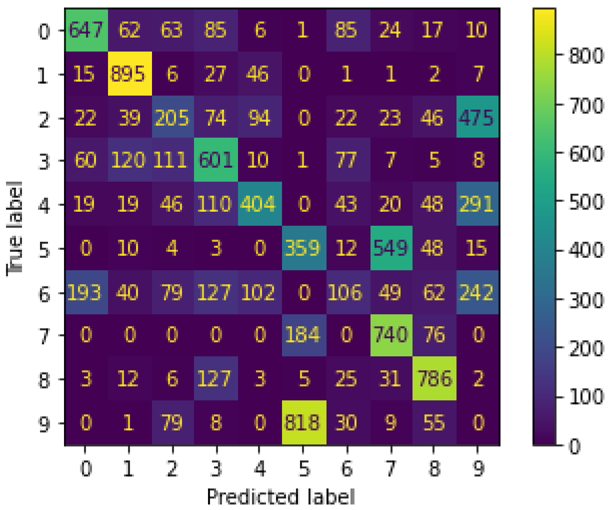

Figure 44 – Perceptron Confusion Matrix in 14D Dataset

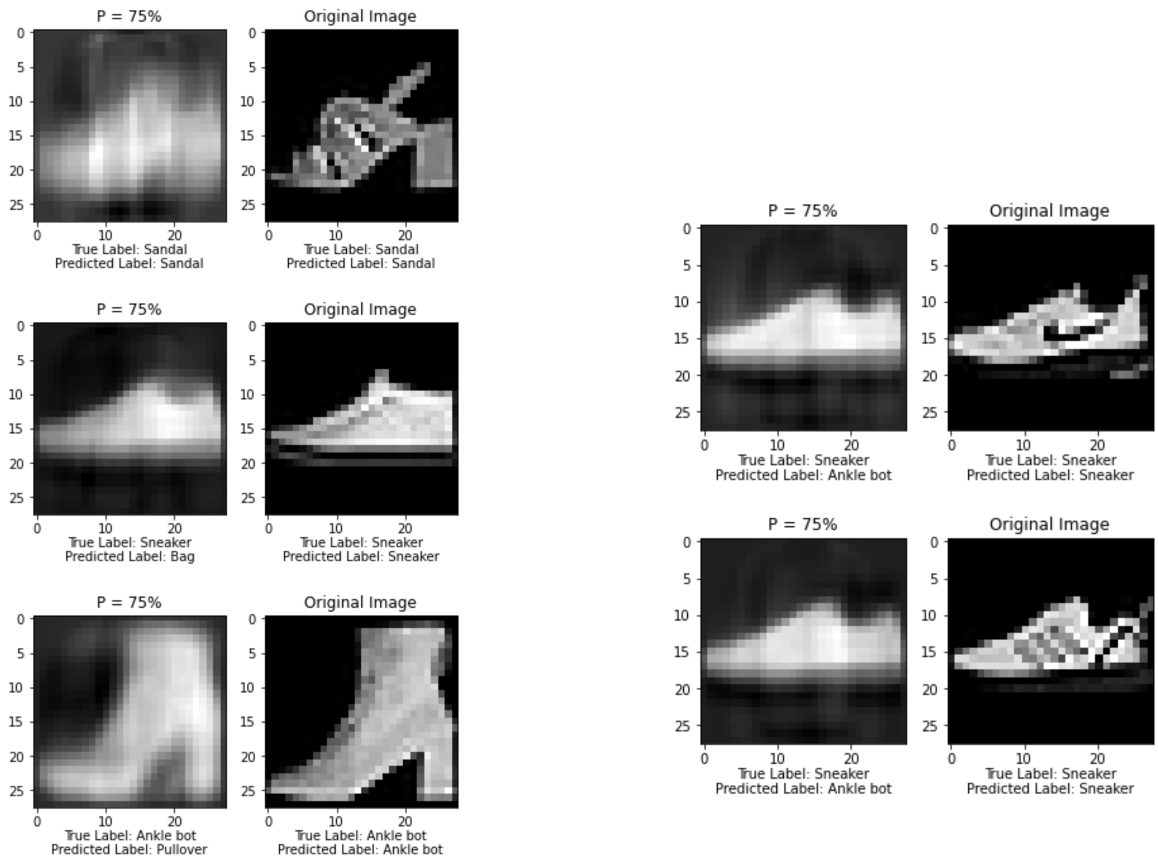

Figure 45 – Sample Results of Comparing Original Image Results and P = 75%

SVM Performance (Number of Components = 14)

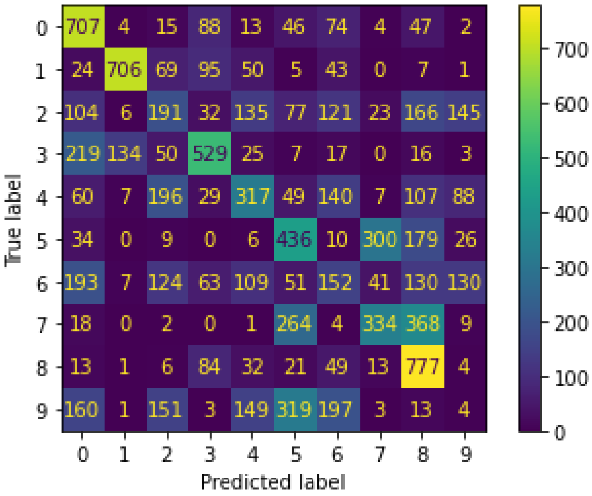

Figure 46 – SVM Confusion Matrix in 14D dataset

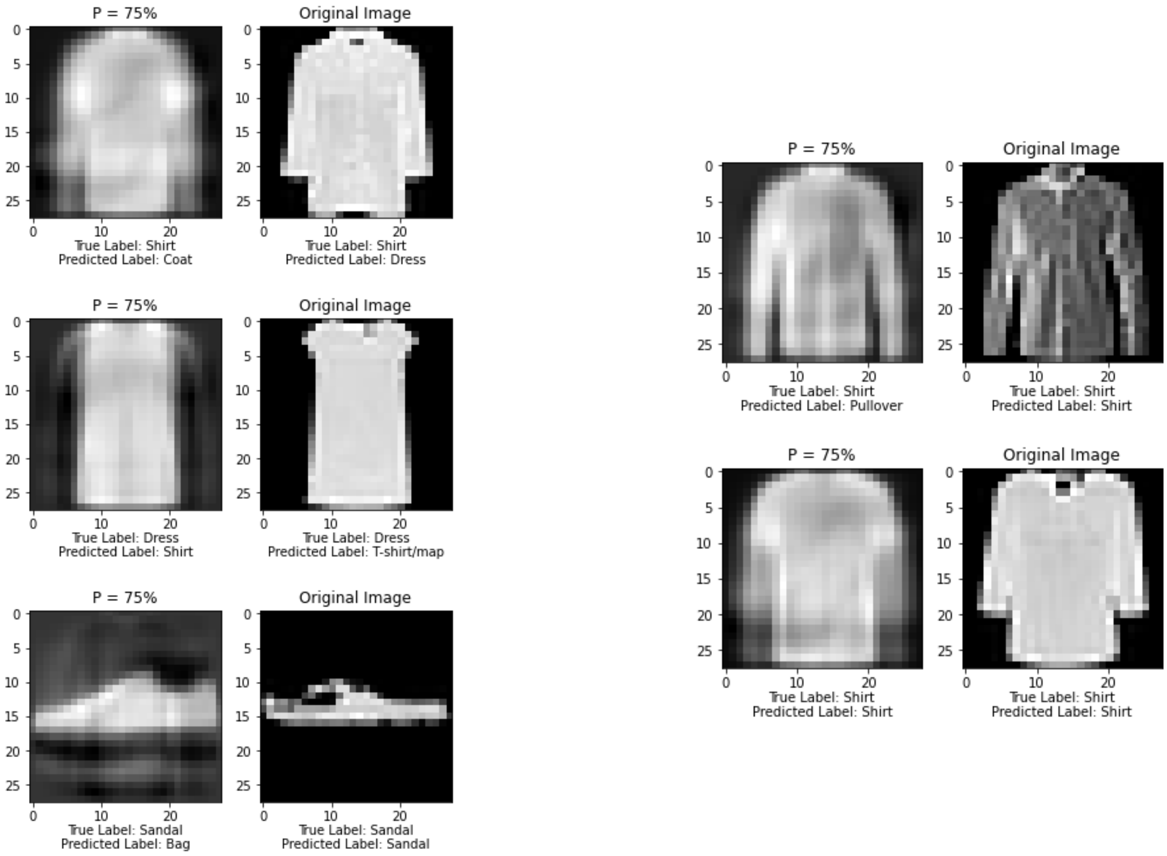

Figure 47 – Sample Results of Comparing Original Image Results and P = 75%

Decision Trees Performance (Number of Components = 14)

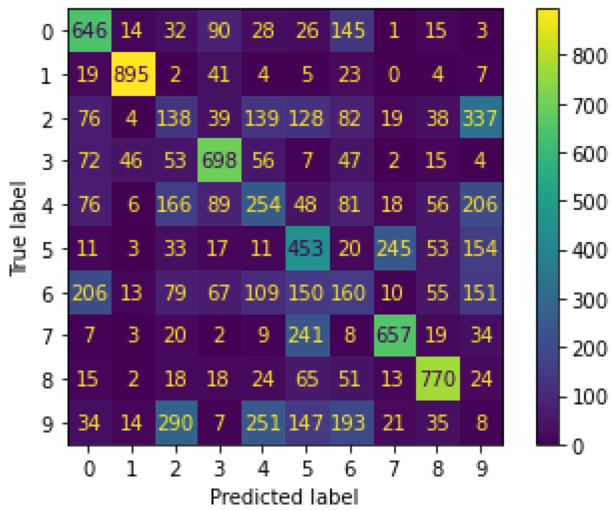

Figure 48 – Decision Trees Confusion Matrix in 14D dataset

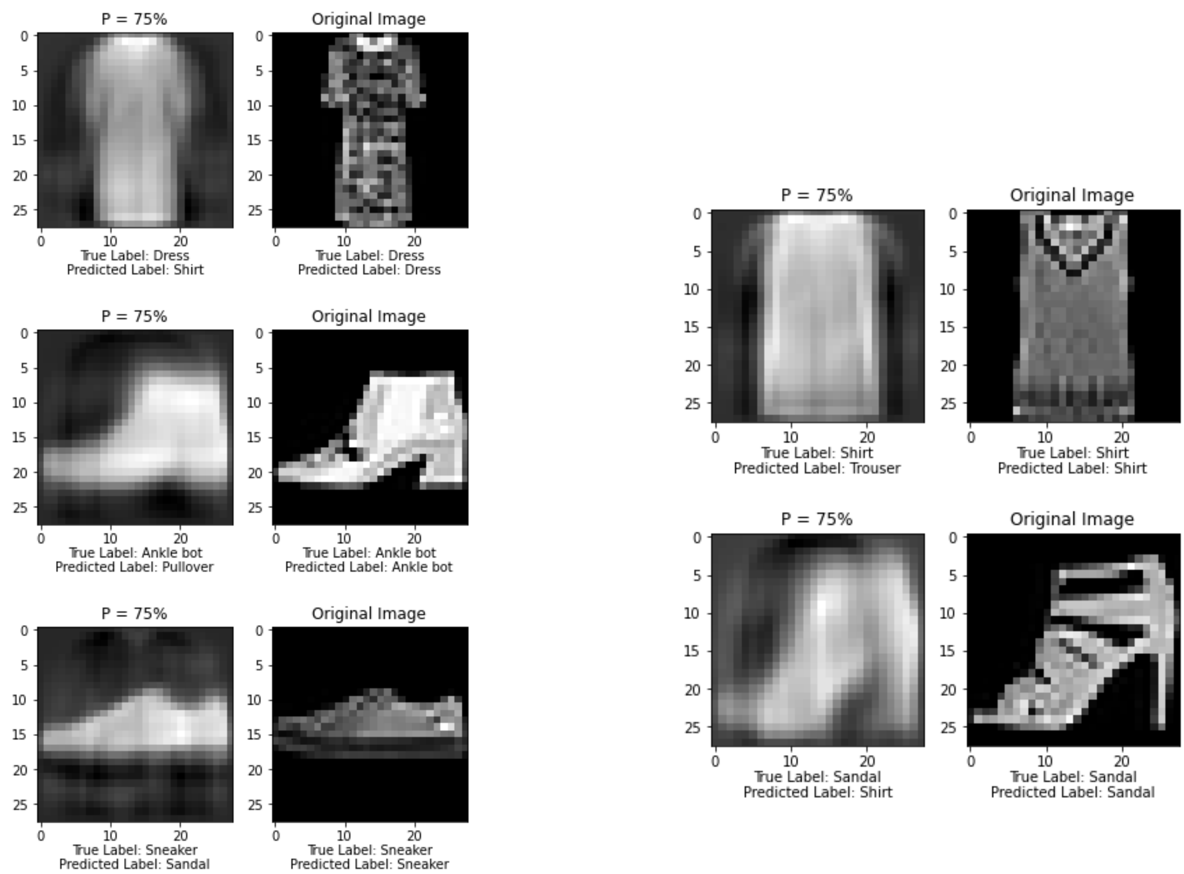

Figure 49 – Sample Results of Comparing Original Image Results and P = 75%

Random Forests Performance (Number of Components = 14)

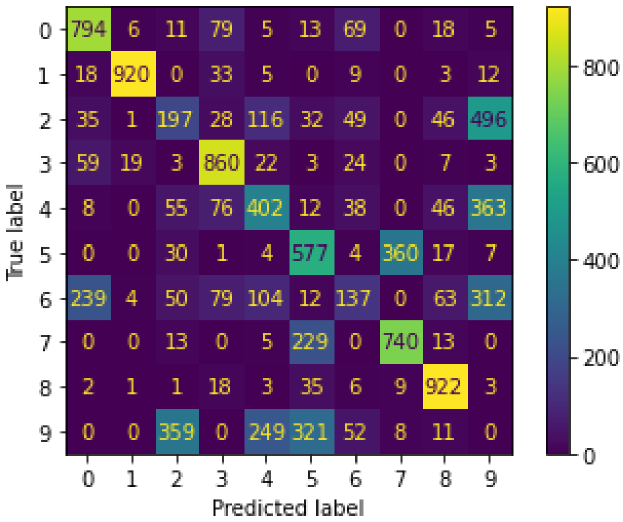

Figure 50 – Random Forest Confusion Matrix in 14D dataset

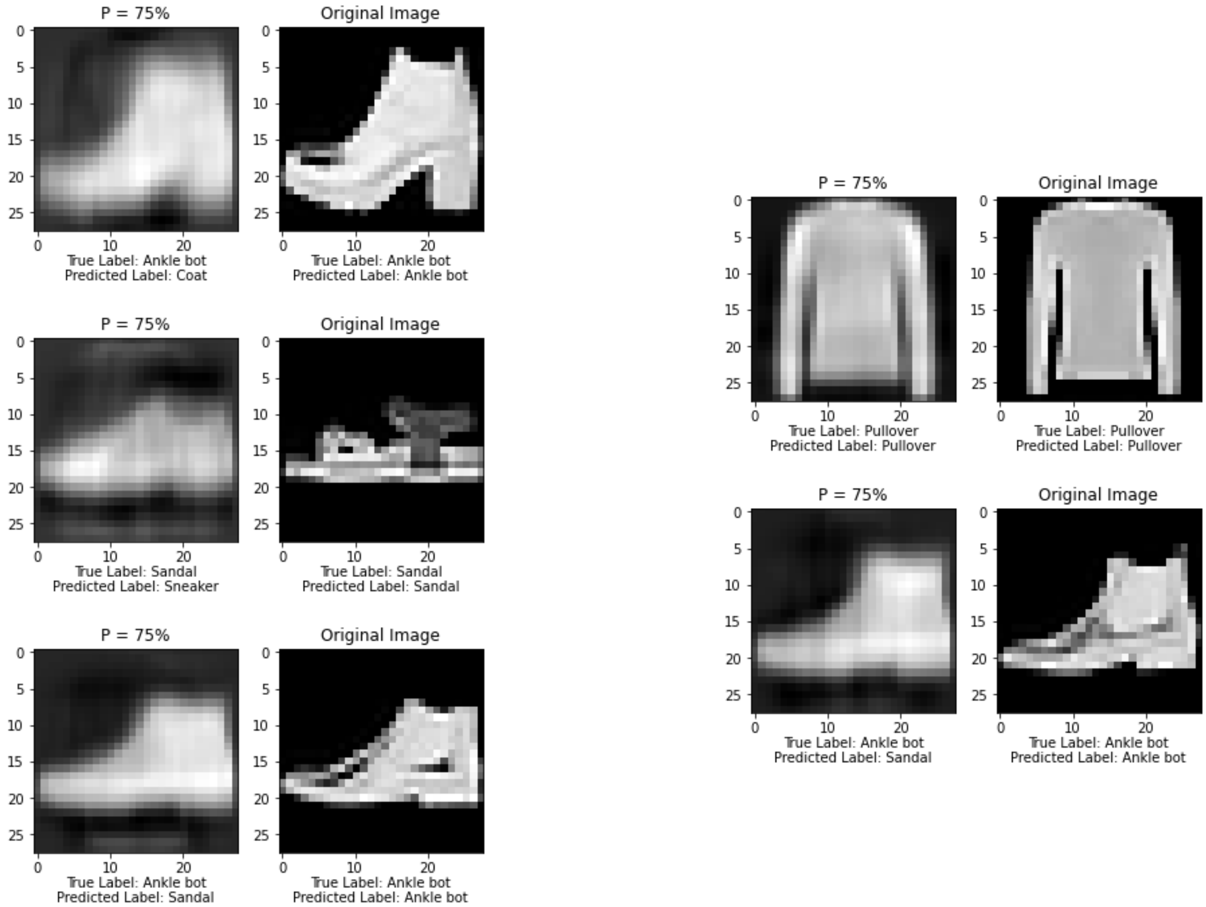

Figure 51 – Sample Results of Comparing Original Image Results and P = 75%

Number of Components = 189

To reach the P value approximately 95% dataset dimension should be reduced to 189 dimensions. After the reduction, 5 different classification model should be implemented the same as the original image.

KNN Performance (Number of Components = 189)

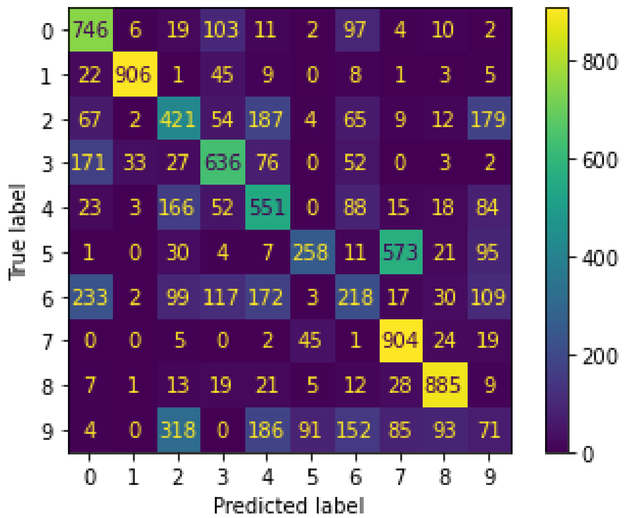

Figure 52 – KNN Confusion Matrix in 189D in dataset

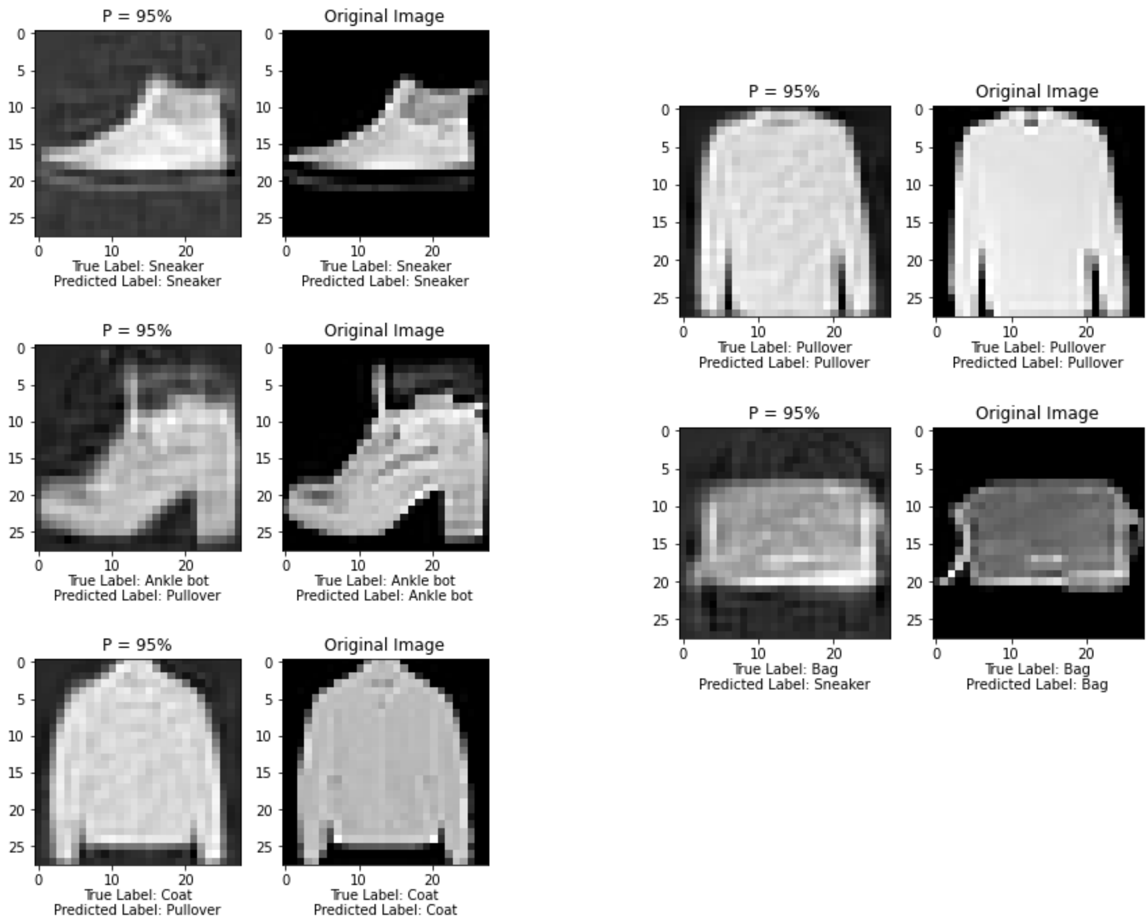

Figure 53 – Sample Results of Comparing Original Image Results and P = 95%

Perceptron Performance (Number of Components = 189)

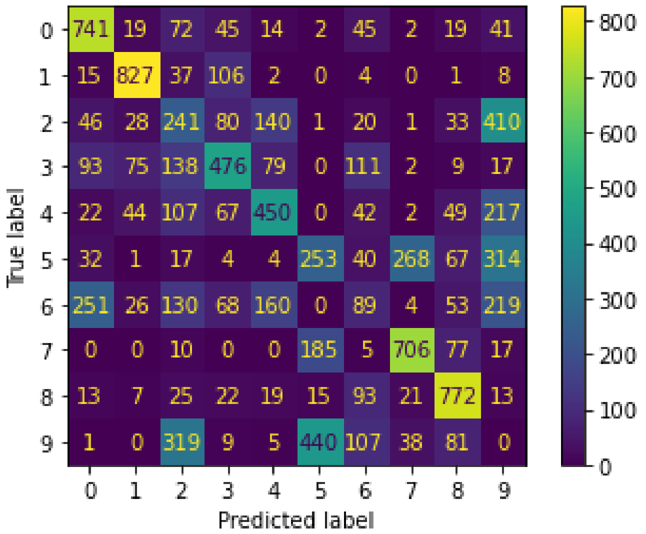

Figure 54 – Perceptron Confusion Matrix in 189D in dataset

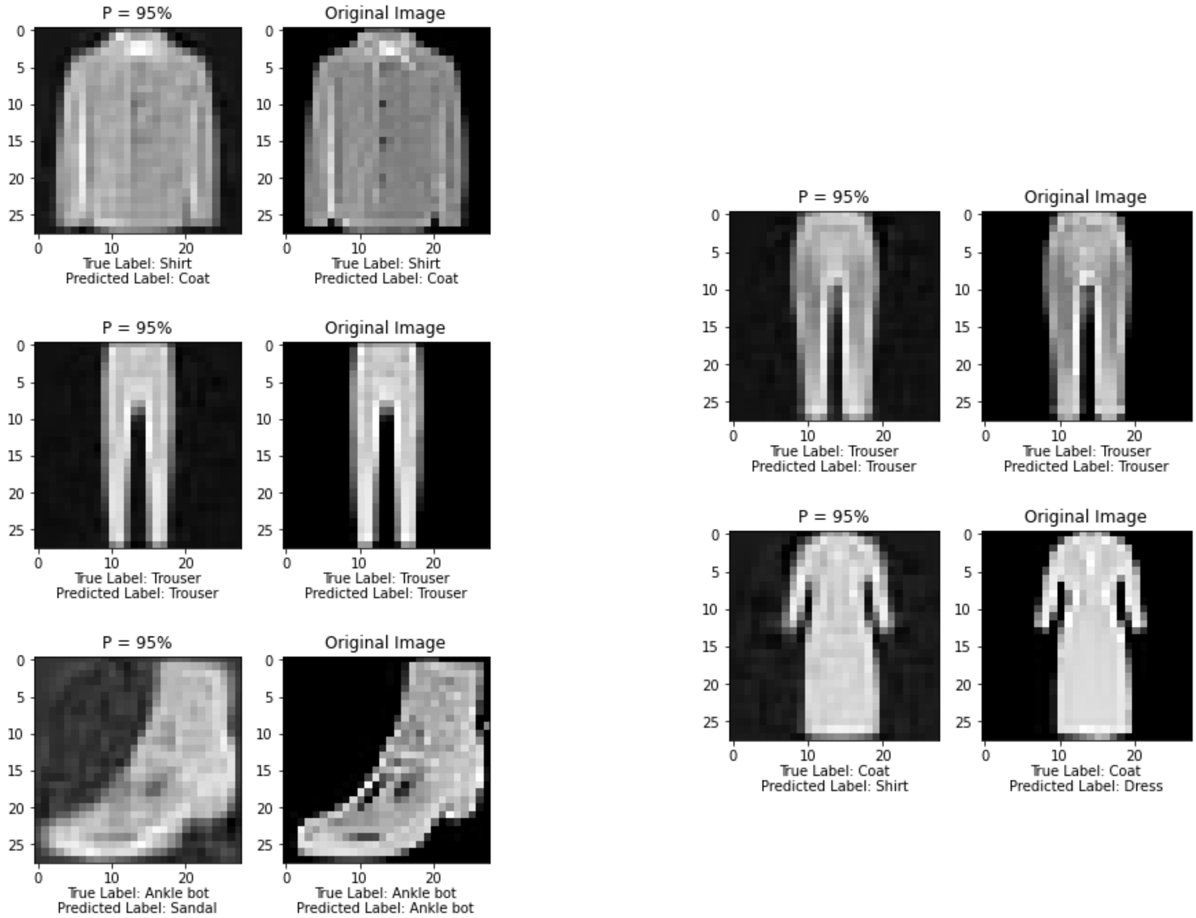

Figure 55 – Sample Results of Comparing Original Image Results and P = 95%

SVM Performance (Number of Components = 189)

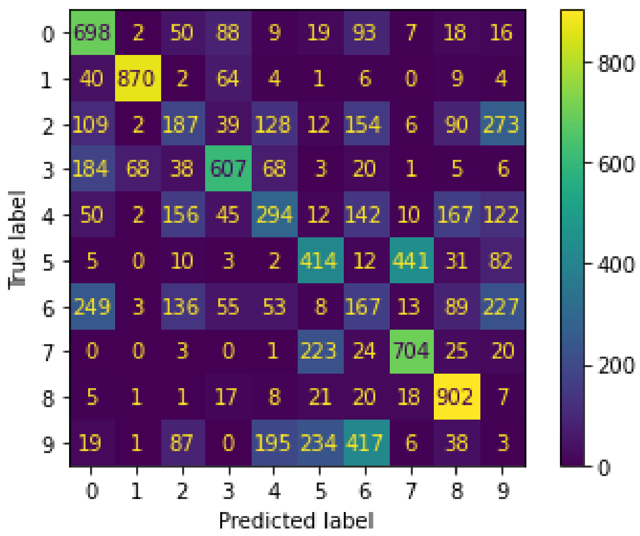

Figure 56 – SVM Confusion Matrix in 189D in dataset

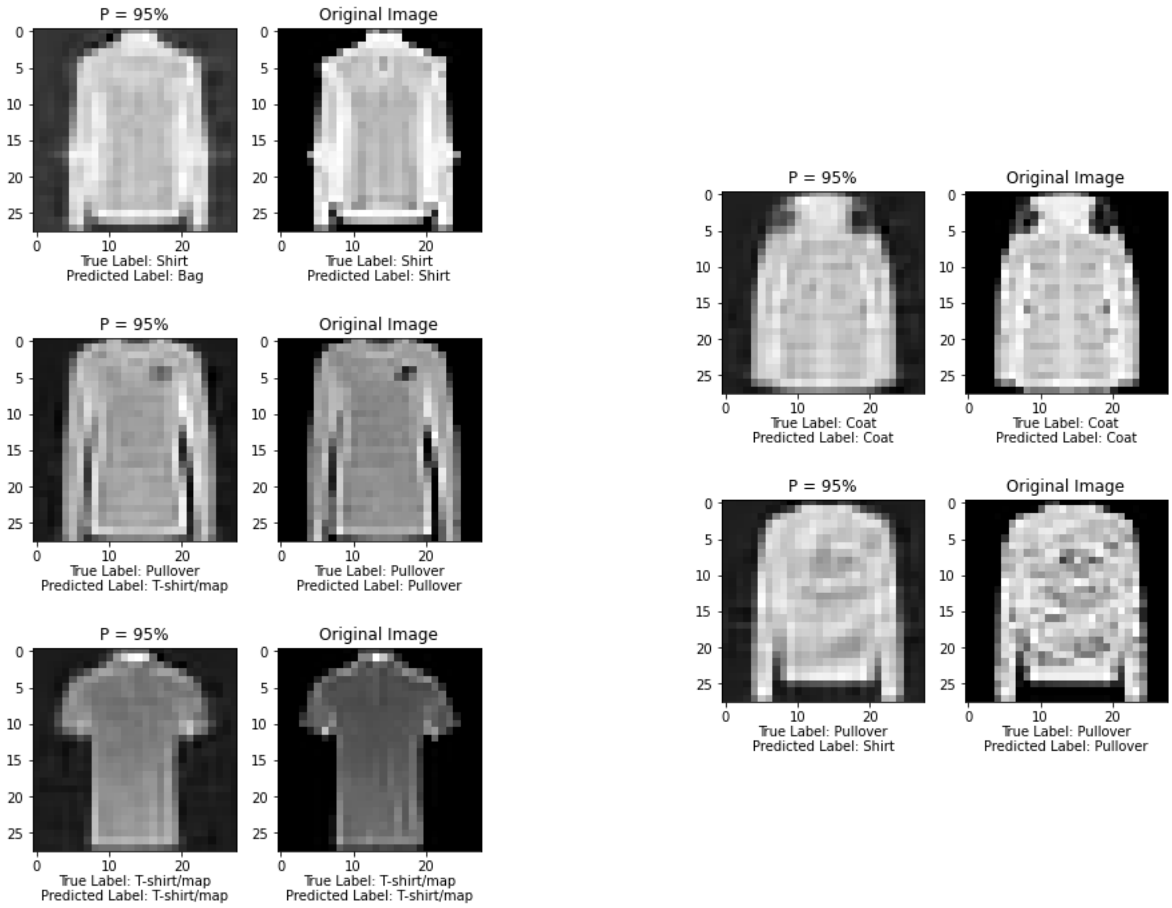

Figure 57 – Sample Results of Comparing Original Image Results and P = 95%

Decision Trees Performance (Number of Components = 189)

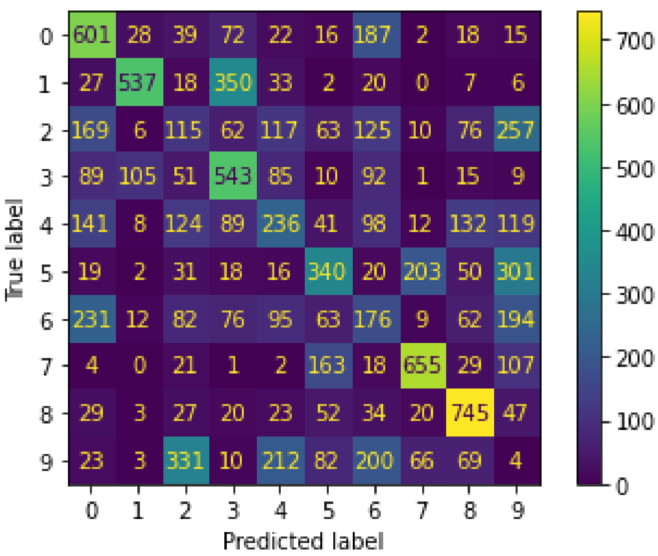

Figure 58 – Decision Trees Confusion Matrix in 189D dataset

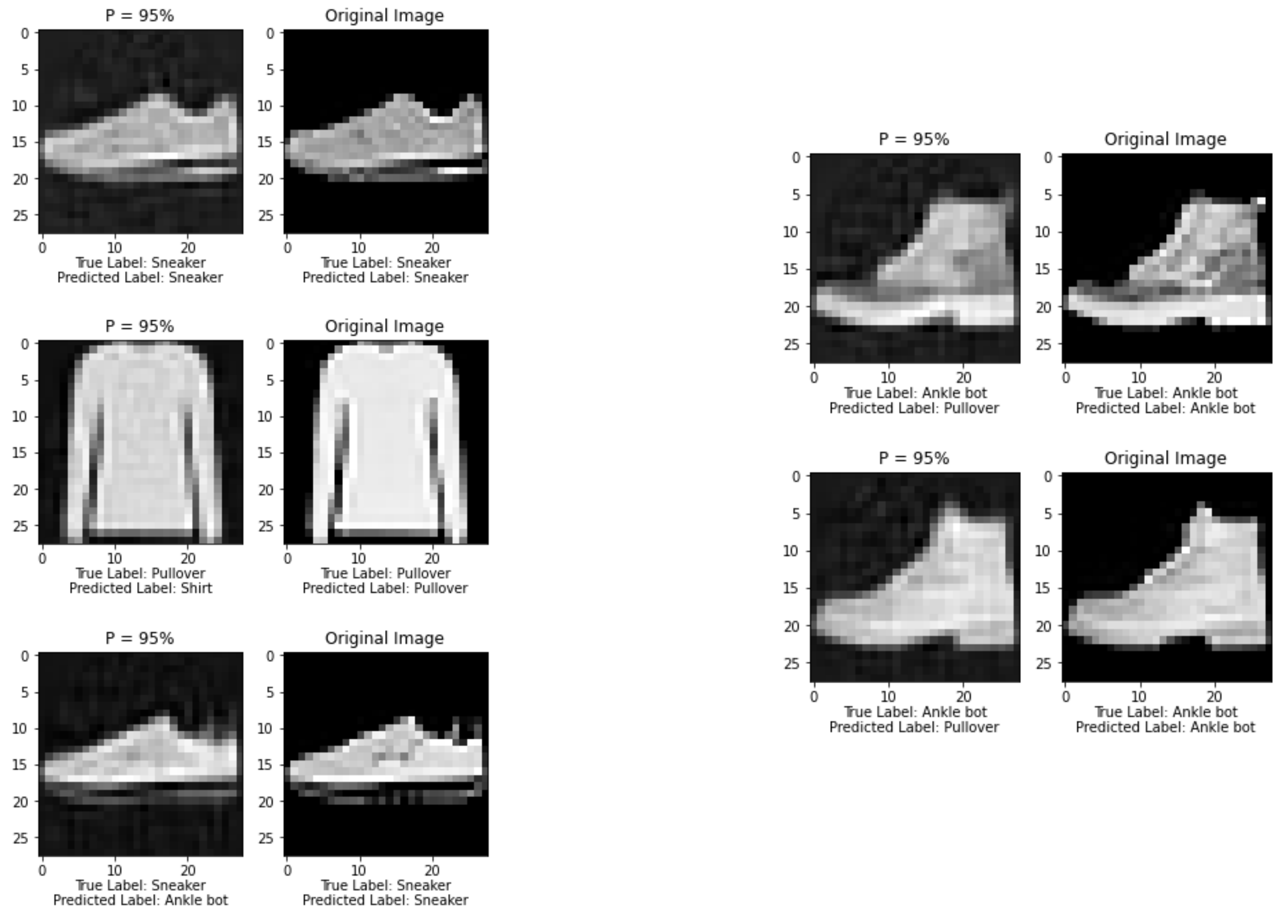

Figure 59 – Sample Results of Comparing Original Image Results and P = 95%

Random Forests Performance (Number of Components = 189)

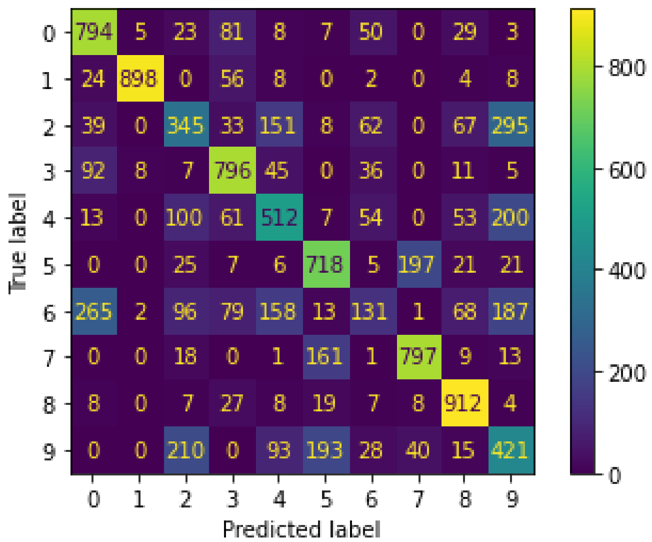

Figure 60 – Random Forests Confusion Matrix in 189D dataset

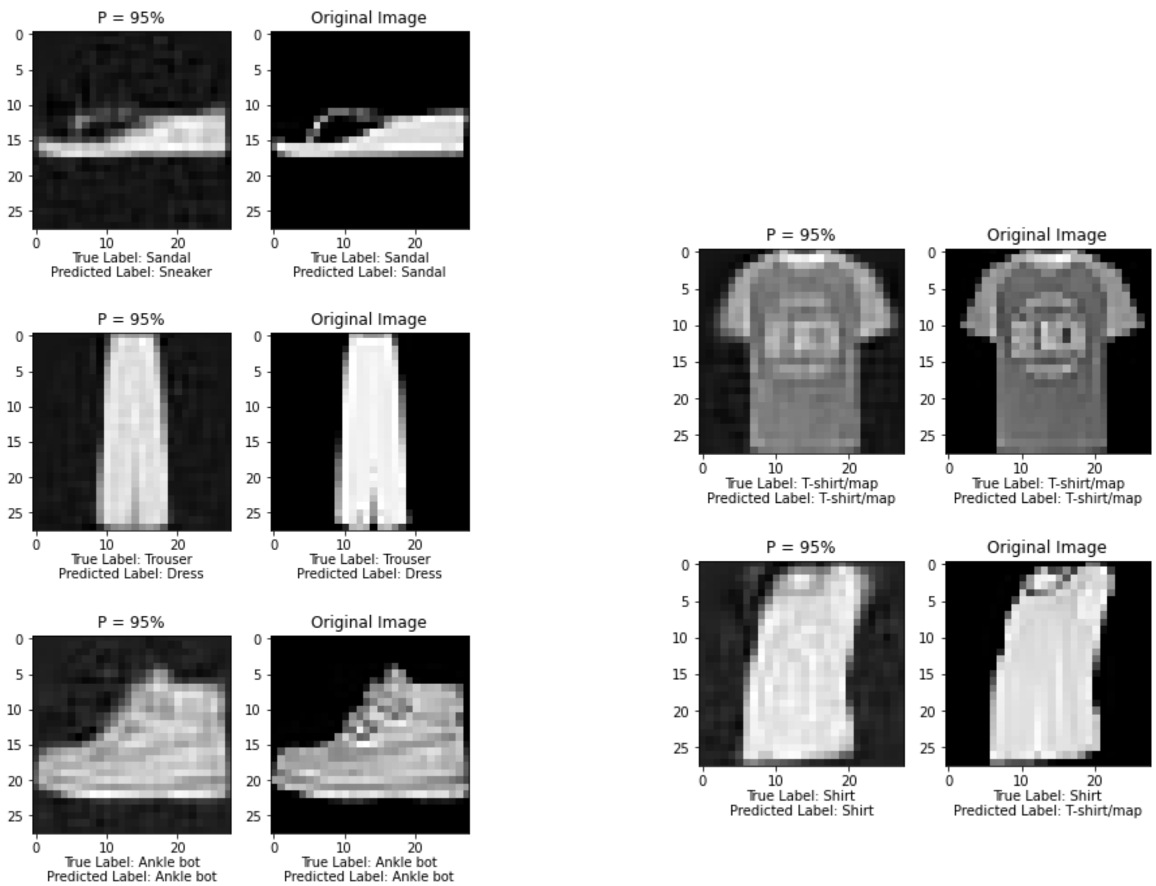

Figure 61 – Sample Results of Comparing Original Image Results and P = 95 %

Results

As seen in the "Figure-62", it can be stated that dimension reduction has a negative effect on accuracy score except for the decision trees model between P≈95% and P≈75%. Also, like the original image, SVM model has a high accuracy score in small variances (25,50). On the contrary, in higher variances (75,95), SVM model is not as successful as in the small variances. Random Forests model is the most successful accuracy scores in higher variances. Therefore, should be used different classification models in different variances according to the accuracy score results table.

	P ≈ 25 %	P ≈ 50 %	P ≈ 75 %	P ≈ 95 %	Original Image
K-Neighbors	0.2510	0.2925	0.5581	0.5596	0.8554
Perceptron	0.1363	0.2298	0.4783	0.4743	0.8142
SVM	0.3142	0.3157	0.4153	0.4846	0.8963
Decision Trees	0.2255	0.2496	0.4679	0.3952	0.8015
Random Forests	0.2256	0.3060	0.5549	0.6324	0.8760
Explained Variance	29.04%	52.81%	75.32%	95.0%	100%

Table 62 – Different Accuracy Score results in 5 different classification models & different variances

Conclusion

Consequently,this experiment shows that the performance of the 5 different classification model in both original dimension and reduced dimensions. Besides, as is mentioned earlier there is no method that gets high accuracy score in every variance. Thus, all models should be trained and evaluated as this experiment does.

References

- https://machinelearningmastery.com/perceptron-algorithm-for-classification-in-python/
- Course Resource - Classification Slide
- Course Resource - Classificaiton Slide Part2
- https://www.datacamp.com/community/tutorials/decision-tree-classification-python
- https://www.datacamp.com/community/tutorials/random-forests-classifier-python
- https://towardsdatascience.com/all-you-need-to-know-about-pca-technique-in-machine-learning-443b0c2be9a1