Northeastern University
CS6220 – Data Mining Techniques
Fall 2017, Derbinsky

# Self Test

Name: ______________________________

| Problem | Points |
|---|---|
| 1. Bayes' Rule | /0 |
| 2. Probability Distributions | /0 |
| 3. Discrete Expectation | /0 |
| 4. Expectation Properties | /0 |
| 5. Matrices/Linear Equations | /0 |
| 6. Matrices | /0 |
| 7. Eigenvalues/Eigenvectors | /0 |
| 8. Norms | /0 |
| 9. Naïve Bayes | /0 |
| **Total** | /0 |

## Instructions

- This assignment will not be graded for correctness
- Use this as an opportunity to self-assess, and get yourself up to speed (in math, coding, & LaTeX)

## (0 pts.) 1. Bayes' Rule

The Weatherly app predicts rain tomorrow. In recent years, it has rained only 73 days each year. When it actually rains, the Weatherly app correctly forecasts rain 70% of the time. When it doesn't rain, the app incorrectly forecasts rain 30% of the time. What is the probability that it will rain tomorrow?

Hint: $P(H|D) = \frac{P(H)P(D|H)}{P(D)}$

## (0 pts.) 2. Probability Distributions

Given the following probability density function (PDF) of a random variable $x$ . . .

$$p(x) = \begin{cases} 4x & 0 \leq x \leq \frac{1}{2} \\ -4x + 4 & \frac{1}{2} \leq x \leq 1 \end{cases}$$

What is the equation and graph of the corresponding cumulative density function (CDF)?

## (0 pts.) 3. Discrete Expectation

Calculate the expected value of $X$, $E[X]$, where $X$ is a random variable representing the outcome of a roll of a trick die. Use the sample space $x \in \{1, 2, 3, 4, 5, 6\}$ (i.e. six-sided die) and let

$$P(X = x) = \begin{cases} \frac{1}{2} & x = 1 \\ \frac{1}{10} & x \neq 1 \end{cases}$$

## (0 pts.) 4. Expectation Properties

Use the properties of expectation to show that we can rewrite the variance of a random variable $X$ ...

$Var[X] = E[(X - \mu)^2]$

as ...

$Var[X] = E[X^2] - (E[X])^2$

## (0 pts.) 5. Matrices/Linear Equations

Consider the following system of equations .....

$2x_1 + x_2 + x_3 = 3$
$4x_1 + 2x_3 = 10$
$2x_1 + 2x_2 = -2$

a. Write the system as a matrix equation of the form $Ax = b$.

b. Write the solution of the system as a column $s$ and verify by matrix multiplication that $As = b$.

c. Write $b$ as a linear combination of the columns in $A$.

## (0 pts.) 6. Matrices

Consider the following matrix ...

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 4 & 3 \\ 1 & 3 & 4 \end{pmatrix}$$

a. What is the determinant, $det(A)$ or $|A|$, of the matrix?

b. Is the matrix invertible?

c. What is the rank of the matrix?

## (0 pts.) 7. Eigenvalues/Eigenvectors

The eigenvalues of the following matrix ...

$$A = \begin{pmatrix} 3 & 6 \\ 1 & 4 \end{pmatrix}$$

are $\lambda = 6$ and $\lambda = 1$. Which of the following is an eigenvector for $\lambda = 1$?

a. $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$

b. $\begin{pmatrix} -3 \\ 1 \end{pmatrix}$

c. $\begin{pmatrix} 3 \\ 1 \end{pmatrix}$

d. $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$

## (0 pts.) 8. Norms

Find the 0, 1, 2, and $\infty$ norms of $x$ ...

$$x = \begin{pmatrix} 2 \\ 1 \\ -4 \\ -2 \end{pmatrix}$$

## (0 pts.) 9. Naïve Bayes

Your task is to implement a verbose Naïve Bayes classifier. The goals for this task:

a. Self-assess/self-teach Python comfort – this is a very useful language in the data mining/science community

b. Gain experience implementing a simple probabilistic model, one which is quite handy for many tasks, including spam classification and sentiment analysis

Some implementation details:

- You should program in Python (v3). You are not allowed to make use of modules (aside from basic sqrt/pow/log functions available via `math`, and command-line arguments/exit in `sys`). The goal here is to write the code from scratch.
- You should only have to read the data file once, and should not need to store any of the rows (in their original form) to re-process. While the supplied data files are small, t'he intent here is to start thinking at scale, where memory/processing are not unlimited.
- You should use additive smoothing *only when necessary* (i.e. to avoid $P(x) = 0$) for discrete features, and assume a Gaussian for discrete features.

Below are sample runs to show what I expect as outputs (parameters are [data file] [target feature name] [alpha] [test feature 1] [test feature 2] ... [test feature n]). Notice you are first provided the NB model (all priors and likelihoods), then pertinent values to the test input (smoothed as appropriate), and then finally a classification. The first dataset is the classic weather example, and the shapes for the second are actually in the NB slides from class (in case a visualization helps).

```
$ ./solution.py data1.csv play 1 overcast 83 86 FALSE
P(yes;alpha=1) = 9 / 14 = 0.643
P(no;alpha=1) = 5 / 14 = 0.357
P(outlook=rainy|yes;alpha=1) = 3 / 9 = 0.333
P(outlook=overcast|yes;alpha=1) = 4 / 9 = 0.444
P(outlook=sunny|yes;alpha=1) = 2 / 9 = 0.222
P(outlook=rainy|no;alpha=1) = (2 + 1) / (5 + 1*3) = 0.375
P(outlook=overcast|no;alpha=1) = (0 + 1) / (5 + 1*3) = 0.125
P(outlook=sunny|no;alpha=1) = (3 + 1) / (5 + 1*3) = 0.500
P(temp|yes;alpha=1) = N(mean=73.000, sd=6.164)
P(temp|no;alpha=1) = N(mean=74.600, sd=7.893)
P(humidity|yes;alpha=1) = N(mean=79.111, sd=10.216)
P(humidity|no;alpha=1) = N(mean=86.200, sd=9.731)
P(windy=FALSE|yes;alpha=1) = 6 / 9 = 0.667
P(windy=TRUE|yes;alpha=1) = 3 / 9 = 0.333
P(windy=FALSE|no;alpha=1) = 2 / 5 = 0.400
P(windy=TRUE|no;alpha=1) = 3 / 5 = 0.600
Input: ['overcast', '83', '86', 'FALSE']
P(yes;alpha=1) = 9 / 14 = 0.643
P(outlook=overcast|yes;alpha=1) = 4 / 9 = 0.444
P(temp=83.000|yes;alpha=1) = N(mean=73.000, sd=6.164) = 1.736e-02
P(humidity=86.000|yes;alpha=1) = N(mean=79.111, sd=10.216) = 3.111e-02
P(windy=FALSE|yes;alpha=1) = 6 / 9 = 0.667
P(x|yes) = 1.600e-04
```

```
P(no;alpha=1) = 5 / 14 = 0.357
P(outlook=overcast|no;alpha=1) = (0 + 1) / (5 + 1*3) = 0.125
P(temp=83.000|no;alpha=1) = N(mean=74.600, sd=7.893) = 2.869e-02
P(humidity=86.000|no;alpha=1) = N(mean=86.200, sd=9.731) = 4.099e-02
P(windy=FALSE|no;alpha=1) = 2 / 5 = 0.400
P(x|no) = 5.880e-05
P(x) = 1.239e-04
P(yes|x) = 0.830
P(no|x) = 0.170
P(yes)P(x|yes) = 1.029e-04
P(no)P(x|no) = 2.100e-05
log(yes) = -3.988
log(no) = -4.678
argmax_Ck = yes

$ ./solution.py data2.csv sign 1 blue square
P(minus;alpha=1) = 5 / 12 = 0.417
P(plus;alpha=1) = 7 / 12 = 0.583
P(color=blue|minus;alpha=1) = 3 / 5 = 0.600
P(color=black|minus;alpha=1) = 1 / 5 = 0.200
P(color=red|minus;alpha=1) = 1 / 5 = 0.200
P(color=blue|plus;alpha=1) = 3 / 7 = 0.429
P(color=black|plus;alpha=1) = 2 / 7 = 0.286
P(color=red|plus;alpha=1) = 2 / 7 = 0.286
P(shape=circle|minus;alpha=1) = 2 / 5 = 0.400
P(shape=square|minus;alpha=1) = 3 / 5 = 0.600
P(shape=circle|plus;alpha=1) = 2 / 7 = 0.286
P(shape=square|plus;alpha=1) = 5 / 7 = 0.714
Input: ['blue', 'square']
P(minus;alpha=1) = 5 / 12 = 0.417
P(color=blue|minus;alpha=1) = 3 / 5 = 0.600
P(shape=square|minus;alpha=1) = 3 / 5 = 0.600
P(x|minus) = 3.600e-01
P(plus;alpha=1) = 7 / 12 = 0.583
P(color=blue|plus;alpha=1) = 3 / 7 = 0.429
P(shape=square|plus;alpha=1) = 5 / 7 = 0.714
P(x|plus) = 3.061e-01
P(x) = 3.286e-01
P(minus|x) = 0.457
P(plus|x) = 0.543
P(minus)P(x|minus) = 1.500e-01
P(plus)P(x|plus) = 1.786e-01
log(minus) = -0.824
log(plus) = -0.748
argmax_Ck = plus

$ ./solution.py data2.csv sign 1 orange square
P(minus;alpha=1) = 5 / 12 = 0.417
P(plus;alpha=1) = 7 / 12 = 0.583
P(color=red|minus;alpha=1) = 1 / 5 = 0.200
```

```
P(color=black|minus;alpha=1) = 1 / 5 = 0.200
P(color=blue|minus;alpha=1) = 3 / 5 = 0.600
P(color=red|plus;alpha=1) = 2 / 7 = 0.286
P(color=black|plus;alpha=1) = 2 / 7 = 0.286
P(color=blue|plus;alpha=1) = 3 / 7 = 0.429
P(shape=square|minus;alpha=1) = 3 / 5 = 0.600
P(shape=circle|minus;alpha=1) = 2 / 5 = 0.400
P(shape=square|plus;alpha=1) = 5 / 7 = 0.714
P(shape=circle|plus;alpha=1) = 2 / 7 = 0.286
Input: ['orange', 'square']
P(minus;alpha=1) = 5 / 12 = 0.417
P(color=orange|minus;alpha=1) = (0 + 1) / (5 + 1*4) = 0.111
P(shape=square|minus;alpha=1) = 3 / 5 = 0.600
P(x|minus) = 6.667e-02
P(plus;alpha=1) = 7 / 12 = 0.583
P(color=orange|plus;alpha=1) = (0 + 1) / (7 + 1*4) = 0.091
P(shape=square|plus;alpha=1) = 5 / 7 = 0.714
P(x|plus) = 6.494e-02
P(x) = 6.566e-02
P(minus|x) = 0.423
P(plus|x) = 0.577
P(minus)P(x|minus) = 2.778e-02
P(plus)P(x|plus) = 3.788e-02
log(minus) = -1.556
log(plus) = -1.422
argmax_Ck = plus
```