Feng Wei

wei8

ECE 8560 Takehome#3

SVM & C-Means

Description of SVM software used.

In this assignment I used SVM model from scikit-learn, it is a open source project supported by Google and the implementation is based on libsvm.

Parameters:

- **C** : Penalty parameter C of the error term.
- **kernel** : Specifies the kernel type to be used in the algorithm.
- **degree** : Degree of the polynomial kernel function (‘poly’). Ignored by all other kernels.
- **gamma** : Kernel coefficient for ‘rbf’, ‘poly’ and ‘sigmoid’.
- **coef0** : It is only significant in ‘poly’ and ‘sigmoid’.
- **probability** : boolean, optional (default=False) Whether to enable probability estimates.
- **shrinking** : boolean, optional (default=True) Whether to use the shrinking heuristic.
- **tol** : float, optional (default=1e-3) Tolerance for stopping criterion.
- **cache_size** : float, optional Specify the size of the kernel cache (in MB).
- **class_weight** : {dict, ‘balanced’}, optional Set the parameter C of class i to class_weight[i]*C for SVC. If not given, all classes are supposed to have weight one. The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y))
- **max_iter** : int, optional (default=-1) Hard limit on iterations within solver, or -1 for no limit.
- **decision_function_shape** : ‘ovo’, ‘ovr’ or None, default=None Whether to return a one-vs-rest (‘ovr’) decision function of shape (n_samples, n_classes) as all other classifiers, or the original one-vs-one (‘ovo’) decision function of libsvm which has shape (n_samples, n_classes * (n_classes - 1) / 2).
- **random_state** : int seed, RandomState instance, or None (default)

Attributes:

- **support_** : Indices of support vectors.
- **support_vectors_** : Support vectors.
- **n_support_** : Number of support vectors for each class.
- **dual_coef_** : Coefficients of the support vector in the decision function.
- **coef_** : Weights assigned to the features (coefficients in the primal problem). This is only available in the case of a linear kernel. coef_ is a readonly property derived from dual_coef_ and support_vectors_.
- **intercept_** : Constants in decision function.

SVM results with a linear (dot-product) kernel.

- Show the support vector set.

  There are 2288 support vectors, and each of the vector has 2289 features

- Show the hyperplane parameters (This is important).

  Hyperplane parameters

$$W = [-0.01877985, 0.00894769, -0.00117651, 0.03578019]$$
$$b = 1.61494972$$

- Determine the classification performance with this SVM.

$$P(error) = P(\bar{x}\ is\ assigned\ to\ the\ wrong\ class)$$
$$P(error) = \frac{620 + 1246}{5000 + 5000} = 18.66\%$$

$Confusion\ Matrix$

Class\Assigned	C_1	C_2	$Total$
C_1	4380	620	5000
C_2	1246	3754	5000

- Compare these results with your results from Takehomes #1 and #2.

$Method$	P_{error}	$Compexity$
Bayesian classifier	8.9%	/
Separating Hyperplanes	18.9%	/
K-NN (5-NN)	10.5%	Use K-D tree
PCA	22.4%	Reduce dimension
Linear SVM	18.66%	Very slow

SVM results with a rbf kernel.

When use $\gamma = 0.001$ we get the highest accuracy.

- Show the support vector set.

  There are 1191 support vectors, and each of the vector has 1352 features

- Show the hyperplane parameters (This is important).

  Hyperplane parameter: W is a 1352 dimensional vector

$$W = [-1, -0.36269108, -1, ..., 1., 1., 1.]$$
$$b = 0.56387271$$

- Determine the classification performance with this SVM.

$$P(error) = P(\bar{x}\ is\ assigned\ to\ the\ wrong\ class)$$
$$P(error) = \frac{289 + 668}{5000 + 5000} = 9.57\%$$

$Confusion\ Matrix$

Class\Assigned	C_1	C_2	$Total$
C_1	4711	289	5000
C_2	668	4332	5000

- Compare these results with your results from Takehomes #1 and #2.

$Method$	P_{error}	$Compexity$
Bayesian classifier	8.9%	/
Separating Hyperplanes	18.9%	/
K-NN (5-NN)	10.5%	Use K-D tree
PCA	22.4%	Reduce dimension
Linear SVM	18.66%	Slow
Rbf SVM	9.57%	Faster than linear svm

Results of crisp c-means

- **Original classes**

Class	Number	Center
1	5000	(50.11712203 -4.97038793 -24.81182102 -49.81198585)
2	5000	(24.13111767 -0.11837553 -25.04828746 0.29147143)
3	5000	(49.70218541 5.40154144 24.55009373 -49.93734216)

- **First use Euclidean distances.**

- **2-means**

Cluster	Number	Cluster center
1	11193	(51.37486812 0.29416636 -2.60798235 -49.38854549)
2	3807	(11.80736318 -0.4529109 -25.53752299 14.48213362)

- **3-means**

Cluster	Number	Cluster center
1	3539	(8.88992477 1.04749464 -25.14278388 16.12511828)
2	8479	(52.29343506 -13.17339471 -14.78625645 -47.86449585)
3	2982	(48.63096293 36.75761886 29.4844124 -49.86887789)

- **4-means**

Cluster	Number	Cluster center
1	2098	(50.20289906 -39.13463488 31.1432395 -49.98872054)
2	2336	(48.29196953 47.62616538 26.78523444 -49.94902897)
3	7254	(52.72354363 -4.03597486 -23.65981069 -45.90313343)
4	3312	(5.74396175 0.39908105 -24.98864206 17.34142622)

- **5-means**

Cluster	Number	Cluster center

1	2369	(60.61477337 -1.18877172 -25.01607198 15.36479156)
2	6185	(48.47298907 -4.10012992 -23.36122253 -53.13249433)
3	2046	(-21.30394363 0.45176967 -24.852292 6.94437635)
4	2094	(50.95312435 -39.03222411 31.23243838 -49.70491145)
5	2306	(49.12455411 47.92816658 27.18944266 -49.86174619)

- **Use Manhattan distances.**

Since we compute μ in a Euclidean space, i think there will have some bias if we just use Manhattan distance to compute the cluster center. Below is the Manhattan distance cluster result. Actually i didn't have much confidence about it.

- **K-means Manhattan distance**

Cluster	Number	Cluster center
1	4332	(18.38992477 4.0678473 -5.13238388 6.14541728)
2	7563	(25.435057836 -12.27449661 -17.38525753 -27.86339787)
3	3105	(46.47055273 26.74361886 19.3832154 -39.56553389)

- ❖ **Compare these results see if any clusters naturally develop.**

In 3-means 4-means and 5 means the below clusters have some similarity, their means is similar to each other.

Method	Cluster	Number	Cluster Center
3-means	3	2982	(48.63096293 36.75761886 29.4844124 -49.86887789)
4-means	2	2336	(48.29196953 47.62616538 26.78523444 -49.94902897)
5-means	5	2306	(49.12455411 47.92816658 27.18944266 -49.86174619)

- ❖ **Assess whether the clusters found above are related to the known (estimated) class means.**

Method	Cluster	Number	Cluster Center
Original	1	5000	(50.11712203 -4.97038793 -24.81182102 -49.81198585)
4-means	3	7254	(52.72354363 -4.03597486 -23.65981069 -45.90313343)
5-means	2	6185	(48.47298907 -4.10012992 -23.36122253 -53.13249433)