Shrikanth Mahale (smahale6)

1. SVM

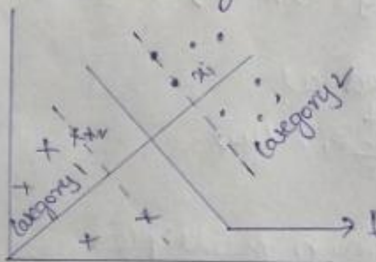
Q 1a)

(a) (6 points) Explain why can we set the margin $c = 1$ to derive an equivalent SVM formulation?

Solution 1a)

Solution 1a)

Support vector machines also known as SVM are supervised learning models, that maximizes the gap of training samples from decision boundary. For visualization purposes plots have been pointed for category 1 and category 2 that are separated by decision boundary.



w is orthogonal to the Decision Boundary

Decision Boundary $= w^T x + b = 0$

$b \rightarrow$ scalar

$c \rightarrow$ margin from the decision boundary

For all points in Class 1 $w^T x + b \leq -c$

For all points in class 2 $w^T x + b \geq c$

with class 1 label as $y = -1$ and class 2 label as $y = 1$

when we combine the above $(w^T x + b)\, y \leq c$

We need to maximize the gap or margin.

The unnormalized margin $\gamma = w^T(x^1 - x^2) = 2c$ where x_1 and x_2 are from category 1 and category 2 respectively.

The margin $\gamma = \frac{2c}{\|w\|}$ where $\|w\|$ is norm of vector w.

$\max_{w,b} \gamma = \frac{2c}{\|w\|}$ such that $y^i(w^T x^i + b) \geq c \ \forall i$

Since magnitude of c ~~merely~~ scales w and b and does not change relative of different classifiers, we can set $c = 1$ and drop 2 to get $\max_{w,b} \gamma = \frac{1}{\|w\|}$ such that

$y^i(w^T x^i + b) \geq 1, \forall i$.

In other words, increasing w, b, c by same constant does not change objective function. So c can be set to 1 to derive equivalent SVM formation

Q 1b)

(b) (7 points) Using Lagrangian dual formulation, show that the weight vector can be represented as

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i.$$

where $\alpha_i \geq 0$ are the dual variables. What does this imply in terms of how to relate data to w?

Solution 1b)

Solution (1b)

From 1(a), we have $\max_{w,b} \gamma = \frac{1}{\|w\|}$ such that $y^i(w^T x^i + b) \geq 1, \forall i$

This can be written as $\min_{w,b} \|w\|^2$ such that $y^i(w^T x^i + b) \geq 1, \forall i$

in standard form.

$$\min_{w,b} \frac{1}{2} w^T w \text{ such that } 1 - y^i(w^T x^i + b) \leq 0, \forall i$$

The Lagrangian function is defined as:

$$L(w, \alpha, \beta) = f(w) + \sum_{i}^{k} \alpha_i g_i(w) + \sum_{i}^{l} \beta_i h_i(w)$$

such that $g_i(w) \leq 0$ $i = 1$ to K

$\alpha_i \geq 0$ and β_i are called Lagrangain multiples

$$L(w, \alpha, b) = \frac{1}{2} w^T w + \sum_{i=1}^{m} \alpha_i (1 - y^i(w^T x^i + b)) \text{ such that } h_i(w) = 0$$

$\alpha \rightarrow$ price of vilolating the constraints

Since KKT conditions are $dL/dw = 0$

$$\frac{dL}{dw} = w - \sum_{i=1}^{m} \alpha_i y^i x^i = 0 \quad (\text{since } d/dw\, w^T w = 2w)$$

$$= \sum_{i=1}^{m} \alpha_i y^i x^i \text{ where } \alpha_i \geq 0 \text{ is Lagrangain multiplier.}$$

For any new point (test point z) $w^T z + b = \sum \alpha_i y^i (x^i z) + b$ {support vectors}

This means classify z as class 1 if the result is positive otherwise class 2.

The optimal w is linear combination of a small number of data points. So to compute the weights α_i & to use the svm's we need to specify only the inner products between examples $x^{i^T} x_j$.

w is the optimal weight with linear combination of features (x^i) & $y^i \rightarrow$ defines the sign and α_i will define the weight from $w = \sum_{i=1}^{m} \alpha_i y^i x_i$

Q1c)

(c) (7 points) Explain why only the data points on the "margin" will contribute to the sum above, i.e., playing a role in defining w. Hint: use the Lagrangian multiplier derivation and KKT condition we discussed in class.

Solution 1c)

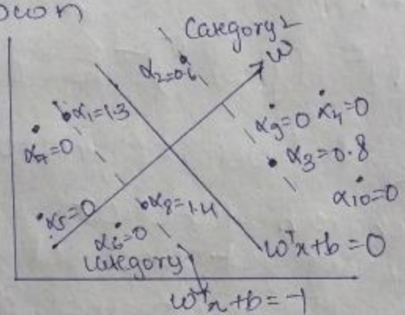
Solution 1c)

One of the KKT condition states that $\alpha_i g_i(w) = 0$

Since $g_i(w) = 1 - y^i(w^T x^i + b)$ from Lagrangian multiplier derivation in part (b)

For $\alpha_i g_i(w) = 0$ either $\alpha_i = 0$ or $g_i(w) = 0$ when $\alpha_i = 0$ (you don't pay price to violate constraint) means. $(1 - y^i(w^T x^i + b)) < 0 \rightarrow$ indicates points are not on margin. When $g_i(w) = 0$ (we pay price α_i since $\alpha_i \neq 0$ to violate constraint) means $(1 - y_i(w^T x^i + b)) = 0$, $\alpha_i > 0$

These points are on margin with distance away from the decision boundary. These are called support vectors as these will only have non zero α coefficient as shown



As seen in above graph only points 1, 2 and 8 have non-zero coefficients out of 10 points. This means the weight vector $W = \sum_{i=1}^{n} \alpha_i y_i x_i$ will only consist of 3 points. So w only depends on points on margin. This simplifies optimization problem of sum as only points on margin will define the sum.

2. Naive Bayes for spam filtering

In this problem we will use the Naive Bayes algorithm to fit a spam filter by hand. This will enhance your understanding to Bayes classifier and build intuition.

Spam filters are used in all email services to classify received emails as "Spam" or "Not Spam". A simple approach involves maintaining a vocabulary of words that commonly occur in "Spam" emails and classifying an email as "Spam" if the number of words from the dictionary that are present in the email is over a certain threshold. We are given the vocabulary consists of 15 words

$V = \{$secret, offer, low, price, valued, customer, today, dollar, million, sports, is, for, play, healthy, pizza$\}$.

We will use V_i to represent the ith word in V. As our training dataset, we are also given 3 example spam messages,

- million dollar offer
- secret offer today
- secret is secret

and 4 example non-spam messages

- low price for valued customer
- play secret sports today
- sports is healthy
- low price pizza

Recall that the Naive Bayes classifier assumes the probability of an input $x = [x_1, x_2, \ldots, x_n]^T$ depends on its class y. In our case the input vector x corresponding to each message has length $n = 15$ equal to the number of words in the vocabulary V, where each entry x_i is equal to the number of times word V_i occurs in x.

Q 2a)

(a) (10 points) Calculate $\mathbb{P}(y = 0)$ and $\mathbb{P}(y = 1)$ from the training data, where $y = 0$ corresponds to spam messages, and $y = 1$ corresponds to non-spam messages.

Solution 2a)

$\boldsymbol{y = 0}$ → Spam
$\boldsymbol{y = 1}$ → Non spam

$\boldsymbol{P(y = 0)}$ = number of spam messages / total number of messages = 3/7

$\boldsymbol{P(y = 1)}$ = number of non-spam messages/ total number of messages = 4/7

Q 2b)

(b) (10 points) List the feature vector x for each spam and non-spam message.

Solution 2b)

Feature vectors:

I. (million-dollar offer = [0,1,0,0,0,0,0,1,1,0,0,0,0,0,0]
II. secret offer today = [1,1,0,0,0,0,1,0,0,0,0,0,0,0,0]
III. secret is secret = [2,0,0,0,0,0,0,0,0,0,1,0,0,0,0]
IV. low price for valued customer = [0,0,1,1,1,1,0,0,0,0,0,1,0,0,0,0]
V. play secret sports today = [1,0,0,0,0,0,1,0,0,1,0,0,1,0,0]
VI. sports is healthy = [0,0,0,0,0,0,0,0,0,1,1,0,0,1,0]
VII. low price pizza = [0,0,1,1,0,0,0,0,0,0,0,0,0,0,1]

Q2c)

(c) (10 points) In the Naive Bayes model, the likelihood of a sentence with feature vector x given a class c is

$$\mathbb{P}(x|y=c) = \prod_{k=1}^{n} \theta_{c,k}^{x_k}$$

where $\theta_{c,k} \in (0,1)$ is the weight of word k in class c, which satisfies $\sum_{k=1}^{n} \theta_{c,k} = 1, \forall c$. Calculate the maximum likelihood estimates of $\theta_{0,1}$, $\theta_{0,7}$, $\theta_{1,1}$, $\theta_{1,15}$ by maximizing $\mathbb{P}(x|y=c)$ with respect to $\theta_{c,k}$ and given data. (Hint: Consider the Lagrangian function for solving this constrained optimization problem. You only need to introduce one Lagrangian multiplier because you only have one constraint. Consider log-likelihood (i.e., taking log of the cost function). Then solve it from there.)

Solution 2c)

Solution 2(c)

Bayes rule $q_i(x) = P(y=i|x) = \frac{P(x|y)P(y)}{P(x)}$

If $q_i(x) > q_k(x)$, then $y=i$ else $y=j$

Our goal is to maximize $P(x|y=c)$ w.r.t $\theta_{c,k}$

$P(x,y) = \prod_{i=1}^{m} (x^i, y^i)$ $m \to$ no of data points

$n \to$ length of feature vector x (15)

$= \prod_{i=1}^{m} P(y^i) P(x^i|y^i) = \prod_{i=1}^{m} \left[P(y^i) \prod_{k=1}^{n} \theta_{c,k}^{x_k} \right]$

Taking log of above

$\log P(x,y) = \sum_{i=1}^{m} \log \left(P(y^i) \prod_{k=1}^{n} \theta_{c,k}^{x_k^i} \right)$

$= \sum_{i=1}^{m} \left(\log \left[P(y^i=1)^c \cdot P(y^i=0)^{1-c} \right) + \sum_{k=1}^{n} \log \theta_{c,k}^{x_k^i} \right.$

Such that constraint $\sum_{k=1}^{n} \theta_{c,k} = 1$ (Weight of vector $=1$)

Maximize $\log P(x,y)$ using Lagrangian function

Taking first derivative w.r.t $\theta_{c,k}$ we get and set to 0

$\frac{1}{\theta_{c,k}} \sum_{i=1}^{m} x_k^i \, 1(y^i=c) + \lambda = 0 \to$ (Since 1st term doesnot depend on $\theta_{c,k}$ zero)

$\theta_{c,k} = -\sum_{i=1}^{m} x_k^i \, 1(y^i=c)/\lambda \to (a)$

Since $\sum_{k=1}^{n} \theta_{c,k} = 1$, Plugging it back, we get

$\lambda = -\sum_{k=1}^{n} \sum_{i=1}^{m} x_k^i \, 1(y^i=c)$. Putting λ back in (a) we get

$\theta_{c,k} = -\sum_{i=1}^{m} x_k^i \, 1(y^i=1) \Big/ \sum_{k=1}^{n} \sum_{i=1}^{m} x_k^i \, 1(y^i=1)$

where 1 is an indicator function that take value of c (0 or 1 in this case)

- θ (0,1): is likelihood estimate of word *secret* in spam as *secret* is 1st word in the vocabulary list. Since *secret* appears three times in the spam messages out of total 9 words in spam so θ (0,1)=3/9=1/3=0.3333
- θ (0,7): is likelihood estimate of word *today* in spam as today is 7th word in the vocabulary list. Since *today* appears once in the spam messages out of total 9 words in spam so θ (0,7)=1/9=0.11111
- θ (1,1): is likelihood estimate of word *secret* in non-spam as *secret* is 1st word in the vocabulary list. Since *secret* appears once in the non-spam messages out of total 15 words in non-spam so θ (1,1)=1/15= 0.06667
- θ (1,15): is likelihood estimate of word *pizza* in non-spam as *pizza* is 15th word in the vocabulary list. Since *pizza* appears once in the non-spam messages out of total 15 words in non-spam so θ (1,15)=1/15= 0.06667

(d) (10 points) Given a new message "today is secret", decide whether it is spam or not spam, based on the Naive Bayes classifier, learned from the above data.

Solution 2d)

- The poster probability of a test point "today is secret" in spam:
  As *today* is vocabulary word # 7, *is* vocabulary word # 11 and *secret* is word #1 .
  q_i(x):=P(y=0|x)= θ (0,7)* θ (0,11)* θ (0,1)*p(y=0) (a)
  Not considering (ignoring) other vocabulary words in calculating q_i(x) & q_j(x), since they are not in test point, therefore it will be raised to power of zero and their value will turn out to be 1.
  It will not impact the output θ (0,7) from above =1/9=0.11111 θ (0,1) from above =1/3=0.3333
  θ (0,11): is likelihood estimate of word *is* in spam as is 11th word in the vocabulary list.
  Since *is* appears once in the spam messages out of total 9 spam words so
  θ (0,11)=1/9=0.11111
  P(y=0) from part a is 3/7.
  Plugging in values in (a) q_i(x)=(1/9)*(1/9)*(1/3)*3/7= 0.0017637

- The poster probability of a test point "*today is secret*" in non-spam:
  As *today* is vocabulary word # 7, *is* vocabulary word # 11 and *secret* is word #1 .
  q_j(x):=P(y=1|x)= θ (1,7)* θ (1,11)* θ (1,1)*p(y=1)--(b) θ (1,1) from above =1/15= 0.06667
  θ (1,7): is likelihood estimate of word *today* in non-spam as *today* is 7th word in the vocabulary list
  Since out of the total 15 words, *today* appears only once in the non-spam messages so θ (1,7)=1/15= 0.06667
  θ (1,11): is likelihood estimate of word *is* in non-spam as *is* is 7th word in the vocabulary list.
  Since out of the total 15 words, *is* appears only once in the non-spam messages so θ (1,11)=1/15= 0.06667
  P(y=1) from part a is 4/7.
  Inserting in values in (b) q_j(x)=(1/15)*(1/15)*(1/15)*4/7= 0.00016931
  Since q_i(x) > q_j(x) as (0.0017637>= 0.00016931), we can conclude that
  "today is secret" will be classified as spam message

3. Comparing Bayes, logistic and KNN classifiers.

In lectures we learn three different classifiers. This question is to implement and compare them. We are suggest use Scikit-learn, which is a commonly-used and powerful Python library with various machine learning tools. But you can also use other similar library in other languages of your choice to perform the tasks.

Q 3a)

(a) **Part One (Divorce classification/prediction).** (20 points)

This dataset is about participants who completed the personal information form and a divorce predictors scale.

The data is a modified version of the publicly available at `https://archive.ics.uci.edu/ml/datasets/Divorce+Predictors+data+set` (by injecting noise so you will not replicate the results on uci website). There are 170 participants and 54 attributes (or predictor variables) that are all real-valued. The dataset **q3.csv**. The last column of the CSV file is label y (1 means "divorce", 0 means "no divorce"). Each column is for one feature (predictor variable), and each row is a sample (participant). A detailed explanation for each feature (predictor variable) can be found at the website link above. Our goal is to build a classifier using training data, such that given a test sample, we can classify (or essentially predict) whether its label is 0 ("no divorce") or 1 ("divorce").

Build three classifiers using (Naive Bayes, Logistic Regression, KNN). Use the first 80% data for training and the remaining 20% for testing. If you use scikit-learn you can use train_test_split to split the dataset.

i. Report testing accuracy for each of the three classifiers. Comment on their performance: which performs the best and make a guess why they perform the best in this setting.

ii. Use the first two features to train three new classifiers. Plot the data points and decision boundary of each classifier. Comment on the difference between the decision boundary for the three classifiers. Please clearly represent the data points with different labels using different colors.

Solution 3a)

Below are the performance stats on the 3 classifiers

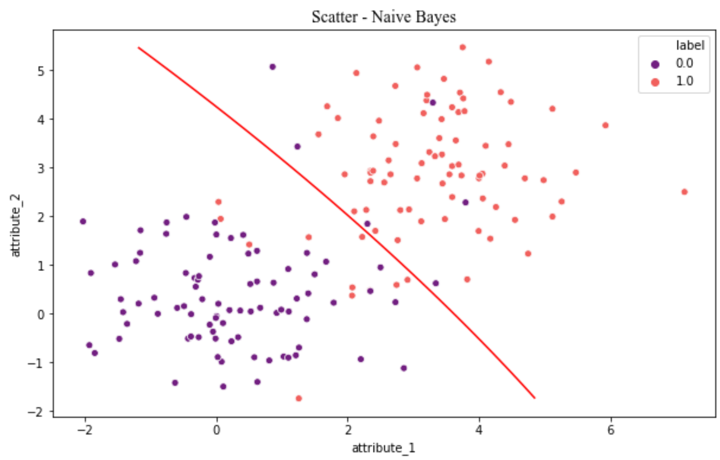
CLASSIFIER	PERFORMANCE %
NAÏVE BAYES	**100 %**
LOGISTIC REGRESSION	**97 %**
KNN	**100 %**

Logistic regression is not adequately capable to catch complex relationships since the decision boundary is linear which may be the reason why logistic regression did not do as well as the other two may be because

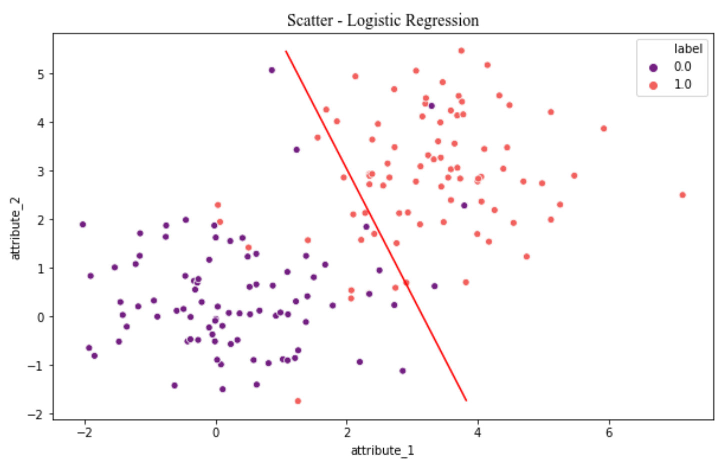
For Naïve Bayes case, the attributes in the models are essentially independent, which plays well with Naive Bayes, and the decision boundary is also non-linear in capturing complex relationships.

KNN is resilient to noise in the training results, when we call several neighbors which may be the reason why it performed well.

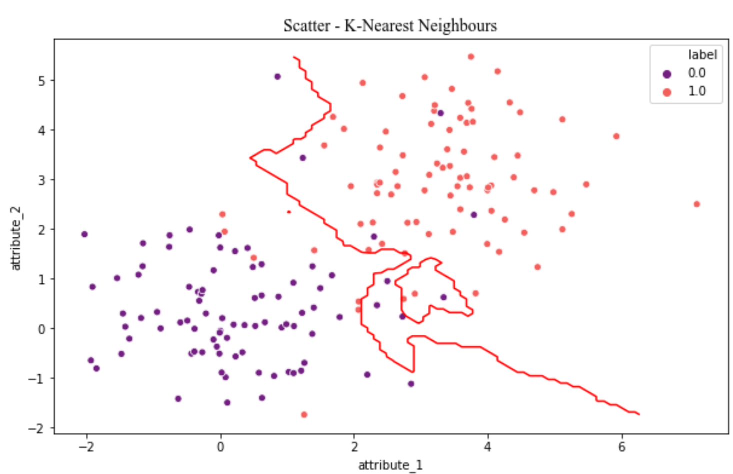
Naive Bayes: Decision Boundary is Quadratic.



Logistic Regression: Decision Boundary is Linear.



KNN: Decision Boundary is non-linear.



Q 3b)

(b) **Part Two (Handwritten digits classification).** (20 points)

Repeat the above using the **MNIST Data** in our previous homework. Here, give "digit" 6 label $y = 1$, and give "digit" 2 label $y = 0$. All the pixels in each image will be the feature (predictor variables) for that sample (i.e., image). Our goal is to build classifier to such that given a new test sample, we can tell is it a 2 or a 6. Using the first 80% of the samples for training and remaining 20% for testing. Report the classification accuracy on testing data, for each of the three classifiers. Comment on their performance: which performs the best and make a guess why they perform the best in this setting.

Solution 3b)

CLASSIFIER	PERFORMANCE %
NAÏVE BAYES	76 %
LOGISTIC REGRESSION	97 %
KNN	99 %

It appears that KNN did the best. KNN in the training data is resilient to noise which is why it must have done well. Even the outliers are distributed uniformly in the same cluster withno aggregations. Those clustered outliers won't fool the KNN algorithm.