**ML Major 1**

**Submmiters:**

**Husain Azaqy: 213382104**

**Galor Lazar: 325028876**

1. Load the dataset into a Pandas DataFrame.

**Answer** (in your report)**:** how many rows and columns are in the dataset?

**25 columns**

**1250 rows**

1. Print the value\_counts of the conversations\_per\_day feature (see Tutorial 01).   
   Copy the obtained output to your report. Describe in one short sentence what you think this feature refers to in the real world.

This feature’s type is “ordinal”. Explain briefly why.

conversations\_per\_day

3 218

2 204

5 179

4 168

1 108

6 107

7 94

8 54

9 42

10 29

11 16

13 8

12 7

14 6

16 5

15 3

17 1

29 1

Name: count, dtype: int64

I think this feature refers to how many times a person spoke to other people that they can infect with the virus. It is an ordinal variable since the count of conversations is a natural number with natural order as a counting variable.

1. In your report, write a table describing each feature. The columns must be:
   1. Feature name: the name of the feature as it is written in the dataset.
   2. Description: a short sentence with your understanding of the feature’s meaning in the real world.
   3. Type: Continuous, Categorical, Ordinal, or Other.

Don’t overthink this (especially the “ordinal” type), some variable may be suitable for two types.

Note: do not include the target columns (“spread” and “risk”).

|  |  |  |
| --- | --- | --- |
| Name | Description | Type |
| patient\_id | Unique identifier for each patient | Ordinal |
| age | Patient age in years | Ordinal |
| sex | Patient gender | Categorical |
| weight | Patient weight in kilograms | Continuous |
| blood\_type | Patient blood group | Categorical |
| current\_location | Patient present location | Categorical |
| num\_of\_siblings | Number of patient's siblings | Ordinal |
| happiness\_score | Patient reported happiness level | Ordinal |
| household\_income | Annual/Monthly income of patient's household | Continuous |
| conversations\_per\_day | Daily conversations count for patient | Ordinal |
| sugar\_levels | Patient blood sugar levels | Ordinal |
| sport\_activity | Frequency of the patient's sports participation | Ordinal |
| pcr\_date | Date of PCR test | Ordinal |
| PCR\_01 | PCR test result 1 | Continuous |
| PCR\_02 | PCR test result 2 | Continuous |
| PCR\_03 | PCR test result 3 | Continuous |
| PCR\_04 | PCR test result 4 | Continuous |
| PCR\_05 | PCR test result 5 | Continuous |
| PCR\_06 | PCR test result 6 | Continuous |
| PCR\_07 | PCR test result 7 | Continuous |
| PCR\_08 | PCR test result 8 | Continuous |
| PCR\_09 | PCR test result 9 | Continuous |
| PCR\_10 | PCR test result 10 | Continuous |

1. [Split](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html) the data randomly into a training set (80% of the data) and a test set (20% of the data). As the random\_state, use the sum of the last two digits of each of your IDs[[1]](#footnote-1) (two or three IDs).

The random state will ensure that you get the same split every time.   
Answer: Why is it important that we use the exact same split for all our analyses?

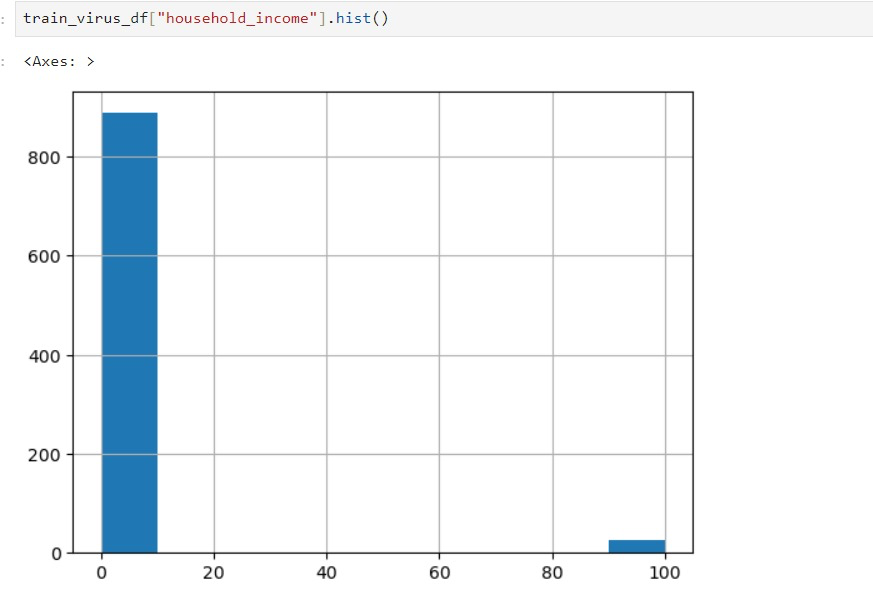
* Since the default split is random, we want the same random selection between multiple analyses to be able to compare them reliably and easily.

1. For **both the training set and test set**, report which fields have missing values and how many missing values there are. You can use Panda's function [isnull()](https://pandas.pydata.org/docs/reference/api/pandas.isnull.html).

* The only field that has missing values is “ household\_income” with 109 missing values.
* 86 of them are in the training set.
* 23 of them are in the test set.

1. Plot a histogram (see Tutorial 01) for each field where you found missing values in **‎(Q5)**. Add these plots to your report. Answer: Can you recognize outliers?

**Reminder:** Create plots using only the training set.



There are outliers as we can see in the far-right corner of the plot.

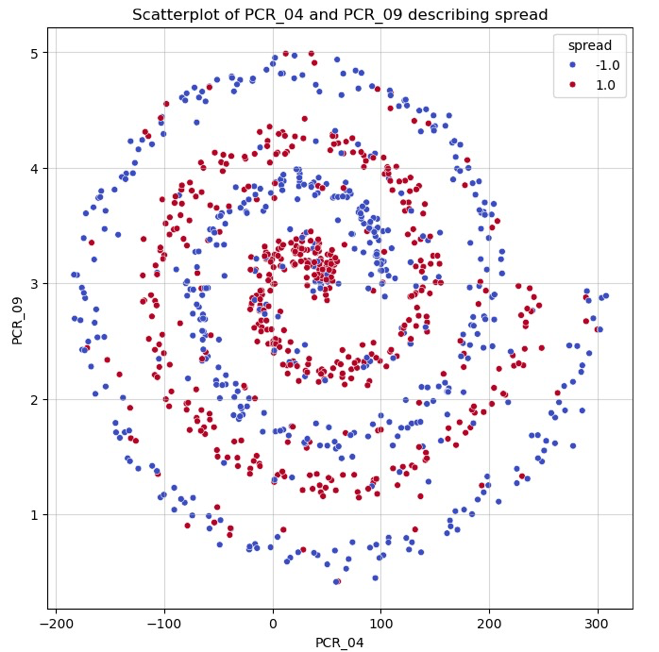
1. For each field where you found missing values, calculate the median and the mean in the training set, **and report it**.

If there is a significant difference between the mean and median values, explain the reason. Which filling method do you prefer to use in our case, and why?

* The mean household income 3.535886214442013
* The median household income 0.7
* We found that 888 of the 914 values in the training set are under the mean household-income. It means that the mean value doesn’t represent a good average household-income value. That is in corollary to the extremely far outliers in the training data plot.
* As a result we choose to fill the NaN’s/nulls with the median, that is closer to most of the current elements.

1. Answer briefly: Based on the plots you created on **(Task B)**, what pair of features is useful for predicting the spread?

Attach the [seaborn.pairplot](https://seaborn.pydata.org/generated/seaborn.pairplot.html) of only this pair of features to the report. Make sure your plots are readable and clear, and that they have proper titles, grid lines, axis labels, etc. Any missing requirement will lead to a points deduction!



We chose the plot of {PCR\_04,PCR\_09}, because it is the most sparable one. We can separate it by spiral function (function that is similar to “Archimedean spiral” function).

1. What is the time complexity of the prediction function you wrote, applied on a single test datapoint, in terms of the number of neighbors , the number of training   
   datapoints and the data dimension ? Explain. It is okay to “estimate” the complexity of python library functions. For instance, if you use np.argsort on elements, then its complexity should be . Use your reason and CS knowledge.

We will break the complexity to parts:

* Computing distances:
* Computing for m training data points the distance from a fixed point is O(d) operations per point. Totaling O(m\*d)
* The function np.argpartition:
* We estimate it has time complexity O(m) on average case, and O(m^2) in the worst case.We assume it used the quick select algorithm for finding the k-th element, then going over the rest of the elements it added only the elements less than the k-th element to the left part of the partition. Done in O(m) worst case.
* Total O(m) amortized.

Note:

* Using merge sort or any sort with O(mlogm) complexity , the task can be done in O(mlogm) worst case, instead of O(m^2) worst case with quick select.
* Taking the smallest k elements: O(k)
* Summing all elements and taking the sign: O(k)

Total:

* On average : (quick select) O(m\*d+m+k), in other words: O(m\*d) since k<=m.
* Worst case: (Sorting)) O(mlogm + m\*d +k), since k<=m, we really get : O(mlogm+m\*d)

1. Attach the figure to your report. Specify the model’s training and test accuracies.

(The plot should exhibit a bizarre behavior which we will discuss next.)

1. [↑](#footnote-ref-1)