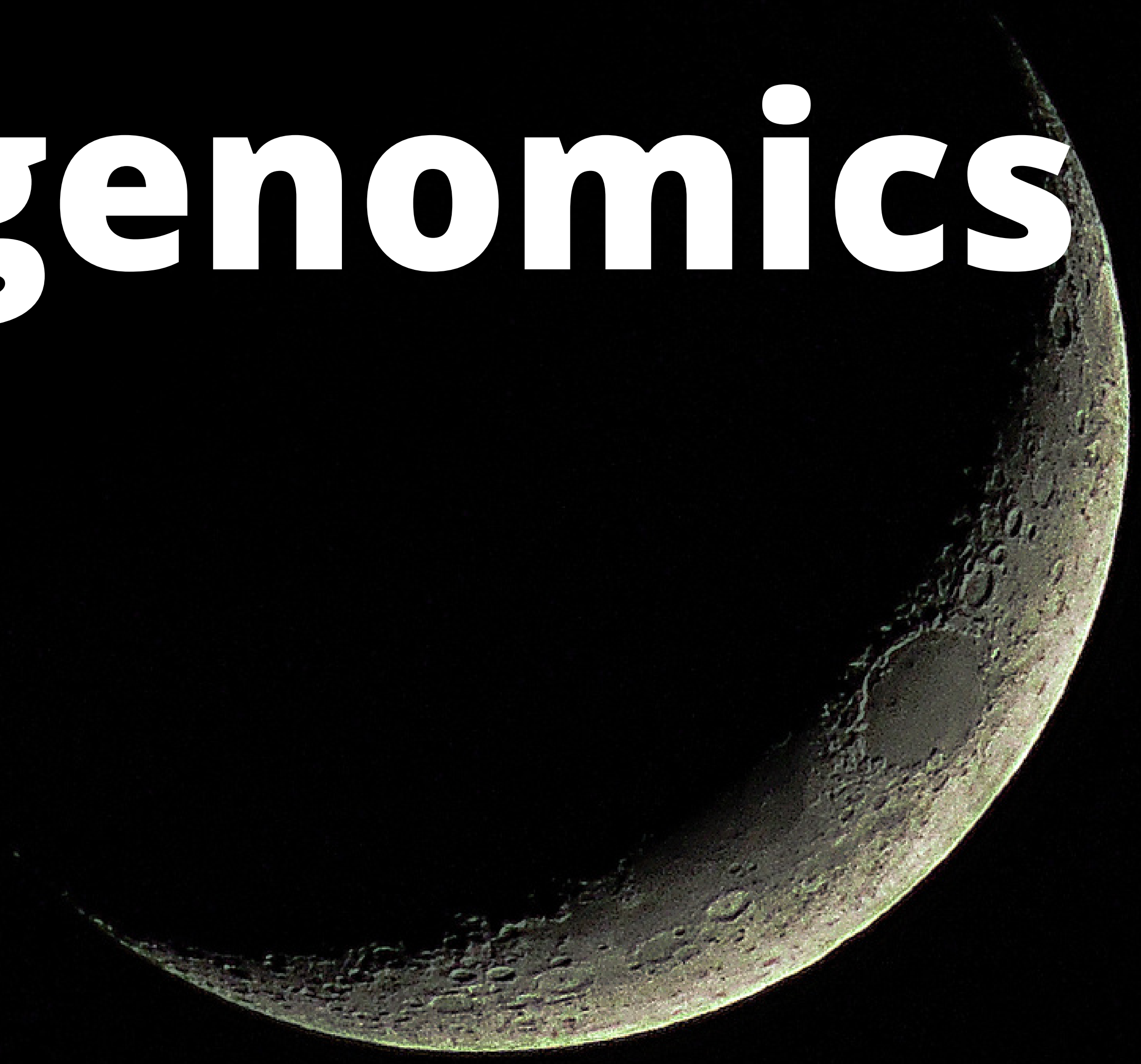
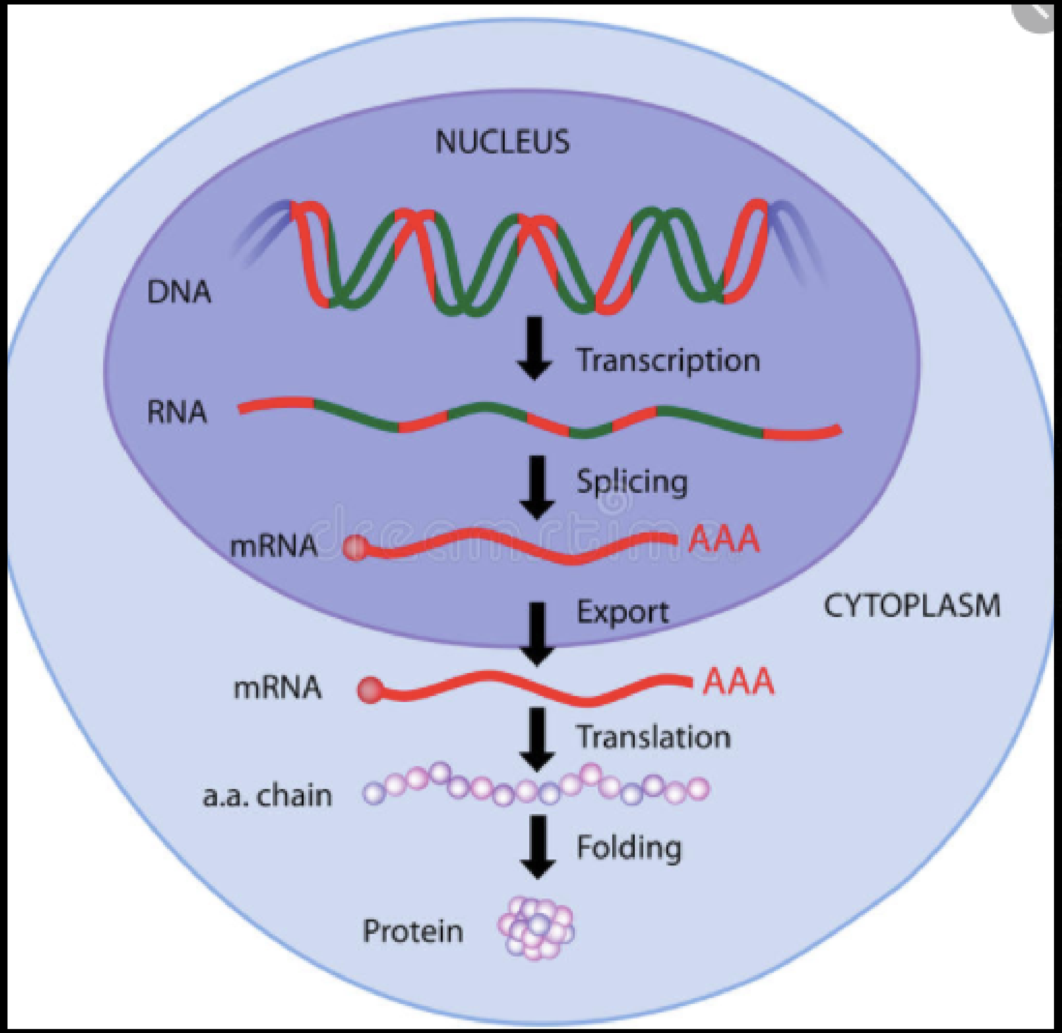
Cross-over bioimaging and genomics



CONSENSUS CLASSIFIER

Gene Expression

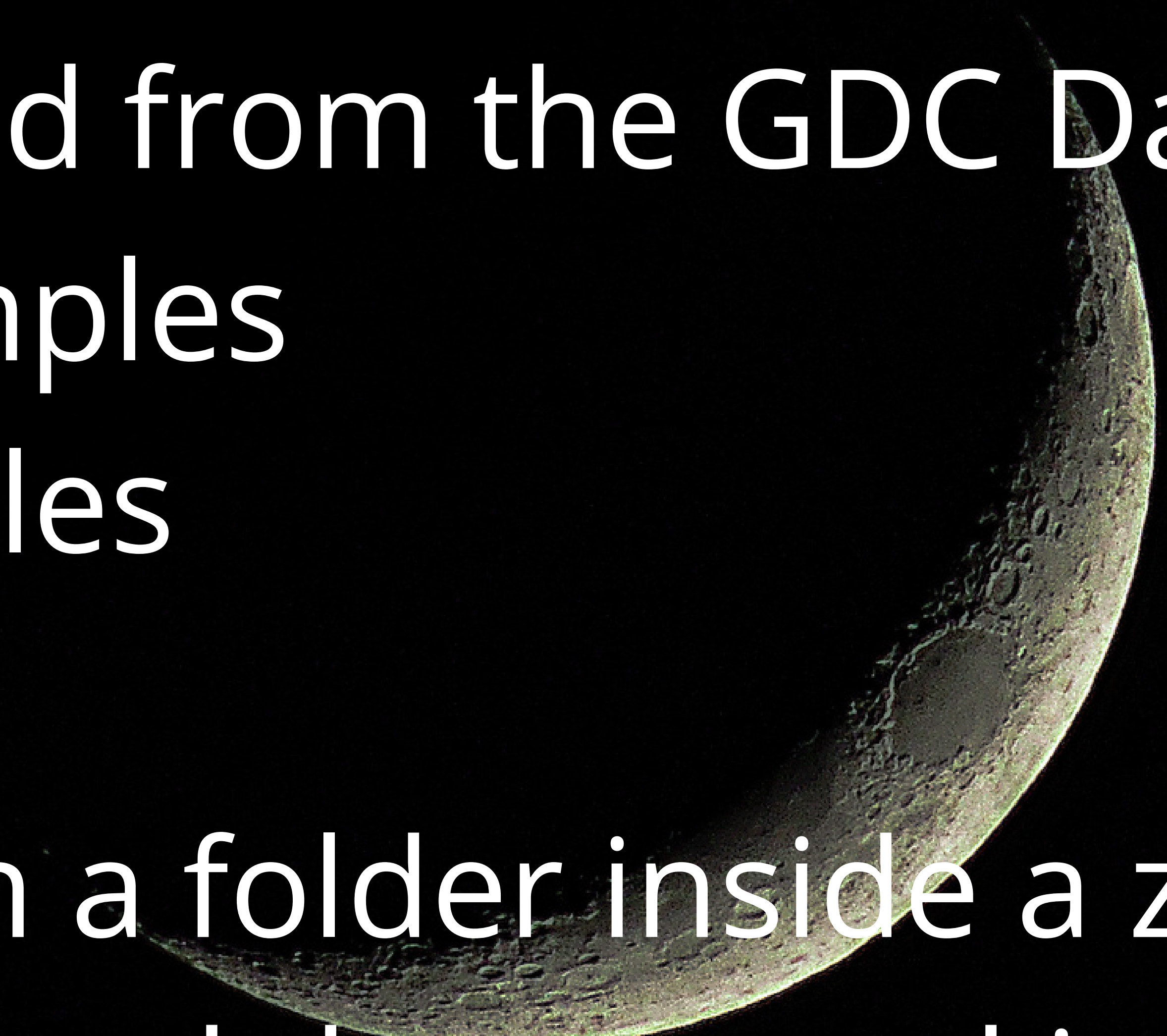
The gene expression is the process by which information is transfered from DNA to RNA (Transcription) and from RNA to protein (Translation).



Gene expression quantifies with real numbers the expression of some genes.

The data analyzed in this project are the gene expression of mRNA

Data Preparation



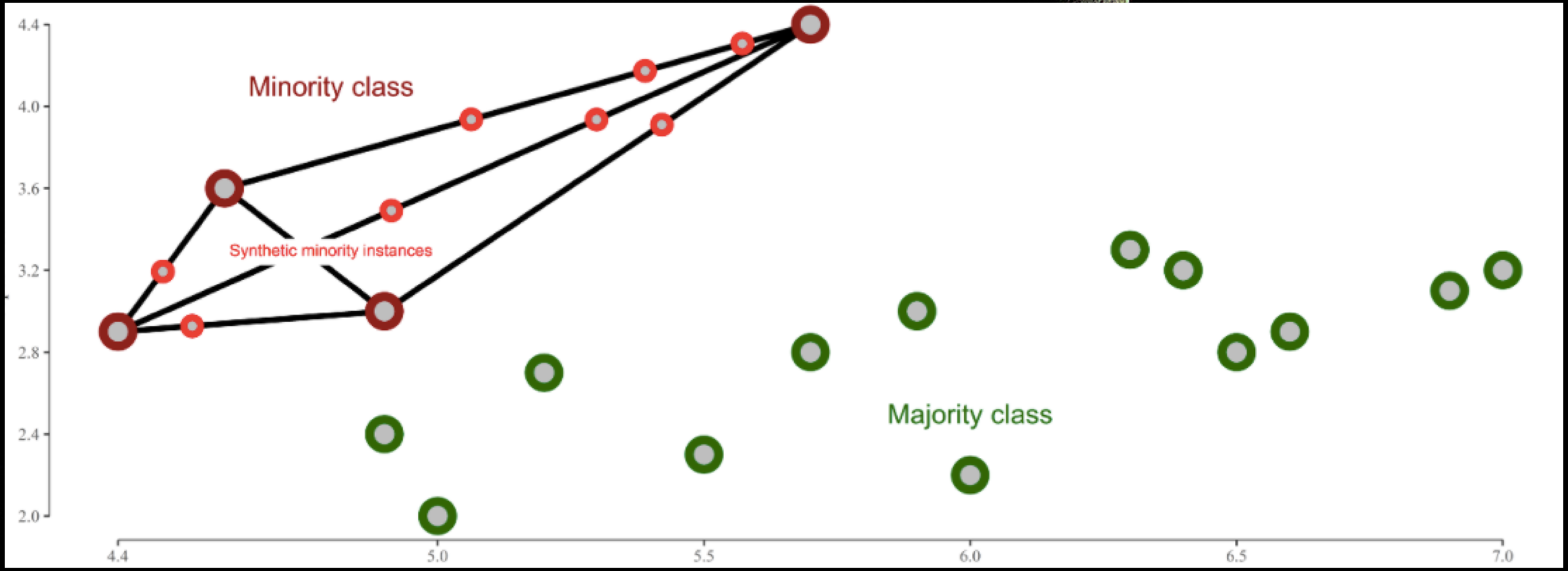
Data are downloaded from the GDC Data Portal.

- 682 tumoral samples
- 42 healthy samples
- 60483 features

All data are stored in a folder inside a zip folder, so data are firstly extracted and than stored in a pandas dataframe structure, and also the labels of these samples are added to this dataframe.

Balance Dataset

Data are balanced augmenting the healthy samples.
The technique used is the so called SMOTE oversampling.



Data Preprocessing

MinMax Scaling PREPROCESSING

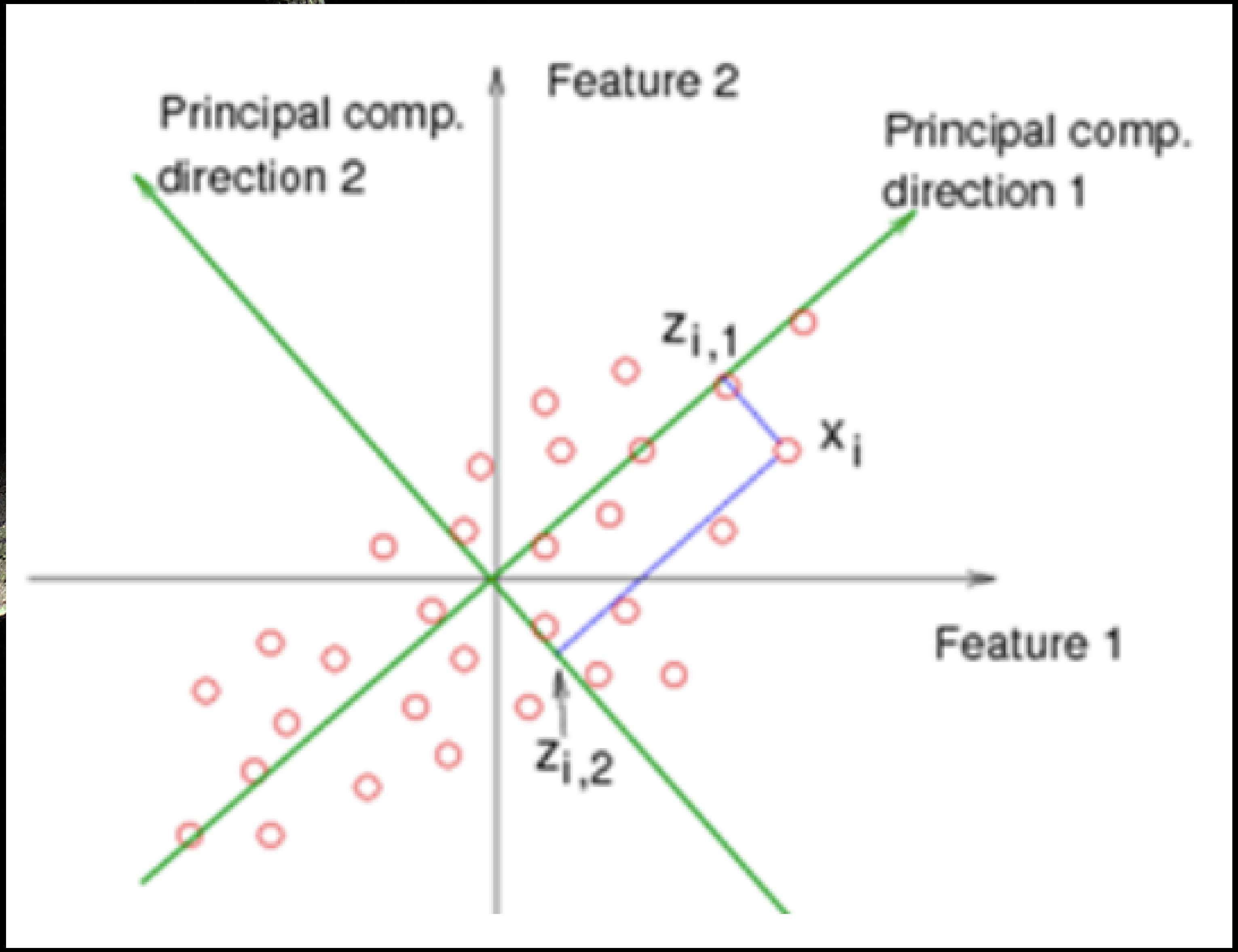
Data are normalized between 0 and 1 through the MinMaxScaler function of sklearn preprocessing package.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Data Preprocessing

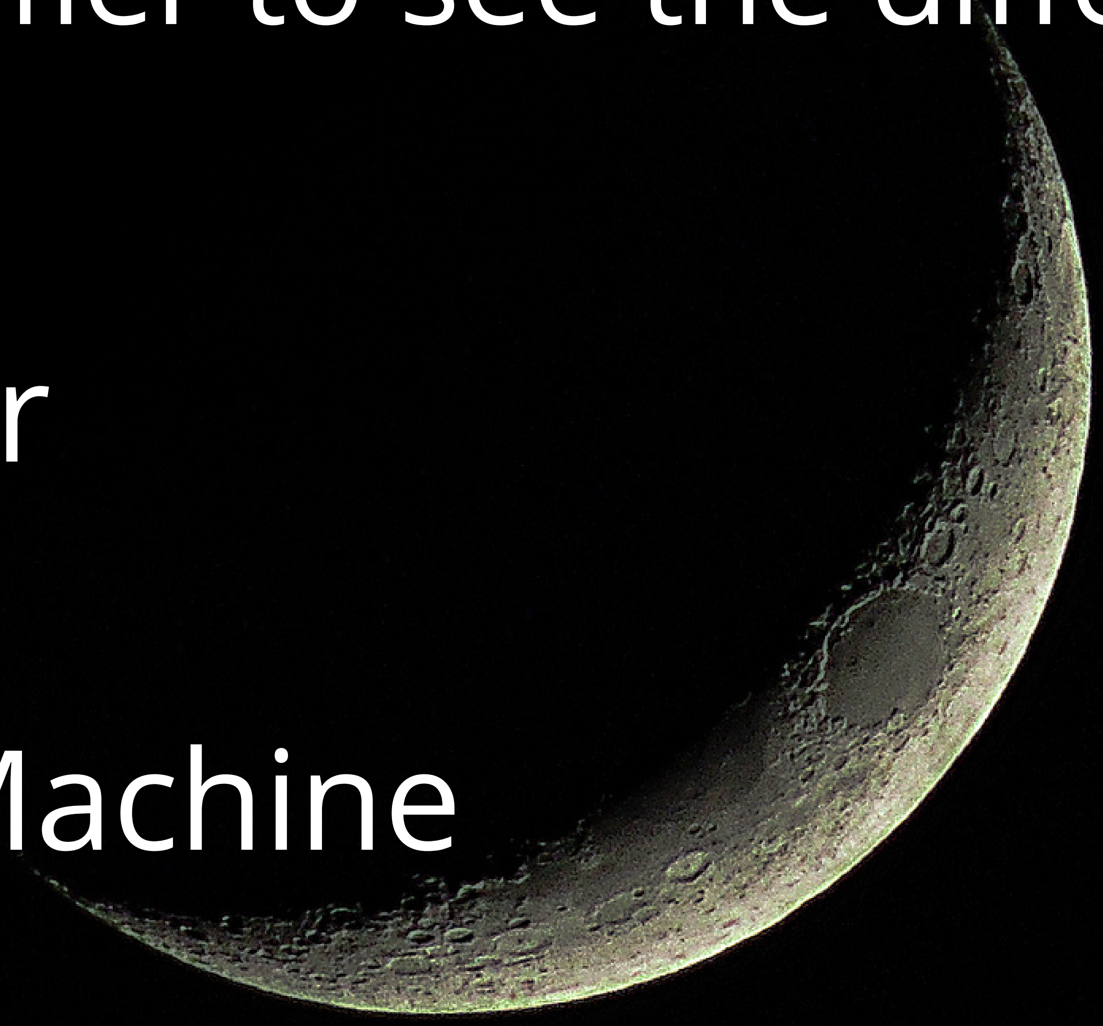
PRINCIPAL COMPONENT ANALYSIS

PCA is a feature reduction technique that try to find some metafeatures in which samples has a large variance, so the classification may have better performance



Model Definition

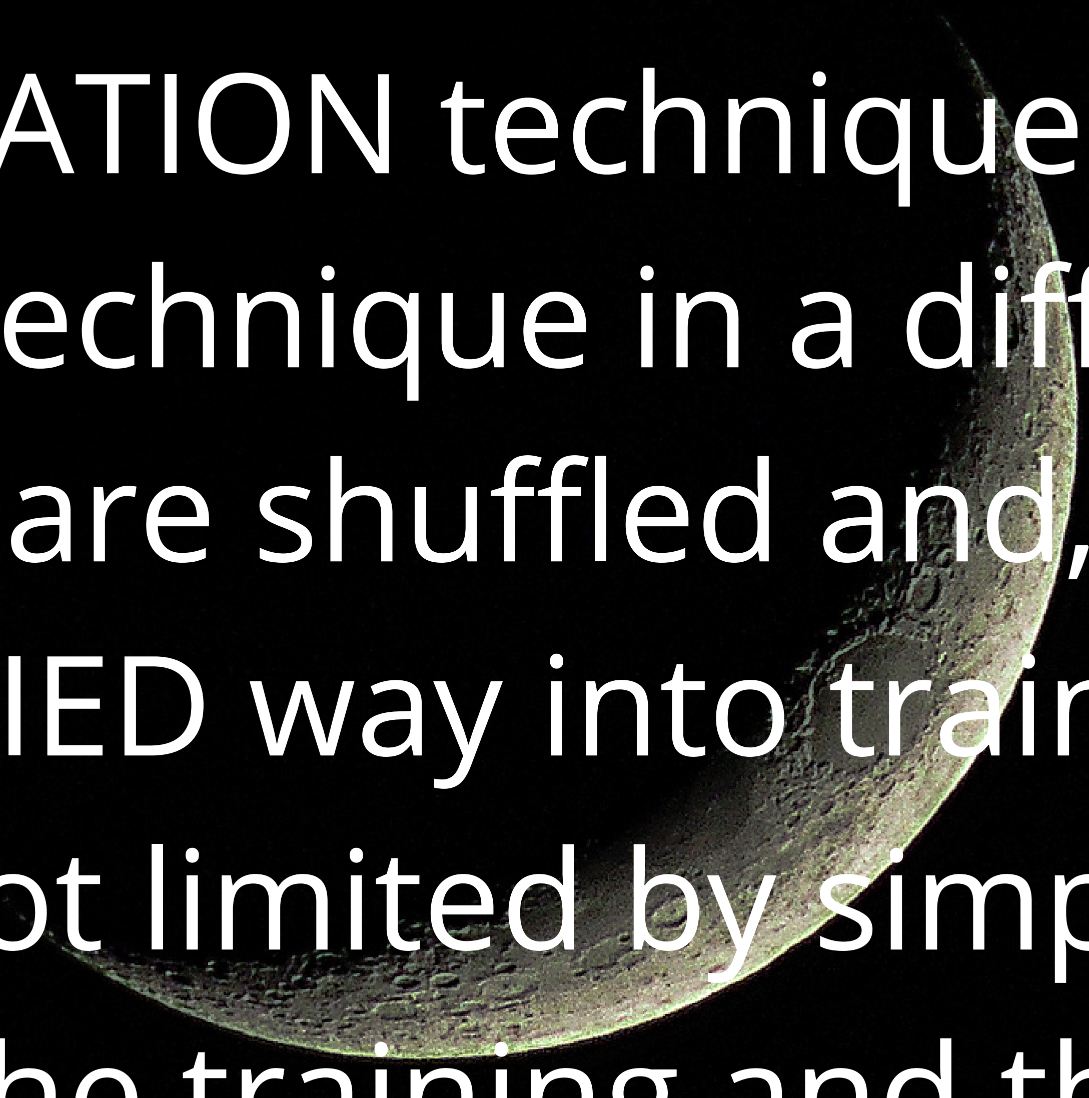
I choose different classifier to see the different performances



- Nearest Mean
- Nearest Neighbor
- Naive Bayes
- Support Vector Machine
- Decision Tree
- Random Forest
- Neural Network

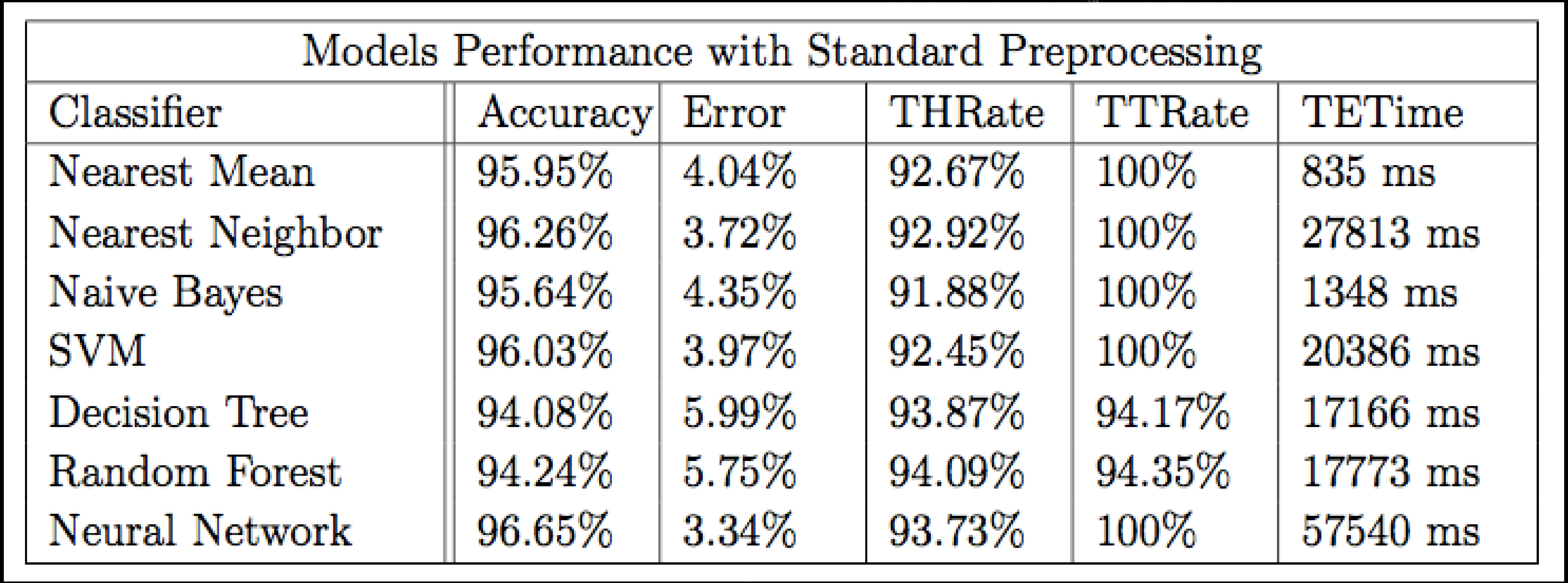
Workflow

In order to see the performance on different training set the CROSSVALIDATION technique is performed. I decided to use this technique in a different way.
In detail all the data are shuffled and, after that, are splitted in a STRATIFIED way into training and test set.
In this way we are not limited by simply divide data in different folds, but the training and the test set are determined by randomly shuffle the overall dataset and split it into training and test set.



Results

MinMax Preprocessing

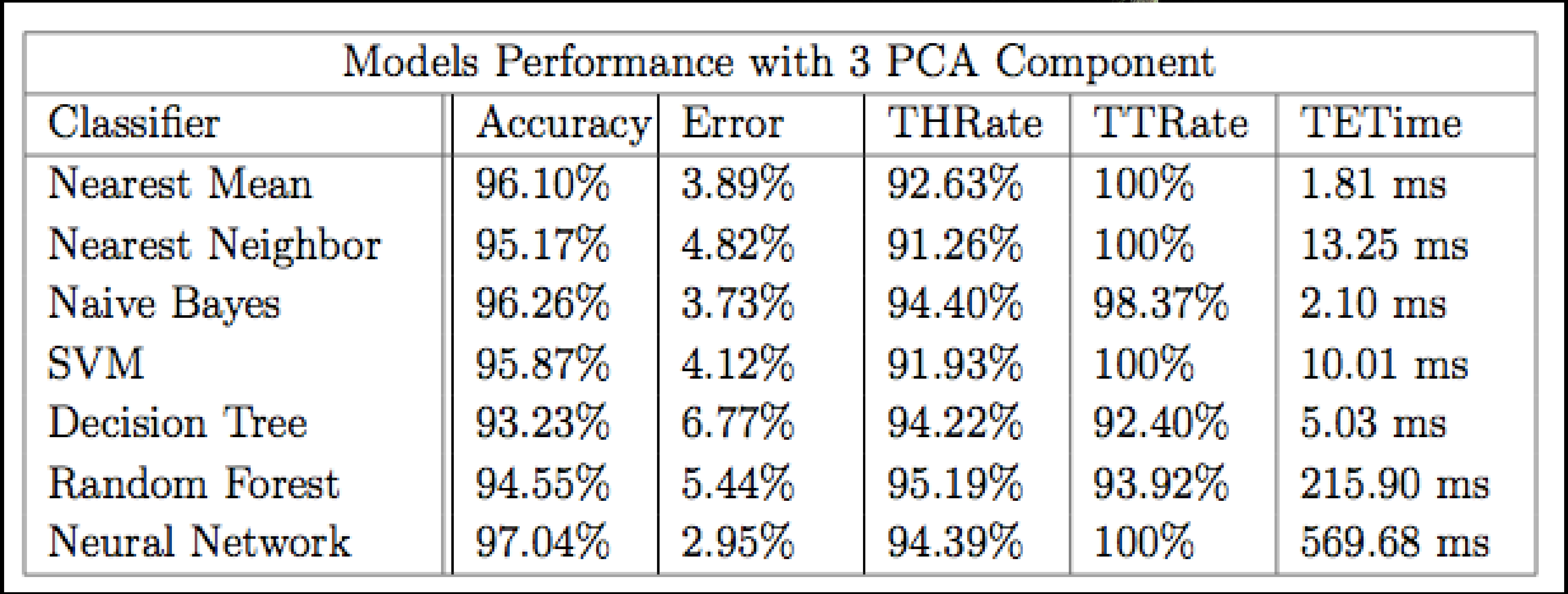


Models Performance with Standard Preprocessing					
Classifier	Accuracy	Error	THRate	TTRate	TETime
Nearest Mean	95.95%	4.04%	92.67%	100%	835 ms
Nearest Neighbor	96.26%	3.72%	92.92%	100%	27813 ms
Naive Bayes	95.64%	4.35%	91.88%	100%	1348 ms
SVM	96.03%	3.97%	92.45%	100%	20386 ms
Decision Tree	94.08%	5.99%	93.87%	94.17%	17166 ms
Random Forest	94.24%	5.75%	94.09%	94.35%	17773 ms
Neural Network	96.65%	3.34%	93.73%	100%	57540 ms

Results

PCA Preprocessing

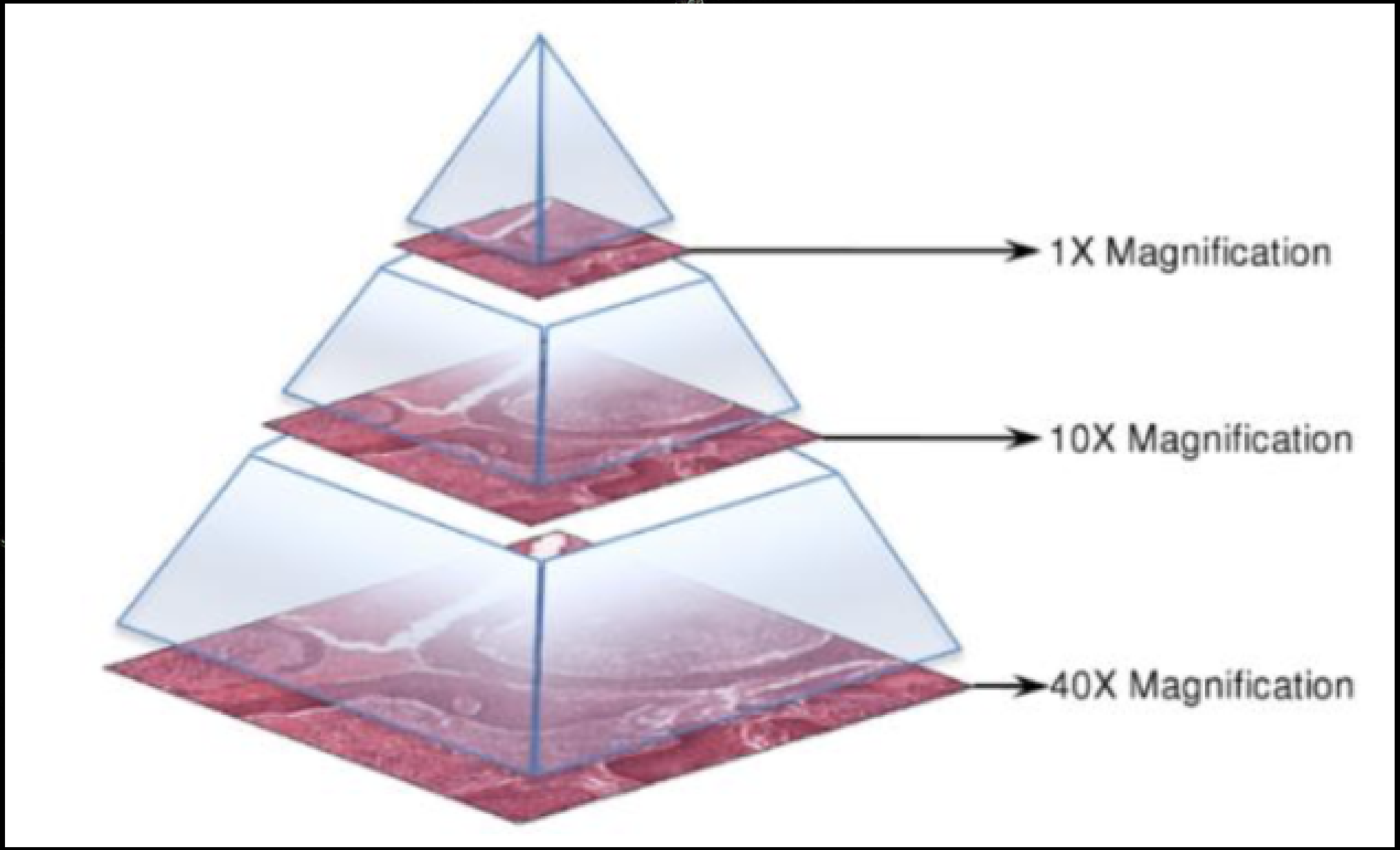
The best results are achived with 3 PCA



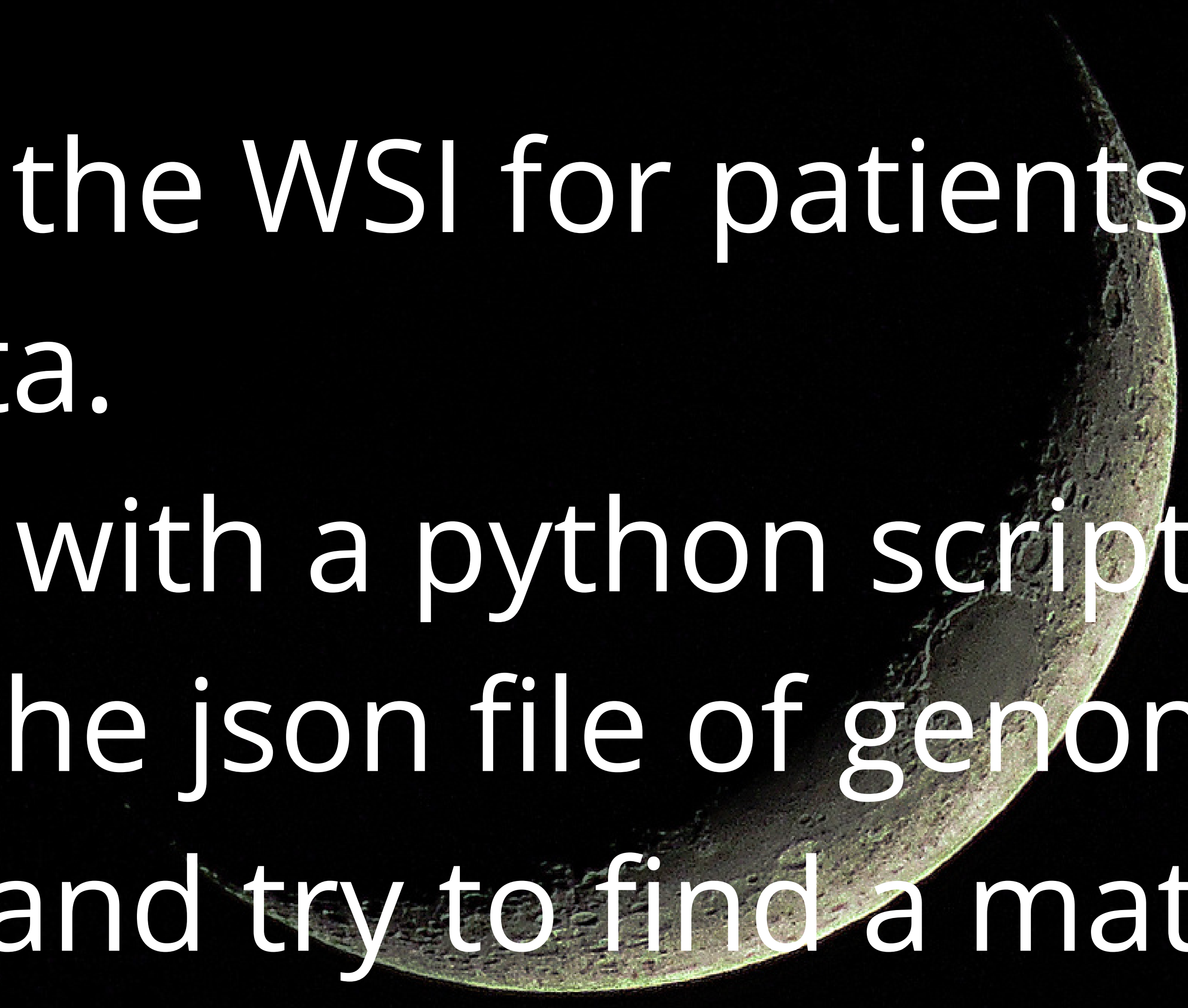
Models Performance with 3 PCA Component					
Classifier	Accuracy	Error	THRate	TTRate	TETime
Nearest Mean	96.10%	3.89%	92.63%	100%	1.81 ms
Nearest Neighbor	95.17%	4.82%	91.26%	100%	13.25 ms
Naive Bayes	96.26%	3.73%	94.40%	98.37%	2.10 ms
SVM	95.87%	4.12%	91.93%	100%	10.01 ms
Decision Tree	93.23%	6.77%	94.22%	92.40%	5.03 ms
Random Forest	94.55%	5.44%	95.19%	93.92%	215.90 ms
Neural Network	97.04%	2.95%	94.39%	100%	569.68 ms

Whole Slide Images

A WSI is a multiresolutional image made by digitizing microscope slides at diagnostic resolution.



Data Preparation



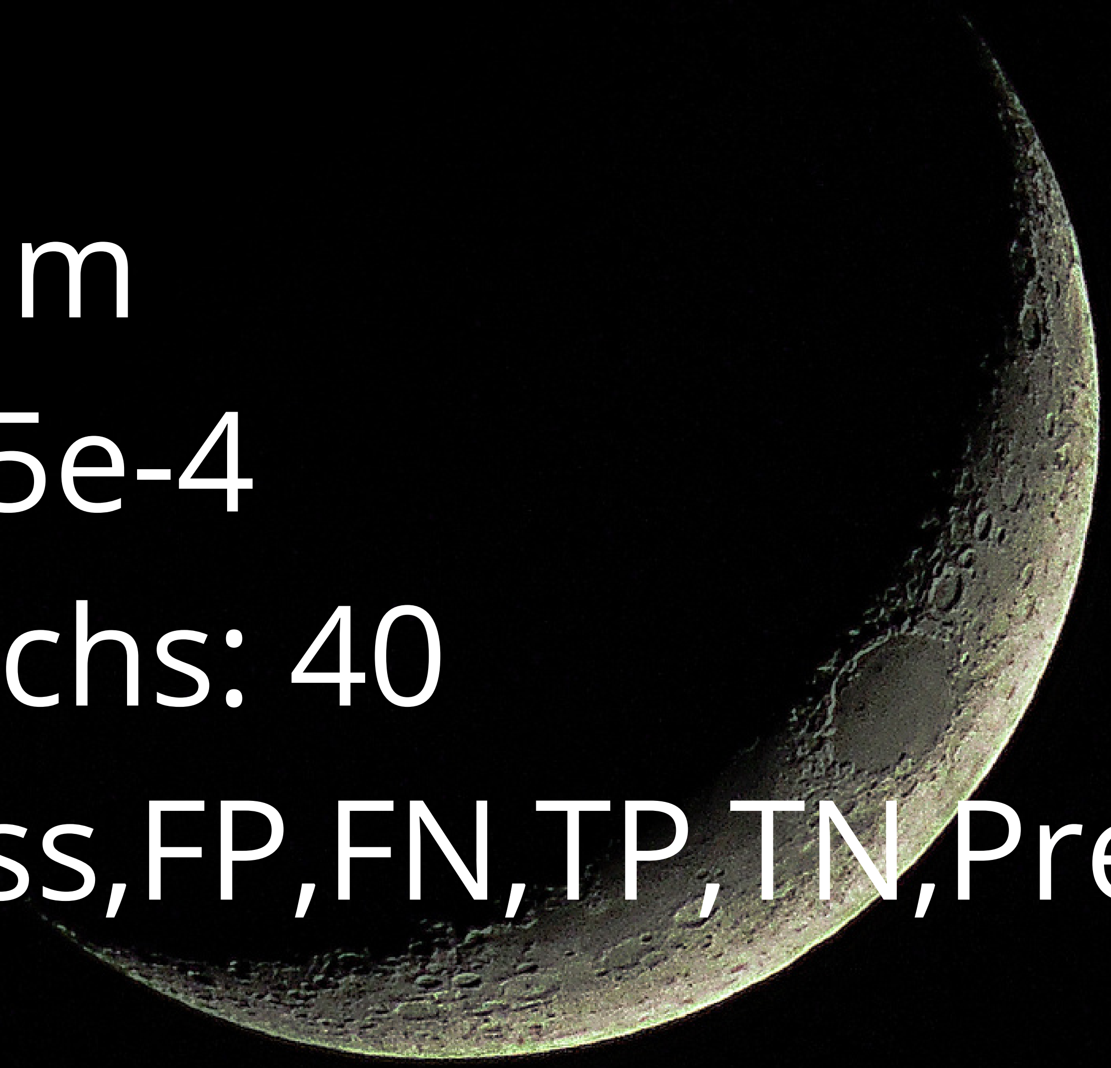
I decide to select the WSI for patients for which i had the genomics data.
This step is done with a python script that search these elements inside the json file of genomics data,
save the case_id and try to find a match in the json file of the WSI data, and delete the others images inside the manifest file of the WSI data.

Data Preprocessing

- Extract relevant patches from Whole Slide Images
- Balance the dataset
- Training, Validation and Test split
- Labels Encoding

A preprocessing layer is also added as first layer of the model in orther to augment and rescale images runtime .

Workflow Parameter



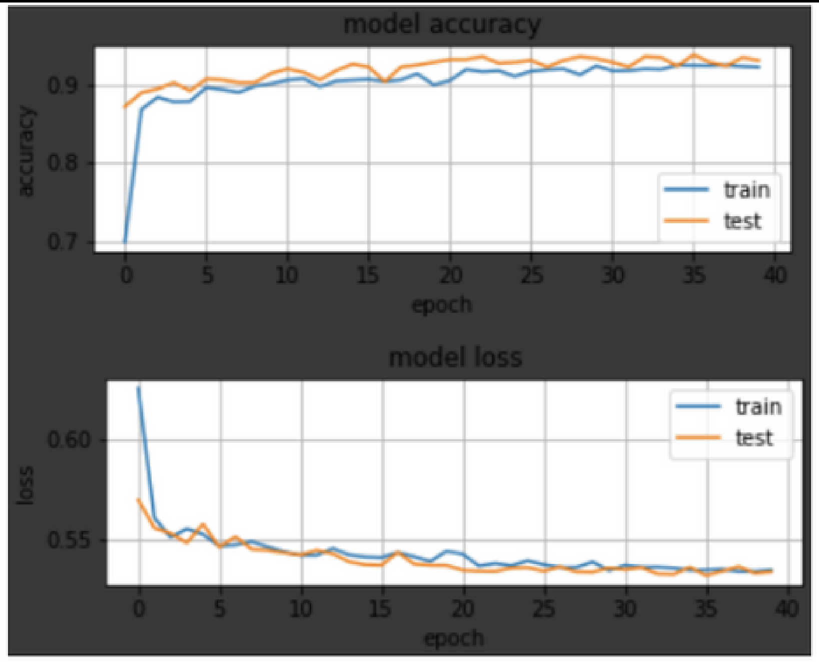
- Optimizer : Adam
- Learning Rate: 5e-4
- Number of epochs: 40
- Metrics: Acc,Loss,FP,FN,TP,TN,Precision, Recall
- Early Stopping
- Reduce Learning Rate on Plateau

Benchmark Models

- Xception:
  - This model exploits the Deepwise and Pointwise Convolution.
- ResNet152
  - This model has introduce the concept of residual module between convolutional layers.

Results of TL and FT

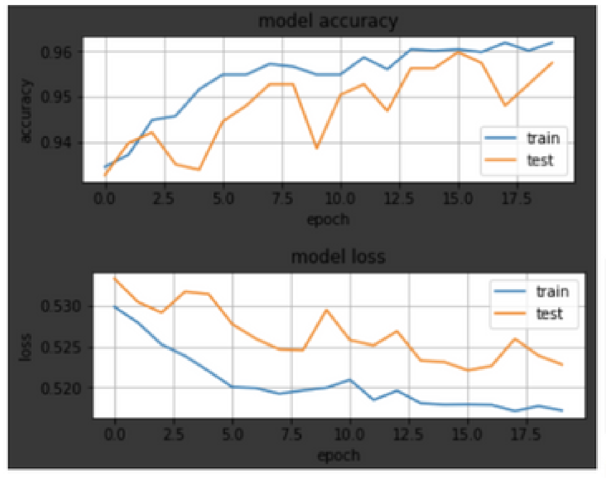
XCEPTION



```
Loss:             0.5965609550476074
Accuracy:         0.7510638236999512
False Positive:   0.0
False Negative:   117.0
True Positive:    118.0
True Negative:    235.0
Precision     :   1.0
Recall:           0.5021276473999023
```

Figure 1.10: XCEPTION TRANSFER LEARNING TRAINING PARAMETER

Figure 1.11: XCEPTION TRANSFER LEARNING RESULTS ON TEST SET

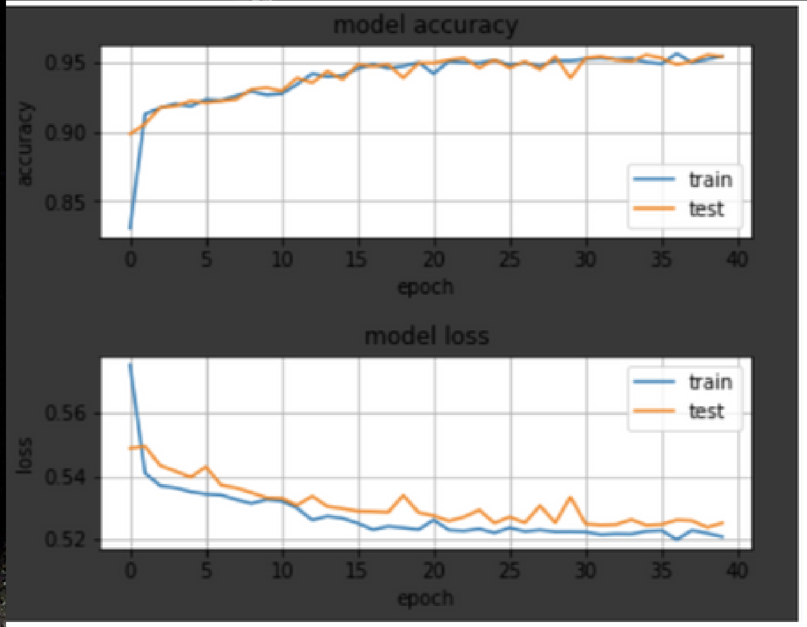


```
Loss:             0.5767318606376648
Accuracy:         0.8042553067207336
False Positive:   0.0
False Negative:   92.0
True Positive:    143.0
True Negative:    235.0
Precision     :   1.0
Recall:           0.6085106134414673
```

Figure 1.12: XCEPTION FINE TUNING TRAINING PARAMETER

Figure 1.13: XCEPTION FINE TUNING RESULTS ON TEST SET

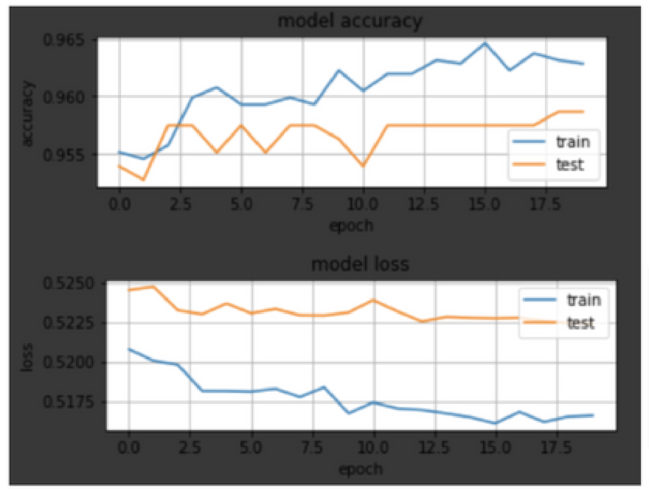
RESNET-152



```
Loss:             0.5825520157814026
Accuracy:         0.7914893627166748
False Positive:   1.0
False Negative:   97.0
True Positive:    138.0
True Negative:    234.0
Precision     :   0.9928057789802551
Recall:           0.5872340202331543
```

igure 1.14: RESNET TRANSFER LEARNING TRAINING PARAMETER

Figure 1.15: RESNET TRANSFER LEARNING RESULTS ON TEST SET

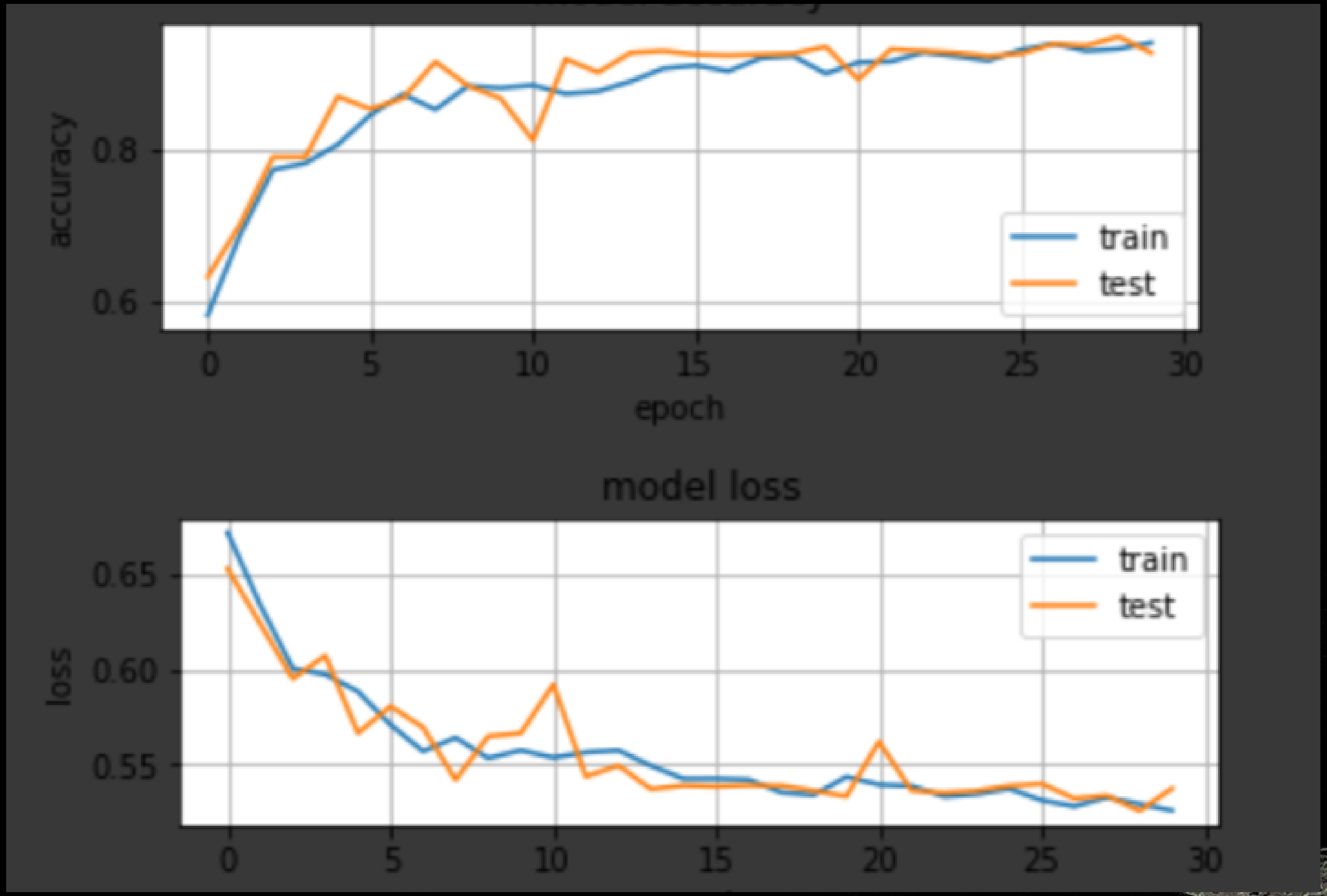


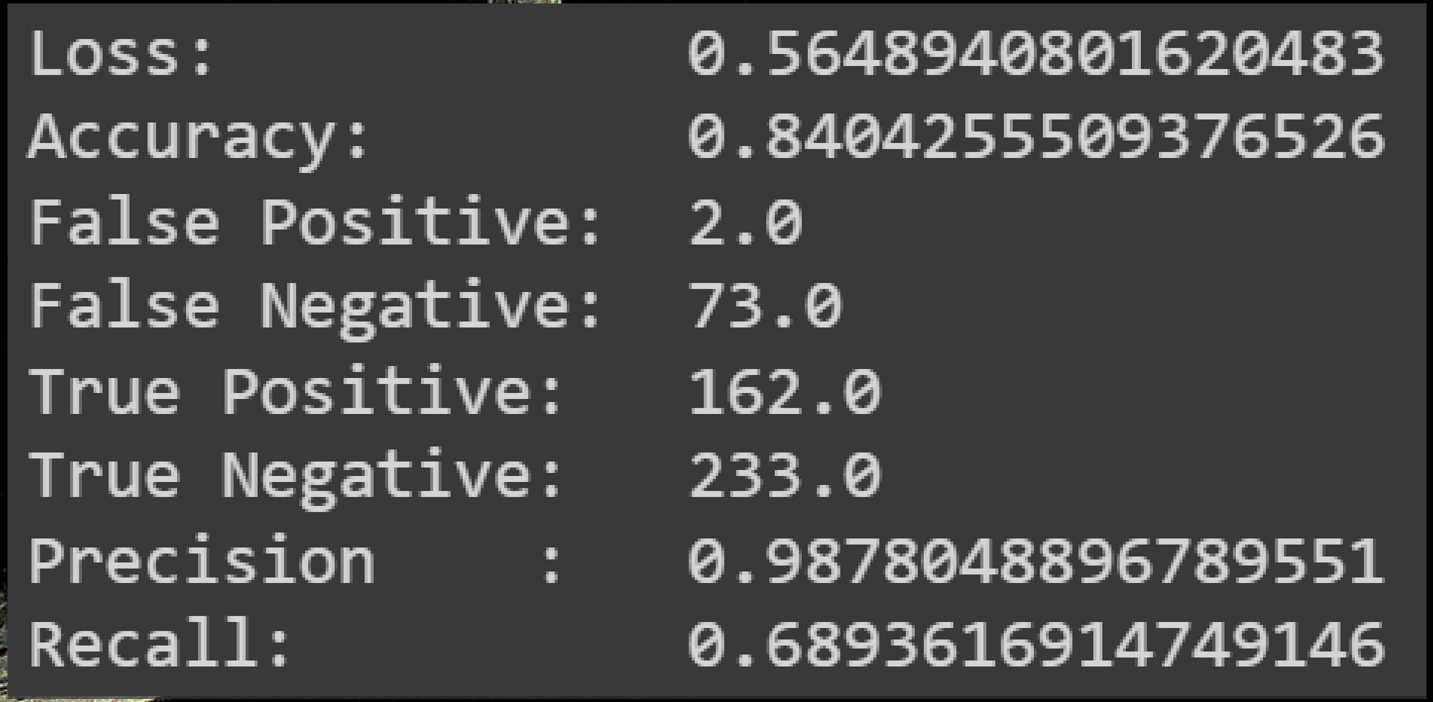
```
Loss:             0.5810348391532898
Accuracy:         0.7957446575164795
False Positive:   0.0
False Negative:   96.0
True Positive:    139.0
True Negative:    235.0
Precision     :   1.0
Recall:           0.5914893746376038
```

Figure 1.16: RESNET TRANSFER LEARNING TRAINING PARAMETER

Figure 1.17: RESNET TRANSFER LEARNING RESULTS ON TEST SET

Results Training from Scratch





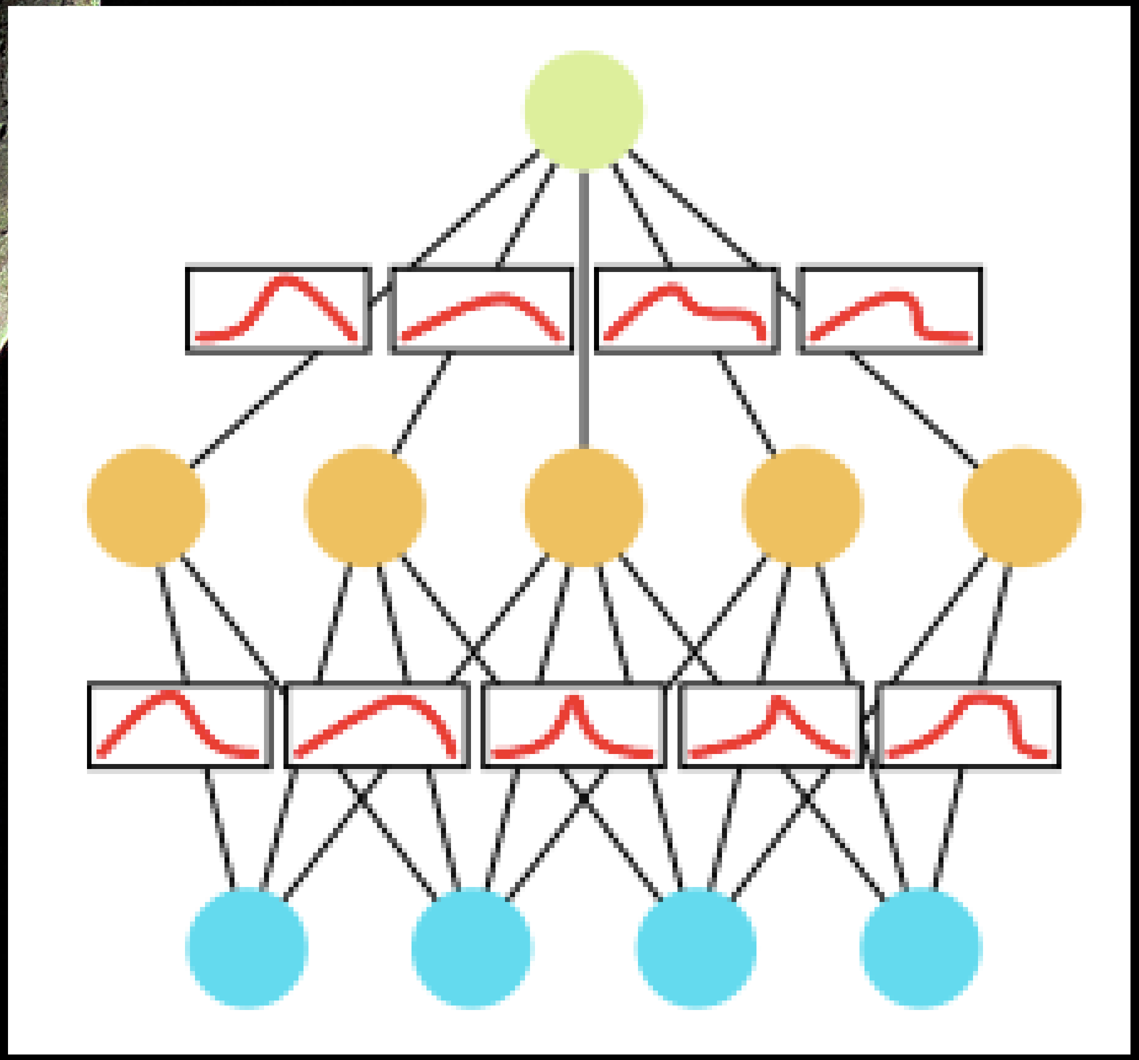
```
Loss:               0.5648940801620483
Accuracy:           0.8404255509376526
False Positive:     2.0
False Negative:     73.0
True Positive:      162.0
True Negative:      233.0
Precision      :    0.9878048896789551
Recall:             0.6893616914749146
```

This proofs that for different image classification problems, the weights of the model, may be different in order to shows the best performance for that particular classification problem.

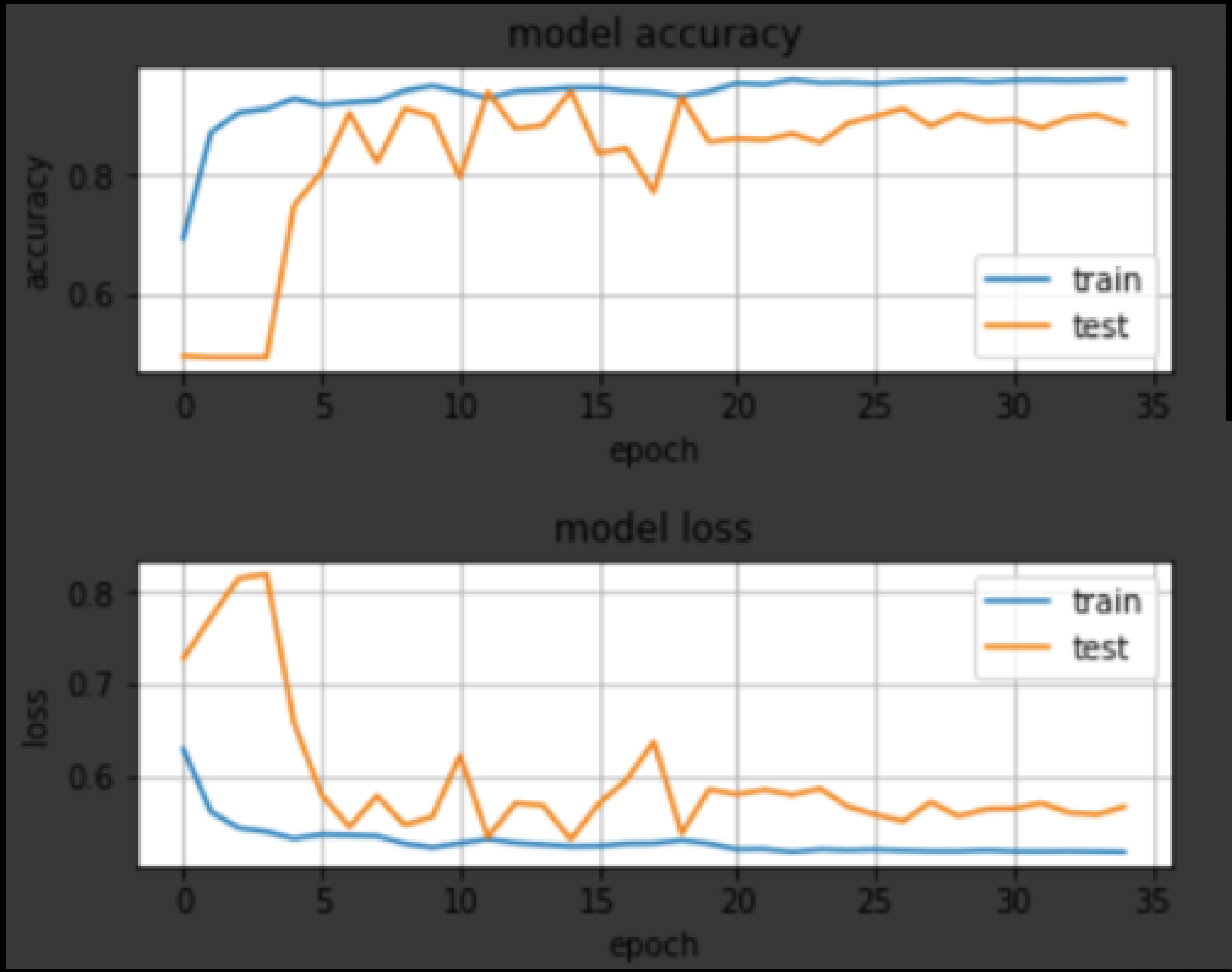
My Model

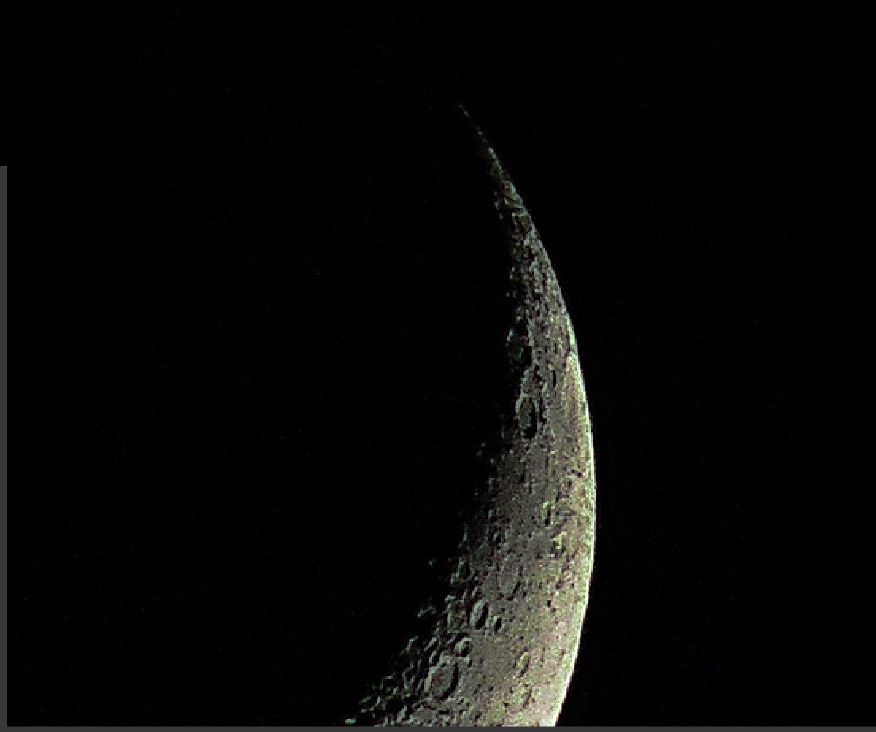
My model exploit the Deepwise and Pointwise convolution.And and also the residual module. As in the
 Xception and ResNet Networks.

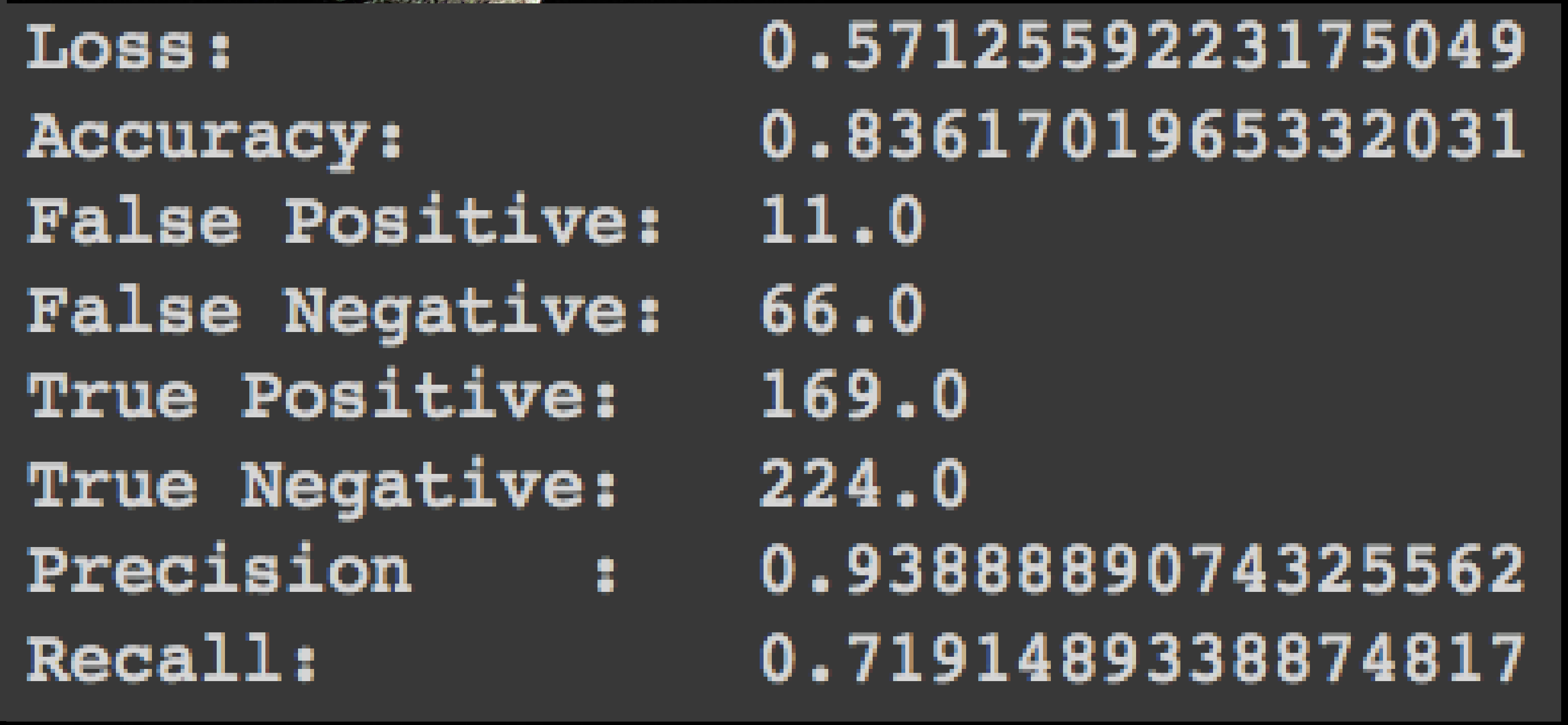
I also tried to implement the Network as a Bayesian Neural Network.
It exploits the mean and the uncertainty of the predicted label.
This is done with several Dropout layers that works in a Variational Dropout way..



Results

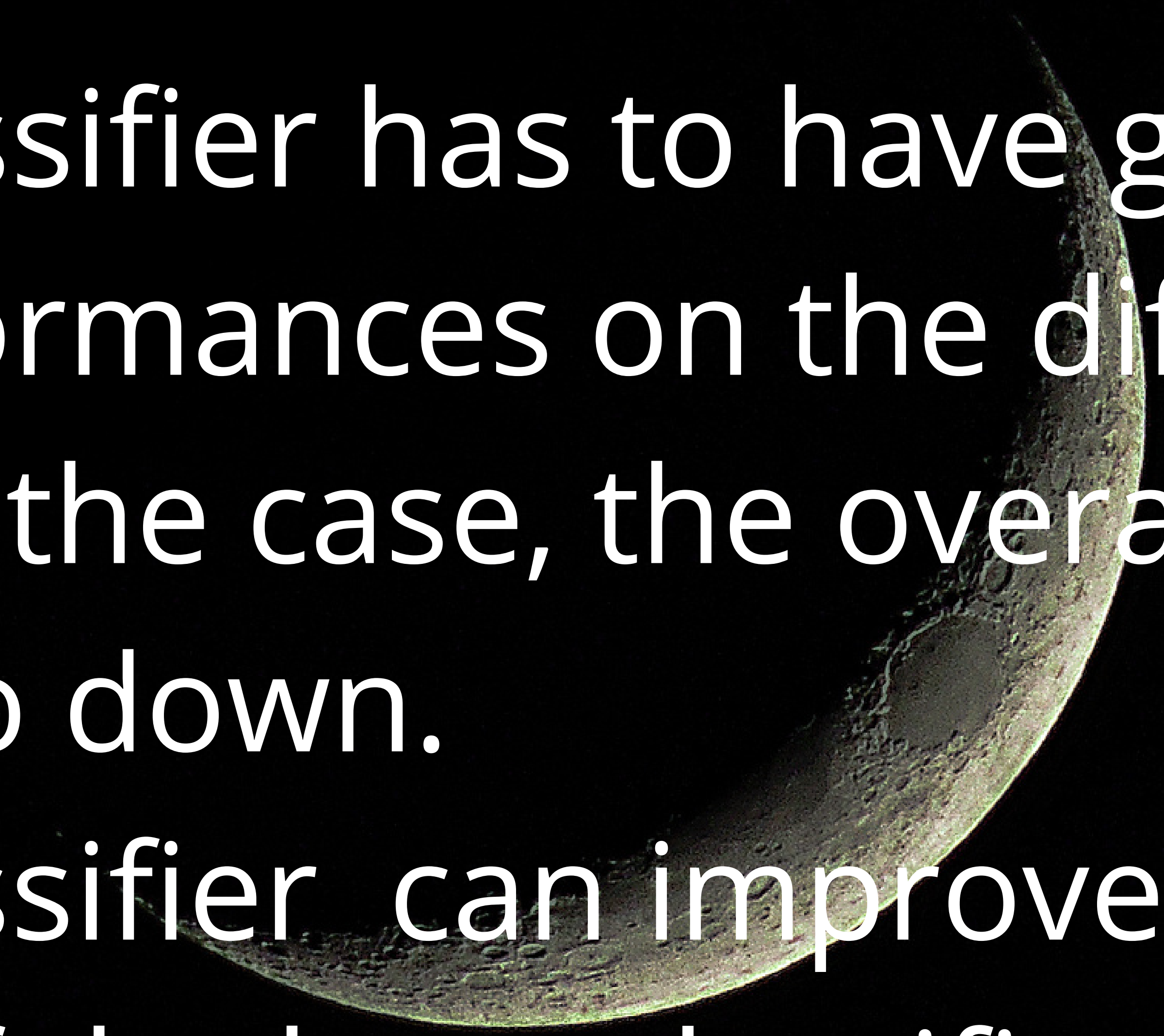






```
Loss:              0.5712559223175049
Accuracy:          0.8361701965332031
False Positive:    11.0
False Negative:    66.0
True Positive:     169.0
True Negative:     224.0
Precision     :    0.9388889074325562
Recall:            0.7191489338874817
```

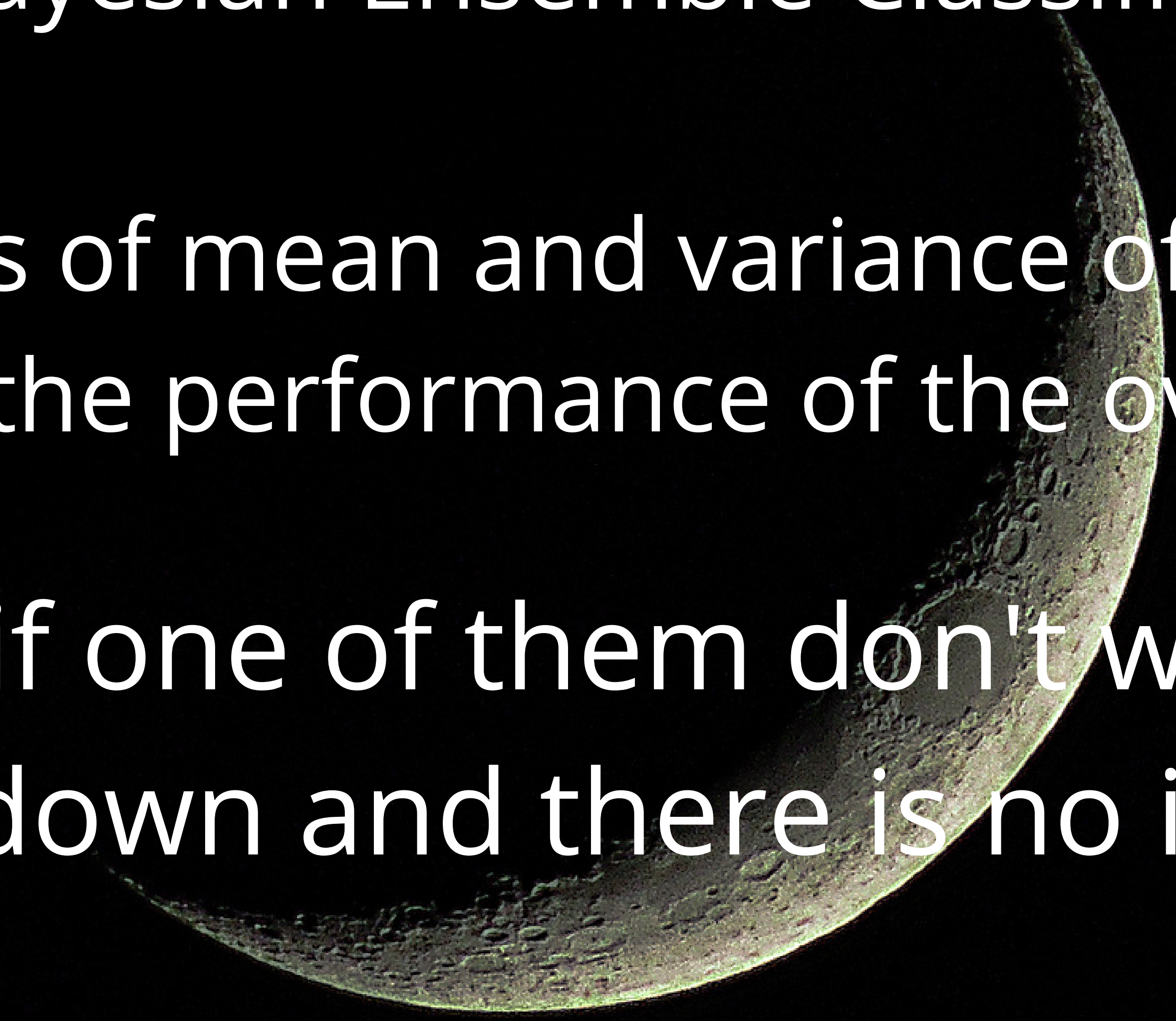
CONSENSUS CLASSIFIER



Main hypothesis for this workflow:

- A consensus classifier has to have good classifiers in terms of performances on the different data domain. If is not the case, the overall performances go down.
- A consensus classifier can improve the performances of the base classifiers introducing some heuristic on the results coming from different domains.

CONSENSUS CLASSIFIER



- The idea is to build a Bayesian Ensemble Classifier that works with heterogenous data.
- The results are in terms of mean and variance of the classification.
- The goal is to improve the performance of the overall classification.

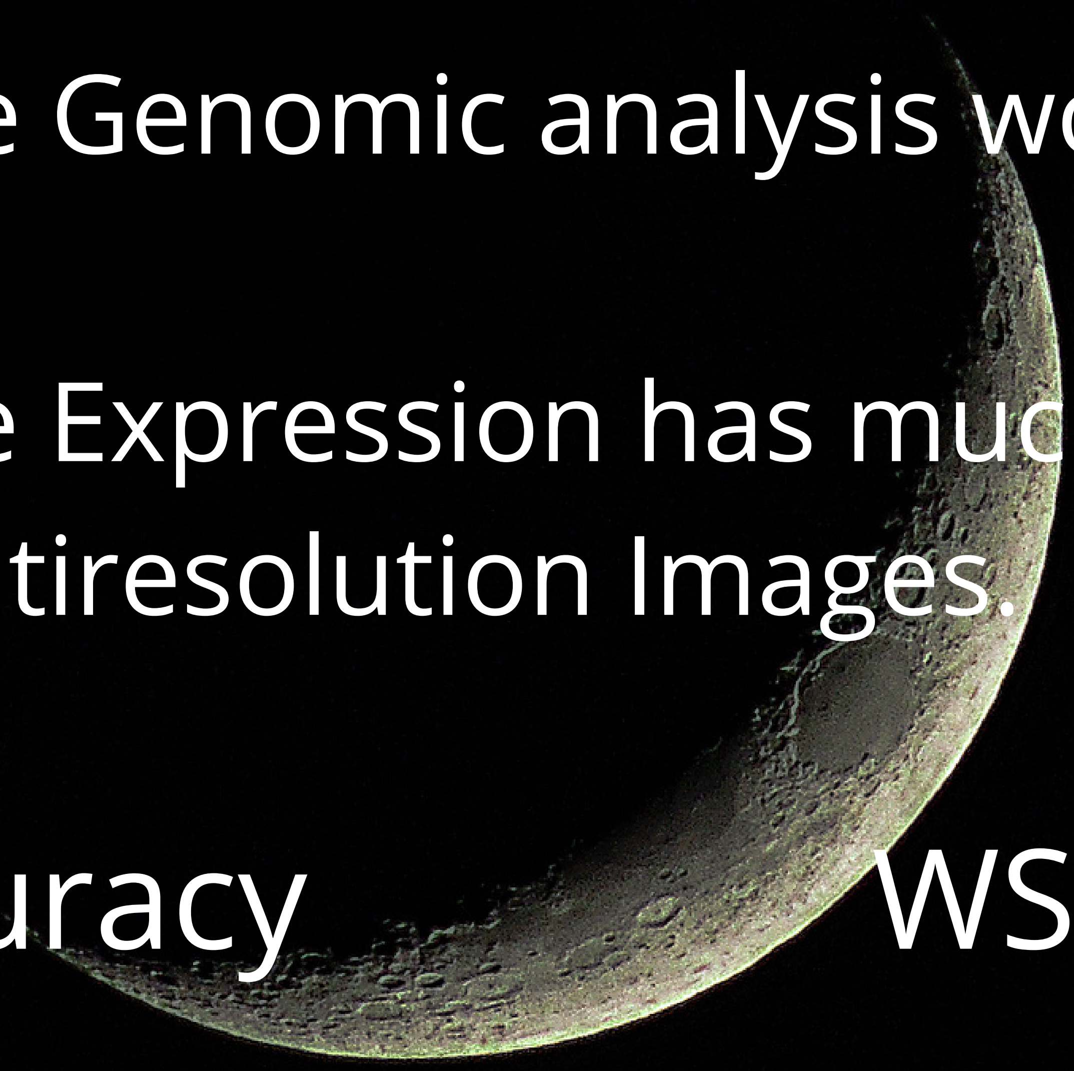
The problem is that if one of them don't well, the overall performances slow down and there is no improvement

NOTE: This problem, can leads to find out an answer in terms of which are the data that contain the most relavant information for a correct classification.

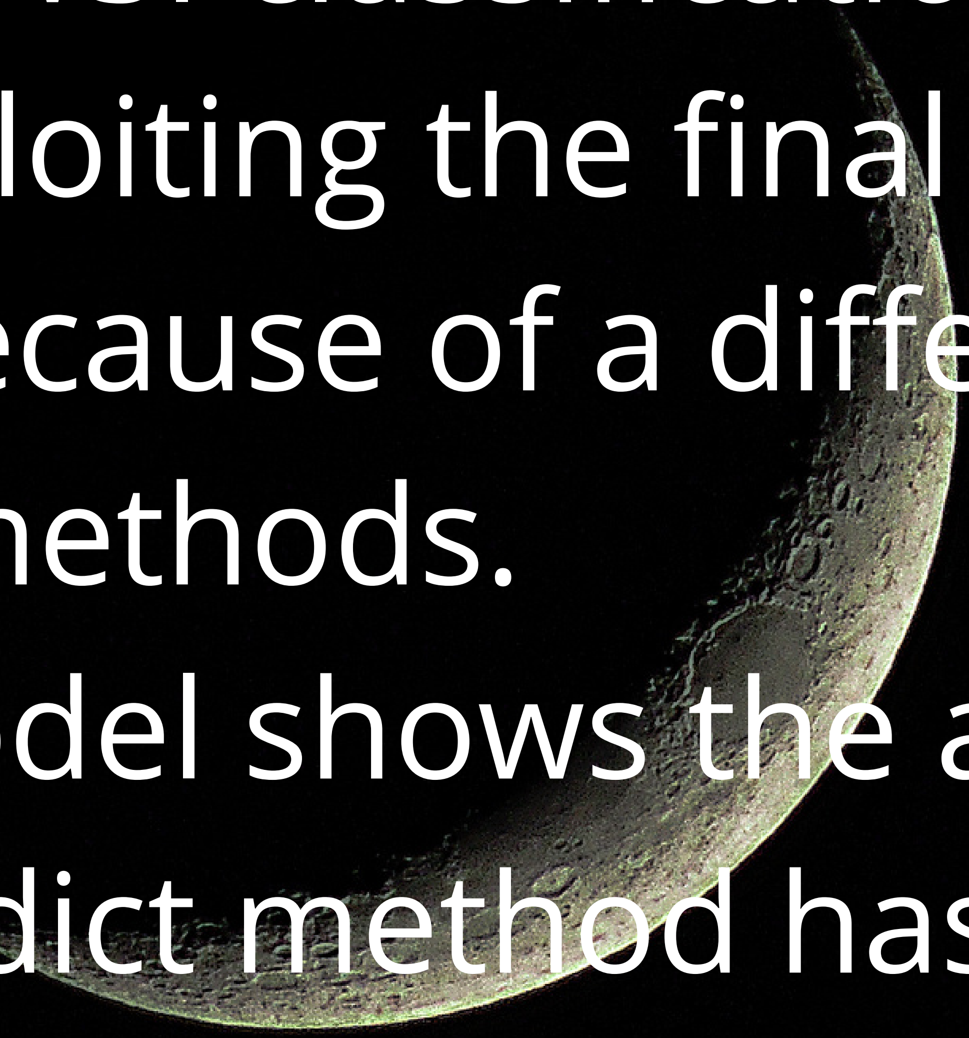
CONSENSUS CLASSIFIER

For this case of study, the Genomic analysis works better then the WSI analysis.
This proofs that the Gene Expression has much more relevant information then the Multiresolution Images.

Genomic Mean Accuracy	WSI Mean Accuracy
97.64%	83.19%

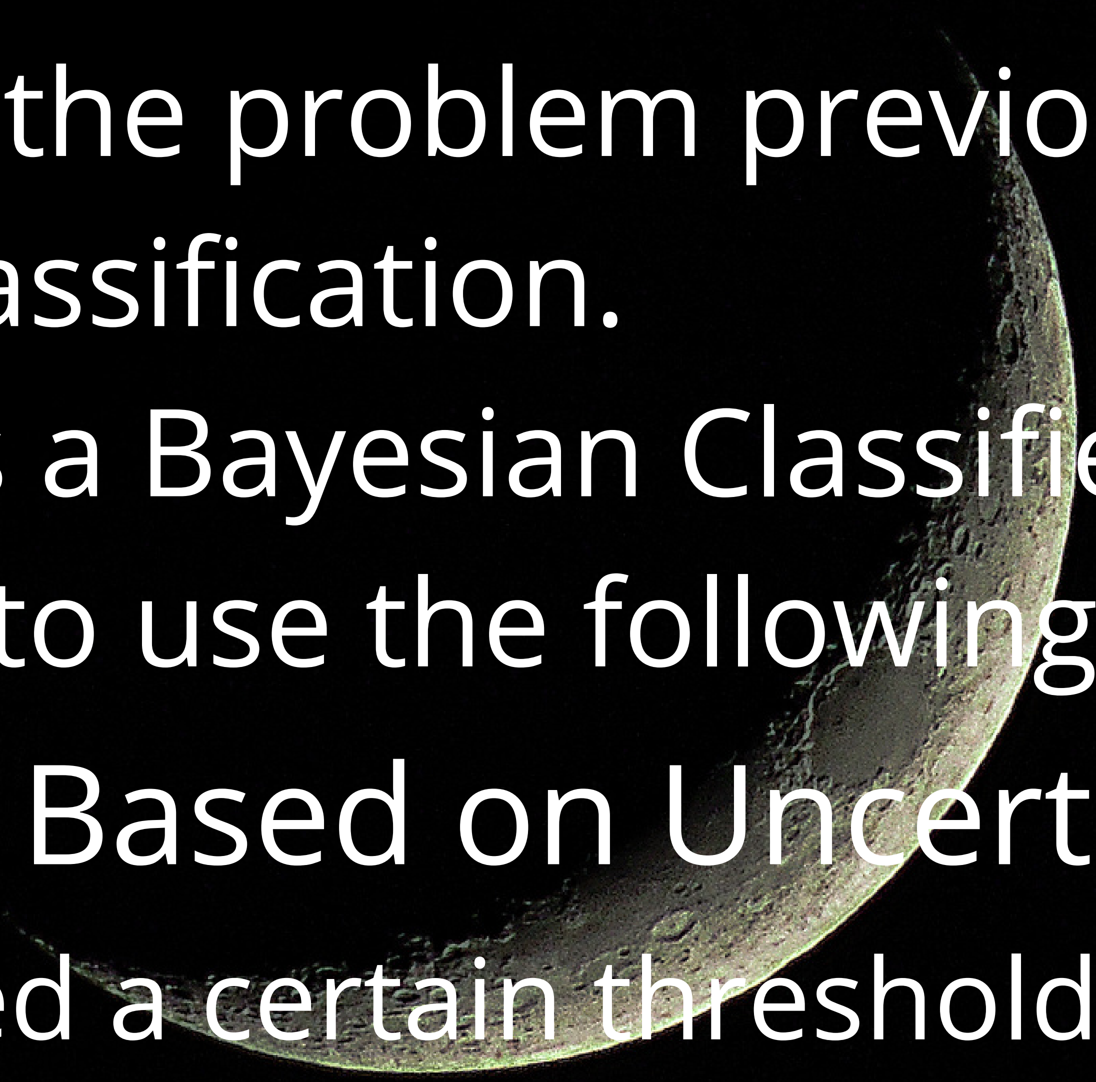


WSI Classification Problems



For what concern the WSI classification there were some problems in exploiting the final result of the classification, this is because of a difference between evaluate and predict methods.
When evaluate the model shows the accuracy wrote before, intead the predict method has an accuracy around 50%. I tried in different way to exploit the final output of the sigmoid activation and to round it, but without success.

IMPROVEMENTS

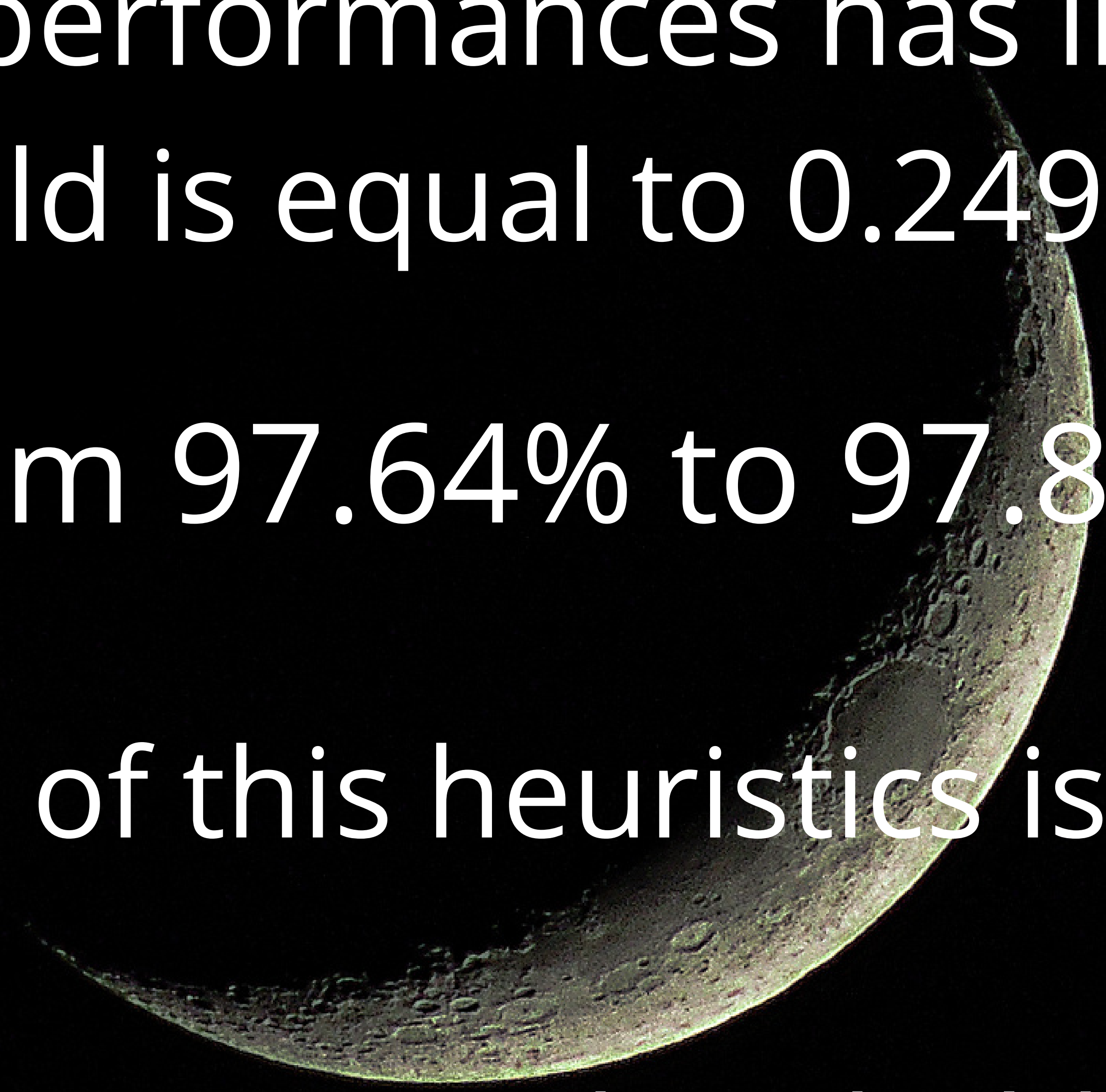


The idea was to use the different classifier for the different domains, but because of the problem previously explained let's consider the Genomic Classification.
The fact that my model is a Bayesian Classifier enabling to work in the following way and to use the following heuristics:

'Reverse Label Based on Uncertainty'

When the uncertainty exeed a certain threshold, means that the classification is uncertain.
So if the opposite label is guessed, the correct prediction may occur.
The new hyperparameter is the Uncertainty Threshold.

Results



With this approach the performances has improved a little bit:

- Uncertainty Threshold is equal to 0.249:

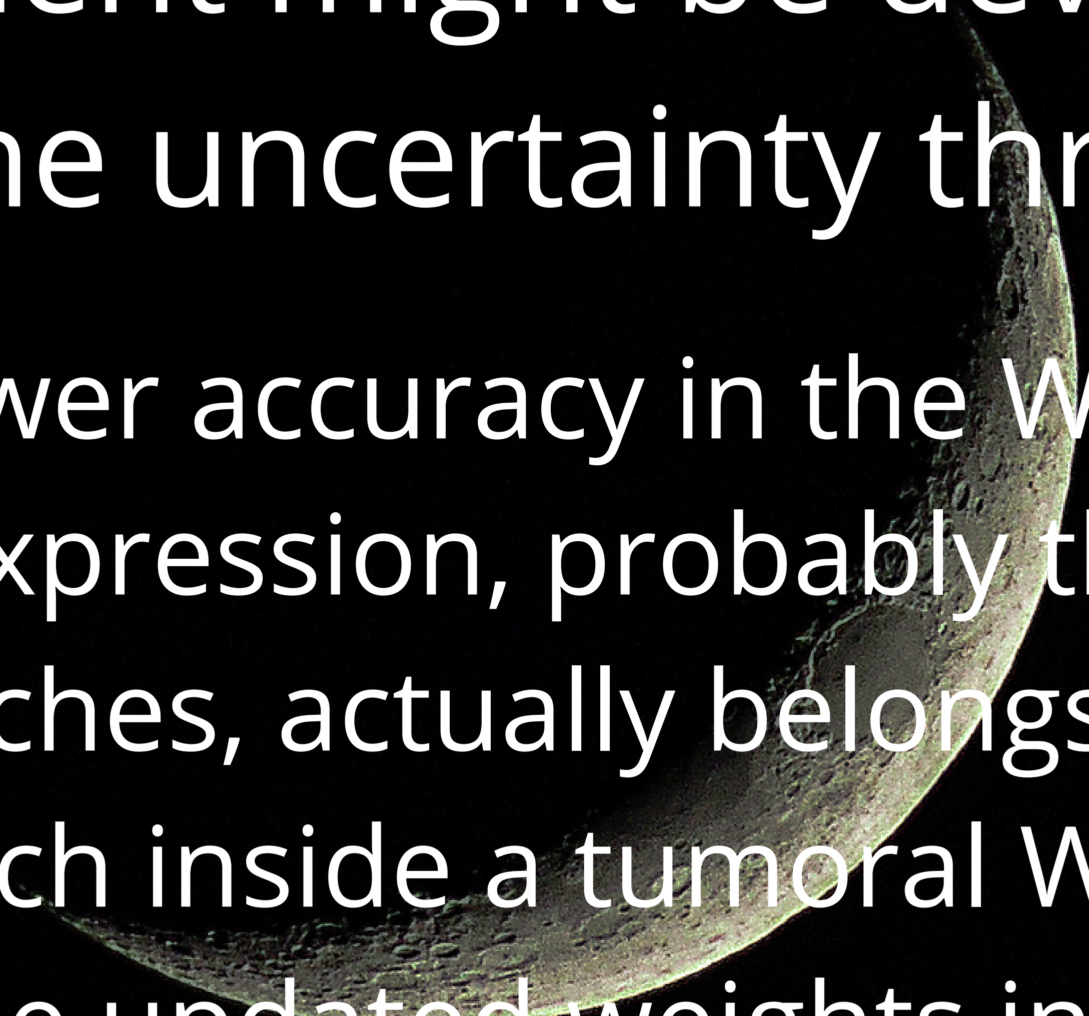
Performances: from 97.64% to 97.86% of accuracy

The main characteristics of this heuristics is the tuning of a parameter:

'Uncertainty Threshold'

That enable the reverse label based on uncertainty to improve the performances.

Final Improvements



A further improvement might be developing an algorithm to tune the uncertainty threshold value.

For what concern the lower accuracy in the WSI Classification compared to the gene expression, probably the problem is that some extracted patches, actually belongs to the opposite label, like an healthy patch inside a tumoral WSI or viceversa. This has an impact on the updated weights in the backpropagation phase during training, and the recognition of this patches may improve the performances.