University of Ioannina,
Department of Computer Science & Engineering

Data Mining – CSE 012/CS 059
Fall 2021

# Assignment 1

This is the first assignment. The deadline for the assignment is December 15, 11:59 pm. Hand in the Notebooks with the code and reports for all Questions. For Question 1, you can hand in a pdf with the proof, or photos from hand-written notes. For late submissions the late policy on the page of the course will be applied. The submission of the assignment will be via ecourse. Details are on the Assignments web page of the course. Assignments will be done **individually**.

## Question 1

In this question you are required to modify the Reservoir Sampling algorithm we described in class to sample $\boldsymbol{K}$ items, uniformly at random, from a stream of $N$ items, so that each element has probability $K/N$ to appear in the sample. Your algorithm should work with a single pass over the stream, reading the items one-by-one, without prior knowledge of the length of the stream (the size $N$), using $O(K)$ amount of memory (assume that the size of each item is constant. This means that you cannot store the whole stream in memory.

1. Describe the algorithm for sampling $K$ items uniformly at random from a stream of $N$ items. The description of the algorithm should **not** be in code or pseudocode, or the translation of code in natural language. In this case your answer will receive no marks. The description should describe the logic of the algorithm is plain English.
   For example, this is a description, in plain English, of the algorithm for sampling a single item that we saw in class:
   *The algorithm keeps space in memory for one item and a counter of the number of items it has seen. It goes through the items one by one, as they come from the stream. When it sees the $n$-th item, it selects it with probability $\frac{1}{n}$, and stores it, replacing the existing item (if not the first one). It updates the counter. When the stream ends, it returns the stored item.*
2. Prove that your algorithm produces a uniform sample, that is, for every $i, 1 \leq i \leq N$, the $i$-th element has probability $K/N$ to appear in the sample.
3. Write a function **sample in Python** that implements the sampling algorithm. The function takes as input the name of a file, and the number $K$, and returns a list that holds a sample of $K$ lines selected uniformly at random from the file. You should read the file line-by-line, and never load it in memory. Use the function in a program to sample 10 lines from the text file *input.txt* that is given to you. Print the lines that you sampled.
   Explain the mapping between your description in Step 1, and your code. For example, for the algorithm that samples a single item, you could write something like: "*In lines 2-3 we implement the sampling of the $n$-th item with probability $\frac{1}{n}$*".

Create a Notebook with two code cells. One with the imports and the definition of the function sample, and one with the code that uses the function on the file "input.txt", stores the sample, and prints it. You can download the input.txt file from the Assignments page. Also add two cells with Markdown text, one with the description of the algorithm (Step 1), and one with the mapping between the description and the code. You can add the proof (Step 2) in a separate cell, or write it separately and submit a pdf file with the text, or a photo if handwritten. Hand in the Notebook and the input.txt file, and the pdf or photo files, if any.

## Question 2

On the Assignments page of the course there is a file "data.csv". The file contains three comma-separated columns of 1000 values, with column names A, B, and C, and 1000 lines. The values in B and C columns are a function of those in A. Specifically, for each value $x$ in column A, the corresponding value in columns B and C are $f_B(x) \cdot (1 + \epsilon)$ and $f_C(x) \cdot (1 + \epsilon)$ respectively, where $\epsilon$ is random noise (different for each $x$, and each column). Your goal is to find the functions $f_B$ and $f_C$.

To determine the functions, load the data into a Pandas dataframe and create plots of B and C against A, as described in class, as well as any other plot you need. Hand in a Python Notebook, which should contain the code for processing the data, the plots and computations that you did, and a report with your conclusions.

## Question 3

In the past years, analysis of sports data has become a scientific field of its own. A sport where it has found widespread application is basketball. In this question we will study data from the NBA (National Basketball Association). Our goal is to make some observations about the date, find interesting correlations, and investigate some hypotheses. Also, to practice with the use of Pandas for data analysis.

You can download the data that we will use from the Assignments page of the course. We have two datasets: Dataset1 (source: Kaggle) and dataset2 (source: Kaggle).

**Dataset1** consists of three files. You can read about them in the Kaggle link. We will only use the Seasons_Stats.csv file. It contains collective statistics for the players for different seasons, starting from 1950. Here is a glossary for the fields (columns) in the file.

You will do the following pre-processing of the data:

1. You will keep only the years (Year) after 1980.
2. You will keep only players (lines) with age (Age) greater than 18, and smaller than 40.
3. You will keep only the seasons (lines) of a player where he has played more than 500 minutes (MP).
4. You will keep only the positions (Pos) 'PG' (Point Guard – position 1), 'SG' (Scoring Guard – position 2), 'SF' (Small Forward – position 3), 'PF' (Power Forward – position 4), 'C' (Center – position 5). (You may find handy the method isin for a column).

We will not use all the columns. We will mostly use the following columns: 'G' (Games), 'PTS' (Points), 'AST' (Assists), 'TRB' (Total Rebounds), 'BLK' (Blocks), 'PER' (Player Efficiency Rating – a metric for the overall evaluation of a player), 'TS%' (True Shooting Percentage). You can see the definition of these fields in the glossary. It will be useful to create some fields of your own, such as per game statistics (e.g., points per game). When using a field, make sure to remove the null values.

For preprocessing you will use Pandas methods.

**Dataset2** contains more detailed data for the seasons 2014-2021 (up to a few days ago). It contains data about the teams (teams.csv), players (players.csv), games (games.csv), the ranking of teams at each game (ranking.csv), and the statistics of the players per game (games_details.csv). The statistics are a subset of those appearing in dataset1. We will mostly use the files games.csv and games_details.csv.

The question has the following parts. The goal is to implement the following by loading the data into Pandas dataframes, and using mostly Pandas methods (plus you own functions that you can use with apply).

**A.** In this part, we are interested in understanding the distribution that the total number of points scored by a player follows. You should do the following plots:

1. A histogram of the points with 100 bins, using the function of the Pandas library
2. A histogram of the logarithm of the points with 100 bins using again Pandas methods.
3. A histogram of the points with 100 bins of equal size that you will construct. In the X axis you will have the lower end of the bin, and in the Y axis the number of points that fall in the bin. You will use logarithmic scale for both axes.
4. The Zipf plot of the distribution. The Zipf plot is constructed by having on the Y axes the values (the points in our case), and on the X axis the rank of the values. For example, the largest number of points has rank 1, the second largest has rank 2, and so on. The plot will be in log-log scale.

Present your plots in a grid, and comment on the distribution.

Note: There is no clear conclusion about the distribution, but there are some observations that you can make about its shape. You can also add a plot of your own if you believe it will help. Steps 3,4 are harder to implement using Pandas methods (especially Step 3), so you can implement them by transferring the data in lists.

**B.** In this part, we are interested in the evolution of the metrics over time. Specifically, we will look at how the player performance changes with age. Specifically, from the existing data, we will look at PER and TS%, while you will also compute the fields, Points Per Game (PPG), Assists Per Game (APG), Total Rebounds Per Game (RPG), and Blocks Per Game (BPG) per season. Use the lineplot method of seaborn to plot the mean value of the statistics as a function of age. Present your plots in a 2X3 grid. What do you observe? At what age do players peak? How does this differ between different statistics? Comment on your results.

We will then look if the player performance over age is affected by the position. Do the same plots, this time having different discriminating between the different positions (use the parameter hue of lineplot). Do you see any difference depending on the position? Comment on the results.

**C.** In this part, we are interesting to investigate if there is a correlation between the basic statistics of players. Compute again the statistics PPG, APG, RPG, BPG that you computed in Part B, this time not per season, but using the data from all seasons. Create a plot with all the pairwise scatter plots (use the pairplot method of seaborn), and two 4X4 tables, with the Pearson correlation coefficients, and the corresponding p-values. (Alternatively, you can use heatmaps, with the values, to present your results). Comment on the results. Do you observe any interesting correlation? A correlation is interesting if it has a large coefficient, and a p-value smaller than 0.05.

Bonus: Create the same plots and measurements per position (use hue for the plots). Comment on the results.

**D.** In this part we are interested in studying if there is a difference between the performance of the players in different positions. Use the statistics PPG, APG, RPG, BPG that you computed in Part C, and create barplots with the means for each of position, with 95%-confidence intervals (using seaborn). Place them on a grid with 4 positions. Inspect visually, and report your conclusions.

Then do the same bar plot for the mean PER of the players over all seasons. The differences are no longer so clear. Use the t-test to decide which differences are statistically significant. Create ta 5X5 table with the p-values for all combination of positions, and comment on what you observe.

**E.** Russell Westbrook is a polarizing player. Although he has recorded multiple triple-doubles (double digit numbers for points, assists, and rebounds in a game), many think that this is achieved at the expense of his team. Compute the conditional probability that the team of Westbrook wins when he records a triple-double, and compare it with the probability that the team wins regardless of the triple double when Westbrook plays. Use the $\chi^2$-test to examine if a Westbrook triple-double and a team win are independent. Comment on your results.

For this part, you will use dataset2. The PLAYER_ID of Westbrook is 201566. The information about which team won a game is in games.csv, while the statistics of Westbrook per game are in games_details.csv.

Bonus: Westbrook haters will say that he records triple-doubles against weaker teams. Use the data to examine this hypothesis.

**Z**. In this part we will make some hypotheses and examine them in the data.

Hypothesis 1: There is a correlation between the number of assists by the backcourt of a team (PG, SG), and the points scored by the frontcourt of the team (SF, PF, C).

Use dataset1 and the Pearson correlation coefficient to test this hypothesis.

Hypothesis 2: Teams score on average more points at home, than away.

To test this hypothesis, use dataset2. We will examine two teams: Boston Celtics (BOS – TEAM_ID = 1610612738) and Minnesota Timberwolves (MIN – TEAM_ID = 1610612750). You will perform two experiments to test the hypothesis:

1. In the first experiment use the t-test to test the hypothesis.

2. In the second experiment you will do a permutation test. Given a team, for each game of the team we can compute the number of points it scored, and a label “home”, or “away”. We can now compute the difference between the mean number of points at home and away. This is the observed difference (or observed value). Now, keep the column with the points fixed, and permute the values in the column with the labels, creating a random permutation. Compute again the difference between the mean number of points at home and away. Repeat this process 1000 times, to obtain 1000 differences. Compute the empirical p-value for the observed value (the fraction of the 1000 experiments that have difference value greater or equal from the observed difference). The observed difference is statistically significant if the empirical p-value is less than 0.05. Report your results for the two teams.
   Create also the histogram of the 1000 values you computed. Place the observed value as a vertical line on the histogram, to visually demonstrate the result of your experiment (use the method axvline of pyplot for the vertical line at the point of the observed difference).

**H.** Postulate a hypothesis of your own and test it using the data.

Hand in a Notebook that contains your code for the data processing, plots and computations, as well as your observations and your conclusions. Add headers in the notebook, to clearly separate the different parts of the question.