NO MORE LATE DAYS LEFT

CIS419 Machine Learning

Assignment III

Part I

James Wang
Oct. 28, 2015

PennKey: jamwang
PennID: 46576241

Part I: Problem Set

1 Probability Decision Boundary

(a) The goal is to minimize expected loss of making a prediction. We observe that if we predict $\hat{y} = 0$ and $y = 1$ is true, then we incur a penalty of 10. And if we predict $\hat{y} = 1$ and $y = 0$ is true, then we incur a penalty of 5. Since falsely predicting $\hat{y} = 0$ is more costly than falsely predicting $\hat{y} = 1$, we would like to shift our probability threshold θ *downwards*, thus making it more likely to predict $\hat{y} = 1$ since $p_1 \geq \theta$ is now more likely to occur than $p_1 \leq \theta$. Now the question becomes: how low do we set θ? It will have to be lower than 0.5, since $\theta = 0.5$ would give predicting $p_1 \leq \theta$ and $p_1 \geq \theta$ an equal probability of occurring. The optimal value to which to set our probability threshold θ would be the point in which the expected loss of predicting $\hat{y} = 0$ given **x** is equal to the expected loss of predicting $\hat{y} = 1$ given **x**.

$$Expected\ Loss\ (\hat{y} = 0 \mid \boldsymbol{x}) = P(y = 0 \mid \boldsymbol{x}) * penalty_{TP} + P(y = 1 \mid \boldsymbol{x}) * penalty_{FP}$$
$$Expected\ Loss\ (\hat{y} = 1 \mid \boldsymbol{x}) = P(y = 0 \mid \boldsymbol{x}) * penalty_{FN} + P(y = 1 \mid \boldsymbol{x}) * penalty_{TN}$$

If $P(y = 0 \mid \boldsymbol{x}) + P(y = 1 \mid \boldsymbol{x}) = 1$

Then $P(y = 0 \mid \boldsymbol{x}) = p_0 = p$

And $P(y = 1 \mid \boldsymbol{x}) = p_1 = (1 - p)$

$$Expected\ Loss\ (\hat{y} = 0 \mid \boldsymbol{x}) = p * penalty_{TP} + (1 - p) * penalty_{FP}$$
$$Expected\ Loss\ (\hat{y} = 1 \mid \boldsymbol{x}) = p * penalty_{FN} + (1 - p) * penalty_{TN}$$

$$penalty_{TP} = 0$$
$$penalty_{FN} = 10$$
$$penalty_{FP} = 0$$
$$penalty_{TN} = 5$$

$$Expected\ Loss\ (\hat{y} = 0 \mid \boldsymbol{x}) = Expected\ Loss\ (\hat{y} = 1 \mid \boldsymbol{x})$$

$$p * penalty_{TP} + (1 - p) * penalty_{FP} = p * penalty_{FN} + (1 - p) * penalty_{TN}$$
$$p * 0 + (1 - p) * 10 = p * 5 + (1 - p) * 0$$
$$\cancel{p * 0} + (1 - p) * 10 = p * 5 + \cancel{(1 - p) * 0}$$
$$(1 - p) * 10 = p * 5$$
$$\frac{p}{(1 - p)} = 2$$

Solving for p, we get…

$$p = \frac{2}{3}$$
$$(1 - p) = \frac{1}{3}$$

Since we want to predict $p = p_1$, $\frac{2}{3}$ of the time, and $(1-p) = p_2$, $\frac{1}{3}$ of the time, we set $\theta = \frac{1}{3}$. This way, the decision rules becomes…

Predict $\hat{y} = 0$ if $p_1 < \theta$ → Predict $\hat{y} = 0$ if $p_1 < \frac{1}{3}$
Predict $\hat{y} = 1$ if $p_1 \geq \theta$ → Predict $\hat{y} = 1$ if $p_1 \geq \frac{1}{3}$

(b) Given this particular loss matrix, the threshold is 2.

2 Double counting the evidence

Given…

$P(Y)$	
$Y = T$	$Y = F$
0.5	0.5

$P(X1 \mid Y)$		
	$Y = T$	$Y = F$
$X_1 = T$	0.8	0.3
$X_1 = F$	0.2	0.7

$P(X2 \mid Y)$		
	$Y = T$	$Y = F$
$X_2 = T$	0.5	0.1
$X_2 = F$	0.5	0.9

$$P(A = a \cap B = b) = P(A = a \mid B = b) * P(B = b)$$

(a)

	$\hat{p}(X_1, Y = T)$	$\hat{p}(X_1, Y = F)$	$\hat{Y}(X_1)$
$X_1 = T$	0.4	0.15	$\hat{Y} = T$
$X_1 = F$	0.1	0.35	$\hat{Y} = F$

Expected Error Rate of Naïve Bayes using only X_1:

$$0.15 + 0.1 = 0.25$$

	$\hat{p}(X_1, Y = T)$	$\hat{p}(X_1, Y = F)$	$\hat{Y}(X_1)$
$X_1 = T$	0.25	0.25	$\hat{Y} = T \text{ or } F$
$X_1 = F$	0.05	0.45	$\hat{Y} = F$

Expected Error Rate of Naïve Bayes using only X_2:

$$0.25 + 0.05 = 0.30$$

(b)

	$\hat{p}(X_1, X_2, Y = T)$	$\hat{p}(X_1, X_2, Y = T)$	$\hat{Y}(X_1, X_2)$
$X_1 = T, X_2 = T$	0.2	0.015	$\hat{Y} = T$
$X_1 = F, X_2 = F$	0.05	0.315	$\hat{Y} = F$
$X_1 = T, X_2 = F$	0.2	0.135	$\hat{Y} = T$
$X_1 = F, X_2 = T$	0.05	0.035	$\hat{Y} = T$

Expected Error Rate of Naïve Bayes using both X_1 & X_2:

$$0.015 + 0.05 + 0.135 + 0.035 = 0.235$$

(c)

	$\hat{p}(X_1, X_2 = X_3, Y = T)$	$\hat{p}(X_1, X_2 = X_3, Y = T)$	$\hat{Y}(X_1, X_2 = X_3)$
$X_1 = T, X_2 = X_3 = T$	0.1	0.0015	$\hat{Y} = T$
$X_1 = F, X_2 = X_3 = F$	0.025	0.2835	$\hat{Y} = F$
$X_1 = T, X_2 = X_3 = F$	0.1	0.1215	$\hat{Y} = F$
$X_1 = F, X_2 = X_3 = T$	0.025	0.0035	$\hat{Y} = T$

Expected Error Rate of Naïve Bayes using both X_1 & X_2 & X_3:

$$0.0015 + 0.025 + 0.1 + 0.0035 = 0.13$$

(d)

The error rate improved from 0.235 to 0.13 when adding in the term X_3, because the predicted label for (X_1 = T, X_2 = X_3 = F) changed from $\hat{Y} = T$ to $\hat{Y} = F$. Naïve Bayes (naively) assumes all attributes are conditionally independent given Y (in Part (C) above, this assumption is clearly violated because X_3 by definition is a copy of X_2), thus, adding an identical variable to the NB classifier will amplify the original variable's effect (if the originally variable improved the error rate, then its identical copy would also improve the error rate, as is the case in Part (C)).

(e)

No, because logistic regression does not make this assumption of conditional independence between the attributes given Y. Two highly correlated attributes will be weighted lower in logistic regression (thus, the effect is not amplified, as is the case in Naïve Bayes).

Part II: Challenge
Generalizing to Unseen Data

1.2 Training the Best Classifier

For this challenge, I chose to train the following ML classifier…because of the Random Forest classifier is simple, yet highly effective and accurate. It does not require pruning, and yet avoids overfitting the data by achieving diversity amongst the various trees it generates through bootstrap sampling and splitting on a small subset of features at each node, at random. The optimal parameters include how many trees to aggregate, number of features to randomly subset at each node, the option of enabling bootstrap sampling, and calculating estimates of accuracy on out-of-sample data with out-of-bootstrap data. I enabled the model to include bootstrap sampling, set the number of features to randomly subset at each split equal to the square root of the total number of features in the data, set the number of jobs to run in parallel equal to the number of cores, ran the classifier through 100 iterations on 10, 100, 1000, and 10000 trees, computed their respective out-of-bootstrap accuracy on each iteration, and averaged the out-of-bootstrap accuracies of each of the classifiers (10, 100, 1000, and 10000). I chose the 100-tree Random Forest because (a) continuing to increase the number of trees showed minimal gain in out-of-bootstrap accuracy and (b) after 100 trees, the run-time starts increasing, so the 100-tree Random Forest maximizes out-of-bootstrap accuracy while minimizing the cost associated with longer run-time.