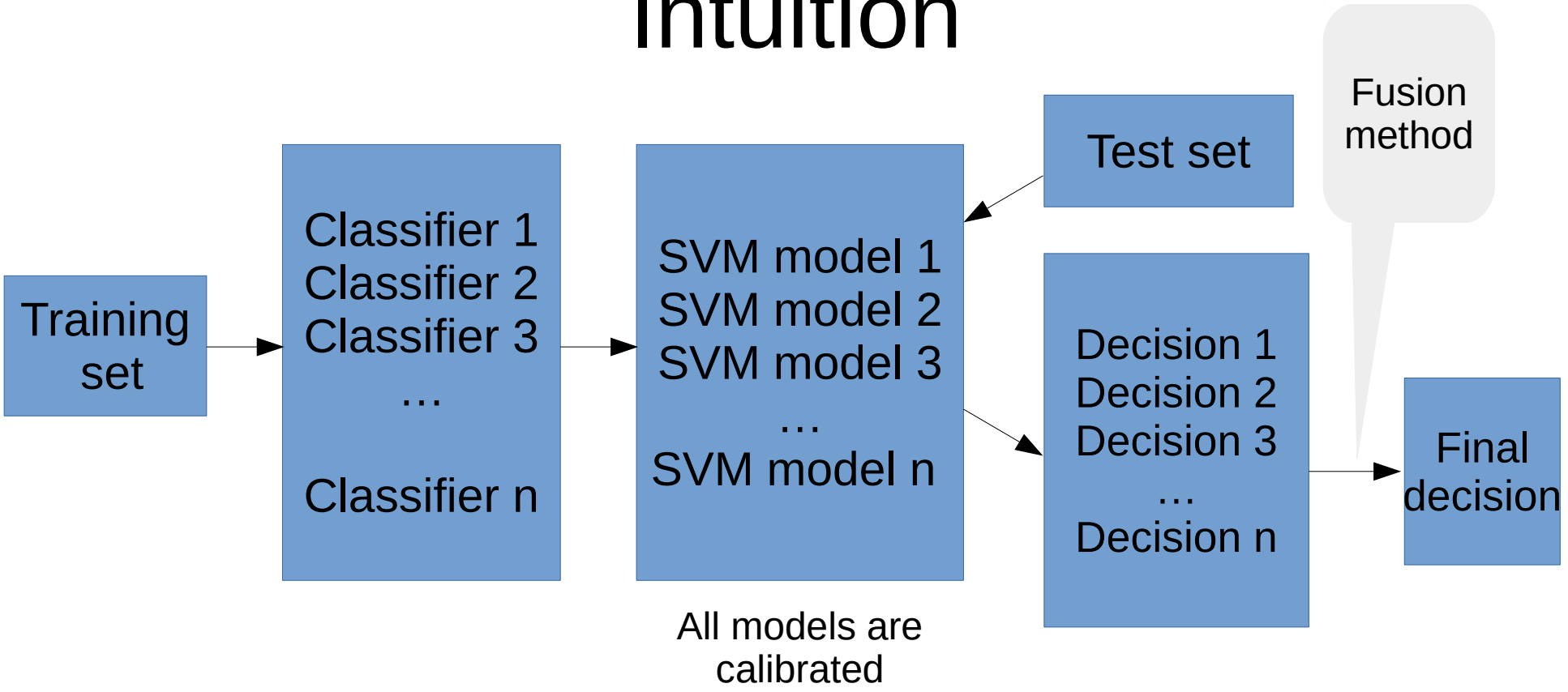
SVM ensembles approach intro

Nianheng Wu

Intuition



Nianheng Wu

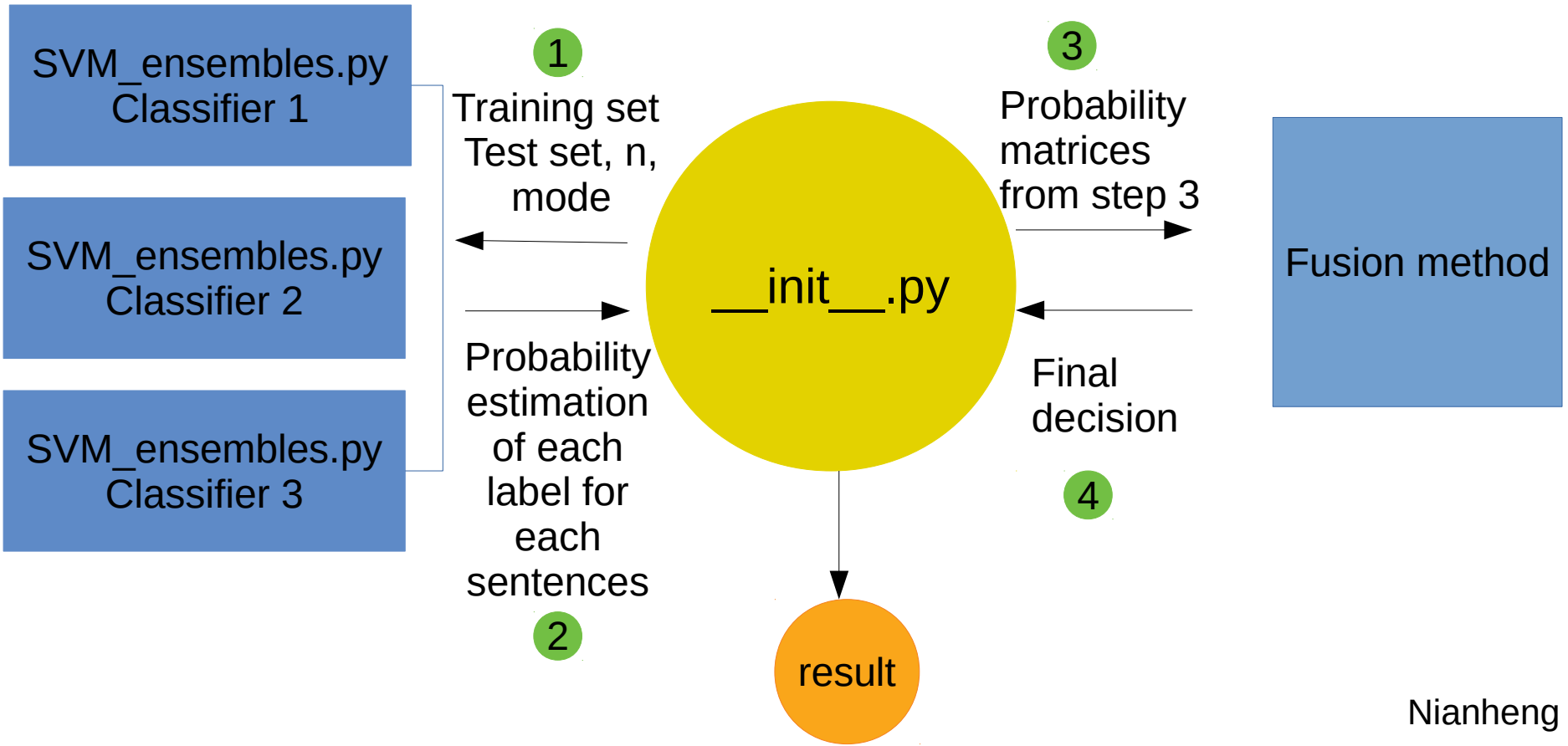
Our program structure

- __init__.py
- SVM_ensembles.py
- Fusion method.py
- Other testing functions.py

Our program structure

- __init__.py
  - An interface, also like a control center
  - Determine how many classifiers we wants to generate
  - Get the path to training set and test set, parse the paths to SVM_ensembles
  - Get the desion result back from SVM_ensembles. Then send the results to fusion method, and get the result from it.

Our program structure



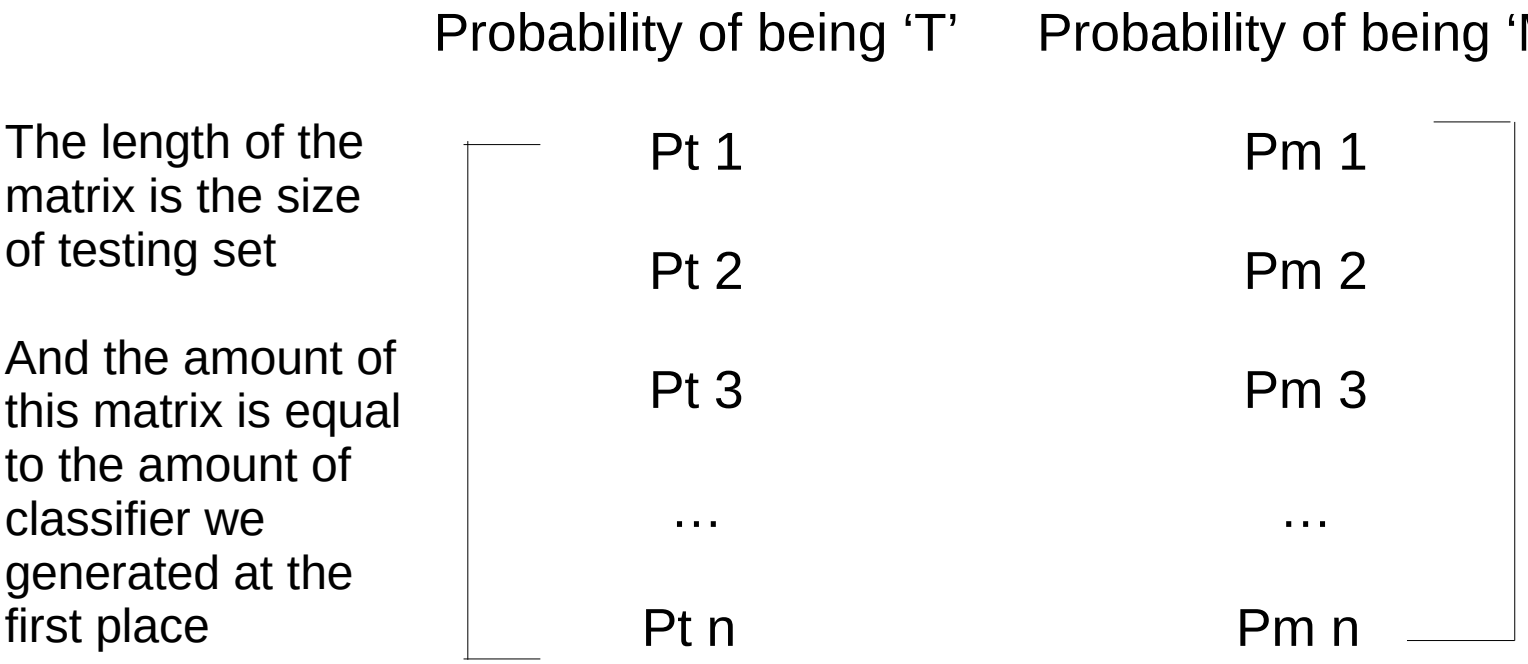
Nianheng Wu

Our program structure

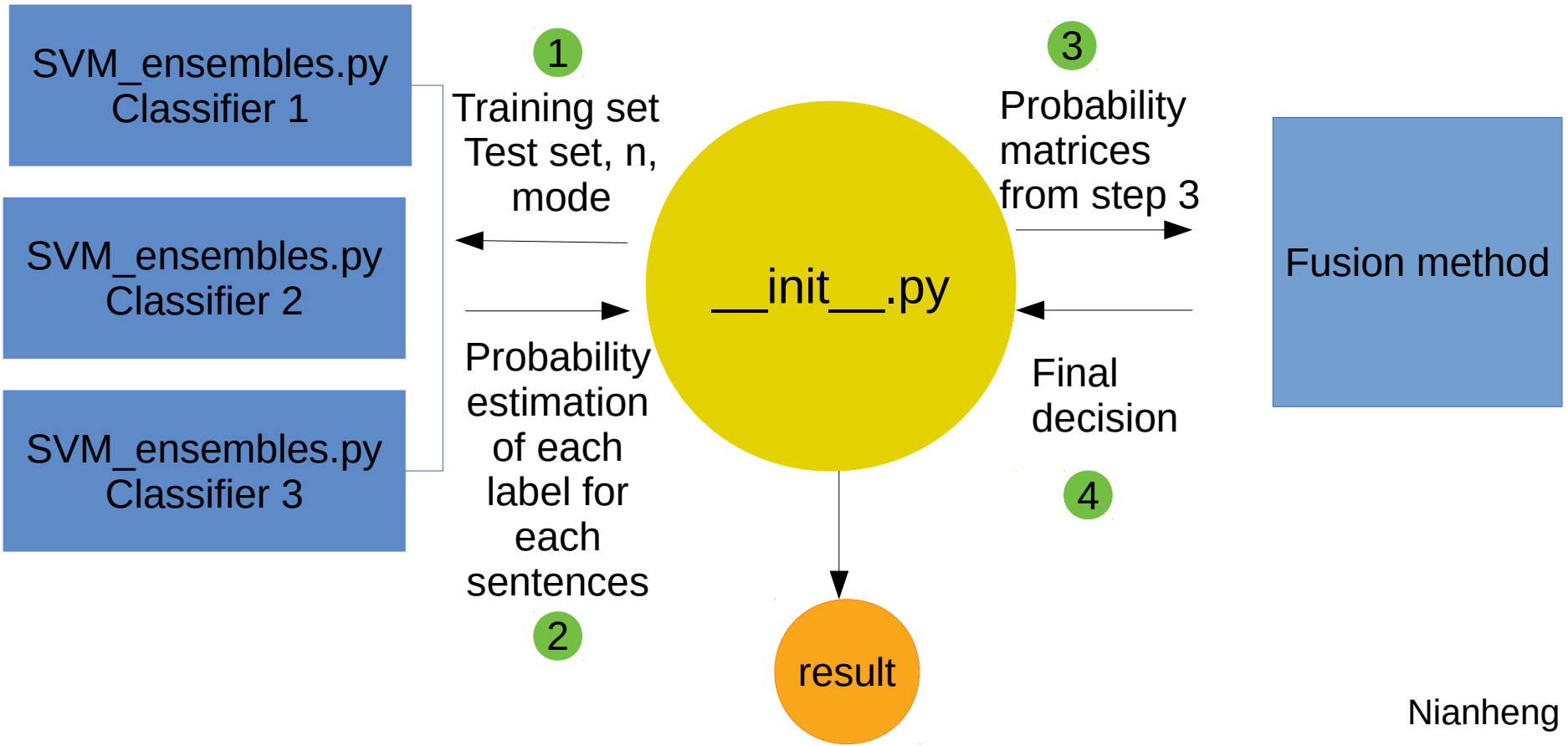
- SVM_ensembles.py
  - Contains a class, and two major functions: *training* and *testing*
  - Function *training* applys matrix and tf-idf and generate a matrix of features (n value and mode depend on parameters that parsed into it)
  - SVM model = LinearSVC(the matrix of features)
  - Because SVM model do not work based on probabilities, but in order to apply Mean Probability Rule as fusion method, we have to use a trick called *calibration*
  - So: clf = CalibrationClassifierCV(SVM model)
  - Now we have a classifier, we put the feature matrix of test set into the classifier and get a matrix of probabilities.

Our program structure

- SVM_ensembles.py



Our program structure



Nianheng Wu

Our program structure

- fusion_method.py
  - We have already gotten x of probability matrices, the __init__.py will parse them to the fusion method. Here I’ll just explain how Mean Probability Rule works.

Our program structure

- fusion_method.py

$$\begin{bmatrix} Pt1 & Pm1 \\ Pt2 & Pm2 \\ Pt3 & Pm3 \\ \cdots & \\ Ptn & Pmn \end{bmatrix} + \begin{bmatrix} Pt1' & Pm1' \\ Pt2' & Pm2' \\ Pt3' & Pm3' \\ \cdots & \\ Ptn' & Pmn' \end{bmatrix} + \begin{bmatrix} Pt1'' & Pm1'' \\ Pt2'' & Pm2'' \\ Pt3'' & Pm3'' \\ \cdots & \\ Ptn'' & Pmn'' \end{bmatrix}$$

$$= \begin{bmatrix} Pt1 + Pt1' + Pt1'' & Pm1 + Pm1' + Pm1'' \\ Pt2 + Pt2' + Pt2'' & Pm2 + Pm2' + Pm2'' \\ Pt3 + Pt3' + Pt3'' & Pm3 + Pm3' + Pm3'' \\ & \cdots \\ Ptn + Ptn' + Ptn'' & Pmn + Pmn' + Pmn'' \end{bmatrix}$$

Nianheng Wu

Our program structure

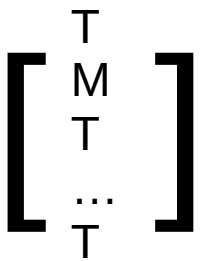
$$\begin{bmatrix} Pt1 + Pt1' + Pt1'' & Pm1 + Pm1' + Pm1'' \\ Pt2 + Pt2' + Pt2'' & Pm2 + Pm2' + Pm2'' \\ Pt3 + Pt3' + Pt3'' & Pm3 + Pm3' + Pm3'' \\ & \dots \\ Ptn + Ptn' + Ptn'' & Pmn + Pmn' + Pmn'' \end{bmatrix} / 3$$

$$= \begin{bmatrix} Pt1\ avg & Pm1\ avg \\ Pt2\ avg & Pm2\ avg \\ Pt3\ avg & Pm3\ avg \\ & \dots \\ Ptn\ avg & Pmn\ avg \end{bmatrix}$$

And we compare each row in this matrix, if Ptn avg> Pmn avg then this sentence is Taiwan Guoyu. And vice versa.

Nianheng Wu

Our program structure



And we get this ←

Then we compare this to the label list we got from test set, and calculate the f1 score