TECHNISCHE UNIVERSITÄT DRESDEN

SEMANTIC COMPUTING

Lecture 4: Introduction to Machine Learning

Dagmar Gromann

International Center For Computational Logic

TU Dresden, 9 November 2018

Overview

- Introduction to Machine Learning
- Supervised Machine Learning
  - Naïve Bayes
  - Support Vector Machine(s) (SVM)

Introduction To Machine Learning

Definition Machine Learning

Tom M. Mitchell

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E."

Alan Turing

"Can machines think?"

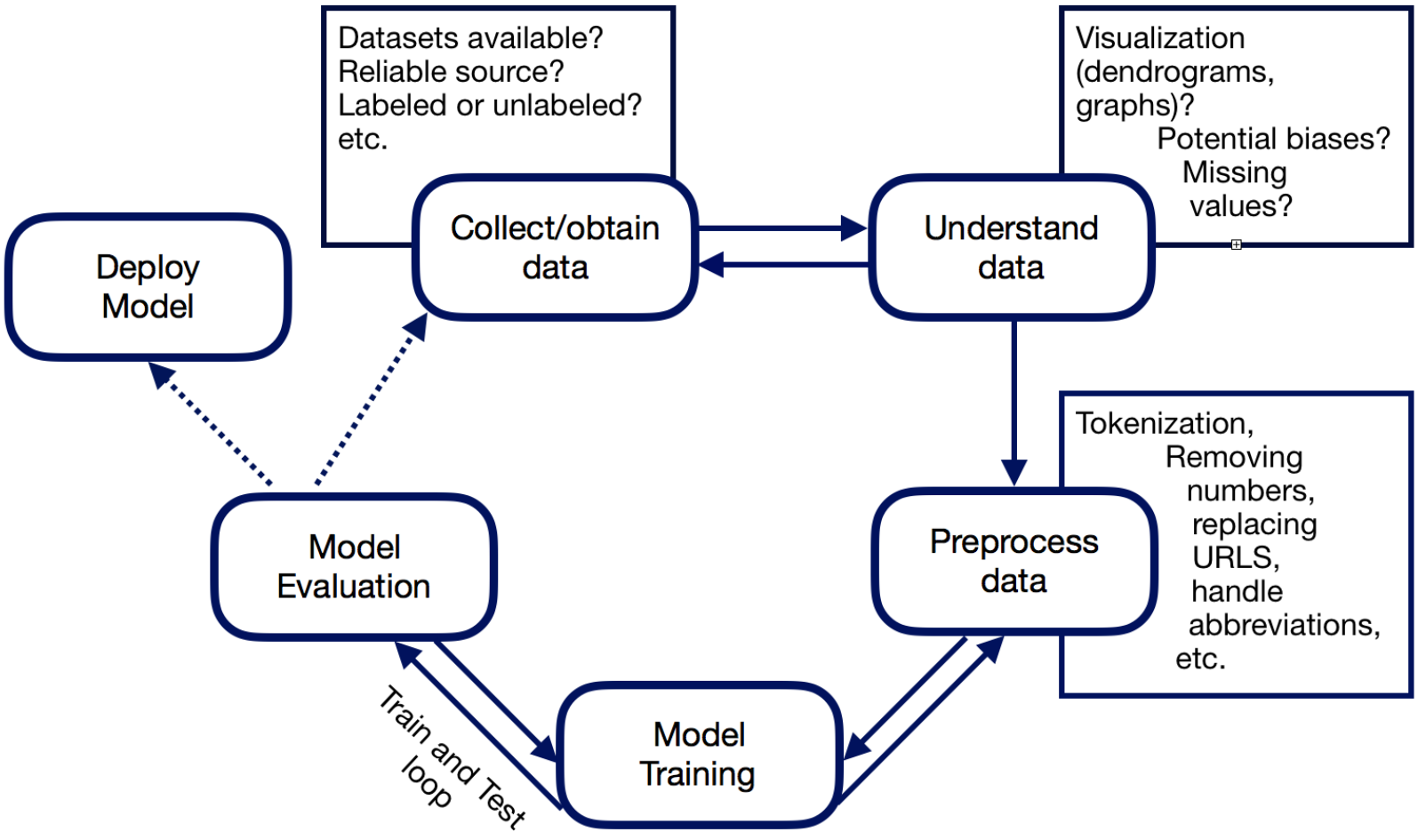
More general

A set of paradigms and algorithms that target the detection of regularities in real-world observable records in order to infer patterns and extract insights. The goal in most tasks is to **predict** outcomes for new data (previously unseen by the model).

Machine Learning Applications

- Image Recognition
- Text Classification
- Document Categorization
- Speech Recognition
- Spam Detection
- Fraud Detection
- NLP
- Playing Games
- ...

Workflow


Datasets available?
Reliable source?
Labeled or unlabeled?
etc.
Visualization (dendrograms, graphs)?
Potential biases?
Missing values?
Collect/obtain data
Understand data
Deploy Model
Tokenization,
Removing numbers,
replacing URLS,
handle abbreviations,
etc.
Model Evaluation
Preprocess data
Train and Test loop
Model Training

Some Machine Learning Paradigms

- **Supervised learning**: learning with labeled examples
- **Semi-supervised** or **weakly supervised learning**: learning with labeled and unlabeled data
- **Unsupervised learning**: learning with unlabeled data
- **Distant supervision**: automatically generate labeled data (by creating examples from existing resources, e.g. large knowledge bases such as Freebase or sentiment lexicons)
- **Reinforcement learning**: learning with indirect or delayed feedback (trial and error / reward and punishment)

Supervised Machine Learning

Supervised Machine Learning

Definition

Supervised machine learning refers to a set of algorithms that learn a hypothesis that approximates a function to map an input to an output given a set of training input-output pairs. It learns from examples and thus requires labeled training data.

Data

Data sets

Training set: data annotated or usually hand-labeled with the correct category/answer

Validation set: to adjust (hyper-)parameters of model without running risk of the model fitting to the test set without you noticing

Test set: new examples that test the generalizability of the model trained on the training set

Feature Selection

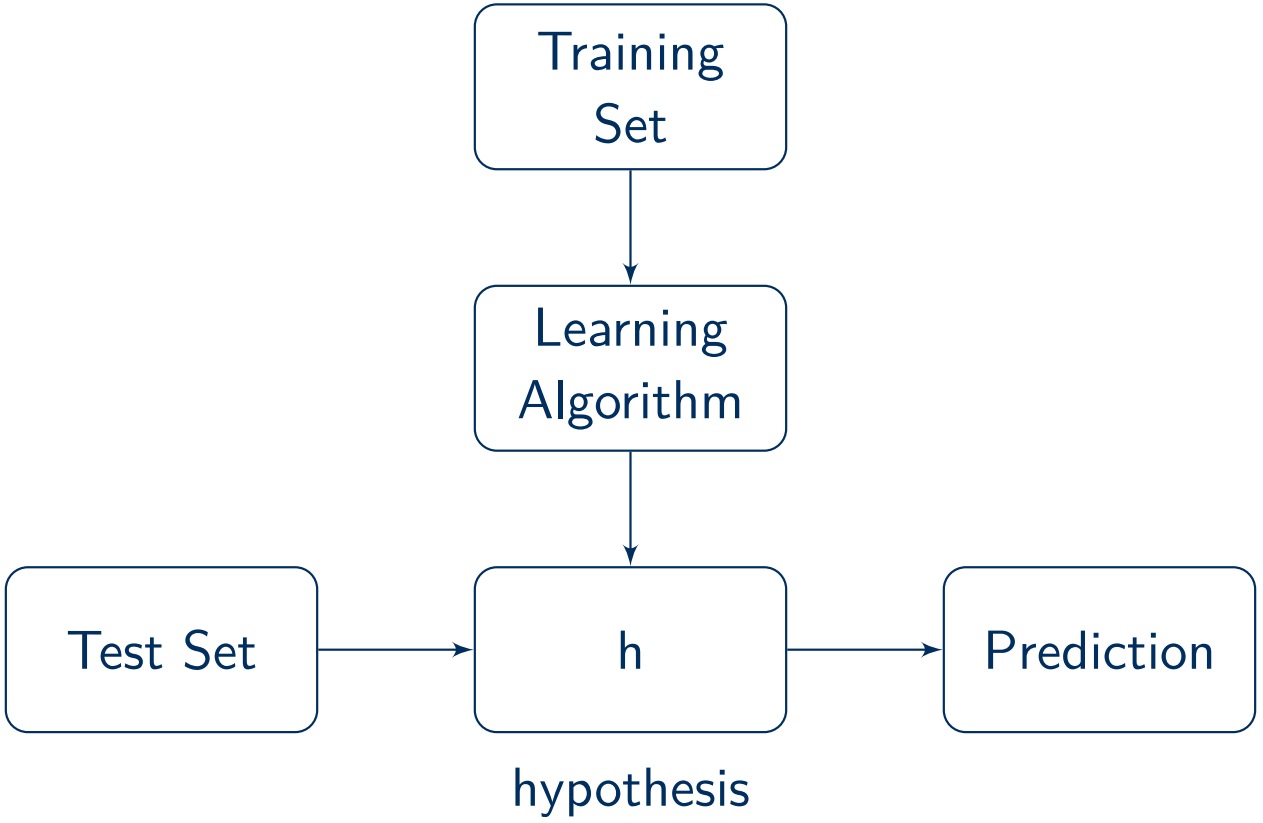
Features are the inputs we provide to our model

Can be provided:

Problem: Predict someones salary based on the years of experience they have.
Which features do we have here?

Answer: only the years of experience

Supervised Learning Process



Inductive Learning

Given a number of input/output examples, inductive learning provides a general rule that fits the examples.

Inductive Inferences

A set of examples (x_i, y_i) is generated from a target function $f(x)$ such that $f(x_i) = y_i$ where x refers to the input and y to the output.

Task: Find a hypothesis h such that $h \approx f$ given training examples (x_i, y_i)

Evaluation: on the test set, that is, a set of previously unseen examples (x_i, y_i)

Generalization

- The identified hypothesis must generalize to correctly classify/include instances that are not in the training set
- Memorizing the training examples leads to an hypothesis that is consistent with the training data but does not generalize to the test data ("overfitting")
- **Ockham's razor**: the simplest solution tends to be the correct one, that is, the simplest hypothesis that is consistent with the data might be the best one

Overfitting and Underfitting

Overfitting

Producing a model that performs very well on the data you used for training but generalizes poorly on new data.

Underfitting

Producing a model that does not perform well, not even on the training data

Avoid Overfitting

- **Cross-Validation**: one way to find prediction errors is to use k-fold cross validation where the partition used as a test set varies with each iteration.
- **Early Stopping**: stop the training procedure at the point of the smallest error with respect to the validation set
- **Pruning**: pruning removes nodes from decision trees that add little predictive power to the problem at hand
- **Regularization**: introducing a penalty term to the error function in order to discourage the coefficients form reaching large values

One Way to Avoid Overfitting

k-Fold Cross-Validation:

- Create K training and test sets ("folds") within training set
- For each iteration of K run classifier and test performance against k test set

Typical Supervised Machine Learning Problems

- **Classification**: from data to discrete classes
- **Regression**: predicting a continuous value

Classification

Text classification

Task of assigning categories to input text, such as

- Assign subject categories, topics, or genres,
- Classify as spam or not spam (spam detection),
- Classify into a set of given languages (language identification),
- Assign an emotion class (Pepper Multi-Modal Emotion Classification example),
- Assign a text to an author from a set (authorship identification),
- ...

Example problems

Which of the followings are examples of supervised classification problems?

- Analyze bank data for fraudulent transactions ✖
- Recommend music to someone given their previous choices and many features of that music ✔
- Recognize someone in a photo given a repository of tagged photos ✔
- Cluster students into types based on their learning styles ✖

Classification Method

- Input:
  - a document d
  - a number of features f of the documents
  - a fixed set of classes $\mathrm{C} = \mathrm{c}_1, \mathrm{c}_2, ..., \mathrm{c}_\mathrm{m}$
  - a training set m or hand-labeled documents $(\mathrm{d}_1, \mathrm{c}_1), ..., (\mathrm{d}_\mathrm{m}, \mathrm{c}_\mathrm{m})$
- Output:
  - a learned classifier $\mathrm{y} : \mathrm{d}, \mathrm{f} -> \mathrm{c}$

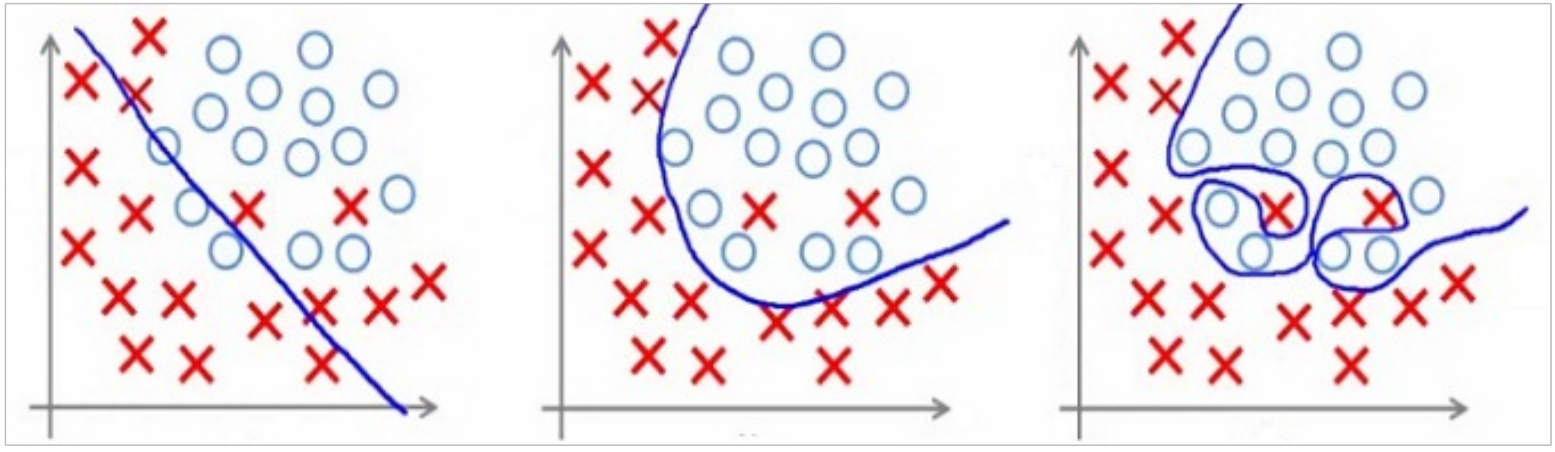
Overfitting in a Classification Task

Example dataset for binary classification in sklearn.

```
from sklearn import datasets
cancer = datasets.load_breast_cancer()
list(data.target_names)
```

['malignant', 'benign']

Example of overfitting/underfitting in a classification setting.



Common Classifiers

- Naïve Bayes
- Support-Vector Machines (SVM)
- Decision Trees
- k-Nearest Neighbors
- Logistic regression
- ...

Naïve Bayes

Bayes' rule

Bayes' rule or Bayes' theorem

Probability of an event is based on prior knowledge of conditions that might be related to the event.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

A = event "wrinkles"; B = event "age"; P(B) != zero

$P(A|B)$ = likelihood of event A occurring given that B is true

$P(B|A)$ = likelihood of event B occurring given that A is true

$P(A)$ and $P(B)$ are the probabilities of observing A and B independently of each other

Bayes' rule example

Given a number of patients with liver disease (P(A) = 0.10) and a number of patients who are alcoholics (P(B) = 0.05) - which are independent from each other - and of the patients diagnosed with liver disease, 7% are alcoholics (P(B|A) = 0.07), we want to know what a patient's probability is for having liver disease if they are an alcoholic?

Bayes' theorem tells us:
$\mathrm{P(A|B)} = \frac{(0.07 * 0.1)}{0.05} = 0.14$

If the patient is an alcoholic their chances of having liver disease is 0.14 (14%).

Bayes' rule continued

- P(A) and P(B) are often referred to as **prior probabilities** because they are probabilities of events A and B that we know before (prior to) obtaining additional information
- P(A|B) are often referred to as **posterior probabilities** because they are the probabilities of the event after we have obtained additional information

Naïve Bayes

- Build a model that is able to classify input given a number of categories.
- This model is able to recognize features, such as edges or color of objects
- As a model we have:
  - Classes C = rectangle, circle
  - Features that are boolean variables for edges and color
- Given the value of the features of a new input, we can calculate the class with the highest probability.

Naïve Bayes continued

Two main assumptions:

- Bayes' rule applies
- Each word (or other input) is independent form all other words (conditional independence)

In text classification, the objective is to find the best class C for a specific document d. The best class in Naïve Bayes is the most likely or maximum a posterior (MAP) class c_{map}:

$$
\begin{aligned}
c_{map} &= \text{argmax}_{c \in C} P(c|x_1, x_2, x_3) \\
&= \text{argmax}_{c \in C} \frac{P(x_1, x_2, x_3|c)P(c)}{P(x_1, x_2, x_3)} \ \#\text{Bayes}'\text{rule} \\
&= \text{argmax}_{c \in C} P(c) \prod_{i=1}^{n} P(x_i|c) \ \ \#\text{conditional independence}
\end{aligned}
$$

Maximum Likelihood Estimate (MLE)

If all hypotheses are a priori equally likely, we only need to consider the P(d|c) term (or rather as a representation of all of d's features x_i):

$$c_{MLE} = \text{argmax}_{c \in C} \prod_{i=1}^{n} P(x_i|c)$$

Why is it called Naïve?

"Each word (or other input) is independent form all other words (or inputs)"

In reality this is not true! Words depend on each other. But this ignoring of word order and dependencies greatly simplifies the algorithm and it works well in practice. This is why it is called Naïve.

Naïve Bayes MLE example

word	topic	count
a	sports	0
ball	sports	1
carrot	sports	0
game	sports	2
I	sports	2
saw	sports	2
the	sports	3

Assume 5 sports documents

Counts are number of documents on the sports topic containing each word:

$$\mathrm{P}(\mathrm{a}|\mathrm{sports}) = \frac{0}{5}$$
$$\mathrm{P}(\mathrm{ball}|\mathrm{sports}) = \frac{1}{5}$$

We know 0 is not a good estimator!

Laplace Smoothing

Make it very unlikely but not impossible that a word occurs in a certain document with a specific topic.

word	topic	count
a	sports	1
ball	sports	2
carrot	sports	1
game	sports	3
I	sports	3
saw	sports	3
the	sports	4

Assume 5 sports documents

Add γ (here $\gamma = 1$) to the count of the words and adjust the number of documents by the vocabulary (v):

$$\mathrm{P}(\mathrm{word}|\mathrm{topic}) = \frac{\mathrm{N}(\mathrm{word},\mathrm{topic})+\gamma}{\mathrm{N}(\mathrm{topic})+\gamma \mathrm{v}}$$

$$\mathrm{P}(\mathrm{a}|\mathrm{sports}) = \frac{1}{12}$$

$$\mathrm{P}(\mathrm{ball}|\mathrm{sports}) = \frac{2}{12}$$

Feature extraction from text

A sequence of symbols cannot be directly fed into a machine learning algorithm since they usually expect numerical feature vectors with a fixed size rather than documents of variable length.

We need to extract features from strings by:

- tokenizing
- counting
- normalizing and weighting

= Bag of Words (BoW) representation

A corpus of documents can then be represented by a matrix with one row per document and one column per token (e.g. word). Frequently, this process of turning a text into feature vectors is called **vectorization**.

Bag-of-Words (BoW)

Common representation of vocabulary in classification; represents the number of known words as a vocabulary and measures their occurrence without considering the order of occurrence.

I live in Dresden
Dresden is a city
I really hate pigeons in the city

unique words = [I,live,in,Dresden,is,a,city,really,hate,pigeons,the]
"I live in Dresden" = [1,1,1,1,0,0,0,0,0,0,0]
"Dresden is a city" = [0,0,0,1,1,1,1,0,0,0,0]

In sklearn: CountVectorizer that does tokenization and filtering of stopwords automatically and builds a dictionary of features and transforms documents into feature vectors.

TF-IDF

Measure from information extraction that turns the occurrence counts to frequencies => necessary, because longer documents = higher average counts - potential discrepancies.

Definition

- **Term Frequency**: measures how frequently a term occurs in a document:
  TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document)
- **Inverse Document Frequency**: measures how important a term is (weigh down frequent terms, such as “the” or “is”, and scale up the rare ones)
  IDF(t) = log_e(Total number of documents / Number of documents with term t in it)

Evaluation: Binary Classification

		Prediction	
		Yes	No
Actual label	Yes	True Positive (TP)	False Negative (FN)
	No	False Positive (FP)	True Negative (TN)

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

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

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

$$\text{F1 score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Evaluation: Multi-Class Classification

A **confusion matrix** also called error matrix is used to visualize and calculate the performance of a supervised algorithm. Here, 3 of the 8 cats where wrongly classified as dogs.

		Prediction		
		Cat	Dog	Rabbit
Actual label	Cat	**5**	2	0
	Dog	3	**3**	2
	Rabbit	0	1	**11**

$$\text{Observed accuracy} = \frac{\text{sum of diagonal}}{\text{test set size}} = \frac{5+3+11}{27}$$

test set size = addition of column sums

Review of Lecture 4

- What are some important machine learning paradigms?
- What is supervised machine learning?
- For which scenarios could supervised machine learning be used?
- What are important problems of supervised machine learning?
- How can overfitting be avoided and what is it?
- What is Naïve Bayes? What makes it Naïve?
- What is a Bag-of-Words (BoW)? What is TF-IDF?
- How can we evaluate a classification algorithm?