# Data Mining Practice Exam 1: Classification

## Part 1

After the first exam in a data mining course, the results of the exam were recorded along with some information about each student. The data is below:

| ID | Passed All Assignments | GPA | Language | Passed Exam |
|---|---|---|---|---|
| 1 | No | 3.1 | Python | Yes |
| 2 | No | 2.0 | Python | No |
| 3 | Yes | 3.5 | C++ | Yes |
| 4 | Yes | 2.5 | Java | Yes |
| 5 | Yes | 3.9 | Python | No |
| 6 | No | 2.9 | C++ | No |
| 7 | Yes | 3.2 | Java | Yes |

**1. Using a KNN classifier with K=3, predict whether the following student will pass the exam. (Do not worry about normalizing the data.)**

| | | | | |
|---|---|---|---|---|
| 8 | Yes | 3.0 | C++ | ? |

dist(1,8) = sqrt($1^2$ + $0.1^2$ + $1^2$) = sqrt(2.01)
dist(2,8) = sqrt($1^2$ + $1.0^2$ + $1^2$) = sqrt(3.0)
dist(3,8) = sqrt($0^2$ + $0.5^2$ + $0^2$) = sqrt(0.25)
dist(4,8) = sqrt($0^2$ + $0.5^2$ + $1^2$) = sqrt(1.25)
dist(5,8) = sqrt($0^2$ + $0.9^2$ + $1^2$) = sqrt(1.81)
dist(6,8) = sqrt($1^2$ + $0.1^2$ + $0^2$) = sqrt(1.01)
dist(7,8) = sqrt($0^2$ + $0.2^2$ + $1^2$) = sqrt(1.04)

Students 3, 6, and 7 are the nearest neighbors, so that's 2 votes for yes and 1 vote for no. We predict the student will pass.

**2. Using a Naive Bayes classifier, predict if the student will pass. Bin the GPA feature into >=3.0 and <3.0**

P(Yes | x) = P(Yes) * P(passed assignments=yes | Yes) * P(GPA>=3.0 | Yes) * P(C++ | Yes)
= (4/7) * (3/4) * (3/4) * (1/4) = 0.08

P(No | x) = P(No) * P(passed assignments=yes | No) * P(GPA>=3.0 | No) * P(C++ | No)
= (3/7) * (1/3) * (1/3) * (1/3) = 0.016
Probability of yes is higher, so we predict student will pass.

**3. If we want to create a decision tree to classify the data, what is the best attribute to split on first?**

- **Bin the GPA feature into >=3.0 and <3.0.**
- **Use Gini index as the measure of impurity**
- **(Also know how to use entropy as the measure of impurity, either one is fair game for the exam!)**
- **Because 'Language' can be split 3 ways, use Gain Ratio to compare split quality**

Gini(Overall data) = $1 - (4/7)^2 - (3/7)^2 = 1 - 0.327 - 0.184 = 0.489$

Calculate the Gini index & gain ratio of splitting on 'passes assignments':
Gini(pass assignments = yes) = $1 - (3/4)^2 - (1/4)^2 = 1 - 0.563 - 0.063 = 0.374$
Gini(pass assignments = no) = $1 - (1/3)^2 - (2/3)^2 = 1 - 0.111 - 0.0.444 = 0.445$
Gini(of this split) = $(4/7)(0.374) + (3/7)(0.445) = 0.214 + 0.191 = 0.405$
Gain = $0.489 - 0.405 = 0.084$
SplitInfo = $-(4/7)\log_2(4/7) - (3/7)\log_2(3/7) = 0.985$
GainRatio = $0.084 / 0.985 = 0.0853$

*[This is a side-note to show one example of calculating entropy.*
*The rest of the entropy calculations are not shown.]*
Calculate the entropy of splitting on 'passes assignments':
Entropy(pass assignments = yes) = $-(3/4)\log_2(3/4) - (1/4)\log2(1/4) = 0.811$
Entropy(pass assignments = no) = $-(1/3)\log_2(1/3) - (2/3)\log_2(2/3) = 0.918$
Entropy(this split) = $(4/7)(0.811) + (3/7)(0.918) = 0.856$

Calculate the Gini index & gain ratio of splitting on 'GPA':
Gini(GPA>=3.0) = $1 - (3/4)^2 - (1/4)^2 = 0.374$
Gini(GPA < 3.0) = $1 - (1/3)^2 - (2/3)^2 = 0.445$
Gini(of this split) = $(4/7)(0.374) + (3/7)(0.445) = 0.214 + 0.191 = 0.405$
Gain = $0.489 - 0.405 = 0.084$
SplitInfo = $-(4/7)\log_2(4/7) - (3/7)\log_2(3/7) = 0.985$
GainRatio = $0.084 / 0.985 = 0.0853$

Calculate the Gini index & gain ratio of splitting on 'Language':
Gini(Lang = Python) = $1 - (1/3)^2 - (2/3)^2 = 0.445$
Gini(Lang = C++ ) = $1 - (1/2)^2 - (1/2)^2 = 0.5$
Gini(Lang = Java) = $1 - (2/2)^2 - (0/2)^2 = 0$
Gini(of this split) = $(3/7)(0.445) + (2/7)(0.5) + (2/7)(0) = 0.191 + 0.143 + 0 = 0.334$
Gain = $0.489 - 0.334 = 0.155$
SplitInfo = $-(3/7)\log_2(3/7) - (2/7)\log2(2/7) - (2/7)\log2(2/7) = 1.556$
GainRatio = $0.155 / 1.556 = 0.0996$
Splitting on language gives the highest gain ratio, so we would split on language first.

## Part 2

Given the following cost matrix and the confusion matrices for two different classifiers:

| Cost Matrix | | Predicted | |
|---|---|---|---|
| | | + | - |
| Actual | + | -1 | 20 |
| | - | 2 | 0 |

| Classifier 1 | | Predicted | |
|---|---|---|---|
| | | + | - |
| Actual | + | 50 | 20 |
| | - | 130 | 300 |

| Classifier 2 | | Predicted | |
|---|---|---|---|
| | | + | - |
| Actual | + | 60 | 10 |
| | - | 30 | 400 |

**1. Which classifier is better on the basis of error rate?**
Error(C1) = 150/500 = 0.3
Error(C2) = 40/500 = 0.08
C2 has lower error, so is better

**2. Which classifier is better on the basis of F-measure (for the positive class only)?**
Precision(C1) = 50/180 = 0.278
Recall(C1) = 50/70 = 0.714
F(C1) = (2 * 0.278 * 0.714) / (0.278 + 0.714) = 0.396 / 0.992 = 0.399

Precision(C2) = 60/90 = 0.667
Recall(C2) = 60/70 = 0.857
F(C2) = (2 * 0.667 * 0.857) / (0.667 + 0.857) = 1.143 / 1.524 = 0.75

C2 has a higher F-measure, so is better

**3. Which classifier is better on the basis of cost?**
Cost(C1) = 50(-1) + 20(20) + 130(2) + 300(0) = 610
Cost(C2) = 60(-1) + 10(20) + 30(2) + 400(0) = 200
C2 has a lower cost, so is better

**4. Given a KNN classifier with k=7, and a new record to classify, the KNN classifier finds that of the 7 nearest neighbors to this new record, 6 of them are in the negative class, and 1 of them is in the positive class. Using the cost matrix above, what would you classify this new record as, based on Risk?**
P(-) = 6/7 = 0.857
P(+) = 1/7 = 0.143
Risk(-) = 20(1/7) = 2.857
Risk(+) = 2(6/7) = 1.714
We would classify it as the positive class because it has less risk.

## Part 3 - Short Answers

**1. What is the difference between noise and outliers?**
Noise are errors in the data. Outliers are legitimate data points that are different than the typical values in the data.

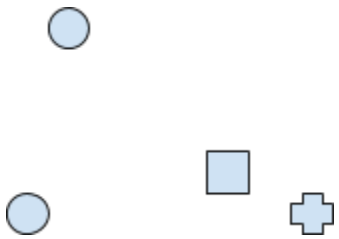
**2. Give 2 ways of dealing with missing values in a dataset.**
Eliminate them via list-wise deletion or pair-wise deletion
Imputation: fill missing values with a mean/median/mode, or other reasonable value
Prediction model: use data mining to predict the missing values

**3. What is the curse of dimensionality?**
As dimensionality increases, volume of the space increases rapidly and the data becomes increasingly sparse in the space it occupies. Many types of data mining become significantly harder as the dimensionality of the data increases. Distances between points and differences between points become less meaningful in high-dimensional spaces.

**4. Explain the difference between bagging and boosting.**
Bagging chooses multiple bootstrap samples (sampling with replacement). Each record always has the same probability of being selected for the sample. A base classifier is trained on each bootstrap sample and they can all be trained in parallel.

Boosting first chooses one bootstrap sample and trains one base classifier on that sample. Records that are classified incorrectly by that classifier are given a larger weight and records classified correctly are given a smaller weight. Then, another bootstrap sample is chosen with the records' weights being used as probability for selection. Another base classifier is trained on this new sample. The process repeats with the weights adjusting at each step. Base classifiers must be trained iteratively with this method.

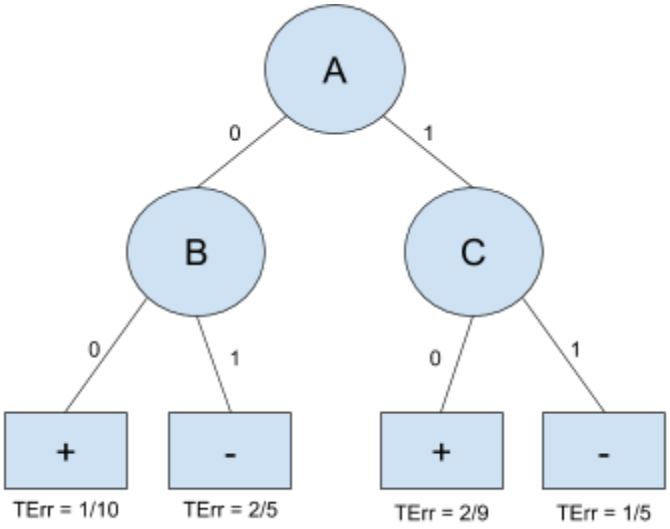
**5. Describe (or draw) a situation in which using un-weighted voting for KNN gives you a different classification than weighted voting.**



If there are 2 classes: circles and squares, and we are trying to classify the plus '+' with KNN using k=3, unweighted voting would have 2 votes for circle and 1 for square. Weighted voting would have a higher weight for the square's vote and the square's vote could outweigh the votes of the 2 circles.

## Part 4

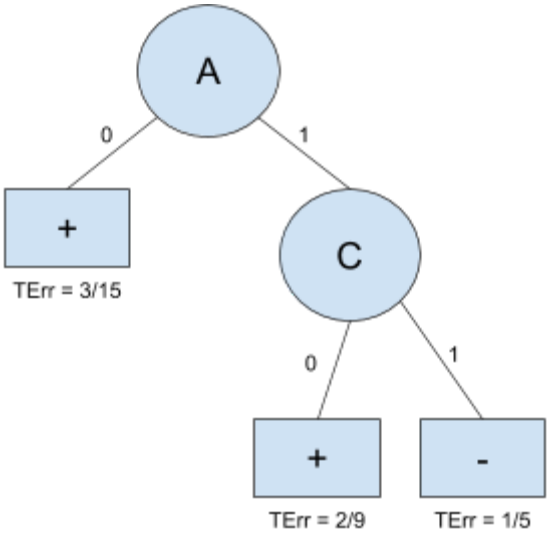
**1. Given the following decision tree and validation data, which nodes would be pruned with the Reduced Error Pruning algorithm? The error rate from the training data is noted at each leaf node.**




Validation set:

| A | B | C | Class |
|---|---|---|---|
| 0 | 0 | 0 | + |
| 0 | 1 | 1 | + |
| 1 | 1 | 0 | - |
| 1 | 0 | 0 | - |
| 0 | 1 | 0 | + |

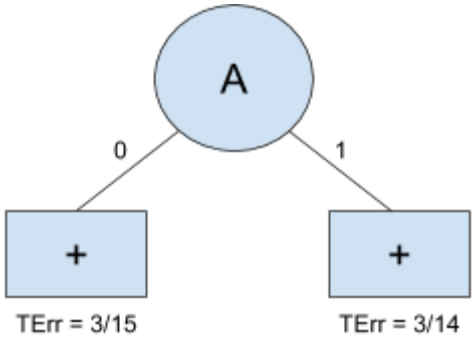
If we prune node B, the majority class would be '+' and the training error is 3/15 = 0.2.
The resulting tree would look as follows:




When we run the validation data through this tree, the validation error at the newly pruned node is 0/3 = 0. Because this error is lower, we prune this node.

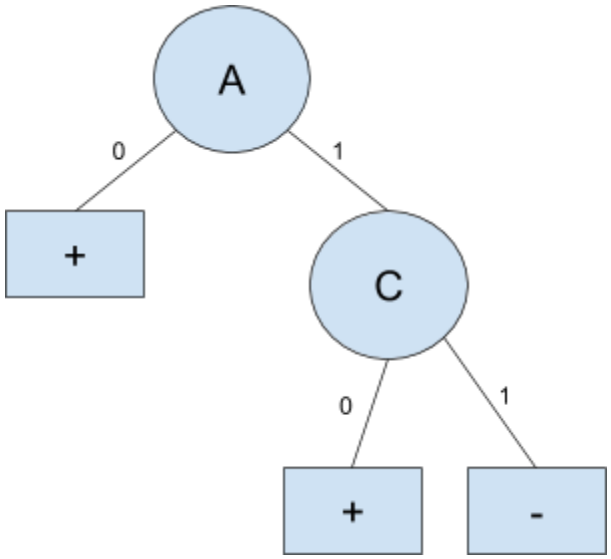
If we prune node C, the majority class would be ‘+’ and the training error is 3/14 = 0.214.

The resulting tree would look at follows




When we run the validation data through this tree, the validation error at the newly pruned node is 2/2 = 1. Because this error is higher, we do not prune this node.

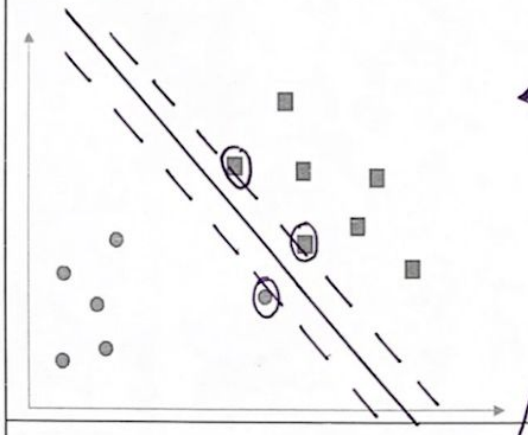
The final tree looks like this:



**Part 5**

Name ANSWERS

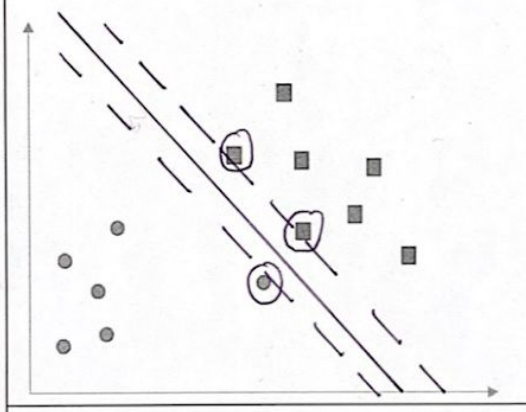
1. Draw a hyperplane generated by a linear hard-margin SVM. Circle the support vectors.



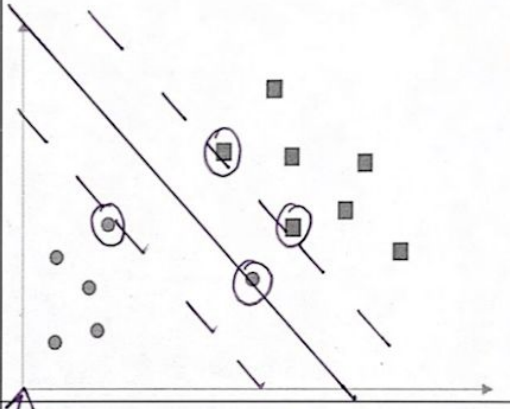
(1) A hard-margin SVM does not allow any misclassifications.

(2a) When C is large, the cost for misclassifications is high, so the SVM will try to avoid misclassifications by narrowing the margin.

2. Draw a hyperplane generated by a linear soft-margin SVM with a large value of C (i.e. $C \to \infty$). Circle the support vectors.

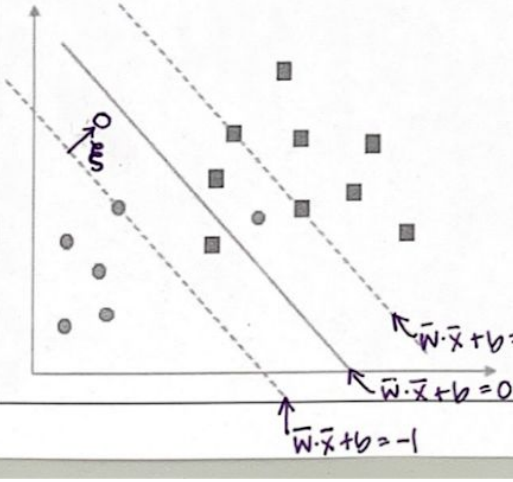


2. Draw a hyperplane generated by a linear soft-margin SVM with a small value of C (i.e. $C \to 0$). Circle the support vectors.



(2b) When C is small, the cost for misclassifications is low, so the SVM will be able to widen the margin, even if it makes more misclassifications.

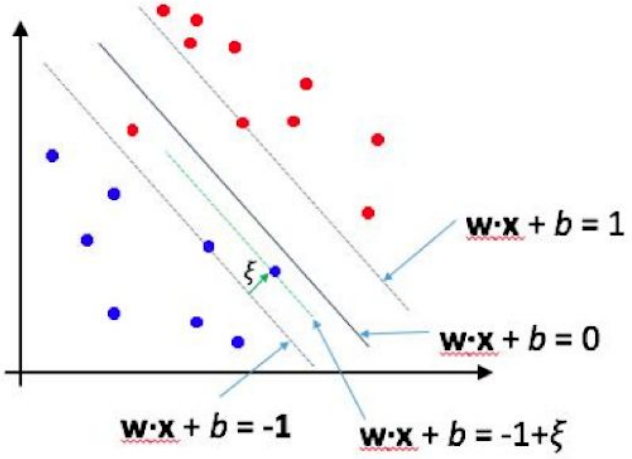
3. Draw a circle-class data point with slack $0 < \xi < 1$




(3) Explanation below...

**#3 Answer:**

Recall the equations for the hyperplane, the margins, and slack:



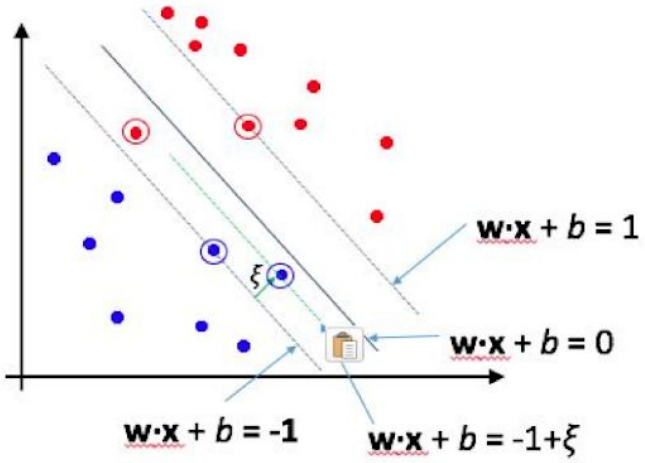

The margins are defined by wx+b=-1 / wx+b=1. And the hyperplane is defined by wx+b=0.

Slack is the distance from the margin [on the side where the point *should* be classified] to the point that is on the wrong side of the margin (i.e. For a negative data point, the distance from the margin on the negative side to that point; or for a positive data point, the distance from the margin on the positive side to that point.)

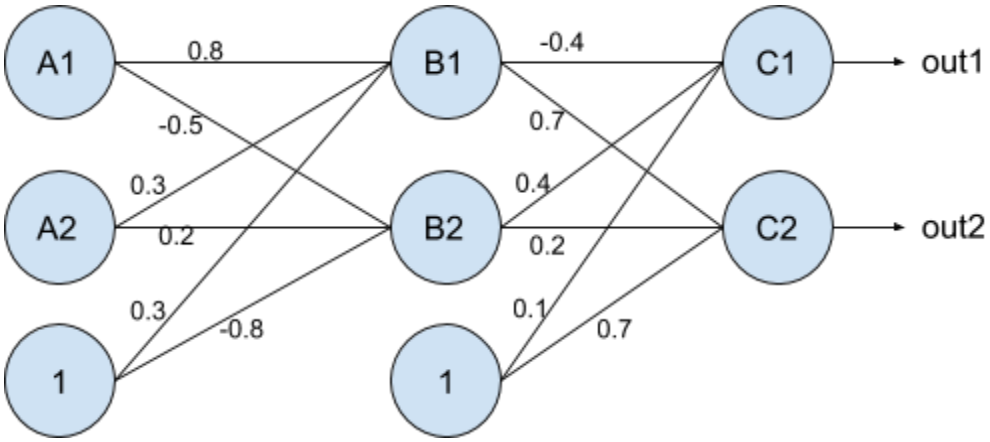
If the slack was 0, the point would be on the margin. For a negative point, this means on the *negative-side* margin. If the slack was 1, the point would be on the hyperplane. So a negative point with slack in between 0 and 1 is between the negative margin and the hyperplane. *Note that this point is still classified correctly.*

Note: We only calculate slack for points that are beyond the margin on their respective side. There is no notion of 'negative slack.' Points that are on the correct side of their margin are essentially ignored - they are not contributing to the placement of the optimal decision boundary. Recall that support vectors are the points that define the location of the hyperplane - the points that *do* contribute to the placement of the optimal decision boundary. Points on the wrong side of the margin are contributing to the placement of the hyperplane because their slack is part of the constrained optimization equation, so they are also support vectors.

Support vectors are circled:



## Part 6




**1. For the above neural network, out1 represents Class A and out2 represents Class B. All nodes use the sign function as their activation function.**

**Show the forward pass, in matrix form, of the following record.**
**How would this neural network classify this record (Class A or Class B)?**

| Feature1 | Feature2 | Class |
|---|---|---|
| 5 | 10 | ? |

$A1 = 5$
$A2 = 10$

net $B1 = (5)(0.8) + (10)(0.3) + (1)(0.3) = 7.3$
net $B2 = (5)(-0.5) + (10)(0.2) + (1)(-0.8) = -1.3$

activation function (sign function):

$B1 = \sigma(7.3) = 1$
$B2 = \sigma(-1.3) = -1$

net $C1 = (1)(-0.4) + (-1)(0.4) + (1)(0.1) = -0.7$
net $C2 = (1)(0.7) + (-1)(0.2) + (1)(0.7) = 1.2$

activation function (sign function):

$C1 = \sigma(-0.7) = -1$
$C2 = \sigma(1.2) = 1$

C2 is activated more than C1, so we classify as class B.