CSE 569: Fundamentals of Statistical Learning
Final Project Report
Topic: Combining Multiple Features in Classification

Team:

Vignesh Iyer 1207653833
Sagar Sangani 1205141934

Preprocessing

- For the training data, X1, X2, X3 we calculated the mean vector for each feature matrix.
- For X1, the minimum mean vector value was 2.0240e-04 and the maximum value was 0.0040. Thus the max : min ratio was 19.8522.
- This ratio is quite high. In order to avoid attributes biasing some of the other attributes, we normalized the values to [0,1].
- After normalization, the ratio turned out to be 3.8671e-16.
- We preserved the max and the min values of training sets X1,X2 and X3 for normalizing their corresponding test sets.

Step 0

- For each of the feature matrix (X1, X2, X3), we trained a linear SVM model (h1, h2, h3) using libSVM. Following are the accuracy rates that we achieved.

Feature Matrix	Classification Accuracy (Un-Normalized Dataset)	Classification Accuracy (Normalized Dataset)
X1	29.31%	42.01%
X2	28.09%	39.19%
X3	28.94%	41.80%

- We trained the model using the parameters -t 0 -c 10 -b 1
- -b 1 was used to estimate the posterior probabilities.

- To experiment with the feature matrices, we tried to use the libSVM feature selection tool. The feature selection tool returned the top features based on F-Score.
- We selected those features whose F-Score values >= 0.1
- Although, the classification accuracies were almost in the same range of the values mentioned in above table, we got these accuracies by reducing a significant number of dimensions. Following are our observations.

Feature Matrix	Number of dimensions (originally 1000)	Classification Accuracy
X1	862	41.32%
X2	583	35.68%
X3	506	36.75%

- We used the fselect.py provided by libsvm to calculate the F-Score values

Step 1

- Using the previously trained model on testing set we not only get prediction vector, but also posterior probabilities as $p(w_i|x_k)$ for corresponding feature set X_k
- In this step we combine features by obtaining fused probability $p(w_i|x)$ is obtained as arithmetic mean of individual feature probabilities and can be used to predict class i for argmax $p(w_i|x)$
- The model used with both given test data and normalized test data we observe following:

Fused Posterior Probability	Classification Accuracy (Un-Normalized Dataset)	Classification Accuracy (Normalized Dataset)
argmax $p(w_i\|x)$	45.41 %	60.86%

- Comparing the accuracy of Step 1 with Step 0, we see,

Step 1 Accuracy (Accuracy by fusion of classifiers)	Step 0 Accuracy (Max accuracy reported using X1 alone)
60.86 %	42.01 %

Step 2

- In this step we combine all three features by linear cascade and obtain single input vector X = [X1, X2, X3] and train a linear SVM model.
- We trained the model with parameters -t 0 -c 10 -b 1
- Although linear cascade model accuracy dropped as compared to Step 1, but still showed a drastic improvement over classification using any of the three single feature models.
- The observations made for this feature combination method were as follows:

Cascaded Feature Matrix	Classification Accuracy (Un-Normalized Dataset)	Classification Accuracy (Normalized Dataset)
[X1 X2 X3]	48.16%	57.36%

- Comparing the accuracy of step 2 with step 1, we get

Step 2 Accuracy (Accuracy by cascade of classifiers)	Step 1 Accuracy (Accuracy by fusion of classifiers)
57.36 %	60.86 %

Step 3

- For the first part of this step as per the given kernel function for chi-squared kernel, we pre-compute kernel matrices for modelling and prediction as K_j and KK_j respectively for corresponding feature set X_j where $1 \leq j \leq 3$
- We precomputed the kernel matrix for training (K_j) the model by computing the chi square function value for every (training,training) pair. We got a kernel matrix of dimension 4786x4786
- For precomputed kernel matrix of testing set (KK_j), we computed the chi square function value for every (testing,training) pair. We got a kernel matrix dimension of 1883x4786
- Using this training kernel matrix K_j and setting '-t 4' as training parameter, we trained the model.
- We used the trained model with prediction kernel matrix KK_j and made the following observations for each feature set.

Feature Matrix	Classification Accuracy (Normalized Dataset)
X1	46.26%
X2	45.94%
X3	51.14%

- We can conclude from above that using precomputed chi-square kernel function instead of a linear kernel, the SVM model improved accuracy by 6% on average . However these results are still low compared to results obtained by feature combination techniques in Step 1 & Step 2. Comparing the results with Step 0, we see

Feature Matrix	Classification Accuracy in Step 3	Classification Accuracy in Step 0
X1	46.26%	42.01%
X2	45.94%	39.19%
X3	51.14%	41.80%

- Keeping, this in mind we move on to combine the kernel matrices to form a single kernel using two possible approaches
- For first method (a) of kernel fusion, we fuse all three kernel matrices previously obtained as arithmetic mean of them. Following was the prediction result of this model

Feature Matrix	Classification Accuracy
Ka	64.1%

- As we can see and conclude, that fusion of kernel vastly improved the accuracy as compared to each of individual unfused feature as well as all the fusion methods tried previously
- The next method (b) of kernel fusion involves geometric mean of all individual feature kernels and using it to train a model. This model produced following prediction results

Feature Matrix	Classification Accuracy
Kb	64.1%

- On comparing Ka and Kb, we see that both techniques resulted in same accuracy.

Contributions

The contributions by the team members were equal. Since the individual contributions were requested, we are listing them below.

Vignesh Iyer

- Explored the libSVM package and got familiarized with the library usage.
- Tried the feature selection tool to select features based on F-score.
- Developed the fusion of classifiers and linear cascaded models and calculated its accuracies.

Sagar Sangani

- Explored the libSVM package and got familiarized with the library usage.
- Normalized the data to improve accuracy.
- Developed the precomputed chi-square kernel for training and testing sets and calculated its accuracies.

Acknowledgements

Firstly, We would like to thank Prof. Baoxin Li for giving us this opportunity to explore the field of machine learning by implementing the course project.

We would also like to thank the TAs, Parag and Xu for helping us in our queries regarding the project and also suggesting some variations to improve our SVM model.

References

Sources that were useful in successful completion of this project are listed below

- https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf
- README file in the libsvm package
- https://www.csie.ntu.edu.tw/~cjlin/papers/features.pdf
- http://stackoverflow.com/questions/7177753/bad-result-when-using-precomputed-chi2-kernel-with-libsvm-matlab
- http://stackoverflow.com/questions/7715138/using-precomputed-kernels-with-libsvm
- http://stackoverflow.com/questions/3047940/feature-selection-methods-in-matlab
- https://www.youtube.com/watch?v=SRVswRH5Q7E&list=PLTbgWjKpS53joS5QxGep5a98TnoQg3EUW&index=29
- http://crsouza.com/2010/03/kernel-functions-for-machine-learning-applications/