Supervised learning

Aprendizaje Automático para la Robótica
Máster Universitario en Ingeniería Industrial

Departamento de Automática

Objectives

1. Extend supervised learning algorithms
2. Apply supervised learning to real-world problems

Bibliography

- Géron, Aurélien. *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow*. O'Reilly. 2020
- Müller, Andreas C., Guido, Sarah. *Introduction to Machine Learning with Python*. O'Reilly. 2016

Table of Contents

k-Nearest Neighbors

kNN classification (I)

Diagrama 1-NN y 3-NN.

k-Nearest Neighbors

kNN classification (II)

Diagrama frontera para varios valores de K

k-Nearest Neighbors

kNN regression

TODO

k-Nearest Neighbors

Scikit-learn

TODO

`sklearn.cluster.AgglomerativeClustering`

Constructor arguments:

- `linkage`: 'ward', 'complete', 'average', 'single'

Attributes:

- `n_clusters`: int
- `labels_`: ndarray (n_samples)

Methods: `fit()`, `fit_predict()`

(Scikit-Learn reference)

k-Nearest Neighbors

Summary

Hyperparameters	Advantages	Disadvantages

Linear models

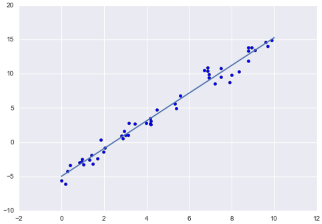
Linear regression (I)

Lineal regression assumes a linear relationship among variables

- This limitation can be easely overcome
- Surprisingly good results in high dimensional spaces

Lineal regression

$$y = a_0 + a_1x_1 + a_2x_2 + \cdots + a_nx_n$$



Linear models (II)

Several methods to fit coefficients

- Ordinary Least Squares (OLS)
- Generalized Least Squares (GSL)
- Weighted Least Squares (WLS)
- Generalized Least Squares with AR Covariance Structure (GLSAR)

Regularization: Term that penalizes complexity

- L1 (Lasso regression)
- L2 (Ridge regression)
- ElasticNet: L1 and L2

Lasso

$\lambda \sum_j^n \beta_j^2$

Ridge

$\lambda \sum_j^n |\beta_j|$

ElasticNet

$\alpha \sum_j^n \beta_j^2 + (1-\alpha) \sum_j^n |\beta_j|$

Linear models

Scikit-learn

TODO

`sklearn.cluster.AgglomerativeClustering`

Constructor arguments:

- `linkage`: 'ward', 'complete', 'average', 'single'

Attributes:

- `n_clusters`: int
- `labels_`: ndarray (n_samples)

Methods: `fit()`, `fit_predict()`

(Scikit-Learn reference)

Linear models

Summary

Hyperparameters	Advantages	Disadvantages

Naive Bayes Classifiers

TODO

Naive Bayes Classifiers

Scikit-learn

`sklearn.cluster.AgglomerativeClustering`

Constructor arguments:

- `linkage`: 'ward', 'complete', 'average', 'single'

Methods: `fit()`, `fit_predict()`

Attributes:

- `n_clusters`: int
- `labels_`: ndarray (n_samples)

(Scikit-Learn reference)

Naive Bayes Classifiers

Summary

Hyperparameters	Advantages	Disadvantages

Decission Trees

TODO

Decission Trees

Scikit-learn

`sklearn.cluster.AgglomerativeClustering`

Constructor arguments:

- `linkage`: 'ward', 'complete', 'average', 'single'

Attributes:

- `n_clusters`: int
- `labels_`: ndarray (n_samples)

Methods: `fit()`, `fit_predict()`

(Scikit-Learn reference)

Decission Trees

Summary

Hyperparameters	Advantages	Disadvantages

Ensembles of Decision Trees

TODO

Ensembles of Decision Trees

Ensembles of Decision Trees : Scikit-learn

`sklearn.cluster.AgglomerativeClustering`

Constructor arguments:

- `linkage`: 'ward', 'complete', 'average', 'single'

Attributes:

- `n_clusters`: int
- `labels_`: ndarray (n_samples)

Methods: `fit()`, `fit_predict()`

(Scikit-Learn reference)

Ensembles of Decision Trees

Summary

Hyperparameters	Advantages	Disadvantages

Support Vector Machines

TODO

Support Vector Machines

Kernelized Support Vector Machines

TODO

Scikit-Learn

Support Vector Machines

Scikit-learn

`sklearn.cluster.AgglomerativeClustering`

Constructor arguments:

- `linkage`: 'ward', 'complete', 'average', 'single'

Methods: `fit()`, `fit_predict()`

Attributes:

- `n_clusters`: int
- `labels_`: ndarray (n_samples)

(Scikit-Learn reference)

Support Vector Machines

Summary

Hyperparameters	Advantages	Disadvantages

A

B

TODO

A

B: Scikit-learn

`sklearn.cluster.AgglomerativeClustering`

Constructor arguments:

- `linkage`: 'ward', 'complete', 'average', 'single'

Attributes:

- `n_clusters`: int
- `labels_`: ndarray (n_samples)

Methods: `fit()`, `fit_predict()`

(Scikit-Learn reference)

A

B: Summary

Hyperparameters	Advantages	Disadvantages

Algorithms

ARIMA (I)

AR: Autoregressive model

- Current observation depends on the last p observations
- Long term memory

MA: Moving Average model

- Current observation linearly depends on the last q innovations
- Short term memory

AR(p)

$$X_t = c + \sum_{i=1}^{p} \phi_i X_{t-1} + \epsilon_t$$

MA(q)

$$X_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1} + \ldots + \theta_q \epsilon_{t-q}$$

ARMA model = AR + MA

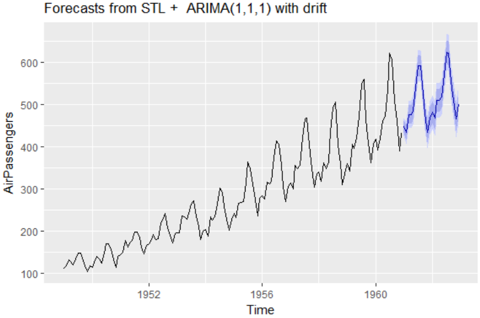
- ARMA(p, q): Two hyperparameters, p and q

Algorithms

ARIMA (II)

ARIMA = AR + i + MA (AR integrated MA)

- ARIMA(p, d, q)
- Three integer parameters: p, q and d (in practice, low order models)



(Source)

autoarima: search over p, q and d