COS 584

Spring 2021

P2: Classification

A few clarifications

- 584 precept participation points (10%):
  - Primarily depends on answering pre-lecture questions and attending precepts
    - Don’t have to get the question ‘right’ - they are usually open-ended, we just expect reasonable responses to show you have thought about the problem
  - Comments on Perusall also help your score (so please annotate away)
    - They also help your friends get more out of the readings!
  - If you do the above reasonably well, you can expect to get full participation points

Baselines and Bigrams: Simple, Good Sentiment and Topic Classification

Sida Wang and **Christopher D. Manning**
Department of Computer Science
Stanford University
Stanford, CA 94305
{sidaw,manning}@stanford.edu

Key takeaways

- Naive Bayes classifiers are not as bad as people thought at the time
  - Perform quite well for classification of short snippets (compared to even more ‘complex’ models)
  - However, SVMs do maintain their advantage on longer texts
- Adding bigram features helps classification quite a bit
- A novel interpolation of SVM and NB using log-count ratios as feature values performs well on all tasks

Formulation

- Model template:

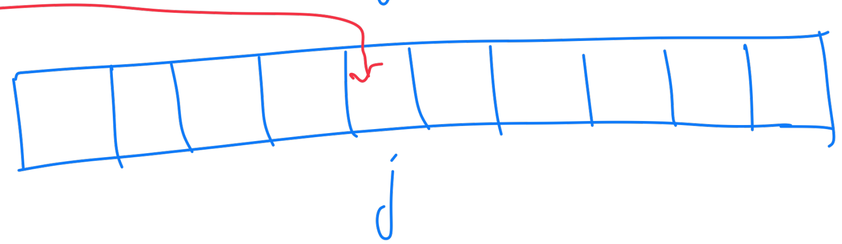
$$y^{(k)} = \text{sign}\left(w^T x^{(k)} + b\right)$$

$f^{(i)} \in R^{|V|}$ $\longrightarrow$ Feature count vector

$y^{(i)} \in \{-1, 1\}$ $\longrightarrow$ Label

V $\longrightarrow$ set of features

$f_j^{(i)}$ $\longrightarrow$ count of j^{th} feature V_j



$$p = \alpha + \sum_{i:\, y^{(i)}=1} f^{(i)}$$

$$q = \alpha + \sum_{i:\, y^{(i)}=-1} f^{(i)}$$

$\bar{p} = \frac{p}{\|p\|_1}$ $\equiv$ Normalized feature counts

$\bar{q} = q/\|q\|_1$

$$r = \log\left(\frac{\bar{p}}{\bar{q}}\right)$$

Naive Bayes

① $x^{(k)} = f^{(k)}$

② $w = \log\left(\frac{\bar{p}}{\bar{q}}\right)$

③ $b = \log\left(N^+ / N^-\right)$

$$y^{(k)} = \text{sign}\left(w^T x^{(k)} + b\right)$$

$$\Rightarrow y^{(k)} = \begin{cases} +1 & \text{if } wx+b > 0 \\ -1 & \text{if } wx+b < 0 \end{cases}$$

$$\Rightarrow y = \begin{cases} +1 & \text{if } e^{wx+b} > 1 \\ -1 & \text{if } e^{wx+b} < 1 \end{cases}$$

$$e^{wx+b} = e^{f \cdot \log\left(\frac{\bar{p}}{\bar{q}}\right) + \log\left(\frac{N^+}{N^-}\right)}$$

$$= \frac{\prod_j \bar{p}_j^{f_j} \cdot N^+}{\prod_j \bar{q}_j^{f_j} \cdot N^-} \approx \frac{P(x|+1)\,P(+1)}{P(x|-1)\,P(-1)}$$

$$\Rightarrow y = \begin{cases} +1 & \text{if } P(x|+1)P(+1) > P(x|-1)P(-1) \\ -1 & \text{otherwise} \end{cases}$$

Support Vector Machine (SVM)

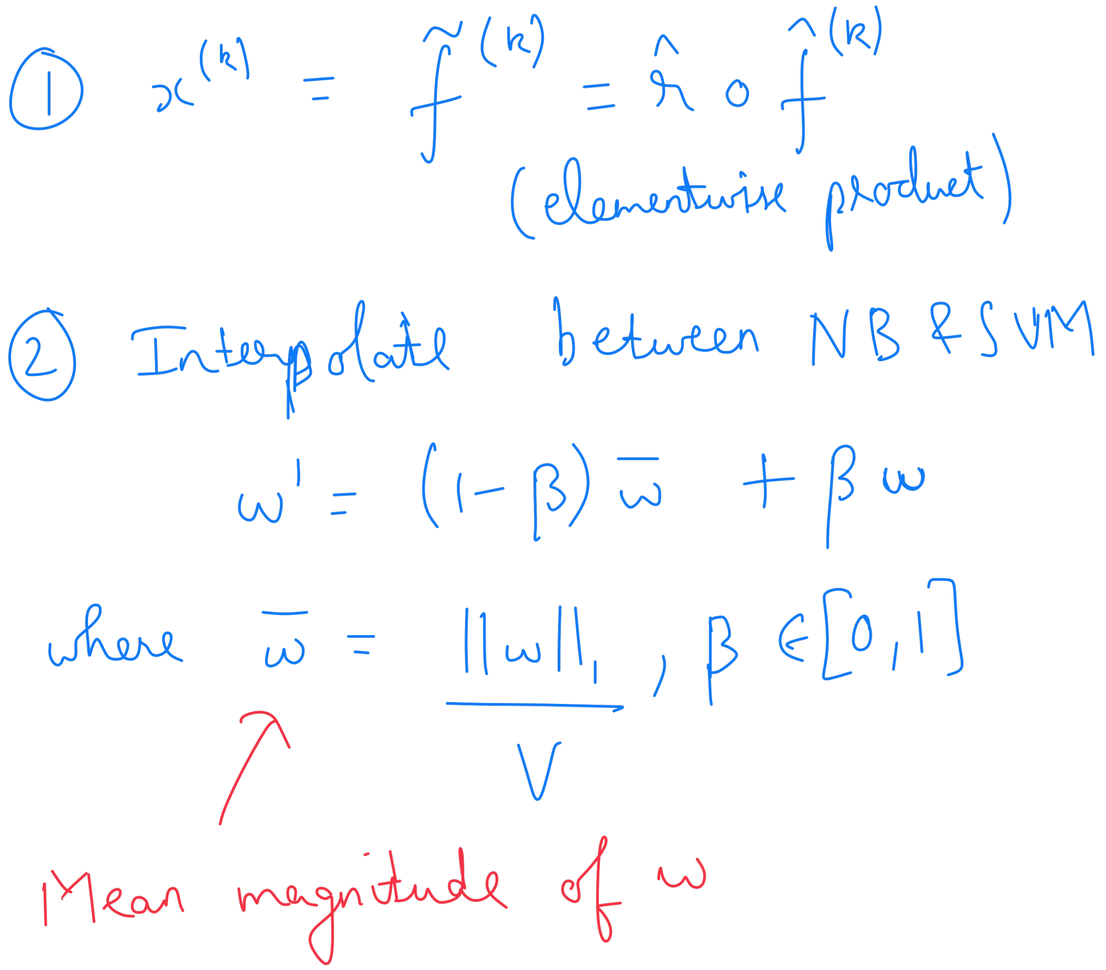
① $x^{(k)} = \hat{f}^{(k)} = 1\{f^{(k)} > 0\}$ (binarized features)

② Obtain w & b by:

$$\min_{w,b} \; w^T w + C \sum_i \max\left(0, 1 - y^{(i)} \underbrace{\left(w^T \hat{f}^{(i)} + b\right)}_{\text{prediction}}\right)^2$$

- $w^T w$: L2 regularization
- C: reg. coeff.
- $y^{(i)}$: True class ($+1$ & -1)
- Penalize misclassifications

NB SVM



- SVM with NB features
- If all weights are of similar magnitude, SVM is not very confident
  - Trust weighted ($\bar{w}$, scalar) NB model more to make predictions
- If only some ws are high while others are not, βw term may dominate -> trust SVM more

Result 1: MNB is better at snippets

Method	RT-s	MPQA	CR	Subj.
MNB-uni	77.9	85.3	79.8	**92.6**
MNB-bi	**79.0**	**86.3**	80.0	**<u>93.6</u>**
SVM-uni	76.2	86.1	79.0	90.8
SVM-bi	77.7	**<u>86.7</u>**	80.8	91.7
NBSVM-uni	**78.1**	85.3	80.5	92.4
NBSVM-bi	**<u>79.4</u>**	**86.3**	**<u>81.8</u>**	**93.2**
RAE	76.8	85.7	–	–
RAE-pretrain	77.7	**86.4**	–	–
Voting-w/Rev.	63.1	81.7	74.2	–
Rule	62.9	81.8	74.3	–
BoF-noDic.	75.7	81.8	79.3	–
BoF-w/Rev.	76.4	84.1	**81.4**	–
Tree-CRF	77.3	86.1	**81.4**	–
BoWSVM	–	–	–	90.0

Table 2: Results for snippets datasets. Tree-CRF: (Nakagawa et al., 2010) RAE: Recursive Autoencoders (Socher et al., 2011). RAE-pretrain: train on Wikipedia (Collobert and Weston, 2008). "Voting" and "Rule": use a sentiment lexicon and hard-coded reversal rules. "w/Rev": "the polarities of phrases which have odd numbers of reversal phrases in their ancestors". The top 3 methods are in **bold** and the best is also **<u>underlined</u>**.

- Outperforms several rule-based systems and more "complex" models that use syntax, etc.
- SVM ends up being a weak baseline (despite being used frequently as one!)
- MNB is better even on large training datasets (RT-s, MPQA, Subj)
  - Not just limited to small datasets

Result 2: SVM better at longer texts

Our results	RT-2k	IMDB	Subj.
MNB-uni	83.45	83.55	**92.58**
MNB-bi	85.85	86.59	**93.56**
SVM-uni	86.25	86.95	90.84
SVM-bi	87.40	**89.16**	91.74
NBSVM-uni	87.80	88.29	92.40
NBSVM-bi	**89.45**	**91.22**	**93.18**
BoW (bnc)	85.45	87.8	87.77
BoW (bΔt′c)	85.8	88.23	85.65
LDA	66.7	67.42	66.65
Full+BoW	87.85	88.33	88.45
Full+Unlab'd+BoW	**88.9**	88.89	88.13
BoWSVM	87.15	–	90.00
Valence Shifter	86.2	–	–
tf.Δidf	88.1	–	–
Appr. Taxonomy	**90.20**	–	–
WRRBM	–	87.42	–
WRRBM + BoW(bnc)	–	**89.23**	–

- Poor independence assumptions of NB don't hold for longer texts
- SVMs still worse than other methods
- Bigrams seem to help consistently

Result 3: NB-SVM is jack of all trades

Method	RT-s	MPQA	CR	Subj.
MNB-uni	77.9	85.3	79.8	**92.6**
MNB-bi	**79.0**	**86.3**	80.0	**93.6**
SVM-uni	76.2	86.1	79.0	90.8
SVM-bi	77.7	**86.7**	80.8	91.7
NBSVM-uni	**78.1**	85.3	80.5	92.4
NBSVM-bi	**79.4**	**86.3**	**81.8**	**93.2**

Our results	RT-2k	IMDB	Subj.
MNB-uni	83.45	83.55	**92.58**
MNB-bi	85.85	86.59	**93.56**
SVM-uni	86.25	86.95	90.84
SVM-bi	87.40	**89.16**	91.74
NBSVM-uni	87.80	88.29	92.40
NBSVM-bi	**89.45**	**91.22**	**93.18**

Method	AthR		XGraph		BbCrypt	
MNB-uni	85.0		90.0		**99.3**	
MNB-bi	**85.1**	+0.1	**91.2**	+1.2	99.4	+0.1
SVM-uni	82.6		85.1		98.3	
SVM-bi	83.7	+1.1	86.2	+0.9	97.7	−0.5
NBSVM-uni	**87.9**		**91.2**		99.7	
NBSVM-bi	**87.7**	−0.2	**90.7**	−0.5	99.5	−0.2

- Consistently one of the top performers
- Value of β seems to be important
  - Don't want to be too aggressive
  - $\beta \in [0.25, 0.5]$ seems to work best

Discussion questions

- Q1: Wang and Manning (2012) find that Multinomial Naive Bayes (MNB) works quite well for classifying short snippets but doesn't do well on longer text (e.g. full-length reviews). Can you venture an explanation for why this might happen? Can you think of any modifications that might help MNB handle longer texts?

- Q2: Using bigram features (instead of unigrams) in the model helps to add in some positional information on the words that may help the model classify better (e.g. differentiate between 'very good' and 'not good'). Can you think of other ways to add in more positional information to the model?