

MATH 3190 Homework 5

Focus: Notes 7 Part 2

Due March 16, 2024

Your homework should be completed in R Markdown or Quarto and Knitted to an html or pdf document. You will “turn in” this homework by uploading to your GitHub Math_3190_Assignment repository in the Homework directory.

Problem 1 (20 points)

Suppose $\mathbf{x} = (x_1, \dots, x_n)^T$ are independent and identically distributed random variables with probability density function (pdf) given by

$$f(x_i|\theta) = \theta(1 - x_i)^{\theta-1}; \quad 0 \leq x_i \leq 1, \quad 0 < \theta < \infty, \quad i = 1, \dots, n$$

Part a (4 points)

Using ggplot, plot the pdf for an individual x_i given $\theta = 0.5$ and then again for $\theta = 5$.

Part b (5 points)

Give the likelihood $L(\theta|\mathbf{x})$ and log-likelihood $\ell(\theta|\mathbf{x})$ functions in this situation.

Part c (4 points)

Find the Maximum Likelihood Estimator (MLE) $\hat{\theta}$ for θ .

Part d (2 points)

Show that your estimator is in fact a maximum. That is, check the second derivative of the log-likelihood.

Part e (1 point)

Find the MLE if the data values are given below. Note, the actual θ value used to generate these data was $\theta = 7.3$.

```
x <- c(0.0194, 0.0053, 0.2488, 0.0456, 0.2059, 0.0992, 0.1168, 0.3705,  
       0.2129, 0.018, 0.0464, 0.1401, 0.0759, 0.1588, 0.0334, 0.2931,  
       0.0662, 0.2292, 0.1581, 0.3462, 0.035, 0.1086, 0.0793, 0.2095,  
       0.1419, 0.1835, 0.1107, 0.0764, 0.1331, 0.042, 0.0911, 0.1608)
```

Part f (4 points)

To get an idea of how variable the maximum likelihood estimator is in this case, let's generate many samples of size 32 from this distribution, which is a Beta distribution with parameters 1 and θ . We can do this by using `rbeta(32, 1, 7.3)`.

Generate at least 10,000 samples this way, compute the MLE for each, and then plot a histogram of the values using ggplot. On the plot somewhere, indicate what the standard deviation in those estimates is.

Now repeat this but instead of taking samples of size 32, take samples of size 100. How does this change the histogram and standard deviation of the estimates?

Problem 2 (35 points)

In the **AER** package is the data set “ShipAccidents”. You can install that package and load that data using `data(ShipAccidents)`. Type `?ShipAccidents` to read about that data set.

Part a (3 points)

Load in the data, remove the rows for which service is equal to 0, and then fit a Poisson regression model for predicting incidents from all other variables using the ShipAccidents data. Briefly explain why it makes sense to do Poisson regression here.

Part b (4 points)

Using that model, interpret the slope of “service” and the slope of “construction1970-74” in original (not log) units.

Part c (5 points)

Make a residual plot of the Pearson residuals vs the linear predictors and make a QQ plot of the Pearson residuals. Comment on what these imply about the fit.

Part d (4 points)

Conduct a deviance goodness-of-fit (GOF) test for a lack of fit here. Type out the null and alternative hypotheses, report the test statistic, give the p-value, and provide an interpretation.

Part e (2 points)

Is there evidence that this model has overdispersion? Explain.

Part f (6 points)

Fit a “quasipoisson” model to these data. Using this model, obtain an approximate 95% confidence interval for the mean response for a ship of type “B” with construction between 1965 and 1969, with operation between 1960 and 1974, and with 2000 aggregate months of service. You can construct this interval using a t critical value with 24 degrees of freedom since the quasipoisson family more closely follows a t distribution than a normal one. Interpret the interval.

Part g (5 points)

Now construct an approximate 95% prediction interval for the number of incidents for the ship described in part f using the `qpois()` function. Interpret the interval.

Part h (6 points)

Suppose we did not catch the fact that this model is overdispersed. Repeat parts f and g using the model fit with the “poisson” family, not Poisson and using a z interval for the mean response rather than a t interval. Compare your results to parts f and g and comment on what is different.

Problem 3 (45 points)

In Homework 4, we briefly looked at a small sample from this data set. Now we have the full data set with 20,640 districts. For districts in California from 1990, we want to predict the median house value for the district based on the district location (longitude and latitude), the median house age in the block group, the total number of rooms in the homes, the total number of bedrooms, the population in the district, the number of households, and the median income (in \$10,000). These data were obtained from scikit-learn.org and can be found in the Data folder of the Math3190_Sp24 GitHub repo.

Part a (3 points)

Read in the data, assign appropriate column names to the columns and then take logs of all variables except lat and long. Once this is done, split the data into training and testing sets with the testing containing 20% of the values. Set a seed when you do this.

Part b (3 points)

Use the following command to view pair-wise scatter plots for the data here. Change `housing_train` to whatever you called the training set and be sure to change `eval = F` to `eval = T` in the option for the code chunk. Note, that `lower` option changes the points to be plotted with periods instead of circles. This speeds up plotting time considerably and makes the plots more readable since there are many data points here.

Based on these plots, does multicollinearity appear to be an issue here?

```
library(GGally)
ggpairs(housing_train,
        lower = list(continuous = wrap("points", shape = "."))
)
```

Part c (3 points)

Fit a OLS regression model predicting house prices from all other variables. Check its summary output and its VIF values. Comment on the VIFs.

Part d (3 points)

Fit a principal component model using the `pcr()` function in the `pls` package. Since the variables are on very different scales, use the `scale = TRUE` option in the function. Then find the VIF values for this model. The `vif()` function from the `car` library won't work here. Instead, take the diagonals of the inverse of the correlation matrix (like we did in the regularization section of Notes 7) for the `scores` output.

Part e (5 points)

Now take the summary of your principal component model. With PCA, the common amount of variation we want to explain in the predictors is 90% or more. How many components are needed to achieve this? For that number of components, how much of the variation in log of home values is explained? How many components are needed to explain a “good” amount of the variation in the log of home values? Note: the upper bound on the amount of variation in the log of home values here with all 8 components will be equal to the R^2 value of the OLS model.

Part f (3 points)

Now fit a partial least squares model using the `plsr()` function in the `pls` package. Then find the VIF values for this model like you did in part d.

Part g (4 points)

Now take the summary of your partial least squares model. Compare this output to the summary of the PCA model. Explain what the differences are.

How many components are needed to balance a good amount of variation explained in X and in y for the PLS model?

Part h (5 points)

In part e, you said how many components were needed for the PCA. Using that number of components, use the PCA and PLS models to find the root MSE of the model when predicting the testing set. Note: the root MSE (or residual standard error, also abbreviated RMSE) is found by taking the square root of the sum of squared residuals divided by the residual degrees of freedom, which is $n - p$. Also, be aware that when you use the `predict()` function on your model, it will give you predictions for each number of components. You can access the predictions for using 3 components, for example, by typing `predict(pca_model, test_set)[,3]`.

Then in part g, you said how many components were needed for the PLS. Using that number of components, use the PCA and PLS models to find the root MSE of the model when predicting the testing set.

Compare the results of the predictions in both cases.

Part i (10 points)

Using the number of components you said were needed for the PLS model in part g, come up with reasonable surrogates for each of those components. Look at the projection output for this. Make sure to center and scale each variable. You can do this with the `scale()` function.

Then fit a OLS model using those surrogates in the training set. Check the VIF of this model to make sure each one is below 5. If they are not, your surrogates should be changed.

Part j (6 points)

Now using that model you fit in the previous part with the surrogates, find the root MSE for predicting the testing set using that model, and compare it to what you got in part h. Describe some pros and cons of the surrogate model.

Hint: to predict using this model, you will have to create a `newdata` data frame (or tibble) and then redefine your surrogates in that data frame using the scaled testing data set. Make sure the variable names in the `newdata` data frame match the variables names used in the model you defined in part i.