Part 1:

Def entropy(X):

f = np.bincount(X)

f = f[f>0]

p = f/np.sum(f)

ent = -np.sum(p\*np.log2(p))

return ent

people = fetch\_lfw\_people(min\_faces\_per\_person=20, resize=0.7)

mask = np.zeros(people.target.shape, dtype=np.bool)

for target in np.unique(people.target):

mask[np.where(people.target == target)[0][:50]] = 1

X\_people = people.data[mask]

y\_people = people.target[mask]

X\_people = X\_people / 255.

k = 100

km = KMeans(n\_clusters = k)

km.fit(X\_people)

km\_assignments = km.labels\_

km\_means = km.cluster\_centers\_

for i in range(k):

ind = km\_assignments == i

ent = entropy(y\_people[ind])

print("Cluster {:d}, size = {:d}, entropy = {:.3f}".format(i, np.sum(ind), ent))

if ent > 4 and np.sum(ind) > 10:

fig, axes = plt.subplots(2, 5, figsize=(15, 8), subplot\_kw={'xticks': (), 'yticks': ()})

for i, (component, ax) in enumerate(zip(y\_people[ind], axes.ravel())):

ax.imshow(component.reshape(image\_shape), cmap = 'grey')

ax.set\_title("{}. component".format(i+1))

dbs = DBS()

dbs.fit(X\_people)

dbs\_assignments = dbs.labels\_

dbs\_means = dbs.core\_sample\_indices\_

print(len(dbs\_means))

a = 100

ac = AC(n\_clusters = a)

ac.fit(X\_people)

ac\_assignments = ac.labels\_

ac\_means = ac.n\_clusters

print(ac\_means)

for i in range(a):

ind = ac\_assignments == i

ent = entropy(y\_people[ind])

print("Cluster {:d}, size = {:d}, entropy = {:.3f}".format(i, np.sum(ind), ent))

if ent > 4 and np.sum(ind) > 10:

fig, axes = plt.subplots(2, 5, figsize=(15, 8), subplot\_kw={'xticks': (), 'yticks': ()})

for i, (component, ax) in enumerate(zip(y\_people[ind], axes.ravel())):

ax.imshow(component.reshape(image\_shape), cmap = 'grey')

ax.set\_title("{}. component".format(i+1))

DBS only makes one cluster even with changes to the algorithm, and such doesn’t have a printing method.

Part 2:

Problem Description:

What combination of preprocessing and algorithm give the best regression prediction and classification score respectively for the energy, people, and mnist data set?

Descriptions:

K-Nearest Neighbors: Algorithm which makes predictions for the testing set based on the k nearest data points.

Decisions Trees: A hierarchy of if/else statements to go from observations about an item based on its attributes to conclusions about the item’s target value.

Random Forests: A extension of decision trees, which constructs multiple trees and returns the mean prediction of the individual trees.

Support Vector Regression: Creates bounders along the categories of the training examples to separate data into the corresponding category, with an emphasis on creating large gaps. Employs the kernel trick to add a nonlinear feature to the representation to make the prediction more powerful.

Multilayer Perceptron: An algorithm which employs nodes that represent the input features, each using a nonlinear activation function, which together represent the weighted sum of the inputs.

Experimental Results:

* “()” next to a preprocessing method indicates that certain default variables in the \_\_init\_\_() call were changed. The most common changes were as follows:
* Number: the number of clusters, components, etc. was altered
* Default: nothing was changed
* prev opt: all previous optimizations were included
* random: randomization was employed
* *Italicized*: preprocessing worsened, underlined: no change, **bold**: preprocessing improved
* Changes tested on all algorithms unless change had no effect, otherwise only test on algorithms affected

People:

-Preprocessing

Starting point

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| People | KNC | DTC | RFC | SVC | MLPC | Overall |
| None | 0.208 | 0.157 | 0.196 | 0.092 | 0.075 | 0.513 |
| Normalization | 0.23 | 0.15 | **0.242** | 0.116 | 0.438 |  |
| Standardization | 0.235 | **0.16** | 0.194 | **0.395** | 0.504 |  |
| PCA (default) | 0.208 | 0.087 | 0.029 | 0.165 | **0.513** |  |
| NMF (50)\* | **0.254** | 0.136 | 0.169 | 0.104 | 0.448 |  |
| KMC (100) | 0.094 | 0.075 | 0.111 | 0.133 | 0.111 |  |
| MAX | 0.254 | 0.16 | 0.242 | 0.395 | **0.513** |  |

Changes to PCA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| randomized | 0.208 | **0.092** | 0.027 | 0.165 | **0.516** |

Changes to NMF

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NMF (100) | *0.22* | | *0.126* | | **0.179** | | **0.107** | **0.511** |
|  |  | |  | |  | |  |  |
| NMF(prev opt, random) | *0.186* | | *0.116* | | **0.131** | | **0.046** | **0.523** |
|  |  | |  | |  | |  |  |
| NMF(prev opt, nndsvda) | *0.08* | | *0.153* | | **0.128** | | **0.065** | **0.128** |
|  |  | |  | |  | |  |  |
| NMF(prev opt, mu) | *0.252* | | *0.126* | | **0.167** | | **0.143** | **0.533** |
|  |  | |  | |  | |  |  |
|  | | SVC | | MLPC | |
| NMF(prev opt, 1 (kl)) | | **0.145** | | *0.528* | |

Changes to KMeans

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| KMC (200) | **0.102** | **0.111** | *0.107* | *0.126* | *0.056* |
|  |  |  |  |  |  |
| KMC(prev opt, random) | *0.099* | *0.097* | *0.104* | *0.126* | *0.119* |
|  |  |  |  |  |  |
| KMC(prev opt, n\_init 25) | *0.099* | *0.104* | *0.094* | *0.126* | **0.126** |

Fully optimized preprocessing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| People Opt. | KNC | DTC | RFC | SVC | MLPC | Overall |
| None | 0.208 | 0.157 | 0.196 | 0.092 | 0.075 | 0.533 |
| Normalization | 0.23 | 0.15 | **0.242** | 0.116 | 0.438 |  |
| Standardizaiton | 0.235 | **0.16** | 0.194 | **0.395** | 0.504 |  |
| PCA | 0.208 | 0.092 | 0.029 | 0.165 | 0.516 |  |
| NMF | **0.254** | 0.153 | 0.179 | 0.145 | **0.533** |  |
| KMC | 0.102 | 0.111 | 0.111 | 0.133 | 0.126 |  |
| MAX | 0.254 | 0.16 | 0.242 | 0.395 | **0.533** |  |
|  |  |  |  |  |  |  |

Changes to algorithms:

kNNC: n\_neighbors = 10, weights = distance, p = 1

RFC: n\_jobs = 100, max\_depth = 25, bootstrap = False

SVC: C = 4, kernel = linear

Applied to Algorithms: best preprocessor used

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| People | KNC | DTC | RFC | SVC | MLPC | MAX |
| Score | 0.351 | 0.16 | 0.458 | 0.569 | 0.567 | SVC |

NMIST:

-Preprocessing

Starting point

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MNIST | KNC | DTC | RFC | SVC | MLPC | Overall |
| None | 0.94 | 0.803 | 0.91 | 0.114 | 0.904 | 0.952 |
| Normalization | **0.952** | 0.804 | 0.892 | 0.167 | 0.943 |  |
| Standardization | 0.908 | 0.803 | **0.91** | 0.928 | 0.941 |  |
| PCA (default) | 0.94 | 0.743 | 0.666 | 0.114 | 0.862 |  |
| NMF(50)\* | 0.917 | 0.802 | 0.897 | **0.948** | **0.945** |  |
| KMC(100) | 0.919 | **0.807** | 0.87 | 0.114 | 0.77 |  |
| MAX | **0.952** | 0.807 | 0.91 | 0.948 | 0.945 |  |
|  |  |  |  |  |  |  |

Changes to PCA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| whiten | *0.522* | *0.726* | *0.652* | **0.829** | **0.882** |
|  |  |  |  |  |  |
| PCA(prev opt, randomized) | 0.94 | 0.743 | 0.666 | *0.781* | **0.885** |

Changes to NMF

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NMF(100) | **0.925** | *0.777* | 0.897 | *0.939* | **0.955** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NMF(prev opt, randomized) | *0.729* | *0.79* | **0.901** | *0.754* | *0.942* |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NMF(prev opt, nndsvda) | *0.759* | **0.817** | *0.897* | *0.0385* | *0.896* |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NMF(prev opt, mu) | *0.914* | *0.763* | **0.907** | *0.937* | *0.946* |

|  |  |
| --- | --- |
|  | RFC |
| NMF(prev opt, 1(kl)) | *0.905* |

Changes to KMeans

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| KMC(200) | **0.922** | *0.806* | **0.893\*** | 0.114 | *0.708* |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| KMC(prev opt, random) | 0.922 | **0.815** | *0.877* | 0.114 | **0.801** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| KMC(prev opt, n\_init 25) | 0.922 | *0.799* | 0.893 | 0.114 | *0.721* |

Fully Optimized Preprocessing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MNIST | KNC | DTC | RFC | SVC | MLPC | Overall |
| None | 0.94 | 0.803 | 0.91 | 0.114 | 0.904 | 0.955 |
| Normalization | **0.952** | 0.804 | 0.892 | 0.167 | 0.943 |  |
| Standardization | 0.908 | 0.803 | **0.91** | 0.928 | 0.941 |  |
| PCA | 0.94 | 0.743 | 0.666 | 0.892 | 0.885 |  |
| NMF | 0.925 | **0.817** | 0.907 | **0.948** | **0.955** |  |
| KMC | 0.922 | 0.807 | 0.893 | 0.114 | 0.801 |  |
| MAX | 0.952 | 0.817 | 0.91 | 0.948 | **0.955** |  |

Changes to algorithms:

kNNC: weights = distance

DTC: criterion = entropy, max\_depth = 15, min\_samples\_split = 3

RFC: n\_estimators = 50, max\_depth = 19

SVC: C = 3

MLPC: hidden\_layer\_sizes = (200,), activation = tanh,

Applied to Algorithms: best preprocessor used

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MNIST | KNC | DTC | RFC | SCV | MLPC | MAX |
| Score | 0.955 | 0.821 | 0.944 | 0.958 | 0.958 | SVC /MLPC |

Energy:

-Preprocessing

Starting point

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Energy | KNR | DTR | RFR | SVR | MLPR | Overall |
| None | 3.56 | 1.917 | 1.064 | 7.897 | 1.555 | 0.78 |
| Normalization | 4.769 | 7.469 | 3.596 | 7.755 | 7.038 |  |
| Standardization | **1.128** | 1.917 | 1.064 | **0.887** | **0.78** |  |
| PCA (default) | 3.56 | 2.18 | 1.184 | 7.897 | 4.829 |  |
| NMF (10) | 3.577 | 2.105 | 1.123 | 3.173 | 2.618 |  |
| KMC (10) | 4.769 | 7.554 | 4.713 | 6.913 | 14.249 |  |
| MIN | 1.128 | 1.917 | 1.064 | 0.887 | **0.78** |  |

Changes to PCA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| whiten | **1.034** | 2.18 | 1.184 | **0.929** | **0.786** |

Changes NMF

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NMF(50) | **2.673** | *2.175* | *1.191* | **2.737** | **1.215** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NMF(5, nndsvd) | **2.221** | **1.945** | **1.157** | **1.432** | **0.832** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NMF(prev opt, mu) | *1.742* | *1.684* | *1.069* | *1.112* | *1.025* |

Changes to KMeans

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| KMR(7) | *4.846* | **6.846** | *4.855* | *7.006* | **4.373** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| KMR(prev opt, random) | *4.816* | *6.908* | *4.831* | *6.897* | *5.945* |
|  |  |  |  |  |  |
| KMR(prev opt, n\_init 25) | 4.769 | 6.846 | 4.713 | 6.892 | 4.373 |

Fully Optimized Preprocessing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Energy | KNR | DTR | RFR | SVR | MLPR | Overall |
| None | 3.56 | 1.917 | 1.064 | 7.897 | 1.555 | 0.78 |
| Normalization | 4.769 | 7.469 | 3.596 | 7.755 | 7.038 |  |
| Standardization | 1.128 | 1.917 | 1.064 | **0.887** | **0.78** |  |
| PCA | **1.034** | 2.18 | 1.184 | 0.928 | 0.786 |  |
| NMF | 2.221 | 1.945 | 1.157 | 1.432 | 0.832 |  |
| KMC | 4.769 | 6.846 | 4.713 | 6.224 | 4.373 |  |
| MIN | 1.034 | 1.917 | 1.064 | 0.887 | **0.78** |  |

Changes to Algorithms:

kNNR: weights = distance

DTR: max\_depth = 11, min\_samples\_split = 16, min\_samples\_leaf = 2

RFR: n\_estimators = 100, min\_samples\_split = 2, max\_leaf\_nodes = 1000

MLPR: hidden\_layer\_sizes = (100,100), alpha = .3, beta\_1 = .89, beta\_2 = .9995

Applied to Algorithms: best preprocessor used

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Energy | KNR | DTR | RTR | SVR | MLPR | MIN |
| Score | 1.008 | 1.547 | 0.965 | 0.887 | 0.76 | MLPR |

Analysis:

Normalize:

Normalization transforms the data so that each feature vector has a Euclidian length of one. Normalization worked well for RFC when preprocessing people, as well as KNC for the MNIST dataset. It found no use in preprocessing the energy data set, returning worse MSE for all the algorithms after use. The same didn’t hold true for the classification algorithms however, and all saw benefit, even if minimal.

Standardization:

Standardization is a subtraction of the mean and division by the standard deviation to scale the data so that different metrics don’t skew the data. This allows for potentially incomparable data to be compared by a new standard metric. Standardization found the most use in the regression algorithms, similar to the last lab, as well as by the DTC and SVC for faces in the wild, and RFC for the MNIST dataset. Standardization for regression makes sense as it ensures consistency in the data being used to build the algorithm allowing for consistent outputs.

PCA:

PCA allows for the reduction of data, such that dimensionality can be reduced, and keeping the principal components that make up the data. PCA found limited use in preprocessing, being only used for the kNNR for the energy data set, with its effectiveness ranging from worse than no preprocessing, to second best. Due to the high dimensionality of the MNIST dataset it could be expected that PCA preprocessing would benefit the algorithms, but the opposite held true. For the energy dataset, PCA worked effectively on all the algorithms sans DTR and RFR which preformed just as well with and without preprocessing. PCA overall worked better for regression than classification.

NMF:

NMF works by factorizing a vector into, typically, two other matrixes such that none of the matrixes have zeros. This is helpful as it can make certain unreadable data for algorithms readable, but also by giving all the data a value other than zero, which could be a more meaningful value when processing. NMF found most use in classification, as well as by SVR and MLPR for regression. MLP and SV found the most use out of NMF across the testing both in regression and in classification, as both work to either distinguish the data or give weights to the data and as such working with non-zero values helped in both regression and classification processing.

KMeans:

KMeans creates clusters out of the data, making noise out of data that doesn’t fit into a cluster. KMeans found no use, when fully optimized not being good enough to be used by any algorithm. KMeans didn’t preform very well for any of the algorithms, returning low scores and high MSE.

kNN:

k-Nearest Neighbor finished better than decision trees on all tests and better than random forests on the MNIST dataset. Overall for such a simple algorithm with minimal changes to the standard setting for the MNIST and Energy datasets the algorithm worked well in comparison to the other algorithms.

DT:

The decision tree overall preformed worst. It returned the lowest score on the faces in the wild data set by twenty percent, with changes to the algorithm only resulting in worse scores. Due to the complexity of the faces data set underfitting is the likely culprit as the random forest was able to return a higher score, using one hundred trees. This means that the way the data was processed when done only once wasn’t enough to get the information out of the attributes, which was remedied by the random forest.

RF:

Random forests preformed much better on the faces in the wild data set than a single decision tree, five percent away from a fifty percent classification score. As seen in lab two, increasing the number of trees leads to increasingly better scores, with an eventual plateauing due to overfitting. The way that the random forest works allows for more information to be drawn out of each decision tree as each difference in attribute splitting goes to the overall mean classification or regressive value given at the end, which returns better values than a stand-alone decision tree as seen on the three datasets.

SV:

As seen in lab two, without preprocessing the support vector machine suffers and returns bad scores relative to the other algorithms with no preprocessing. Once the preprocessing is applied the algorithm works very well as the boundaries between data has a standard transformation throughout. The SVC returned the best classification of the faces in the wild and tied with MLPC for the MNIST dataset.

MLP:

The multilayer perceptron performed well returning the best or second-best score or MSE of the algorithms. Standardization and NMF helped the MLP most, as having a standard metric for the data, and no trivial data allows the weights to be altered and set in accordance to their importance. Overall MLP provided the most constant results in terms of high classification score and low MSE.

Conclusion:

The bulk of this reports analysis and experimental results is dedicated to preprocessing, not just because lab 2 focused more on the algorithms themselves, but also because as the data showed the biggest attributer to the success of an algorithm lies in large part to preprocessing. MLP and SVM algorithms benefit from good preprocessing, with the SVC and MLPC going from the worst scores for faces in the wild to the best scores simply by using the correct preprocessing. And while all the algorithms did benefit from adjusting the settings of the algorithm to better fit the data, the correct preprocessing is what allowed the data to be fit to the algorithm and give the best results. NMF and standardization saw the most use, which makes sense, as both give every attribute, as well as every value, a meaningful value, whether it be through nonzero values, which all values can be given a none trivial weight, or transforming the data to have a standard uniform metric throughout, making comparisons among the data more meaningful. The only conclusion that can come from the data is the true importance of preprocessing, as it gives the algorithms meaningful data to fit and predict.