Rowan'sEditsOnEric'sEditsOnPavitraMAGC2-Copy1

November 14, 2025

```
packages imported
```

1 Problem Statement

Construct a mathematical model to estimate the **total annual income** of Loafe Café at both the Alumni Centre and Sauder locations as a function of daily temperature, daily precipitation, daily humidity and frequency of customers.

2 Variables and Parameters

Symbol	Description	Type	Dimension	Units
d	day index within a (non-leap) year, $d \in \{1, \dots, 365\}$	independent variable	1	—
T_d, P_d, H_d	temperature, precipitation, humidity on day d	independent variable (weather input)	$T, L, 1$	°C,mm,%
C_d	expected total customers on day d (both locations)	dependent variable (model output)	1	customers/day
s	average spend per customer	parameter	1	CAD/customer
R	total annual income	derived aggregate output	1	CAD/year
T_0, P_0, H_0	reference temperature, precipitation, humidity level	parameter (reference level)	$T, L, 1$	°C,mm,%
C_0	baseline customers at reference weather (T_0, P_0, H_0)	parameter	1	customers/day

Symbol	Description	Type	Dimension	Units
a	rate of change of customers w.r.t temperature (squared)	parameter (sensitivity)	T^{-2}	customers / (°C)^2
b	rate of change of customers w.r.t precipitation	parameter (sensitivity)	L^{-1}	customers / mm
c	rate of change of customers w.r.t humidity (squared)	parameter (sensitivity)	1	customers / (%)^2

3 Assumptions and Constraints:

- The number of customers is non-negative and proportional to temperature squared, precipitation, and humidity squared.
- Annual revenue linearly scales with the number of customers per year.
- The average amount of money individual customers spend remains constant all year round.
- On average, customers at the Alumni Centre spend the same amount of money as customers at the Sauder building. This allows us to combine the customer data of the two.
- To evaluate the potential correlation of weather to customer visits, we do not account for external factors such as the school year schedule.
- For usage of this model in current and future years, we assume that weather trends each month remain the same in 2025 and onwards.
- We assume weather trends are different each month, so we will construct separate probability density functions for each to estimate customer visits over the whole year more accurately.
- A a typical non-leap year with 365 days can be used to model annual revenue.
- Leap years are included in the data we use to build the model, but since we sample from each month's pool of data without consideration for what specific day it is, we can still take 365 samples to make a total customer estimate for a non-leap year.
- Reference values are the mean values taken from the entire dataset, as these are the points from which deviating weather measurements would cause a difference in the expected number of customers.

4 Building the Solution:

We estimate total annual revenue of Loafe by: 1) Loading the provided datasets and doing some light exploratory data analysis.
2) Constructing probability density functions for temperature, precipitation, and humidity (T, P, H) for each month. We achieve this with kernel density estimation over the data set using a gaussian kernel.
3) Estimate the parameters of the daily customers equation from data. - Estimate a, b, c, c_0 using the January/July customer dataset. - Estimate t_0, p_0, h_0 using the entire weather dataset. 4) Simulate a year of revenue by sampling weather conditions for each day (ex. if a day is in March then we sample (T_d, P_d, H_d) off of each respective March PDF). We then use those weather samples

(T_d, P_d, H_d) as inputs to the daily customer function and then multiply the daily customers by an average purchase amount. We do this over all 365 days of the year. 5) Finally, we perform uncertainty analysis via Monte Carlo methods on our yearly revenue calculations to get a range of plausible values.

Data processing

In the code cell below we combine both Loafe locations customer data into a January total and a July total respectively.

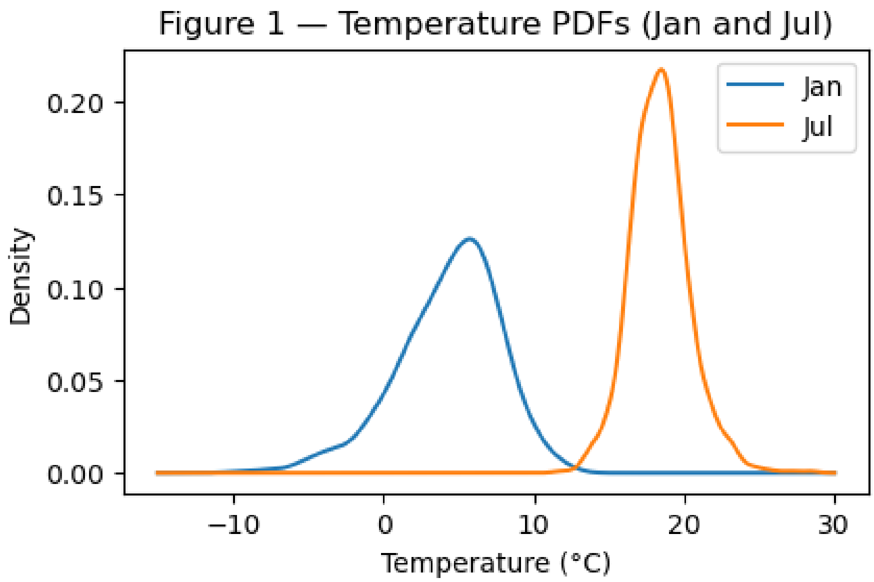
Build monthly PDFs

We split weather data by month and fit Gaussian KDEs for temperature, precipitation, and humidity. We use KDEs to build these monthly weather PDFs (for temperature, precipitation, humidity) because KDEs give a smooth probability density function without having to make any assumptions about what type of distribution the data follows.

Figure 1 overlays **January and July** PDFs for Temperature so we can visually confirm that the winter/summer weather patterns that will drive the customer model are different.

[44]:
```
# ========= Monthly splits & KDEs =========
# split weather into 12 months
listOfMonthData = []
for i in range(1, 13):
    listOfMonthData.append(weather.loc[weather["Month"] == i, ["temperature",␣
 ↪"precipitation", "humidity"]].dropna())

class Month:
    def __init__(self, index, month_pdf_T, month_pdf_P, month_pdf_H):
        self.index = index # 1 indexed from January
        self.month_pdf_T, self.month_pdf_P, self.month_pdf_H = month_pdf_T,␣
 ↪month_pdf_P, month_pdf_H

# build KDEs per month
listOfMonthPDFs = []
for i in range(12):
    month = listOfMonthData[i]
    pdf_T, pdf_P, pdf_H = stats.gaussian_kde(month["temperature"]), stats.
 ↪gaussian_kde(month["precipitation"]), stats.gaussian_kde(month["humidity"])
    listOfMonthPDFs.append(Month(str(i+1), pdf_T, pdf_P, pdf_H))

# spaces to evaluate estimated PDFs over
gT, gP, gH = np.linspace(-15, 30, 400), np.linspace(0, 60, 400), np.
 ↪linspace(20, 100, 400)

# Figure 1
plt.figure(figsize=[5, 3])
plt.plot(gT, listOfMonthPDFs[0].month_pdf_T(gT))
plt.plot(gT, listOfMonthPDFs[6].month_pdf_T(gT))
plt.legend(["Jan", "Jul"])
```

```
plt.xlabel("Temperature (°C)")
plt.ylabel("Density")
plt.title("Figure 1 - Temperature PDFs (Jan and Jul)")
plt.show()
```



Specify the daily customer amount equation as

$$C_d(T_d, P_d, H_d) = \max\{\, C_0 + a\,(T_d - T_0)^2 - b\,(P_d - P_0) - c\,(H_d - H_0)^2,\ 0\,\}$$

- We chose T_0, P_0, and H_0 to be the mean value for each of the respective weather measurements over the entire dataset. The justification for this is that from the structure of the equation if the inputted values (T_d, P_d, H_d) are exactly equal to each respective parameter (T_0, P_0, and H_0) the amount of customers for that day will just be equal to C_0. Therefore the reference values should represent what typical weather looks like, and the mean value is the statistic that does that.

- We get **Jan/Jul mean weather** from the KDEs (numerical mean via integration on a fine grid).

- We estimate a, b, c using a finite-difference idea based on the **change in average daily customers** from January to July.

- We chose C_0 to be the mean between the average daily number of customers for January and the average daily number of customers for July.

```
Reference mean averages: T0=10.60°C, P0=3.19mm, H0=78.10%
Jan vs Jul daily customers: 350.4 → 175.4
a=-8.121, b=36.557, c=-14.826, C0=262.9
```

4.1 Yearly Simulation

We simulate a **typical** (non-leap) year by: - mapping day $d \in \{1..365\}$ to its **month**,
- drawing a single (T, P, H) sample from that month's PDFs,
- applying $C_d(T_d, P_d, H_d)$, and
- multiplying by a constant $s = 10.8$ \$ **spend per customer**.

Note we chose $s = 10.8$ by taking the average price of all items on the menu at the sauder Loafe location. Assuming that a random customer spends around s \$ per visit we will multiply the average amount of revenue per customer(s) by how many customers we have per day(C_d) to get an estimate of revenue per day.

This produces a **sample** R of annual revenue that follows the equation

$$R = \sum_{d=1}^{365} C(T_d, P_d, H_d) \times s.$$

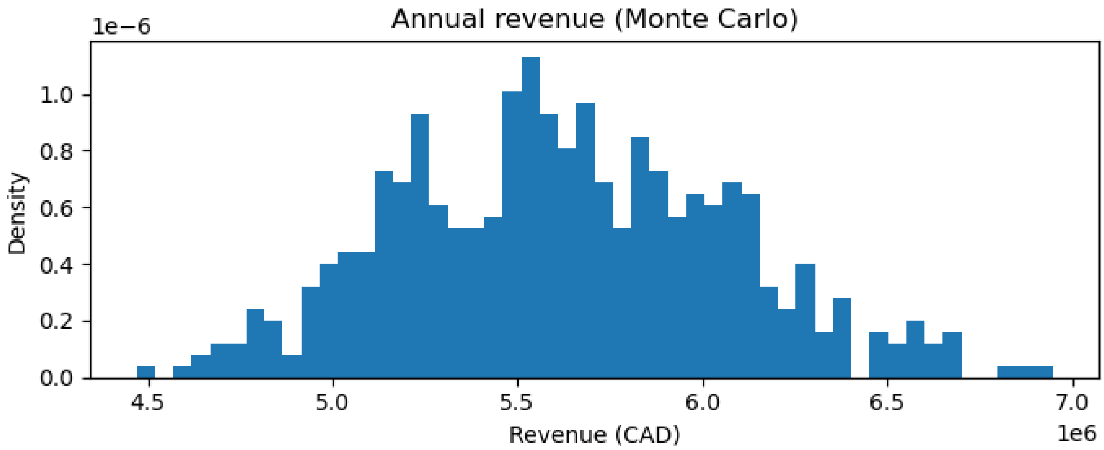
[27]:
```
# Defining functions

# Used for determining which month to sample weather data from
days_per_month = np.array([31,28,31,30,31,30,31,31,30,31,30,31])
month_edges = np.cumsum(days_per_month)

# Used for determining which month to sample from
def dayToMonth(day):
    return int(np.searchsorted(month_edges, day))  # 0..11

# Daily revune function definition
def dailyRev(month, spend_per_customer=10.8):
    T_hat = listOfMonthPDFs[month].month_pdf_T.resample(1).ravel()[0] #bug?
    P_hat = listOfMonthPDFs[month].month_pdf_P.resample(1).ravel()[0]
    H_hat = listOfMonthPDFs[month].month_pdf_H.resample(1).ravel()[0]
    customersDay = cFunc(T_hat, P_hat, H_hat)
    return customersDay * spend_per_customer

# Yearly revune function definition
def yearlyRev(spend_per_customer=10.8):
    total = 0.0
    for i in range(1, 365+1): #bug?
        m = dayToMonth(i)  # 0..11
        total += dailyRev(m, spend_per_customer=spend_per_customer)
    return total

s=10.8
yearRev = yearlyRev()
print(f"Sample R revenue/year (s=${s:.2f}): ${yearRev:,.0f}")
```

```
Sample R revenue/year (s=$10.80): $4,994,541
```

4.2 Monte Carlo Estimation

Below is a unpretrubed monte carlo simulation of our yearly revunue:

[38]:
```
N = 500 # simulate 500 years of revune
yearlyRevenues = [yearlyRev() for _ in range(N)] # list unpacking python
 ↪notation
```




```
Mean annual revenue: $5,625,642 |Median annual revenue: $5,601,271 | 90% CI:
[$4,929,930, $6,389,604]
```

Below is a pertrebued monte carlo simulation

To acknowledge limited Jan/Jul labels, we run a **Monte Carlo** simulation: - perturb $a, b, c, C_0, T_0, P_0, H_0$ by adding modest Gaussian noise,
- resample weather each day from the month's KDEs,
- compute the annual revenue, and repeat $N = 1000$ times.

We then visualize the distribution (Figure 3) and report the median and a 90% interval.

- **Figure 3:** Histogram of annual revenue from Monte Carlo.

- **Figure 4:** Simple one-dimensional sensitivities of expected customers against each weather variable around the reference levels (helps interpret a, b, c).
  We also print the median and 90% interval of revenue.

```
MC median: $56,311 | 90% CI: [$40,810, $73,798]
```

Figure 3 — Annual revenue uncertainty (Monte Carlo)

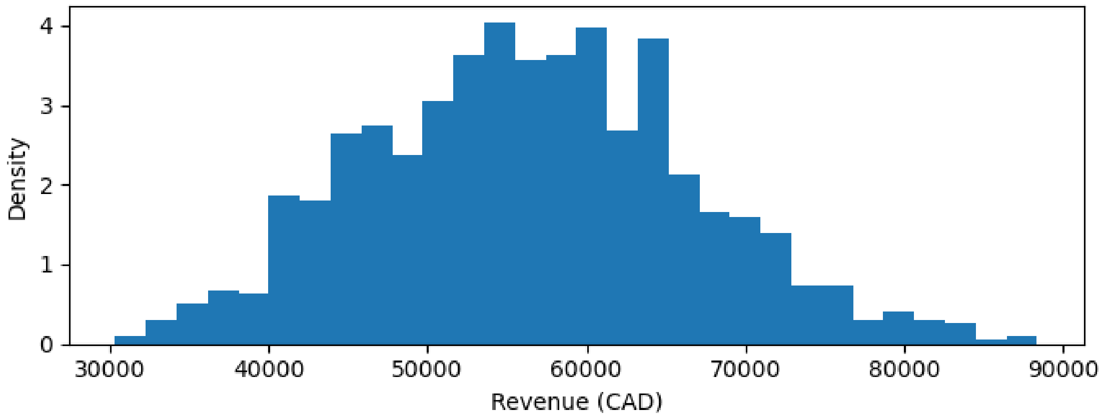
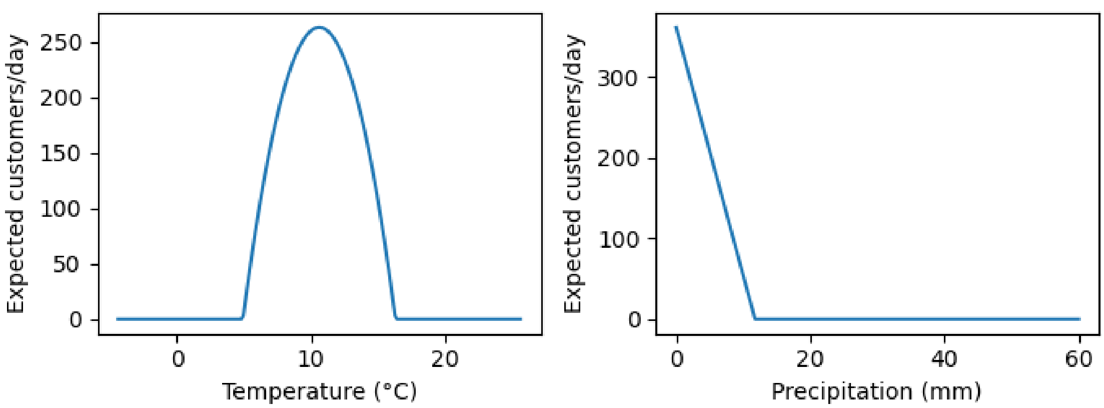


Figure 4 — Local sensitivities around reference weather



5 Analysis and Assessment:

From the sensitivity plots, we see: - The number of customers changes in the same direction as the sign of a or $-c$ as temperature or humidity deviate further from their reference values. - The number of customers changes in the same direction as the sign of $-b$ if precipitation is greater than the reference value, or goes the opposite direction if it is less than it.

Reporting some of the statistics from our monte carlo simulation of total annual revune, we have that the average annual revune for Loafe across all locations is BLANK\$ per year, the mode of annual revune is 56,311\$, and the variance is BLANK ($E[x^2]-\mu^2$). Our use of monte carlo analyis here is justified due to the fact that the quantity we are trying to predict is dependant on random varibles(temprature,precepatiation,humidatity), so we must use stochastic numerical methods to construct a good estimate.

```
[ ]:
```