# Course Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Subject/Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2018/09/10</td>
<td>Course Orientation for Big Data Mining</td>
</tr>
<tr>
<td>2</td>
<td>2018/09/17</td>
<td>ABC: AI, Big Data, Cloud Computing</td>
</tr>
<tr>
<td>3</td>
<td>2018/09/24</td>
<td>Mid-Autumn Festival (Day off)</td>
</tr>
<tr>
<td>4</td>
<td>2018/10/01</td>
<td>Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data</td>
</tr>
<tr>
<td>5</td>
<td>2018/10/08</td>
<td>Fundamental Big Data: MapReduce Paradigm, Hadoop and Spark Ecosystem</td>
</tr>
<tr>
<td>6</td>
<td>2018/10/15</td>
<td>Foundations of Big Data Mining in Python</td>
</tr>
<tr>
<td>7</td>
<td>2018/10/22</td>
<td>Supervised Learning: Classification and Prediction</td>
</tr>
<tr>
<td>8</td>
<td>2018/10/29</td>
<td>Unsupervised Learning: Cluster Analysis</td>
</tr>
<tr>
<td>9</td>
<td>2018/11/05</td>
<td>Unsupervised Learning: Association Analysis</td>
</tr>
</tbody>
</table>
## Course Schedule (2/2)

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Subject/Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2018/11/12</td>
<td>Midterm Project Report</td>
</tr>
<tr>
<td>11</td>
<td>2018/11/19</td>
<td>Machine Learning with Scikit-Learn in Python</td>
</tr>
<tr>
<td>12</td>
<td>2018/11/26</td>
<td>Deep Learning for Finance Big Data with TensorFlow</td>
</tr>
<tr>
<td>13</td>
<td>2018/12/03</td>
<td>Convolutional Neural Networks (CNN)</td>
</tr>
<tr>
<td>14</td>
<td>2018/12/10</td>
<td>Recurrent Neural Networks (RNN)</td>
</tr>
<tr>
<td>15</td>
<td>2018/12/17</td>
<td>Reinforcement Learning (RL)</td>
</tr>
<tr>
<td>16</td>
<td>2018/12/24</td>
<td>Social Network Analysis (SNA)</td>
</tr>
<tr>
<td>17</td>
<td>2018/12/31</td>
<td>Bridge Holiday (Extra Day Off)</td>
</tr>
<tr>
<td>18</td>
<td>2019/01/07</td>
<td>Final Project Presentation</td>
</tr>
</tbody>
</table>
Supervised Learning: Classification and Prediction
Outline

• Supervised Learning
• Classification and Prediction
• Decision Tree (DT)
  – Information Gain (IG)
• Support Vector Machine (SVM)
• Data Mining Evaluation
  – Accuracy
  – Precision
  – Recall
  – F1 score (F-measure) (F-score)
Data Mining Tasks and Machine Learning

Supervised Learning: Classification and Prediction

Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson
Scikit-Learn Machine Learning Map

classification
- SVC
- Ensemble Classifiers
- Naive Bayes Classifier
- Text Data
- Linear SVC
- KNNeighbours Classifier
- SGD Classifier
- kernel approximation

<100K samples

<10K categories known
- Spectral Clustering
- GMM
- KMeans
- MiniBatch KMeans
- MeanShift
- VBGMM

>50 samples

number of categories known
- just looking
- tough luck

clustering

<10K samples

<10K samples

predicting a category
- get more data

>50 samples

predicting a quantity
- few features should be important

dimensionality reduction
- Randomized PCA
- Isomap
- Spectral Embedding
- LLE

scikit-learn algorithm cheat-sheet
- Lasso
- ElasticNet
- SVR(kernel='rbf')
- EnsembleRegressors
- RidgeRegression
- SVR(kernel='linear')

Scikit-Learn Machine Learning Map

Scikit-Learn

scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.

Algorithms: SVM, nearest neighbors, random forest, ...

— Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: SVR, ridge regression, Lasso, ...

— Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift, ...

— Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: PCA, feature selection, non-negative matrix factorization.

— Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: grid search, cross validation, metrics.

— Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms

Modules: preprocessing, feature extraction.

— Examples

Source: http://scikit-learn.org/
Classification vs. Prediction

- **Classification**
  - predicts *categorical class labels* (discrete or nominal)
  - classifies data (constructs a model) based on the training set and the values (*class labels*) in a classifying attribute and uses it in classifying new data

- **Prediction**
  - models *continuous-valued* functions
    - i.e., predicts unknown or missing values

- **Typical applications**
  - Credit approval
  - Target marketing
  - Medical diagnosis
  - Fraud detection

Source: Han & Kamber (2006)
Data Mining Methods: Classification

- Most frequently used DM method
- Part of the machine-learning family
- Employ supervised learning
- Learn from past data, classify new data
- The output variable is categorical (nominal or ordinal) in nature
- Classification versus regression?
- Classification versus clustering?

Source: Turban et al. (2011), Decision Support and Business Intelligence Systems
Classification Techniques

• Decision Tree analysis (DT)
• Statistical analysis
• Neural networks (NN)
• Deep Learning (DL)
• Support Vector Machines (SVM)
• Case-based reasoning
• Bayesian classifiers
• Genetic algorithms (GA)
• Rough sets

Source: Turban et al. (2011), Decision Support and Business Intelligence Systems
Text Mining
(Text Data Mining)
Example of Opinion:
review segment on iPhone

“I bought an iPhone a few days ago.
It was such a nice phone.
The touch screen was really cool.
The voice quality was clear too.
However, my mother was mad with me as I did not tell her before I bought it.
She also thought the phone was too expensive, and wanted me to return it to the shop. ... ”

“(1) I bought an iPhone a few days ago.

(2) It was such a nice phone.

(3) The touch screen was really cool.

(4) The voice quality was clear too.

(5) However, my mother was mad with me as I did not tell her before I bought it.

(6) She also thought the phone was too expensive, and wanted me to return it to the shop. ...”
Text mining

Text Data Mining

Intelligent Text Analysis

Knowledge-Discovery in Text (KDT)

Text Mining: the process of extracting interesting and non-trivial information and knowledge from unstructured text.

Text Mining: discovery by computer of new, previously unknown information, by automatically extracting information from different written resources.

Text Mining (TM)

Natural Language Processing (NLP)
An Example of Text Mining

Analyze Text
- Information Extraction
- Classification
- Summarization
- Clustering

Retrieve and preprocess document

Document Collection

Knowledge

Management Information System

Overview of Information Extraction based Text Mining Framework

Text Data Mining

Text → Information Extraction → DB → Data Mining → Rule

Data Mining versus Text Mining

• Both seek for novel and useful patterns
• Both are semi-automated processes
• Difference is the nature of the data:
  – Structured versus unstructured data
  – Structured data: in databases
  – Unstructured data: Word documents, PDF files, text excerpts, XML files, and so on
• Text mining – first, impose structure to the data, then mine the structured data

Source: Turban et al. (2011), Decision Support and Business Intelligence Systems
Sentiment Classification Based on Emoticons

Lexicon-Based Model

Preassembled Word Lists

Merged Lexicon

Generic Word Lists

Tokenized Document Collection

Sentiment Scoring and Classification: Polarity

Sentiment Polarity

Sentiment Analysis Tasks

Opinionated Document → Subjectivity Classification → Sentiment Classification → Opinion holder extraction → Object/Feature extraction → Subjectivity Classification

### Sentiment Analysis vs. Subjectivity Analysis

<table>
<thead>
<tr>
<th>Sentiment Analysis</th>
<th>Subjectivity Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>Subjective</td>
</tr>
<tr>
<td>Negative</td>
<td>Objective</td>
</tr>
<tr>
<td>Neutral</td>
<td></td>
</tr>
</tbody>
</table>
Levels of Sentiment Analysis

- **Word level Sentiment Analysis**
- **Sentence level Sentiment Analysis**
- **Document level Sentiment Analysis**
- **Feature level Sentiment Analysis**

---

Sentiment Analysis

Tasks

- Subjectivity Classification
- Sentiment Classification
- Review Usefulness Measurement
- Opinion Spam Detection
- Lexicon Creation
- Aspect Extraction
- Application

Approaches

- Machine Learning based
- Lexicon based
- Hybrid approaches
- Ontology based
- Non-Ontology based

Sentiment Classification Techniques

Machine Learning Models

- Deep Learning
- Association rules
- Decision tree
- Clustering
- Bayesian

- Kernel
- Ensemble
- Dimensionality reduction
- Regression Analysis
- Instance based

Source: Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
Example of Classification

• Loan Application Data
  – Which loan applicants are “safe” and which are “risky” for the bank?
  – “Safe” or “risky” for load application data

• Marketing Data
  – Whether a customer with a given profile will buy a new computer?
  – “yes” or “no” for marketing data

• Classification
  – Data analysis task
  – A model or Classifier is constructed to predict categorical labels
    • Labels: “safe” or “risky”; “yes” or “no”; “treatment A”, “treatment B”, “treatment C”
What Is Prediction?

- (Numerical) prediction is similar to classification
  - construct a model
  - use model to predict continuous or ordered value for a given input

- Prediction is different from classification
  - Classification refers to predict categorical class label
  - Prediction models continuous-valued functions

- Major method for prediction: regression
  - model the relationship between one or more independent or predictor variables and a dependent or response variable

- Regression analysis
  - Linear and multiple regression
  - Non-linear regression
  - Other regression methods: generalized linear model, Poisson regression, log-linear models, regression trees

Source: Han & Kamber (2006)
Prediction Methods

• Linear Regression
• Nonlinear Regression
• Other Regression Methods

Salary data.

<table>
<thead>
<tr>
<th>x</th>
<th>y salary (in $1000s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>8</td>
<td>57</td>
</tr>
<tr>
<td>9</td>
<td>64</td>
</tr>
<tr>
<td>13</td>
<td>72</td>
</tr>
<tr>
<td>3</td>
<td>36</td>
</tr>
<tr>
<td>6</td>
<td>43</td>
</tr>
<tr>
<td>11</td>
<td>59</td>
</tr>
<tr>
<td>21</td>
<td>90</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>16</td>
<td>83</td>
</tr>
</tbody>
</table>

Source: Han & Kamber (2006)
Classification and Prediction

- **Classification** and **prediction** are two forms of data analysis that can be used to extract **models** describing important data classes or to predict future data trends.
- **Classification**
  - Effective and scalable methods have been developed for decision trees induction, Naive Bayesian classification, Bayesian belief network, rule-based classifier, Backpropagation, Support Vector Machine (SVM), associative classification, nearest neighbor classifiers, and case-based reasoning, and other classification methods such as genetic algorithms, rough set and fuzzy set approaches.
- **Prediction**
  - Linear, nonlinear, and generalized linear models of regression can be used for prediction. Many nonlinear problems can be converted to linear problems by performing transformations on the predictor variables. Regression trees and model trees are also used for prediction.

Source: Han & Kamber (2006)
Classification
—A Two-Step Process

1. Model construction: describing a set of predetermined classes
   – Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
   – The set of tuples used for model construction is training set
   – The model is represented as classification rules, decision trees, or mathematical formulae

2. Model usage: for classifying future or unknown objects
   – Estimate accuracy of the model
     • The known label of test sample is compared with the classified result from the model
     • Accuracy rate is the percentage of test set samples that are correctly classified by the model
     • Test set is independent of training set, otherwise over-fitting will occur
   – If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known

Source: Han & Kamber (2006)
Supervised Learning vs. Unsupervised Learning

• **Supervised learning (classification)**
  
  – Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  
  – New data is classified based on the training set

• **Unsupervised learning (clustering)**
  
  – The class labels of training data is unknown
  
  – Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

Source: Han & Kamber (2006)
Issues Regarding Classification and Prediction: Data Preparation

• Data cleaning
  – Preprocess data in order to reduce noise and handle missing values
• Relevance analysis (feature selection)
  – Remove the irrelevant or redundant attributes
  – Attribute subset selection
    • Feature Selection in machine learning
• Data transformation
  – Generalize and/or normalize data
  – Example
    • Income: low, medium, high

Source: Han & Kamber (2006)
Issues:
Evaluating Classification and Prediction Methods

• Accuracy
  – classifier accuracy: predicting class label
  – predictor accuracy: guessing value of predicted attributes
  – estimation techniques: cross-validation and bootstrapping
• Speed
  – time to construct the model (training time)
  – time to use the model (classification/prediction time)
• Robustness
  – handling noise and missing values
• Scalability
  – ability to construct the classifier or predictor efficiently given large amounts of data
• Interpretability
  – understanding and insight provided by the model

Source: Han & Kamber (2006)
Data Classification Process 1: **Learning (Training) Step**

(a) **Learning**: **Training data** are analyzed by classification algorithm

\[ y = f(X) \]

---

### Training data

<table>
<thead>
<tr>
<th>name</th>
<th>age</th>
<th>income</th>
<th>loan_decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sandy Jones</td>
<td>young</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>Bill Lee</td>
<td>young</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>Caroline Fox</td>
<td>middle-aged</td>
<td>high</td>
<td>safe</td>
</tr>
<tr>
<td>Rick Field</td>
<td>middle-aged</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>Susan Lake</td>
<td>senior</td>
<td>low</td>
<td>safe</td>
</tr>
<tr>
<td>Claire Phips</td>
<td>senior</td>
<td>medium</td>
<td>safe</td>
</tr>
<tr>
<td>Joe Smith</td>
<td>middle-aged</td>
<td>high</td>
<td>safe</td>
</tr>
</tbody>
</table>

Classification rules:

- IF age = youth THEN loan_decision = risky
- IF income = high THEN loan_decision = safe
- IF age = middle-aged AND income = low THEN loan_decision = risky

---

Source: Han & Kamber (2006)
Data Classification Process 2

(b) **Classification**: Test data are used to estimate the accuracy of the classification rules.
Process (1): Model Construction

### Training Data

<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike</td>
<td>Assistant Prof</td>
<td>3</td>
<td>no</td>
</tr>
<tr>
<td>Mary</td>
<td>Assistant Prof</td>
<td>7</td>
<td>yes</td>
</tr>
<tr>
<td>Bill</td>
<td>Professor</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>Jim</td>
<td>Associate Prof</td>
<td>7</td>
<td>yes</td>
</tr>
<tr>
<td>Dave</td>
<td>Assistant Prof</td>
<td>6</td>
<td>no</td>
</tr>
<tr>
<td>Anne</td>
<td>Associate Prof</td>
<td>3</td>
<td>no</td>
</tr>
</tbody>
</table>

### Classification Algorithms

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’

Source: Han & Kamber (2006)
Process (2): Using the Model in Prediction

Unseen Data

[(Jeff, Professor, 4)]

Tenured?

Yes

<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tom</td>
<td>Assistant Prof</td>
<td>2</td>
<td>no</td>
</tr>
<tr>
<td>Merlisa</td>
<td>Associate Prof</td>
<td>7</td>
<td>no</td>
</tr>
<tr>
<td>George</td>
<td>Professor</td>
<td>5</td>
<td>yes</td>
</tr>
<tr>
<td>Joseph</td>
<td>Assistant Prof</td>
<td>7</td>
<td>yes</td>
</tr>
</tbody>
</table>

Source: Han & Kamber (2006)
Decision Trees
Decision Trees

A general algorithm for decision tree building

- Employs the divide and conquer method
- Recursively divides a training set until each division consists of examples from one class
  1. Create a root node and assign all of the training data to it
  2. Select the best splitting attribute
  3. Add a branch to the root node for each value of the split. Split the data into mutually exclusive subsets along the lines of the specific split
  4. Repeat the steps 2 and 3 for each and every leaf node until the stopping criteria is reached

Source: Turban et al. (2011), Decision Support and Business Intelligence Systems
Decision Trees

• DT algorithms mainly differ on
  – Splitting criteria
    • Which variable to split first?
    • What values to use to split?
    • How many splits to form for each node?
  – Stopping criteria
    • When to stop building the tree
  – Pruning (generalization method)
    • Pre-pruning versus post-pruning

• Most popular DT algorithms include
  – ID3, C4.5, C5; CART; CHAID; M5

Source: Turban et al. (2011), Decision Support and Business Intelligence Systems
Decision Trees

• Alternative splitting criteria
  – Gini index determines the purity of a specific class as a result of a decision to branch along a particular attribute/value
    • Used in CART
  – Information gain uses entropy to measure the extent of uncertainty or randomness of a particular attribute/value split
    • Used in ID3, C4.5, C5
  – Chi-square statistics (used in CHAID)

Source: Turban et al. (2011), Decision Support and Business Intelligence Systems
## Classification by Decision Tree Induction

### Training Dataset

<table>
<thead>
<tr>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;=30</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>31…40</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>31…40</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>medium</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>31…40</td>
<td>medium</td>
<td>no</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>31…40</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>medium</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
</tbody>
</table>

This follows an example of Quinlan’s ID3 (Playing Tennis)

Source: Han & Kamber (2006)
Classification by Decision Tree Induction

Output: A Decision Tree for “buys_computer”

- **age?**
  - youth <=30
  - middle_aged 31..40
  - senior >40
    - credit rating?
      - fair
      - excellent
        - yes
        - no
    - yes
    - no

*buys_computer* = “yes” or *buys_computer* = “no”

Source: Han & Kamber (2006)
Three possibilities for partitioning tuples based on the splitting Criterion

Source: Han & Kamber (2006)
Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Source: Han & Kamber (2006)
Attribute Selection Measure

• Notation: Let $D$, the data partition, be a training set of class-labeled tuples.
Suppose the class label attribute has $m$ distinct values defining $m$ distinct classes, $C_i$ (for $i = 1, \ldots, m$).
Let $C_{i,D}$ be the set of tuples of class $C_i$ in $D$.
Let $|D|$ and $|C_{i,D}|$ denote the number of tuples in $D$ and $C_{i,D}$, respectively.

• Example:
  – Class: buys_computer = “yes” or “no”
  – Two distinct classes ($m=2$)
    • Class $C_i$ ($i=1,2$):
      $C_1$ = “yes”,
      $C_2$ = “no”

Source: Han & Kamber (2006)
Attribute Selection Measure: 
Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let $p_i$ be the probability that an arbitrary tuple in $D$ belongs to class $C_i$, estimated by $|C_{i, D}|/|D|
- **Expected information** (entropy) needed to classify a tuple in $D$:
  $$\text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$
- **Information** needed (after using $A$ to split $D$ into $v$ partitions) to classify $D$:
  $$\text{Info}_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)$$
- **Information gained** by branching on attribute $A$
  $$\text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D)$$

Source: Han & Kamber (2006)
\[
\begin{align*}
\log_2 (1) &= 0 \\
\log_2 (2) &= 1 \\
\log_2 (3) &= 1.5850 \\
\log_2 (4) &= 2 \\
\log_2 (5) &= 2.3219 \\
\log_2 (6) &= 2.5850 \\
\log_2 (7) &= 2.8074 \\
\log_2 (8) &= 3 \\
\log_2 (9) &= 3.1699 \\
\log_2 (10) &= 3.3219 \\
\end{align*}
\]
The attribute age has the highest information gain and therefore becomes the splitting attribute at the root node of the decision tree.
Attribute Selection: Information Gain

- Class P: buys_computer = “yes”
- Class N: buys_computer = “no”

\[
Info(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940
\]

<table>
<thead>
<tr>
<th>age</th>
<th>(p_i)</th>
<th>(n_i)</th>
<th>(I(p_i, n_i))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\leq 30)</td>
<td>2</td>
<td>3</td>
<td>0.971</td>
</tr>
<tr>
<td>31…40</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(&gt; 40)</td>
<td>3</td>
<td>2</td>
<td>0.971</td>
</tr>
</tbody>
</table>

\[
I(2,3) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0)
\]

\[
I(2,3) = \frac{5}{14} \cdot \left(\log_2\left(\frac{5}{14}\right) + \log_2\left(\frac{9}{14}\right)\right) = 0.971
\]

\[
5 \frac{5}{14} I(2,3) \quad \text{means “age \(\leq 30\)” has 5 out of 14 samples, with 2 yes’es and 3 no’s. Hence}
\]

\[
Gain(\text{age}) = Info(D) - Info_{\text{age}}(D) = 0.246
\]

Similarly,

\[
Gain(\text{income}) = 0.029
\]

\[
Gain(\text{student}) = 0.151
\]

\[
Gain(\text{credit_rating}) = 0.048
\]
Decision Tree
Information Gain
# Customer database

<table>
<thead>
<tr>
<th>ID</th>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>middle_aged</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>senior</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>senior</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>middle_aged</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>youth</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>youth</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>senior</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>
What is the class (\texttt{buys\_computer} = “yes” or \texttt{buys\_computer} = “no”) for a customer (age=youth, income=medium, student =yes, credit= fair )?
Customer database

<table>
<thead>
<tr>
<th>ID</th>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>middle_aged</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>senior</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>senior</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>middle_aged</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>youth</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>youth</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>senior</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>youth</td>
<td>medium</td>
<td>yes</td>
<td>fair</td>
<td>?</td>
</tr>
</tbody>
</table>
What is the class (\texttt{buys\_computer = “yes”} or \texttt{buys\_computer = “no”}) for a customer (age=youth, income=medium, student =yes, credit= fair )?

Yes  = 0.0889

No     = 0.0167
Table 1 shows the class-labeled training tuples from customer database. Please calculate and illustrate the final decision tree returned by decision tree induction using information gain.
(1) What is the Information Gain of “age”?
(2) What is the Information Gain of “income”?
(3) What is the Information Gain of “student”?
(4) What is the Information Gain of “credit_rating”?
(5) What is the class (buys_computer = “yes” or buys_computer = “no”) for a customer (age=youth, income=medium, student =yes, credit= fair ) based on the classification result by decision three induction?

<table>
<thead>
<tr>
<th>ID</th>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>middle_aged</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>senior</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>senior</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>middle_aged</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>youth</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>youth</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>senior</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>
Attribute Selection Measure: Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let \( p_i \) be the probability that an arbitrary tuple in \( D \) belongs to class \( C_i \), estimated by \( |C_{i,D}|/|D| \)
- **Expected information** (entropy) needed to classify a tuple in \( D \):
  \[
  Info(D) = - \sum_{i=1}^{m} p_i \log_2(p_i)
  \]
- **Information** needed (after using \( A \) to split \( D \) into \( v \) partitions) to classify \( D \):
  \[
  Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)
  \]
- **Information gained** by branching on attribute \( A \)
  \[
  Gain(A) = Info(D) - Info_A(D)
  \]

Source: Han & Kamber (2006)
<table>
<thead>
<tr>
<th>(\log_2 (n))</th>
<th>(\log_2 (n))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-3.3219</td>
</tr>
<tr>
<td>2</td>
<td>-2.3219</td>
</tr>
<tr>
<td>3</td>
<td>-1.7370</td>
</tr>
<tr>
<td>4</td>
<td>-1.3219</td>
</tr>
<tr>
<td>5</td>
<td>-1</td>
</tr>
<tr>
<td>6</td>
<td>-0.7370</td>
</tr>
<tr>
<td>7</td>
<td>-0.5146</td>
</tr>
<tr>
<td>8</td>
<td>-0.3219</td>
</tr>
<tr>
<td>9</td>
<td>-0.1520</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>
Class P (Positive): buys_computer = “yes”
Class N (Negative): buys_computer = “no”

\[ P(buys = yes) = P_i = \frac{6}{10} = 0.6 \]
\[ P(buys = no) = P_i = \frac{4}{10} = 0.4 \]

\[
\text{Step 1: Expected information}
\]

\[
\text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2 (p_i)
\]

\[
\text{Info}(D) = I(6,4) = -\frac{6}{10} \log_2 \left(\frac{6}{10}\right) + \left(- \frac{4}{10} \log_2 \left(\frac{4}{10}\right)\right)
\]
\[
= -0.6 \times \log_2 (0.6) - 0.4 \times \log_2 (0.4)
\]
\[
= -0.6 \times (-0.737) - 0.4 \times (-1.3219)
\]
\[
= 0.4422 + 0.5288
\]
\[
= 0.971
\]

\[ \text{Info}(D) = I(6,4) = 0.971 \]
<table>
<thead>
<tr>
<th>ID</th>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>middle_aged</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>senior</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>senior</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>middle_aged</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>youth</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>youth</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>senior</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>

### Frequency Tables

#### Age

<table>
<thead>
<tr>
<th>age</th>
<th>$p_i$</th>
<th>$n_i$</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>youth</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>middle_aged</td>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>senior</td>
<td>3</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

#### Income

<table>
<thead>
<tr>
<th>income</th>
<th>$p_i$</th>
<th>$n_i$</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>high</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>medium</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>low</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

#### Student

<table>
<thead>
<tr>
<th>student</th>
<th>$p_i$</th>
<th>$n_i$</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>4</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>no</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

#### Credit Rating

<table>
<thead>
<tr>
<th>credit_rating</th>
<th>$p_i$</th>
<th>$n_i$</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>excellent</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>fair</td>
<td>4</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>
### Step 2: Information Gain

**Gain(age)**

\[
I(1,3) = -\frac{1}{4} \log_2 \left( \frac{1}{4} \right) + \left( -\frac{3}{4} \log_2 \left( \frac{3}{4} \right) \right) \\
= -0.25 \times \left[ \log_2 1 - \log_2 4 \right] + \left( -0.75 \times \left[ \log_2 3 - \log_2 4 \right] \right) \\
= -0.25 \times [0 - 2] - 0.75 \times [1.585 - 2] \\
= -0.25 \times [-2] - 0.75 \times [-0.415] \\
= 0.5 + 0.3112 = 0.8112
\]

**Gain(age)**

\[
I(2,0) = -\frac{2}{2} \log_2 \left( \frac{2}{2} \right) + \left( -\frac{0}{2} \log_2 \left( \frac{0}{2} \right) \right) \\
= -1 \times \log_2 1 + (-0 \times \log_2 0) \\
= -1 \times 0 + (-0 \times -\infty) \\
= 0 + 0 = 0
\]

**Gain(age)**

\[
I(3,1) = -\frac{3}{4} \log_2 \left( \frac{3}{4} \right) + \left( -\frac{1}{4} \log_2 \left( \frac{1}{4} \right) \right) \\
= -0.75 \times \left[ \log_2 3 - \log_2 4 \right] + \left( -0.25 \times \left[ \log_2 1 - \log_2 4 \right] \right) \\
= -0.75 \times [1.585 - 2] - 0.25 \times [0 - 2] \\
= -0.75 \times [-0.415] - 0.25 \times [-2] \\
= 0.3112 + 0.5 = 0.8112
\]

\[
\text{(1) Gain(age)} = 0.3221
\]
<table>
<thead>
<tr>
<th>income</th>
<th>$p_i$</th>
<th>$n_i$</th>
<th>total</th>
<th>$I(p_i, n_i)$</th>
<th>$I(p_i, n_i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>high</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>$I(2,2)$</td>
<td>1</td>
</tr>
<tr>
<td>medium</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>$I(2,1)$</td>
<td>0.9182</td>
</tr>
<tr>
<td>low</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>$I(2,1)$</td>
<td>0.9182</td>
</tr>
</tbody>
</table>

\[
\text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)
\]

\[
\text{Info}(D) = I(6,4) = 0.971
\]

\[
\text{Info}_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)
\]

\[
\text{Info}_{\text{income}}(D) = \frac{4}{10} \cdot I(2,2) + \frac{3}{10} \cdot I(2,1) + \frac{3}{10} \cdot I(2,1)
\]

\[
= \frac{4}{10} \times 1 + \frac{3}{10} \times 0.9182 + \frac{3}{10} \times 0.9182
\]

\[
= 0.4 + 0.2755 + 0.2755 = 0.951
\]

\[
\text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D)
\]

\[
\text{Gain}(\text{income}) = \text{Info}(D) - \text{Info}_{\text{income}}(D)
\]

\[
= 0.971 - 0.951 = 0.02
\]

\[
I(2,2) = -\frac{2}{4} \log_2 \left(\frac{2}{4}\right) + (-\frac{2}{4} \log_2 \left(\frac{2}{4}\right))
\]

\[
= -0.5 \times [\log_2 2 - \log_2 4] + (-0.5 \times [\log_2 2 - \log_2 4])
\]

\[
= -0.5 \times [1 - 2] - 0.5 \times [1 - 2]
\]

\[
= -0.5 \times [-1] - 0.5 \times [-1]
\]

\[
= 0.5 + 0.5 = 1
\]

\[
I(2,1) = -\frac{2}{3} \log_2 \left(\frac{2}{3}\right) + (-\frac{1}{3} \log_2 \left(\frac{1}{3}\right))
\]

\[
= -0.67 \times [\log_2 2 - \log_2 3] + (-0.33 \times [\log_2 1 - \log_2 3])
\]

\[
= -0.67 \times [1 - 1.585] - 0.33 \times [0 - 1.585]
\]

\[
= -0.67 \times [-0.585] - 0.33 \times [-1.585]
\]

\[
= 0.9182
\]

(2) \[\text{Gain}(\text{income}) = 0.02\]
\[
\text{student} \quad p_i \quad n_i \quad \text{total} \quad I(p_i, n_i) \quad I(p_i, n_i)
\]

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>I(4,1)</td>
</tr>
<tr>
<td>no</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>I(2,3)</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{Info}(D) &= -\sum_{i=1}^{m} p_i \log_2(p_i) \\
\text{Info}(D) &= I(6,4) = 0.971 \\
\text{Info}_A(D) &= \sum_{j=1}^{v} \left| \frac{D_j}{D} \right| \times I(D_j) \\
\text{Info}_{\text{student}}(D) &= \frac{5}{10} I(4,1) + \frac{5}{10} I(2,3) \\
&= 0.5 \times 0.7219 + 0.5 \times 0.971 \\
&= 0.36095 + 0.48545 = 0.8464 \\
\text{Gain}(A) &= \text{Info}(D) - \text{Info}_A(D) \\
\text{Gain}(\text{student}) &= \text{Info}(D) - \text{Info}_{\text{student}}(D) \\
&= 0.971 - 0.8464 = 0.1245
\end{align*}
\]

\[
I(4,1) = -\frac{4}{5} \log_2\left(\frac{4}{5}\right) + (-\frac{1}{5} \log_2\left(\frac{1}{5}\right)) \\
= -0.8 \times [\log_2 4 - \log_2 5] + (-0.2 \times [\log_2 1 - \log_2 5]) \\
= -0.8 \times [2 - 2.3219] - 0.2 \times [0 - 2.3219] \\
= -0.8 \times [-0.3219] - 0.2 \times [-2.3219] \\
= 0.25752 + 0.46438 = 0.7219
\]

\[
I(2,3) = -\frac{2}{5} \log_2\left(\frac{2}{5}\right) + (-\frac{3}{5} \log_2\left(\frac{3}{5}\right)) \\
= -0.4 \times [\log_2 0.4] + (-0.6 \times [\log_2 0.6]) \\
= -0.4 \times [-1.3219] - 0.6 \times [-0.737] \\
= 0.5288 + 0.4422 = 0.971
\]

\(3\) \(\text{Gain(}\text{student})\) = 0.1245
<table>
<thead>
<tr>
<th>credit</th>
<th>$p_i$</th>
<th>$n_i$</th>
<th>total</th>
<th>$I(p_i, n_i)$</th>
<th>$I(p_i, n_i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>excellent</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>$I(2, 2)$</td>
<td>1</td>
</tr>
<tr>
<td>fair</td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>$I(4, 2)$</td>
<td>0.9183</td>
</tr>
</tbody>
</table>

\[
I(2, 2) = -\frac{2}{4} \log_2 \left(\frac{2}{4}\right) + (-\frac{2}{4} \log_2 \left(\frac{2}{4}\right)) \\
= -0.5 \times [\log_2 2 - \log_2 4] + (-0.5 \times [\log_2 2 - \log_2 4]) \\
= -0.5 \times [1 - 2] - 0.5 \times [1 - 2] \\
= -0.5 \times [-1] - 0.5 \times [-1] \\
= 0.5 + 0.5 = 1
\]

\[
I(4, 2) = -\frac{4}{6} \log_2 \left(\frac{4}{6}\right) + (-\frac{2}{6} \log_2 \left(\frac{2}{6}\right)) \\
= -0.67 \times [\log_2 2 - \log_2 3] + (-0.33 \times [\log_2 1 - \log_2 3]) \\
= -0.67 \times [1 - 1.585] - 0.33 \times [0 - 1.585] \\
= -0.67 \times [-0.585] - 0.33 \times [-1.585] \\
= 0.9182
\]

\[
\text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2 (p_i) \\
\text{Info}(D) = I(6, 4) = 0.971
\]

\[
\text{Info}_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)
\]

\[
\text{Info}_{\text{credit}}(D) = \frac{4}{10} I(2, 2) + \frac{6}{10} I(4, 2) \\
= \frac{4}{10} \times 1 + \frac{6}{10} \times 0.9182 \\
= 0.4 + 0.5509 = 0.9509
\]

\[
\text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D)
\]

\[
\text{Gain(credit)} = \text{Info}(D) - \text{Info}_{\text{credit}}(D) \\
= 0.971 - 0.9509 = 0.019
\]

\[(4) \text{ Gain(credit)} = 0.019\]
What is the class
(buys_computer = “yes” or
buys_computer = “no”)
for a customer
(age=youth, income=medium,
student =yes, credit= fair )?
(5) What is the class (buys_computer = “yes” or buys_computer = “no”) for a customer (age=youth, income=medium, student =yes, credit= fair ) based on the classification result by decision three induction?

(5) Yes =0.0889  (No=0.0167)
age (0.3221) > student (0.1245) > income (0.02) > credit (0.019)
buys_computer = “yes”
age:youth (1/4) x student:yes (4/5) x income:medium (2/3) x credit:fair (4/6)
Yes: $\frac{1}{4} \times \frac{4}{5} \times \frac{2}{3} \times \frac{4}{6} = \frac{4}{45} = 0.0889$
buys_computer = “no”
age:youth (3/4) x student:yes (1/5) x income:medium (1/3) x credit:fair (2/6)
No: $\frac{3}{4} \times \frac{1}{5} \times \frac{1}{3} \times \frac{2}{6} = 0.01667$
What is the class
\[
(\text{bombs\_computer} = \text{“yes”}\) \text{ or } \text{bombs\_computer} = \text{“no”})
\]
for a customer
\[
\text{(age=youth, income=medium, student =yes, credit= fair )}?
\]
\[
\text{Yes } = 0.0889 \\
\text{No } = 0.0167
\]
## Customer database

<table>
<thead>
<tr>
<th>ID</th>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>middle_aged</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>senior</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>senior</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>middle_aged</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>youth</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>youth</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>senior</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>
## Customer database

<table>
<thead>
<tr>
<th>ID</th>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>middle_aged</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>senior</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>senior</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>middle_aged</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>youth</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>youth</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>senior</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>youth</td>
<td>medium</td>
<td>yes</td>
<td>fair</td>
<td>?</td>
</tr>
</tbody>
</table>
# Customer database

<table>
<thead>
<tr>
<th>ID</th>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>Class: buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>middle_aged</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>senior</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>senior</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>senior</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>middle_aged</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>youth</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>youth</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>senior</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>youth</td>
<td>medium</td>
<td>yes</td>
<td>fair</td>
<td>Yes (0.0889)</td>
</tr>
</tbody>
</table>
Support Vector Machines (SVM)
SVM—Support Vector Machines

• A new classification method for both linear and nonlinear data
• It uses a nonlinear mapping to transform the original training data into a higher dimension
• With the new dimension, it searches for the linear optimal separating hyperplane (i.e., “decision boundary”)
• With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
• SVM finds this hyperplane using support vectors (“essential” training tuples) and margins (defined by the support vectors)

Source: Han & Kamber (2006)
SVM—History and Applications

• Vapnik and colleagues (1992)—groundwork from Vapnik & Chervonenkis’ statistical learning theory in 1960s
• Features: training can be slow but accuracy is high owing to their ability to model complex nonlinear decision boundaries (margin maximization)
• Used both for classification and prediction
• Applications:
  – handwritten digit recognition, object recognition, speaker identification, benchmarking time-series prediction tests, document classification

Source: Han & Kamber (2006)
SVM—General Philosophy

Small Margin

Large Margin

Support Vectors

Source: Han & Kamber (2006)
The 2-D training data are linearly separable. There are an infinite number of (possible) separating hyperplanes or “decision boundaries.” Which one is best?

Source: Han & Kamber (2006)
Which one is better? The one with the larger margin should have greater generalization accuracy.

Source: Han & Kamber (2006)
Let data $D$ be $(X_1, y_1), \ldots, (X_{|D|}, y_{|D|})$, where $X_i$ is the set of training tuples associated with the class labels $y_i$

There are infinite lines (hyperplanes) separating the two classes but we want to find the best one (the one that minimizes classification error on unseen data)

SVM searches for the hyperplane with the largest margin, i.e., **maximum marginal hyperplane (MMH)**
SVM—Linearly Separable

- A separating hyperplane can be written as
  \[ \mathbf{W} \cdot \mathbf{X} + b = 0 \]
  where \( \mathbf{W} = \{w_1, w_2, \ldots, w_n\} \) is a weight vector and \( b \) a scalar (bias)

- For 2-D it can be written as
  \[ w_0 + w_1 x_1 + w_2 x_2 = 0 \]

- The hyperplane defining the sides of the margin:
  \[ H_1: w_0 + w_1 x_1 + w_2 x_2 \geq 1 \quad \text{for } y_i = +1, \text{ and} \]
  \[ H_2: w_0 + w_1 x_1 + w_2 x_2 \leq -1 \quad \text{for } y_i = -1 \]

- Any training tuples that fall on hyperplanes \( H_1 \) or \( H_2 \) (i.e., the sides defining the margin) are support vectors

- This becomes a constrained (convex) quadratic optimization problem: Quadratic objective function and linear constraints → Quadratic Programming (QP) → Lagrangian multipliers

Source: Han & Kamber (2006)
Why Is SVM Effective on High Dimensional Data?

- The complexity of trained classifier is characterized by the # of support vectors rather than the dimensionality of the data.
- The support vectors are the essential or critical training examples — they lie closest to the decision boundary (MMH).
- If all other training examples are removed and the training is repeated, the same separating hyperplane would be found.
- The number of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality.
- Thus, an SVM with a small number of support vectors can have good generalization, even when the dimensionality of the data is high.

Source: Han & Kamber (2006)
SVM—Linearly Inseparable

- Transform the original input data into a higher dimensional space

Example 6.8 Nonlinear transformation of original input data into a higher dimensional space. Consider the following example. A 3D input vector $\mathbf{X} = (x_1, x_2, x_3)$ is mapped into a 6D space $\mathbf{Z}$ using the mappings $\phi_1(\mathbf{X}) = x_1, \phi_2(\mathbf{X}) = x_2, \phi_3(\mathbf{X}) = x_3$, $\phi_4(\mathbf{X}) = (x_1)^2$, $\phi_5(\mathbf{X}) = x_1 x_2$, and $\phi_6(\mathbf{X}) = x_1 x_3$. A decision hyperplane in the new space is $d(\mathbf{Z}) = \mathbf{WZ} + b$, where $\mathbf{W}$ and $\mathbf{Z}$ are vectors. This is linear. We solve for $\mathbf{W}$ and $b$ and then substitute back so that we see that the linear decision hyperplane in the new ($\mathbf{Z}$) space corresponds to a nonlinear second order polynomial in the original 3-D input space,

$$d(\mathbf{Z}) = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 (x_1)^2 + w_5 x_1 x_2 + w_6 x_1 x_3 + b$$

$$= w_1 z_1 + w_2 z_2 + w_3 z_3 + w_4 z_4 + w_5 z_5 + w_6 z_6 + b$$

- Search for a linear separating hyperplane in the new space

Source: Han & Kamber (2006)
Mapping Input Space to Feature Space

Source: http://www.statsoft.com/textbook/support-vector-machines/
Instead of computing the dot product on the transformed data tuples, it is mathematically equivalent to instead applying a kernel function $K(X_i, X_j)$ to the original data, i.e., $K(X_i, X_j) = \Phi(X_i) \Phi(X_j)$

Typical Kernel Functions

- Polynomial kernel of degree $h$: $K(X_i, X_j) = (X_i \cdot X_j + 1)^h$
- Gaussian radial basis function kernel: $K(X_i, X_j) = e^{-\|X_i - X_j\|^2 / 2\sigma^2}$
- Sigmoid kernel: $K(X_i, X_j) = \tanh(\kappa X_i \cdot X_j - \delta)$

SVM can also be used for classifying multiple (> 2) classes and for regression analysis (with additional user parameters)

Source: Han & Kamber (2006)
SVM Related Links

• SVM Website

• Representative implementations
  – LIBSVM
    • an efficient implementation of SVM, multi-class classifications, nu-SVM, one-class SVM, including also various interfaces with java, python, etc.
  – SVM-light
    • simpler but performance is not better than LIBSVM, support only binary classification and only C language
  – SVM-torch
    • another recent implementation also written in C.

Source: Han & Kamber (2006)
Deep Learning and Neural Networks
Deep Learning and Neural Networks

Input Layer (X)  Hidden Layer (H)  Output Layer (Y)

X1  

X2  

Y
Deep Learning and Neural Networks

Input Layer (X)   Hidden Layer (H)   Output Layer (Y)
Deep Learning and Neural Networks

Input Layer (X)

Hidden Layers (H)

Output Layer (Y)

Deep Neural Networks
Deep Learning
Data Mining
Evaluation
Evaluation

(Accuracy of Classification Model)
Assessing the Classification Model

• Predictive accuracy
  —Hit rate
• Speed
  —Model building; predicting
• Robustness
• Scalability
• Interpretability
  —Transparency, explainability

Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson
Accuracy vs. Precision

A: High Accuracy, High Precision
B: Low Accuracy, High Precision
C: High Accuracy, Low Precision
D: Low Accuracy, Low Precision
Accuracy vs. Precision

A
High Accuracy
High Precision
High Validity
High Reliability

B
Low Accuracy
High Precision
Low Validity
High Reliability

C
High Accuracy
Low Precision
High Validity
Low Reliability

D
Low Accuracy
Low Precision
Low Validity
Low Reliability
Accuracy vs. Precision

A: High Accuracy, High Precision, High Validity, High Reliability
B: Low Accuracy, High Precision, Low Validity, High Reliability
C: High Accuracy, Low Precision, High Validity, Low Reliability
D: Low Accuracy, Low Precision, Low Validity, Low Reliability
**Accuracy of Classification Models**

- In classification problems, the primary source for accuracy estimation is the **confusion matrix**

![Confusion Matrix Diagram]

- **Accuracy**
  \[ \text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \]

- **True Positive Rate**
  \[ \text{True Positive Rate} = \frac{TP}{TP + FN} \]

- **True Negative Rate**
  \[ \text{True Negative Rate} = \frac{TN}{TN + FP} \]

- **Precision**
  \[ \text{Precision} = \frac{TP}{TP + FP} \]

- **Recall**
  \[ \text{Recall} = \frac{TP}{TP + FN} \]

Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), *Business Intelligence, Analytics, and Data Science: A Managerial Perspective*, 4th Edition, Pearson
Estimation Methodologies for Classification

- **Simple split** (or holdout or test sample estimation)
  - Split the data into 2 mutually exclusive sets: training (~70%) and testing (30%)
  - For ANN, the data is split into three sub-sets: (training [~60%], validation [~20%], testing [~20%])

Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson
Estimation Methodologies for Classification

• **k-Fold Cross Validation** (rotation estimation)
  – Split the data into $k$ mutually exclusive subsets
  – Use each subset as testing while using the rest of the subsets as training
  – Repeat the experimentation for $k$ times
  – Aggregate the test results for true estimation of prediction accuracy training

• Other estimation methodologies
  – Leave-one-out, bootstrapping, jackknifing
  – Area under the ROC curve

Source: Turban et al. (2011), Decision Support and Business Intelligence Systems
$k$-Fold Cross-Validation

Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson
Estimation Methodologies for Classification
Area under the ROC curve

Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson
Ensemble Models

Heterogeneous Ensemble

Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson
Sensitivity = True Positive Rate

Specificity = True Negative Rate
True Class (actual value)

<table>
<thead>
<tr>
<th>Predictive Class (prediction outcome)</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td>Negative</td>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Predictive Class (prediction outcome)</th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td>Negative</td>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>

True Positive Rate (Sensitivity) = $\frac{TP}{TP + FN}$

True Negative Rate (Specificity) = $\frac{TN}{TN + FP}$

False Positive Rate = $\frac{FP}{FP + TN}$

False Positive Rate (1 - Specificity) = $\frac{FP}{FP + TN}$

Accuracy = $\frac{TP + TN}{TP + TN + FP + FN}$

True Positive Rate = $\frac{TP}{TP + FN}$

True Negative Rate = $\frac{TN}{TN + FP}$

Precision = $\frac{TP}{TP + FP}$

Recall = $\frac{TP}{TP + FN}$

True Positive Rate (Sensitivity) = \( \frac{TP}{TP + FN} \)

**Sensitivity**
- True Positive Rate
- Recall
- Hit rate
- \( TP / (TP + FN) \)

**Specificity**

= True Negative Rate

= \( \frac{TN}{N} \)

= \( \frac{TN}{TN + FP} \)

\( True \ Negative \ Rate \ (Specificity) = \frac{TN}{TN + FP} \)

\( False \ Positive \ Rate \ (1 - Specificity) = \frac{FP}{FP + TN} \)

**True Class** (actual value)  
<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>

**Predictive Class** (prediction outcome)  
<table>
<thead>
<tr>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Positive (TP)</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td>False Negative (FN)</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>

**Total**  
| P | N |

**Precision**  
= Positive Predictive Value (PPV)  
\[ \text{Precision} = \frac{TP}{TP + FP} \]

**Recall**  
= True Positive Rate (TPR)  
= Sensitivity  
= Hit Rate  
\[ \text{Recall} = \frac{TP}{TP + FN} \]

**F1 score (F-score)(F-measure)**  
is the harmonic mean of precision and recall  
= \( \frac{2TP}{P + P'} \)  
= \( \frac{2TP}{2TP + FP + FN} \)

\[ F = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \]

A

<table>
<thead>
<tr>
<th>TP</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>63</td>
<td>28</td>
</tr>
<tr>
<td>37</td>
<td>72</td>
</tr>
</tbody>
</table>

**TPR = 0.63**

FPR = 0.28

PPV = 0.69

F1 = 0.66

ACC = 0.68

**Recall**

= True Positive Rate (TPR)

= Sensitivity

= Hit Rate

= TP / (TP + FN)

**Specificity**

= True Negative Rate

= TN / N

= TN / (TN + FP)

**TPR** = TP / (TP + FN)

**FPR** = FP / (FP + TN)

**PPV** = TP / (TP + FP)

**Precision**

= Positive Predictive Value (PPV)

**F1 score (F-score)**

(F-measure)

is the harmonic mean of precision and recall

F = 2 * \( \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \)

**ACC** = (TP + TN) / (TP + TN + FP + FN)

= 135 / 200 = 67.5

**A**

- **True Positives (TP):** 63
- **False Positives (FP):** 28
- **False Negatives (FN):** 37
- **True Negatives (TN):** 72

- **True Positive Rate (TPR):** $\frac{63}{63+28} = \frac{63}{91} = 0.69$
- **False Positive Rate (FPR):** $\frac{28}{63+28} = \frac{28}{91}$
- **Positive Predictive Value (PPV):** $\frac{63}{63+28} = \frac{63}{91}$
- **F1 Score:** $2 \times \frac{0.63 \times 0.69}{0.63 + 0.69} = \frac{2 \times 63}{100 + 91} = \frac{135}{200} = 0.675$
- **Accuracy (ACC):** $\frac{63 + 72}{200} = \frac{135}{200} = 0.675$

**B**

- **True Positives (TP):** 77
- **False Positives (FP):** 23
- **False Negatives (FN):** 77
- **True Negatives (TN):** 23

- **True Positive Rate (TPR):** $\frac{77}{77+23} = \frac{77}{100} = 0.77$
- **False Positive Rate (FPR):** $\frac{23}{77+23} = \frac{23}{100}$
- **Positive Predictive Value (PPV):** $\frac{77}{77+23} = \frac{77}{100} = 0.77$
- **F1 Score:** $2 \times \frac{0.77 \times 0.50}{0.77 + 0.50} = \frac{2 \times 77}{154} = \frac{154}{200} = 0.77$
- **Accuracy (ACC):** $\frac{77 + 23}{200} = \frac{100}{200} = 0.50$

**Recall**

- **Recall** = True Positive Rate (TPR)
- **Recall** = Sensitivity
- **Recall** = Hit Rate

**Precision**

- **Precision** = Positive Predictive Value (PPV)

Recall
= True Positive Rate (TPR)
= Sensitivity
= Hit Rate

Precision
= Positive Predictive Value (PPV)

\[
Recall = \frac{TP}{TP + FN} \\
Precision = \frac{TP}{TP + FP}
\]

Welcome to Colaboratory!

Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. See our FAQ for more info.

Getting Started

- Overview of Colaboratory
- Loading and saving data: Local files, Drive, Sheets, Google Cloud Storage
- Importing libraries and installing dependencies
- Using Google Cloud BigQuery
- Forms, Charts, Markdown, & Widgets
- TensorFlow with GPU
- Machine Learning Crash Course: Intro to Pandas & First Steps with TensorFlow

Highlighted Features

Seedbank

Looking for Colab notebooks to learn from? Check out Seedbank, a place to discover interactive machine learning examples.

TensorFlow execution

Colaboratory allows you to execute TensorFlow code in your browser with a single click. The example below adds two matrices.

\[
\begin{bmatrix}
1 & 1 & 1
\end{bmatrix}
+\begin{bmatrix}
1 & 2 & 3
\end{bmatrix} = \begin{bmatrix}
2 & 3 & 4
\end{bmatrix}
\]
Python in Google Colab

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

```
# Future Value
pv = 100
r = 0.1
n = 7
fv = pv * ((1 + (r)) ** n)
print(round(fv, 2))
```

```
amount = 100
interest = 10 #10% = 0.01 * 10
years = 7
future_value = amount * ((1 + (0.01 * interest)) ** years)
print(round(future_value, 2))
```

```
# Python Function def
def getfv(pv, r, n):
    fv = pv * ((1 + (r)) ** n)
    return fv
fv = getfv(100, 0.1, 7)
print(round(fv, 2))
```

```
# Python if else
score = 80
if score >=60 :
    print("Pass")
else:
    print("Fail").
```

Pass
Iris flower data set

setosa  versicolor  virginica

Source: https://en.wikipedia.org/wiki/Iris_flower_data_set
Iris Classification

iris.data


5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
4.3,3.0,1.1,0.1,Iris-setosa
5.8,4.0,1.2,0.2,Iris-setosa
5.7,4.4,1.5,0.4,Iris-setosa
5.4,3.9,1.3,0.4,Iris-setosa
5.1,3.5,1.4,0.3,Iris-setosa
5.7,3.8,1.7,0.3,Iris-setosa
5.1,3.8,1.5,0.3,Iris-setosa
5.4,3.4,1.7,0.2,Iris-setosa
5.1,3.7,1.5,0.4,Iris-setosa
4.6,3.6,1.0,0.2,Iris-setosa
5.1,3.3,1.7,0.5,Iris-setosa
4.8,3.4,1.9,0.2,Iris-setosa
5.0,3.0,1.6,0.2,Iris-setosa
5.0,3.4,1.6,0.4,Iris-setosa

setosa

virginica

versicolor
Google Colab
Google Colab

Connect apps to Drive

Colaboratory
offered by https://colab.research.google.com
A data analysis tool that combines code, output, and descriptive text into one collaborative document.
Connect Colaboratory to Google Drive
Google Colab
Google Colab
Google Colab
Run Jupyter Notebook
Python3 GPU
Google Colab
Google Colab Python Hello World
print('Hello World')
Data Visualization in Google Colab

import seaborn as sns
iris = sns.load_dataset("iris")
g = sns.pairplot(iris, hue="species")

Source: https://seaborn.pydata.org/generated/seaborn.pairplot.html
import seaborn as sns
sns.set(style="ticks", color_codes=True)
iris = sns.load_dataset("iris")
g = sns.pairplot(iris, hue="species")

Source: https://seaborn.pydata.org/generated/seaborn.pairplot.html
```
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix

# Load dataset
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)

print(df.head(10))
print(df.tail(10))
print(df.describe())
print(df.info())
print(df.shape)
print(df.groupby('class').size())

plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()

df.hist()
plt.show()

scatter_matrix(df)
plt.show()

sns.pairplot(df, hue="class", size=2)
```
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix

# Import Libraries
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix
print('imported')

imported

names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']

df = pd.read_csv(url, names=names)
print(df.head(10))

# Load dataset
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)
print(df.head(10)).

<table>
<thead>
<tr>
<th>sepal-length</th>
<th>sepal-width</th>
<th>petal-length</th>
<th>petal-width</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>3.5</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.9</td>
<td>3.0</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.7</td>
<td>3.2</td>
<td>1.3</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.6</td>
<td>3.1</td>
<td>1.5</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>3.6</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.4</td>
<td>3.9</td>
<td>1.7</td>
<td>0.4</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.6</td>
<td>3.4</td>
<td>1.4</td>
<td>0.3</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>5.0</td>
<td>3.4</td>
<td>1.5</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.4</td>
<td>2.9</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris-setosa</td>
</tr>
<tr>
<td>4.9</td>
<td>3.1</td>
<td>1.5</td>
<td>0.1</td>
<td>Iris-setosa</td>
</tr>
</tbody>
</table>
**print(df.tail(10)).**

<table>
<thead>
<tr>
<th>sepal-length</th>
<th>sepal-width</th>
<th>petal-length</th>
<th>petal-width</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>140</td>
<td>6.7</td>
<td>3.1</td>
<td>5.6</td>
<td>2.4 Iris-virginica</td>
</tr>
<tr>
<td>141</td>
<td>6.9</td>
<td>3.1</td>
<td>5.1</td>
<td>2.3 Iris-virginica</td>
</tr>
<tr>
<td>142</td>
<td>5.8</td>
<td>2.7</td>
<td>5.1</td>
<td>1.9 Iris-virginica</td>
</tr>
<tr>
<td>143</td>
<td>6.8</td>
<td>3.2</td>
<td>5.9</td>
<td>2.3 Iris-virginica</td>
</tr>
<tr>
<td>144</td>
<td>6.7</td>
<td>3.3</td>
<td>5.7</td>
<td>2.5 Iris-virginica</td>
</tr>
<tr>
<td>145</td>
<td>6.7</td>
<td>3.0</td>
<td>5.2</td>
<td>2.3 Iris-virginica</td>
</tr>
<tr>
<td>146</td>
<td>6.3</td>
<td>2.5</td>
<td>5.0</td>
<td>1.9 Iris-virginica</td>
</tr>
<tr>
<td>147</td>
<td>6.5</td>
<td>3.0</td>
<td>5.2</td>
<td>2.0 Iris-virginica</td>
</tr>
<tr>
<td>148</td>
<td>6.2</td>
<td>3.4</td>
<td>5.4</td>
<td>2.3 Iris-virginica</td>
</tr>
<tr>
<td>149</td>
<td>5.9</td>
<td>3.0</td>
<td>5.1</td>
<td>1.8 Iris-virginica</td>
</tr>
</tbody>
</table>
```python
print(df.describe())
```

<table>
<thead>
<tr>
<th></th>
<th>sepal-length</th>
<th>sepal-width</th>
<th>petal-length</th>
<th>petal-width</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>150.000000</td>
<td>150.000000</td>
<td>150.000000</td>
<td>150.000000</td>
</tr>
<tr>
<td>mean</td>
<td>5.843333</td>
<td>3.054000</td>
<td>3.758667</td>
<td>1.198667</td>
</tr>
<tr>
<td>std</td>
<td>0.828066</td>
<td>0.433594</td>
<td>1.764420</td>
<td>0.763161</td>
</tr>
<tr>
<td>min</td>
<td>4.300000</td>
<td>2.000000</td>
<td>1.000000</td>
<td>0.100000</td>
</tr>
<tr>
<td>25%</td>
<td>5.100000</td>
<td>2.800000</td>
<td>1.600000</td>
<td>0.300000</td>
</tr>
<tr>
<td>50%</td>
<td>5.800000</td>
<td>3.000000</td>
<td>4.350000</td>
<td>1.300000</td>
</tr>
<tr>
<td>75%</td>
<td>6.400000</td>
<td>3.300000</td>
<td>5.100000</td>
<td>1.800000</td>
</tr>
<tr>
<td>max</td>
<td>7.900000</td>
<td>4.400000</td>
<td>6.900000</td>
<td>2.500000</td>
</tr>
</tbody>
</table>
print(df.info())
print(df.shape)

print(df.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
sepal-length 150 non-null float64
sepal-width 150 non-null float64
petal-length 150 non-null float64
petal-width 150 non-null float64
class 150 non-null object
dtypes: float64(4), object(1)
memory usage: 5.9+ KB
None

print(df.shape)

(150, 5)
```python
df.groupby('class').size()
```

```
print(df.groupby('class').size())
```

<table>
<thead>
<tr>
<th>class</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris-setosa</td>
<td>50</td>
</tr>
<tr>
<td>Iris-versicolor</td>
<td>50</td>
</tr>
<tr>
<td>Iris-virginica</td>
<td>50</td>
</tr>
</tbody>
</table>

dtype: int64
```python
plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()
```
```python
df.hist()
plt.show()
```
scatter_matrix(df)
plt.show()
sns.pairplot(df, hue="class", size=2)
Classification and Prediction

https://colab.research.google.com/drive/1QE7fR2OxHiQ0_p6l1nnZDIFF354Nf_Lw

Data Mining and Machine Learning in Google Colab

```python
# Import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix

# Import sklearn
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier

print("Imported")

# Load dataset
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)

df.head(10)
df.tail(10)
df.describe()
df.info()
df.shape
df.groupby('class').size()

plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()
df.hist()
plt.show()
```

https://colab.research.google.com/drive/1QE7fR2OxHiQ0_p6l1nnZDIFF354Nf_Lw
```python
# Load dataset
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)

print(df.head(10))
print(df.tail(10))
print(df.describe())
print(df.info())
print(df.shape)
print(df.groupby('class').size())

plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()

df.hist()
plt.show()

scatter_matrix(df)
plt.show()
sns.pairplot(df, hue="class", size=2).
```
# Load dataset
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)

print(df.head(10))
print(df.tail(10))
print(df.describe())
print(df.info())
print(df.shape)
print(df.groupby('class').size())

plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()

df.hist()
plt.show()

scatter_matrix(df)
plt.show()

sns.pairplot(df, hue="class", size=2).
df.corr()

<table>
<thead>
<tr>
<th></th>
<th>sepal-length</th>
<th>sepal-width</th>
<th>petal-length</th>
<th>petal-width</th>
</tr>
</thead>
<tbody>
<tr>
<td>sepal-length</td>
<td>1.000000</td>
<td>-0.109369</td>
<td>0.871754</td>
<td>0.817954</td>
</tr>
<tr>
<td>sepal-width</td>
<td>-0.109369</td>
<td>1.000000</td>
<td>-0.420516</td>
<td>-0.356544</td>
</tr>
<tr>
<td>petal-length</td>
<td>0.871754</td>
<td>-0.420516</td>
<td>1.000000</td>
<td>0.962757</td>
</tr>
<tr>
<td>petal-width</td>
<td>0.817954</td>
<td>-0.356544</td>
<td>0.962757</td>
<td>1.000000</td>
</tr>
</tbody>
</table>

https://colab.research.google.com/drive/1QE7fR2OxHiQ0_p6l1nnZDIFF354Nf_Lw
# Split-out validation dataset

array = df.values
X = array[:,0:4]
Y = array[:,4]
validation_size = 0.20
seed = 7

X_train, X_validation, Y_train, Y_validation =
model_selection.train_test_split(X, Y, test_size=validation_size, random_state=seed)
scoring = 'accuracy'

len(Y_validation).
# Models
models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('DT', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC()))
# evaluate each model in turn
results = []
names = []
for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=seed)
    cv_results =
model_selection.cross_val_score(model, X_train, Y_train, cv=kfold, scoring=scoring)
results.append(cv_results)
names.append(name)
msg = "%s: %.4f (%.4f)" % (name, cv_results.mean(), cv_results.std())
print(msg)
```python
# Models
models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('DT', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC()))

# evaluate each model in turn
results = []
names = []

for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=seed)
    cv_results = model_selection.cross_val_score(model, X_train, Y_train, cv=kfold, scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg = "%s: %.4f (%.4f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)

LR: 0.9667 (0.0408)
LDA: 0.9750 (0.0382)
KNN: 0.9833 (0.0333)
DT: 0.9750 (0.0382)
NB: 0.9750 (0.0534)
SVM: 0.9917 (0.0250)
```
# Make predictions on validation dataset
model = KNeighborsClassifier()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)
print("%.4f" % accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
print(model)
# Make predictions on validation dataset
model = KNeighborsClassifier()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)
print("%.4f" % accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
print(model)

0.9000
[[ 7  0  0]
 [ 0 11  1]
 [ 0  2  9]]

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris-setosa</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>7</td>
</tr>
<tr>
<td>Iris-versicolor</td>
<td>0.85</td>
<td>0.92</td>
<td>0.88</td>
<td>12</td>
</tr>
<tr>
<td>Iris-virginica</td>
<td>0.90</td>
<td>0.82</td>
<td>0.86</td>
<td>11</td>
</tr>
<tr>
<td>avg / total</td>
<td>0.90</td>
<td>0.90</td>
<td>0.90</td>
<td>30</td>
</tr>
</tbody>
</table>

KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
metric_params=None, n_jobs=1, n_neighbors=5, p=2,
weights='uniform')

https://colab.research.google.com/drive/1QE7fR2OxHiQ0_p6l1nnZDIIFF354Nf_Lw
# Make predictions on validation dataset
model = SVC()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)
print("%.4f" % accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
print(model)
```python
model = SVC()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)

# Make predictions on validation dataset
model = SVC()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)
predictions
print("%.4f" % accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
print(model)
```

```
0.9333
[[ 7  0  0]
 [ 0 10  2]
 [ 0  0 11]]  
precision   recall   f1-score   support
 Iris-setosa    1.00    1.00    1.00    7
 Iris-versicolor 1.00    0.83    0.91   12
 Iris-virginica 0.85    1.00    0.92   11

avg / total    0.94   0.93    0.93   30

SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
   decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
   max_iter=-1, probability=False, random_state=None, shrinking=True,
   tol=0.001, verbose=False)
```
# Make predictions on validation dataset
model = DecisionTreeClassifier()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)
print("%.4f" % accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
print(model)
# Make predictions on validation dataset
model = GaussianNB()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)
print("%.4f" % accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
print(model)

0.8333
[[ 7  0  0]
 [ 0  9  3]
 [ 0  2  9]]

<table>
<thead>
<tr>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris-setosa</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Iris-versicolor</td>
<td>0.82</td>
<td>0.75</td>
<td>0.78</td>
</tr>
<tr>
<td>Iris-virginica</td>
<td>0.75</td>
<td>0.82</td>
<td>0.78</td>
</tr>
</tbody>
</table>

avg / total 0.84 0.83 0.83 30

GaussianNB(priors=None)
```python
# Make predictions on validation dataset
model = LogisticRegression()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)
print("%.4f" % accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
print(model)

0.8000
[[ 7  0  0]
 [ 0  7  5]
 [ 0 11  0]]

                   precision    recall  f1-score   support

   Iris-setosa      1.00      1.00      1.00           7
 Iris-versicolor   0.88      0.58      0.70          12
 Iris-virginica   0.67      0.91      0.77          11

   avg / total    0.83      0.80      0.80          30

LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
                   penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
                   verbose=0, warm_start=False)
```

https://colab.research.google.com/drive/1QE7fR20xHiQ0_p6l1nnZDIFF354Nf_Lw
# Make predictions on validation dataset
model = LinearDiscriminantAnalysis()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)
print("%.4f" % accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
print(model)

0.9667
[[  7   0   0]
 [  0  11   1]
 [  0   0  11]]

<table>
<thead>
<tr>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris-setosa</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Iris-versicolor</td>
<td>1.00</td>
<td>0.92</td>
<td>0.96</td>
</tr>
<tr>
<td>Iris-virginica</td>
<td>0.92</td>
<td>1.00</td>
<td>0.96</td>
</tr>
<tr>
<td>avg / total</td>
<td>0.97</td>
<td>0.97</td>
<td>0.97</td>
</tr>
</tbody>
</table>

LinearDiscriminantAnalysis(n_components=None, priors=None, shrinkage=None, solver='svd', store_covariance=False, tol=0.0001)
# Make predictions on validation dataset
model = MLPClassifier()
model.fit(X_train, Y_train)
predictions = model.predict(X_validation)
print("%.4f" % accuracy_score(Y_validation, predictions))
print(confusion_matrix(Y_validation, predictions))
print(classification_report(Y_validation, predictions))
print(model)

0.9000
[[  7  0  0]
 [  0  9  3]
 [  0  0 11]]

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris-setosa</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>7</td>
</tr>
<tr>
<td>Iris-versicolor</td>
<td>1.00</td>
<td>0.75</td>
<td>0.86</td>
<td>12</td>
</tr>
<tr>
<td>Iris-virginica</td>
<td>0.79</td>
<td>1.00</td>
<td>0.88</td>
<td>11</td>
</tr>
</tbody>
</table>

avg / total 0.92 0.90 0.90 30

MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9, beta_2=0.999, early_stopping=False, epsilon=1e-08, hidden_layer_sizes=(100,), learning_rate='constant', learning_rate_init=0.001, max_iter=200, momentum=0.9, nesterovs_momentum=True, power_t=0.5, random_state=None, shuffle=True, solver='adam', tol=0.0001, validation_fraction=0.1, verbose=False, warm_start=False)
Summary

• Supervised Learning
• Classification and Prediction
• Decision Tree (DT)
  – Information Gain (IG)
• Support Vector Machine (SVM)
• Data Mining Evaluation
  – Accuracy
  – Precision
  – Recall
  – F1 score (F-measure) (F-score)
References

- Jiawei Han, Micheline Kamber and Jian Pei (2011), Data Mining: Concepts and Techniques, Third Edition, Morgan Kaufmann 2011.