Example sheet 3 hint sheet

Model crafting

Foundations of Data Science—DJW—2018/2019

Questions 1–3 test your creativity at inventing features. Questions 5–8 test if you can reason about the linear algebra of features, especially confounding. Questions 4 and 9 test if you understand the probabilistic model behind linear regression. Make sure you can answer questions 1, 6, and 4.

Question 1. For the stop-and-search data in section 4.1.2 of lecture notes, we proposed a model

$$\mathbb{P}(Y_i = \mathsf{find}) = \frac{e^{\xi_i}}{1+e^{\xi_i}} \quad \text{where} \quad \xi_i = \alpha + \beta_{e_i} + \gamma_{g_i}$$

where e_i is the ethnicity of suspect i, g_i is the gender, and $Y_i \in \{\mathsf{find}, \mathsf{nothing}\}$ is the outcome of the search. Rewrite the equation for ξ as a linear model, using one-hot coding.

The question is asking you for a linear model for ξ, i.e. something of the form

$$\xi \approx \eta_1 e_1 + \eta_2 e_2 + \cdots + \eta_K e_K.$$

Ignore the probability equation—it's a red herring. Use the one-hot coding trick from page 64 of lecture notes.

Question 2. For the climate data from section 5.2.2 of lecture notes, we proposed the model

$$\mathsf{temp} \approx \alpha + \beta_1 \sin(2\pi \mathsf{t}) + \beta_2 \cos(2\pi \mathsf{t}) + \gamma \mathsf{t} \tag{1}$$

in which the $+\gamma\mathsf{t}$ term asserts that temperatures are increasing at a constant rate. We might suspect though that temperatures are increasing non-linearly, as discussed in section 5.2.3. To test this, we can create a non-numerical feature out of t by

$$\mathsf{u} = \text{'decade\_'} + \mathsf{str}(\mathsf{math.floor}(\mathsf{t}/10)) + \text{'0s'}$$

(which gives us values like 'decade_1980s', 'decade_1990s', etc.) and fit the model

$$\mathsf{temp} \approx \alpha + \beta_1 \sin(2\pi \mathsf{t}) + \beta_2 \cos(2\pi \mathsf{t}) + \gamma_{\mathsf{u}}.$$

Write this as a linear model, and give pseudocode to fit it. *[You should explain what the feature vectors are, then give a one-line command to estimate the parameters.]*

What are the advantages and disadvantages of this model, as opposed to fitting (1) separately for each decade?

You have to write

$$\mathit{temp} \approx \alpha + \beta_1 \sin(2\pi t) + \beta_2 \cos(2\pi t) + \gamma_u.$$

as a linear model. The first three terms are fine; the last term γ_u needs work. Use the step function trick from page 66 of lecture notes.

One wrinkle, for extra credit: are your feature vectors linearly independent i.e. are your parameters identifiable?

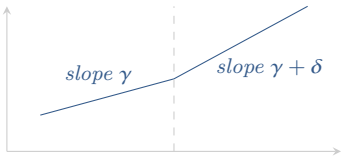
For the pseudocode: I want to see the Python command LinearRegression().fit, and I want to see that you can specify appropriate feature vectors and response vector for the fit command. If the features in your linear model weren't properly explained, this is another chance to get the marks.

Disadvantages of a separate fit per decade: You should invent a scenario in which this model doesn't fit well over the entire dataset. In Lecture 11, a student proposed just such a scenario, based on his belief about the effects of climate change.

Advantages of a separate fit per decade: As a thought experiment, imagine fitting the model (1) separately for each month, or for each quarter. What goes wrong with the fitting procedure if you choose the time bins too small? The same problem could go wrong with a per-decade fit, if there are gaps in the data (as indeed there are, further in the past).

Question 3. As an alternative to the climate model (1), we might suspect that temperatures are increasing linearly up to 1980, and that they are increasing linearly at a different rate from 1980 onwards. Devise a linear model to express this.

This is another question that can be answered with one-hot coding. We want to express "two straight lines" as a linear model. For inspiration, look at page 11 of lecture notes.



Question 4. This question is about inference for the linear regression model

$$\mathsf{temp} = \alpha + \beta_1 \sin(2\pi \mathsf{t}) + \beta_2 \cos(2\pi \mathsf{t}) + \gamma \mathsf{t} + \text{Normal}(0, \sigma^2).$$

(a) Give pseudocode to find the maximum likelihood estimators $\hat{\alpha}$, $\hat{\beta}_1$, $\hat{\beta}_2$, $\hat{\gamma}$, and $\hat{\sigma}$.

(b) What is meant by *parametric resampling*? Explain how to use parametric resampling to synthesize a new version of the climate dataset.

(c) Consider the confidence interval $\gamma \in \hat{\gamma} \pm 0.1$. Explain how to use bootstrap resampling to find the error probability of this confidence interval.

(d) Give a brief outline of how to find a 95% Bayesian confidence interval for γ.

This question goes to the heart of everything in this course! For part (a), explain how to use sklearn.linear_model.LinearRegression *to fit the linear model, and make sure you specify the feature vectors. You can either derive the formula for $\hat{\sigma}$, or just remember it.*

For part (b), look at slides from lecture 11, where resampling is explained more thoroughly than in lecture notes.

For part (c), look back at your answer to example sheet 2 question 4.

For part (d), you don't need to give any formulae. You need to describe (i) what is meant by a prior distribution, (ii) how to calculate the posterior distribution, (iii) marginalizing out the nuisance parameters, (iv) how to obtain a confidence interval. You can pretend that the calculations are all tractable; or you can suggest a Monte Carlo method for steps (iii) and (iv).

Question 5. Here are two different models for the climate data:

$$\mathsf{temp} \approx \alpha + \beta_1 \sin(2\pi \mathsf{t}) + \beta_2 \cos(2\pi \mathsf{t}) + \gamma \mathsf{t}$$

and

$$\mathsf{temp} \approx \alpha + \beta_1 \sin(2\pi \mathsf{t}) + \beta_2 \cos(2\pi \mathsf{t}) + \gamma(\mathsf{t} - 2000).$$

The first model produces a fitted value $\alpha = -63.9$°C and a 95% confidence interval $[-96.5, -34.7]$°C. The second model produces a fitted value $\alpha = 10.5$°C and a 95% confidence interval $[10.4, 10.7]$°C. Why the difference? Why is the confidence interval much smaller in the second case? Which is correct?

Look at lecture notes page 54, where we thought about reparameterizing a model—two different ways to write a model, with different parameters, but amounting to the same thing. In your answer to this question you don't need to go into detail about the exact calculation of the confidence interval; it's enough to explain what the α parameter is measuring in each case. Read lecture notes page 65 for the full hint.

Question 6. In your answer to question 1, are your feature vectors linearly independent? Justify your answer. If not, rewrite the model in terms of a linearly independent set of feature vectors.

A similar question is covered in the slides for lecture 12 (pen and paper answer) and on page 77 of lecture notes (code answer).

There are two ways to turn a non-identifiable model into one with linearly independent features: (i) use common sense, as we did in lecture notes page 54, and think hard about whether there are simple additions and subtractions that leave the model invariant; (ii) use numpy *and* matrix_rank*, and repeatedly remove the problematic feature vectors until what you have left has full rank.*

Question 7. Let $(F_1, F_2, F_3, \dots) = (1, 1, 2, 3, \dots)$ be the Fibonacci numbers, $F_n = F_{n-1} + F_{n-2}$. Define the vectors f, f_1, f_2, and f_3 by

$$\begin{aligned} f &= [F_4, F_5, F_6, \dots, F_{m+3}] \\ f_1 &= [F_3, F_4, F_5, \dots, F_{m+2}] \\ f_2 &= [F_2, F_3, F_4, \dots, F_{m+1}] \\ f_3 &= [F_1, F_2, F_3, \dots, F_m] \end{aligned}$$

for some large value of m. If you were to fit the linear model

$$f \approx \alpha + \beta_1 f_1 + \beta_2 f_2$$

what parameters would you expect? What about the linear model

$$f \approx \alpha + \beta_1 f_1 + \beta_2 f_2 + \beta_3 f_3?$$

[Hint. Are the feature vectors linearly independent?]

For the first model: write out the feature vectors f_1 and f_2 and show that they're linearly independent, as in the slides from lecture 12 (or you can use Python as in page 77 of lecture notes).

For the second model: what does the Fibonacci formula tell you about a linear relationship between these three feature vectors?

If you do use Python, watch out for integer overflow for large m.

Question 8. Three chess players play each other. In a tournament, A won 7 matches against B and lost 3, A won 9 matches against C and lost 1, and B won 6 matches against C and lost 4. We wish to ascribe a skill level to each player, such that the higher the skill difference the more likely it is that the higher-skilled player wins a match. Let μ_A, μ_B, and μ_C be skill levels, and consider this model: if match i is between players $p1(i)$ and $p2(i)$ then the probability that $p1(i)$ wins is $e^{\xi_i}/(1 + e^{\xi_i})$ where $\xi_i = \mu_{p1(i)} - \mu_{p2(i)}$.

(a) Find the log likelihood of (μ_A, μ_B, μ_C)

(b) Show that these parameters are not identifiable, and give an equivalent 'reduced' parameterization that is identifiable.

[Hint. This is like question 6.] In IA Algorithms we learnt the topological sort algorithm, which puts items in order given a set of pairwise comparisons. That algorithm only works with perfect non-noisy data, whereas machine learning models like this chess skill model can cope with noise.

Part (a) is actually just another logistic regression, like page 57 of lecture notes. The question doesn't ask you to simplify, so you can just leave it as a messy sum as in lines 21–25 of the code on page 57.

If you do want to simplify your answer, it's a useful exercise to write out the log likelihood for a simpler logistic regression first. Suppose we toss 10 coins and get 8 heads. Let p be the probability of heads. We know from page 1 of lecture notes that the log likelihood is

$$\log \text{lik}(p) = \textit{const} + 8 \log p + 2 \log(1 - p).$$

and the maximum likelihood is $\hat{p} = 8/10$. Now suppose we have the same data, but laid out long-form in a dataset, one row per coin toss, and we fit the model $\mathbb{P}(Y_i = \textit{heads}) = e^{\xi}/(1 + e^{\xi})$. Write out a

messy sum for the full log likelihood. Can you simplify it, and end up with the same answer as before, $e^{\hat{\xi}}/(1+e^{\hat{\xi}}) = 8/10$? If you can get the algebra right for a dataset of coin tosses, you should be able to simplify your answer for part (a) to obtain

$$\log \text{lik}(\mu_A, \mu_B, \mu_C \mid y_1, \ldots, y_n) = \\ 7\xi_{AB} - 10\log(1-\xi_{AB}) + 9\xi_{AC} - 10\log(1-\xi_{AC}) + 6\xi_{BC} - 10\log(1-\xi_{BC})$$

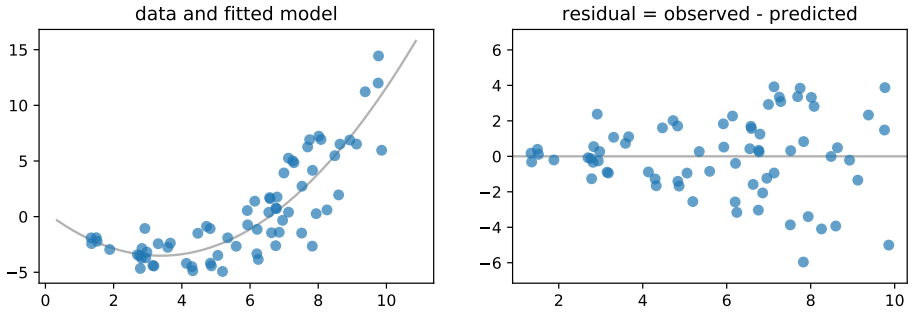
where $\xi_{AB} = \mu_A - \mu_B$ etc.

For part (b): answer as in page 54 of lecture notes.

Question 9. We are given a dataset (`https://teachingfiles.blob.core.windows.net/founds/ex3q9.csv`) with covariate x and response variable y and we fit the linear model

$$y_i \approx \alpha + \beta x_i + \gamma x_i^2.$$

After fitting the model using the least squares estimation, we plot the residuals $\varepsilon_i = y_i - (\hat{\alpha} + \hat{\beta} x_i + \hat{\gamma} x_i^2)$.



(a) Describe what you would expect to see in the residual plot, if the assumptions behind linear regression are correct.

(b) This residual plot suggests that perhaps $\varepsilon_i \sim \text{Normal}\big(0, (\sigma x_i)^2\big)$ where σ is an unknown parameter. Assuming this is the case, give pseudocode to find the maximum likelihood estimators for α, β, and γ.

For part (a): equation (25) on page 76 of lecture notes says that the residuals in a linear regression should all be independent $\text{Normal}(0, \sigma^2)$ random variables.

(There is a very technical caveat to this, which you might come across if you read hard-core statistics texts. If the model is true and if α, β, γ are the true parameter values, which we don't know, then the values $y_i - (\alpha + \beta x_i + \gamma x_i^2)$ are independent $\text{Normal}(0, \sigma^2)$ random variables. This isn't precisely the same as saying that the computed residuals $y_i - (\hat{\alpha} + \hat{\beta} x_i + \hat{\gamma} x_i^2)$ are independent $\text{Normal}(0, \sigma^2)$ random variables. If you're mathematical enough to be bothered by this, you don't need this hint sheet!)

For part (b): just write out the log likelihood and solve it, similar to page 76 of lecture notes. Then rearrange your answer so that the straightforward least-squares `sklearn.linear_model.LinearRegression()` fitting command can be used.