CM146, Winter 2020
Problem Set 3: SVM and Kernels
Due Feb xx, 2020

1 Kernels

Solution:

(a) Let v be some vector that represents a document. We'll define that each word w_i in our dictionary has a unique index i. Let $\phi(v)$ be a function s.t. $\phi(v)_i = 1$ iff the word w_i appears in v, and otherwise $\phi(v)_i = 0$ ($\phi(v)$ is a vector in which there's 1 at indices of words that appear in v, and 0 everywhere else). By using this definition, we can define a $k(x, z) = \phi(x)^T\phi(z)$. This function successfully return the number of unique words that occur in both x and z, because the dot product sums the number of elements i that are 1 in both $\phi(x)$ and $\phi(z)$ (the number of unique words that appear in both documents).
It naturally follows that $k(x, z) = \phi(x)^T\phi(z) = \phi(z)^T\phi(x) = k(z, x)$. Thus, $k(x, z)$ is indeed a kernel function.

(b) $k(x, z) = x \cdot z$ is a kernel. Thus, $\frac{k(x,z)}{||x||\cdot||z||} = \frac{x\cdot z}{||x||\cdot||z||}$ is also a kernel by the scaling rule. The function $k_2(x, z) = 1$ is a kernel since $k_2(x, z) = k_2(z, x) = 1$ and we can define $k_2(x, z) = \phi(x)^T\phi(z)$ where $\phi(x) = \phi(z) = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, so this gives us our kernel $k_2(x, z) = 1$. By using the sum rule, we get that $k_3(x, z) = k_2(x, z) + k(x, z) = 1 + \frac{x\cdot z}{||x||\cdot||z||}$ is a kernel. Lastly, by using the product rule, we get that $k_3(x, z)k_3(x, z)k_3(x, z) = (1 + \frac{x\cdot z}{||x||\cdot||z||})^3 = (1 + (\frac{x}{||x||}) \cdot (\frac{z}{||z||}))^3$ is a kernel ■.

(c) $k_\beta = (1 + \beta x \cdot z)^3 = \sum_{i=0}^{3} \binom{3}{i} 1^i(\beta x \cdot z)^{3-i} =$
$(\beta x \cdot z)^3 + 3(\beta x \cdot z)^2 + 3(\beta x \cdot z) + 1 =$
$\beta^3 \cdot (x_1z_1 + x_2z_2)^3 + 3\beta^2 \cdot (x_1z_1 + x_2z_2)^2 + 3\beta \cdot (x_1z_1 + x_2z_2) + 1 =$
$\beta^3(x_1^3z_1^3 + 3x_1z_1x_2^2z_2^2 + 3x_1^2z_1^2x_2z_2 + x_2^3z_2^3) + 3\beta^2(x_1^2z_1^2 + 2x_1z_1x_2z_2 +$

$x_2^2 z_2^2) + 3\beta(x_1 z_1 + x_2 z_2) + 1 = \phi_\beta(x)^T \phi_\beta(z)$, where:

$$\phi_\beta(x) = \begin{pmatrix} \sqrt{\beta^3} x_1^3 \\ \sqrt{3\beta^3} x_1^2 x_2 \\ \sqrt{3\beta^3} x_1 x_2^2 \\ \sqrt{\beta^3} x_2^3 \\ \sqrt{3\beta^2} x_1^2 \\ \sqrt{6\beta^2} x_1 x_2 \\ \sqrt{3\beta^2} x_2^2 \\ \sqrt{3\beta} x_1 \\ \sqrt{3\beta} x_2 \\ 1 \end{pmatrix}$$

We can see that the difference between $k_\beta(x, z)$ and $k(x, z)$ is that a scaling factor is applied on each feature, which is a function of β. Essentially, β is a hyperparameter that allows us to play with the scaling of the features, so as to give more weight to the cubed, squared, or linear features. For example, for $\beta >> 1$ the cubed features will be dominant (because $\beta^3 > \beta, \beta^2$ for $\beta > 1$), whereas for $\beta << 1$ the linear features will be dominant (because $\beta^3, \beta^2 < \beta$ for $\beta < 1$).

2 SVM

Solution:

(a) We can solve the constrained minimization problem using Lagrange multipliers. Since the constraint is the inequality $1 - y\theta^T x \leq 0$, we will use the Lagrange Duality formulation (this is an SVM so it will give us the optimal solution):

$$\lambda^* = \underset{\lambda}{\text{argmax}} \min_\theta \mathcal{L}(\theta, \lambda) = \underset{\lambda}{\text{argmax}} \min_\theta \frac{1}{2}||\theta||^2 + \lambda(1 - y\theta^T x)$$

$$\frac{\partial \mathcal{L}(\theta, \lambda)}{\partial \theta} = \theta - \lambda y x = 0 \Rightarrow \theta = \lambda y x$$

$$\Downarrow$$

$$\lambda^* = \underset{\lambda}{\text{argmax}} \frac{1}{2}\lambda^2 ||x||^2 + \lambda(1 - \lambda||x||^2) = \frac{1}{||x||^2}$$

Finally, we get: $\boxed{\theta^* = \lambda^* y x = \frac{1}{||x||^2} y x = -\frac{1}{a^2 + e^2} \begin{pmatrix} a \\ e \end{pmatrix}}$ ■

(b) Again, we can solve this by using Lagrage Duality. Lets start by solving for the binding constrain:

$$\lambda^*_{1,2} = \underset{\lambda_{1,2}}{\text{argmax}} \min_{\theta} \frac{1}{2}||\theta||^2 + \lambda_1(1 - y_1\theta^T x_1) + \lambda_2(1 - y_2\theta^T x_2)$$

Lets define: $\Lambda = \begin{pmatrix}\lambda_1\\ \lambda_2\end{pmatrix}, X = \begin{pmatrix}y_1 x_1^T\\ y_2 x_2^T\end{pmatrix}, V = \begin{pmatrix}1\\ 1\end{pmatrix}$ Thus, we get:

$$\Lambda^* = \underset{\Lambda}{\text{argmax}} \min_{\theta} \frac{1}{2}||\theta||^2 + \Lambda^T(V - X\theta)$$

$$\Downarrow$$

$$\frac{\partial \mathcal{L}(\theta, \lambda)}{\partial \theta} = \theta - \lambda_1 y_1 x_1 - \lambda_2 y_2 x_2 = 0 \Rightarrow \theta^* = \lambda_1 y_1 x_1 + \lambda_2 y_2 x_2 = X^T\Lambda$$

$$\Downarrow$$

$$\Lambda^* = \underset{\Lambda}{\text{argmax}} \frac{1}{2}||X^T\Lambda||^2 + \Lambda^T(V - XX^T\Lambda) = \underset{\Lambda}{\text{argmax}} \frac{1}{2}\Lambda^T XX^T\Lambda + \Lambda^T V - \Lambda^T XX^T\Lambda$$

$$= \underset{\Lambda}{\text{argmax}}\, \Lambda^T V - \frac{1}{2}\Lambda^T XX^T\Lambda$$

Lets derive w.r.t Λ (which is a vector) to find the extramum:

$$\Downarrow$$

$$V - XX^T\Lambda = 0 \Rightarrow \Lambda = (XX^T)^{-1}V = \begin{pmatrix} y_1^2 x_1^T x_1 & y_1 y_2 x_1^T x_2 \\ y_1 y_2 x_1^T x_2 & y_2^2 x_2^T x_2 \end{pmatrix}^{-1} V = \begin{pmatrix} 2 & -1 \\ -1 & 1 \end{pmatrix}^{-1} V$$

$$= \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} V = \begin{pmatrix} 2 \\ 3 \end{pmatrix} \Rightarrow \boxed{\lambda_1^* = 2, \lambda_2^* = 3}$$

We can now substitute λ_1^* and λ_2^* into θ^*:

$$\boxed{\theta^* = \lambda_1 y_1 x_1 + \lambda_2 y_2 x_2 = 2x_1 - 3x_2 = \begin{pmatrix} -1 \\ 2 \end{pmatrix}}$$

Thus, the margin is (it's the distance from the line to either of the points - the distance to either of the points is the same because they are both the closest points to the line):

$$\boxed{\gamma = |\frac{\theta^{*T} x_1}{||\theta^*||}| = |\frac{-1}{\sqrt{5}}| = \frac{1}{\sqrt{5}}}$$

(c) When we allow offset, it's obvious that the solution would be a horizontal line separating the points with the maximum margin possible (because both points have $x_{1,1} = x_{2,1} = 1$). This line would be with θ^* of the form $\theta^* = \begin{pmatrix} 0 \\ c \end{pmatrix}$, and b^* which is the bias. Lets find c and b^*:

$$y_1(\theta^T x_1 + b) = c + b \geq 1, y_2(\theta^T x_2 + b) = -b \geq 1$$

$$\Downarrow$$

$$c \geq 1 - b, -b \geq 1$$

We want to choose b to be as big as possible, so as to minimize c (and thus minimize θ). The largest b that we can choose is $b = -1$ because $b \leq -1$, so we get $c \geq 2 \Rightarrow c = 2$. Thus:

$$\boxed{\theta^* = \begin{pmatrix} 0 \\ 2 \end{pmatrix}, b^* = -1}$$

In this case, the line is exactly in the middle between the two points, so the margin is $\gamma = \frac{1}{2}$, which is a bigger (and better) margin than before (the margin before was $\gamma = \frac{1}{\sqrt{5}}$).

3 Twitter analysis using SVMs

3.1 Feature Extraction

Solution: Nothing to submit. This is a coding section.

3.2 Hyperparameter Selection for a Linear-Kernel SVM

Solution:

(a) Coding section.

(b) It might be beneficial to maintain class proportions across folds because otherwise the SVM might do more mistakes on classifying the class which appears less in training. During training, lets say that in some fold there are many more examples of class 1 than class 0. Then, it might be the case that the examples of class 0 are far away from the examples of class 1, and so the SVM would maximize the margin. But, it's possible that many other examples (test examples) of class 0

are close to the examples of class 1, and so the SVM will make many mistakes on such examples. In other words, when we maintain class proportions, there's a higher chance to have better support vectors to train on.

(c) Coding section.

(d) These are the results:

C	accuracy	F1-score	AUROC	precision	sensitivity	specificity
10^{-3}	0.7089	0.8297	0.5	0.7089	1	0.0000
10^{-2}	0.7107	0.8306	0.5031	0.7102	1	0.0063
10^{-1}	0.8060	0.8755	0.7188	0.8357	0.9294	0.5081
10^{0}	0.8146	0.8749	0.7531	0.8562	0.9017	0.6045
10^{1}	0.8182	0.8766	0.7592	0.8595	0.9017	0.6167
10^{2}	0.8182	0.8766	0.7592	0.8595	0.9017	0.6167
best C	10^{1}	10^{1}	10^{1}	10^{1}	10^{-3}	10^{1}

We can see that the metrics accuracy, F1-score, AUROC, precision, and specificity are all consistent with the best C. All of these performance metrics increased as we increased C (except when we increased C from 10^1 to 10^2, in which case the best performance remained the same). The sensitivity performance metric is flipped (smaller C is better).
In general, a larger C in SVM means more chance of overfitting, since larger C causes a larger penalty on false classifications. Thus, amongst the options of C which all resulted in the best performance result, I chose the smaller value of C so as to generalize better.

3.3 Hyperparameter Selection for an RBF-Kernel SVM

Solution:

(a) The γ parameter is the inverse of the standard deviation of the RBF kernel. A small γ value defines a Gaussian function with large variance, in which case two points would be considered similar even if they are far from each other. On the other hand, a large γ defines a Gaussian function with small variance, in which case two points would be considered similar if they are close to each other. If γ is too large, the model could overfit the data since two points would be considered

similar only if they are very close. If γ is too small, the model could underfit the data since two points would almost always be considered similar. Thus, both cases would result in high generalization error.

(b) The grid I used was exponentially growing C and γ. Explicitly, C ranged from 2^{-5} to 2^{7} and γ ranged from 2^{-13} to 2^{-4} (both with step size of multiply of 10). I chose this grid so as to follow the guidelines from the following paper:
https://www.csie.ntu.edu.tw/ cjlin/papers/guide/guide.pdf.
"We recommend a "grid-search" on C and γ using cross-validation. Various pairs of (C,γ) values are tried and the one with the best cross-validation accuracy is picked. **We found that trying exponentially growing sequences of C and γ is a practical method to identify good parameters**."
After exploring the exponential grid, I found that $C = 64, \gamma = 0.0078$ is a good setup for the hyperparameters (except for the sensitivity metric). I then continued to explore a linear grid that ranged from $C = 60$ to $C = 70$, and $\gamma = 0.001$ to $\gamma = 0.009$. Since the sensitivity metric resulted in performance=1 for $C = 0.0625$ and $\gamma = 0.001$, I didn't explore further for different hyperparameters for the sensitivity metric.

(c) I receieved the following results:

metric	score	C	γ
accuracy	0.8200	60.0	0.006
F1-score	0.8779	60.0	0.005
AUROC	0.7605	60.0	0.006
precision	0.8618	60.0	0.006
sensitivity	1.0000	0.0625	0.001
specificity	0.6169	60	0.006

Larger C and smaller γ usually result in better CV performance (except for the sensitivity performance metric, which tends to favor smaller C).

3.4 Test Set Performance

Solution:

(a) For all metrics besides sensitivity: for linear-kernel SVM I chose $C = 10$, and for RBF-kernel I chose $C = 60, \gamma = 0.006$. For the sensitivity metric: for linear-kernel SVM I chose $C = 10^{-3}$, and for RBF-kernel I chose $C = 0.0625, \gamma = 0.001$. I received the following results:

metric	RBF-Kernel SVM test performance	linear-kernel SVM test performance
accuracy	0.7571	0.7429
F1-score	0.4516	0.4375
AUROC	0.6361	0.6259
precision	0.7000	0.6364
sensitivity	1.0000	1.0000
specificity	0.9388	0.9184

We can see that the RBF-kernel SVM almost always perform better than linear-kernel SVM.