COMP680_FinalExam_Fa23_firstname_lastname

June 23, 2024

1 Welcome to COMP 680

1.0.1 Statistics for Computing and Data Science

1.0.2 Take-home Final Exam Due in 4 Hours

2 Instruction

- This exam is a timed exam. You have **4 hours** to complet it.
- You may access any course material including **slides, demo, notes and homework**.
- You may access other related **books or reference material** you can find.
- You may also use the internet in a **passive way**. That is, you may search on the internet for material, but you **cannot** post a question and ask for help, including **ChatGPT or other AI tools** for help.
- If you do use something significant from the internet, please put the **link to the source** in your anwser. For example, you take a piece of code from a website other than the official documentation page.
- You **cannot** get help from anyone or discuss exam problems with anyone inside or outside the class.
- Feel free to include any Python modules you need besides the ones included already.
- Feel free to add more cells for codes and texts, and please make your code sufficiently commented and readable.

Finally, please pledge to the **Rice Honor Code** and print your name in lieu of a signature:

"On my honor, I have neither given nor received any unauthorized aid on this exam."

YOUR NAME:

```
[ ]: ### standard imports
     import numpy as np
     import pandas as pd
     import random

     random.seed(2023)

     import matplotlib.pyplot as plt
     %matplotlib inline
```

```
plt.style.use('fivethirtyeight')
plt.rcParams['patch.force_edgecolor'] = True
import seaborn as sns

import statsmodels.api as sm
import scipy.stats as st

import warnings
warnings.filterwarnings('ignore')
```

3 Part I: Inference

A mixture distribution is often used to model data that are believed to come from a mixture of K populations. For example, the distribution of heights in the general population could be considered to be a mixture of $K = 2$, Normal distributions, one each for males and females. In this problem, you will conduct a simulation study using the following mixture of two Normal distributions to generate your data:

$$f_X(x) = \omega\phi(x; \mu_1, \sigma_1^2) + (1 - \omega)\phi(x; \mu_2, \sigma_2^2)$$

where ω is the mixing parameter (a number between 0 and 1, interpretable as the proportion of individuals in the population that come from the first distribution), $\phi(\cdot)$ is the Normal distribution probability density function (pdf), and μ_1, σ_1 and μ_2, σ_2 are the mean and standard deviation of the first and the second normal distributions, respectively.

3.0.1 1.1 (5 pts)

Write a function `simu_mix_normal` to simulate data from this mixture distribution. Your function should have the following inputs:

- all the parameters of the mixture distribution: ω, μ_1, σ_1, μ_2, σ_2
- sample size n

You are allowed to call any distributions in `np.random`.

```
[ ]: ### enter your code here
     ...
```

3.0.2 1.2 (5 pts)

Use the function you wrote in **1.1** to generate a 1000 data points using the following parameter values: $\omega = 0.3$, $\mu_1 = 0$, $\sigma_1 = 1$, $\mu_2 = 10$, $\sigma_2 = 5$. Plot your simulated data in a histogram.

```
[ ]: ### enter your code here
     ...
```

3.0.3 1.3 (5 pts)

Use the following parameter values: $\omega = 0.5$, $\mu_1 = 0$, $\sigma_1 = 1$, $\mu_2 = 1$, $\sigma_2 = 1$, and generate 1000 datasets of size $n = 100$. For each simulated dataset, construct a 95% confidence interval (CI) for the sample mean. - assuming you only see data and you have no idea of what the population distribution actually is - justify how you construct the CI

```
[ ]: ### enter your code here
     ...
```

3.0.4 1.4 (5 pts)

In the case of **1.3**, assuming now you know what the population distribution is, calculate the population mean and save as `pop_mean`. Write code to calculate the proportion of your 1000 confidence intervals that actually contain `pop_mean`.

```
[ ]: ### enter your code here
     ...
```

```
[ ]: print('True population mean: ', ...)
     print('Proportion of CI that contains true pop mean: ', ...)
```

3.0.5 1.5 (5 pts)

Suppose you know the population distribution is a mixture of two normals, and your goal is to estimate the parameters associated with the population using Maximum Likelihood Estimate. Write a function to compute the **log likelihood function** given data. Your function should have the following inputs:

- all the parameters of the mixture distribution: ω, μ_1, σ_1, μ_2, σ_2
- data as a numpy array

```
[ ]: ### enter your code here
     ...
```

3.0.6 1.6 (5 pts)

Use the data you simulated in **1.2**, and suppose now you know the true normal parameters $\mu_1 = 0$, $\sigma_1 = 1$, $\mu_2 = 10$, $\sigma_2 = 5$ (the values in 1.2), but you don't know the weight ω. Use the log likelihood function you write in **1.5**, and plot the log likelihood as a function of ω. Use your function values to find the Maximun Likelihood Estimate $\hat{\omega}$ numerically.

```
[ ]: ### enter your code here
     ...
```

4 Part II: Simulation

You work for an auto insurance company as an actuary in the pricing department. Your boss asked you to conduct some sensitivity analysis for their current pricing model.

For simulation purpose, you will use a simplified version of a hierarchical model with three components:

- claim frequency: this is the expected claim count per unit of exposure for a particular policy.
- claim type: given that a claim has occurred, it is one of the following categories: insured's own damage, third party injury, and third party property damage.
- claim severity: each claim type has its own severity distribution.

4.0.1 2.1 (5 pts)

To model the frequency of claims, or expected number of claims per exposure, we will use Poisson distribution which is commonly used to model count data. However, the majority of policy holders do not have any claims, which means there are an inflated number of zeros if we look at the number of claims over a fixed time period. Therefore we will need what is known as zero-inflated Poisson distribution to model this.

A zero-inflated Poisson random variable X follows a distribution that has two parameters: π and λ, and its Probability Mass Function (PMF) is given as following for $k = 1, 2, \cdots$:

$$\mathbb{P}(X = 0) = \pi + (1 - \pi)e^{-\lambda}$$

$$\mathbb{P}(X = k) = (1 - \pi)\frac{\lambda^k e^{-\lambda}}{k!}$$

Write a function to simulate expected number of claims for a particular policy holder. Your function should take the following inputs: - exposure of the policy t in unit of calendar year (so for example, if it is a new policy just started a month ago, its exposure up to date should be $t = 1/12$) - π as the parameter in zero-inflated Poisson distribution to account for extra zeros - λ as the Poisson rate, however, the actual distribution that the number of claims this policy follows should have parameters π and $\lambda \cdot t$

```
[ ]: ### enter your code here
     ...
```

4.0.2 2.2 (5 pts)

Given that at least one claim has occurred for a particular policy holder, the number of different types of the claims follows a multinomial distribution. Write a function to simulate the expected number of **each type** of claims for a particular policy holder. Your function should take the following inputs: - total number of claims - parameters for the multinomial distribution

and return the number of the three different types of the claims in the following order: - insured's own damage - third party injury - third party property damage

If there is no claim, then your function should return 0 for all type of claims.

```
[ ]: ### enter your code here
     ...
```

4.0.3 2.3 (5 pts)

It is common to model insurance claim severity using a right-skewed, long-tailed distribution, such as lognormal, gamma, beta prime or pareto distributions. The long right tail of these distributions corresponds to large claim amounts, which are more important for insurance applications.

Write a function to simulate claim amount from lognormal distribution for each type of claim that has occurred. For simplicity, the pricing model assumes that if a policyholder has multiple claims, the amounts are independent. Your function should take the following inputs: - The number of claims that has occurred for each of the three types. - Mean and variance parameters for each lognormal that corresponds to the three type of claims

Your function can return the claim amounts in one array or list, i.e., no need to differentiate the 3 types of claims.

```
[ ]: ### enter your code here
     ...
```

4.0.4 2.4 (5 pts)

Use the functions you created above to simulate the expected amount of claims for the next year for a new customer. Assume exposure of new customers follows a uniform distribution between $[0, 1]$, that is new customers are equally likely to start a new policy over the calendar year. - in a more realistic model, each of the above distribution parameters will be a function of the covariates collected from new policy holders. That is why they always want your data before giving you a quote!

```
[ ]: ### enter your code here
     ...
```

4.0.5 2.5 (5 pts)

Now use your simulation to calculate an empirical probability that for a new customer, a claim larger than some limit (say $25k) will occur by the end of the calendar year. The limit should be one of the inputs.

```
[ ]: ### enter your code here
     ...
```

5 Part III: Real Data Analysis

In this real data analysis we obtain police traffic stop data from the city of San Antonio's public records. **Ultimately, our goal might be to understand policing patterns and identify any racial disparities and bias in policing.**

For your reference, a brief data documentation is provided here. **Please read carefully!**

Column name	Column meaning	Example value
raw_row_number	An number used to join clean data back to the raw data	38299
date	The date of the stop, in YYYY-MM-DD format. Some states do not provide the exact stop date: for example, they only provide the year or quarter in which the stop occurred. For these states, stop_date is set to the date at the beginning of the period: for example, January 1 if only year is provided.	"2017-02-02"
time	The 24-hour time of the stop, in HH:MM format.	20:15
location	The freeform text of the location. Occasionally, this represents the concatenation of several raw fields, i.e. street_number, street_name	"248 Stockton Rd."
lat	The latitude of the stop. If not provided by the department, we attempt to geocode any provided address or location using Google Maps. Google Maps returns a "best effort" response, which may not be completely accurate if the provided location was malformed or underspecified. To protect against suprious responses, geocodes more than 4 standard deviations from the median stop lat/lng are set to NA.	72.23545
lng	The longitude of the stop. If not provided by the department, we attempt to geocode any provided address or location using Google Maps. Google Maps returns a "best effort" response, which may not be completely accurate if the provided location was malformed or underspecified. To protect against suprious responses, geocodes more than 4 standard deviations from the median stop lat/lng are set to NA.	115.2808
geocode_source	The geocoding service used to geocode the address. Either Google Maps Geocoding API or the Stanford Geospatial Center's geocoding serive. This row is only present for locations that are part of the 2023 update. (See below for further details.)	"GM"
district	Police district. If not provided, but we have retrieved police department shapfiles and the location of the stop, we geocode the stop and find the district using the shapefiles.	8
substation	Police substation.	"NORTH"
subject_age	The age of the stopped subject.	18.0

Column name	Column meaning	Example value
subject_race	The race of the stopped subject. Values are standardized to white, black, hispanic, asian/pacific islander, and other/unknown	“hispanic”
subject_sex	The recorded sex of the stopped subject.	“female”
type	Type of stop: vehicular or pedestrian.	“vehicular”
violation	Specific violation of stop where provided. What is recorded here varies widely across police departments.	“SPEEDING 15-20 OVER”
citation_issued	Indicates whether a citation was issued.	TRUE
search_conducted	Indicates whether any type of search was conducted, i.e. driver, passenger, vehicle.	TRUE
contraband_found	Indicates whether contraband was found. When search_conducted is NA, this is coerced to NA under the assumption that contraband_found shouldn’t be discovered when no search occurred and likely represents a data error.	TRUE
arrest_made	Indicates whether an arrest made.	FALSE
outcome	The strictest action taken among arrest, citation, warning, and summons.	“citation”
search_basis	This provides the reason for the search where provided and is categorized into k9, plain view, consent, probable cause, and other. If a serach occurred but the reason wasn’t listed, we assume probable cause.	“consent”
speed	The recorded speed of the vehicle for the stop.	76.2
posted_speed	The speed limit where the stop was recorded.	55
vehicle_color	A freeform text of the vehicle color where provided; format varies widely.	“BLK”
vehicle_make	A freeform text of the vehicle make where provided; format varies widely.	“TOYOTA”
vehicle_model	A freeform text of the vehicle model where provided; format varies widely.	“Cherokee”
vehicle_registration_state	A freeform text of the vehicle registration state where provided.	“TX”
vehicle_year	Vehicle manufacture year where provided.	2008

5.0.1 3.1 Data Preping (10 pts)

- You can use `pd.read_csv` to read data directly from the following url (double click the cell to copy and past the url) and save it as a dataframe. It may takes up to a minute for the data to load.
  - `https://stacks.stanford.edu/file/druid:wb225bk3255/wb225bk3255_tx_san_antonio_2023_01_`
- Do the following steps of data cleaning:
  - drop all columns start with `raw_`.
  - use `pd.to_datetime` to convert the columns `date` and `time` to correct data types. You will be able to easily extract datetime components, for example, extract the hours in the day using `pandas.Series.dt.hour`.

    - drop rows with missing `date` or `time`.
- Impute missing values in the following columns:
    - subject_age: fill in missing values with mean (round to integer)
    - subject_race: change the "other" category to "unknown", and fill in missing values with "unknown".
    - subject_sex: fill in missing values with "male" (mode)
    - type: fill in missing values with "vehicular" (mode)
- Convert the following boolean values into 0 (False) and 1 (True):
    - `arrest_made`

    - `contraband_found`: missing values should be 0 (False)

    - `search_conducted`
- Based on `vehicle_registration_state`, create a binary indicator `tx_licence`:
    - should be 1 if `vehicle_registration_state` is "TX", otherwise 0.
- Based on `time`, create a binary indicator `night_time`:
    - should be 1 if hours is before 6 A.M. or after 6 P.M., otherwise 0.
- Explore important variables and their distributions in the data and summarize your findings briefly:
    - **your anwser here**

[ ]:
```
### enter your code here
...
```

5.0.2 3.2 Hypothesis Testing (15 pts)

Previous studies using traffic stop data has found that blacks are pulled over more frequently than whites in general. But blacks are less likely to be stopped at night when "a veil of darkness" masks their race. Conduct this hypothesis test using **permutation test**. For simplicity, use `night_time` defined above (from 6 PM to 6 AM) as night time indicator regardless of month and date. Use significance level alpha = 1%.

1. Frame your hypotheses
    - Null: **your anwser here**
    - Alternative: **your anwser here**
2. Choose your test statistics.
    - **your anwser here**
3. Compute your observed stat.
4. Compute your simulated stat 1000 times.
5. Report p-value and make conclusion.
    - **your anwser here**

[ ]:
```
### enter your code here
...
```

5.0.3 3.3 Modeling (20 pts)

We will explore some generalized linear models with the focus on inference and interpretation:

- Model: Use a Generalized Additive Model for the outcome search_conducted and use the following covariates:
  - type
  - subject_age
  - subject_race
  - subject_sex
  - year
  - hour of the day
  - tx_licence
- Use the covariates approperiately as numerical or categorical predictors.
- Choose at least one continuous predictor variable and model the effects as a smooth nonlinear function.
  - To be consistent, use cubic spline with degree of freedom 5, but no further penalty for smoothness.
  - Justify your choice, and explain the interpretation of your estimated effects.

Specific analytic issues you must address:

- According to your model results, which demographic group is more likely to get searched and under what conditions?
  - **your anwser here**
- Assuming no limitation in this data (that all the columns/variables in the data are complete and accurate), what model can we use to identify and quantify racial profiling in receiving speeding tickets (citation)? Please describe your model, but no need to implement it.
  - **your anwser here**

```
[ ]: ### enter your code here
     ...
```

You're done with this exam, YaY!!!

- **Save and Checkpoint** from the File menu,
- **Close and Halt** from the File menu,
- **Rename your ipynb file**, replacing with your name,
- **Upload** your file to the course website.