EBA35001 Fall 2022, continuation

Take home exam

Jonas Moss

Introduction

1. Only show output that supports your argument. If you use Jupyter Notebooks, you may hide the output of a cell using a semi-colon `;`. We will deduct points from shoddily written reports plagued by noisy outputs.
2. Make your plots look nice. Add appropriate axis labels, legends and so on.
3. *"Brevity is the soul of wit."* Strive not to write too much. We prefer pithy to lengthy expositions.
4. The exercises are equally weighted. Every subexercise is equally scored, with a minimum score of 0 and a maximum of 2. Since there are 30 subexercises in total, you can get a maximum of 60 points.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels as sm
import statsmodels.formula.api as smf
```

1 Binary regression

We're using the Wine quality data set in this exercise. Take a look at the page for more information.

(a)

(i)

Import the data set `winequalityN.csv` as `wine`. We want to use the data to deduce if a wine is acceptable, with acceptable defined as `quality > 5`. Define a column `acceptable` that contains `True`s when `quality > 5` and `False`s otherwise. We won't use `quality` anymore, so drop it from the table. Finally, drop all rows containing `NA`s.

(ii)

Make a `pairplot` of the data. Take note of any patterns. (*Hint*: If `pairplot` is slow, you can use the `sample` method to reduce the data strain.)

(iii)

Display the correlation matrix between the numerical values in the data. Then find the columns where the difference between the correlations for the bad wines and good wines is greater than 0.15, along with both correlations. (*Hint:* You may need to iterate over all the column names and use a `set`.)

(b)

(i)

Run a logistic regression model on all covariates with `acceptable` as response variable. Make a `significant` array containing all covariates that are significant at the 0.05 level (along with their p-values) and a `not_significant` array containing the rest. Print those two arrays.

(ii)

Fit a new regression model with the non-significant covariates removed, but keeping the intercept. Which model do you prefer?

(iii)

Using the same two models as in (i) and (ii), change the link function from the logistic link to the Probit link, Cauchit link, and cloglog link. Report the AICs of the models in a table like this:

	Logistic	Probit	Cauchit	Cloglog
Model 1				
Model 2				

(*Hint*: You need to take a good look at the documentation of the `glm` function of statsmodels. Also see the lecture notes.)

(c)

(i)

The alcohol covariate appears to have a stronger influence than the other covariates. Fit three models using: (a) `log(alcohol + 1)`, (b) `alcohol**2`, (c) `log(alcohol + 1) + alcohol*^**2`, and report their AICs. Is any of the new models performing better than the others?

(ii)

Fit at least five additional models and report their results in a table containing the formula and the resulting AIC, plus potentially more information. Pick your favorite among these.

(iii)

Make a receiver operating characterstic curve for your favorite model. Explain what it means.

(iv)

Calculate the AUC for the models you tested in exercise (ii).

2 Linear regression

We use the student performance prediction, available in the `exams.csv` file. See the page for more information about this data set.

(a)

(i)

First import the exams file into the variable `exams`.

It is well-known that general intelligence encompasses both math skill and literary skill. Display the correlation matrix between math skill, literary skill, and writing skill.

(ii)

Are the correlations in the previous exercise individually significantly different from 0? You can use any valid method to figure this out, but you might want to use linear regression. Don't use the summary method to display the p-values, as it takes too much space. (*Hint:* Remember to use the `Q` function to access columns with spaces inside.)

(iii)

Find the optimal linear combination of `writing score` and `reading score` to predict `math score`. What is the correlation between `math score` and this optimal linear combination? Recall that a linear combination is on the form `a + b * writing score + c * reading score`.

(b)

(i)

Display the distinct categories in every column that contains only categorical values.

(ii)

Fit a regression model on `math score` using all covariates except `writing skill` and `reading skill`. Show its summary table. Should you report the adjusted R^2 or the ordinary R^2 for this model?

(iii)

From the output above it looks like the results are roughly linear in level of education of the parents. Add a new column (called `education numeric`) to the data where the level of education is numeric, i.e., `some high school` is mapped to 1, `high school` to 2, and so on. (*Hint*: Google `pandas replace`.) Run a linear regression using `education numeric` instead of `parental level of education`. Would you prefer to use this model or the last model?

(c)

(i)

Run the model `Q('writing score') ~ gender + Q('race/ethnicity')` and display its parameter estimates. What is the interpretation of `gender[T.male]` and `Q('race/ethnicity')[T.group E]`?

(ii)

Referring to the model in (i), we would like to check if it is possible to do a similar trick as we did for level of education, linearizing the categories. Plot the estimated values for `Q('race/ethnicity')` (in the appropriate order) against `[0, 1, 2, 3, 4]` and see if there is a pattern.

(iii)

Fit a suitable function to the data obtained in the previous exercise. Make a plot of the function and evaluate its fit to the data.

(iv)

Replace the categorical values of `race/ethnicity` with the predicted values in a column `race numeric` (use the values found in the previous exercise). Run the regression in (i) again, but with `race numeric` instead of `race/ethnicity`. Which model do you prefer?

3 Simulations

(a)

The central limit theorem states that $\sqrt{n}(\overline{X}_n - \mu)/\sigma \to N(0,1)$ when $X_1, X_2, \ldots, X_n$ are iid with mean μ and standard deviation σ, and the empirical mean is $\overline{X}_n = (\sum_{i=1}^n X_i)/n$ and $N(0,1)$ is the standard normal. It's often interesting to see how quick the convergence is; we'll explore that in this problem.

(i)

Make a function `simmean(n, model, n_reps = 10000)`. It simulates `n_reps` times n observations from the random number generator `model` and calculates its empirical mean and empirical standard deviation. Make a histogram (with densities, not counts) of the simulations when the model is the function `lambda dim: np.random.default_rng(seed = 313).exponential(1, dim)`.

(ii)

Make a function `simclt` that extends to function above with `mu` and `sigma`, and returns samples from $\sqrt{n}(\overline{X}_n - \mu)/\sigma$. Make a (density) histogram for the same input as above, and overlay a standard normal on top of the plot.

(iii)

Make similar plots for the Pareto distribution with parameter $b = 3$ (following the Scipy convention) for $n = 10, n = 100, n = 1000$, and extend the exponential analysis to $n = 10$ and $n = 1000$. Comment on the results. You need to figure out the mean and standard deviation for the Pareto yourself. (*Hint:* See the Scipy documentation and wikipedia. Observe that `Numpy` shifts the Pareto distribution towards `0`; see the Numpy docs for details.)

(b)

(i)

Make a function `simmax` that simulates n observations from a standard exponential distribution and finds the maximum of the observations. It must take an `rng` as input. Use it to simulate the maximum of $n = 1000$ exponentials when `rng = np.random.default_rng(seed = 313)`.

(ii)

Extend the function above to `simmaxs`, a function that take an `n_reps = 10,000` argument in addition to `n` and `rng`. It should return a Numpy array with `n_reps` independent simulations of the maximum. Make a histogram of "maxima-log(n)", where $n = 100$ and `n_reps = 10,000`. Make sure the histogram displays the density of the maxima, not the frequency count.

(iii)

Plot the standard Gumbel distribution on top of the histogram. It is part of Scipy, called `gumbel_r`. Make similar plots for $n = 1000$ and $n = 10000$. What do you see?

(c)

(i)

For a total of `n_reps = 10,000` times, draw $n = 100$ samples from the standard Cauchy distribution (`standard_t` with degrees of freedom equal to 1) and calculate the mean over these 100 values. Then make a histogram of its mean. Make sure the histogram shows a density, not the frequency of counts, give it `100` bins, and restrict it to the range `(-50, 50)`.

(ii)

Generalize the previous exercise by making a function taking `n` and `n_reps = 10000` as arguments, returning the simulated values. Make three histograms for `n=10`, `n=100` and `n=10,000`. What do you see?

(iii)

Add a curve for the standard Cauchy distribution to the histograms of the last exercise. What do you see? (*Hint:* Use `Scipy`.)

(iv)

Does the central limit theorem hold for a sequence of iid Cauchy random variables? Why or why not? Demonstrate using a suitable simulation.