SciPy LATAM • 2019

Get your hyperparameters right!

How to tune your machine learning models with SciPy

Hi, I’m Maria Camila

Scientist &
Engineer

mariacamilarg.me

01 HYPER-PARAMETER TUNING

Machine learning problem

02 HOW DOES SCIPY HELP?

With its optimization tools

03 PRACTICAL EXAMPLE

Perceptron to identify natural language PoS

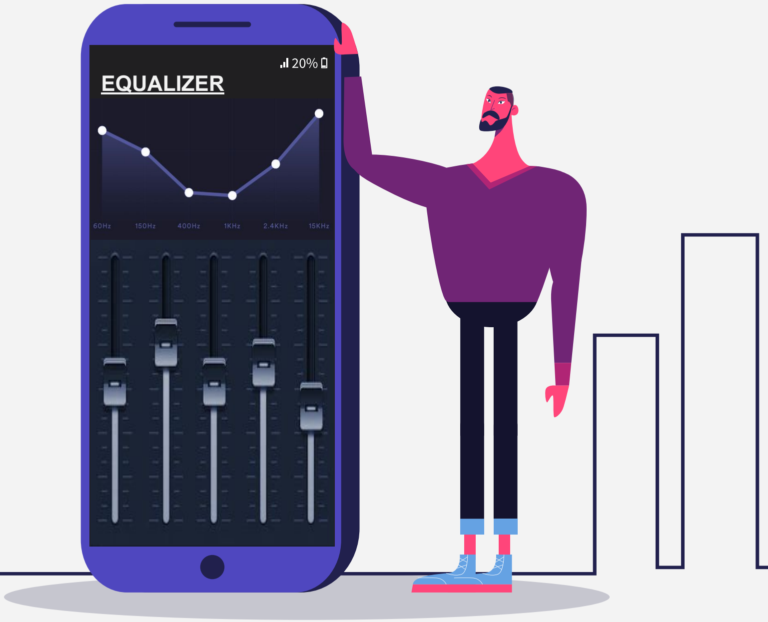
04 TAKE HOME MESSAGE

Summary of this talk

WHAT IS HYPERPARAMETER TUNING?

Choosing a set of optimal* hyperparameters for a learning algorithm.

* minimizing a loss function on validation data



PARAMETER

- Internal to the model
- Estimated or learned from the data
- Required by the model when making predictions.
- Often **not** set manually

- Examples:
  * weights in an artificial neural network
  * support vectors in a SVM
  * coefficients in a linear regression

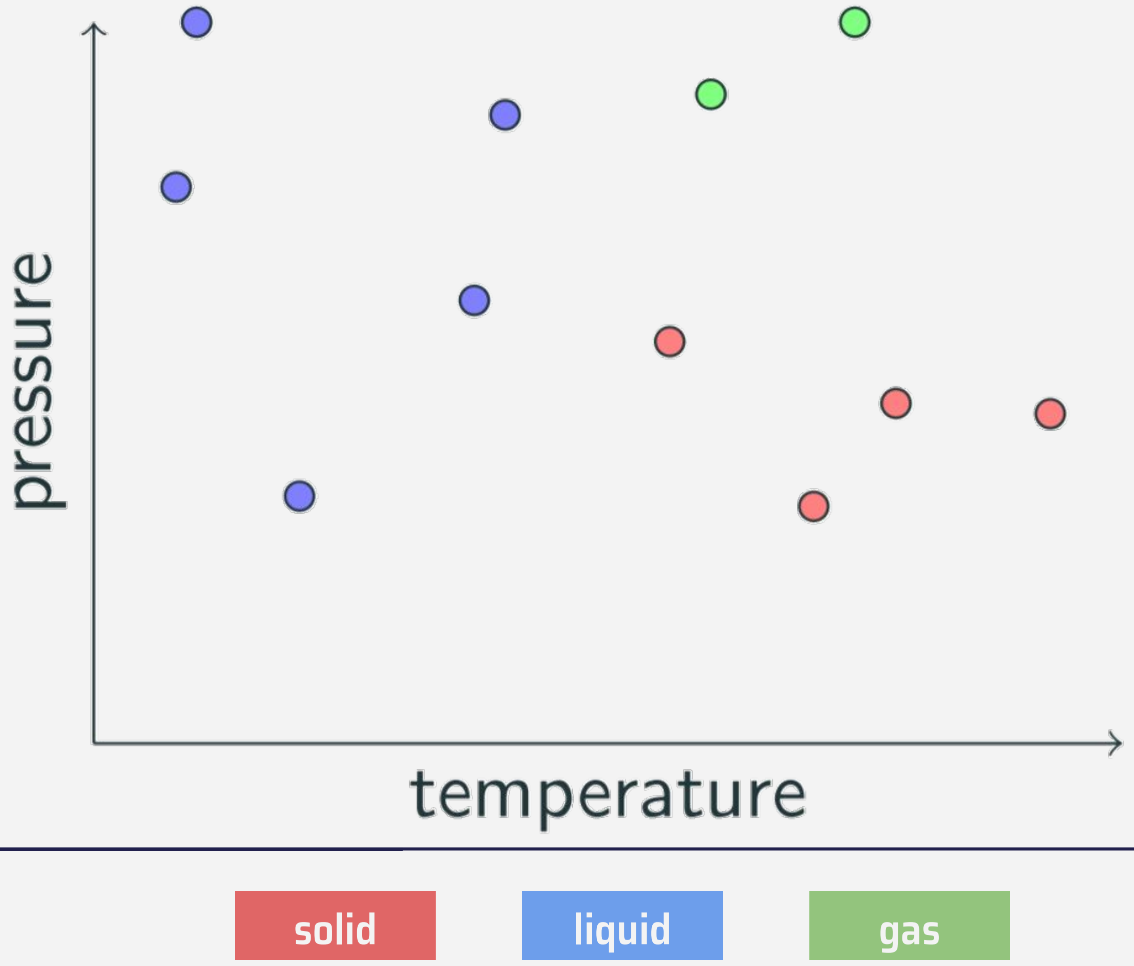
HYPER-PARAMETER

- External to the model
- Can't be estimated or learned from the data
- Helps estimate model parameters
- Often specified, tuned or set using heuristics

- Examples:
  * learning rate for training a neural network
  * the C and sigma hyperparameters for SVM
  * the k in k-nearest neighbors

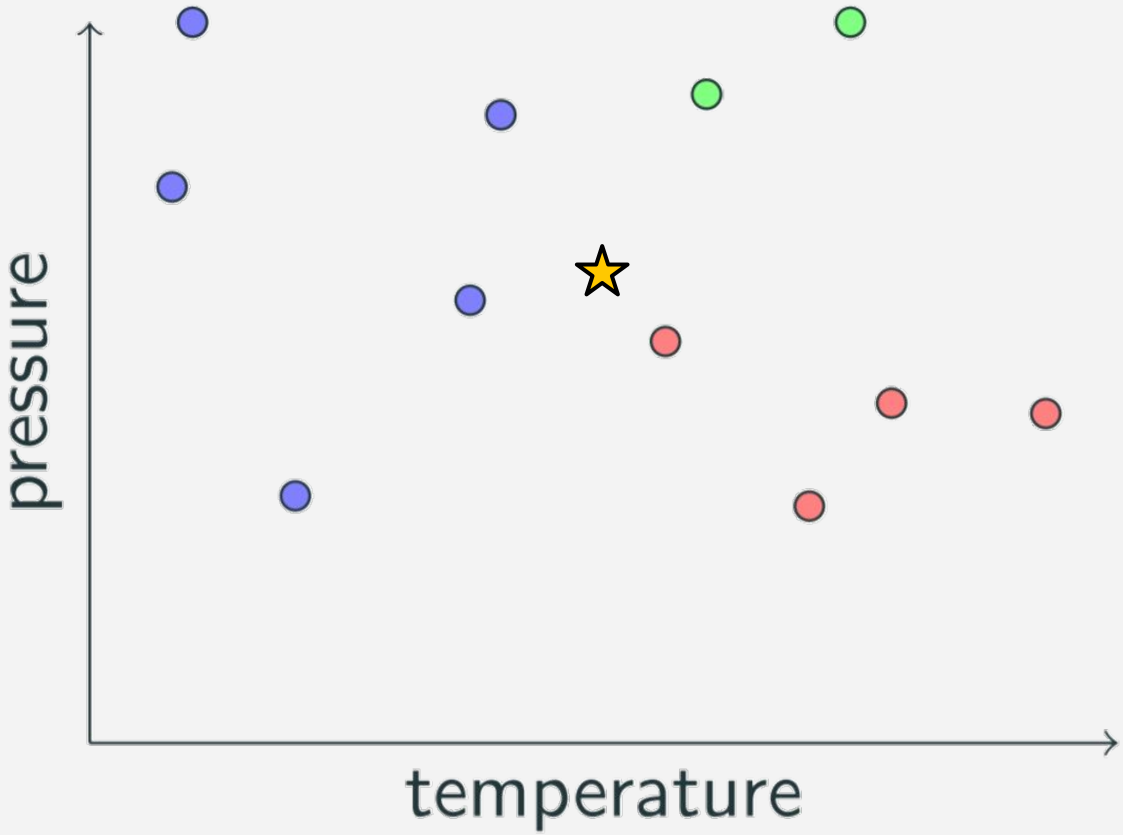
K-NEAREST NEIGHBORS

(classifier)



K-NEAREST NEIGHBORS

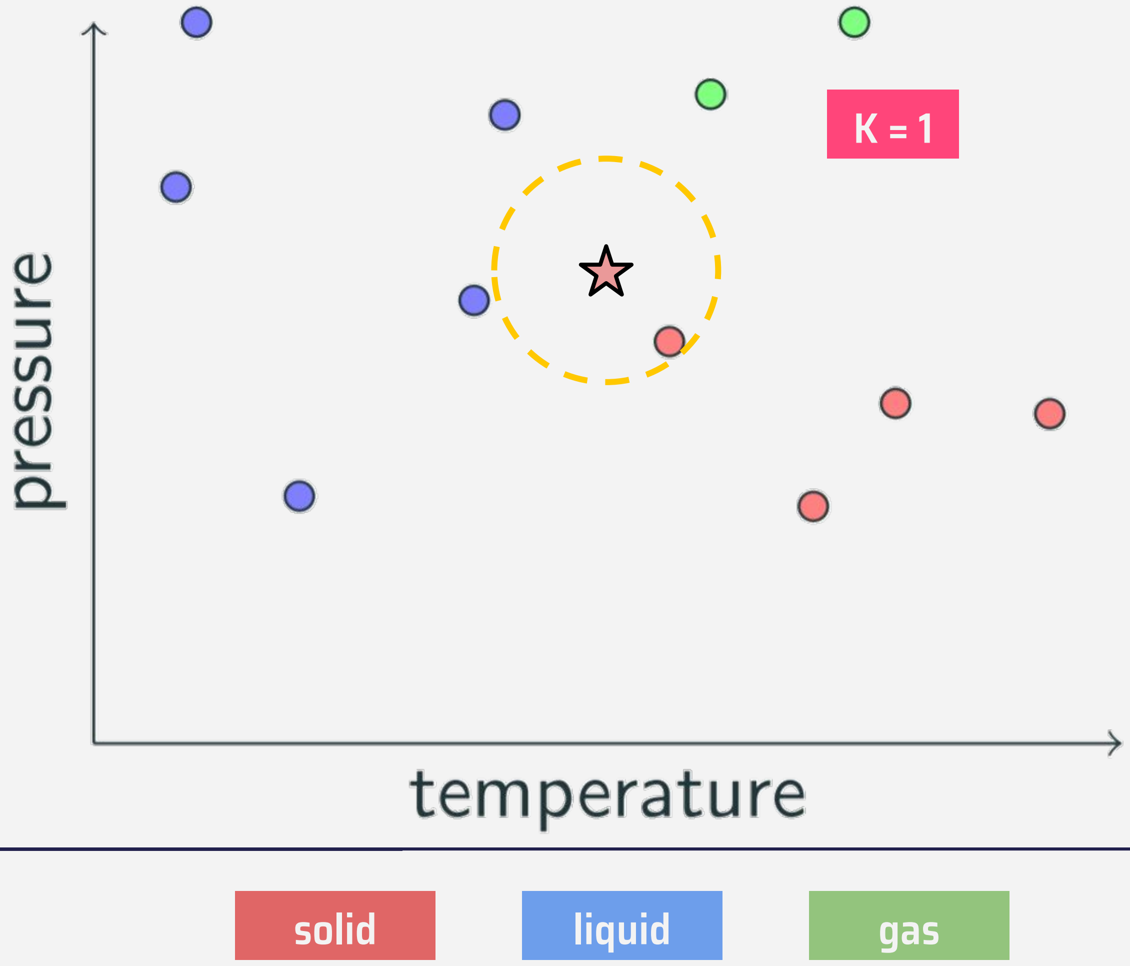
(classifier)



solid liquid gas

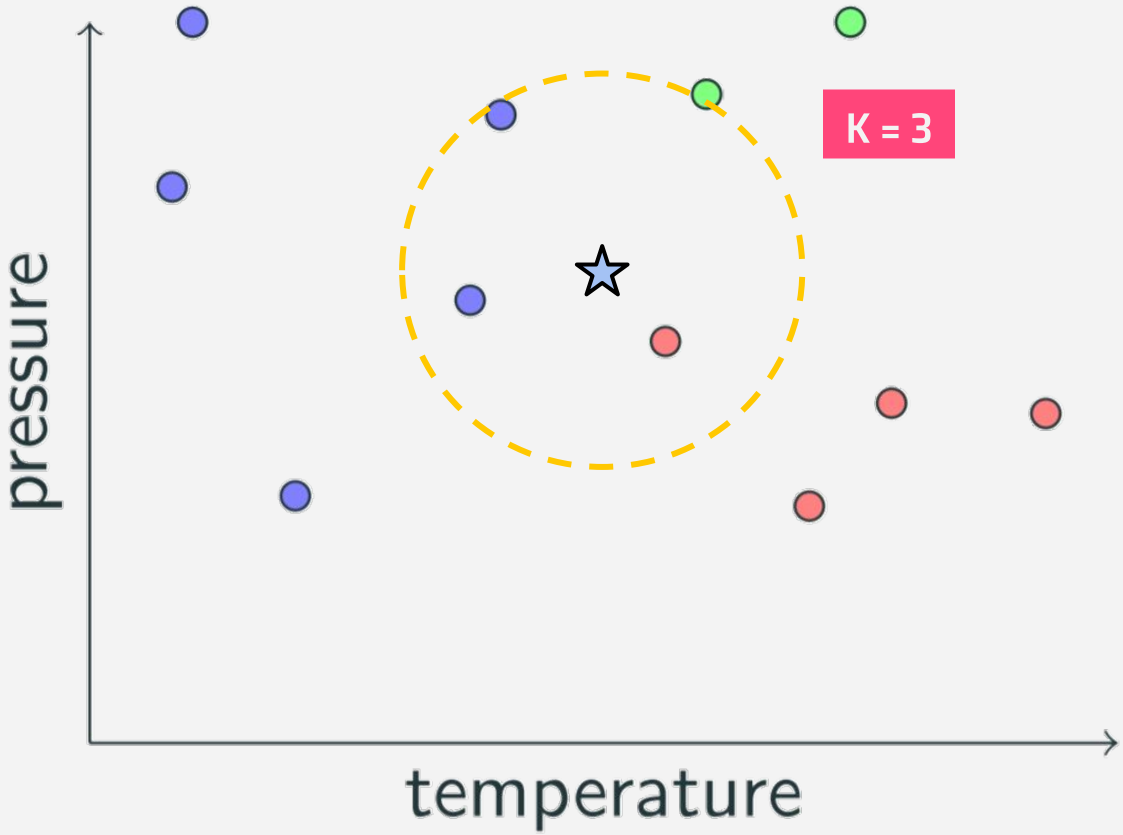
K-NEAREST NEIGHBORS

(classifier)



K-NEAREST NEIGHBORS

(classifier)



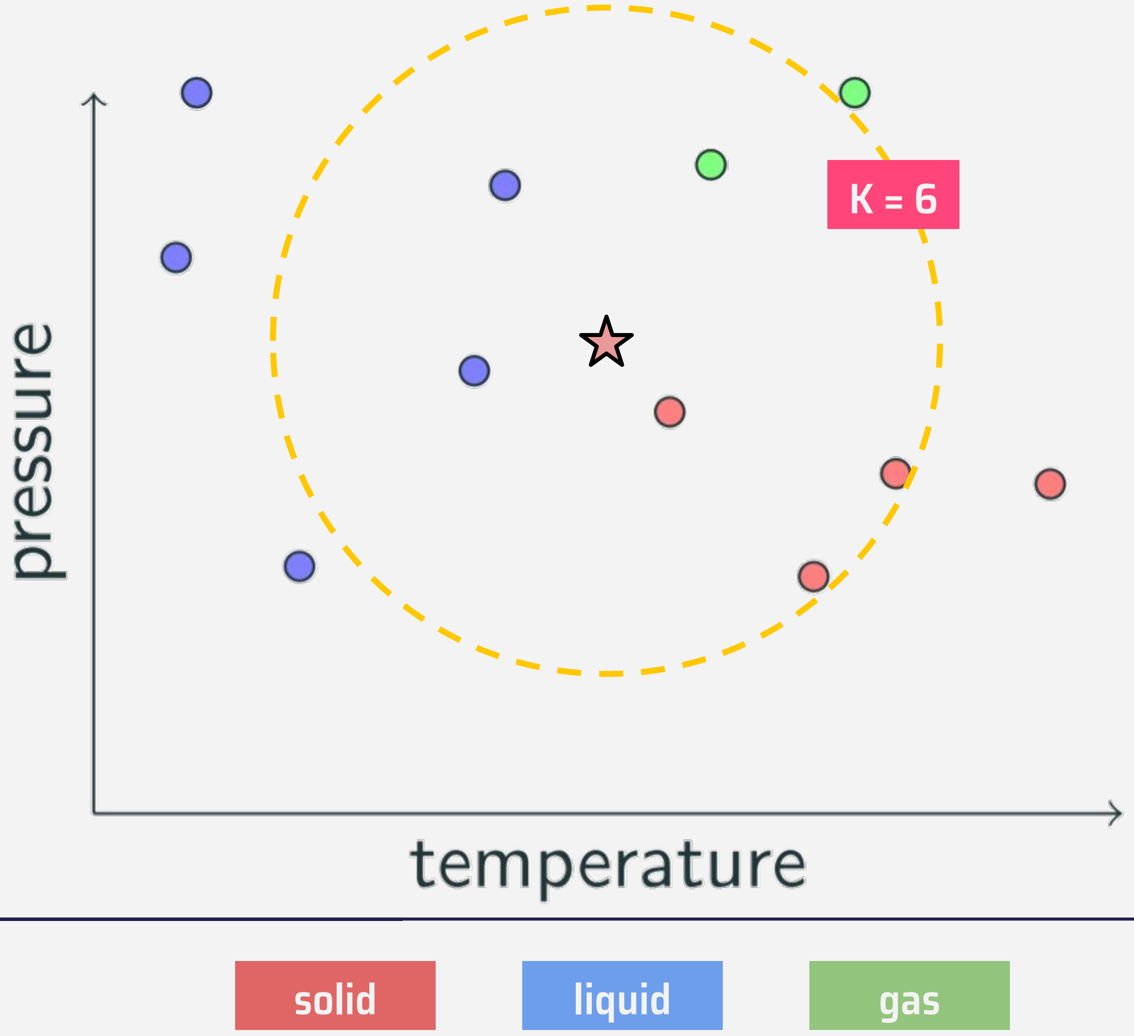
solid

liquid

gas

K-NEAREST NEIGHBORS

(classifier)



HOW DO WE DETERMINE K?

MANUAL
by intuition on the problem

GRID SEARCH
exhaustive searching through the HP space

RANDOM SEARCH
random searching through the HP space

BAYESIAN OPTIMIZATION
choosing the next HP based in past evaluation results

OTHERS
...

HOW DO WE DETERMINE K?

MANUAL
by intuition on the problem

GRID SEARCH
exhaustive searching through the HP space

RANDOM SEARCH
random searching through the HP space

BAYESIAN OPTIMIZATION
choosing the next HP based in past evaluation results

OTHERS
...

HOW DOES SCIPY HELP?

With all the optimization algorithms available within the **SciPy** ecosystem!

```
> from scipy import optimize
```

2

OBJECTIVE FUNCTION:
what we want to optimize

```
def objective_function(hyperparameters):

  m = ml_model(**hyperparameters)

  m.fit(X_train, y_train)

  predictions = m.predict(X_valid)
  rmse = root_mean_squared_error(prediction, y_valid)

  return rmse
```

to minimize

OBJECTIVE FUNCTION:
what we want to optimize

```
def objective_function(hyperparameters):

    m = ml_model(**hyperparameters)

    m.fit(X_train, y_train)

    predictions = m.predict(X_valid)
    rmse = root_mean_squared_error(prediction, y_valid)

    return rmse
```

expensive to compute

GOING ONE STEP FURTHER...

SURROGATE

probability representation of the objective function that selects the next HP to evaluate based on the past

e.g. Tree-structured Parzen Estimator (TPE)

GOING ONE STEP FURTHER...

SURROGATE

probability representation of the objective function that selects the next HP to evaluate based on the past

e.g. Tree-structured Parzen Estimator (TPE)

SELECTION FUNCTION

criteria by which the next set of hyperparameters are chosen from the surrogate function

e.g. expected improvement(EI)

THE IDEA?

To build a probability model of the objective function and use it to select the most promising hyperparameters to evaluate in the true objective function.

+ LIBRARIES

- spearmint
- MOE
- hyperopt
- sdsd

PRACTICAL EXAMPLE

Part-of-Speech (PoS) tagging for out of domain natural language.

3

"Despite being red, Mars is a cold place, not hot. The planet is full of iron oxide dust"

HELENA PATTERSON, 33

"Mercury is the closest planet to the Sun and the smallest one in our Solar System"

JOHN DOE, 45

CHANGE OF DOMAIN

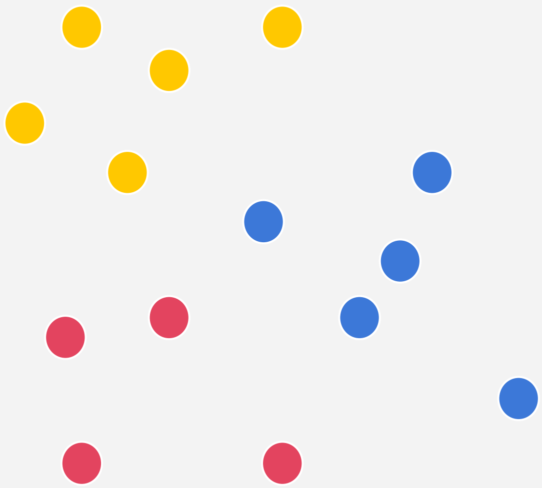
ALGORITHM:

MULTICLASS AVERAGED PERCEPTRON

Each PoS is a class (color)

Each word has different features:

- length
- prefix
- surrounding words
- starts with capital

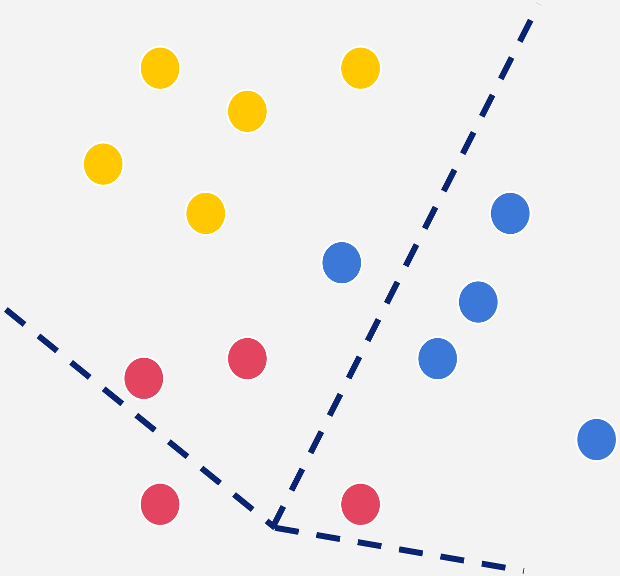


Feature vectors have weights

Word $\longrightarrow$ feature vector

Class $\longrightarrow$ weight vector

Perceptron iterates to find ideal weights in epochs (E).



E = 1

ALGORITHM:

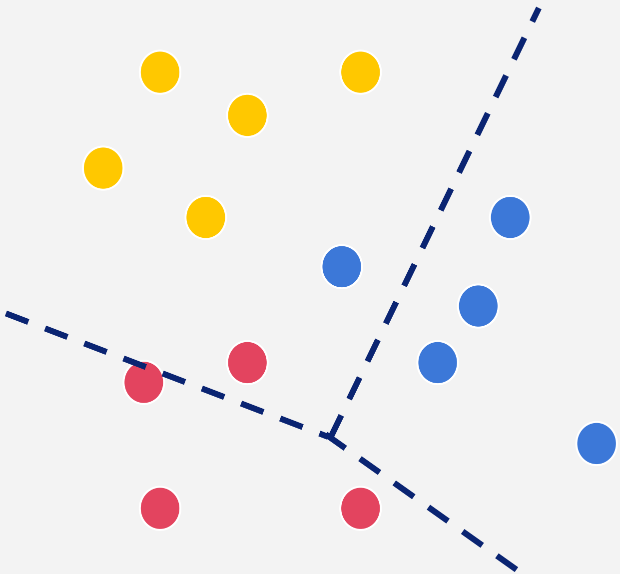
MULTICLASS AVERAGED PERCEPTRON

Word $\longrightarrow$ feature vector

Class $\longrightarrow$ weight vector

Perceptron iterates to find ideal weights in epochs (E).

E = 2



ALGORITHM:

MULTICLASS AVERAGED PERCEPTRON

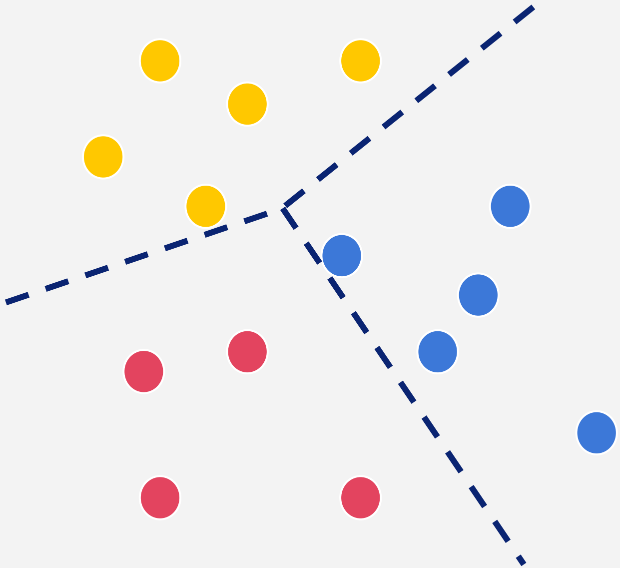
Word $\longrightarrow$ feature vector

Class $\longrightarrow$ weight vector

ALGORITHM:

MULTICLASS AVERAGED PERCEPTRON

Perceptron iterates to find ideal weights in epochs (E).

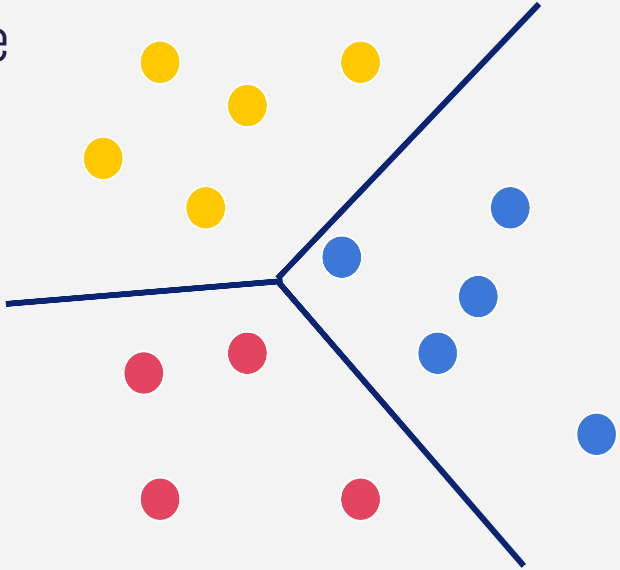


E = n

Finally, it averages all its epochs to compensate for not linearly separable data.

=> Hyperparameter:

E



ALGORITHM:

MULTICLASS AVERAGED PERCEPTRON

OBJECTIVE FUNCTION:

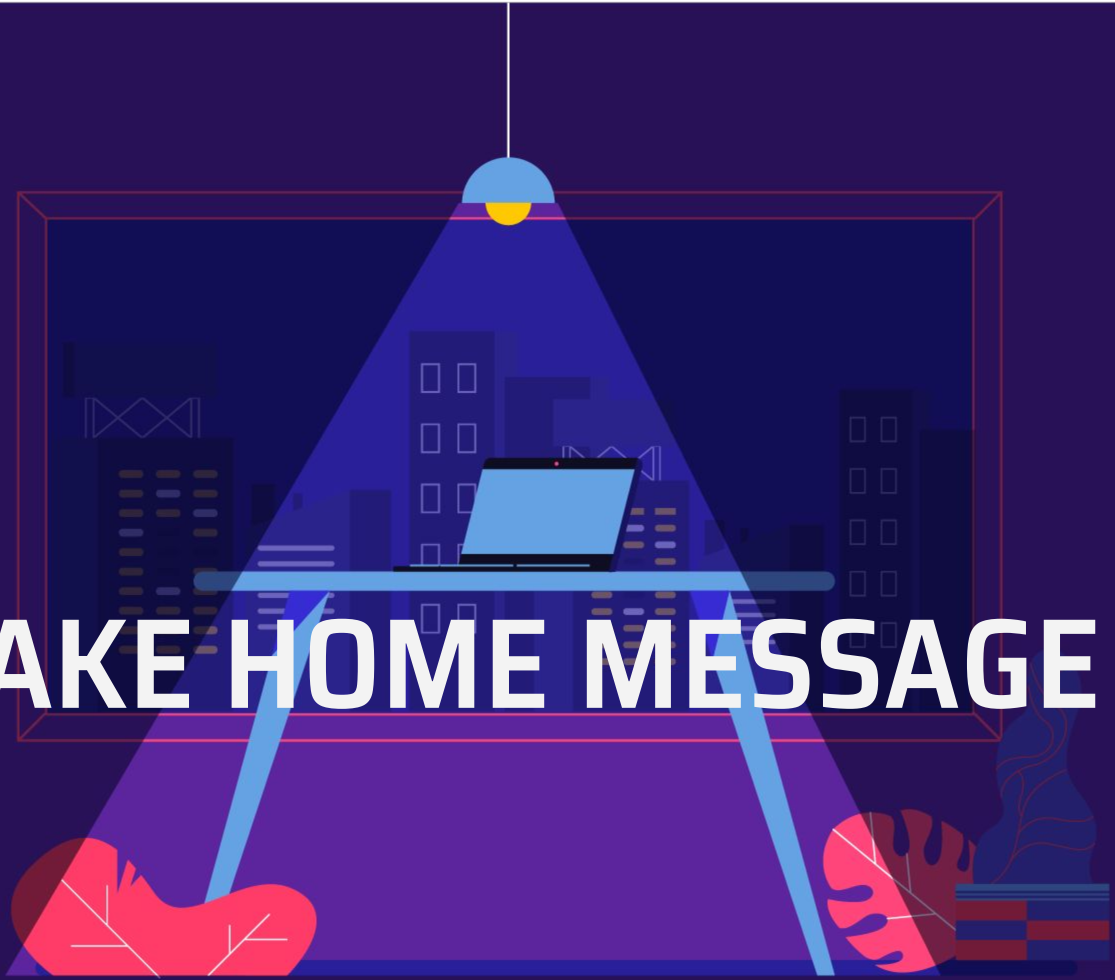
```
def objective_function(epoch):

  model = averaged_perceptron(epochs)

  model.fit(X_train, y_train)

  predictions = m.predict(X_valid)
  rmse = root_mean_squared_error(prediction, y_valid)

  return rmse
```



TAKE HOME MESSAGE

TIL

SciPy optimizations tools can be used to find the hyperparameters of your machine learning model that minimize the validation error leading to a balance between exploration and exploitation.

THANK YOU!

Any questions?

mariacamilarg.me