Exercise for MA-INF 2213 Computer Vision SS20
24.04.2020
Submission deadline: 04.05.2020

Important: Use Python 3.6 for your solutions. You are not allowed to use any additional python modules beyond the ones imported in the templates. Otherwise you won't get any points. Code with runtime errors or returning obviously rubbish results (e.g. nans, inf, meaningless visual output in future exercises) will give you at most half of the points. Pooints are also assigned to quesitons inside the coding exercises.

You can complete the exercise in groups of two, but only onse submission per group is allowed. Include a readme.txt file with your group members into each solution. Points for solutions without readme file will only be given to the uploader.

1 Regression

We consider the problem of object pose regression. The world variables $\mathbf{y} = [y_0, y_1]$ are given by rotating an object along two coordinate axes. Also given are observed 510 dimensional $\mathbf{x} = [x_0, x_1 ... x_{509}]$ PHoG [1] features per image.

Appropriate $regression_*.txt$ files are provided for training regressing parameters and evaluating them. Each row holds the concatenation $[y_i, x_i]_{1\times512}$ for image I_i.

Use the maximum likelihood rule for learning. To evaluate the performance, take the maximum likelihood parameters to predict the values $\hat{y_i} \in 0, 1$ on the val and test sets. As a performance metric, report the MSE relaitve to variance of $\mathbf{y}\frac{MSE(\hat{y_i},y_i)}{Var(y_i)}$ for $i \in 0, 1$.

1. **Linear Regression:** Learn a linear regressor using the training data for both world variables $y_i i \in 0, 1$ independently and evaluate its performance on the test data. *(3 Points)*

2. **Dual Model Regression**: Learn a dual-model regressor for both variables independently using RBF function as the kernel. Do not use regularization, i.e. simply perform linear regression in the feature space you map $\mathbf{x}$ to. Split the training data into a train and a val split of equal size. Estimate the right value for σ (standard deviation in the RBF kernel) using the val set. Also evaluate regressor's performance on the test data. What do you think about the val set proposed in the template? What does the regressor become equivalent to if σ approaches 0? What happens to the regression if σ approaches infinity? *(6 Points)*

3. **Non Linear Regression:** Learn a non-linear regressor for both variables independently using RBF kernels. The centers for RBF kernels will be learnt by reducing the observed features into codebooks. Do not use regularization, i.e. simply perform linear regression in the feature space

you map $\mathbf{x}$ to. Use the same train and val split as for dual regression. Estimate the optimal number of clusters and σ on the val set. Also evaluate regressor's performance on the test data. *(2 Points)*

2 Classification

We consider the problem of binary classification (bottles and horses). Given are the 510 dimensional PHoG features for each class, separated for training and testing. The data is arranged in a similar manner as above.

1. **Logistic Regression:** Using the bottles as positive and the horses as negative examples, learn a linear classifier based on logistic regression. As loss function, use the loss function for logistic linear regression. (It is essentially the cross entropy loss) You may choose a simple gradient descent or the Newton's method for optimization.Train your classifier for 10000 iterations printing the loss and the accuracy every 1000 iterations. You should reach 75% accuracy on the test set. Can your model get stuck in a local minimum and why?

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

   where TP and FN stand for true positive and false negative respectively. *(5 Points)*

2. **Derivatives of loss function:** Given the loss function for logistic linear regression parametrized by w (Eq. 9.6 in [2])

$$L = -\sum_{i=1}^{I} y_i log\left(\frac{1}{1 + exp(-w^T x_i)}\right) - \sum_{i=1}^{I}(1 - y_i) log\left(\frac{exp(-w^T x_i)}{1 + exp(-w^T x_i)}\right)$$

   Show the gradient to be

$$\frac{\partial L}{\partial w} = \sum_{i=1}^{I}\left(\frac{1}{1 + exp(-w^T x_i)}\right) x_i \tag{2}$$

   (4 Points)

References

[1] Bosch, A. and Zisserman, A. and Munoz, X. *Representing shape with a spatial pyramid kernel.* In Proceedings of the International Conference on Image and Video Retrieval, pp 401-408, 2007

[2] Bishop, C. *Pattern Recognition and Machine Learning.* Springer 2006