EE 046202 - Technion - Unsupervised Learning & Data Analysis

Homework 1 - Statistics

Due Date: 29.11.2020

Agenda

- Questions
  - Gaussian RVs
  - Parametric & Non-Parametric Estimation
  - Exam Question - Estimators
- Python Exercise - Parkinson's Disease Classification Data Analysis

Use as many cells as you need

אפשר גם לכתוב בעברית, אבל עדיף באנגלית

- Code Tasks are denoted with:
- Questions (which you need to answer in a Markdown cell) are denoted with:
- $\LaTeX$ Cheat-Sheet (https://kapeli.com/cheat_sheets/LaTeX_Math_Symbols.docset/Contents/Resources/Documents/index) (to write equations)
  - Another Cheat-Sheet (http://tug.ctan.org/info/latex-refsheet/LaTeX_RefSheet.pdf)

Students Information

- Fill in

Name	Campus Email	ID
Student 1	student_1@campus.technion.ac.il	123456789
Student 2	student_2@campus.technion.ac.il	987654321

Submission Guidelines

- Maximal garde: **100** (even with the bonus, the grade will not be above 100).
  - Example: if you got 5 points bonus, but you were right in all sections, your grade will still be 100 (and not 105).
  - Example: if you got 5 points bonus, and 6 points were deducted for wrong answers, your grade will be 99.
- Submission only in **pairs**.
  - Please make sure you have registered your group in Moodle (there is a group creation component on the Moodle where you need to create your group and assign members).
- **ANSWERS TO THEORETICAL/MATHEMATICAL QUESTIONS**:
  - **Typed - 5 points bonus**: you can type directly in a Markdown cell using Latex (see cheatsheets above), or use Word, Overleaf, LyX...
    - This is a really good practice, we encourage you to practice your math typing skills.
  - **Handwritten** - if we can't read your handwriting, we will automatically take off the points of the questions. Please write clearly. No bonus for handwritten submissions.
- SAVE THE NOTEBOOKS WITH THE OUTPUT, CODE CELLS THAT WERE NOT RUN WILL NOT GET ANY POINTS!
- What you have to submit:
  - If you have answered the questions in the notebook, you should submit this file only, with the name: `ee046202_hw1_id1_id2.ipynb` .
  - If you answered the questionss in a different file you should submit a `.zip` file with the name `ee046202_hw1_id1_id2.zip` with content:
    - `ee046202_hw1_id1_id2.ipynb` - the code tasks
    - `ee046202_hw1_id1_id2.pdf` - answers to questions.
  - No other file-types (`.py` , `.docx` ...) will be accepted.
- Submission on the course website (Moodle).

Working Online and Locally

- You can choose your working environment:
  1. `Jupyter Notebook` , **locally** with Anaconda (https://www.anaconda.com/distribution/) or **online** on Google Colab (https://colab.research.google.com/)
     - Colab also supports running code on GPU, so if you don't have one, Colab is the way to go. To enable GPU on Colab, in the menu: `Runtime` $\rightarrow$ `Change Runtime Type` $\rightarrow$ `GPU` .
  2. Python IDE such as PyCharm (https://www.jetbrains.com/pycharm/) or Visual Studio Code (https://code.visualstudio.com/).
     - Both allow editing and running Jupyter Notebooks.
- Please refer to `Setting Up the Working Environment.pdf` on the Moodle or our GitHub (https://github.com/taldatech/ee046202-unsupervised-learning-data-analysis (https://github.com/taldatech/ee046202-unsupervised-learning-data-analysis)) to help you get everything installed.
- If you need any technical assistance, please go to our Piazza forum (`hw1` folder) and describe your problem (preferably with images).

Keyboard Shortcuts

- Run current cell: **Ctrl + Enter**
- Run current cell and move to the next: **Shift + Enter**
- Show lines in a code cell: **Esc + L**
- View function documentation: **Shift + Tab** inside the parenthesis or `help(name_of_module)`
- New cell below: **Esc + B**
- Delete cell: **Esc + D, D** (two D's)

Question 1 - Gaussian RV- Basics

1. Let $Z \sim \mathcal{N}(0,1)$ be a normal Gaussian RV, and $X \sim \mathcal{N}(\mu, \sigma)$. The Cumulative Distribution Function (CDF) of Z is defined as
$$P(Z \leq c) \triangleq \phi(c).$$
Express $P(X \geq x)$ using $\phi(c)$.
2. Consider a sequence of N i.i.d. RVs $\{X_i\}_{i=1}^N$, where $X_i \sim \mathcal{N}(10,1)$. The empirical mean is given by $\overline{X}_N = \frac{1}{N}\sum_{i=1}^N X_i$. What is the distribution of $\overline{X}_N$?
3. What is the probability $P(9.7 \leq \overline{X}_N \leq 10.3)$ for $N = 1, 10, 20$? Express first using the function $\phi(X)$ and then use `scipy.stats.norm.cdf` to calculate $\phi(x)$ and obtain a numerical value.
4. Since Gaussian RVs are not bounded, we cannot use **Hoeffding's** inequality to bound terms of the form $P(9.7 \leq \overline{X}_N \leq 10.3)$. A possible alternative for this is to use the following proposition (which is more general and holds for a sum of sub-Gaussian RVs):
   - **Proposition**: Let $\{X_i\}_{i=1}^N$ be i.i.d. RVs with $X_i \sim \mathcal{N}(\mu, \sigma)$. Then:
   $$P(|\frac{1}{N}\sum_{i=1}^N X_i - \mu| \geq \epsilon) \leq 2\exp(-\frac{N\epsilon^2}{2\sigma^2}).$$
   Use this proposition to find a lower bound for $P(9.7 \leq \overline{X}_N \leq 10.3)$ for $N = 1, 10, 20$.

Question 2 - Parametric and Non-Parametric Estimation

1. Suppose $\hat{\theta}$ is an estimator for an unknown parameter θ. Show that
$$MSE(\hat{\theta}) = Var(\hat{\theta}) + Bias^2(\hat{\theta})$$
2. Let $X_1, \ldots, X_N \sim Bernoulli(p)$ and let $Y_1, \ldots, Y_N \sim Bernoulli(q)$ be i.i.d. RVs.
   - Find a (non-parametric) point estimator and the estimated standard error for p (use Hoeffding).
     - Recall the standard deviation:
     $$s_N(\hat{\theta}) = \sqrt{Var(\hat{\theta})}$$
   - Find an approximated 90%, 95%, 99% confidence intervals for p.
   - Find the point estimator and the estimated standard error for $p - q$.
   - Find an approximated 90% confidence interval for $p - q$.
3. Let $X_1, \ldots, X_N \sim Binomial(10, \theta)$ be i.i.d. RVs. Estimate θ using MLE.
   - $P(X_i|\theta) = \binom{10}{X_i}\theta^{X_i}(1-\theta)^{10-X_i}$
4. Let $X_1, \ldots, X_N \sim F$ be i.i.d. RVs, where F is an arbitrary, unknown CDF. Let $\hat{F}$ be the empirical distribution function. For a fixed F, use the central limit theorem (CLT) to find the limiting distribution of $\hat{F}_n(x)$.
5. In the lecture and tutorial, we stated the **DKW** theorem and derived a C.I. for the empirical CDF, for 1D type of data. Derive a C.I. for the empirical CDF in a general dimension, i.e., as a function of $C(k)$[see non-parametric chapter in the lectures]. If $C(k) \sim \exp(k)$, what does it mean about the *hardness* of the problem in high dimension?
6. Calculate $\mathbb{E}[\hat{F}_N]$ and $Var[\hat{F}_N]$ using the definition of the empirical distribution function (remember that $X_1, \ldots, X_N$ are i.i.d. from F).

Question 3 - Exam Question - Estimators

In order to check electrical devices, a system performs repeated tests in a device until its first failure appears. The tests were performed on N devices. Denote K_i as the number if tests that were performed on the i^{th} device (including the final test where the failure appeared), where $i \in \{1, \ldots, N\}$. Assume that $K_1, \ldots, K_N$ are i.i.d. random variables.

1. Find a non-parametric estimator(i.e. the plug-in/point estimator) for $\mathbb{E}[K]$ (hint: use the Tail Sum formula).
2. Find a non-parametric estimator for $\hat{p}_3 \triangleq P(K = 3)$, the probability that a failure will occur in the third test.
3. Suggest a confidence interval (CI) for $\hat{p}_3$ for $\alpha = 0.05, N = 100$.

Assume $K \sim Geom(p)$ (Geometric Distribution), where p is unknown.

1. Calculate the MLE for p and the mean $\mu = \mathbb{E}[K]$.
2. Calculate the probability that the number of tests is odd, i.e., that K is odd, $P(K \text{ is odd})$. Simplify as much as possible.

Question 4 - Python - Parkinson's Disease Classification Data Analysis

In this exercise, we are going to do data analysis with Python and Pandas. As this is the first "real" exercise, we will add guidance for some of the tasks.

1. Warmup - Generate 100 samples from $\mathcal{N}(0,1)$ (`np.random.randn`). Compute a 95% CI for the CDF. Plot the true CDF, the CDF estimation and the CI in a single plot. To estimate $\hat{F}_n$ use a histogram (`np.histogram`). Repeat this $K = 1000$ times and compute the percentage of time that the interval contained the CDF (print the value) . In addition, plot in another single figure the *true* CDF, and the best and worst experiments (use $\max_x |F(x) - \hat{F}_n(x)|$ as quality measure).
   - To compare np arrays element-wise use `np.less_equal(x1, x2 + eps), np.greater_equal(x1, x2 - eps)` , use `.all()` to verify if all the comaprisons were `True` .

We are now going to perform some real data analysis on the "Parkinson's Disease Classification Data Set": the data used in this study were gathered from 188 patients with PD (107 men and 81 women) with ages ranging from 33 to 87 at the Department of Neurology in CerrahpaÅŸa Faculty of Medicine, Istanbul University. The control group consists of 64 healthy individuals (23 men and 41 women) with ages varying between 41 and 82. During the data collection process, the microphone is set to 44.1 KHz and following the examination, the sustained phonation of the vowel /a/ was collected from each subject with three repetitions.

The features are various speech signal processing algorithms including Time Frequency Features, Mel Frequency Cepstral Coefficients (MFCCs), Wavelet Transform based Features, Vocal Fold Features and TWQT features have been applied to the speech recordings of Parkinson's Disease (PD) patients to extract clinically useful information for PD assessment.

1. Load the data with pandas, drop the 'id' column, take a sample ($k = 10$, `dataframe.sample(k)`) and view it.
   - The filename is `pd_speech_features.csv` .
2. Compute the empirical correlation between all pairs of features. Show the results both in a heatmap.
   - Use pandas `.corr()` to calculate the correaltion, and `plt.imshow()` to view the heatmap (2 heatmaps, one for the correlation and one for the absolute correlation). Add a color bar using `plt.colorbar()`
3. Print the top-20 most correlated features. Follow this steps:
   - Take the lower triangle of the correlation matrix (as it is symmetrical and we don't care about $Corr(X_i, X_i)$). Use `np.tril()`
   - Consider only positive correlation (because negative correlation has a different, useful meaning). You can do that by `X = X[X >0]` .
   - From here, these are recommended steps, feel free to achieve the goal in a different way.
     - Assignment to a pandas DataFrame: `X.loc[:,:] = np.(...)`
     - Unstacking the DataFrame (creates a new pivot, read the doc): `df.unstack()`
     - Sorting: `df.sort()`
4. What is the meaning when 2 different features are highly correlated? From a machine learning perspective, can a classifier learn new insights from highly-correlated features? In your answer, address the process of "feature selection" in ML (usually performed as a pre-processing step).
5. Compute the **in-class** correlation between features. Plot a heat map for each class. Address the differences between the heat maps.
6. Consider the features 'numPulses' and 'app_entropy_log_5_coef'. We wish to calculate a 95% confidence interval for the correlation between these features. We will use *Bootstrapping* and the *Chebyshev inequality* (as in Tutorial 2).
   - Implement the bootstrap algorithm to calculate the standard deviation (σ) of the correlation.
     - You can use the algorithm from the tutorial, but you have to modify it to support 2 arrays.
     - The algorithm will output the empirical correlation of the two input features, and a bootstrap estimation for the std (σ). Use `K=300` bootstrap samples and `m=100` experiments.
     - Tips:
       - To get values for two columns: `data[['numPulses', 'app_entropy_log_5_coef']]`
       - To calculate correlation, check out `numpy.corrcoef` .
   - Use σ to calculate a 95% CI using Chebyshev inequality.
     - Remember to to normalize by the size of the data (N). The final print should look something like that: `95% confidence interval for the correlation: *** ± ***`

In [1]:
```python
# imports for q-4
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy.stats  import norm
```

Credits

- Icons from Icon8.com (https://icons8.com/) - https://icons8.com (https://icons8.com)
- Datasets from Kaggle (https://www.kaggle.com/) - https://www.kaggle.com/ (https://www.kaggle.com/)