AI in Medicine

Intro to Machine Learning

Konstantine Tsafatinos

Charité - Universitätsmedizin Berlin
konstantinos.tsafatinos@bccn-berlin.de

09/07/2020

Overview

Unsupervised Learning

Overview

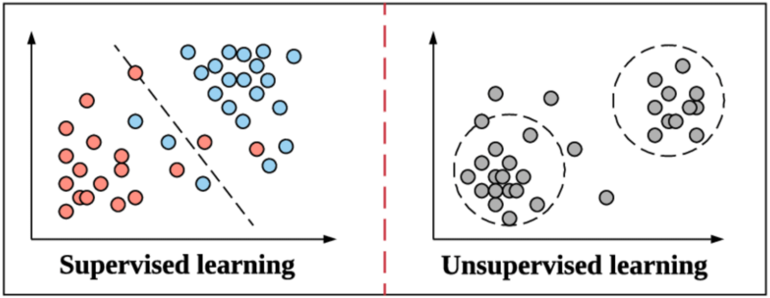


Figure 1: A visualized difference in unsupervised learning and supervised learning

- Used for data exploration, preprocessing, and clustering

Unsupervised Learning

Overview

- Unsupervised learning is used when we want to extract features or find representations of the data distribution. Structures in the data are learned and labels are unnecessary for learning. Some examples are:
  - finding directions of highest variance
  - finding clusters or groups in the data
  - finding a lower dimension representation

Unsupervised Learning

Overview

- It is sometimes necessary to use "whitening", a way to normalize your data using PCA, before running your classifier

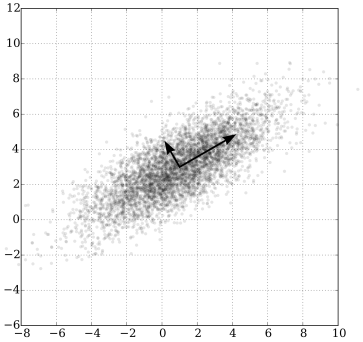


Figure 2: PCA of a multivariate Gaussian distribution

Supervised Learning

Overview

- Sources:
  1. https://www.researchgate.net/figure/Examples-of-Supervised-Learning-Linear-Regression-and-Unsupervised-Learning_fig3_336642133
  2. https://en.wikipedia.org/wiki/Principal_component_analysis
  3. https://zhuanlan.zhihu.com/p/28298776

Supervised Learning
Introduction

- **Goal:** find a function that maps a vector of input features X, to a vector of labels y using example input-output pairs.

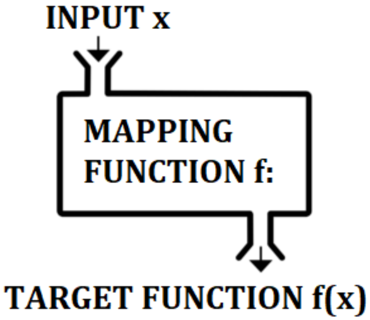


Figure 3: Mapping visualization

Supervised Learning

Introduction

- **Classification:** Output maps to a class, discrete variable

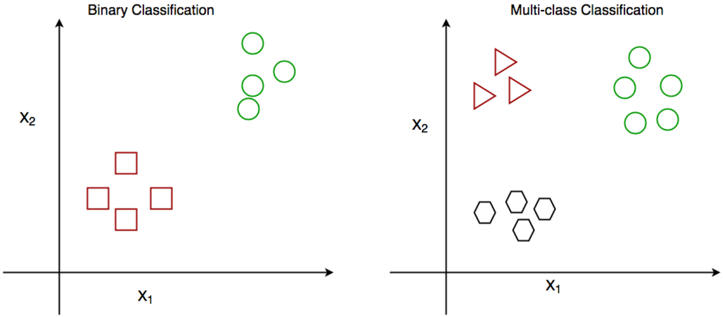


Figure 4: Discrete learning problem

Supervised Learning

Introduction

- **Regression:** Output maps to a continuous variable

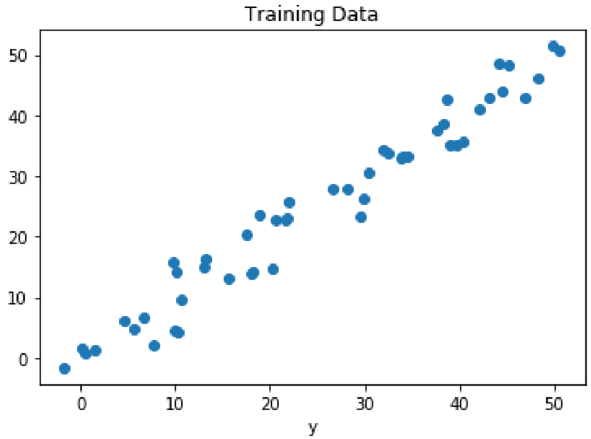


Figure 5: Continuous learning problem

Supervised Learning
Models

- It is important that we choose models that are appropriate for the task and learning problem
- We will learn how to use the following models in scikit-learn:
    - Logistic Regression
    - Support Vector Machines

Supervised Learning

Logistic Regression

- The Logistic function is a sigmoidal function; a function that takes in real value $t, (t \in \mathbb{R})$ and outputs values between 0 and 1
- This can be interpreted as a probability of output class in terms of input

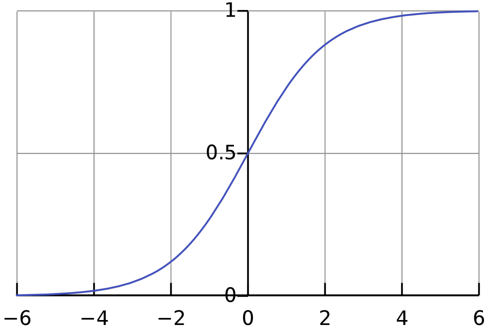


Figure 6: Logistic Function

Supervised Learning

Logistic Regression

- The logistic regression model itself does not perform statistical classification, although it can be used to make a classifier
- A common way to do this is to set a threshold value so that everything above a cutoff belongs to one class, and everything below belongs to the other
- Let $\sigma(t)$ be the logistic function

$$p(X) = \sigma(t) = \frac{1}{1+e^{-t}},\ t = \theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n$$

- t is a linear combination of inputs

Supervised Learning

Logistic Regression

- In order to learn the parameters θ_0, θ_1, $\cdots \theta_n$, we need to define a cost function that we can optimize with respect to our weights
- We then update the weights using the back propagated error

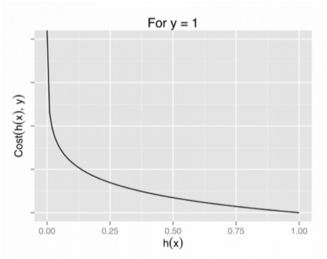


Figure 7: Cost for $y = 1$

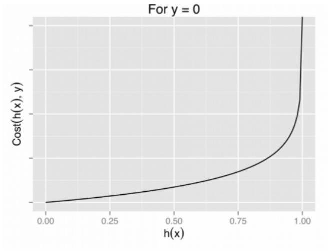


Figure 8: Cost for $y = 0$

Supervised Learning

Logistic Regression

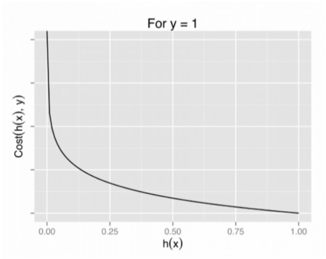


Figure 9: Cost for $y = 1$

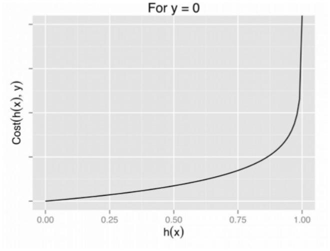


Figure 10: Cost for $y = 0$

- Combined Cost Function:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}log(h_\theta(x^{(i)})) + (1 - y^{(i)})log(1 - h_\theta(x^{(i)}))] \tag{1}$$

Supervised Learning
Logistic Regression

- We want to minimize our cost function w.r.t. the weights θ_j
- Using the gradient descent algorithm we update our weights using in the direction of our gradient (change in cost) w.r.t θ_j :

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \tag{2}$$

- α, the learning rate. It controls the size of the step we take in the direction of the gradient during learning

Supervised Learning

Logistic Regression

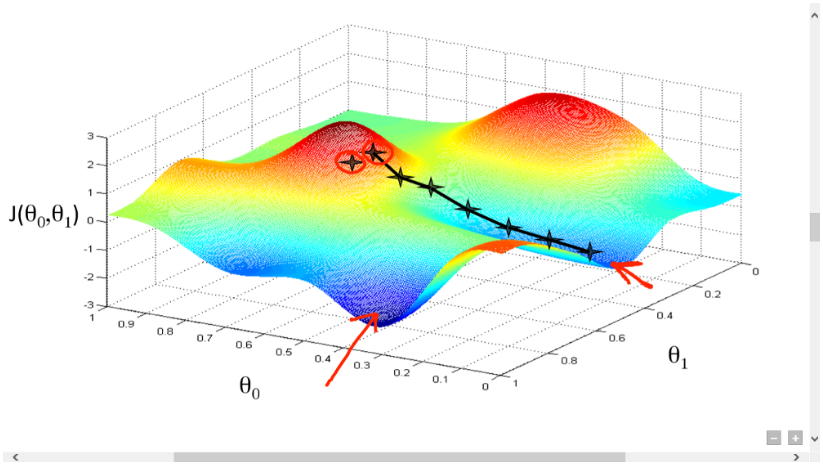


Figure 11: Visualizing gradient descent

Supervised Learning

Logistic Regression

- This process of learning the optimal parameters that minimize cost is similar to how neural network models use gradient descent for learning.
- Sources:
  1. https://www.geeksforgeeks.org/ml-classification-vs-regression/
  2. https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148
  3. https://en.wikipedia.org/wiki/Logistic_regression

Supervised Learning
Support Vector Machines

- Support vector machines (SVMs) can solve linear and non-linear problems
- The algorithm finds a line or hyperplane that can separate the data into classes
- We find points closest to the line of separation called support vectors
- The distance between the line and the support vectors is called the margin. Our goal is to maximize this margin.

Supervised Learning

Support Vector Machines

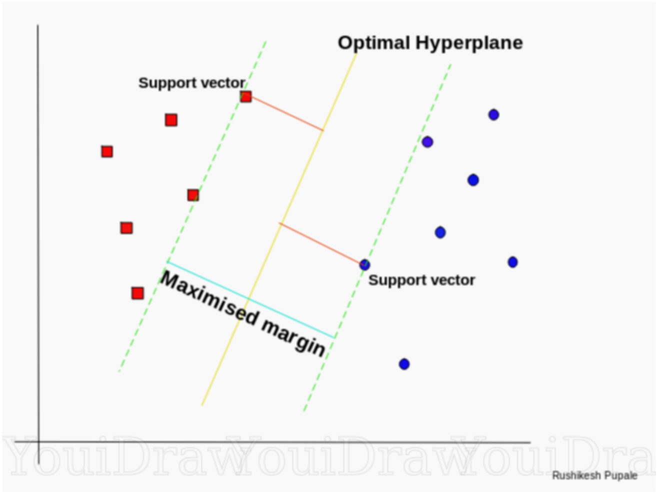


Figure 12: Learning the optimal hyperplane

Supervised Learning

Support Vector Machines

- Using kernels, we can extend support vector machines to deal with non-linearly separable data
- A common non-linear kernel to use is a radial basis function (RBF), which uses a Gaussian to measure the distance between the input and a fixed point
- Good for radially distributed data or for data in clusters

Supervised Learning
Support Vector Machines

- Examples of data distribution:

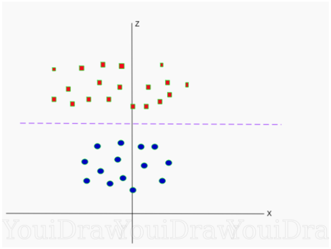


Figure 13: linear problem

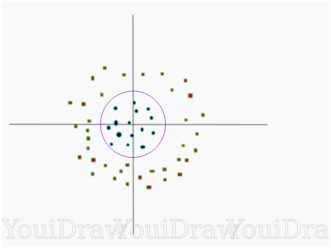


Figure 14: non-linear problem

Supervised Learning

Support Vector Machines

- The SVM has two parameters (C and γ) we can tune to change the behaviour of the classifier
  - Tuning C controls the trade off between a smooth decision boundary and classifying all training data correctly. Small C means a smooth decision boundary. This can be useful to avoid over fitting the training data
  - Tuning γ controls how much to weight points when calculating the decision boundary. Small γ means that points are weighted more equally. Large γ means that points close to the decision boundary have more weight

Supervised Learning

Support Vector Machines

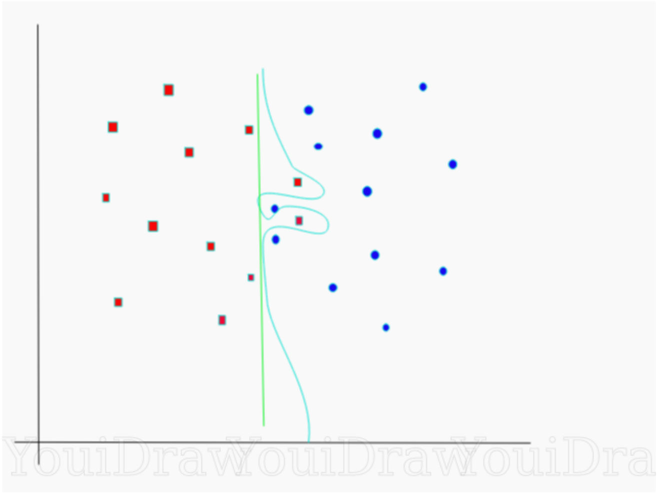


Figure 15: Effects of tuning hyper parameters for SVMs

Supervised Learning

Support Vector Machines

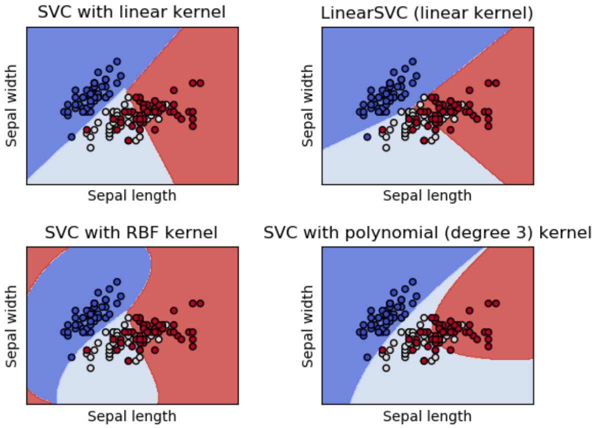


Figure 16: SVMs using different kernels

Supervised Learning

Support Vector Machines

- Sources:
  1. https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989
  2. https://en.wikipedia.org/wiki/Radial_basis_function
  3. https://en.wikipedia.org/wiki/Support_vector_machine
  4. https://scikit-learn.org/stable/auto_examples/svm/plot_iris_svc.html

Data Preparation

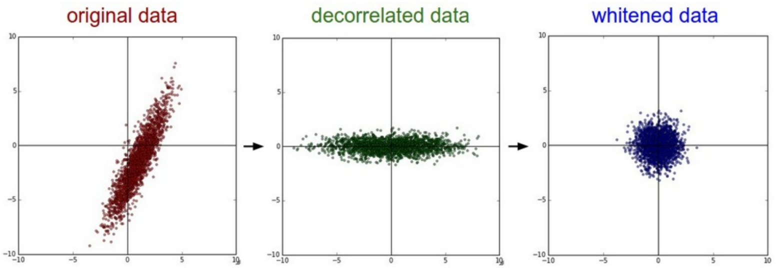
Preprocessing

Before data can be fed into a classifier, it usually needs to be preprocessed. Some preprocessing steps include:

- Dropping non-numeric values
- Cleaning unneeded columns
- Standardizing labels and/or magnitudes
- Normalizing ranges
- Converting measurement units
- Finding a good representation

Data Preparation

Preprocessing



PCA / Whitening. **Left**: Original toy, 2-dimensional input data. **Middle**: After performing PCA. The data is centered at zero and then rotated into the eigenbasis of the data covariance matrix. This decorrelates the data (the covariance matrix becomes diagonal). **Right**: Each dimension is additionally scaled by the eigenvalues, transforming the data covariance matrix into the identity matrix. Geometrically, this corresponds to stretching and squeezing the data into an isotropic gaussian blob.

Figure 17: Visualization of whitening

Data Preparation

Train Test Split

- Once we have a cleaned and processed data set, we then need to split up our data set into a training and a test set
- The proportions in which you split your data are usually determined via a heuristic that is model and data set size dependent
- Using 20% of our data set as the test set is a common starting point

Training

Fitting a Model

- Now that we have a data set ready to be used in training, we can fit several models to the training data
- A learning algorithm analyzes the training set and learns a mapping function from features to labels
- Tuning hyperparameters and using regularization techniques can prevent overfitting of the training data
- Ideally, your model should not over fit the data so that it can perform well and generalize to unseen data.

Training

Fitting a Model

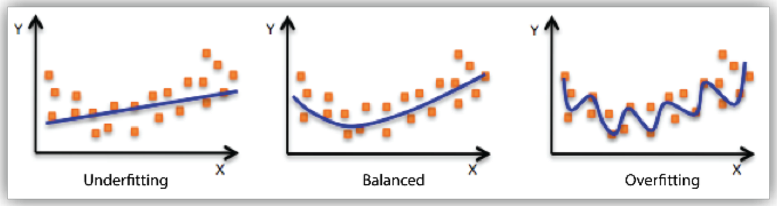


Figure 18: Example of models fitting data

- Source:
  1. https://docs.aws.amazon.com/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html

Model Evaluation

Metrics

- After we have fit a model to the training data, it is vital that we evaluate its performance on the test set. This will give us an idea of:
  - how well our model fit the data
  - how well our classifier approximated our target function and
  - how well it can generalize to new data
- A common metric to use is accuracy:

$$\text{accuracy} = \frac{tp + tn}{tp + fp + tn + fn} \tag{3}$$

- Accuracy is a measure of how well our classifier can determine the true labels of inputs

Model Evaluation

Metrics

Outputs of a classifier can be generalized using the following four outcomes:

- True positives, False positives, True negatives, False negatives

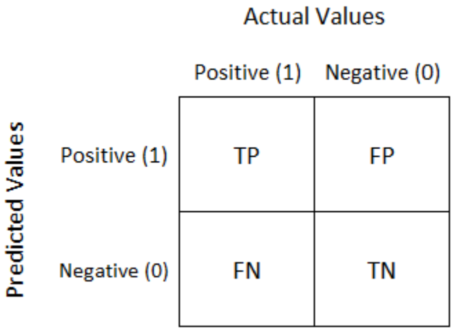


Figure 19: Confusion matrix

Model Evaluation

Metrics

Other metrics used to evaluate your classifier:

- Variable Definitions:
    - tp = true positives
    - fp = false positives
    - tn = true negatives
    - fn = false negatives
- Precision
    - $p = \frac{tp}{tp+fp}$, how precise the model is when classifying a sample as positive.
    - It can be thought of as the classifiers ability to **not** misclassify negatives samples.

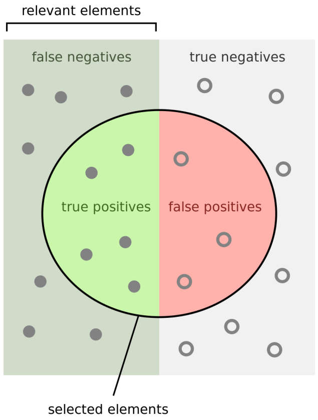
Model Evaluation

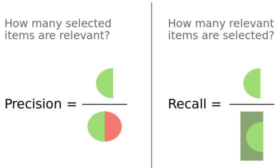
Metrics

- Recall
  - $r = \frac{tp}{tp+fn}$, is a measure of the ability of the classifer to find all positive samples
- F_1-Score
  - $F_1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = 2 \cdot \frac{\text{recall} \cdot \text{precision}}{\text{recall} + \text{precision}}$, the harmonic mean between precision and recall
  - An F_1-score reaches its best value at 1; perfect precision and recall
- Support
  - The support is the number of occurrences of each class in our test set

Model Evaluation

Metrics




How many selected items are relevant?
Precision =
How many relevant items are selected?
Recall =

Model Evaluation

Metrics

- AUC - ROC Curve
  - A performance measure for our classifier at various threshold levels
  - Area under the curve (AUC) of the Receiver Operating Curve (ROC) is a measure of how well the classifier can distinguish between the classes
  - ROC is plotted with the False Positive Rate (FPR) along the x-axis and the True Positive Rate along y-axis

Model Evaluation

Metrics

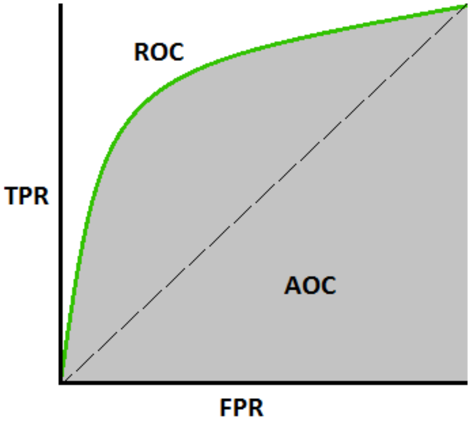


Figure 20: ROC Visualization - Dotted line is chance level

Model Evaluation

Metrics

- Sources:
  1. https://en.wikipedia.org/wiki/Precision_and_recall
  2. https://en.wikipedia.org/wiki/F1_score
  3. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
  4. https://en.wikipedia.org/wiki/Accuracy_and_precision
  5. https://en.wikipedia.org/wiki/Accuracy_and_precision
  6. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5
  7. https://towardsdatascience.com/building-a-logistic-regression-in-python-step-by-step-becd4d56c9c8

Exercises

scikit-learn

- Make sure you have updated versions of all the python packages listed on the notebook
- We will be implementing a logistic regression classifier and an SVM classifier in an interactive python environment notebook
- The jupyter notebook will walk you through an implementation of a pipeline for analyzing a diabetes dataset. You will follow along and then implement the following:
  - Loading data from a csv file
  - Separating data into training and test sets
  - Training a Logistic Regression and a SVM classifier
  - Evaluating both Classifiers using various metrics

Intro to ML and scikit-learn

python

Thank you for your continued attention! :)
Let's dive into implementing what we have just learned in code