MEDNARODNA
PODIPLOMSKA ŠOLA
JOŽEFA STEFANA

INFORMATION AND COMMUNICATION TECHNOLOGIES
PhD study programme

Data Mining and Knowledge Discovery

Petra Kralj Novak

December 18, 2019

http://kt.ijs.si/petra_kralj/dmkd3.html

So far …

- Nov. 11. 2019
  - Basic classification
  - Orange hands on data visualization and classification
- Dec. 11 2019
  - Fitting and overfitting
  - Data leakage
  - Decision boundary
  - Evaluation methods
  - Classification evaluation metrics: confusion matrix, TP, FP, TN, FN, accuracy, precision, recall, F1, ROC
  - Imbalanced data and unequal misclassification costs
  - Probabilistic classification
  - Naïve Bayes classifier

Assignment 1: Home reading

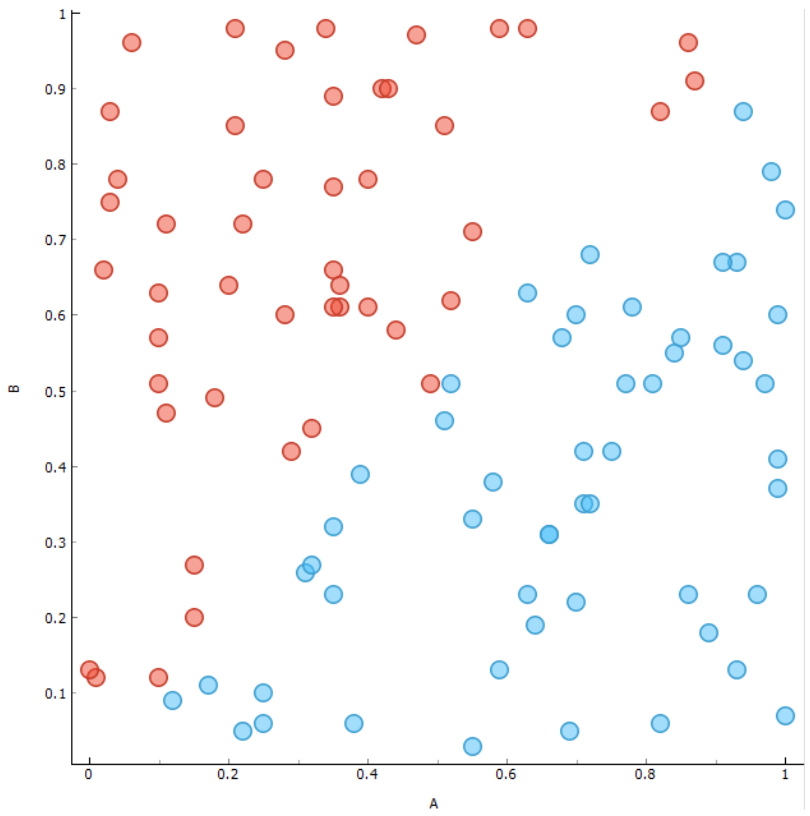
Read: Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning, Second edition*. New York: Springer series in statistics. https://web.stanford.edu/~hastie/Papers/ESLII.pdf

Pages 9 – 18:

Assignment 2: Decision boundary

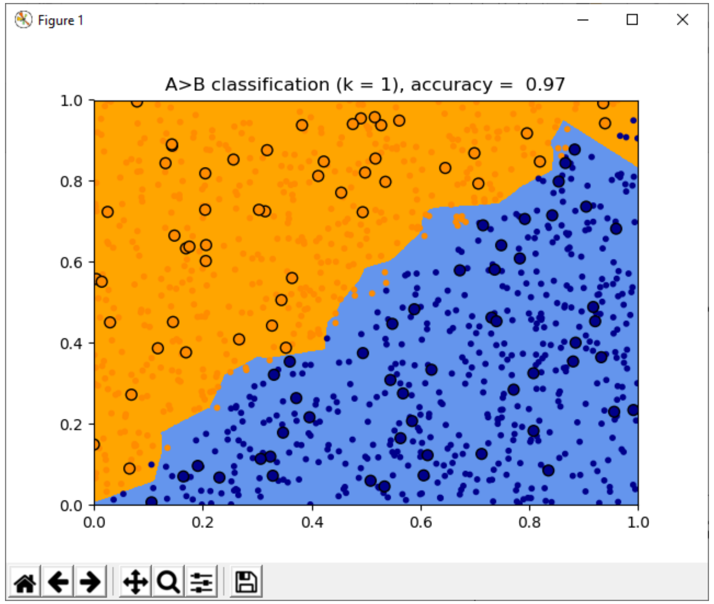
- What is a decision boundary like for KNN?
  - K=1
  - K=3
  - K=10

You can do it by hand, in Orange or in SciKit.

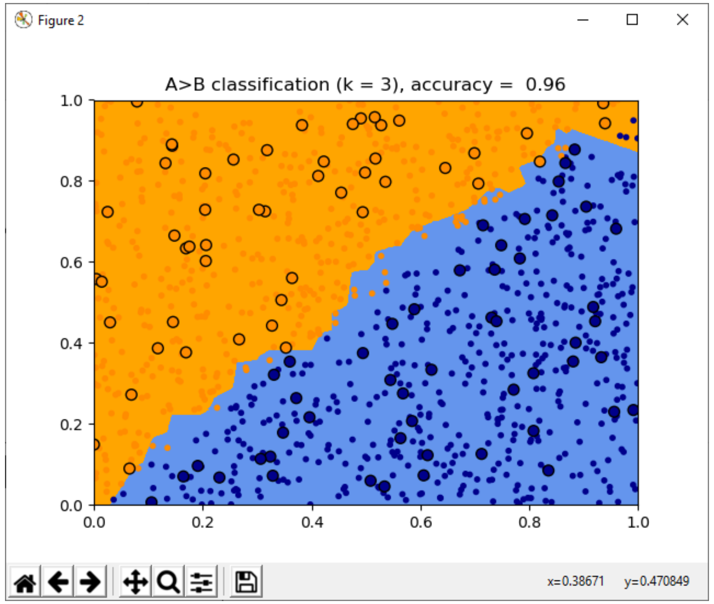


What is a decision boundary like for KNN

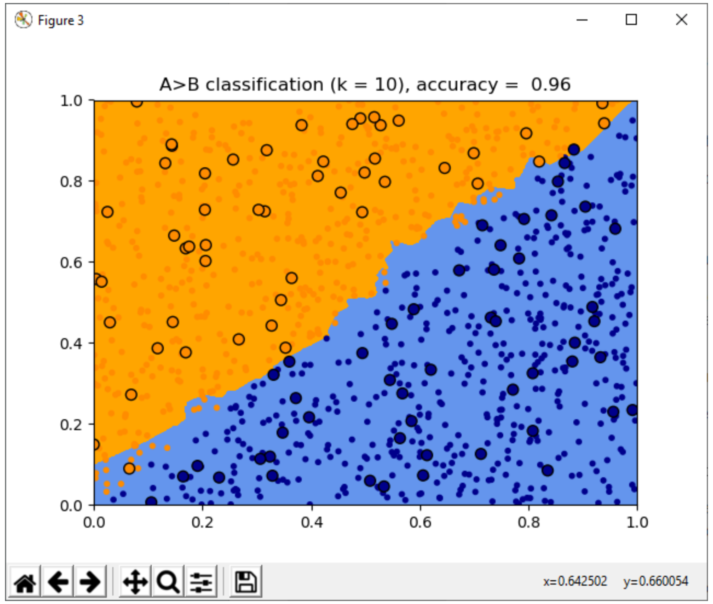
K=1



K=3



K=10



The large circles are the training set, the small ones are the test set – colored by the real labels. The background colors represent the decision boundary.

The source code for this is available at

Assignment 3: Confusion matrix

Titanic

Predicted

Actual	no	yes	Σ
no	1364	126	1490
yes	362	349	711
Σ	1726	475	2201

Car

Predicted

Actual	unacc	acc	good	v-good	Σ
unacc	1154	54	2	0	1210
acc	94	276	14	0	384
good	0	44	22	3	69
v-good	0	25	0	40	65
Σ	1248	399	38	43	1728

	Titanic	Car
Number of examples		
Number of classes		
Number of examples in each class		
Number of examples classified in individual classes		
Number of misclassified examples		
Classification accuracy		

Assignment 4: F1

- Express F1 in terms of TP, FP, TN, FN

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}$$

		Predicted class		Total instances
		+	−	
Actual class	+	TP	FN	P
	−	FP	TN	N

Exercise: Naïve Bayes Classifier

Color	Size	Time	Caught
black	large	day	YES
white	small	night	YES
black	small	day	YES
red	large	night	NO
black	large	night	NO
white	large	night	NO

$$P(c_i \mid a_1 = v_1, a_2 = v_2, \ldots, a_j = v_j) \propto P(c_i) \times \prod_{j=1}^{n} P(a_j = v_j \mid class = c_i)$$

- Does the spider catch a white ant during the night?
- Does the spider catch the big black ant at daytime?

Exercise: Naïve Bayes Classifier

Does the spider catch a white ant during the night?

Color	Size	Time	Caught
black	large	day	YES
white	small	night	YES
black	small	day	YES
red	large	night	NO
black	large	night	NO
white	large	night	NO

$$P(c_i \mid a_1 = v_1, a_2 = v_2, \ldots, a_j = v_j) \propto P(c_i) \times \prod_{j=1}^{n} P(a_j = v_j \mid class = c_i)$$

$$v_1 = \text{“}Color = white\text{”}$$

$$v_2 = \text{“}Time = night\text{”}$$

$$c_1 = YES$$

$$c_2 = NO$$

$$\begin{aligned} P(C_1 \mid v_1, v_2) &= \\ &= P(\text{YES} \mid C = w, T = n) \\ &= P(\text{YES}) \cdot P(C = w \mid \text{YES}) \cdot P(T = n \mid \text{YES}) \\ &= \frac{1}{2} \cdot \frac{1}{3} \cdot \frac{1}{3} \\ &= \frac{1}{18} \end{aligned}$$

$$\begin{aligned} P(C_2 \mid v_1, v_2) &= \\ &= P(\text{NO} \mid C = w, T = n) \\ &= P(\text{NO}) \cdot P(C = w \mid \text{NO}) \cdot P(T = n \mid \text{NO}) \\ &= \frac{1}{2} \cdot \frac{1}{3} \cdot 1 \\ &= \frac{1}{6} \end{aligned}$$

Exercise: Naïve Bayes Classifier

Does the spider catch the big black ant at daytime?

Color	Size	Time	Caught
black	large	day	YES
white	small	night	YES
black	small	day	YES
red	large	night	NO
black	large	night	NO
white	large	night	NO

$$P(c_i \mid a_1 = v_1, a_2 = v_2, \ldots, a_j = v_j) \propto P(c_i) \times \prod_{j=1}^{n} P(a_j = v_j \mid class = c_i)$$

Ant 2: Color = black, Size = large, Time = day

$$v_1 = \text{“}Color = black\text{”} = \text{“}C = b\text{”}$$
$$v_2 = \text{“}Size = large\text{”} = \text{“}S = l\text{”}$$
$$v_3 = \text{“}Time = day\text{”} = \text{“}T = d\text{”}$$
$$c_1 = YES$$
$$c_2 = NO$$

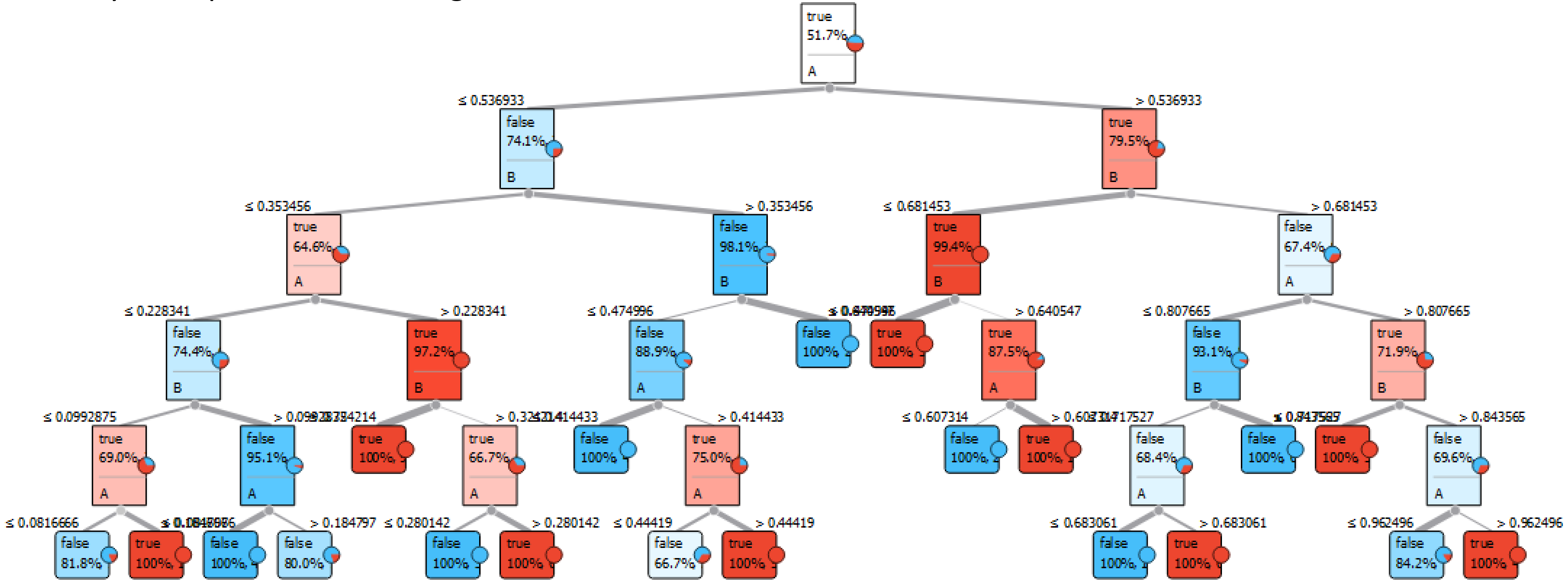
$$\begin{aligned} P(C_1|v_1, v_2, v_3) &= \\ &= P(\text{YES}|C = b, S = l, T = d) \\ &= P(\text{YES}) \cdot P(C = b|\text{YES}) \cdot P(S = l|\text{YES}) \cdot P(T = d|\text{YES}) \\ &= \frac{1}{2} \cdot \frac{2}{3} \cdot \frac{1}{3} \cdot \frac{2}{3} \\ &= \frac{4}{54} = \frac{2}{27} \end{aligned}$$

$$\begin{aligned} P(C_2|v_1, v_2, v_3) &= \\ &= P(\text{NO}|C = b, S = l, T = d) \\ &= P(\text{NO}) \cdot P(C = b|\text{NO}) \cdot P(S = l|\text{NO}) \cdot P(T = d|\text{NO}) \\ &= \frac{1}{2} \cdot \frac{1}{3} \cdot \frac{3}{3} \cdot 0 \\ &= 0 \end{aligned}$$

Probability Estimation

A decision tree of depth 5

How many examples are on average in each leaf at level 5?



Estimating probability

- In machine learning we often estimate probabilities from small samples of data and their subsets:
  - In the 5th depth of a decision tree we have just about 1/32 of all training examples.
- Estimate the probability based on the amount of evidence and of the prior probability
  - Coin flip: prior probability 50% - 50%
  - One coin flip does not make us believe that the probability of heads is 100%
  - More evidence can make us suspect that the coin is biased

Estimating probability

Relative frequency

- **P(c) = n(c) /N**
- A disadvantage of using relative frequencies for probability estimation arises with small sample sizes, especially if the probabilities are either very close to zero, or very close to one.
- In our spider example:

  P(Time=day|caught=NO) =

  = 0/3 = 0

n(c) … number of times an event occurred
N … total number of events
k … number of possible outcomes

Relative frequency vs. Laplace estimate

Relative frequency

- **P(c) = n(c) /N**
- A disadvantage of using relative frequencies for probability estimation arises with small sample sizes, especially if the probabilities are either very close to zero, or very close to one.
- In our spider example:

  P(Time=day|caught=NO) =

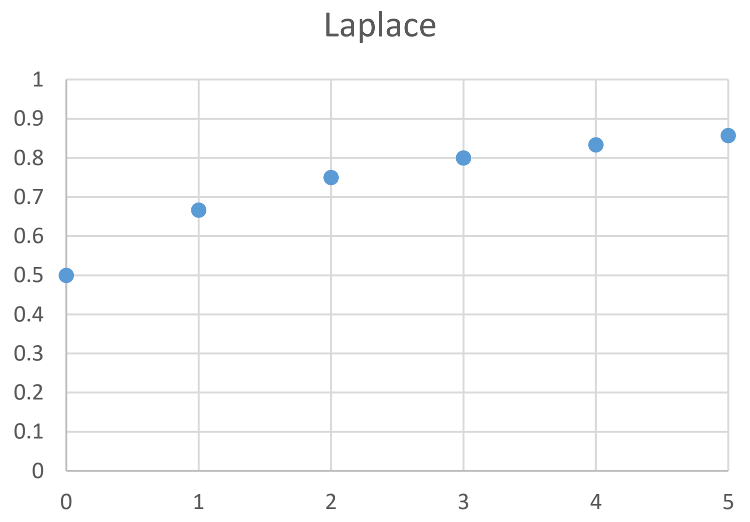
  = 0/3 = 0

n(c) … number of times an event occurred
N … total number of events
k … number of possible outcomes

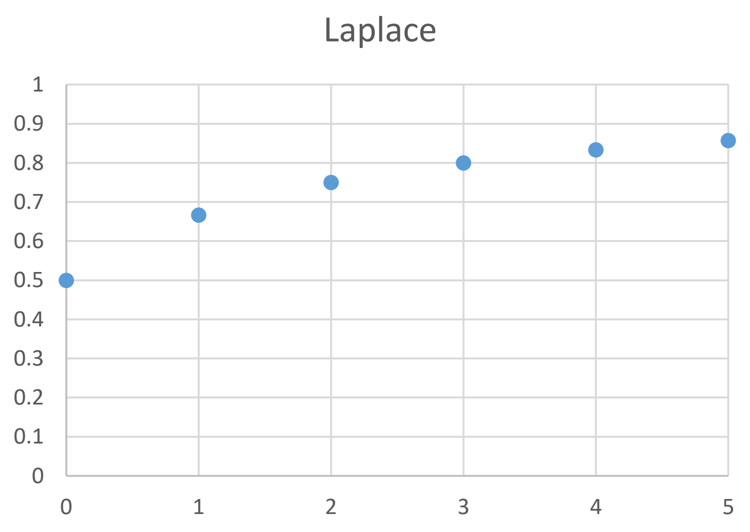
Laplace estimate

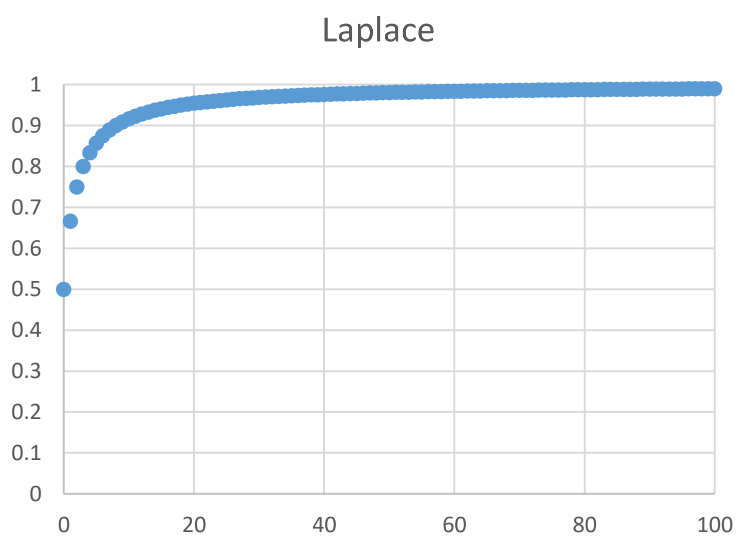
- Assumes uniform prior distribution over the probabilities for each possible event
- **P(c) = (n(c) + 1) / (N + k)**
- In our spider example: P(Time=day|caught=NO) = (0+1)/(3+2) = 1/5
- With lots of evidence it approximates relative frequency
- If there were 300 cases when the spider didn't catch ants at night: P(Time=day|caught=NO) = (0+1)/(300+2) = 1/302 = 0.003
- With Laplace estimate probabilities can never be 0.

Laplace estimate (Additive smoothing)



Laplace estimate (Additive smoothing)





Exercise

- Estimate the probabilities of C1 and C2 in the table below by relative frequency and Laplace estimate.

- **P(c) = (n(c) + 1) / (N + k)**

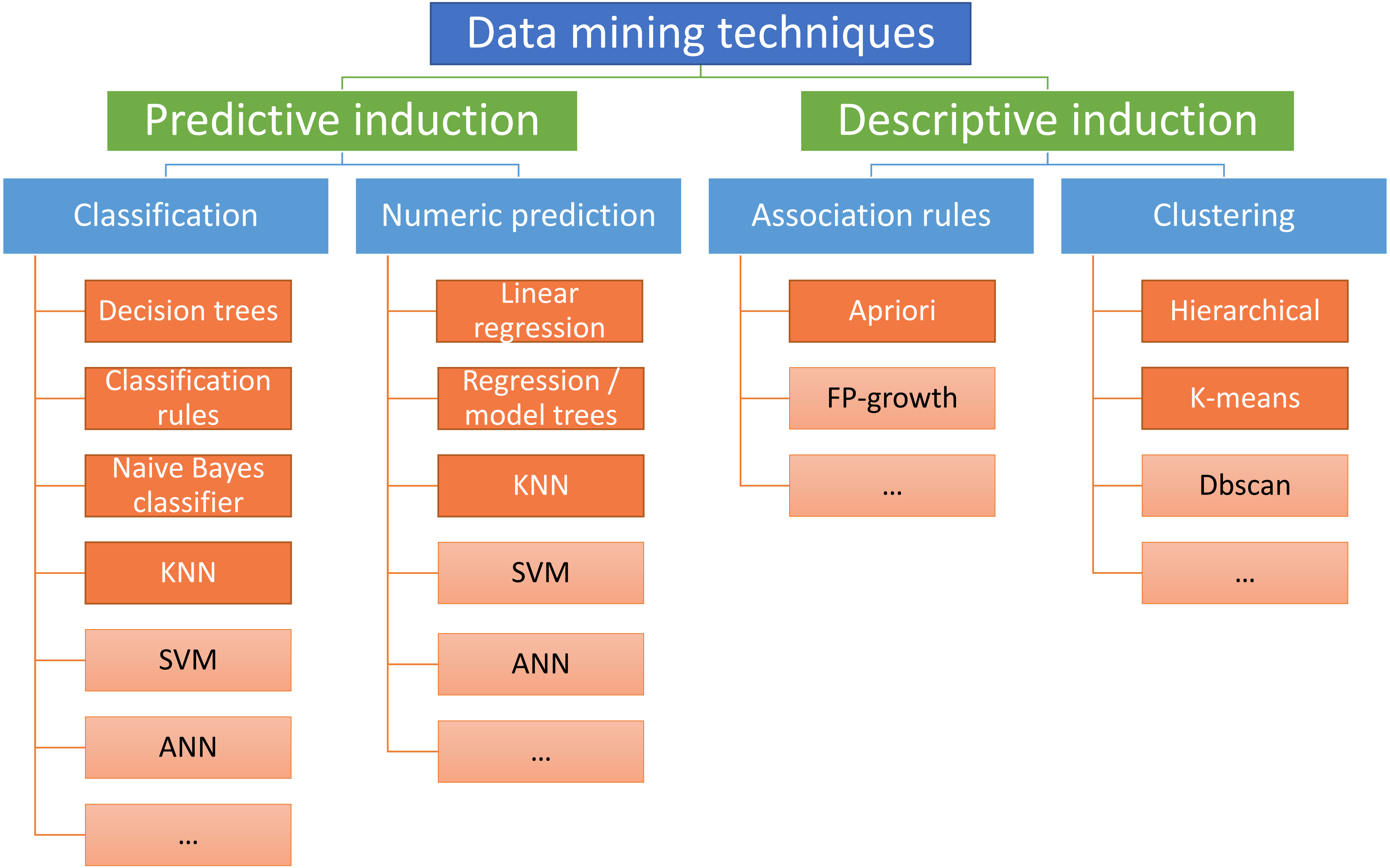
n(c) … number of times an event occurred
N … total number of events
k … number of possible outcomes

Number of events		**Relative frequency**		**Laplace estimate**	
Class C1	Class C2	P(C1)	P(C2)	P(C1)	P(C2)
0	2				
12	88				
12	988				
120	880				

Exercise

- Estimate the probabilities of C1 and C2 in the table below by relative frequency and Laplace estimate.

Number of events		**Relative frequency**		**Laplace estimate**	
Class C1	Class C2	P(C1)	P(C2)	P(C1)	P(C2)
0	2	0	1	0.25	0.75
12	88	0.12	0.88	0.127451	0.872549
12	988	0.012	0.988	0.012974	0.987026
120	880	0.12	0.88	0.120758	0.879242

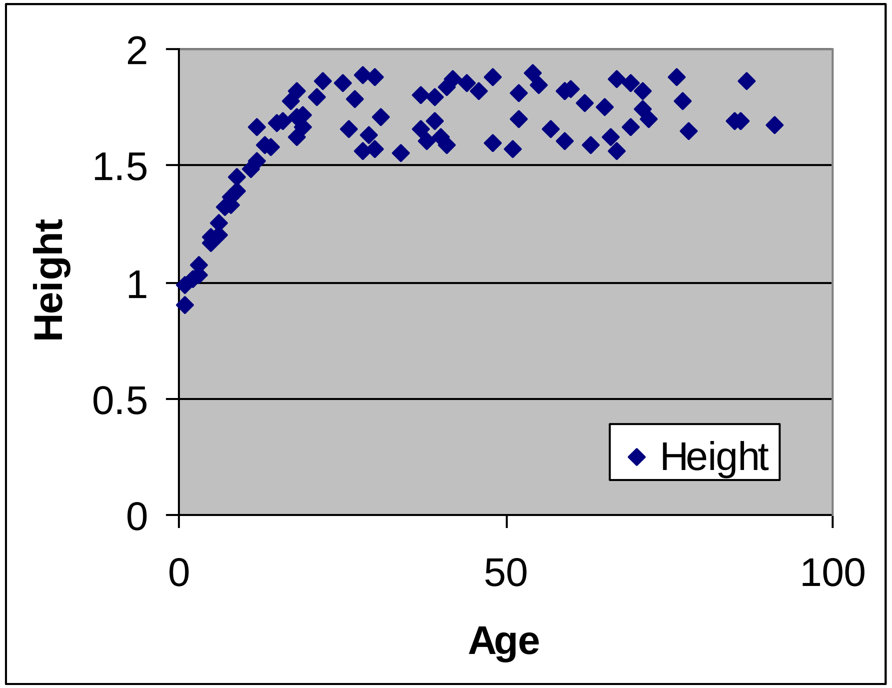

Data mining techniques
Predictive induction
Descriptive induction
Classification
Numeric prediction
Association rules
Clustering
Decision trees
Classification rules
Naive Bayes classifier
KNN
SVM
ANN
…
Linear regression
Regression / model trees
KNN
SVM
ANN
…
Apriori
FP-growth
…
Hierarchical
K-means
Dbscan
…

Numeric prediction

Regression

Example

- data about 80 people: Age and Height



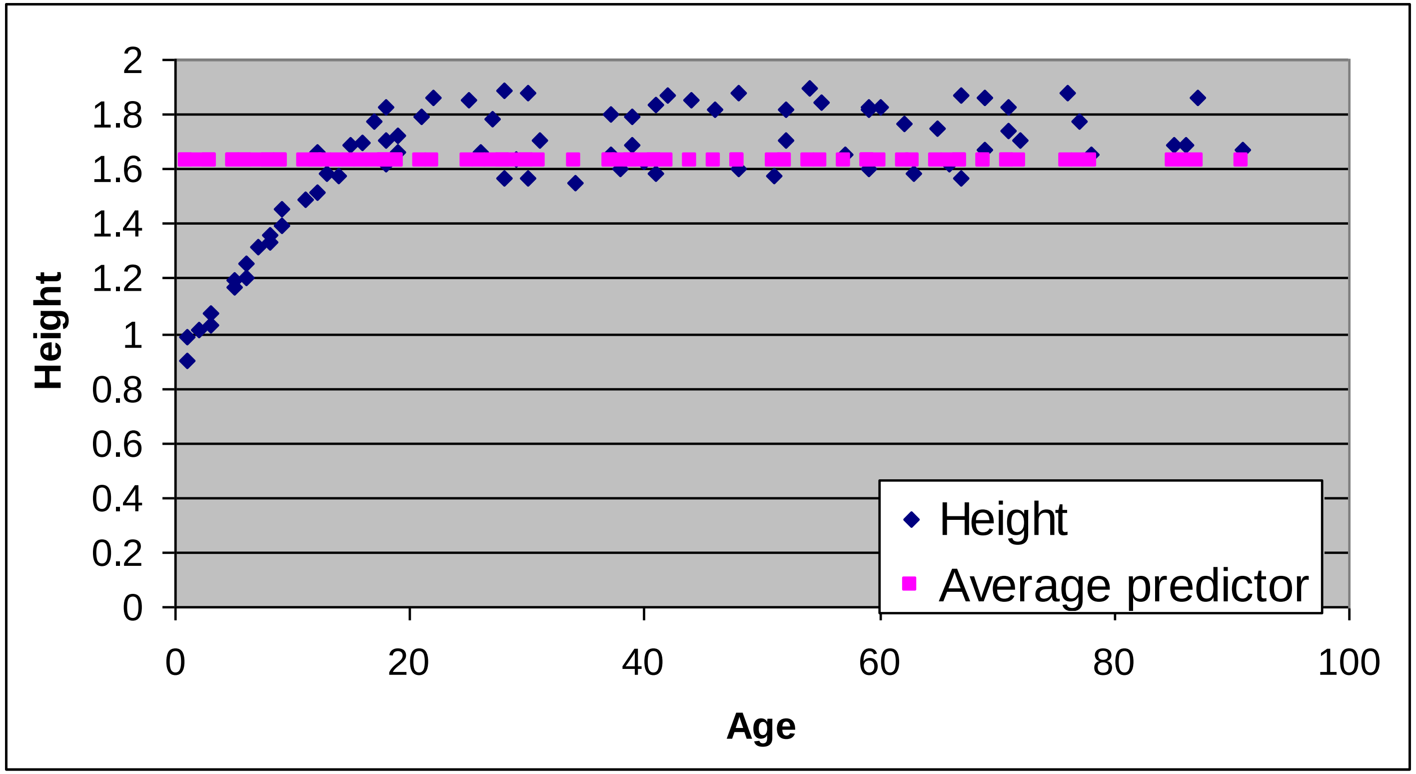
Age	Height
3	1.03
5	1.19
6	1.26
9	1.39
15	1.69
19	1.67
22	1.86
25	1.85
41	1.59
48	1.60
54	1.90
71	1.82
...	...

Test set

Age	Height
2	0.85
10	1.4
35	1.7
70	1.6

Baseline numeric predictor

- Average of the target variable



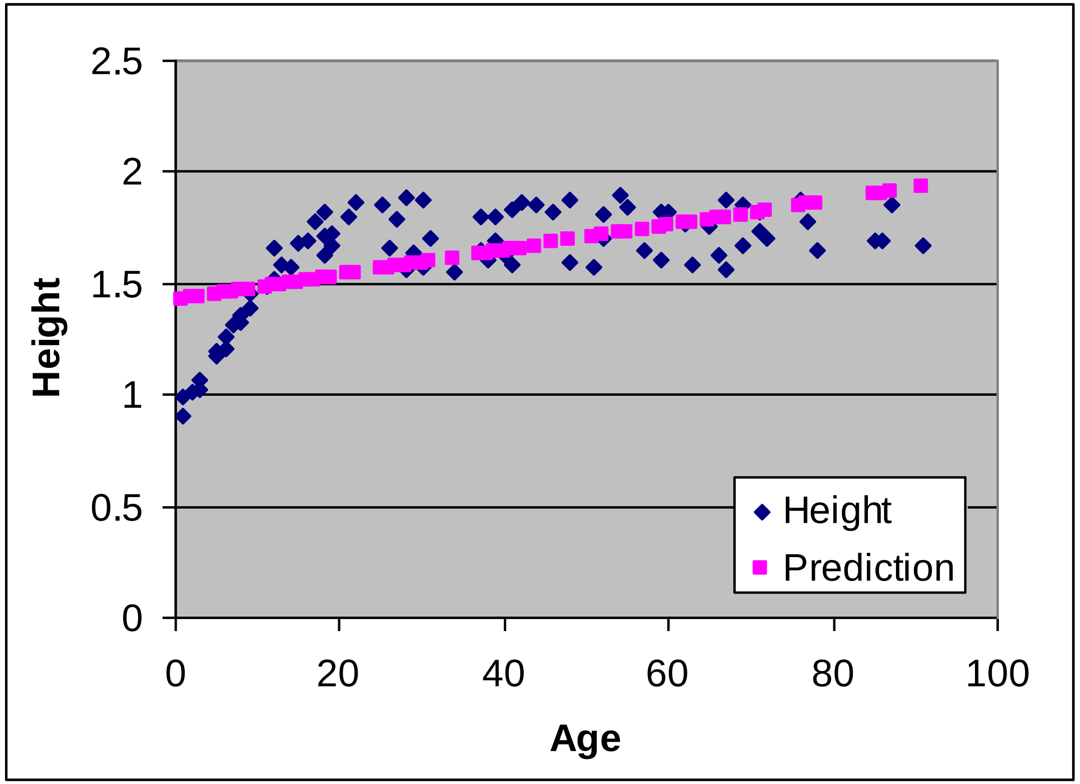
Baseline predictor: prediction

Average of the target variable is 1.63

Age	Height	Baseline
2	0.85	
10	1.4	
35	1.7	
70	1.6	

Linear Regression Model

Height = 0.0056 * Age + 1.4181

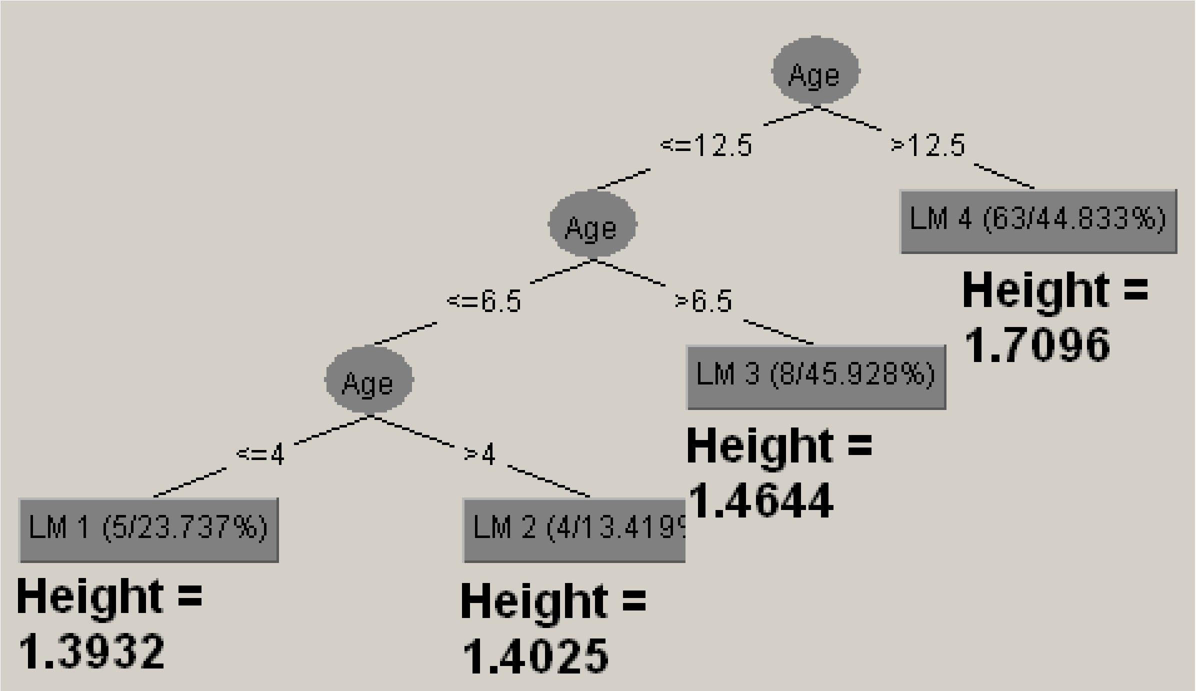


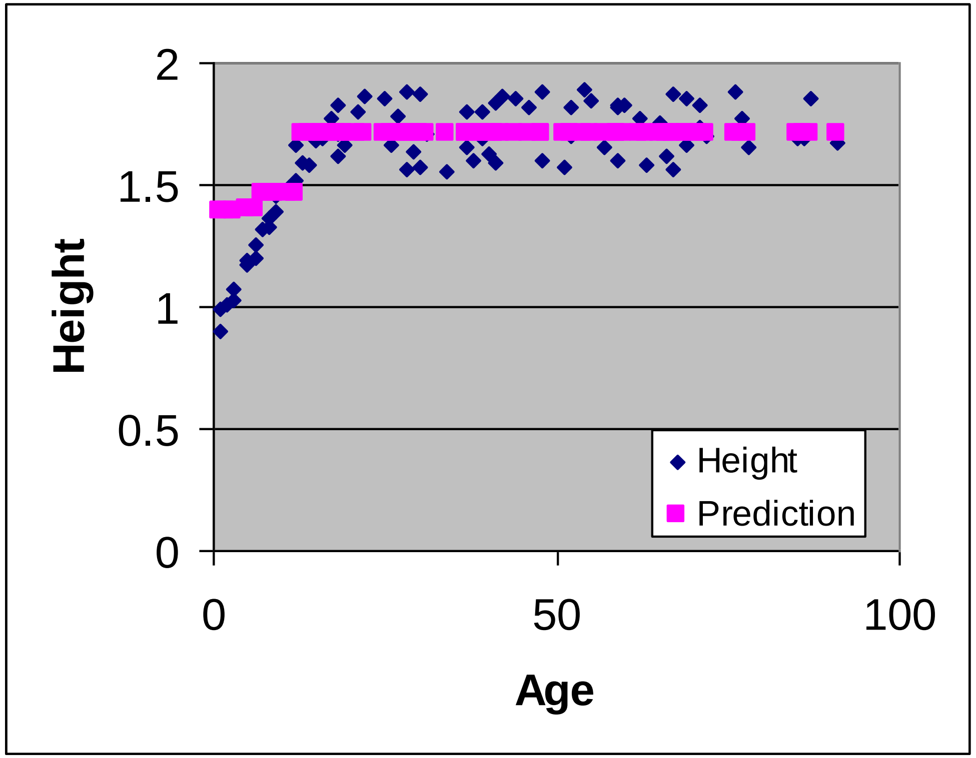
Linear Regression: prediction

Height = 0.0056 * Age + 1.4181

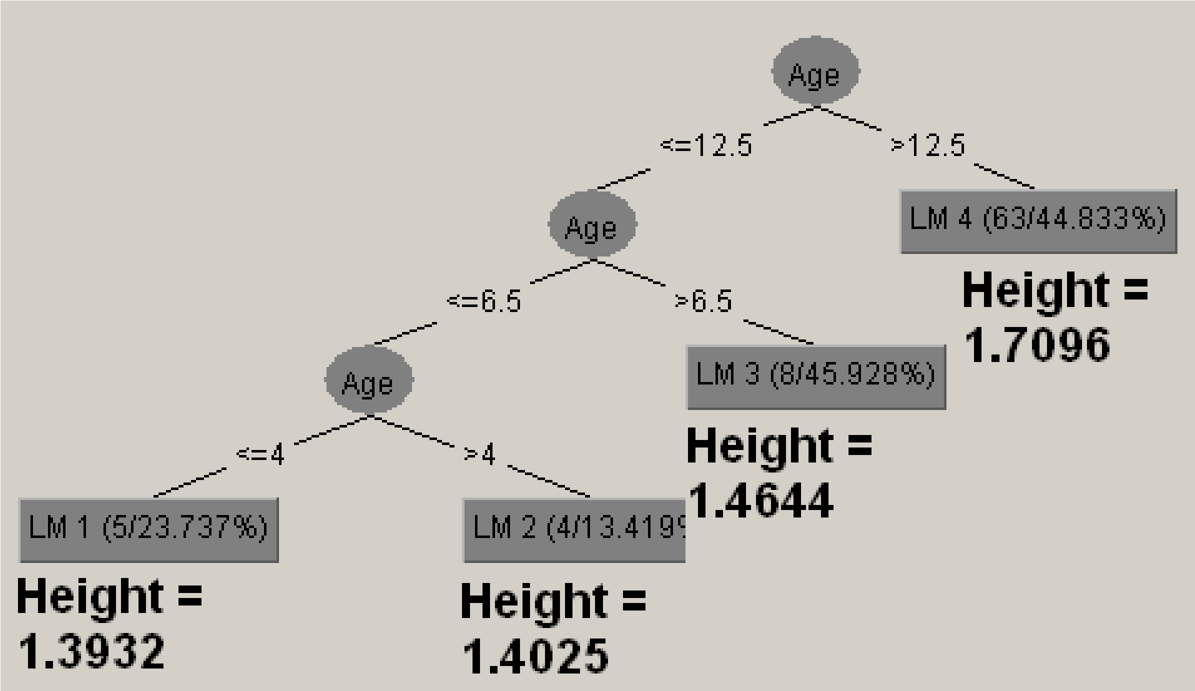
Age	Height	Linear regression
2	0.85	
10	1.4	
35	1.7	
70	1.6	

Regression tree



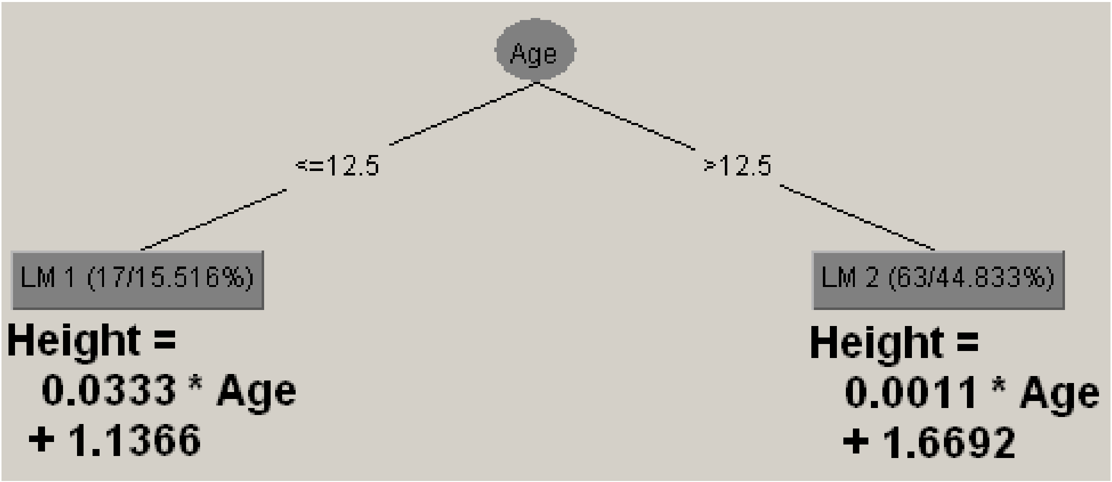


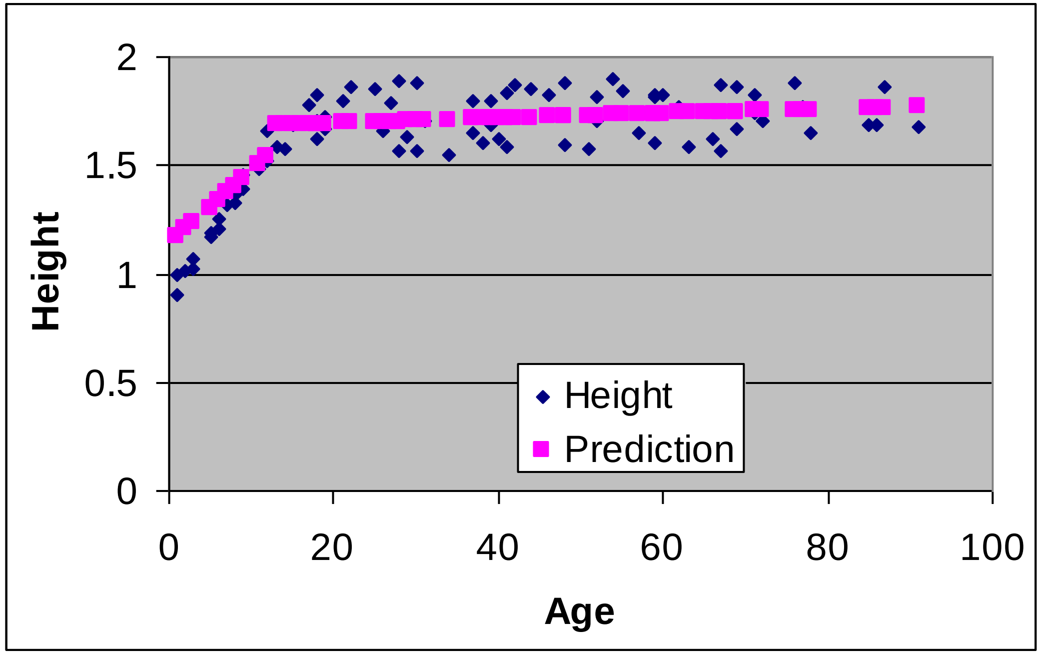
Regression tree: prediction



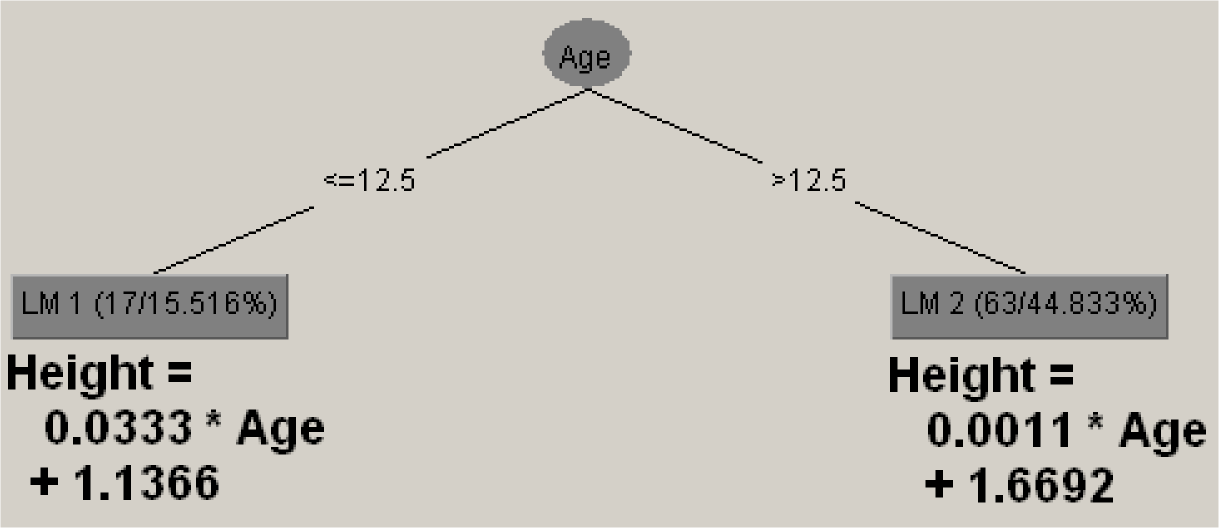
Age	Height	Regression tree
2	0.85	
10	1.4	
35	1.7	
70	1.6	

Model tree


Age
<=12.5
>12.5
LM 1 (17/15.516%)
LM 2 (63/44.833%)
Height = 0.0333 * Age + 1.1366
Height = 0.0011 * Age + 1.6692


Height
Age
2
1.5
1
0.5
0
0
20
40
60
80
100
Height
Prediction

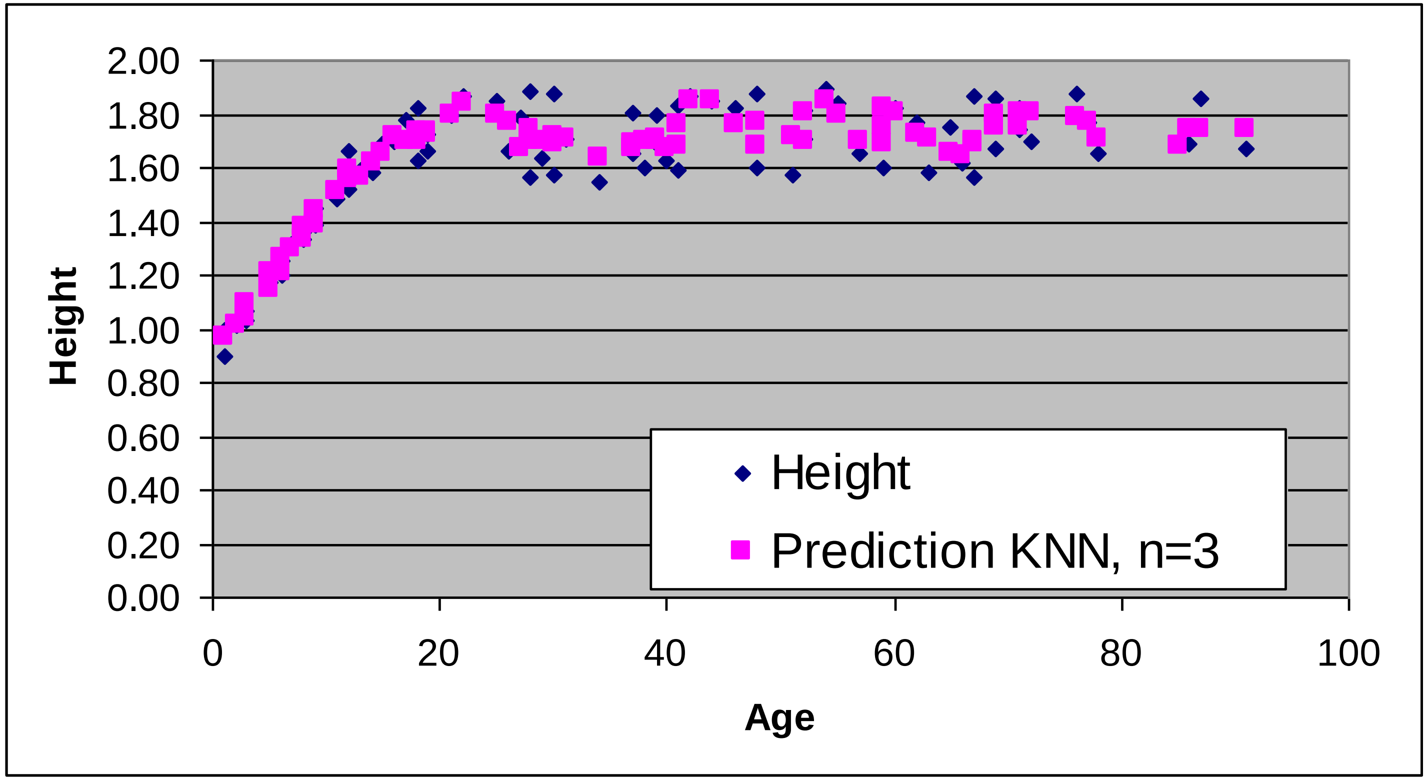
Model tree: prediction



Age	Height	Model tree
2	0.85	
10	1.4	
35	1.7	
70	1.6	

KNN – K nearest neighbors

- Looks at K closest examples (by non-target attributes) and predicts the average of their target variable
- In this example, K=3



KNN prediction

Age	Height
1	0.90
1	0.99
2	1.01
3	1.03
3	1.07
5	1.19
5	1.17

Age	Height	kNN
2	0.85	
10	1.4	
35	1.7	
70	1.6	

KNN prediction

Age	Height
8	1.36
8	1.33
9	1.45
9	1.39
11	1.49
12	1.66
12	1.52
13	1.59
14	1.58

Age	Height	kNN
2	0.85	
10	1.4	
35	1.7	
70	1.6	

KNN prediction

Age	Height
30	1.57
30	1.88
31	1.71
34	1.55
37	1.65
37	1.80
38	1.60
39	1.69
39	1.80

Age	Height	kNN
2	0.85	
10	1.4	
35	1.7	
70	1.6	

KNN prediction

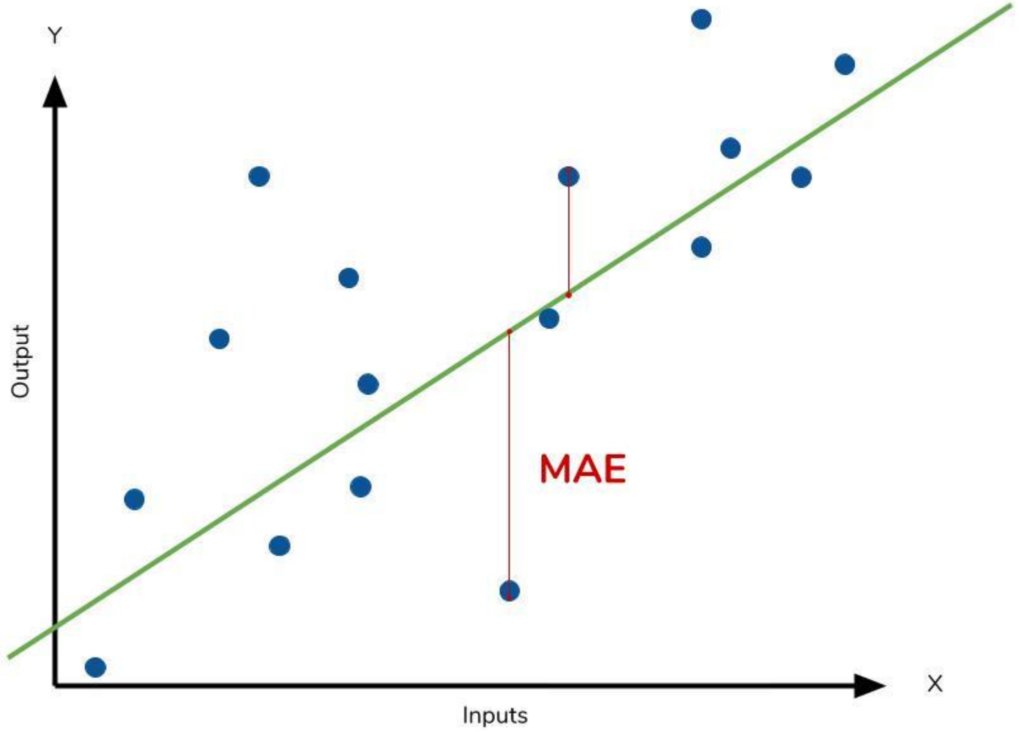
Age	Height
67	1.56
67	1.87
69	1.67
69	1.86
71	1.74
71	1.82
72	1.70
76	1.88

Age	Height	kNN
2	0.85	
10	1.4	
35	1.7	
70	1.6	

Which predictor is the best?

Age	Height	Baseline	Linear regression	Regression tree	Model tree	kNN
2	0.85	1.63	1.43	1.39	1.20	1.00
10	1.4	1.63	1.47	1.46	1.47	1.44
35	1.7	1.63	1.61	1.71	1.71	1.67
70	1.6	1.63	1.81	1.71	1.75	1.77

MAE: Mean absolute error



Divide by the total number of data points

Predicted output value

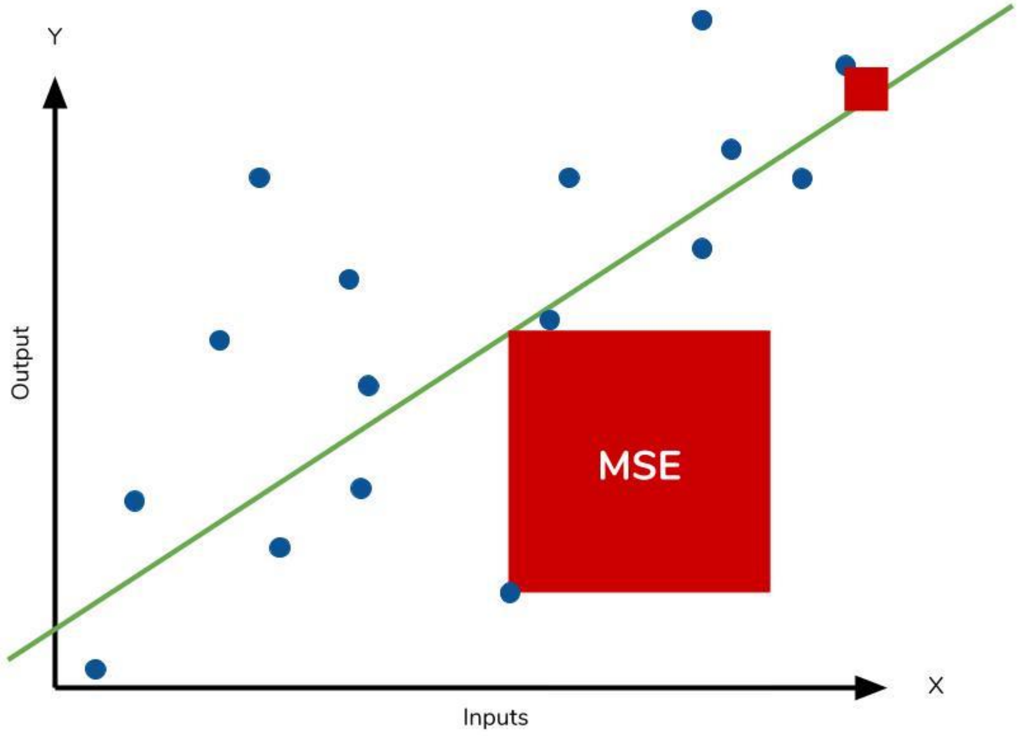
Actual output value

$$MAE = \boxed{\frac{1}{n}} \sum \underbrace{\left| \boxed{y} - \boxed{\hat{y}} \right|}_{\text{The absolute value of the residual}}$$

Sum of

The average difference between the predicted and the actual values.
The units are the same as the unites in the target variable.

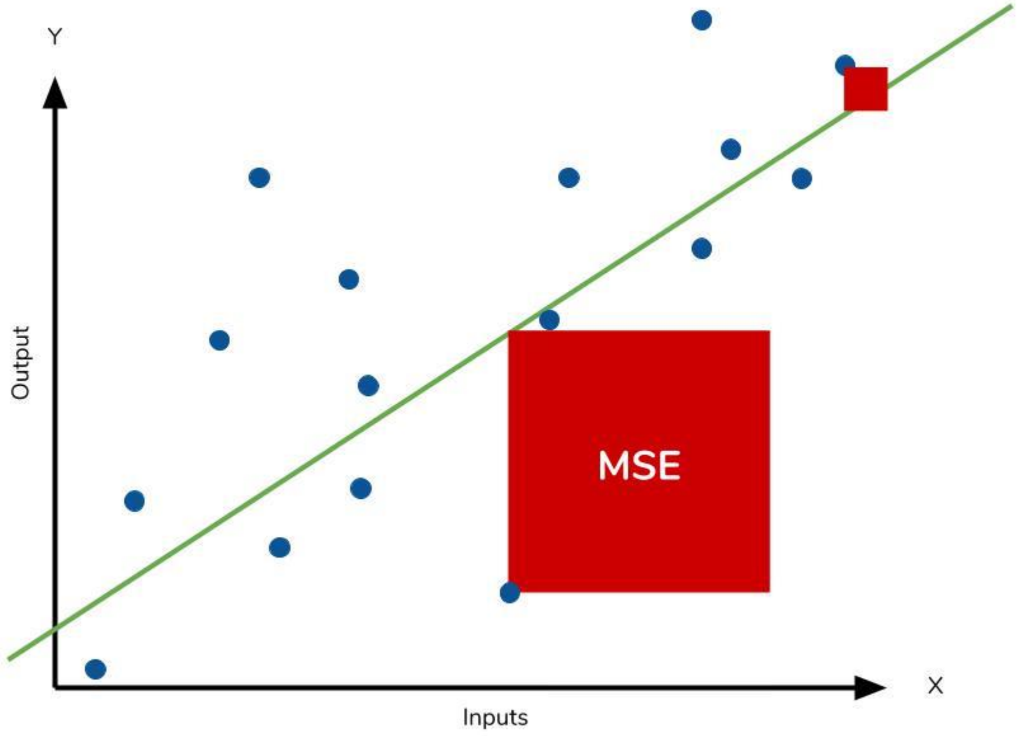
MSE: Mean squared error



$$MSE = \frac{1}{n} \Sigma \underbrace{\left(y - \widehat{y} \right)^2}_{\text{The square of the difference between actual and predicted}}$$

Mean squared error measures the average squared difference between the estimated values and the actual value.
Weights large errors more heavily than small ones.
The units of the errors are squared.

RMSE: Root mean square error

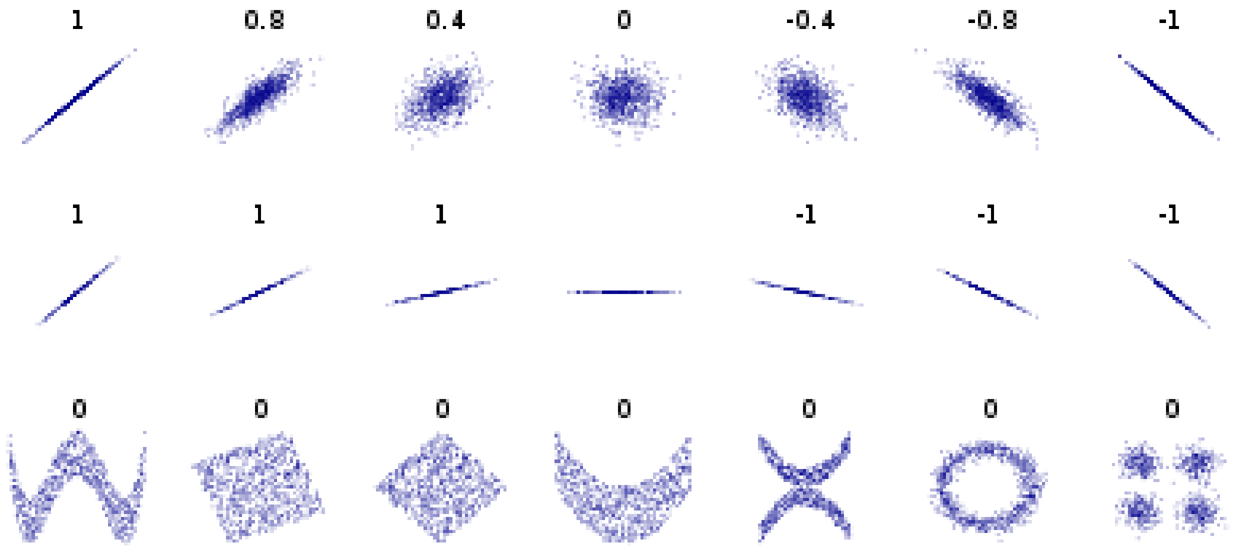


$$RMSE = \sqrt{MSE}$$

Taking the square root of MSE yields the root-mean-square error (RMSE), which has the same units as the quantity being estimated.

Correlation coefficient

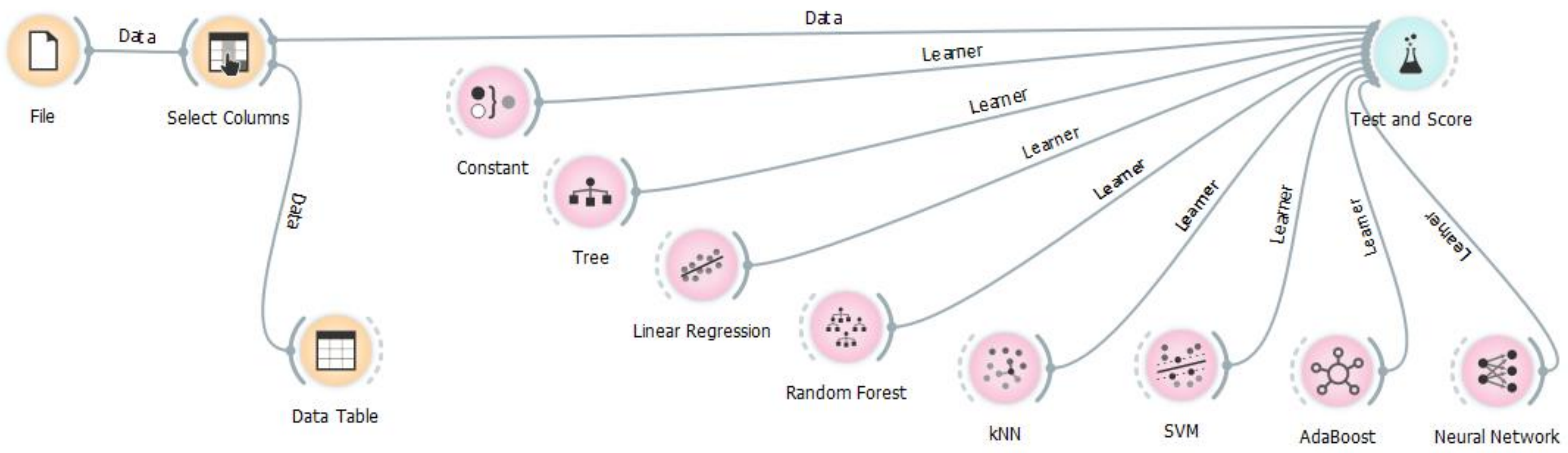
- Pearson correlation coefficient is a statistical formula that measures the strength between variables and relationships.



Similar to confusion matrix in the classification case.
No unit.

Numeric prediction in Orange

Models



Metrics

- MSE – mean squared error
- RMSE – root mean squared error
- MAE – mean absolute error
- R^2 – correlation coefficient

Evaluation Results

Model	MSE	RMSE	MAE	R2
Constant	0.055	0.236	0.175	-0.005
Linear Regression	0.033	0.181	0.142	0.405
SVM	0.032	0.179	0.128	0.423
Neural Network	0.026	0.161	0.118	0.533
kNN	0.011	0.107	0.086	0.794
Tree	0.010	0.100	0.073	0.817
AdaBoost	0.004	0.066	0.057	0.922
Random Forest	0.003	0.057	0.048	0.940

Numeric prediction	Classification
Data: attribute-value description	
Target variable: Continuous	**Target variable**: Categorical (nominal)
Evaluation: cross validation, separate test set, …	
Error: MSE, MAE, RMSE, …	**Error**: 1-accuracy
Algorithms: Linear regression, regression trees,…	**Algorithms**: Decision trees, Naïve Bayes, …
Baseline predictor: Mean of the target variable	**Baseline predictor**: Majority class

Performance measures for numeric prediction

Performance measure	Formula
mean-squared error	$\frac{(p_1-a_1)^2+\ldots+(p_n-a_n)^2}{n}$
root mean-squared error	$\sqrt{\frac{(p_1-a_1)^2+\ldots+(p_n-a_n)^2}{n}}$
mean absolute error	$\frac{\lvert p_1-a_1\rvert+\ldots+\lvert p_n-a_n\rvert}{n}$
relative squared error	$\frac{(p_1-a_1)^2+\ldots+(p_n-a_n)^2}{(a_1-\bar{a})^2+\ldots+(a_n-\bar{a})^2}$, where $\bar{a}=\frac{1}{n}\sum_i a_i$
root relative squared error	$\sqrt{\frac{(p_1-a_1)^2+\ldots+(p_n-a_n)^2}{(a_1-\bar{a})^2+\ldots+(a_n-\bar{a})^2}}$
relative absolute error	$\frac{\lvert p_1-a_1\rvert+\ldots+\lvert p_n-a_n\rvert}{\lvert a_1-\bar{a}\rvert+\ldots+\lvert a_n-\bar{a}\rvert}$
correlation coefficient	$\frac{S_{PA}}{\sqrt{S_P S_A}}$, where $S_{PA}=\frac{\sum_i(p_i-\bar{p})(a_i-\bar{a})}{n-1}$, $S_p=\frac{\sum_i(p_i-\bar{p})^2}{n-1}$, and $S_A=\frac{\sum_i(a_i-\bar{a})^2}{n-1}$

* p are predicted values and a are actual values.

Witten, Ian H., Eibe Frank, and Mark A. Hall. "Practical machine learning tools and techniques." *Morgan Kaufmann* (2005): 578. pg. 178

Relative squared error

"The error is made relative to what it would have been if a simple predictor had been used. The simple predictor in question is just the average of the actual values from the **training** data. Thus relative squared error takes the total squared error and normalizes it by dividing by the total squared error of the default predictor."

Witten, Ian H., Eibe Frank, and Mark A. Hall. "Practical machine learning tools and techniques." *Morgan Kaufmann* (2005): 578. pg. 177

Exercise: RRSE

- Use SciKit (or Orange) to compute RRSE of regression models
- RRSE = root relative squared error
  - Nominator: sum of squared differences between the actual and the expected values
  - Denominator: sum of squared errors (the sum of the squared differences between each observation and its group's mean)

$$RRSE = \sqrt{\frac{\sum_{i=1}^{n}(p_i - a_i)^2}{\sum_{i=1}^{n}(\bar{a} - a_i)^2}}$$

p – predicted, a – actual, ā – the mean of the actual

  - RRSE: Ratio between the error of the model and the error of the naïve model (predicting the average)

Regression in scikit … 4_regression.py

```python
import pandas as pd
from sklearn import dummy
from sklearn import linear_model
from sklearn import tree
from sklearn.neighbors import KNeighborsRegressor
from sklearn.model_selection import train_test_split
from sklearn import metrics

print("_______________________________________________________________________________")
print("Regression models, train-test validation on regressionAgeHeight.csv. ")
print("_______________________________________________________________________________")

print(""" Load the data """)
csvFileName = r"./Datasets/regressionAgeHeight.csv"
df = pd.read_csv(csvFileName)
print(df.head())
print("data shape: ", df.shape)

feature_cols = ['Age']
target_var = 'Height'

X = df[feature_cols].values
y = df[target_var].values

""" Train-test split """
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=42)
```

Regression in scikit … 4_regression.py

```
""" Initialize the learners """
dummy = dummy.DummyRegressor()
regr = linear_model.LinearRegression()
reg_tree = tree.DecisionTreeRegressor(min_samples_leaf=8)
knn = KNeighborsRegressor(n_neighbors=2)

learner = reg_tree

"""" Train and apply """
learner.fit(X_train, y_train)
y_pred = learner.predict(X_test)

print ("\n Actual   Predicted")
for i in range(len(y_test)):
     print("{0:6.2f}  {1:8.2f}".format(y_test[i], y_pred[i]))

print("Performance:")
print("MAE  \t{0:5.2f}".format( metrics.mean_absolute_error(y_test,y_pred)))
print("MSE  \t{0:5.2f}".format( metrics.mean_squared_error(y_test,y_pred)))
print("R2   \t{0:5.2f}".format( metrics.r2_score(y_test,y_pred)))
```