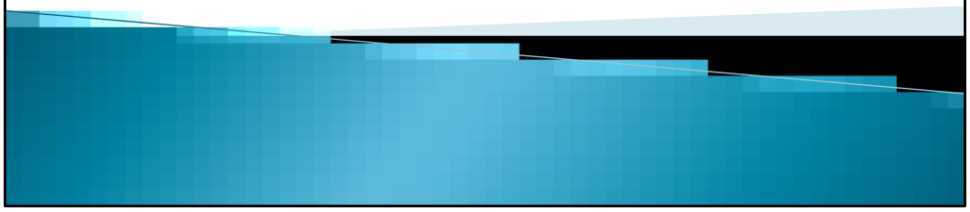
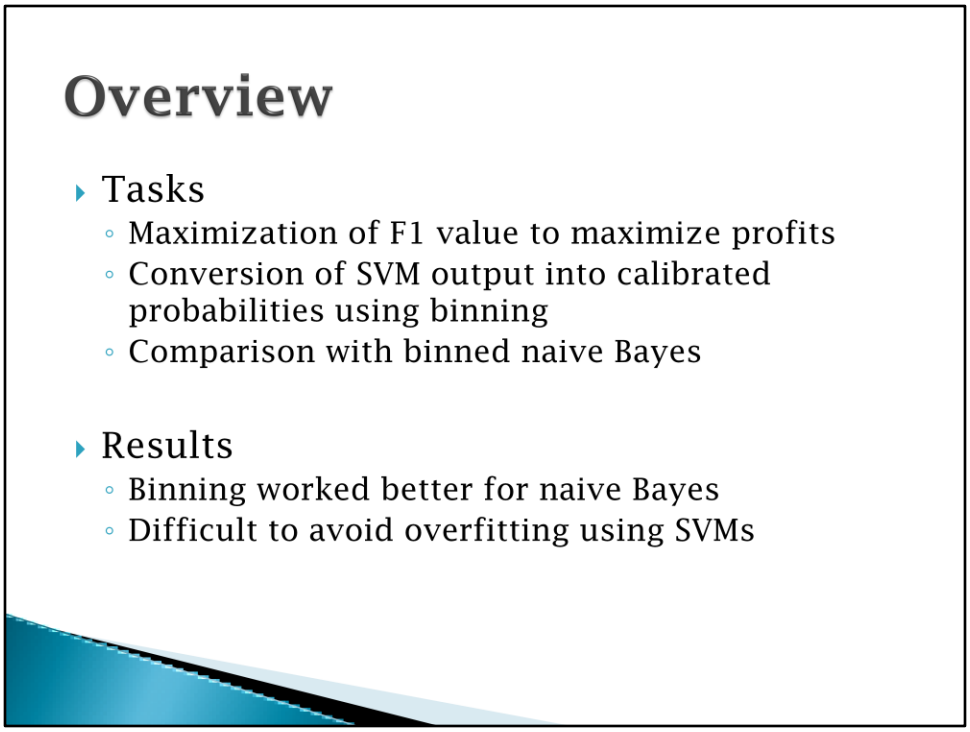
Obtaining Calibrated Probability Estimates from SVMs

Joseph Drish

Presented by:
Amit Belani

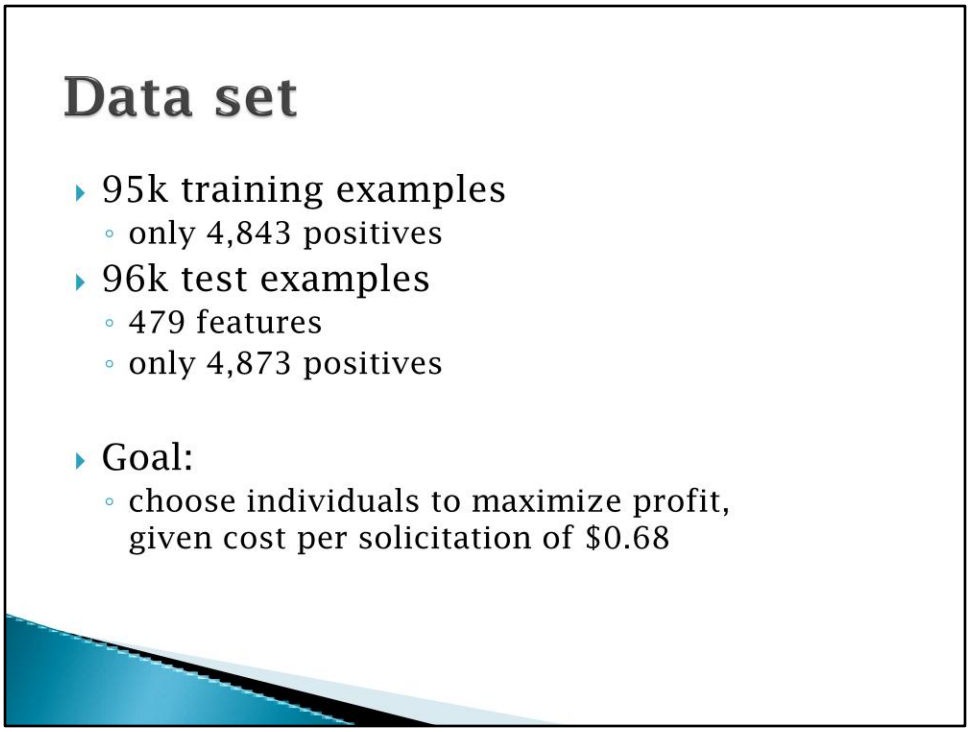




•In this presentation, we first see how the F1 value metric is computed and optimized in order to maximize profit.
•The LIBSVM library, which is a popular SVM library developed at National Taiwan University, is used.

•In 2001, Zadrozny, B. and Elkan, C. published a paper that discussed converting the outputs of naive Bayes and decision tree classifiers into well-calibrated probabilities using binning.
•This paper is a replication of the work using SVMs instead.

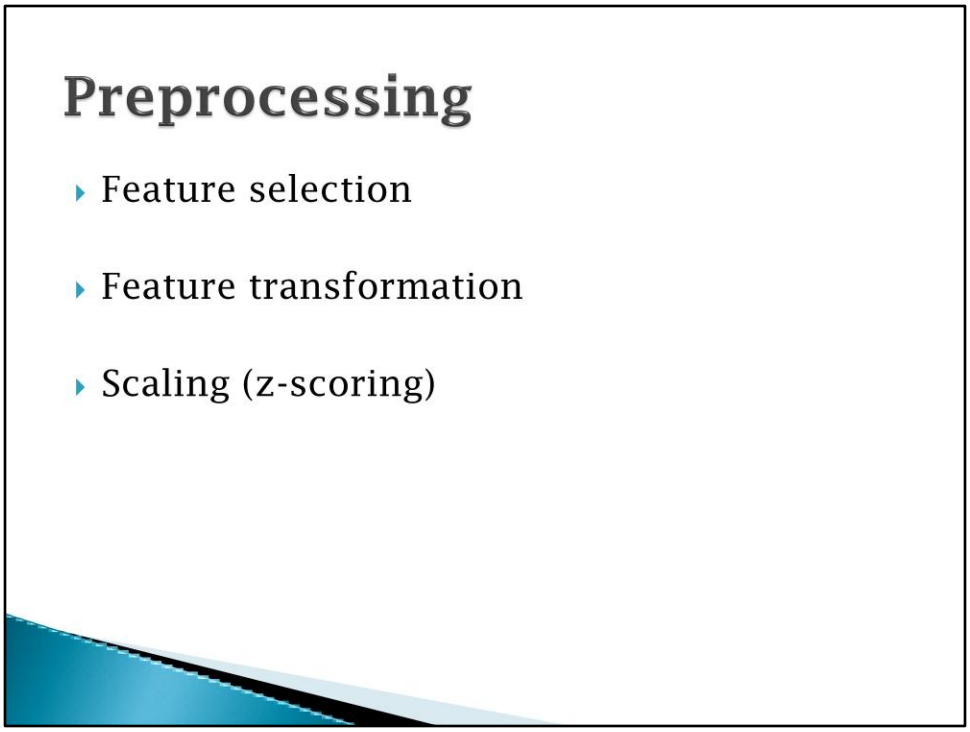
•The predicted probabilities are evaluated using four metrics, and surprisingly, binning worked better for naive Bayes than for SVMs.



•The data set from the KDD'98 data mining competition is used.
•This is a fundraiser dataset where the objective to identify who will respond to the fundraising mailings and make a donation.

•The goal therefore is to selectively solicit only the individuals that will allow profits to be maximized, given that there is a cost per solicitation.

•As we see, the data set is highly unbalanced, with only about 5% positives.
•The LIBSVM package has a training option for unbalanced data sets, which is why it was used.



•Out of all the available features, only the most predictive ones were used.
•These were also the same features used in the previous experiment with naive Bayes and decision trees.
•This allows for a comparison of results across learning methods.

•Features with non-numerical values were mapped to numbers.

•The data was scaled by subtracting the feature mean, and dividing by the feature standard deviation.
•This z-scoring of the data prevents outliers from having too much influence.

Primal SVM problem

- Given a set of training vectors $x_i \in R^n$, $i = 1, \ldots l$, and a vector $y \in R^l$ such that $y_i \in \{1,-1\}$, the primal SVM problem is:

$$\min_{w,b,\xi} \quad \frac{1}{2} w^T w + C_+ \sum_{y_i=1} \xi_i + C_- \sum_{y_i=-1} \xi_i$$

$$y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i,$$

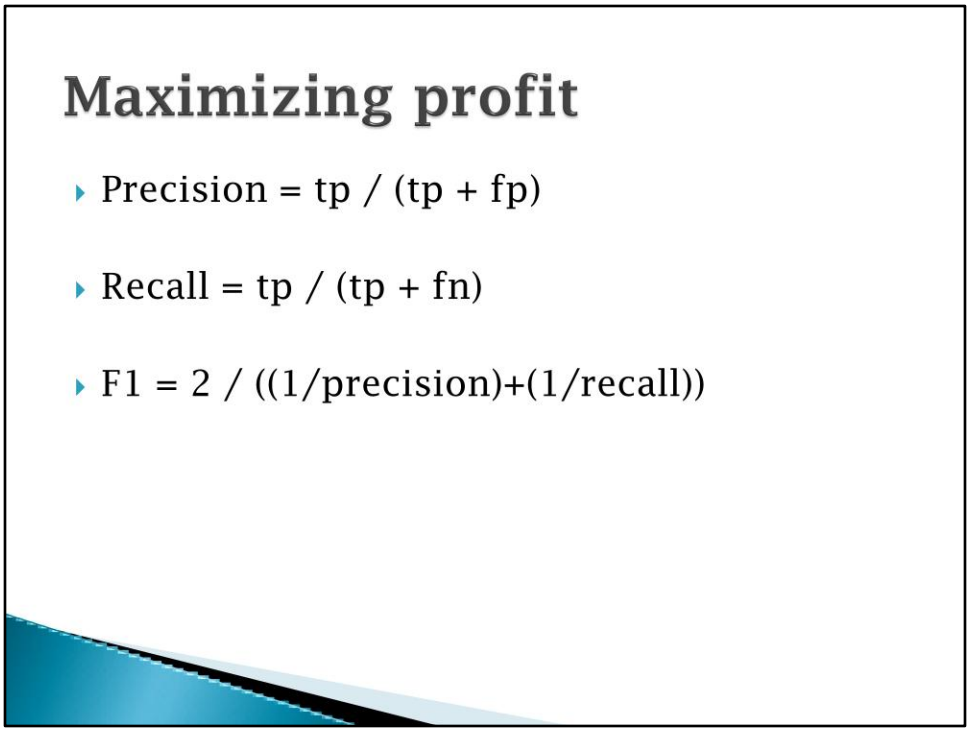
$$\xi_i \geq 0, i = 1, \ldots, l.$$

•As I mentioned, this data set is unbalanced, and the LIBSVM package provides an option for learning classifiers on unbalanced data sets.
•Two penalty parameters, C_+ and C_-, are used to tradeoff generalization ability and misclassification error for positive and negative training examples, respectively.

•Their ratio determines how the training examples are penalized.
•As the ratio of C_+ to C_- increases, the rate at which the SVM classifier predicts $j = 1$ for a given example x increases.
•As the ratio of C_+ to C_- decreases, the rate at which the SVM classifier predicts $j = -1$ increases.

•With C_- fixed at 1, C_+ can be changed to find the best ratio of C_+ to C_-.
•Thus, as C_+ is increased, the test examples classified as positive are increased, which means the number of solicitations increases.
•A higher C_+ also means that more non-donors can be incorrectly classified as being donors.

•A radial basis kernel is used.



•Because the goal is to maximize profits on the test set, this translates to correctly predicting as many donors as possible, and also not soliciting too many non-donors.
•This means that the metrics of precision and recall should be balanced, and using the F1 value is a way to do this.

•The F1 value is between 0 and 1, and higher values are better.
•Note that there is no clear link between optimizing F1 and obtaining good probabilities from the SVM.

Effect of varying C_+

C_+	classification	solicitations	tp	fp	fn	F1
–	0.9479	35	2	33	488	0.0076
5	0.8891	709	45	664	445	0.0750
7.5	0.8704	924	59	865	431	0.0835
8	0.8670	968	64	904	426	0.0878
9	0.8551	1095	68	1027	422	0.0858
10	0.8440	1214	72	1142	418	0.0845
12.5	0.8231	1437	79	1358	411	0.0820
15	0.8086	1594	85	1509	405	0.0816
20	0.7758	1946	97	1849	393	0.0796

- As seen in the table, as C_+ is increased, F1 increases as well.
- This happens until F1 reaches its peak when C_+ equals 8, and then F1 decreases.

Finding probabilities

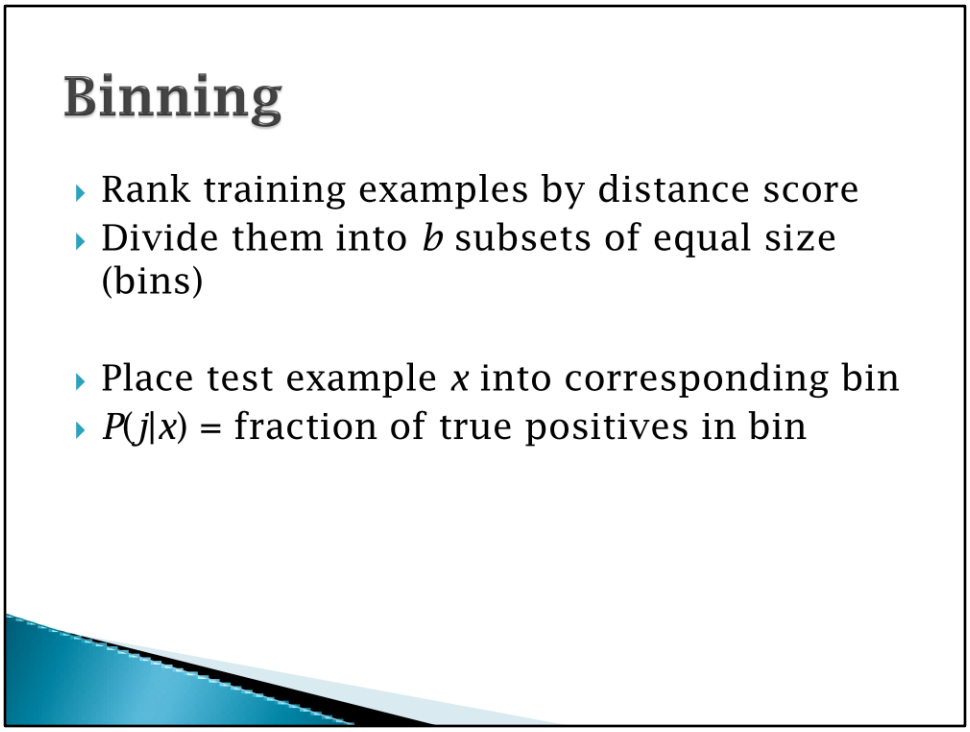
Given test example x:

knowing:
- distance of x from separating hyperplane
- class j to which x belongs

determine:
- class conditional posterior probability $P(j|x)$

•We now reach the part where the outputs of the classifier are converted into well calibrated posterior probabilities.

•Given a test example x, knowing its distance from the separating hyperplane, and the class to which it belongs, the problem of interest is to know the probability with which x belongs to class j.



•Binning is a simple histogram technique.

•Training examples are first ranked according to their distance scores.
•They are then divided into subsets of equal size – these are called bins.
•Each bin therefore has an upper bound and a lower bound distance score value.

•The number of subsets *b* is chosen experimentally so that the variance in the resulting probability estimates is reduced.

•Given a test example *x*, it is placed in the bin according to the distance score predicted for it by the SVM.
•The corresponding estimated probability that *x* belongs to its predicted class *j* is the fraction of true positives in the bin, i.e. the fraction of training examples in the bin that actually belong to the class that has been predicted for *x*.

Probability estimates from binning

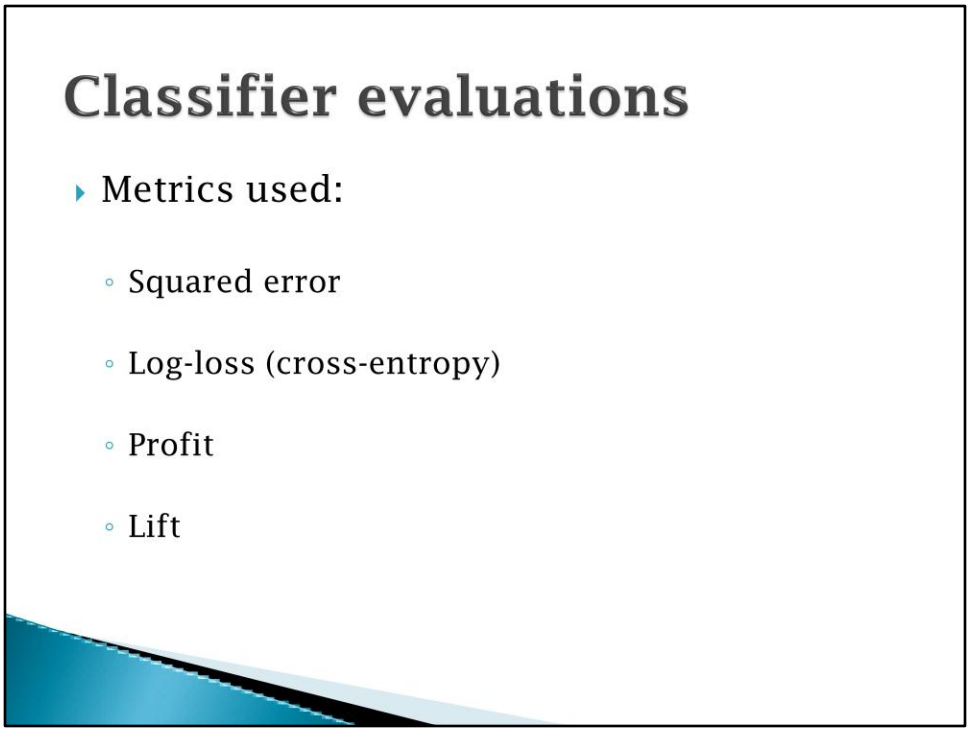
bin	ALL(1)	ALL(8)	SPLIT(1)	SPLIT(8)	SPLIT(15)	SPLIT(20)
1	0.1523	0.2080	0.0625	0.0604	0.0583	0.0534
2	0.0456	0.1157	0.0486	0.0688	0.0625	0.0548
3	0.0428	0.0770	0.0496	0.0520	0.0562	0.0615
4	0.0487	0.0238	0.0451	0.0454	0.0538	0.0569
5	0.0476	0.0169	0.0475	0.0363	0.0507	0.0489
6	0.0593	0.0308	0.0482	0.0437	0.0384	0.0493
7	0.0662	0.0258	0.0503	0.0503	0.0454	0.0402
8	0.0362	0.0053	0.0489	0.0552	0.0517	0.0531
9	0.0021	0.0022	0.0528	0.0461	0.0465	0.0479
10	0.0068	0.0020	0.0542	0.0493	0.0440	0.0416

•The number of bins was set to 10 in order to match the previous experiment with naive Bayes and decision trees.

•Using all the training examples from the training set can result in overfitting.
•To counter this, 70% of the training examples are used to learn the SVM classifier, and the remaining 30% are used for the binning process.
•This method is described further in the 2001 paper by Zadrozny, B. and Elkan, C., titled "Learning and making decisions when costs and probabilities are both unknown."

•This table shows the resulting binned SVM class-conditional probability estimates.
•The columns indicated as ALL refer to those classifiers that use all of the training set for both learning the classifier and binning.
•The columns indicated as SPLIT refer to those where the data was split 70-30.
•The number in the parenthesis in the top row is the value of C_+ that was used.

•From the table, we see that for the ALL classifiers the probability estimates decrease quickly, whereas the estimates for the SPLIT classifiers decrease slowly.
•Let us recall that the actual number of positive class training cases is about five percent, or that there is a 0.05 probability that a training case is observed to be of positive class.
•We see values indicative of severe overfitting with both the ALL classifiers.



•Four metrics were used to evaluate the binned SVM probability estimates.
•These are squared error, log-loss or cross-entropy, lift charts, and profit.

•Profit was calculated by multiplying the probability of donating for person x, by the expected donation amount for that person.
•Calculation of the expected donation amounts is discussed in the 2001 paper by Zadrozny, B. and Elkan, C., titled "Learning and making decisions when costs and probabilities are both unknown."

•These were the same four that were used as in the 2001 paper on binning with naive Bayes and decisions trees.
•This allows for a comparison of results.

Probability evaluation results

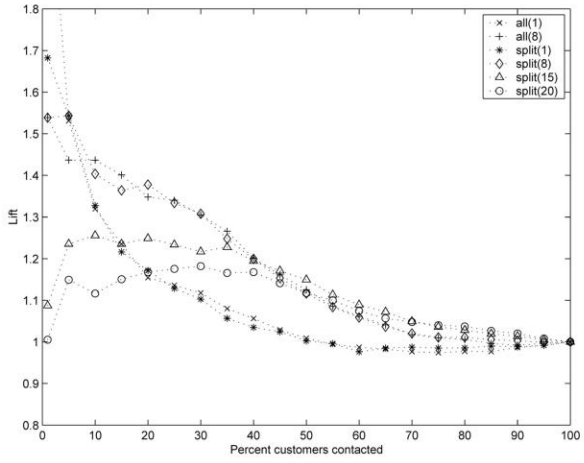
classifier	Training set MSE	MLL	Profit	Test set MSE	MLL	Profit
ALL(1)	0.09325	0.26715	$25699	0.10314	0.33692	$7919
ALL(8)	0.08850	0.23948	$38040	0.10215	0.34670	$8026
SPLIT(1)	0.09596	0.28685	$13699	0.09600	0.28862	$12512
SPLIT(8)	0.09534	0.28220	$17550	0.09586	0.28763	$12969
SPLIT(15)	0.09551	0.28322	$16748	0.09592	0.28807	$12817
SPLIT(20)	0.09593	0.28604	$14700	0.09594	0.28823	$12483
NAIVE BAYES	0.10089	0.30198	$10083	0.10111	0.30400	$9531
BINNED NAIVE BAYES	0.09546	0.28336	$14897	0.09528	0.28365	$14642
ALL BASE RATE	0.09636	0.28961	$11923	0.09602	0.28880	$12252
ALL ONE	1.89848	∞	$10790	1.89886	∞	$10560

•In this table, the mean squared error i.e. MSE, mean log-loss i.e. MLL, and profit are reported.

•Again, we see that both the SVM ALL classifiers severely overfit the data.
•The SVM SPLIT classifiers overfit the training data also, but not as badly.

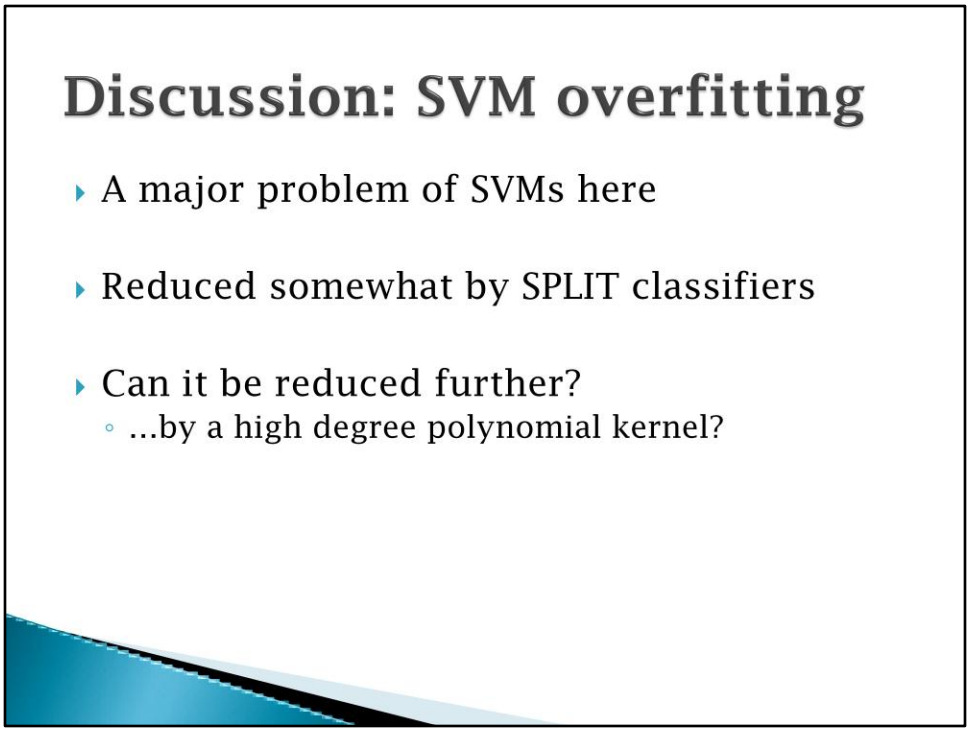
•For the binned SVM classifiers, we see that the SPLIT(8) has the best scores.
•Overall, binned naive Bayes does the best, followed by SPLIT(8).

SVM lift charts for the test set

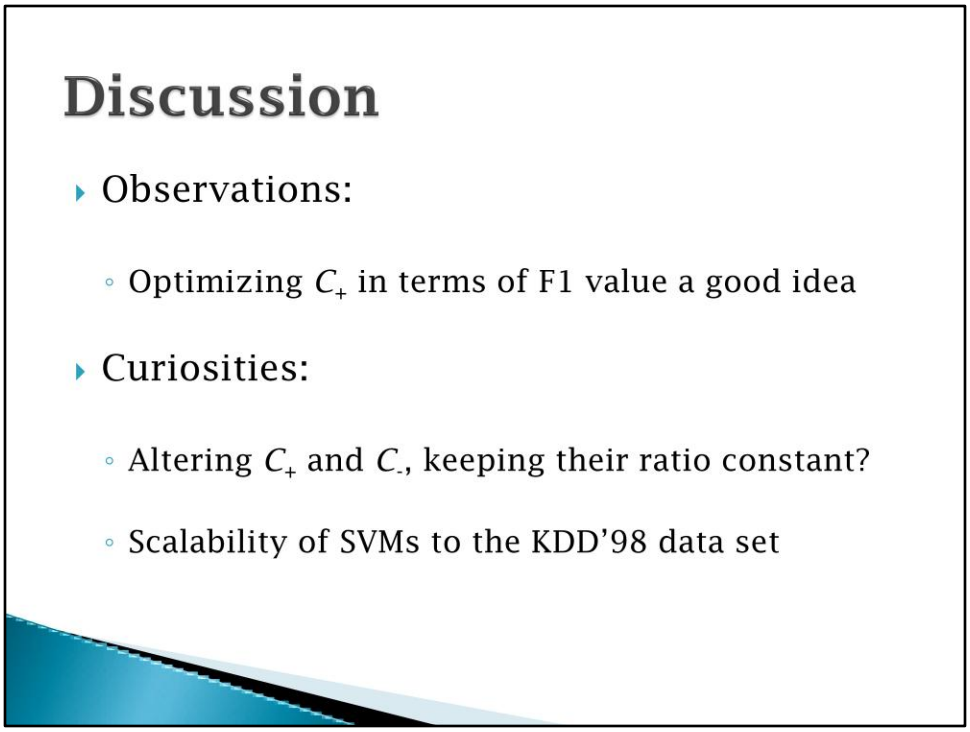


•This slide shows the lift charts for the SVM classifiers over the test set.
•As with the other metrics, SPLIT(8) achieves the best performance for SVMs.

•The lift charts for binned naive Bayes are not shown here, but as before, binned naive Bayes, and in this case also raw naive Bayes performed better than the SVMs.
•This again shows SVM overfitting.



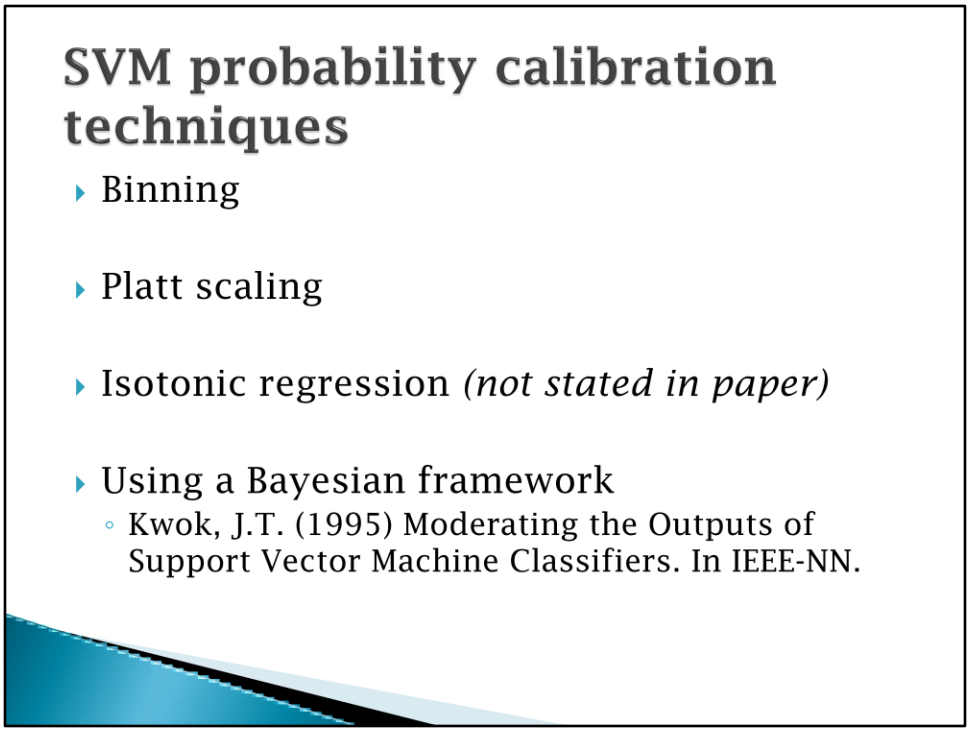
•The experimental results show that overfitting is a major problems for the SVMs here.
•Overfitting was reduced somewhat by using the SPLIT classifiers, but this wasn't enough to match the binned naive Bayes classifier.
•Perhaps using a different kernel, such as a sufficiently high degree polynomial kernel can generalize well without overfitting.



•During the training process, C_+ was optimized in terms of the F1 value. This turned out to be a good idea because all of the probability metrics achieved the best performance when C_+ was equal to 8.

•From these experiments, it is not known whether the ratio of C_+ to C_- or their values have the greatest impact on the results.
•For instance, C_+ can be set to 400, and C_- to 50, keeping the same ratio of 8.

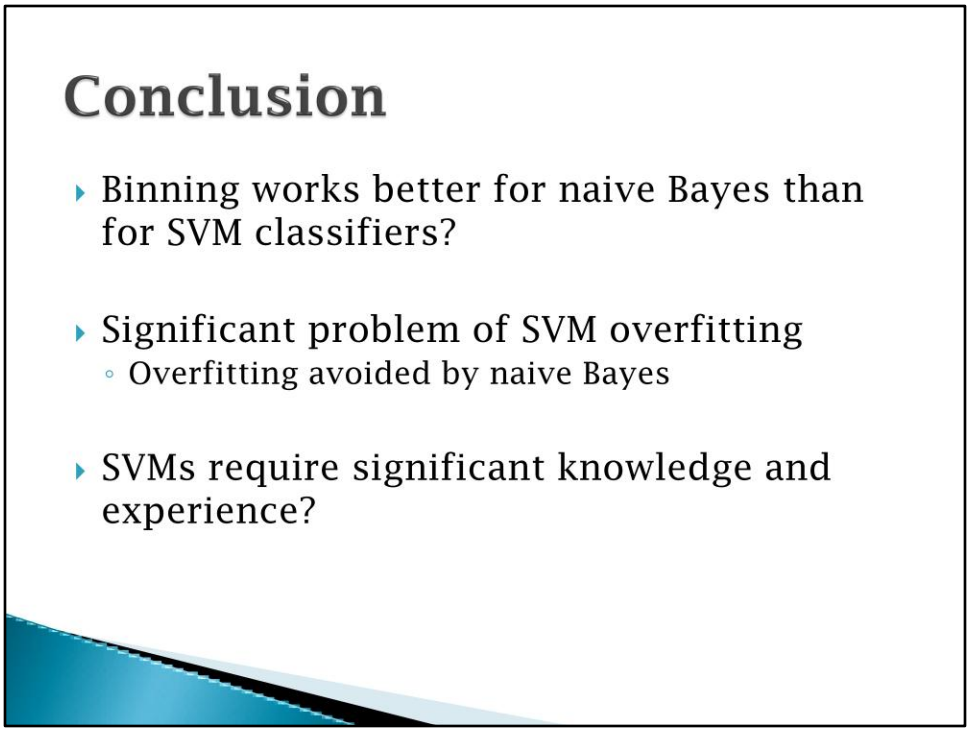
•Experiments were also done with increasing numbers of training examples.
•As expected, the training time correspondingly increased.
•The SVM training time is a quadratic function of the number of training examples, and this is independent of the number of features in the data set.
•It is important to note that having a larger number of features will not have a significant effect on training time.



•To recap, binning is a general technique that can be applied to any classifier that outputs a score.
•It is assumed to be effective as long as the classifier ranks examples well.

•We already know two other techniques for probability calibration – Platt scaling and isotonic regression.

•There is another method for converting the distance scores of SVMs into probabilities.
•This is based on a Bayesian framework, and is discussed in the paper titled "Moderating the Outputs of SVM Classifiers."
•It would be interesting to understand how this technique compares with the ones already tried.



•Having used binning with SVMs over the KDD'98 data set, as stated by the author, it seems that binning just works better with naive Bayes than with SVMs.
•Personally I feel that analysis with multiple data sets should give a better idea of this assertion.

•A significant problem of SVM overfitting was encountered, and this problem is avoided by the naive Bayes classifiers.

•Finally, given that it was difficult to get SVMs to work well on the KDD'98 data set, it seems fair that the practical usage of SVMs requires a significant amount of user knowledge and experience.
•It should again be noted that this paper was published in 2001, and it is probably safe to assume that SVM packages have since made it somewhat easier to optimize the classifier.