HW1 Report

Q1

There are **1250** rows and **25** columns in the data set.

Q2

The output of `value_counts` on `conversations_per_day`:

```
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
```

We think this feature refers in the real world to the number of conversations the patient has in a day.

The feature's type is ordinal because its values are natural numbers, thus they can be treated as categories but they also have a natural order - the number of conversations per day can be compared and ranked. For example, 1 conversation per day is less than 2 conversations per day.

Q3

Features Description Table

Feature name	Description	Type
patient_id	the unique identifier of the patient	categorical
age	the age of the patient (in years)	ordinal
sex	the gender of the patient (M - male, F - female)	categorical
weight	the weight of the patient (in kg)	continuous
blood_type	the blood type of the patient	categorical
current_location	the geographic coordinates of the patient's current location (latitude, longitude)	other
num_of_siblings	the number of siblings the patient has	ordinal
happiness_score	the happiness level of the patient, on a scale of natural numbers	ordinal
household_income	the yearly income of the patient's household, in hundred thousands (ILS)	continuous
conversations_per_day	the number of conversations the patient has in a day	ordinal
sugar_levels	the blood sugar level of the patient	continuous
sport_activity	the sports activity level of the patient, measured by the number of workouts per week	ordinal
pcr_date	the date the patient took the PCR test	ordinal
PCR_01	the expression measurement of a specific virus gene (1)	continuous
PCR_02	the expression measurement of a specific virus gene (2)	continuous
PCR_03	the expression measurement of a specific virus gene (3)	continuous
PCR_04	the expression measurement of a specific virus gene (4)	continuous
PCR_05	the expression measurement of a specific virus gene (5)	continuous
PCR_06	the expression measurement of a specific virus gene (6)	continuous
PCR_07	the expression measurement of a specific virus gene (7)	continuous
PCR_08	the expression measurement of a specific virus gene (8)	continuous
PCR_09	the expression measurement of a specific virus gene (9)	continuous
PCR_10	the expression measurement of a specific virus gene (10)	continuous

Q4

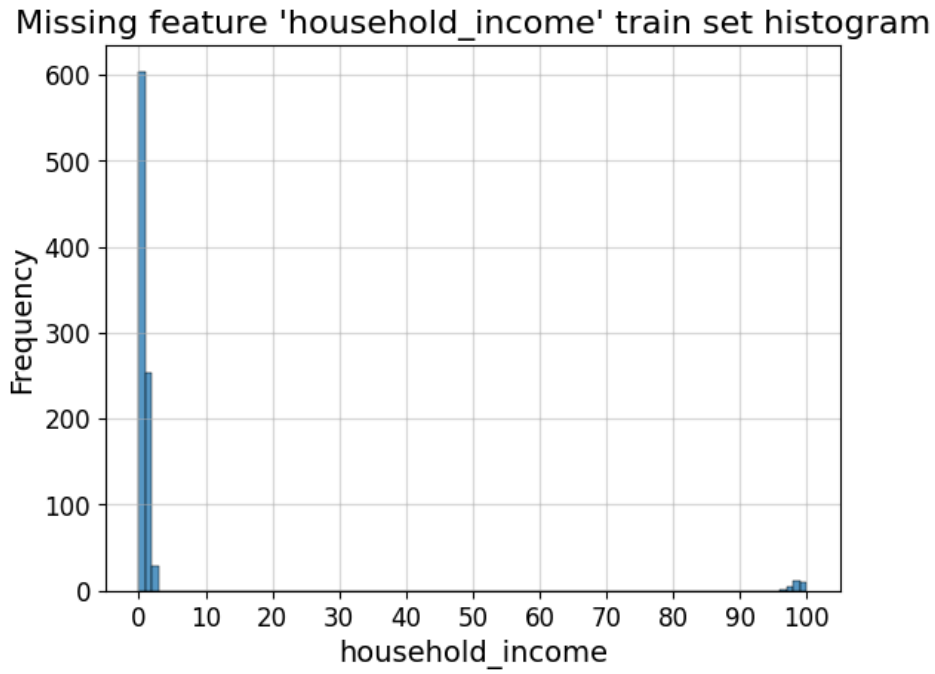
It is important that we use the exact same split for all our analyses to ensure consistency - each analysis results, conclusions, and hyperparameters should be made on the same train set (additionally, we want the train set to be the same in every code run). For example, we fill in missing values with the mean/median from the train set, evaluate the kNN model on the train set, conduct univariate/bivariate analysis on the train set, and perform data normalization (Standard and MinMax scaling) using hyperparameters calculated from the train set.

Q5

Fields with missing values

Field name	Number of missing values in the training set	Number of missing values in the test set
household_income	87	22

Q6



We can recognize outliers: there is a small number of values in the range [95,100], while most of the values are in the range [0,2.5].

Q7

train set mean and median of fields with missing values

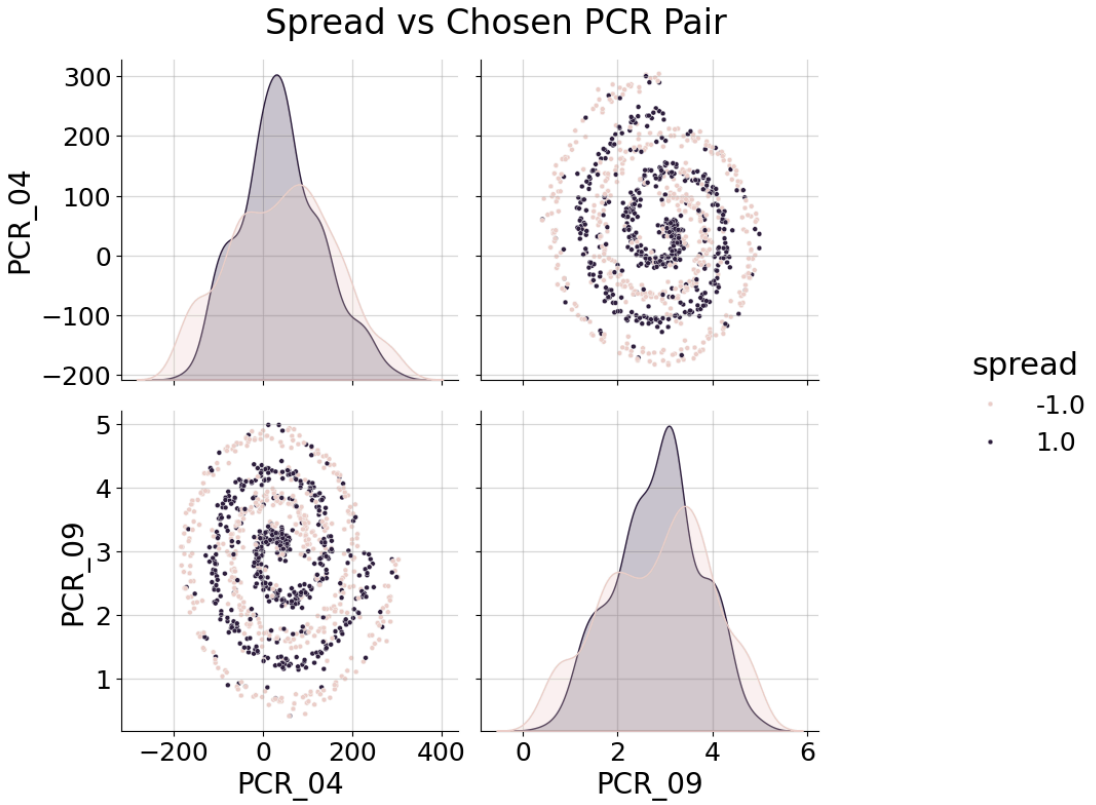
Field name	Mean	Median
household_income	3.64	0.7

There is a significant difference between the mean and the median because of the outliers that shift the mean strongly towards the higher side.

In our case, we prefer filling with the median to only account for most of the incomes in the normal range.

Q8

PCR_04 and PCR_09 are useful for predicting `spread` because the scatter of data points of the same `spread` is dense and data points of different `spread`s are separable in polar coordinates (due to the spiral patterns).



Q9

Time complexity analysis:

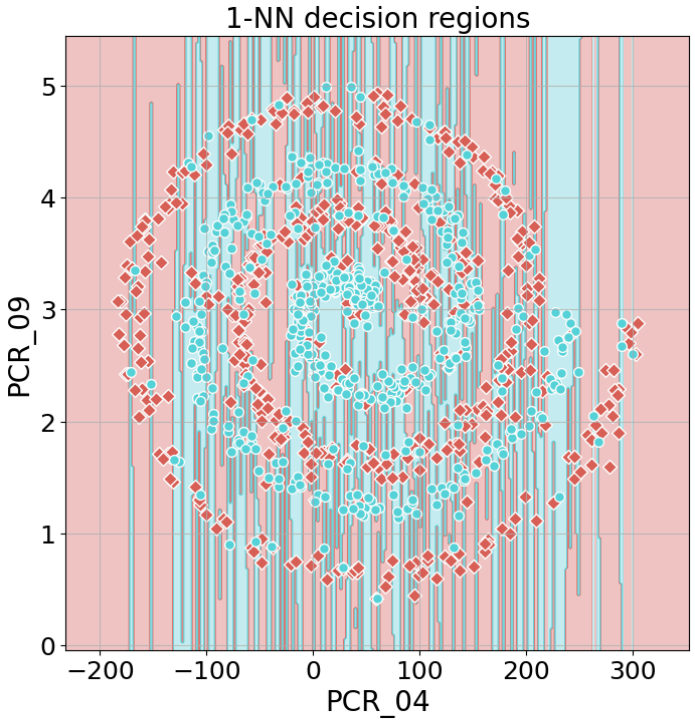
1. computing the distance of a test datapoint to one train datapoint takes O(d) time since d is the data dimension. Thus computing the distances of the m training datapoints to the test datapoint takes O(md) time.
2. finding the k neighbors with the smallest distance to the test point takes O(m) time because np.argpartition uses the partition algorithm.
3. getting the labels of those k nearest neighbors takes O(k) time.
4. computing the majority vote of the k nearest neighbors labels takes O(k) time.

Overall the time complexity of the prediction of a single test point is:
O(md)+O(m)+O(k)=**O(md)** (since k<=m)

Q10

The training accuracy is: 100%

The test accuracy is: 62.4%

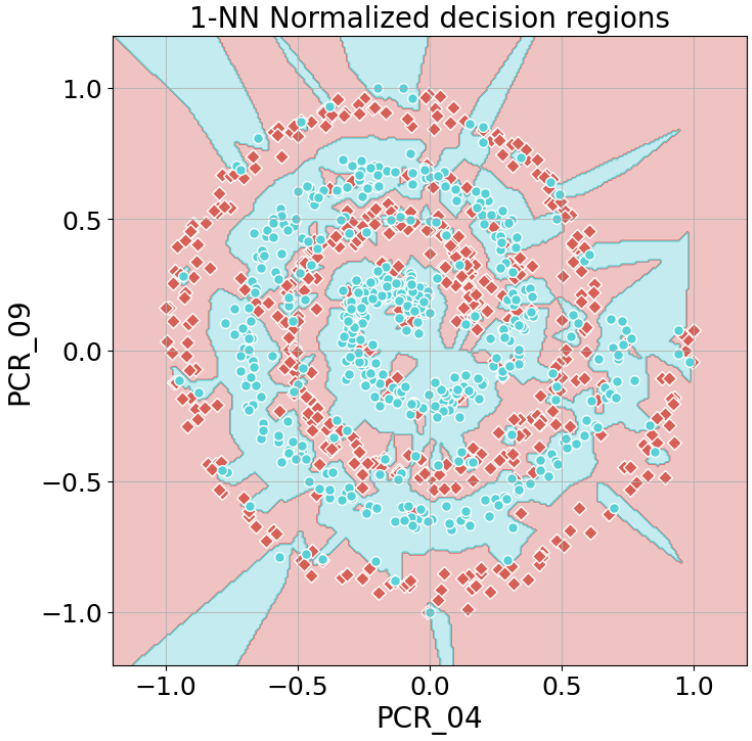


Q11

After normalizing features `PCR_04` and `PCR_09`:

The new training accuracy is: 100%

The new test accuracy is: 72.4%



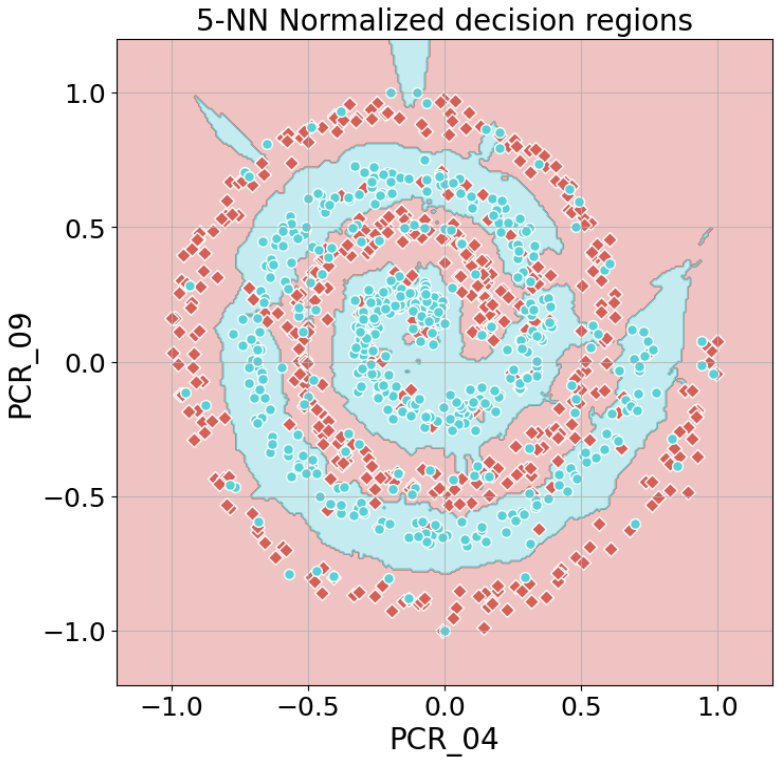
compared to Q10:

- training accuracy: the same (100%) because the 1 nearest neighbor to a train datapoint is the train datapoint itself thus the prediction is the true label of the train datapoint.
- test accuracy: increased because after normalization, both features are on the same scale, whereas the original `PCR_04` was on a much larger scale (range [-200,310]) than `PCR_09` (range [0,5]), thus `PCR_04` dominated at the distance calculation. After normalization, both features contribute equally to the distance calculation which can improve test accuracy.
- decision regions: more clear and defined. Following the explanation from the previous point, the decision boundaries without normalization were skewed towards `PCR_04`. After normalization, both features contribute equally to forming the decision regions and the separation between the `spread` classes is clearer.

Overall, we can see from these results why normalization is important for the nearest neighbor algorithm - it minimizes the dominance the larger scale features have on the distance calculation, by bringing the features to the same scale.

Q12

Using the normalized dataset and kNN model with k=5:
The training accuracy is: 84.6%
The test accuracy is: 80.4%



compared to Q11:

- training accuracy: lower because 1-NN accuracy for the training dataset is always 100% (as we explained in Q11), while in 5-NN, the 5 nearest neighbors can have a different majority label than the true train datapoint label. Thus, for some train data points, 5-NN is wrong which decreases the training accuracy.
- test accuracy: higher because 5-NN considered multiple neighbors (compared to 1-NN which considers only one). This reduces overfitting and sensitivity to noise in the training set, which leads to a more general classifier.
- decision regions: there are fewer regions, they are more defined, and they match the spiral patterns better. Following the explanation from the previous point, 5-NN decision regions are more general, and the decision boundaries are less sensitive to individual noisy data points.

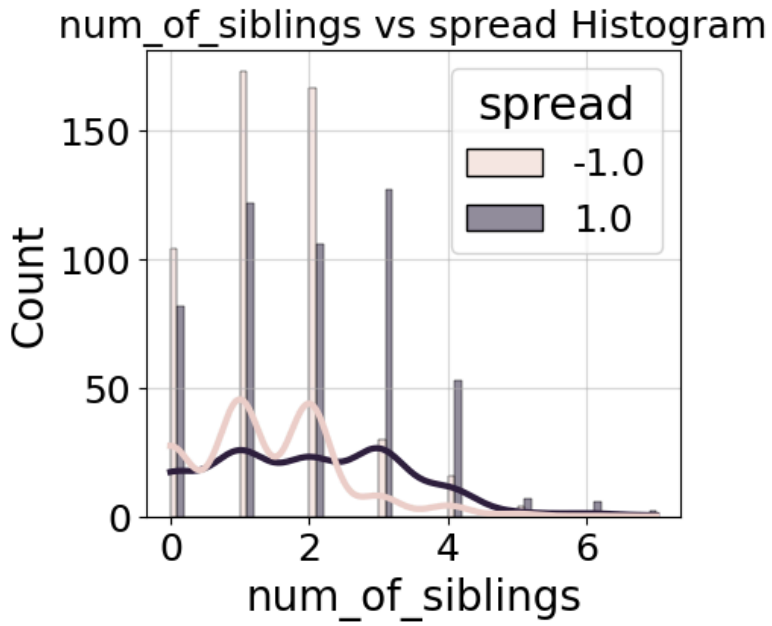
We can see from the results the effect of k on the decision regions - when k is low (as we saw for k=1), there are many small fragmented decision regions due to the sensitivity to individual data points. On the other hand, when k is larger (as we saw for k=5), there are larger, fewer, and more defined regions because taking into consideration more neighbors averages out the noise and leads to a more generalized model.

Q13:

Normalizing both features using min-max scaling is a bad idea because while the uniformly distributed feature is suited for min-max scaling, the chi-squared distributed feature is not. That is because in chi-squared distribution with k=2, most of the sampled points will be near the mean, but a few large outliers can exist. These outliers will cause the other points to be compressed into a narrow range after min-max scaling. This distorts the original chi-squared distribution and can lead to less informative data.

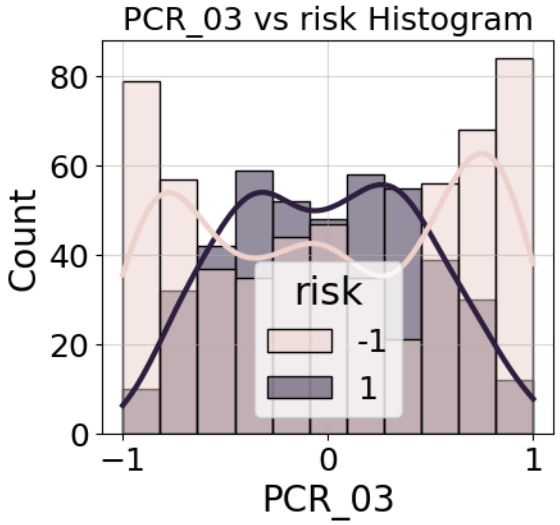
Q14:

`num_of_siblings` seems informative in predicting the `spread` because it shows the best difference between the heights of the low-spread bars to the high-spread bars. In the range of 0-2 siblings, the low-spread bars are much higher, while in the range of 3+ siblings, the high-spread bars are much higher.



Q15

`PCR_3` seems informative in predicting the `risk` because it shows the best separation between low-risk and high-risk curves. They have different peaks - the low-risk curve is higher towards the start and end of the PCR_03 range, while the high-risk curve is higher in the center.



Q16

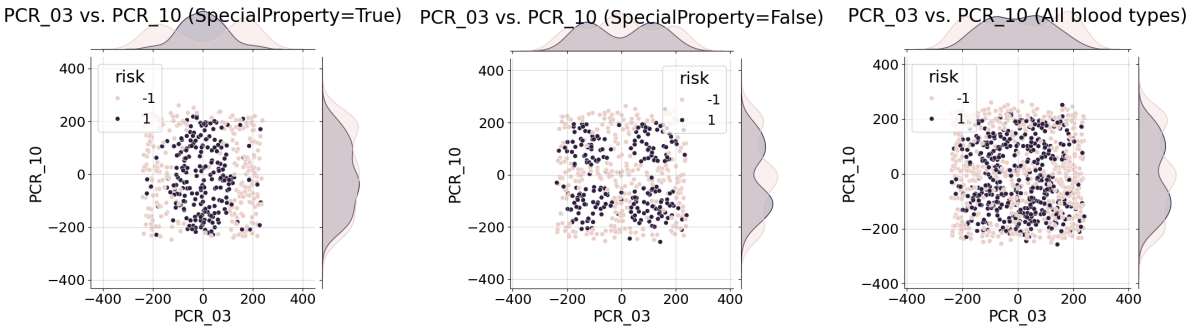
The pair **PCR_03 and PCR_10** look useful in predicting the `risk` because their plot looks the most separated.

When `SpecialProperty` is true, the high-risk points are concentrated in a rectangular area in the middle, while the low-risk points are concentrated in two rectangular areas outside both sides of the previous rectangular.

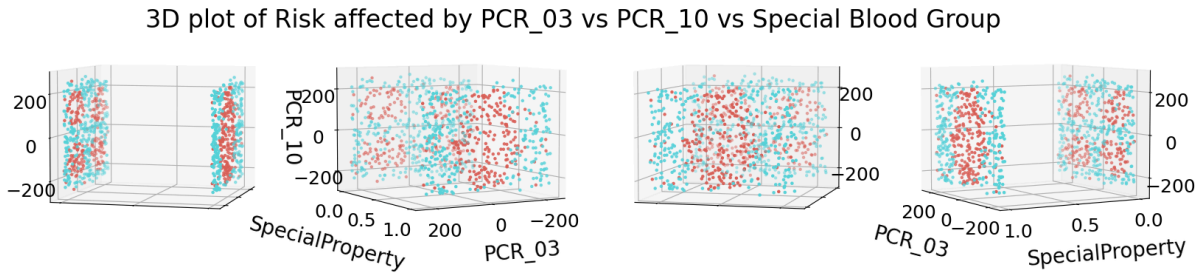
When `SpecialProperty` is false, the high-risk points are concentrated in 4 areas while the low-risk points are concentrated in the areas between them.

Q17

The Joint Plots:



Q18



Q19

Decision tree of `max-depth=3` will not fit the training data very well. As we observed and explained in Q16, and can also be seen from the join plots, the first split in the tree can be according to `SpecialProperty`. For `SpecialProperty`=True, the next two splits can be according to `PCR_03` thresholds (since the plot shows a distinct rectangle area of points of high risk). For `SpecialProperty`=False, we need more than two splits to create decision regions that capture the 4 distinct areas where the points are of high risk. Two splits only will lead to the misclassification of many points with `SpecialProperty`=False.

Q20

Decision tree of `max-depth=30` will fit the training data very well. Now we can extend the tree we described in Q19, and overcome the problem we had for points

with `SpecialProperty`=False (depth 30 suffices to capture the 4 distinct areas where the points are of high risk).

Q21

1-NN model will not fit the training data well. This is because the `PCR_03` and `PCR_10` features are on a much larger scale ([-260,260]) than `SpecialProperty` (can be treated as 0 and 1). Thus, they will dominate in the distance calculation, making the impact of `SpecialPropery` negligible. By that, we will lose the advantage of the separation achieved by considering the `SpecialProperty` value (as can be seen in Q17 joint plots). As we can see from the scatter plot of all blood types, the nearest neighbor of many train datapoints is a datapoint of the opposite risk class thus they will be misclassified.

Q22

The effect of normalization on the answer to Q19:

A decision tree of `max-depth=3` will fit the training data similarly / a little worse than in Q19. This is because the initial split can still be according to `SpecialProperty`, but `PCR_03` and `PCR_10` were originally on a large scale ([-260,260]), thus squishing their values to a small scale can lower the quality of the thresholds (their ability to separate the risk classes) in the next splits. However, since the tree has a very small depth and originally (in Q19) it did not fit the training data well, normalization may not significantly impair the fit (even more).

The effect of normalization on the answer to Q20:

A decision tree of `max-depth=30` will fit the training data less. Following the explanation for depth 3, the negative impact on the quality of the splits will be much stronger now (since originally in Q20 the tree fit the training data well). Thus the level of overfitting (to the train set) will be lower than what was achieved in Q20 (thanks to the large depth).

The effect of normalization on the answer to Q21:

1-NN model will fit the training data better because scaling to the same scale will reduce the dominance `PCR_03` and `PCR_10` have on the distance calculation over `SpecialPropery`. This will aid in minimizing the issue described in Q21, thus the fit to the training data will increase.

Q23

Data preparation summary

Feature name	Keep	New	Normalization Method
patient_id	V		
age	V		
sex	V		
weight	V		
blood_type	X		
SpecialProperty	V	V	
current_location	V		
num_of_siblings	V		
happiness_score	V		
conversations_per_day	V		
household_income	V		
sugar_levels	V		
sport_activity	V		
pcr_date	V		
PCR_01	V		Standard
PCR_02	V		Standard
PCR_03	V		MinMax
PCR_04	V		MinMax
PCR_05	V		Standard
PCR_06	V		Standard
PCR_07	V		Standard
PCR_08	V		Standard
PCR_09	V		MinMax
PCR_10	V		MinMax