Machine learning second programming exercise: Digits Classification

Atit Bashyal
Rana Hamza Intisar

April 2019

1 Task outline

The objective of this report is to serve as a documentation for the training and testing of a linear classification function used to classify digit images in the Digits data-set used in the machine learning course, spring 2019. To train a linear classifier for the data-set as outlined in the assignment task sheet we have implemented a processing pipeline from feature extraction leading to classifier training. In training the classifier cross-validation and regularization techniques have been preformed to tune the classification performance of the classifier.

2 Data-set preparation

2.1 Testing and Training Data split

The original data-set contains image vectors x_i of dimension m = 240. The values of each of the vector components in the m dimensions indicate greyscale values,integers ranging from 0 to 6. The original data-set contains 200 instances of each digit ranging from 0 to 9, i.e. a total of N=2000 instances. In mathematical formalism the original dataset is reperesented by

$$x_i \in \mathcal{R}^m \mid i \in 1, 2....N \tag{1}$$

Figure 1: examples of images contained in digits data-set

In order to train and finally test the performance of classifier, we create training and testing subsets of the original data-set. From the original data-set the ordered instances of the 200 image vectors corresponding to a particular digit was taken and divided into training and testing data points. Resulting in training and testing subsets containing 100 instances of each digit, i.e a total of 1000 image data points.
As classification is a supervised learning task, each image i ($i \in 1, 2...N$) in both the training and testing data-set must be represented as a labelled pair:

$$(x_i, y_i) \mid x_i \in \mathcal{R}^m, y_i \in (c_1, ...c_k) \tag{2}$$

To create such labelled pairs of data, each image in both the training and testing subsets were assigned a class label $c_j \mid j \in 1, 2..k$ where k is the number of classes present, in our case k= 10. The class labelling was done with a map corresponding to the mapping: c_1 = the "1" patterns, ..., c_9 = the "9" patterns, c_{10} = the "0" patterns. As a result the original data-set where each image was represented by a vector x_i is transformed into data-set where each image is represented by a labelled pair of image vector x_i and class label y_i.

The final step in preparing the data is to represent the class labels y_i of each image with binary class-indicator vector (z_i) $z_i \in \mathcal{R}^k$. With the class labels represented as numerical vectors, both the training and test data-set become labelled pairs $(x_i, z_i) \mid i \in 1, 2...N$ where, $z_i \in \mathcal{R}^k$ and $x_i \in \mathcal{R}^m$.

2.2 Feature extraction

Feature extraction is seen as the first stage of any classification task. Working with the raw data poses several problems in many machine learning tasks. The most common problem is the problem of over-fitting and under-fitting. A second problem that arises while using high dimensional features is the inherit difficulty of estimating distributions of data-points that are widely scattered in the data value space. Even if we estimate the distributions somehow, there is a greater chance we will face the problem of numerical overflow arising form the dimensionality of our data value space. Condensing the high dimensional raw input patterns i.e $x_i \in \mathcal{R}^m$ in to a lower dimensional feature vector $f(x_i) \in \mathcal{R}^l$ where l<<m helps us in overcoming the both of these problems. More specifically, fine tuning the classification algorithm to figure out the optimal number features to be used helps us overcome the problem of over-fitting or under-fitting. In this particular project the fine tuning of the number of features to be used in our classification algorithm will be done using k-fold cross-validation, which will be discussed in the sections that follow. For now we list the two different ways in which we will implement the feature extraction process.

- Principal Component (PC) Features: Running a Principle Component Analysis on the training data-set, we extract the PC vectors and select a group of leading PC $u_1.....u_l$ based on the eigen value spectrum of the PC vectors. The features we extract using these selected PC vectors are defined as the projections of the centralized image patterns $\tilde{x_i}$ on these few PC vectors. Mathematically for any image pattern $\tilde{x_i}$ we obtain a feature map

$$\mathcal{F} : \mathcal{R}^m \rightarrow \mathcal{R}^l, \qquad x_i \mapsto (u_1^T(\tilde{x_i}), ..., u_l^T(\tilde{x_i})) \tag{3}$$

- K-means clustering based features: In this method we identify l cluster groups/sets $S_1....S_l$ in the training data cloud and represent them by l codebook vectors $C_1.....C_l$. The codebook vectors mathematically represent the mean of all the image vector points falling in the respective cluster set. Then for a image pattern x_i its features will be represented by a K-dimensional feature vector which contains the metric distances of the pattern to each of the codebook vectors.Mathematically for any image pattern x_i we obtain a feature map

$$\mathcal{F} : \mathcal{R}^m \rightarrow \mathcal{R}^l, \qquad x_i \mapsto (|| \ (x_i - C_1) \ ||, ..., || \ (x_i - C_l) \ ||) \tag{4}$$

With the image vectors $x_i \in \mathcal{R}^m$ reduced into feature vectors $f(x_i) \in \mathcal{R}^l$, both the training and test data-set are transformed into labelled pairs $(f(x_i), z_i) \mid i \in 1, 2...N$ where, $z_i \in \mathcal{R}^k$ and $f(x_i) \in \mathcal{R}^l$.

3 The Classification algorithm

The classification algorithm that we implement in classifying the images of the digits data-set, using the extracted feature vectors $f(x_i) \in \mathcal{R}^l$ will be built around a decision function D:

$$d : \mathcal{R}^l \mapsto \mathcal{R}^k \tag{5}$$

In this assignment we assume that the decision function comes from the class of linear functions and search for a decision function d within the linear functions. Within the search space of the linear functions we limit ourselves to affine functions, by padding a bias vector with the feature vector. In general mathematical terms the affine function consists of a linear function plus a constant offset. The linear function will take the form of lXk matrix W and the offset term will manifest itself because of the padded bias vector. In this assignment we use a constant offset by padding 1 the end of the original feature vector. Mathematically our decision function now takes the form:

$$d(f(x_i)) = W f(x_i) + b \tag{6}$$

The decision function, that we aim to find must take the extracted feature vectors $f(x_i)$ and return a predicted class indicator vectors $z_i^{'}$, such that the predicted class indicator vectors $z_i^{'}$ has the smallest possible squared distance from the original class indicator vectors z_i. This means that we must find an optimal linear function weight matrix W_{opt} such that on average minimizes $|| \ z_i^{'} - z_i \ ||^2$. The objective of finding this optimal function is thus set up as a linear regression problem which mathematically takes the form;

$$W_{opt} = argmin_{W \in \mathcal{R}^l} \frac{1}{N} \ || \ z_i^{'} - z_i \ ||^2 \tag{7}$$

The result of this linear regression problem is given by the solution:

$$W_{opt} = (\Phi^{'}\Phi)^{-1}\Phi Z \tag{8}$$

where Φ is a Nxl matrix of the row-wise sorted feature vectors and Z is the Nxk matrix of the class indicator vectors(target vectors). We can regularize the weight matrix W_{opt} by using the regularization technique of ridge regression. The ridge regression control the flexibility of linear regression with an L2 norm regularizer. Adding this regularizing parameter in the computation of W_{opt} changes the solution in eqn 8 to

$$W_{opt} = (\Phi^{'}\Phi - \alpha^2 I)^{-1}\Phi Z \tag{9}$$

where I is a l x l Identity matrix. We find the best regularization coefficient α^2 by cross-validation, and will be discussed in deatail in the section that follows.

3.1 The classification steps

After the classification problem has been set up as a ridge regression, the immediate step is to find the optimal weight matrix that gives us the best predicted class indicator vectors $z_i^{'}$, by which we understand that the absolute squared distance of the predicted indicator vector from the original indicator vector must me minimal. In this particular assignment we have decided to extract features in two different ways, one using the PC vectors and the second Using the K means cluster center vectors. For both these model choices In order to compute this optimal weight matrix W_{opt} using equation 9, we need to consider finding from cross validation:

- The optimal number of features to use i.e. running cross validation to choose the optimal number of PC vectors to use or the optimal number K of cluster centers to use.
- The optimal regularization coefficient α^2 for the ridge regression weight matrix W_{opt} computed using on the optimal number of features extracted from the training data.

Turning back to the original task of classifying the digits dataset, at this point, our original data set has been transformed into two subsets of training and test data set where an image i, is represented as a labelled pair $(f(x_i), z_i) \mid i \in 1, 2...N$ where, $z_i \in \mathcal{R}^k$ and $f(x_i) \in \mathcal{R}^l$. The steps we now follow to train a classification algorithm using this training data-set as follows:

- Step1 - determine the number of features to be used: for both the features extracted from the PCA method and the K means clustering method, we determine the optimal number of feature to be extracted. To do so we use the we setup a 5 fold cross validation scheme. We run the cross-validation while doing a grid search of the optimal number of features. We provide a grid of possible values for the number of features. The for each of these grid values the we run a 5-fold cross validation to compute a linear regression weight matrix W_{opt} to approximate the target vector z_i for any given feature $f(x_i)$ with dimension corresponding to the particular grid value of feature number being used. In this part we set the regularization coefficient α^2 to zero and run a linear regression. Finally for each run of the Cross validation scheme, i.e for each grid value provided the Cross validation run outputs four metrics: the Mean Square Error , Miss Classification rate for the cross validation training sets (MSE_{train}, $MISS_{train}$) and the Mean Square Error , Miss Classification rate for the cross validation test sets ($MSE_{validate}$, $MISS_{validate}$). Finally comparing the performance of the different

trained models using these four metrics we find the optimal number of features i.e the number of features when used results in a model with the least MSE and MISS for the validation set

- Step2- determine the optimal ridge regression regularization coefficient: After we determine the optimal number of features we use in our model, in the next step we run a 5 fold cross validation scheme to determine the optimal regularization coefficient. similar to the cross validation scheme used before, we provide a grid of possible α^2 values, for each grid value, using the optimal number of features determined previously the cross validation scheme computes the optimal regression weight matrix W_{opt},given by formula 9. For each grid value provided the Cross validation run outputs four metrics: the Mean Square Error , Miss Classification rate for the cross validation training sets (MSE_{train}, $MISS_{train}$) and the Mean Square Error , Miss Classification rate for the cross validation test sets ($MSE_{validate}$, $MISS_{validate}$). Finally comparing the performance of the different trained models using these four metrics we find the optimal regularization coefficient.i.e. the α^2 value when used results in a model with the least MSE and MISS for the validation set.

- step3: With the optimal values of the model parameters(features) and the regularization coefficient, we finally train a final best model using these fine tuned values of model parameter and regularization coefficient. We will actually train two final models built from the two different feature extraction methods. as our final step of classification we look at how these two models generalize to the use of our test data. we will compare the MSE_{test}, $MISS_{test}$ for the two models, which will enable us to see if feature extraction using PCA or K-means clustering works the best for the digit-classification task at hand.

4 Classification algorithm Implementation and Results

4.1 Linear regression and ridge regression with PCA extracted features

we first compute a linear regression weight matrix with the full PCA extracted features from the training data-set. using this weight matrix we compute the predicted class label for both the training and test data-set. We then compute the four metrics MSE_{train}, $MISS_{train}$, MSE_{test}, $MISS_{test}$ for this model. the figure below shows the value of these metrics of this initial basic model.

```
    ....
mse_train: 0.0134
miss_train: 0.06699999999999995
mse_test: 0.0148
miss_test: 0.07399999999999995
```

Figure 2: evaluation metrics of the initial basic model: linear regression with all PCA features

Next we plot the eigen value spectrum for the PC vectors and try to find a cutoff value for the maximum number of PC vectors we can use to extract our features. The graph below shows the eigen value spectrum of the PC vectors obtained for images in our training data-set.

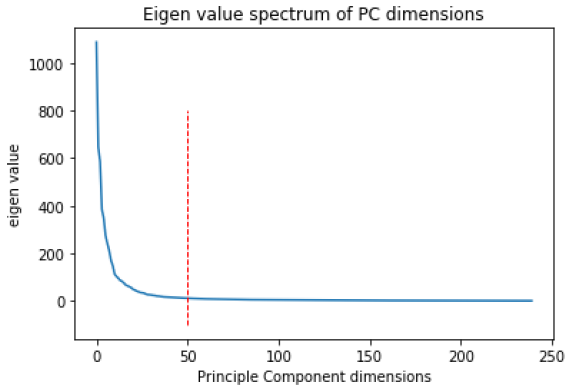

Figure 3: Eigenvalue spectrum for all the the PC vectors

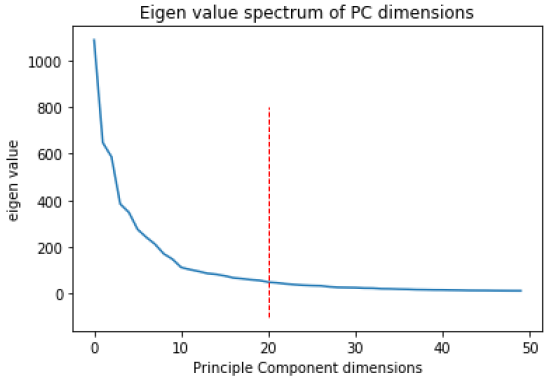

Figure 4: Eigenvalue spectrum for all the the top 50 PC vectors

Figure 3 shows the eigen value spectrum plot for all PC vetors of our data-set, from figure 3 we see that we need to use less than 50 PC vectors to be able to capture the maximum variance in the data-set. Figure 4 where the plot shows the eigen value spectrum for only the 50 PC vectors, the plot more clearly shows that even using the top 20 PC vectors will explain much of the variance in our data-set. In the next step we thus compute a linear regression weight matrix with the PCA features extratced from the top 20 PC vectors from the training data-set. using this weight matrix we compute the predicted class label for both the training and test data-set. We then compute the four metrics figure 5 shows the values of these metrics.

```
mse_train: 0.014
miss_train: 0.06999999999999995
mse_test: 0.0138
miss_test: 0.06899999999999995
```

Figure 5: Evaluation metrics of model where linear regression with 20 extracted features from PCA

Reducing the number of features used shows a drop in the mean square error and miss-classification rate of the model when used on the test data. Next we run a k-fold cross validation on the linear regression model to obtain the optimal number of feature to use. we do a grid search for the number of features by using a grid with values ranging from 10 to 100. Figure 6 shows the graph of the four evaluation metrics of the cross validation step for the different choices of feature number in the grid.

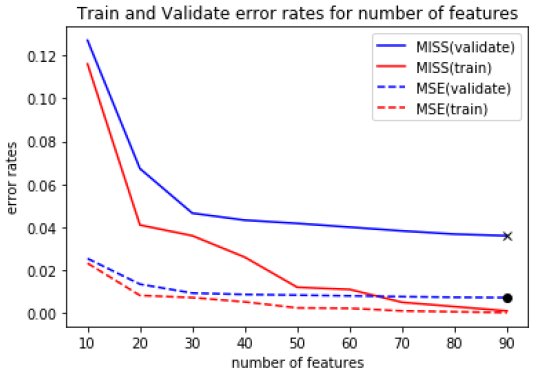

Figure 6: Evaluation metrics of cross validation on the number of features

As shown by the graph the optimal number of features to use in training our model is 90. In the next step, using these features we run a cross-validation scheme to find the optimal value of the regularization coefficient α^2. Figure below shows the graph of the four evaluation metrics of the cross validation step for the different choices of α^2 number in the grid. Cross-Validation shows that the optimal alpha has a value of 0.3

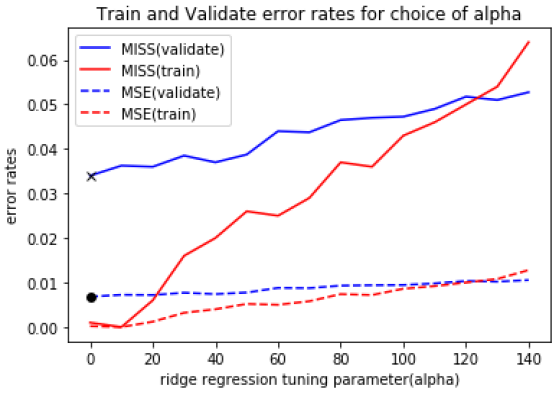

Figure 7: Evaluation metrics of cross validation on alpha

In the final step, we train a ridge regression model using the selected optimal number of PCA feature and the optimal regularization coefficient. we compute the evaluation metrics to see how this model will generalize when the test data is used, the metrics are shown in figure 8.

```
mse_train: 0.0086
miss_train: 0.04300000000000004
mse_test: 0.008
miss_test: 0.040000000000000036
```

Figure 8: Evaluation metrics of optimal model with PCA features and ridge regression

4.2 Linear regression and Ridge regression with K-means extracted features

Similar to the basic initial model of the PCA extracted features, we first compute a linear regression weight matrix with features extracted using 100 cluster centers from the training data-set. using this weight matrix we compute the predicted class label for both the training and test data-set. We then compute the four metrics MSE_{train}, $MISS_{train}$, MSE_{test}, $MISS_{test}$ for this model. the figure below shows the value of these metrics of this initial basic model.

```
mse_train: 0.1616
miss_train: 0.808
mse_test: 0.1742
miss_test: 0.871
```

Figure 9: Evaluation metrics of initial model using features from 100 cluster centers

Next we run a k-fold cross validation on the linear regression model to obtain the optimal number of feature (cluster centers) to use. we do a grid search for the number of features by using a grid with values ranging from 50 to 1000. Figure 10 shows the graph of the four evaluation metrics of the cross validation step for the different choices of feature number in the grid.

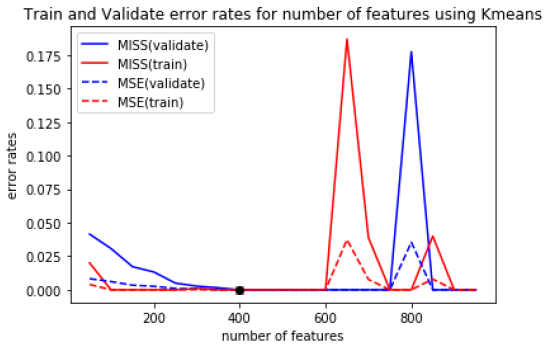

Figure 10: Evaluation metrics of cross validation on number of clusters

as shown by the graph the optimal number of features to use in training our model is 400. in the next step using these features we run a cross-validation scheme to find the optimal value of the regularization coefficient alpha,figure 11 shows the graph of the four evaluation metrics of the cross validation step for the different choices of α^2 in the grid.

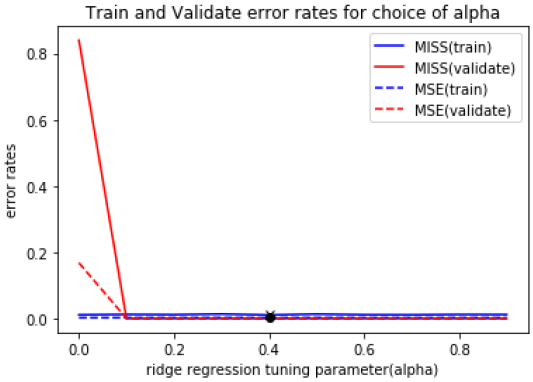

Figure 11: Evaluation metrics of cross validation on alpha

In the final step, we train a ridge regression model using the selected optimal number of K-means feature and the optimal regularization coefficient. we compute the evaluation metrics to see how this model will generalize when the test data is used, the metrics are shown in figure 12.

```
mse_train: 0.0088
miss_train: 0.04400000000000004
mse_test: 0.0092
miss_test: 0.04600000000000004
```

Figure 12: Evaluation metrics of optimal model using features from 400 cluster centers and optimal alpha

5 Conclusion

The implemented steps of classification shows as expected that the ridge regression model preforms better than the linear regression model in the classification task. The steps taken by us in tuning the model parameters show that for both method of feature extraction, using the optimal number of feature along with the optimal ridge regression coefficient force the betterment of the performance of the models when used on the testing dataset.