**Project Proposal**

Image processing for the apparel and fashion industry

**Problem:**

Be able to identify images of clothing items from targeted social media sites.

An apparel/fashion company wanting to start a new marketing campaign targeting users of social media such as Pinterest and Instagram.

People who post a lot of apparel pictures to **Pinterest or Instagram** may be planning a purchase or an event like a vacation. By identifying these images, the apparel/fashion company can develop:

* Models that can identify what makes a *blue blouse,* can use the information to recommend products that are visually similar.
* Understanding a person’s style in clothing can also help target them for related products.
* A following on these social media sites.

Also identify people who are following specific sites or people on Pinterest and Instagram that focus on apparel, can be potential customers

**Outline:**

* Data
* Process
  + Define approach
  + Get and explore the data
  + Prepare the data
    - Reduce the dimensionality of a data set to optimize performance of Machine-learning algorithms
  + Evaluate Models
  + Fine-tuning
* Path forward

Detail Code: [MNIST\_Milestone\_Report.ipynb](https://github.com/rivasjmr/Springboard/blob/master/MNIST_Milestone_Report.ipynb)

**Data**

The data used for this project is from [Kaggle](https://www.kaggle.com/zalando-research/fashionmnist).

Fashion-MNIST is a dataset of Zalando's article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes:

0-Tshirt/top

1-Trousers

2-Pullover

3-Dress

4-Coat

5-Sandal

6-Shirt

7-Sneaker

8-Bag

9-Ankle boot

**Process**

**Approach**

We are trying to process images and identify what type of clothing it is. Since data provided has features and labels, solutions, we will use machine learning and take a *supervised learning* approach using a *classification* task.

**Get and explore the data**

The data is provided in CVS format as two sets, training and test. We first read the files into data frames and split into features and labels and convert them in numpy arrays.

**Prepare data**

We have done some initial and simple preparation of the data already as stated above. Below we will go into a more complex task.

**Reduce the dimensionality of a data set**

Principal component analysis (PCA) is an unsupervised method for reducing the dimensionality of large datasets, increasing interpretability but at the same time minimizing information loss. It does by creating new uncorrelated variables that successively maximize variance. PCA is mainly used for two purposes:

* Data Visualization
* Improve performance of Machine-learning algorithms

The main focus used in this project was for improving performance in benchmarking a number of Supervised learning models for classification

There are few stepsto reduce the dimensionality of a data set.

1. Standardize the Data

PCA is affected by scale, so you need to scale the features in your data before applying PCA. Use StandardScaler to help standardize the dataset’s features onto unit scale (mean = 0 and variance = 1) which is a requirement for the optimal performance of many machine learning algorithms.

1. Explained Variance

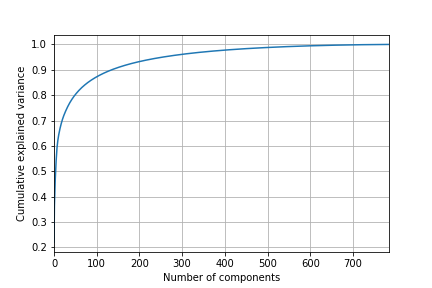
The explained variance tells how much information

(variance) can be attributed to each of the principal components.

1. Choose the number of components to use

Based on the acceptable accuracy

**PCA data based on Training Data**



The above plot shows:

~80% variance for 50 components

~88% variance for 100 components

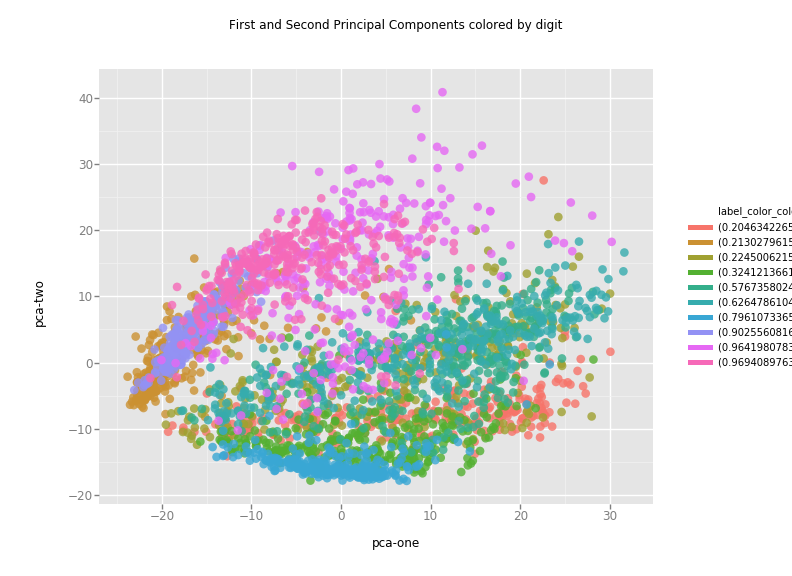
~94% variance for 200 components

~96% variance for 300 components

We are able to observe that around 50 components the curve starts to level off and the variance is about ~80%. We will use 50 components for the PCA to reduce the dimensionality.

We were able to take advantage of the data visualization opportunities with PCA. Looking at the first two components, representing about 40% Cumulative explained variance

From the graph you can see the clustering, but not enough to set them apart.



**Evaluate Models**

Used Supervised learning models for classification, since all the data is labeled. Used

the PCA-50 and PCA-600 datasets to analyze the different models, for performance and evaluation the metrics.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Classification Models** | |  |  |  |  |  |
|  | **PCA-50** |  |  | **PCA-600** |  |  |  |
|  | **fitting the model time** | **Predict time** | **Accuracy** | **fitting the model time** | **Predict time** | **Accuracy** | **Accuracy Diff.** |
| **LogisticRegression** | 00:21.75 | 00:00.01 | 0.61 | 02:08.91 | 00:00.12 | 0.63 | 0.02 |
| **DecisionTree** | 00:17.40 | 00:00.01 | 0.59 | 04:29.79 | 00:00.07 | 0.61 | 0.02 |
| **Kneighbors** | 00:00.86 | 01:26.60 | 0.73 | 00:07.58 | 16:56.05 | 0.68 | -0.05 |
| **RandomForest** | 00:07.00 | 0:00.42 | 0.69 | 00:23.76 | 0:00.16 | 0.58 | -0.11 |
| **SVM** | 02:56.27 | 00:28.33 | 0.59 | 15:10.32 | 04:53.87 | 0.7 | 0.11 |
| **Neural Networks (MLP)** | 00:47.39 | 00:00.04 | 0.63 | 01:41.08 | 00:00.16 | 0.6 | -0.03 |

Based on performance and accuracy will use the Random Forest model with PCA-50

**Fine Tuning model**

We look at the available hyperparameters in random forest and examine the default values.

Used RandomizedSearchCV, we first need to create a parameter grid to sample from during fitting

Used the random grid to search for best hyperparameters. First create the base model to tune. Then Random search of parameters, using 2 fold cross validation, search across 10 different combinations, and use all available cores.

Fitted the random search model to identify best parameters

To determine if random search yielded a better model, we compare the base model with the best random search model

Base Model Accuracy: 0.6858

New Model Accuracy : 0.7734

For Detail Code: [Fine\_tuning\_RF.ipynb](https://github.com/rivasjmr/Springboard/blob/master/Fine_tuning_RF.ipynb)

**Path Forward**

We have gone through the process of identifying the best data preparation and model to process images and classify them.

After launching into production, the systems performance will require sampling the system’s prediction and evaluating them. Therefore, prior to launching into production we need to:

* Have processes and code to check systems performance at regular interval
* Have processes and code to check systems input data quality
* Have a process defined how to train the model on a regular basis
  + Offline Learning (Batch learning)

train a new version from scratch on a full dataset

* + Online Learning

train incrementally by feeding it data instances sequentially