ISTA 421/521 – Homework 4

Due: Monday, October 18, 8pm

15 pts total for Undergrads, 20 pts total for Grads

STUDENT NAME

Undergraduate / Graduate

Instructions

This assignment is shorter, in order to have it due before the midterm.

Exercise 2 requires you to fill out the small python script, details provided in the Exercise 2 description. All exercises in this homework requires written derivations, so you will submit a .pdf of your written answers. (You can use LaTeX or any other system (including handwritten; plots, of course, must be program-generated) as long as the final version is in PDF.)

NOTE: Problem 3 is required for Graduate students only; Undergraduates may complete this problem for extra credit equal to the point value.

As in previous homework, pytest "unit tests" are provided to help guide your progress.

You may work with others in the course on the homework. However, if you do, you **must** list he names of everyone you worked with, along with which problems you collaborated. Your final submissions of code and written answers **MUST ALL BE IN YOUR OWN CODE FORMULATION AND WORDS**; you cannot submit copies of the same work – doing so will be considered cheating.

(FCMA refers to the course text: Rogers and Girolami (2016), *A First Course in Machine Learning*, second edition. For general notes on using LaTeX to typeset math, see:
`http://en.wikibooks.org/wiki/LaTeX/Mathematics`)

1. [5 points] Adapted from **Exercise 3.5** FCMA p.134:

   If a random variable R has a beta density

   $$p(r) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} r^{\alpha-1}(1-r)^{\beta-1},$$

   derive an expression for the expected value of r, $\mathbb{E}_{p(r)}\{r\}$ (We made use of this expectation in Lectures 13 and 14 when describing the expected value of the posterior as we considered different priors combined with the likelihood of the data). You will need the following identity for the gamma function:

   $$\Gamma(n+1) = n\Gamma(n).$$

   Hint: Use the definition of the Beta function:

   $$\mathcal{B}(\alpha,\beta) = \int_{r=0}^{r=1} r^{\alpha-1}(1-r)^{\beta-1}\,\mathrm{d}r = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$$

   Solution.

2. [10 points]

   In this Exercise you will complete the implementation for calculating four types of values in the provided python script `code/bayes_coin_game.py`.

   The function `run_scenario` will compute the following four variable values: `r_prior`, `r_posterior`, `marginal_likelihood`, `probability_of_winning`. The top-level script will call `run_scenario` under the conditions of each of the three prior scenarios in FCML and that we discussed in lectures 13 and 14.

   You will need to fill out how `run_scenario` sets these variables by implementing each of the following four functions:

   1. The function `calculate_prior_density` calculates the prior density of r, the probability of the coin being heads as a function of the prior beliefs about the probability of heads (α) and tails (β).
   2. The function `calculate_posterior_density` calculates the posterior density of r after taking into account the observations (number of heads, `y_obs`, out of `n` total coin tosses).
   3. The function `calculate_marginal_likelihood` calculates the marginal likelihood of the data under the prior beliefs about the probability of heads and tails.
   4. The function `calculate_probability_of_winning` calculates the probability of winning the coin game given the prior beliefs and the observed data.

   In the case of `calculate_prior_density` and `calculate_posterior_density`, it is up to you whether you implement these functions as computing single scalar values for the density of a particular value of r, or whether you implement them as vectorized functions that simultaneously compute a vector of densities given a vector of r values. In either case, the final values for `r_prior` and `r_posterior` in `run_scenario` must be vectors (1-dimensional numpy arrays) that contain the prior and posterior densities (respectively) for each of the values of r in the variable `r_values` of `run_scenario`; these vectors are provided to `plot_densities` to plot the prior and posterior density distributions. See the comments in the code.

   You can test your implementations of `calculate_prior_density`, `calculate_posterior_density`, `calculate_marginal_likelihood` and `calculate_probability_of_winning` as soon as you finish each as long as you assign the corresponding variable (`r_prior`, `r_posterior`, `marginal_likelihood`, or `probability_of_winning`) in `run_scenario` – you don't need to implement all four at once to start testing (there is an individual unit test that checks each variable individually for each scenario).

For you implementations of `calculate_marginal_likelihood` and `calculate_probability_of_winning`, it is recommended that you first perform the computations in *log space* (i.e., computing log-probability). This means taking the log of the equation that computes the respective density, and then after computing the value in log space, take the exponential of the result to recover the probability.

`bayes_coin_game.py` imports the following functions and modules:

1. `gamma`: Computes the Gamma function.
2. `loggamma`: Computes the log version of the Gamma function. This can be used in the computation of the log probability implementation for `calculate_marginal_likelihood` and `calculate_probabiity_of_winning`.
3. `binom`: Computes the binomial coefficient, aka computing the number of *combinations* of N choose k. (NOTE: You don't need a corresponding "log" version of this function for log-probability computations in `calculate_marginal_likelihood` or `calculate_probability_of_winning`; instead, simply doing `numpy.log(binom(...))` will be sufficient).
4. `numpy` module: I'll point out that this provides `numpy.log` and `numpy.exp`.

You should not import any additional functions or modules for your implementation.

Running `bayes_coin_game.py` will call the `run_scenario` for each of the three prior scenarios, and this in turn will generate for each scenario a plot of the prior and posterior densities for r. In your written solution, include these three plots; describe what the priors represent, and explain the differences between the priors and posteriors (why do they have the shapes they do). Also explain what makes the posteriors between the three scenarios not the same.

Solution.

3. [5 points; **Required only for Graduates**] Adapted from **Exercise 3.12** of FCMA p.135:

When performing a Bayesian analysis of the Olympics data, we assumed that σ^2 was known. If instead we assume that $\mathbf{w}$ is known and an Inverse Gamma prior is placed on σ^2,

$$p(\sigma^2|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}(\sigma^2)^{-\alpha-1}\exp\left\{-\frac{\beta}{\sigma^2}\right\},$$

then the posterior over σ^2 will also be Inverse Gamma. Derive the parameters for the posterior belief in the variance.

Solution.