CSE512 - Machine Learning - Homework 3

Yang Wang, 110014939
Computer Science Department
Stony Brook University
yang.wang@stonybrook.edu

1. Question 1 – Boosting

1.1. Surrogate Loss Function

$$
\begin{aligned}
\epsilon_{Training} &= \frac{1}{N}\sum_{j=1}^{N}\delta(\ H(x^j) \neq y^j\ ) \\
&= \frac{1}{N}\sum_{j=1}^{N}\begin{cases} 1, & sgn\{f(x^j)\} \neq y^j \\ 0, & else \end{cases} \\
&= \frac{1}{N}\sum_{j=1}^{N}\begin{cases} 1, & f(x^j)\cdot y^j \leq 0 \\ 0, & else \end{cases} \\
&\leq \frac{1}{N}\sum_{j=1}^{N} exp(-f(x^j)\cdot y^j)
\end{aligned}
$$

1.2. Surrogate Loss Function

$$
\begin{aligned}
w_j^{T+1} &= w_j^T \frac{exp(-\alpha_T y^j h_T(x^j))}{Z_T} \\
&= w_j^1 \frac{exp(-\alpha_1 y^j h_1(x^j))}{Z_1}\cdot\cdot\frac{exp(-\alpha_T y^j h_T(x^j))}{Z_T} \\
&= w_j^1 \frac{exp(-y^j \sum_{t=1}^{T}\alpha_t h_t(x^j))}{\prod_{t=1}^{T} Z_t} \\
&= \frac{1}{N}\frac{exp(-y^j f(x^j))}{\prod_{t=1}^{T} Z_t}
\end{aligned}
$$

$$
\sum_{j=1}^{N} w_j^{T+1} = \sum_{j=1}^{N}\frac{1}{N}\frac{exp(-y^j f(x^j))}{\prod_{t=1}^{T} Z_t} = 1
$$

$$
\prod_{t=1}^{T} Z_t = \frac{1}{N}\sum_{j=1}^{N} exp(-f(x^j)y^j)
$$

1.3. Greedy Optimization

1.3.1

$$
\text{Set}\quad \frac{\partial Z_t}{\partial \alpha_t} = -(1-\epsilon_t)exp(-\alpha_t) + \epsilon_t exp(\alpha_t) = 0
$$

$$
\text{We have}\quad \alpha_t = \frac{1}{2}ln(\frac{1-\epsilon_t}{\epsilon_t})
$$

$$
\begin{aligned}
\text{Thus}\quad Z_t^{opt} &= (1-\epsilon_t)\cdot\sqrt{\frac{\epsilon_t}{1-\epsilon_t}} + \epsilon_t\cdot\sqrt{\frac{1-\epsilon_t}{\epsilon_t}} \\
&= 2\sqrt{\epsilon_t(1-\epsilon_t)}
\end{aligned}
$$

1.3.2

$$
\begin{aligned}
Z_t^{opt} &= 2\sqrt{\epsilon_t(1-\epsilon_t)} \\
&= 2\sqrt{(\frac{1}{2}-\gamma_t)(\frac{1}{2}+\gamma_t)} \\
&= \sqrt{1-(2\gamma_t)^2} \\
&= exp[\frac{1}{2}ln(1-(2\gamma_t)^2)] \\
&\leq exp[-\frac{1}{2}(2\gamma_t)^2] \\
&= exp(-2\gamma_t^2)
\end{aligned}
$$

1.3.3

$$
\begin{aligned}
\epsilon_{training} &\leq exp(-2\sum_{t=1}^{T}\gamma_t^2),\quad \gamma_t \geq \gamma > 0 \\
&\leq exp(-2\sum_{t=1}^{T}\gamma^2) \\
&= exp(-2T\gamma^2)
\end{aligned}
$$

2. Question 2 – PCA via Successive Deflation

2.1. Covariance of the deflated matrix

$$
\begin{aligned}
&\tilde{C} \\
&= \frac{1}{n}\tilde{X}\tilde{X}^T \\
&= \frac{1}{n}[\,(I - v_1v_1^T)X\,][\,(I - v_1v_1^T)X\,]^T \\
&= \frac{1}{n}(I - v_1v_1^T)XX^T(I - v_1v_1^T) \\
&= \frac{1}{n}[XX^T - v_1v_1^TXX^T - XX^Tv_1v_1^T + v_1v_1^TXX^Tv_1v_1^T] \\
&= \frac{1}{n}[XX^T - v_1(n\lambda_1v_1)^T - (n\lambda_1v_1)v_1^T + v_1v_1^T(n\lambda_1v_1)v_1^T] \\
&= \frac{1}{n}XX^T - \lambda_1v_1v_1^T
\end{aligned}
$$

2.2. Principal eigenvectors of the deflated matrix

If v_j $(j \neq 1)$ is a principal eigenvector of C with corresponding eigenvalue λ_j, that is, $Cv_j = \lambda_jv_j$, we also have

$$
\begin{aligned}
\tilde{C}v_j &= (C - \lambda_1v_1v_1^T)v_j \\
&= Cv_j - \lambda_1v_1(v_1^Tv_j),\ \ j \neq 1 \\
&= Cv_j \\
&= \lambda_jv_j
\end{aligned}
$$

That means, v_j is also a principal eigenvector of $\tilde{C}$ with the same eigenvalue λ_j.

2.3. First principlal eigenvector of the deflated matrix

From last question, we know that $\{v_j\}_{j=2}^n$ are the $n-1$ principal eigenvectors of $\tilde{C}$ with corresponding eigenvalues $\{\lambda_j\}_{j=2}^n$. We also know that $rank(\tilde{C}) \leq min\{rank(\tilde{X}), rank(\tilde{X}^T)\} = n-1$, that means, $\tilde{C}$ has at most $n-1$ eigenvectors. Thus $\tilde{C}$ only has $n-1$ eigenvectors $\{v_j\}_{j=2}^n$, and its first principal eigenvector $u = v_2$, because $\lambda_2 \geq \lambda_j,\ j = 3..n$.

2.4. Pseudocode for Successive Deflation

Algorithm 1 : Find the first k principal basis vectors of X

Input: C, k, f
Output: $\{\lambda_j,\ v_j\},\ j = 1, ..., k$
$\tilde{C} \leftarrow C$
for $j \in \{1, 2, ..., k\}$ **do**
  $\{\lambda_j,\ v_j\} = f(\tilde{C})$
  $\tilde{C} \leftarrow \tilde{C} - \lambda_jv_jv_j^T$
end for

3. Question 3 – Clustering with K-means

3.1.

	$k=2$	$k=4$	$k=6$
$Iteration$	20	11	8
SS_{total}	5.36E8	4.61E8	4.31E8
p_1	79.82%	67.88%	55.18%
p_2	54.88%	86.95%	94.56%
p_3	67.35%	77.42%	74.87%

Table 1. **Results for different number of clusters.**.

3.2.

When $k = 6$, k-means converges at iteration 8.

3.3.

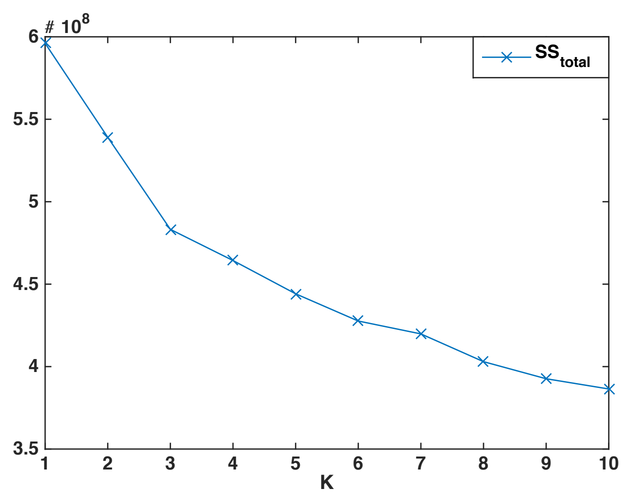

Figure 1. SS_{total} **versus** k**.**

3.4.

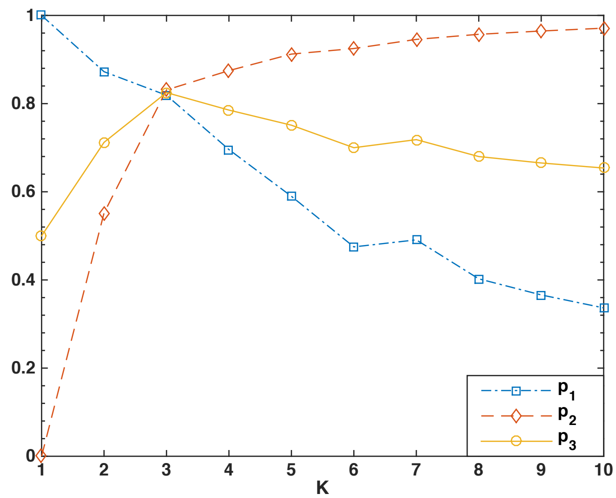

Figure 2. p_1, p_2, p_3 **versus** k**.**

4. Question 4 – Scene Classification

4.1.

Using SVM with RBF kernel, under default $C(=1)$ and $\gamma(=0.001)$, the 5-fold cross validation accuracy is 15.64%.

4.2.

Using SVM with RBF kernel, under tuned $C(=1000)$ and $\gamma(=10)$, the 5-fold cross validation accuracy is 75.13%.

4.3.

Using SVM with exponential χ^2 kernel, under tuned $C(=10)$ and $\gamma(=0.5)$, the 5-fold cross validation accuracy is 83.25%.

4.4.

I got 79.38% accuracy on test data using tuned parameters: $C(=10)$ and $\gamma(=0.5)$.

4.5.

My best test accuracy achieved on Kaggle is 81.75%.