Kend'or Wilson, Matthew Merritt
2/13/2024

1. A random variable is discrete if its support is countable and there exists an associated probability density function (pdf).

True

2. Probability mass functions have a lower bound of 0 and an upper bound of 1.

True

3. The set of all possible realizations of a random variable is called its probability density.

False

4. The expected value of a discrete random variable is always part of its support, that is, $\mathbb{E}[X] \in R_X$.

False

5. Continuous random variables are functions which map points from the sample space to the real numbers.

True

6. We can formulate most parametric Bayesian models as a generative process, by which we first sample from the likelihood and then use the synthetic data point to sample from the prior.

False

7. The Bayesian posterior $p(\theta \mid y)$ for continuous parameter vectors $\theta \in \mathbb{R}^D$ is just another density function. That means, its integral $\int p(\theta \mid Y = y)\, d\theta \neq 1$ for some y.

False

8. Each realization of a continuous random variable has a probability of zero.

True

Problem 2

```
. Explain the differences between the following git commands
(a) git restore
(b) git checkout
(c) git reset
(d) git revert
in terms of undoing changes to a repository by providing a minimal (actual
or a synthetic) example.

* the exmamples providecd below represent only a local repository.


(a): git restore
- depending on the argument, it does different things, but in both cases
it changes/restores a location's contents to be equal to the contents at
another locaiton.
- In the case that ```git restore [--worktree] <file>``` was used; the
<file> equivalent in your working directory will now have the same
contents as the respective <file> in your staging area.
- In the case that ```git restore --staged <file>``` was used; the <file>
present in your staging area will now have the same contents as the
respective <file> present in the head commit.
Example:
```
staging area:
<file> : <file contents v1>
working area:
<file> : <file contents v2>

the following is the case after the user runs: git restore [--worktree]
<file>

staging area:
<file> : <file contents v1>
working area:
<file> : <file contents v1>
```

(b): git checkout <branch>
```

- Switches the current user to the <branch>, and updates the files in the staging area and working tree for that branch. This command also points the HEAD at the selected branch. Local modifications to files in the working directory are kepted (sometimes with the ability to be stashed) / with the ability to commit them to the selected branch.

```
branch 1:
staging area:
<file> : <file contents v1>
working area:
<file> : <file contents v2>

users current branch:
staging area:

working area:


the following is the case after the user runs: git checkout <branch 1>

branch 1:
staging area:
<file> : <file contents v1>
working area:
<file> : <file contents v2>

users current branch:
staging area:
<file> : <file contents v1>
working area:
<file> : <file contents v2>
```

(c) git reset <git commit hash> <flag>:

- removes all changes for all local changes to the staging area and working directory for the specified commit, in addition to moving both the HEAD and branch refs to the specified commit (what it removes depends on the flag you use e.g. soft, mixed, hard)

```
branch 1:
```

```
staging area:
<file> : <create with file contents v1>
working area:
<file> : <modifications to file>

the following is the case after the user runs: git reset --hard

staging area:

working area:

```

(d) git revert <commit ref>
- will inverse the changes from the specified <commit ref>, and create a new "revert commit", therefore preseving history.

```
branch 1:
* assuming user made the following commits:
commit 1: create <file> with <contents = 'a'>
commit 2: modify <file> by appending 'b' <file contents = 'ab'>
commit 3: modify <file> by appending 'c' <file contents = 'abc'>

the following is the case after the user runs: git revert HEAD

the resulting branch history will look like:
commit 1: create <file> with <contents = 'a'>
commit 2: modify <file> by appending 'b' <file contents = 'ab'>
commit 3: modify <file> by appending 'c' <file contents = 'abc'>
commit 4: revert <commit 3's message> <file contents = 'ab'>


```

Problem 3:

> Suppose that you want to invest some money in the oil market. You believe that the probability of the market going up is 0.8 and the market going down is 0.2. Further, if the market goes up, oil prices will increase by 1%, if it goes down, oil prices will drop by 10%. What is your expectation? Would you invest in this market? Assuming the increase/drop in prices is fixed, what is the minimal probability of prices going up in order for you to invest rationally (according to expectation) in this market? Discuss a limitation of expectations when making single-shot, real-life decisions.

The expected value of the drop in price is a decrease of 1.2%. It would be wise to not invest in this market, as our calculation reflects a reasonable cost-benefit analysis. It would be only 1% more expensive to buy in to the oil market if we wait and witness positive market changes, while we will get a 10% discount on stocks if outcomes are negative. As the mantra goes, buy low and sell high.

Assuming the price fluctuation is fixed we would need to see a pretty high probability of the price going up to make measly 1% gains worth it. By solving for x, (shown below) we can see that the minimum probability to have a positive price change is at least $\frac{10}{11}$ or 91% minimum odds to get an expected value that shows positive change.

One of the core problems in using expectancy theory to make real life decisions in how it limits scope to a narrow range of factors and fails to account for the individual making the decision, and their own unique needs. To be more blunt, in this specific example 1% gains are more valuable depending on interest rates and other outside factors, not accounted for in our decision making process.

Problem 4: Expectations II (4 points)

1. Show the following identity for the variance of a random variable X:
Var [X] = E X2 − E [X] ^2

$$Var[x] = E\left[x^2\right] - E[x]$$

We can show this identity by looking at the defintion for variance:

$$Var[x] = E\left[\left(x - E[x]^2\right)\right]$$

$$= E\left[x^2 - 2E[x]x + E(x)^2\right]$$

$$= E\left[x^2\right] - E[2E(x)x] + E\left[E(x)^2\right]$$

$$= E\left[x^2\right] - 2E(x)E[x] + E(x)^2$$

$$= E\left[x^2\right] - 2E(x)^2 + E(x)^2$$

$$= E\left[x^2\right] - E(x)^2$$

$$Var[x] = E\left[x^2\right] - E[x]$$

Law of Linearity
Move out constants
Since E[X] is a constant
it can be moved out

Therefore, we have shown the previous statement
true by proving it's identify

2. Show the following property for the variance of a random variable X and a scalar α: Var [αX + β] = α 2Var [X]

$$Var[\alpha X + \beta] = Var[\alpha X + \beta] = \alpha^2 Var[X]$$

$$= Var[\alpha X] + Var[\beta]$$

$$= Var[\alpha X] + 0$$

$$= \alpha^2 Var[X]$$

$$Var[\alpha X + \beta] = \alpha^2 Var[X] = \alpha^2 Var[X]$$

Law of Linearity
Varience of a constant = 0
Moving constant times random variable out of varience requires squaring

3. Assume you are given a random variable X with a standard normal distribution (mean zero, variance one). We can write this as X ~ Normal(μ = 0, σ = 1) One way to sample from this distribution is using NumPy's function numpy.random.randn. What transformation do you need to apply to the sampled values (i.e., the outputs of the function) such that they are now distributed according to X̃ ~ Normal(μ = 3, σ = 5)?

In attempting to get an $\tilde{X}$ with the given distribution, the mean would have to be an addition (as it essentially acts as a shifting function. The standard deviation (square root of variance), would be a scalar to be multiplied against the random variable. Therefore this would be representative:

$$numpy.random.randn() * \sigma + \mu$$

Problem 5: Simple Bayesian Inference (4 points)

The inhabitants of an island tell the truth one third of the time. They lie with probability $2/3$. On an occasion, after one of them made a statement, you ask another "Was that statement true?" and he says "yes". What is the probability that the statement was indeed true?

We know that the independent probability of any island resident telling us the truth is $\frac{1}{3}$. Based on Bayes Theorem, we know that the probability of the original statement being true (posterior), is dependent on the probability of the second individual telling the truth if the original statement was true. Filling in the known information, we get this equation (a being the original statement, and b being the second statement):

$P(a|b) = \frac{P(b|a)*P(a)}{P(b)} = \frac{P(b|a)*1/3}{P(b)}$

Solving for $P(b|a)$ is actually rather simple so long as we examine the nature of the problem. If a is telling the truth then b would lie ⅔ times and say no, but since b said yes, then we know this event has a likelihood of ⅓.

$P(a|b) = \frac{P(b|a)*P(a)}{P(b)} = \frac{1/3*1/3}{P(b)}$

Then to solve for $P(b)$ we must perform this equation:

$P(b) = (P(b|a) * P(a)) + (P(b|!\,a) * P(!\,a)) = (1/9) + (4/9) = 5/9$

And then simplify:

$P(a|b) = \frac{P(b|a)*P(a)}{P(b)} = \frac{1/3*1/3}{5/9} = \frac{1/9}{5/9} = 9/45 = 1/5$

Using Bayesian Inferencing, we can tell that the probability of the original statement being true is 20%

Problem 7: Priors, Sensitivity, Specificity (6 points)

Let's revisit the task we disease problem we tackled during class. Imagine you are a medical researcher analyzing the effectiveness of a new diagnostic test for a rare disease **X**. This disease affects 1% of the population. The probability of a true positive (the test correctly identifies an individual as having the disease) is 95%. This is also known as the *sensitivity* of the test. The probability of a true negative (the test correctly identifies an individual as not having the disease) is 90%. This is also known as the *specificity* of the test.

We will now consider a question of sensitivity analysis (not to be confused with the sensitivity of a test): How would the posterior probability change if the prior, the sensitivity, or the specificity of the test were to test. Write a Python program which produces three pretty and annotated 2D graphs depicting

1. The posterior probability (Y-axis) of actually having the disease given a positive test as a function of the prior probability (X-axis), assuming fixed sensitivity and specificity.

2. The posterior probability (Y-axis) of actually having the disease given a positive test as a function of the test's sensitivity (X-axis), assuming fixed prior and specificity.

3. The posterior probability (Y-axis) of actually having the disease given a positive test as a function of the test's specificity (X-axis), assuming fixed prior and sensitivity.

Briefly discuss how the posterior changes as a function of each of the quantities.

In the first scenario, the probability forms a logarithmic curve, with the posterior probability rising rapidly as the infection rate of the disease rises before leveling off. In the second scenario the probability forms a linear curve, with the posteriors being a function of the sensitivity. When a function of specificity, growth is exponential. This reflects how each factor contributes to the Bayesian Inference equation used to calculate posterior probability, behaving like a rational

function. Graphs are in the Question 7 notebook of the repository.

Problem 9: AI-Assisted Programming (4 points)

Use ChatGPT or any other large language model (LLM) to generate a function called `multivariate_normal_density(x, mu, Sigma)` which returns the density of a D-dimensional vector `x` given a D-dimensional mean (location) vector `mu` and a $D \times D$-dimensional covariance matrix `Cov`. Compare the outputs of your function with those obtained using SciPy's `scipy.stats.multivariate_normal` for a few parameterizations including a spherical Gaussian (zero covariance, shared variance across dimensions), a diagonal Gaussian (zero covariance, different variance for each dimension), and a full-covariance Gaussian (non-zero covariance, different variance for each dimension). Describe briefly how the LLM performed.

The code produced by ChatGPT is in its own words, “a basic implementation to calculate the density of a multivariate normal distribution given the mean vector and covariance matrix.” While it fails to match its own prediction for its example input (Predicted Output: 0.09056540022763257), its actual output matches the results of the Scipy function (approximately 0.12030983).
For the Gaussian inputs, the simple function produced by ChatGPT would offer answers close to what the SciPy function produced, for the exact same inputs. (examples included in Question 9 notebook on repo). This outcome is not surprising, but interesting considering ChatGPT was unable to accurately predict the outcome of the function it wrote. In fact, when prompted to predict the outcome of the function it wrote given specific inputs, it opted instead to instruct me to use the function instead, fully reproducing it and explaining how it works, but with the comment on expected output removed this time.