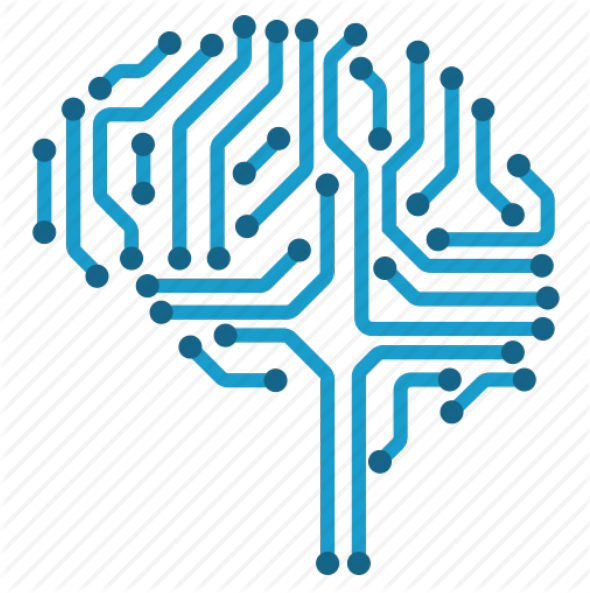
Big Data e Business Intelligence



Machine Learning

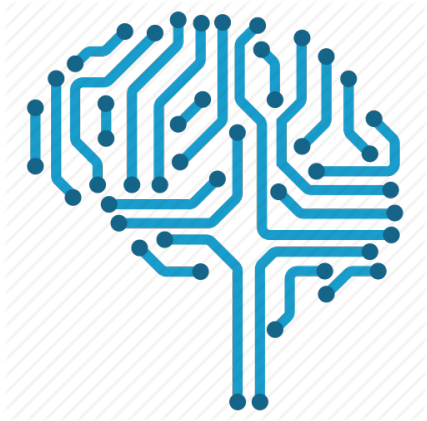
Giulio Angiani - UniPr

Machine Learning - primi passi

Machine Learning

Cos'è ML

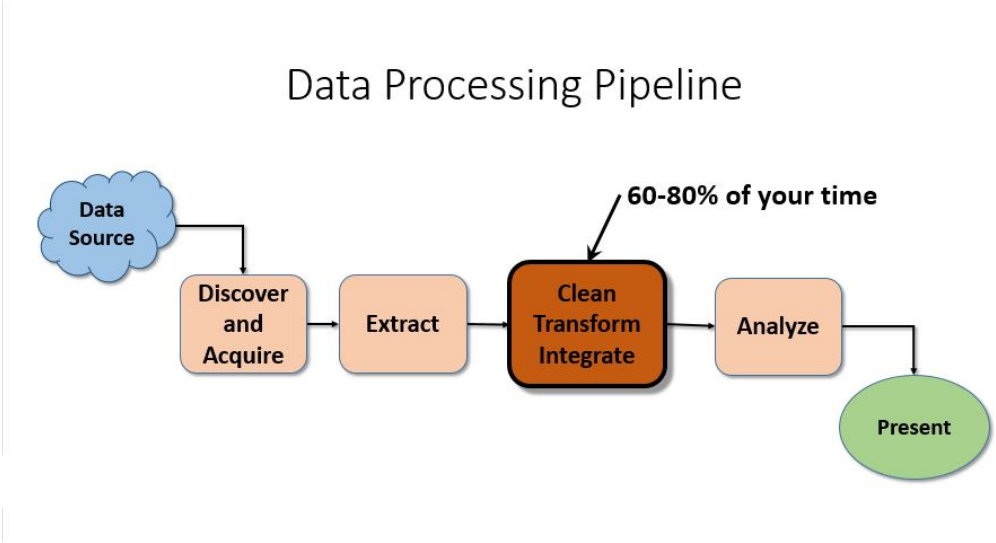
- Vari tipi di Apprendimento automatico
  - Classificazione
  - Regressione
  - Clustering
  - Associazione
  - Previsione
- Tecniche
  - Alberi decisionali
  - Reti neurali
  - Tecniche statistiche
  - Regole di induzione



Machine Learning

I cinque passi della scienza dei dati

- Porre una domanda interessante
- **Ottenere i dati**
- **Esplorare i dati**
- **Creare un modello per i dati**
- Comunicare e presentare i risultati



Machine Learning

l' **Hello world!** del ML

Problema di classificazione

- **apprendimento supervisionato**

Iris Data Set

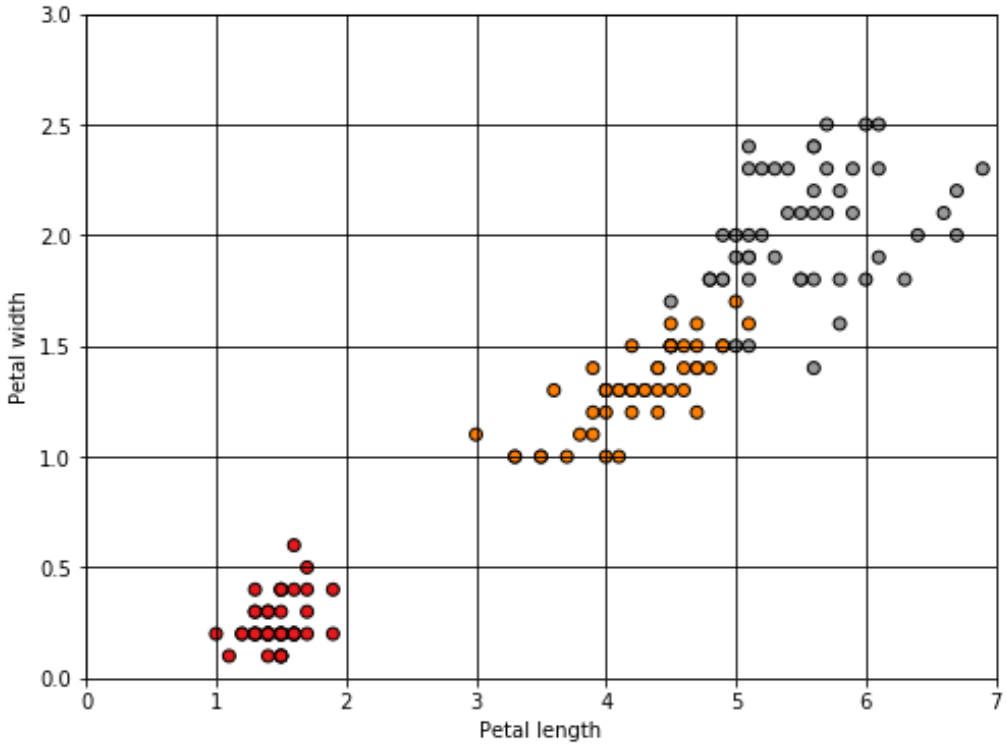
Iris Versicolor **Iris Setosa** **Iris Virginica**

DATASET

```
Petal length,Petal width,Class
1.4,0.2,Iris-setosa
1.4,0.2,Iris-setosa
...
5.0,1.7,Iris-versicolor
4.5,1.5,Iris-versicolor
...
5.4,2.3,Iris-virginica
5.1,1.8,Iris-virginica
```

Iris 2D - distribuzione

distribuzione degli items come punti su un piano



Iris-setosa Iris-versicolor Iris-virginica

Iris 2D - Addestramento

formalismo standard

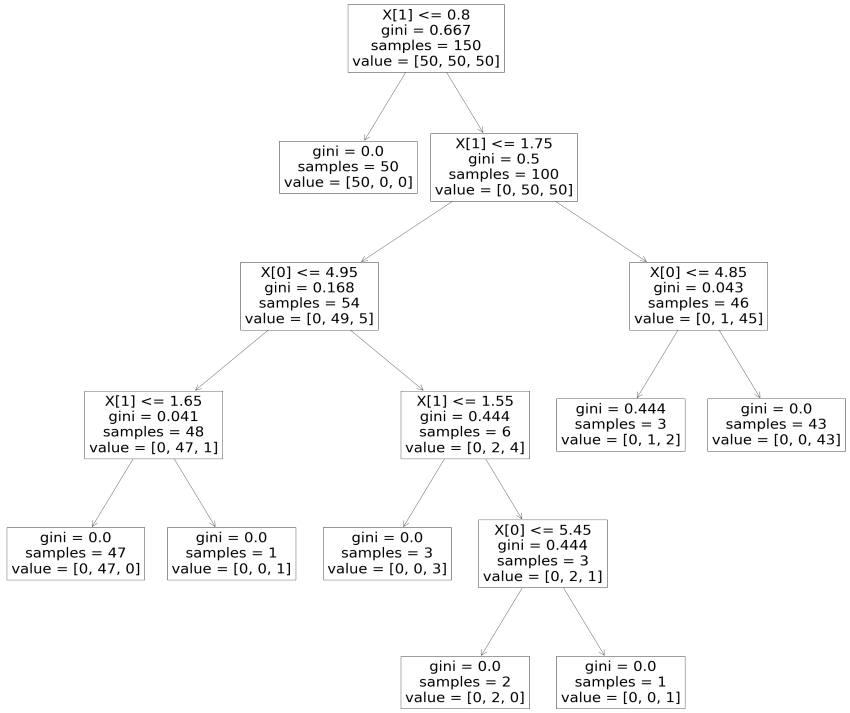
Si indica con **X** l'insieme dei valori delle colonne delle features Si indica con **y** la colonna delle classi

PYTHON

```
# importo il modulo tree da sklearn (scikit-learn) libreria per ML
from sklearn import tree
# fra gli alberi scelgo un albero di decisione
clf = tree.DecisionTreeClassifier()
# l'operazione di fit addestra il classificatore coi dati presenti in X e y
clf = clf.fit(X, y)
```

Iris 2D - Albero decisionale

Cosa costruisce l'operazione di **fit** di un **DecisionTreeClassifier**

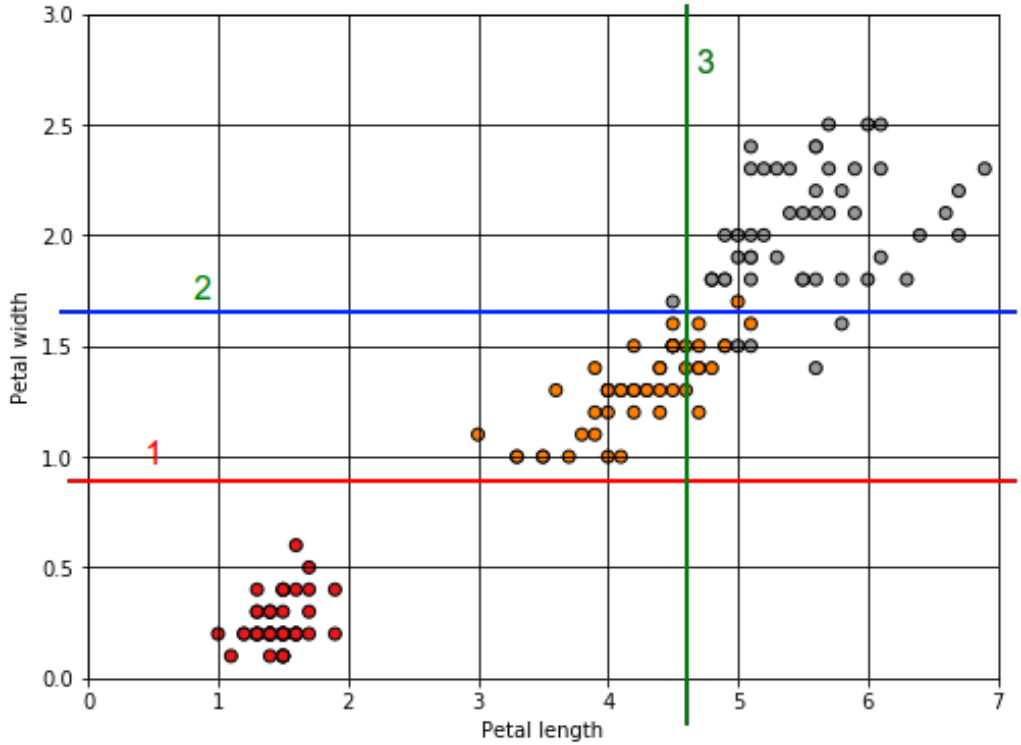


Iris 2D - Albero decisionale (testo)

```
|--- petal_width <= 0.80',
| |--- class: 0',
|--- petal_width > 0.80',
| |--- petal_width <= 1.75',
| | |--- petal_length <= 4.95',
| | | |--- petal_width <= 1.65',
| | | | |--- class: 1',
| | | |--- petal_width > 1.65',
| | | | |--- class: 2',
| | |--- petal_length > 4.95',
| | | |--- petal_width <= 1.55',
| | | | |--- class: 2',
| | | |--- petal_width > 1.55',
| | | | |--- petal_length <= 5.45',
| | | | | |--- class: 1',
| | | | |--- petal_length > 5.45',
| | | | | |--- class: 2',
| |--- petal_width > 1.75',
| | |--- petal_length <= 4.85',
| | | |--- class: 2',
| | |--- petal_length > 4.85',
| | | |--- class: 2',
```

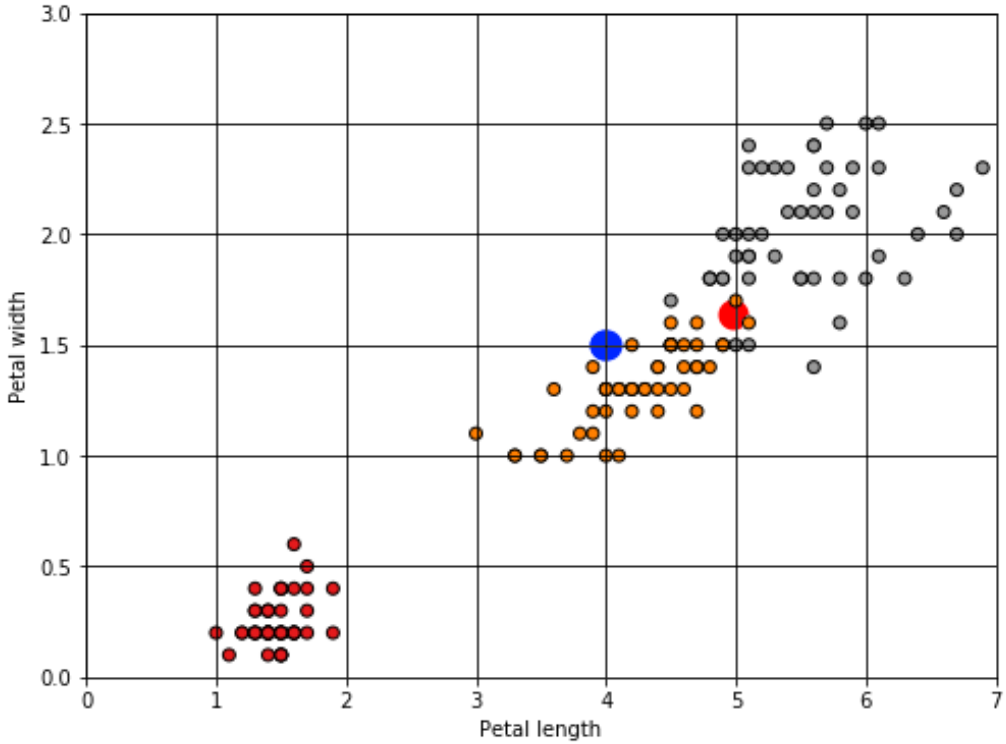
Iris 2D - Albero decisionale (grafica)

tagli decisionali dell'algoritmo J48



Iris 2D - predizione

- prendiamo altri items
- per esempio: (4, 1.5) e (5, 1.6)
- dove si posizionano?



Iris 2D - predizione

- usiamo sklearn

PYTHON

```
clf.predict([[4, 1.5]])
```

OUTPUT

```
array([1])
```

- Predizione sulla classe 2 (Iris-versicolor)

PYTHON

```
clf.predict([[5, 1.6]])
clf.predict_proba([[5, 1.6]])
```

OUTPUT

```
array([1]) # classe 2
array([[ 0.,  1.,  0.]]) # 100% confidenza
```

Iris 2D - predizione

- un esempio più distante dal dataset (outlier)

PYTHON

```
clf.predict([[2, 2]])
clf.predict_proba([[2, 2]])
```

OUTPUT

```
array([2]) # classe 2
array([[ 0,  0.33333333, 0.66666667]]) # 33% e 66% di confidenza
```

Iris 2D - SVM

PYTHON

```python
# cambiamo classificatore
# Support Vector Machine
from sklearn.svm import SVC
clf = SVC(gamma='auto', probability=True)
clf.fit(X, y)
print(clf.predict([[4, 1.5]]))
print(clf.predict_proba([[4, 1.5]]))
```

OUTPUT

```
array([1])
array([[ 0.00839442,  0.97857937,  0.01302621]])
```

- la confidence è spalmata sulle tre classi

Iris 2D - Valutazione del classificatore

Le modalità di valutazione sono diverse a seconda del tipo di algoritmo

i modelli di classificazione
i modelli di regressione
i modelli di clustering

[cit. slides Mordonini]

- Dataset Split
- Cross Validation

Iris 2D - Valutazione del classificatore

Dataset Split

- tipicamente
  - 66% train dataset
  - 33% test dataset
- nel nostro caso: 150 elementi
  - 100 training
  - 50 test
- **classi bilanciate**

Iris 2D - Valutazione del classificatore

- Dividiamo il dataset (X,y) in [(X_train, y_train) e (X_test, y_test)]

PYTHON

```python
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
print(len(X_train), len(y_train))
print(len(X_test), len(y_test))
```

OUTPUT

```
100 100
50 50
```

- addestriamo sul **solo** training set

PYTHON

```python
clf.fit(X_train, y_train)
```

Iris 2D - Valutazione del classificatore

- Testiamo il classificatore addestrato sul **solo** training set
- sia sullo stesso *training set* che sul *test set*

PYTHON

```
print("TRAIN SET", clf.score(X_train, y_train))
print("TEST  SET", clf.score(X_test, y_test))
```

OUTPUT

```
TRAIN SET 0.99  # errore di 1%
TEST  SET 0.98  # errore di 2%
```

Iris 2D - Valutazione del classificatore

- quali sono gli errori?

PYTHON

```
print("Errori in training set")
predictions = clf.predict(X_train)
for elem, prediction, label in zip(X_train, predictions, y_train):
  if prediction != label:
    print(elem, 'has been classified as ', prediction, 'and should be ', label)
# similmente per test set...
```

OUTPUT

```
Errori in training set
[ 4.8  1.8] has been classified as  2 and should be  1

Errori in test set
[ 5.1  1.5] has been classified as  1 and should be  2
```

Iris 2D - Valutazione del classificatore

- Matrice di confusione
  - quanti elementi di una certa classe sono associati alle altre classi

PYTHON

```python
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_train, clf.predict(X_train))
print("CM per Train set\n", cm)
# similmente per test set...
```

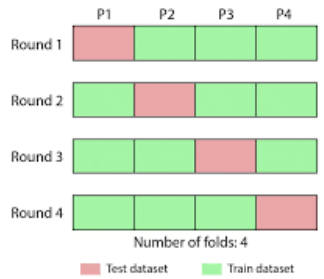
OUTPUT

```
CM per Train set
[[31  0  0]
 [ 0 34  1]
 [ 0  0 34]]


CM per Test set
[[19  0  0]
[  0 15  0]
[  0  1 15]]
```

Iris 2D - Valutazione del classificatore

Cross Validation



- parametri
  - folds (default 5)

PYTHON

```
from sklearn.model_selection import cross_val_score
print(cross_val_score(clf, X, y, cv=4))
```

OUTPUT

```
[ 0.97435897  0.94871795  0.91666667  0.97222222]
```

- media di accuracy: **0.95**

Iris 2D - Valutazione del classificatore

- si lascia per esercizio al lettore
  - addestramento di due classificatore SVM e RF (Random Forest)
  - valutazione dell'accuratezza con CV
  - valutazione dell'accuratezza con Dataset Split
  - visualizzazione matrici di confusione
  - confronto prestazioni fra SVM e RF

Giulio Angiani
Universita' degli Studi di Parma