Intro To Machine Learning – Final Report HW 2

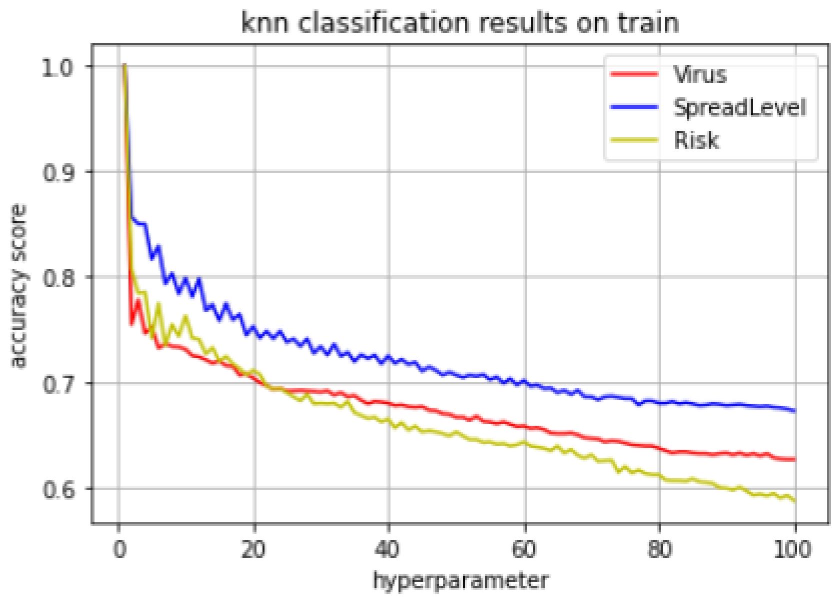
Shachar katz 313574766

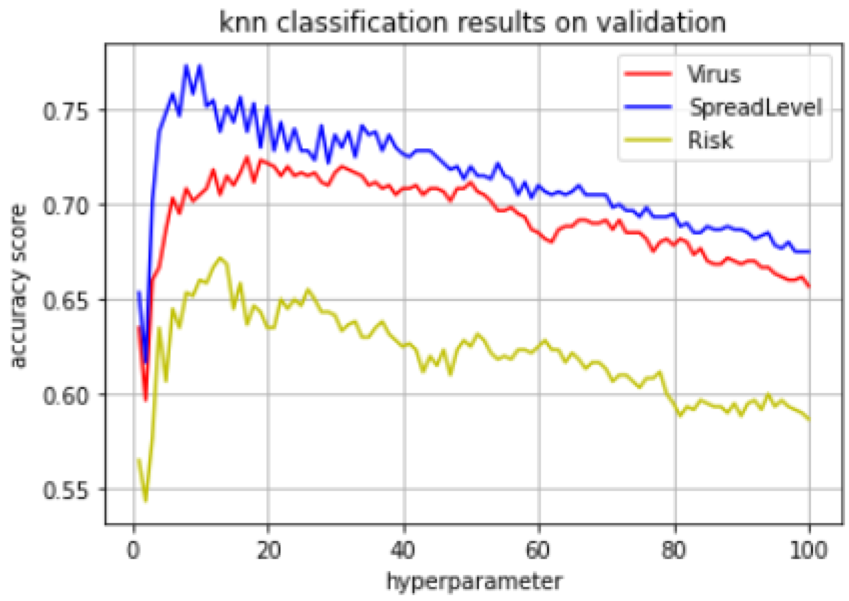
Sivan Yosebashvili 318981586

1. We did not use the validation set for the data preparation process. This is because leaking information from a validation set to a training set will result in biased results in the process of tuning hyper parameters and will therefore impair the efficiency of the classification for new data.
   We did not use the test set for the data preparation process. This is because leaking information from the test set to the training set will lead to biased results in the test process. The results we will get will not reflect the effectiveness of the classifier for new data.
   In general - using a validation set and test set in the data preparation process will cause the overfitting phenomenon. All the choices we will get in the process of adjusting the parameters will be tailored to the information from the validation set that is already in our hands and which is supposed to be according to which we examine and choose the best hyper parameters. A similar phenomenon occurs in the test process, we are likely to get higher results than expected (and we will be happy - in vain) because we are tested after practicing on a set of information that the test set is no stranger to (was part of the data preparation process).
   In conclusion, to avoid such and such undesirable effects, we did not use the validation and test set in the information preparation process.

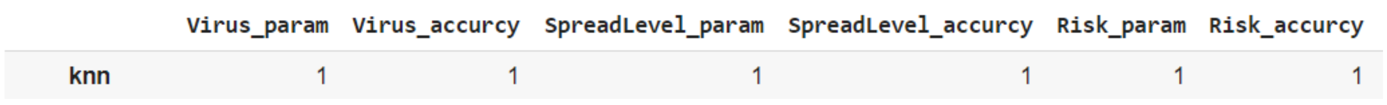
KNN

2. Accuracy results depend on the parameter K in KNN classifier:



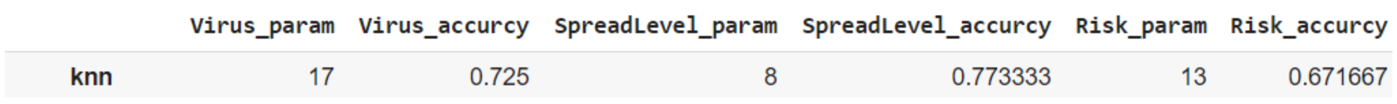


3. Train and validation accuracies of the best KNN model for each task:
   train:



	Virus_param	Virus_accurcy	SpreadLevel_param	SpreadLevel_accurcy	Risk_param	Risk_accurcy
knn	1	1	1	1	1	1

   validation:



	Virus_param	Virus_accurcy	SpreadLevel_param	SpreadLevel_accurcy	Risk_param	Risk_accurcy
knn	17	0.725	8	0.773333	13	0.671667

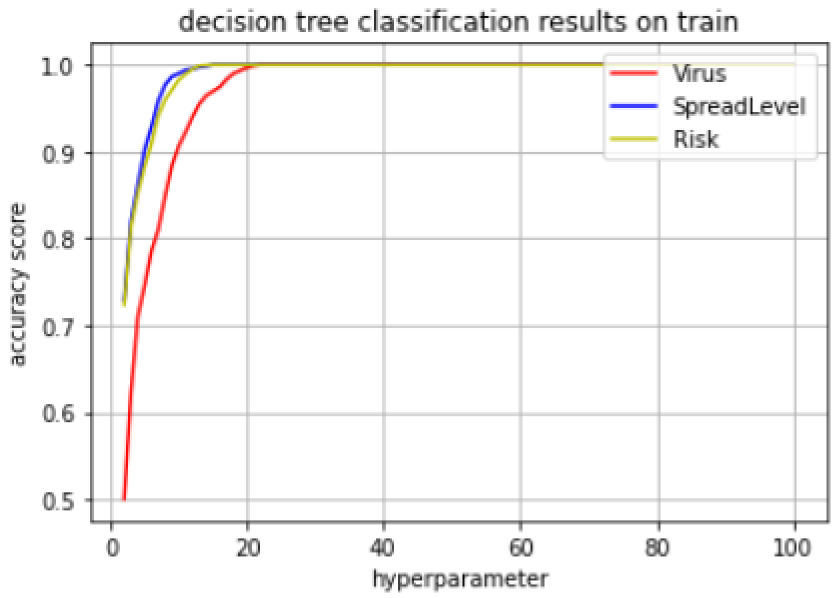
4. The slope of the graph increases up to a certain value k, then the slope decreases a bit, and from there we will swing around the same percentage of accuracy. That is, the accuracy improves to a certain point where it reaches maximum accuracy and from there goes down a little bit and remains around the same accuracy.
   A reason for this behavior, is that up to a certain value, as the parameter increases so the accuracy increases, and then we take into account too many points in space and reach a state of over-adjustment and therefore decrease accuracy.

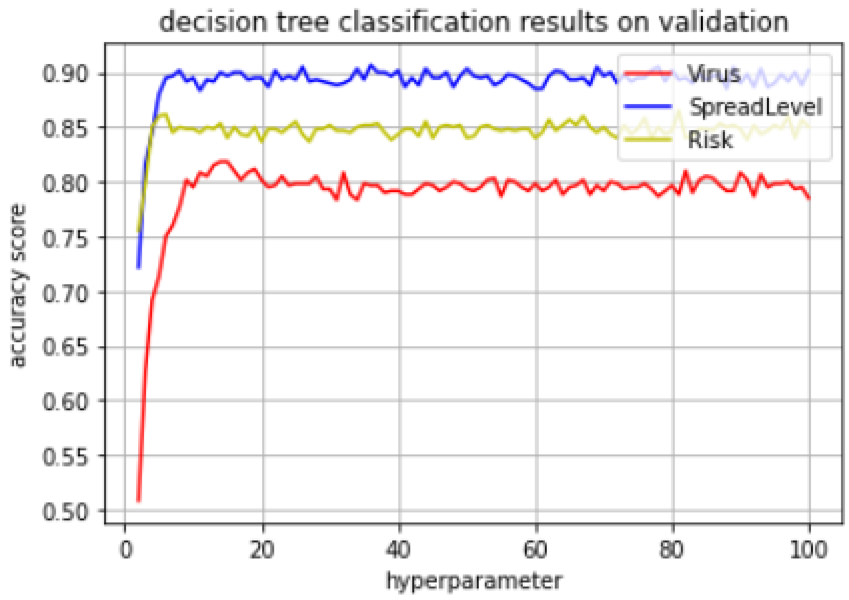
5. As we can see in the figure above, the best:
   a. virus parameter is k=17
   b. Spread level parameter is k=8 (9)
   c. Risk parameter is k=13

(We will mention it is better to choose an even parameter as K, in order to avoid undefined situations).
we built 100 KNN models (the i'th model with k=i), and each time we check what is the accuracy of that model on the validation set.

Decision Tree

6. Accuracy results depend on the parameter depth in decision tree classifier:





7. Train and validation accuracies of the best decision tree model for each task:

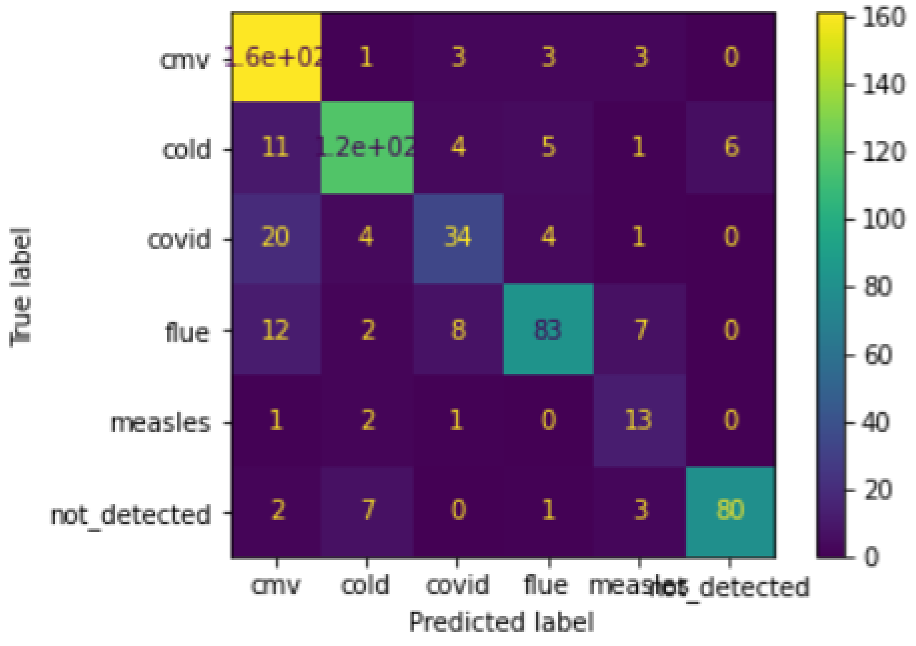
<u>train:</u>

	Virus_param	Virus_accurcy	SpreadLevel_param	SpreadLevel_accurcy	Risk_param	Risk_accurcy
knn	1	1	1	1	1	1
decision tree	23	1	15	1	15	1

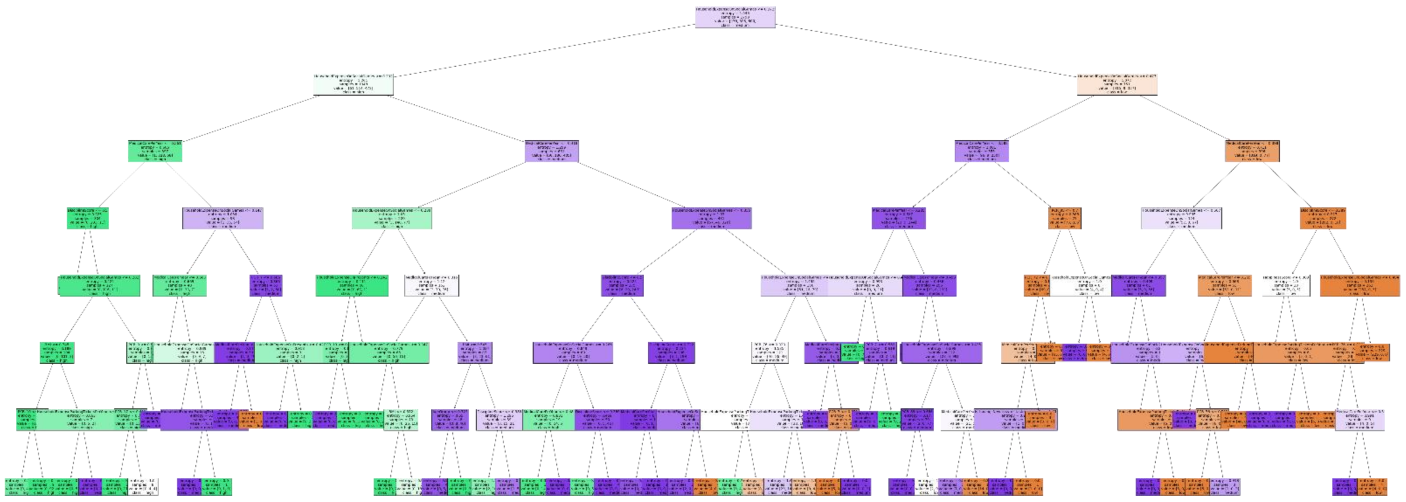
<u>validation:</u>

	Virus_param	Virus_accurcy	SpreadLevel_param	SpreadLevel_accurcy	Risk_param	Risk_accurcy
knn	17	0.725	8	0.773333	13	0.671667
decision tree	14	0.818333	8	0.91	6	0.861667

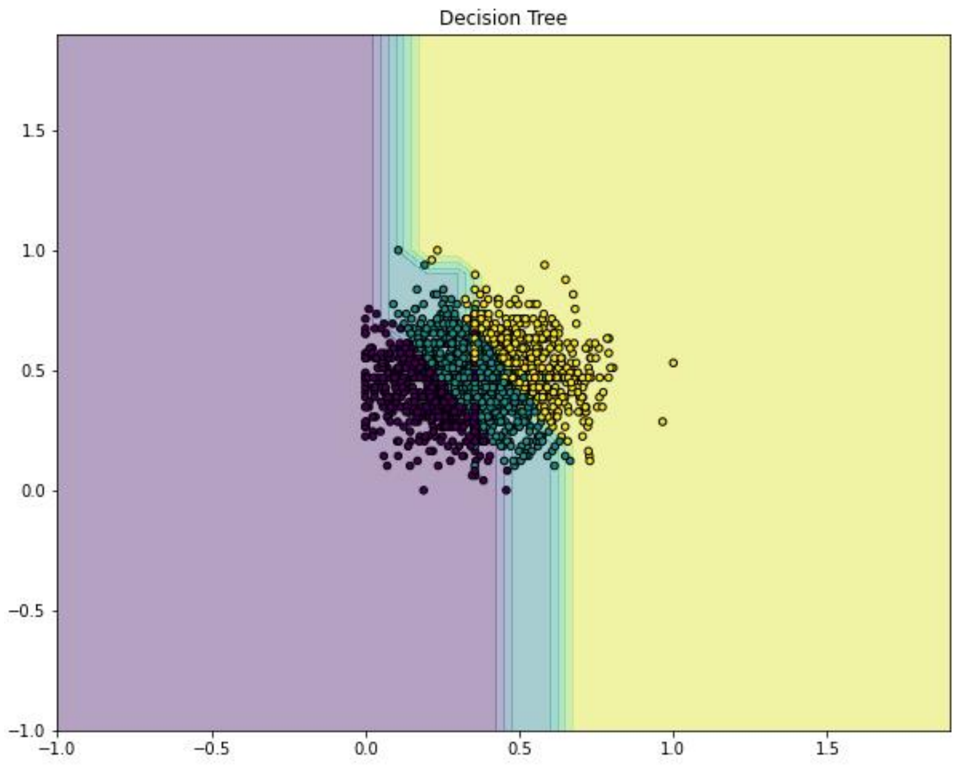
8. Confusion Matrix for the Virus classification:



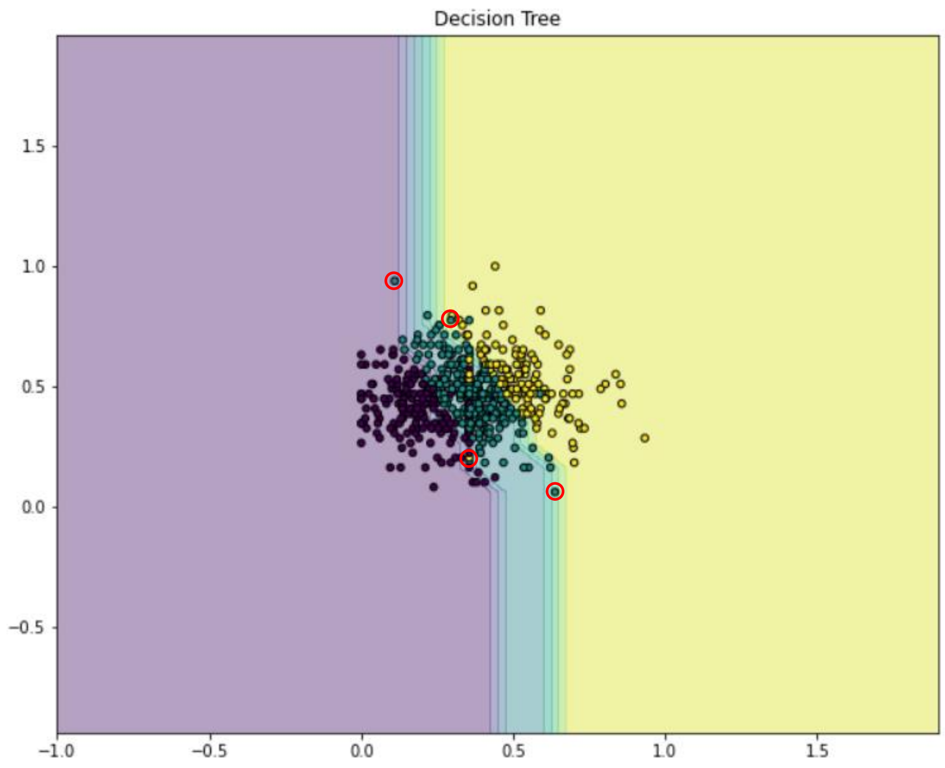
9. The tree is added in the code with option to download it and also added as a png file to the zip folder.



10. The two most important features are the once that occurs the most time as the tree's nodes (they have the max entropy in many levels, especially the firsts nodes). Those features are *HouseholdExpenseOnSocialGames* and *MedicalCarePerYear*.
Decision boundaries with the training set:



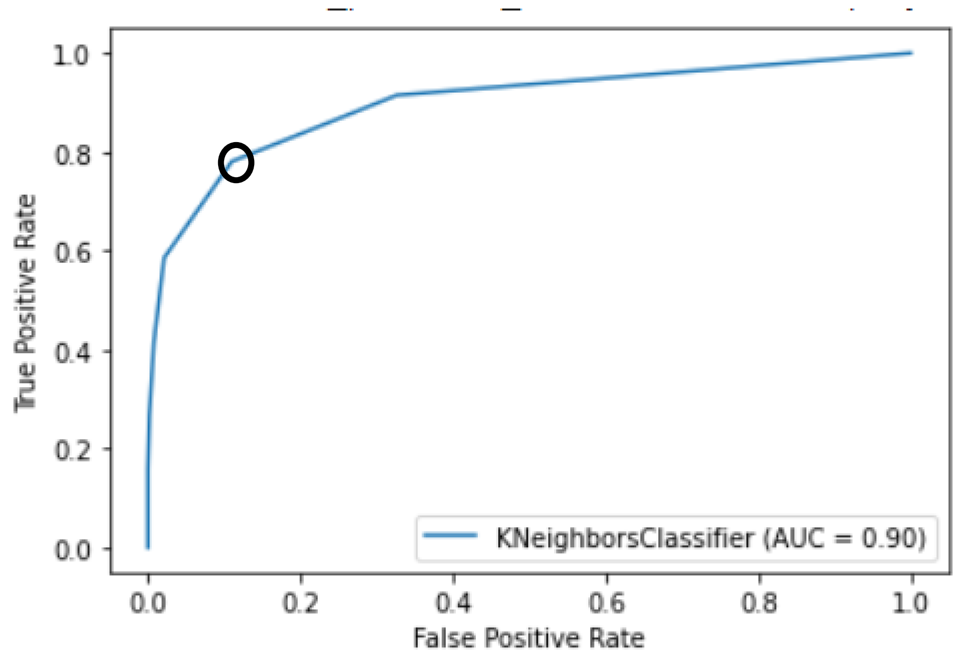
Decision boundaries with the validation set:



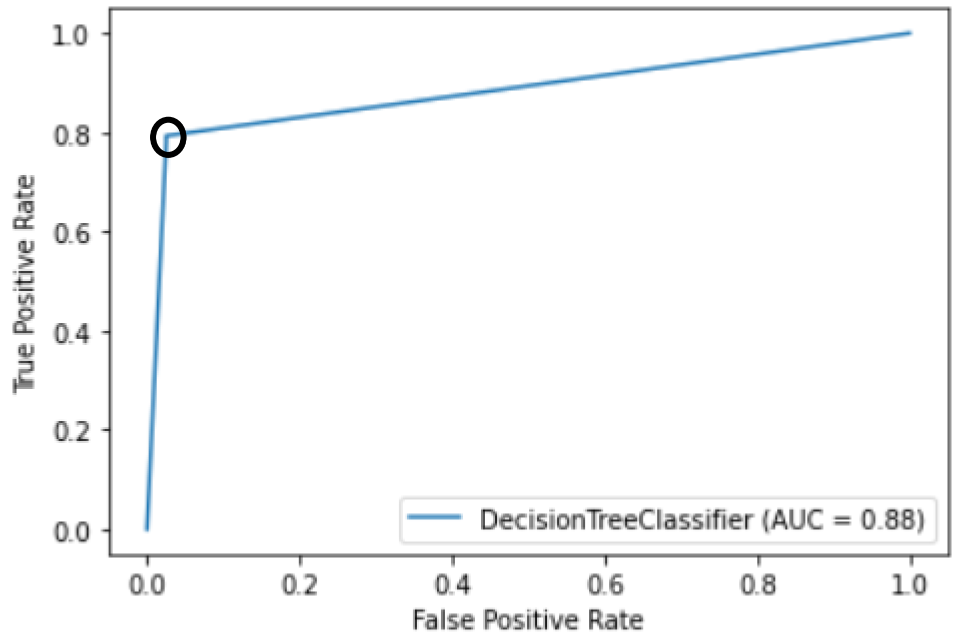
*Note that the two charts are similar in their trends. That is, the tree that trained on the training set produced the same boundaries for the validation set.

At the Decision Tree boundaries (on validation) we can see "mistakes" (marked), which shows that it is not overfit. There is not overfitting in our trees.

11. Roc curve of KNN:



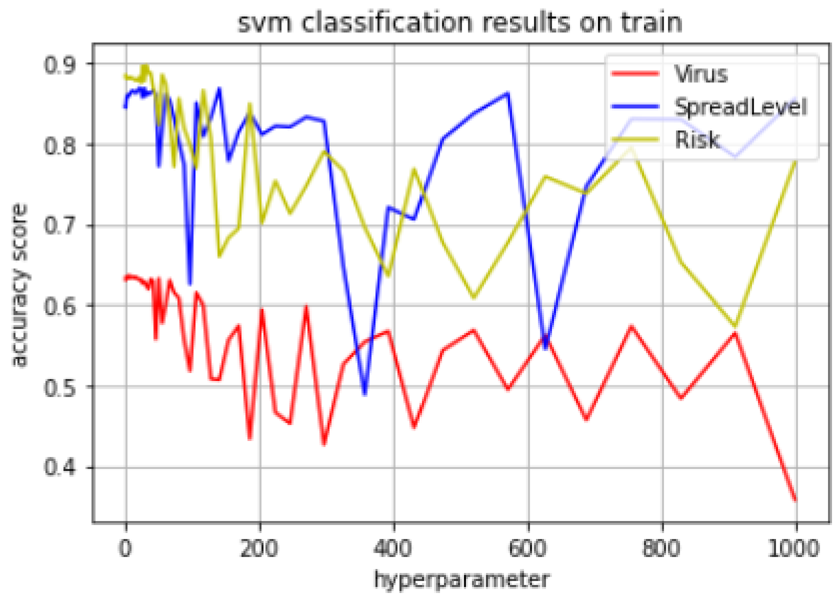
Roc curve of decision tree:

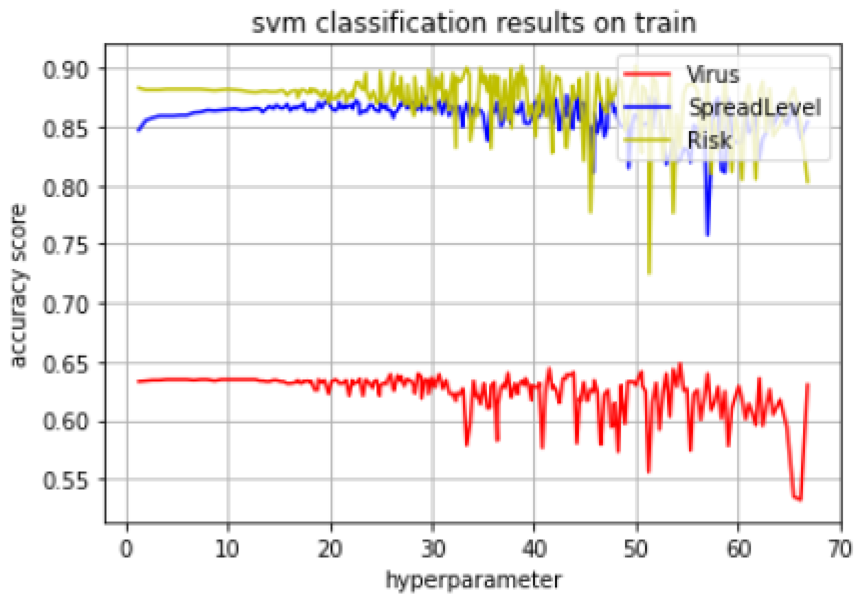


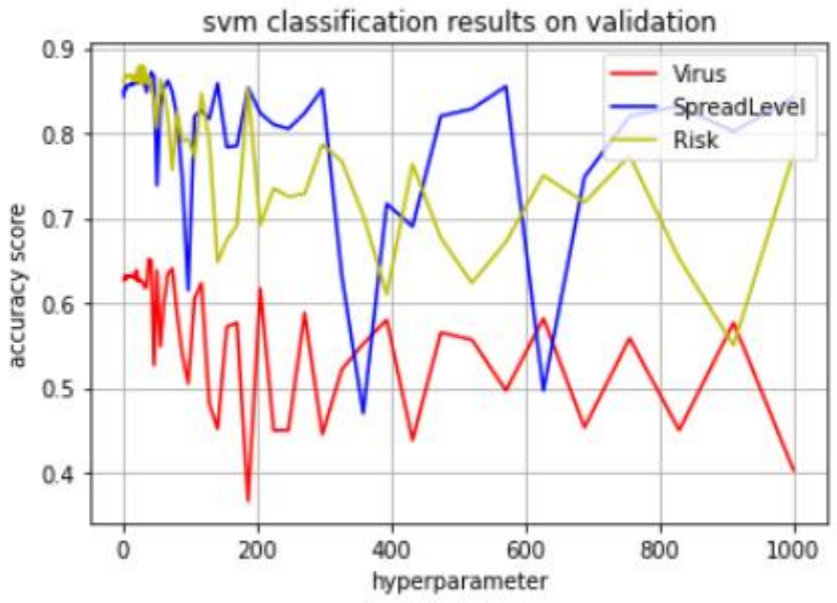
12. We know that the larger the area trapped under the roc curve, the better the situation. Therefore, selecting KNN is better.
But one can say that he will prefer to use the decision tree, because the "optimal point" in its roc curve graph it better than the KNN's (the same TP rate for lower FP rate).
Because the two AUC is almost the same, we prefer to use decision tree model.
The circles above mark the optimal TPR and FPR tradeoff, it has a high TP rate and still low FP rate.
The risk in FP is to tell someone who is not at a high level of spread level that he is at a high level of spread level. Note that FN is extremely dangerous! We have someone who thinks his level of spread is low (and therefore will behave in society accordingly) even though he is contagious.

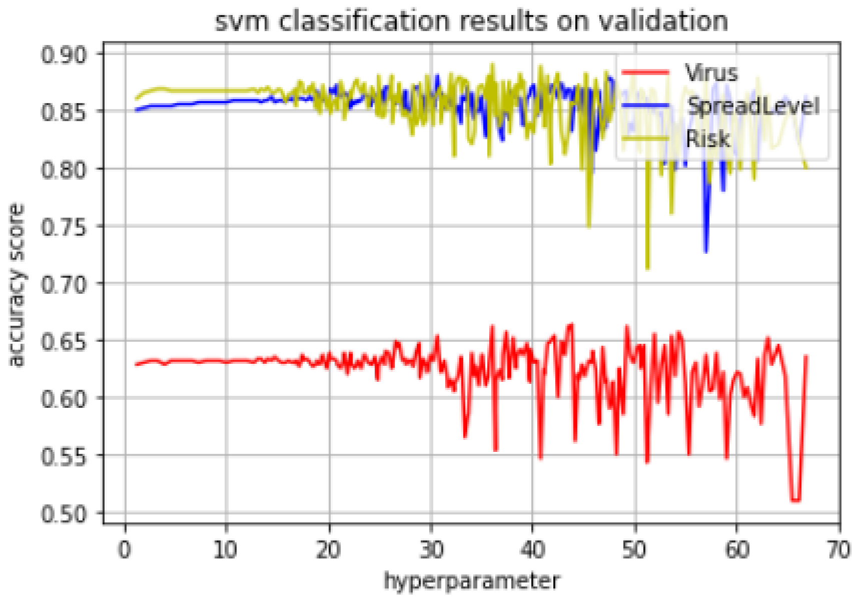
SVM

13. Accuracy results depend on the parameter C in SVM classifier:






svm classification results on validation
accuracy score
hyperparameter
Virus
SpreadLevel
Risk
0.9
0.8
0.7
0.6
0.5
0.4
0
200
400
600
800
1000


svm classification results on validation
accuracy score
hyperparameter
Virus
SpreadLevel
Risk
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0
10
20
30
40
50
60
70

14.Train and validation accuracies of the best SVM model for each task:

train

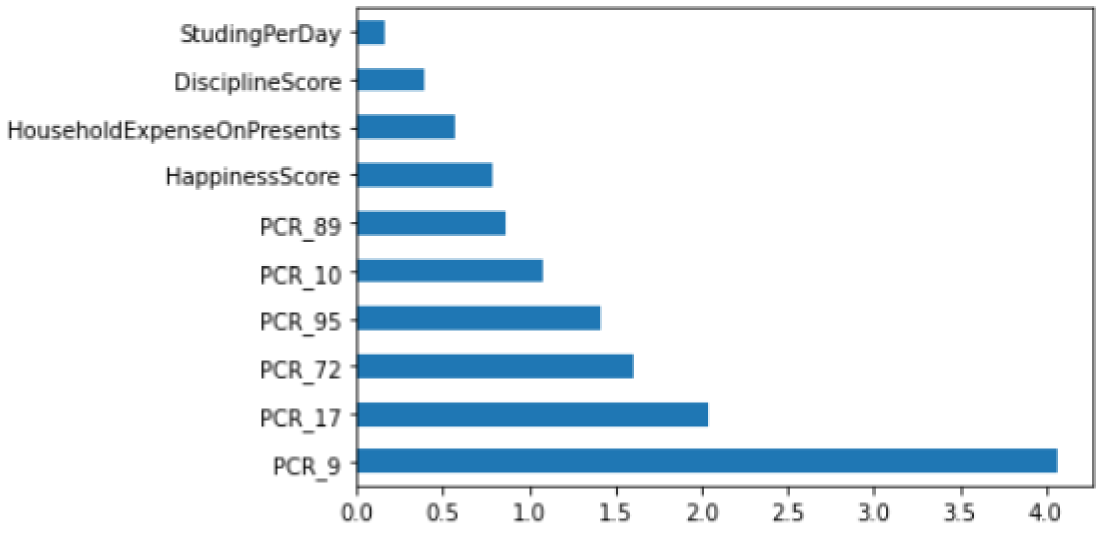
	Virus_param	Virus_accurcy	SpreadLevel_param	SpreadLevel_accurcy	Risk_param	Risk_accurcy
knn	1	1	1	1	1	1
decision tree	23	1	16	1	15	1
svm	54.3299	0.648694	37.7888	0.878266	38.7894	0.902168

validation:

	Virus_param	Virus_accurcy	SpreadLevel_param	SpreadLevel_accurcy	Risk_param	Risk_accurcy
knn	17	0.725	8	0.773333	13	0.671667
decision tree	14	0.815	58	0.91	6	0.863333
svm	43.8699	0.663333	30.6813	0.88	36.0867	0.89

15. We use coef_ method for this assignment. We print the classes of Virus target, and saw that the index of flue is 3, and there for we got:

```
['cmv' 'cold' 'covid' 'flue' 'measles' 'not_detected']
[ 0.06166557 -0.11141697 -1.43675255  0.39293448  0.78407328  0.56964065
 -0.40454662 -0.05939941  0.08540899 -1.47759068 -0.57248902  1.0834873
  2.04175785 -1.09549189 -0.11718377  1.60053188 -0.20110527 -0.33420755
  0.86134693  4.06649826  0.05200984  1.40773994  0.02417534  0.16600389]
```

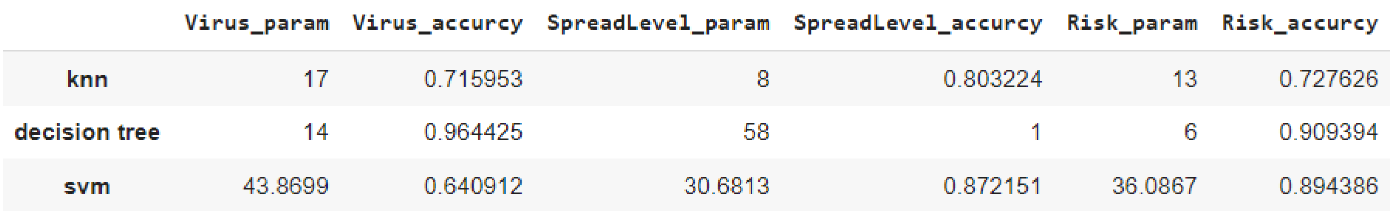


when the array above is the weight of each feature and the pot represent the most important features according to their weight. We can conclude that PCR_9 and PCR_17 are the two most important features when come to detect whether you have a flue.

16.We know that by definition, data is linearly separable iff there exist a hyperplane in the plane with all the first label points on one side of the hyperplane and all the second label points on the other side. We didn't find such hyperplane, the best accuracy of spread level we got in Linear SVM is 0.88.

So we can say that the different spread levels are not totally linearly separable, because if they were, we would probably have got higher accuracy.

17. Test accuracies for each model and for each classification task:

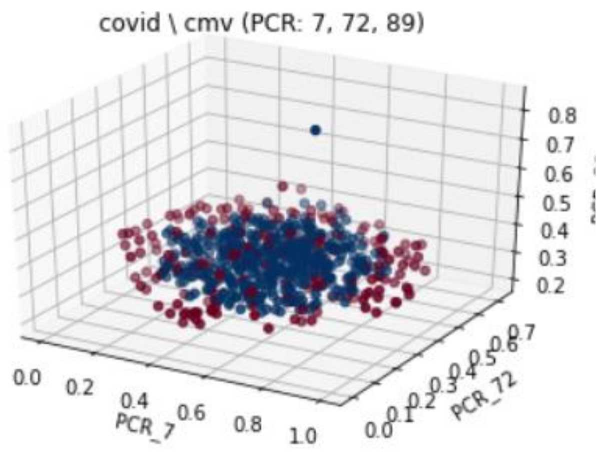


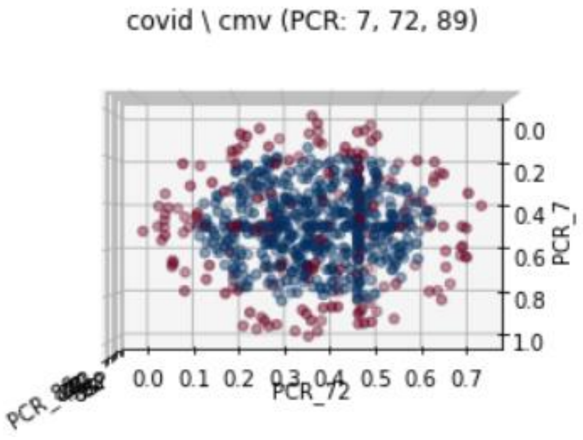
	Virus_param	Virus_accurcy	SpreadLevel_param	SpreadLevel_accurcy	Risk_param	Risk_accurcy
knn	17	0.715953	8	0.803224	13	0.727626
decision tree	14	0.964425	58	1	6	0.909394
svm	43.8699	0.640912	30.6813	0.872151	36.0867	0.894386

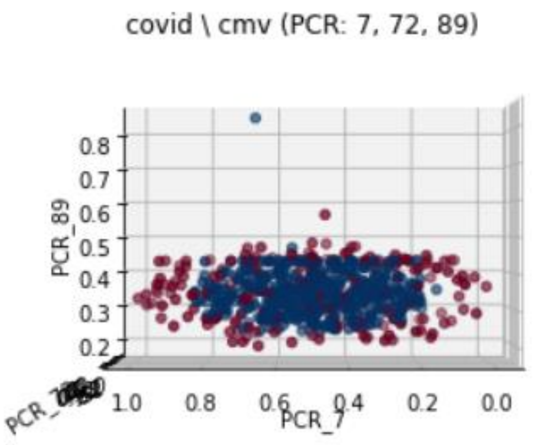
18. For Virus task the best model is decision tree (on validation the best was also decision tree).
For Spread level task the best model is decision tree (on validation the best was also decision tree).
For Risk task the best model is decision tree or KNN (on validation the best was SVM).
If a model performs well in the validation set but not on the test set, it means that the hyper parameters that were chosen in the validation section are fitting to the validation set and not to a general data set. This is overfitting (when we are making decisions based on existing data source in order to get a good score in the learning process instead of trying to "generalize" the process).

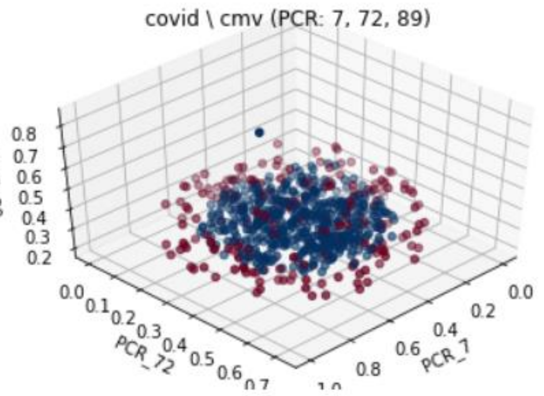
19. If we choose hyper parameters according to the test set, our classifier would make decisions based on information from the test set, which would cause the classifier's accuracy results to be skewed (and give us the illusion that our classifier achieved such and such results).
As a rule, a separation must be created between the hyper-parameter tuning phase and the classified evaluation phase in terms of the information that passes through each process, in order to produce a correct and clear picture.

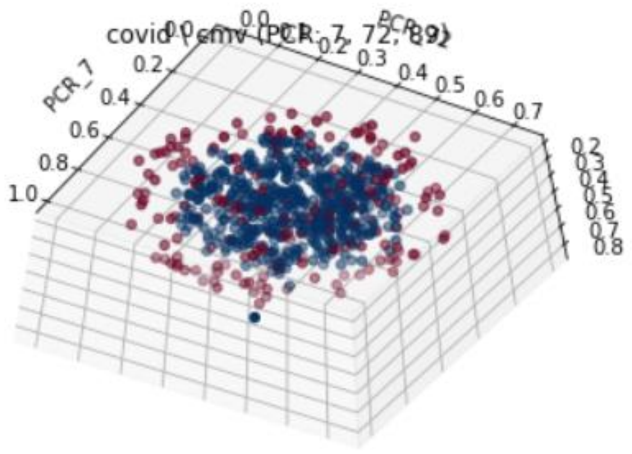
20. Plot of 'covid' and 'cmv' viruses in this feature space:





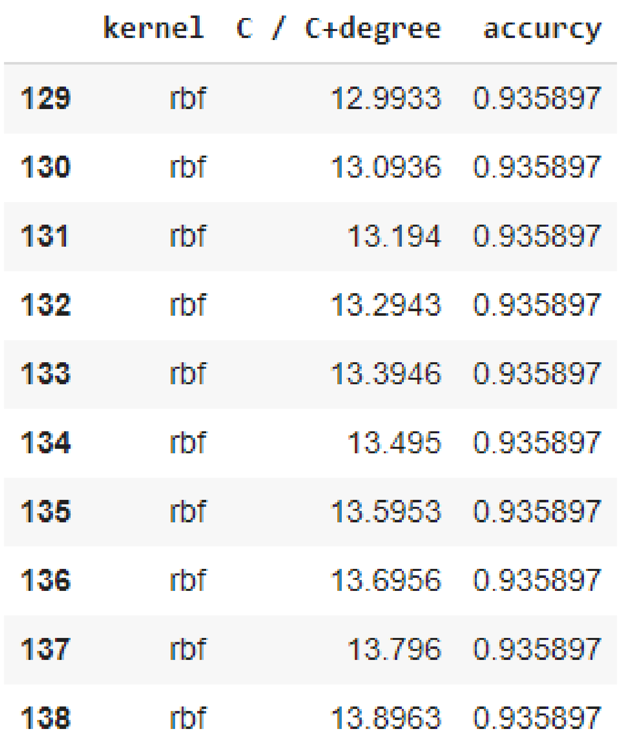






We can clearly see the connection between the two - the red dots surround the blue dots. The blue dots are grouped together in a specific area and the red dots surround them.
This explains why the results of linear separators on this category were low, they are not non-linear separable.

21. We tried to use poly, rbf and sigmoid kernels.
for each of them, we have tunes there hyperparameters and check the accuracies on the validation set. It was very clear that rbf kernel is the best out of the 3 (we have sore all the accuracies and rbf was always the best out of the 3).
it is not a surprising result according to the way the data is separate, this kernel is known as one that can separate Spherically scattered data.
the best kernels are :

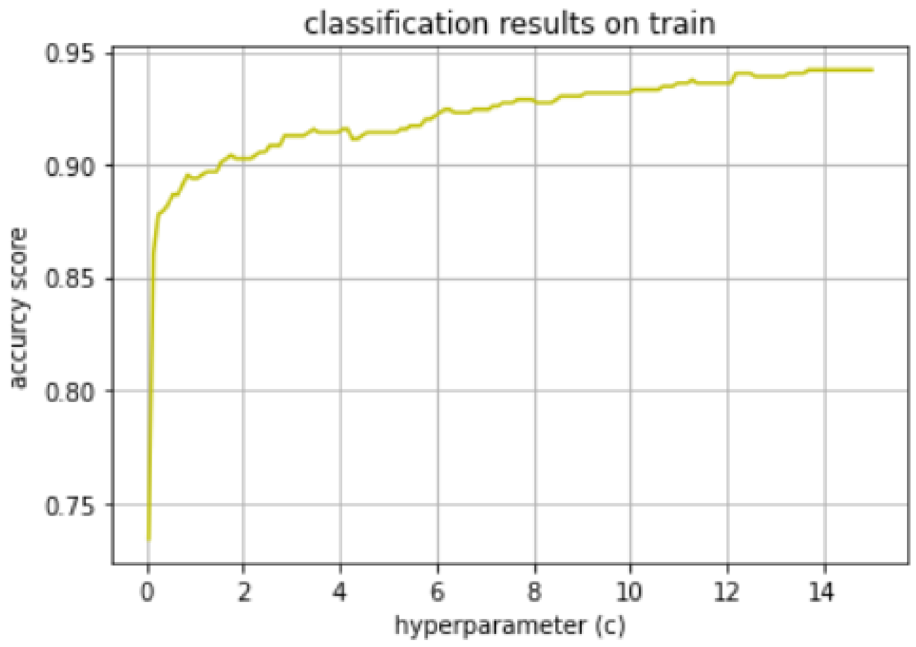


	kernel	C / C+degree	accurcy
129	rbf	12.9933	0.935897
130	rbf	13.0936	0.935897
131	rbf	13.194	0.935897
132	rbf	13.2943	0.935897
133	rbf	13.3946	0.935897
134	rbf	13.495	0.935897
135	rbf	13.5953	0.935897
136	rbf	13.6956	0.935897
137	rbf	13.796	0.935897
138	rbf	13.8963	0.935897

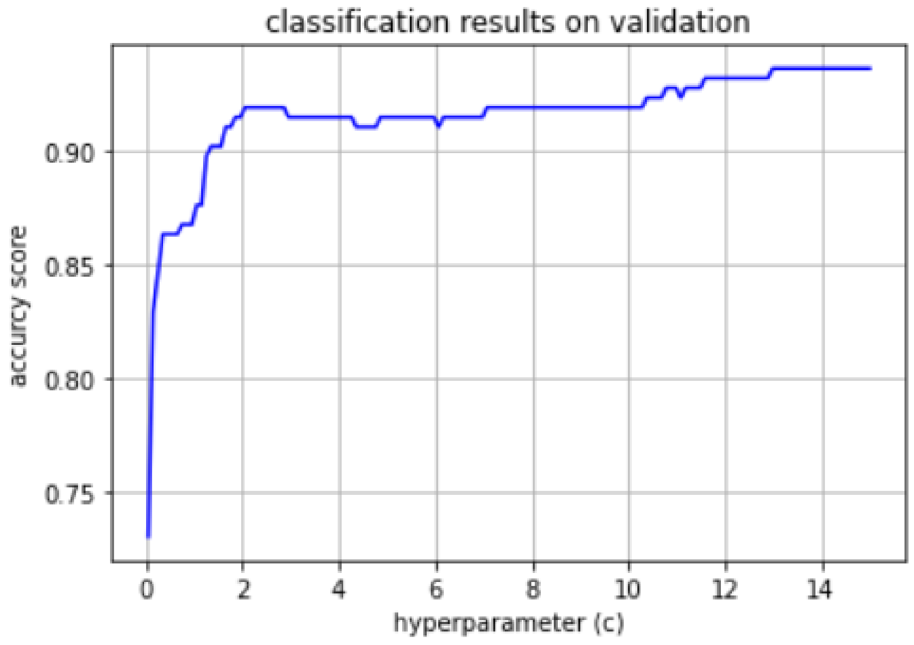
so we got a 0.935 accuracy which is very good and we can almost say that we try to make the data linearly separable !
We can also see that for many different hyperparameters we get the same accuracy, so we can infer that this is kind of "boundary" for our success rate in this target classifier.

22. Accuracy results depend on the parameters C in rbf classifier:

train:

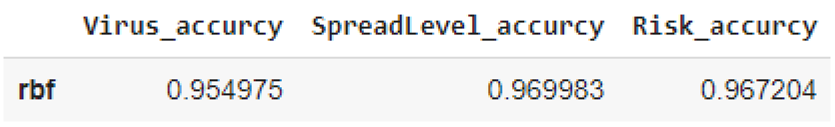


validation:



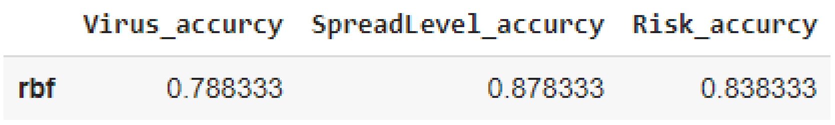
23. Validation accuracies for rbf model that train on training set for each classification task:

train:



	Virus_accurcy	SpreadLevel_accurcy	Risk_accurcy
rbf	0.954975	0.969983	0.967204

validation:



	Virus_accurcy	SpreadLevel_accurcy	Risk_accurcy
rbf	0.788333	0.878333	0.838333

We will mention, we noticed the difference in the accuracies between the train and validation. This is probably an effect of overfitting in the train process.

Part 6 – custom model's challenge:

We chose to participate in the competition. For the 3 targets we chose to use decision tree classifiers due to the result of the previous parts where we saw the highest classification results with this type of classifier (better than KNN and SVM). In order to make our classifiers as precise as we can we chose to use all of our data (train + validation + test) and to use k-cross-validation (k-fold, where k=5) in order to tune our hyperparameters.