**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
  + Reduce the dimensionality of a data set
  + Models
  + Model optimization
* Summary
* Recommendations

**Data**

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

The data consist of 10 different classifications:

0-Tshirt/top 1-Trousers 2-Pullover 3-Dress 4-Coat

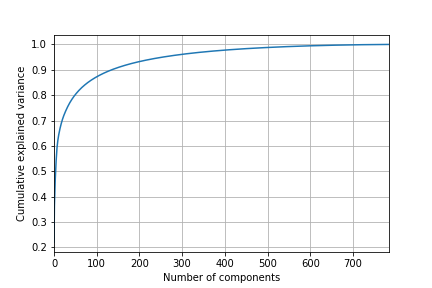
5-Sandal 6-Shirt 7-Sneaker 8-Bag 9-Ankle boot

**Process**

**Reduce the dimensionality of a data set**

To be able to process large datasets used principal component analysis (PCA) to reduce the dimensionality of a data set. This is an Unsupervised clustering algorithm.

**PCA data based on Training Data**



From the initial analysis will use the following for further analysis.

50 components with ~.80 Cumulative explained variance

600 components with ~1.0 Cumulative explained variance

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PCA** | **50** | **100** | **300** | **600** |
| **Fit transform time** | 00:11.06 | 00:13.31 | 00:34.52 | 0:01:18.13 |

**Code:** [PCA\_multi\_components.ipynb](https://github.com/rivasjmr/Springboard/blob/master/PCA_multi_components.ipynb)

**Models**

Used Supervised learning models for classification, since all the data is labeled. As outlined above using 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

**Code:** [Classification models with orginal data and PCA.ipynb](https://github.com/rivasjmr/Springboard/blob/master/Classification%20%20models%20with%20%20orginal%20data%20and%20PCA.ipynb)

[3 more Classification model with PCA .ipynb](https://github.com/rivasjmr/Springboard/blob/master/3%20more%20Classification%20model%20with%20PCA%20.ipynb)

**Model Optimization**

Used Supervised learning models for classification, all data is labeled. As outlined above using the PCA-50 and PCA-600 datasets to analyze the different models

Hyperparameter tuning through experimental results than theory tried different combinations evaluate the performance. However, must be careful not to [overfitting](https://elitedatascience.com/overfitting-in-machine-learning) , one of the most fundamental problems in machine learning

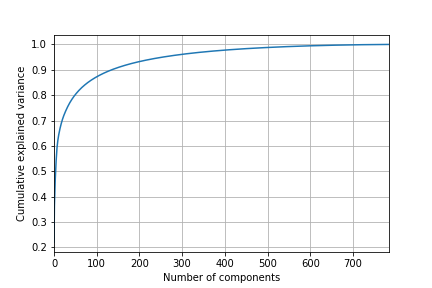
**Code:**

**Summary**

**Recommendations**

(60000, 600)

(60000, 50)



**clasification report:**

**precision recall f1-score support**

**0 0.68 0.74 0.71 1000**

**1 0.91 0.89 0.90 1000**

**2 0.47 0.55 0.51 1000**

**3 0.71 0.75 0.73 1000**

**4 0.48 0.44 0.46 1000**

**5 0.72 0.73 0.73 1000**

**6 0.40 0.32 0.36 1000**

**7 0.73 0.76 0.74 1000**

**8 0.90 0.89 0.90 1000**

**9 0.85 0.79 0.82 1000**

**avg / total 0.69 0.69 0.68 10000**

**confussion matrix:**

**[[742 13 35 61 10 1 118 0 20 0]**

**[ 17 893 10 62 4 0 14 0 0 0]**

**[ 24 2 553 9 265 4 125 1 17 0]**

**[ 86 64 19 754 25 0 48 0 4 0]**

**[ 29 1 288 97 437 3 131 1 13 0]**

**[ 1 0 13 4 4 732 7 177 10 52]**

**[190 6 240 67 143 7 320 0 27 0]**

**[ 0 0 0 0 0 156 0 755 7 82]**

**[ 7 1 24 5 15 13 31 7 893 4]**

**[ 1 0 2 3 1 98 2 95 4 794]]**