Применение многослойных нейронных сетей для решения прикладных задач обработки данных

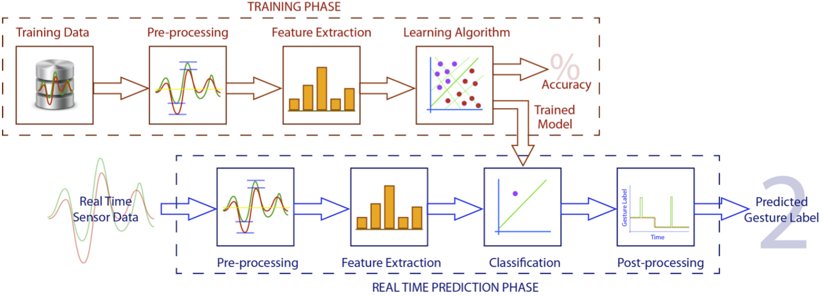
А.Г. Трофимов
к.т.н., доцент, НИЯУ МИФИ

lab@neuroinfo.ru
http://datalearning.ru

Курс “Нейронные сети”

Апрель 2020

Machine Learning Pipeline



- Training phase
- Prediction phase

- Data preprocessing
- Feature extraction
- Learning algorithm
- Accuracy estimation
- Output post-processing

Data Preprocessing

GIGO (Garbage In, Garbage Out) Principle

Nonsense input data produces nonsense output or “garbage”

In real world, a large amount of data sets are usually noisy, inconsistent and unstructured in nature

Data preprocessing is possibly one of the most boring and time-consuming part of building a neural network

Types of data preprocessing:

- Data cleaning
- Data integration
- Data reduction
- Data transformation

Data Cleaning

Definition

Data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate data from a dataset and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty data

Phases of data cleaning:

- Identification phase
  The purpose is to identify and clarify the true nature of the worrisome data points, patterns, and statistics
- Treatment phase
  After identification of errors, missing values, and true (extreme or normal) values, the researcher must decide what to do with problematic observations

Dirty Data

Data cleaning deals with:

- Missing values
- Duplicate data
- Outliers
- Inconsistencies (contradictions in data)

Causes of dirty data:

- Data entry errors (human errors)
- Measurement errors (instrument errors)
- Data processing errors (data manipulation or dataset unintended mutations)
- Intentional (errors made to hide data or complicate processing)
- Sampling errors (extracting or incorrect joining data from wrong or various sources)

Approaches to Missing Values Treatment

- Substitute missing values by default value
- Manually filling missing values
  Manually replacing NaN values by some supposed value
- Deleting training examples that contain missing values
  Leads to losing data which may be valuable (even though incomplete)
- Deleting features that contain missing values
  Can lead to huge loss of information
- Imputation of missing values
  It is the best approach to missing values treatment

Definition

Data imputation is the process of replacing missing data with substituted values

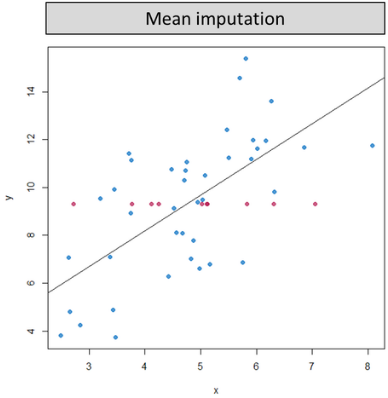
Data Imputation Strategies

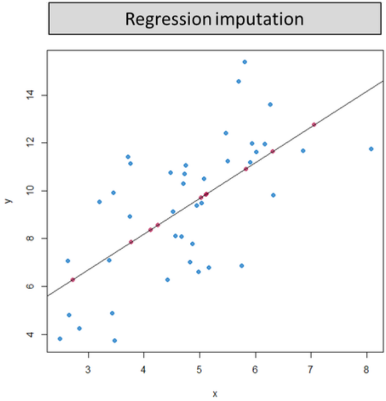
- Single imputation
  Missing value is replaced by a value
- Multiple imputation
  Missing values are imputed m times that leads to m different completed datasets. The final result is obtained by pooling these m datasets

Single imputation strategies:

- Mean (median, mode) substitution
  Substitution to mean (median, mode) value of the feature
- K nearest neighbours (KNN)
  Missing value is filled by feature's mean value over nearest neighbours of the example that contains missing value
- Regression
  Substitution to value estimated by a regression model

Data Imputation.Illustrations

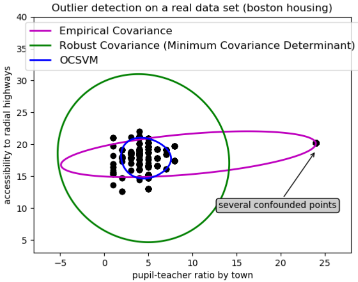


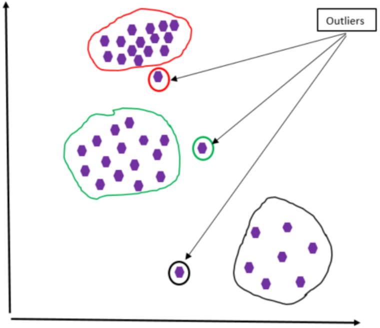


Outlier Detection

Definition

An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism





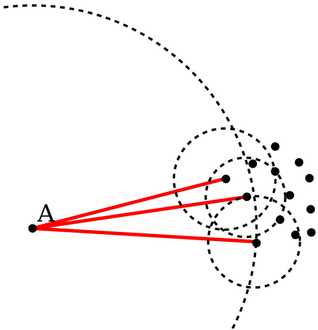
Approaches to Outlier Detection

- Model-based approaches
  Idea: determine a probabilistic model of the data. Outliers are points that do not fit to the model
  Methods: statistical tests, fitting an elliptic envelope (for Gaussian model), isolation forest, PCA, etc.
- Proximity-based approaches
  Idea: examine the spatial proximity of each object in the data space. If the proximity of an object considerably deviates from the proximity of other objects it is considered an outlier
  Methods: clustering, nearest neighbours analysis, local outlier factor (LOF), etc.
- Angle-based approaches
  Idea: examine the spectrum of pairwise angles between a given point and all other points. Outliers are points that have a spectrum with low fluctuation
  Methods: Angle-based outlier detection (ABOD), etc.

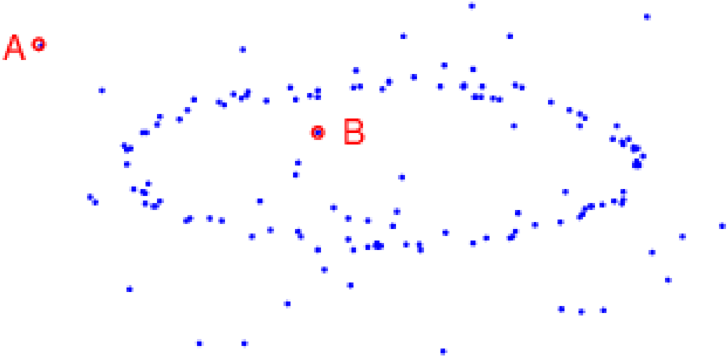
Proximity-Based Approaches

Proximity-based approaches examine spatial proximity of sample data

- Distance-based outlier detection
  An object o is an outlier if its neighbourhood does not have enough other points
- Density-based outlier detection
  An object o is an outlier if its density is relatively much lower than that of its neighbours



Outlier Detection. Illustration



Proximity-based methods are good at detecting outliers similar to point A, while model-based methods can better detect outliers similar to point B

Hypersphere in High-Dimensional Space

For a standard 1-dimensional normal distribution $U \sim N(0,1)$:

$$P(||U|| < 1.6) = 0.9$$

90% of the data are in the interval [–1.6,1.6]

For a standard 2-dimensional normal distribution $U \sim N_2(0, I_2)$:

$$P(||U|| < 1.6) = P(\sqrt{U_1^2 + U_2^2} < 1.6) = P(\chi^2(2) < 2.56) = 0.72$$

72% of the data are in the circle of radius 1.6

For a standard M-dimensional normal distribution:

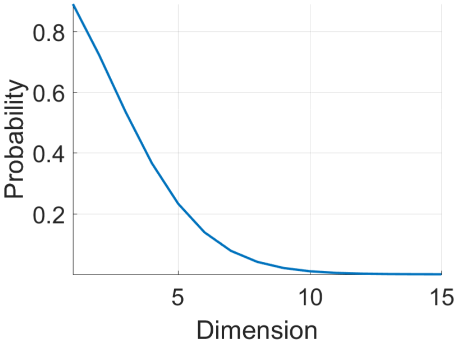
$$P(||U|| < 1.6) = P(\chi^2(M) < 2.56)$$

As M grows this probability tends to 0 for all fixed radius

Phenomenon of the Empty Spaces

As the dimension M of the space increases, the hypersphere of fixed radius becomes an insignificant volume in it. The sample becomes very sparse

Probability $P(\chi^2(M) < 2.56)$



Spatial Proximity in High-Dimensional Space

In high-dimensional space:

- The relative contrast between the nearest and the farthest neighbour becomes rather poor for broad range of data distributions:

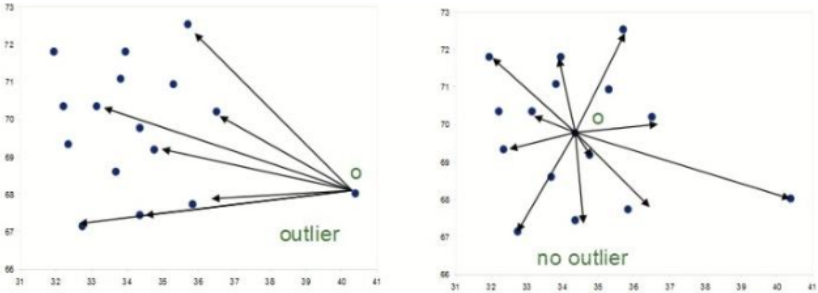
$$\lim_{M\to\infty} \frac{dist_{\max} - dist_{\min}}{dist_{\min}} = 0$$

  All samples are equally far from each other
- Data is very sparse, almost all points are outliers
- Concept of spatial neighbourhood becomes meaningless

Solutions:

- Use high-dimensional approaches (angle-based approaches)
- Find outliers in projections (subspaces) of the original feature space

Angle-Based Approaches. Illustration



The spectrum of angles to pairs of points remains rather small for an outlier whereas the variance of angles is higher for border points of a cluster and very high for inner points of a cluster

Data Integration

Definition

Data integration is a process of carefully merging data from various sources into one dataset

If dataset is received from a single source, nothing is to be done for data integration

Problems in data integration:

- Inconsistency in data schemata
  Different data sources use different data models
- Inconsistency in data
  Different data sources use different data representations, data formats, contain errors, etc.

A good integration strategy ensures data is free of errors, inconsistencies, and duplication

Data Reduction

Definition

Data reduction is a process of obtaining the reduced representation of the data

Data reduction:

- Reduces memory needed to store the data
- Improves the training time
- Decreases model complexity
  Less number of features leads to simpler model that decreases the possibility of overfitting
- Improves the relevancy of the model

Data Reduction Techniques

- **Dimension reduction techniques**
  We can get done away with features that are redundant and have no appreciable affect over model's performance
  - Principal component analysis (PCA)
    Looks for a combination of features that capture well the variance of the original features
  - Random projections
    Projection to lower dimension feature space in such a way that distances between the points are nearly preserved
  - Feature agglomeration
    Applies clustering to group together features that behave similarly
- **Compression-based data reduction methods**
  Similar data can be substituted by a prototype
  - Cluster analysis
    Substitute a chunk of similar data by cluster centroid

Data Transformation Techniques

Data transformation aims to accelerate convergence of training process

- **Task-dependent data transformations**
  Logarithmic transform, transform to unitless features, etc.
- **Input normalization**
  - Scaling to interval $[-1;1]$: $x' = 2\dfrac{x_{-}x_{\min}}{x_{\max} - x_{\min}} - 1$
  - Normalization to zero mean and unit variance (z-score normalization): $x' = \dfrac{x - \bar{x}}{s}$, where $\bar{x}$ and s are mean and standard deviation of feature x
  - Whitening transformation: $x' = W(x - \bar{x})$, where W is whitening matrix, $W^T W = cov(x)^{-1}$
- **Label encoding**
  - One-hot encoding

Data Preprocessing. Overview

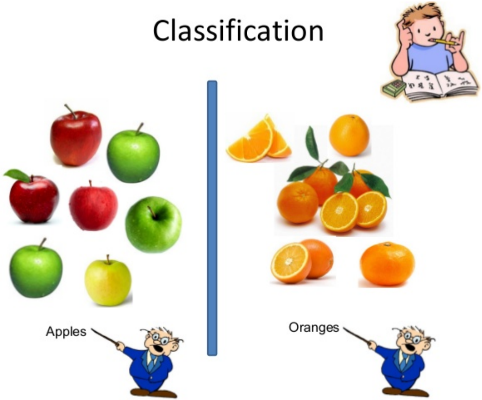
- **Data cleaning**
  - Data imputation
  - Data deduplication
  - Outliers detection
    - Model-based approaches
    - Proximity-based approaches
    - Angle-based approaches
  - Inconsistencies removal
- **Data integration**
- **Data reduction**
  - Dimension reduction techniques
  - Compression-based data reduction methods
- **Data transformation**
  - Task-dependent data transformations
  - Input normalization
  - Label encoding

Objects and Responses

$\mathscr{X}$ — instance domain

$\mathscr{Y}$ — response domain

$\mathscr{F} : \mathscr{X} \to \mathscr{Y}$ — unknown mapping (target function)



Features

$f_j : \mathscr{X} \to D_j$ — j-th feature
D_j — j-th feature domain, $\quad j = 1, ..., M$

Types of features:

- $D_j = \{0, 1\}$ — binary feature f_j
- $|D_j| < \infty$ — nominal (categorical) feature f_j
- $|D_j| < \infty$, D_j is ordered — ordinal feature f_j
- $D_j \subseteq \mathbb{R}$ — real-valued feature f_j

$x \in \mathscr{X}$ — some object from $\mathscr{X}$

$f(x) = (f_1(x), ..., f_M(x))$ — feature vector of object x

$f(x) \in D_1 \times ... \times D_M$

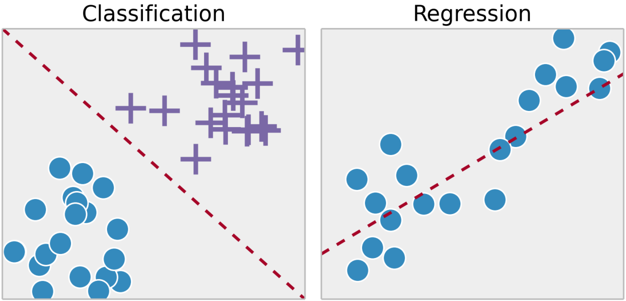
Types of Responses

Regression:

- $Y = \mathbb{R}$ or $Y = \mathbb{R}^L$

Classification:

- $Y = \{-1, 1\}$ or $Y = \{0, 1\}$ — binary classification
- $Y = \{1, ..., K\}$ — multiclass classification



Feature Engineering

Definition

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work

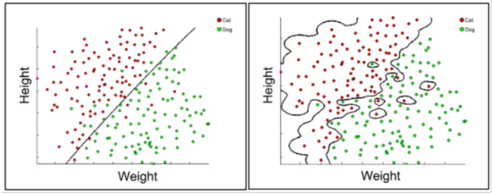
"Coming up with features is difficult, time-consuming, requires expert knowledge. "Applied machine learning" is basically feature engineering."

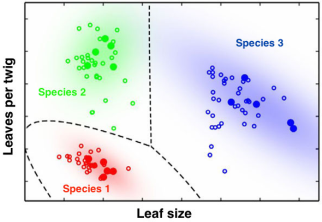
— Andrew Ng, in "Machine Learning and AI via Brain simulations"

"Much of the success of machine learning is actually success in engineering features that a learner can understand."

— Scott Locklin, in "Neglected machine learning ideas"

Domain Knowledge. Examples





The initial pick of feature is always an expression of prior knowledge

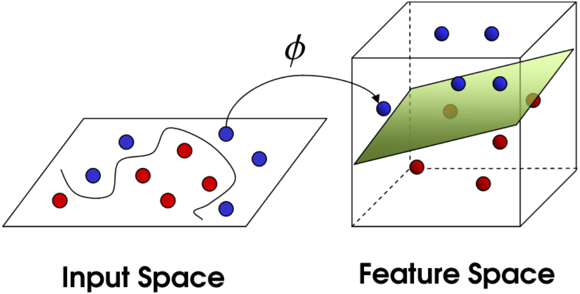
- images → pixels, contours, textures, etc.
- signal → samples, spectrograms, etc.
- time series → ticks, trends, reversals, etc.
- biological data → dna, marker sequences, genes, etc.
- text data → words, grammatical classes and relations, etc.

Importance of Feature Engineering

Feature engineering asks: **what is the best representation of the sample data to learn a solution to your problem?**

Better features means:

- Flexibility
- Simpler models
- Better results



The process of feature engineering

Step 1. Brainstorming features

Step 2. Deciding what features to create

Step 3. Creating features

Step 4. Checking how the features work with your model

Step 5. Improving the features if needed

Step 6. Go back to brainstorming/creating more features until the work is done

Feature engineering approaches:

- Feature construction
- Feature extraction
- Feature selection
- Feature learning

Feature Construction

Definition

Feature construction is the process of manual construction of new features from raw data

Feature construction is the part of feature engineering that is often talked the most about as an artform

This requires spending a lot of time with actual sample data and thinking about the underlying form of the problem

"...some machine learning projects succeed and some fail. What makes the difference? Easily the most important factor is the features used."

— Pedro Domingos, in "A Few Useful Things to Know about Machine Learning"

Feature Extraction

Definition

Feature extraction is the process of automatic construction of new features from raw data usable for machine learning algorithm

Feature extraction is related to dimensionality reduction

Feature extraction methods:

- Principal component analysis (PCA)
- Independent component analysis (ICA)
- Projection to latent structures (PLS)
- Nonlinear dimensionality reduction (manifold learning algorithms)
- Autoencoders
- ...

Feature Selection

Definition

Feature selection is the process of selecting a subset of relevant features for use in model construction

Feature selection methods are applied to extracted features

Feature selection methods:

- Filter methods
  Assign a scoring to each feature, usually univariate and consider the features independently
- Wrapper methods
  Consider the selection of a set of features as a search problem
- Embedded methods
  Learn which features best contribute to the accuracy of the model while the model is being created (e.g. LASSO, ridge regression)

Feature Learning

Definition

Feature learning (representation learning) is the process of automatic identification of features from raw data

The abstract representations are prepared automatically, but you cannot understand and leverage what has been learned, other than in a black-box manner

Feature learning methods:

- Supervised feature learning
  - Deep learning
- Unsupervised feature learning
  - Self-organizing maps (SOM)
  - Independent component analysis (ICA)

Feature Construction, Extraction, Selection and Learning

Feature construction is a process that discovers missing information about the relationships between features and augments the space of features by inferring or creating additional features

Feature extraction is a process that extracts a set of new features from the original features through some functional mapping

Feature selection is a process that chooses a subset from extracted set of features

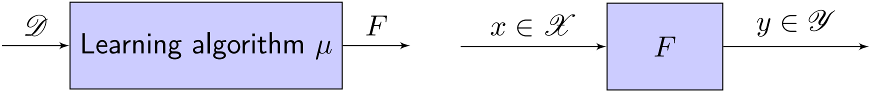
Feature learning is the process of automatic transformation of raw data into features that can be effectively exploited by model

Feature Engineering Techniques. Overview

- **Feature construction**
  - Knowledge-based approach
- **Feature extraction**
  - Dimensionality reduction techniques
- **Feature selection**
  - Filter methods
  - Wrapper methods
  - Embedded methods
- **Feature learning**
  - Supervised feature learning
  - Unsupervised feature learning

Hyper-Parameters of Learning Algorithm

A learning algorithm can be seen as a function μ taking training data $\mathscr{D}$ as input and producing as output a function F



Definition

Hyper-parameter of the learning algorithm μ is a variable to be set prior to the actual application of μ to the data $\mathscr{D}$, one that is not directly selected by the learning algorithm itself

Choosing hyper-parameter values is formally equivalent to the question of model selection, i.e., **given a family or set of learning algorithms, how to pick the most appropriate one inside the set?**

Neural Network Hyper-Parameters

- **Hyper-parameters associated with the model**
  - Architecture parameters
    - Number of layers and hidden neurons
    - Activation functions
  - Loss function
  - Regularization parameters
    - L_1 or L_2 weight decay regularization coefficient λ
    - Dropout probability p
    - Variance σ of injected noise
- **Hyper-parameters associated with the optimizer**
  - Initial learning rate α
  - Learning rate schedule parameters (or adaptive rate methods)
  - Number of training iterations T (or early stopping)
  - Method-specific parameters (momentum μ, forgetting factor ρ)
  - Initial weights (distribution and variance)
  - Mini-batch size P
- **Hyper-parameters associated with preprocessing**

Hyper-Parameter Optimization

Hyper-parameter selection can be viewed as a difficult form of learning

The training criterion for this learning is typically the error on validation sample after network's training is stopped, which is a proxy for generalization error

Evaluation of such training criterion value is a computationally expensive and time-consuming procedure

The relation between hyper-parameters and validation error can be complicated

It is possible to overfit the validation error and get optimistically biased estimators of performance when comparing many hyper-parameter configurations

Approaches to Hyper-Parameter Optimization

- Coordinate descent
  Change only one hyper-parameter at a time, always making a change from the best configuration of hyper-parameters found up to now
- Grid search
  Exhaustive search through all the combinations of hyper-parameters in grid nodes
  Advantage: fully parallelizable
  Disadvantage: it scales exponentially badly with the number of hyper-parameters
- Random sampling
  The idea is to replace the regular grid by a random sampling. Each tested hyper-parameter configuration is selected by independently sampling each hyper-parameter from a prior (typically uniform) distribution
  Advantage: many times more efficient than grid search as soon as the number of hyper-parameters grows

Measures of Classification Performance

Measures of model performance are different for classification and regression problems

For classification problems:

- Confusion matrix based measures
  — Measure the performance of given classifier h
  — Deal with different types of binary classification outcomes
  — Derived from confusion matrix
- Model-wide measures
  — Measure the performance of parametrized set of classifiers $\{h_b, b \in \mathbb{R}\}$, not of given classifier h
  — Calculate multiple confusion matrix based measures for many $b \in \mathbb{R}$
  — Measure the separability of trained classification scores

Binary Confusion Matrix

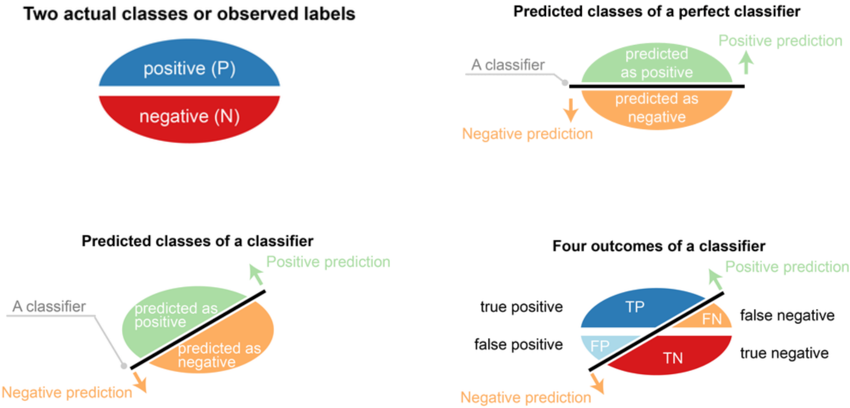
		Prediction	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

Four outcomes of classification:

- TP — True Positive
- FP — False Positive
- TN — True Negative
- FN — False Negative

TP — actually positive, are correctly included in the positive class
FP — actually negative, are incorrectly included in the positive class
TN — actually negative, are correctly included in the negative class
FN — actually positive, are incorrectly included in the negative class

Binary Confusion Matrix. Illustration



Multiclass Classification Performance

Approaches to measuring:

- Micro-averaging
  Generalization to multiclass classification

$$Perf_\mu = Perf\left(\sum_k TP_k, \sum_k FP_k, \sum_k TN_k, \sum_k FN_k\right)$$

- Macro-averaging
  Averaging of per-class measures over classes

$$Perf_M = \frac{1}{K}\sum_k Perf(TP_k, FP_k, TN_k, FN_k)$$

$Perf$ — performance measure for binary classifier
TP_k, TN_k, FP_k, FN_k — TP, TN, FP and FN with respect to k-th class: k-th class is considered as positive, rest classes as negative

All micro-averaged and macro-averaged performance measures are based on one-vs-all binary performance measures

Multiclass Confusion Matrix

		Prediction				
		Class 1	Class 2	Class 3	...	Class K
Actual	Class 1	**Accurate**				
	Class 2		**Accurate**			
	Class 3			**Accurate**		
	...				**Accurate**	
	Class K					**Accurate**

Positive: 1

		Prediction	
		Positive	**Negative**
Actual	**Positive**	TP_1	FN_1
	Negative	FP_1	TN_1

$\ldots \quad \ldots \quad \ldots$

Positive: K

		Prediction	
		Positive	**Negative**
Actual	**Positive**	TP_K	FN_K
	Negative	FP_K	TN_K

Measures of Regression Performance

- Goodness-of-fit analysis
  Coefficient of determination:

$$R^2 = 1 - \frac{D_{residual}}{D_{total}} = 1 - \frac{\sum_{i=1}^{n}(y^{(i)} - F(x^{(i)}))^2}{\sum_{i=1}^{n}(y^{(i)} - \bar{y})^2}$$

- Residual analysis
  Residual at $x^{(i)}$: $e(x^{(i)}) = y^{(i)} - F(x^{(i)})$, $i = 1, ..., n$
  - Graphical analysis
    Histogram of residuals, scatter plots of residuals versus predictors ot fitted value, etc.
  - Quantitative analysis
    Statistical tests for heteroskedasticity and autocorrelation of residuals, etc.
- Regression plot
  Scatter plot of predicted value vs. target value

Regression Plot. Example