ADA project consortium

Co-funded by the
Erasmus+ Programme
of the European Union

METHODOLOGY OF TEACHING DATA SCIENCE

A GUIDE

Belgrade, 2021

ADA project consortium

METHODOLOGY OF TEACHING DATA SCIENCE

A guide

Belgrade, 2021

This booklet is published as a result of the project **Advanced Data Analytics in Business (ADA)** – EACEA 598829-EPP-1-2018-1-RS-EPPKA2-CBHE-JP, co-funded by the Erasmus+ programme of the European Union*.

Methodology of teaching data science – a guide

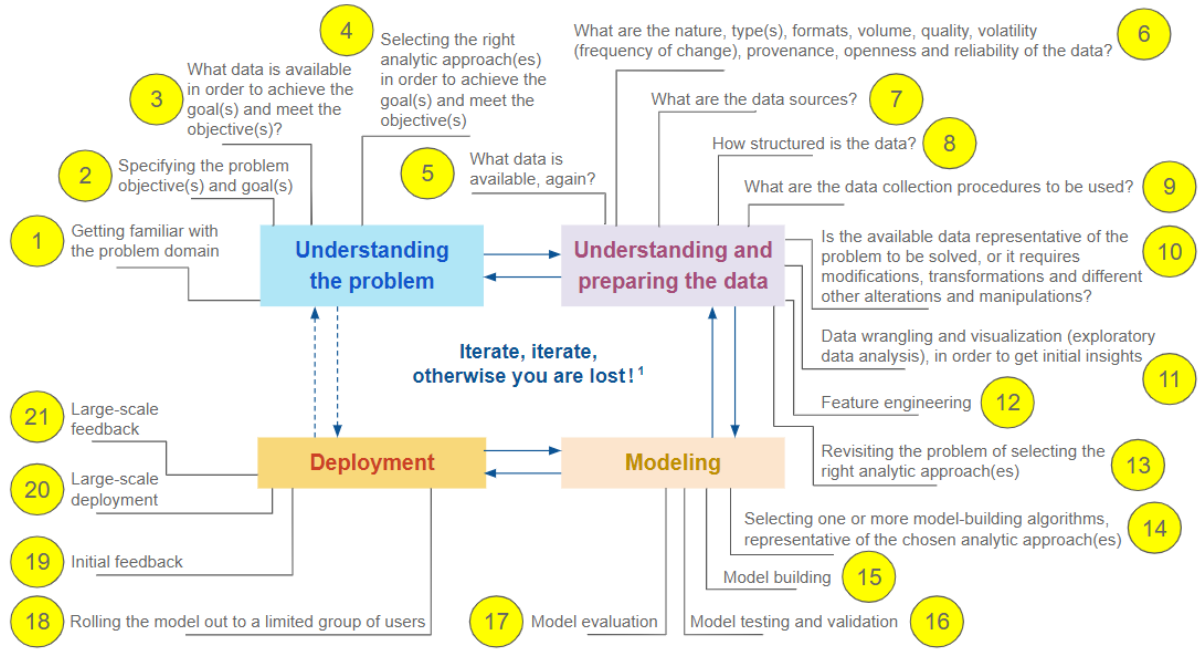
Teaching data science (DS) in higher-education institutions starts from the fact that all sectors and domains of human activities – businesses, services, healthcare, industry, education, science, research, government, transport, communication, environment, sports, law, sports, entertainment, even arts – entail large and rich volumes of data. This data represents imprints of problems of great importance to our society.

Understanding, studying, collecting, curating, mining and analyzing data originating from different sources requires a lot of effort, dedication, expertise and sensemaking. Tackling a DS problem in the real world assumes taking the steps involved and applying them with rigor, patience and discipline. It is the job of DS teachers to explain these steps and the processes involved in a methodical, well-defined way.

The big picture

Both teaching DS and working on a DS problem in the real world happen in several stages. The golden rule to bring right upfront is: all these stages are *iterative*. Moreover, looping back from one stage (or from any of its steps) to the previous one (or to any of its steps) is absolutely common.

The stages are interdependent and are organized in a specific workflow.



Understanding the problem

- *Getting familiar with the problem domain*. This is the very first step in working on every DS problem. It is unlikely that a DS specialist will be completely familiar with the domain of the problem at hand. They have to learn at least the domain basics before going any further. At the later steps and stages, it is highly desirable to have an expert in the domain as an occasional consultant.

[1] Inspired by the famous saying of Pina Bausch: *Dance, dance, otherwise we are lost!*

- *Specifying the problem objective(s) and goal(s)*. It is absolutely necessary that these are defined clearly, in a comprehensive and explainable way. DS specialists have to know as exactly as possible what they will be looking for in the project at hand.[2]
- *What data is available in order to achieve the goal(s) and meet the objective(s)?* All DS is about analyzing data, so some data must be available. This usually entails concerns about data security and privacy, data quality and accuracy, and so on.
- *Selecting the right analytic approach(es) in order to achieve the goal(s) and meet the objective(s)*. In real-world problems, predictive analytics (various forms of classification) is the dominant category of DS problems. It means some form of supervised learning. However, unsupervised learning problems (clustering, anomaly detection, association rules and the like) also get their share.

Understanding and preparing the data

- *What data is available, again?* Once the problem goal(s), objective(s) and analytic approaches to be used are clearly specified, the data availability should be elaborated. It turns out sometimes that some data is not that available as it seems at the first glance.
- *What are the nature, type(s), formats, volume, quality, volatility (frequency of change), provenance, openness and reliability of the data?* The nature of some data can be very volatile, hence the validity and the overall quality of data can be problematic if snapshots of data taken for analysis expire soon. There will typically be some erroneous data, some missing data, as well as some data that are either not easy to obtain or they are not open due to different reasons.
- *What are the data sources?* The data coming from databases can be straightforward to use in analyses (after the necessary data cleansing). The data coming from the Web often require more preprocessing before they can be used. The data can also come from different sensory equipment, so snapshots of data need to be used.
- *How structured is the data?* If the data comes from databases, it is already at least partially structured and is easier to convert into the datasets to be used in the analyses. The data coming from the Web is usually unstructured and requires more preprocessing. If the data comes from sensors, converting it to datasets often involves people trained in working with the corresponding equipment. In all cases, a good knowledge and command of relational database techniques and SQL is assumed.
- *What are the data collection procedures to be used?* Many DS projects get stuck in the very beginning because of the data collection. It is not only a technical problem, nor it is only an administrative problem. The data can be sensitive for various reasons, and specifying the data collection procedures and protocols early in the project is highly recommended.
- *Is the available data representative of the problem to be solved, or it requires modifications, transformations and different other alterations and manipulations?* Actually, it is very often the latter. Thus this step and the next two steps usually constitute a subproblem that requires a number of iterations before the data is understood to a sufficient extent.
- *Data wrangling and visualization (exploratory data analysis), in order to get initial insights*. In order to start working with the data, it has to be cleaned, structured and potentially enriched. This is called data wrangling. It should result in high-quality datasets, without missing and erroneous data, where the data is represented in formats suitable for analysis. This is usually a time-consuming, iterative process. As part of that process, data visualization tools and techniques come handy for data exploration in order to get more familiar with the problem at hand.
- *Feature engineering*. Raw data can be useful in conducting analyses, but transforming it into features that better represent the underlying problem can notably alleviate and improve the modeling process and improve the model accuracy when it is applied to new data. Here the

[2] Yes, it *can* happen that they will discover other things along the way, but it does not happen often.

word "feature" denotes an attribute that is useful or meaningful to the problem at hand. Thus useful/meaningful data should be extracted from raw data for further analysis, using different transformation, construction and modification techniques. This usually goes hand-in-hand with the dimensionality reduction, in a way – from a number of raw attributes in the dataset, the analyst should extract a few really useful ones.

- *Revisiting the problem of selecting the right analytic approach(es)*. Getting to know the data better through the processes of data cleaning, data wrangling, exploratory analysis and feature engineering often brings new insights that can significantly affect the choice of analytical approaches for the modeling stage.

Modeling

- *Selecting one or more model-building algorithms, representative of the chosen analytic approach(es)*. Whatever the selected analytical approach(es) for modeling the problem at hand, there will typically be several candidate algorithms to build the model. All of them have advantages and disadvantages. It is a good idea to try multiple algorithms if the problem nature does not clearly indicate a single algorithm. Likewise, the modeling tools and relevant software libraries used in working on the problem may only provide support for certain algorithms, not for all of them.
- *Model building*. This is typically not a demanding job, compared to the data preparation phase. The tools used for constructing the models often include useful software libraries with well-defined APIs. There are also automated model-building tools, where the job of the DS specialist is to describe the data and the problem goals in detail, and then let the tool build the model and evaluate it. However, these tools may not be available free of charge. Another problem here is that, depending on the size of the datasets used for building the model, powerful hardware may be a must. In such cases, PaaS (Platform as a Service) solutions can be a good idea.
- *Model testing and validation*. Models are built using train data. Then they should be validated using just a part of the test data, with the idea to select the best values of the model parameters and compare the models built using different algorithms, techniques, software libraries, etc.
- *Model evaluation*. The best candidate model is then tested on the remaining test data, in order to calculate its expected accuracy, precision and other statistics used to describe the model quality. These statistics can vary from one modeling problem to another, but serve the same purpose of getting the feeling of the expected quality of the model, once it is applied on unseen data.

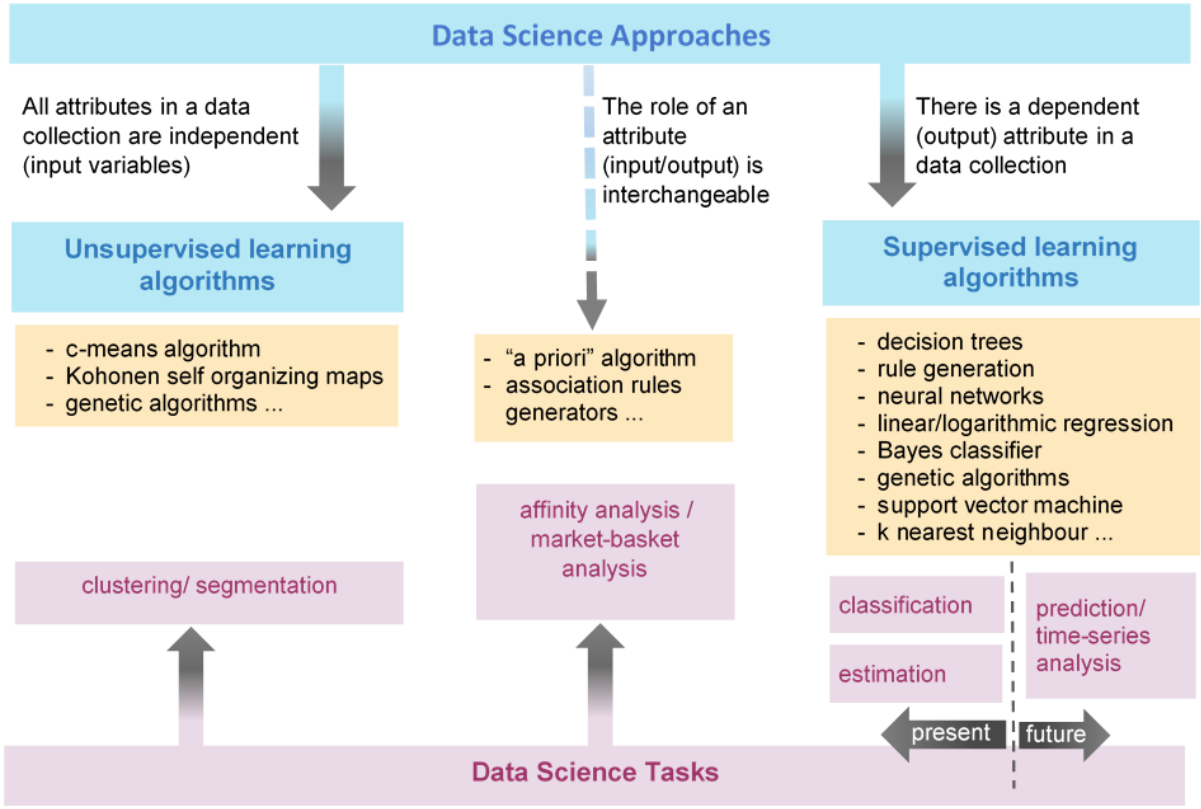
Deployment

- *Rolling the model out to a limited group of users*. Once the model is tested and evaluated by the developers, it is a good idea to run it on unseen real-world data and involve domain experts. The purpose of this step is reality check – eliciting the opinions of unbiased, but highly qualified users coming from the problem domain.
- *Initial feedback*. These users should encourage the large-scale application of the model, or possibly express some doubts and indicate areas for improvement. It can happen that the model performs well only with the data used for testing, or it can happen that it is difficult to use in different cases. In such cases, valuable lessons can be learned about how the model should be improved.
- *Large-scale deployment*. This means rolling out the model to independent users who run different businesses where the model can be useful. Note that this step can be a major caveat in the entire process – mode often than not, a model developed for one domain can prove completely useless even in highly similar domains.

- *Large-scale feedback*. It is a good idea to provide model discussion options through relevant forums and social networks. The comments and suggestions that way are invaluable for model improvement and maintenance!

Selecting the right approach and algorithm for building the model

The whole DS process is about learning new information and knowledge from data. Algorithms developed for this purpose support either *supervised* or *unsupervised learning*. The easiest way to differentiate between the two, and select the right algorithm, is to answer the question: Is there any variable (attribute) in the dataset at hand (a data collection) that depends on other variables? If the answer is negative, the learning is unsupervised. If the answer is affirmative, the learning scheme is supervised. Between supervised and unsupervised learning there is an approach known as *affinity analysis*, with the market-basket analysis task as a most frequent application.



Dominant factors in selecting the approach and the algorithm to be used

A variety of DS methods and techniques was developed in the last few decades and the main question a data scientist is facing is which one to use for a particular task. It is not simply a matter of selecting the best one for all purposes. Instead, one must consider the particular environment and features of the data.

What does one want to do with the data?

Algorithms implementing unsupervised learning imply segmentation: we can divide the data collection into segments of similar entities, the process also known as *clustering*.

Most common supervised learning algorithms classify the data collection (divide it into subsets / categories / classes, all being synonyms) according to the nominal values of the dependent variable.

When the dependent variable is numerical, the task of modelling is either *evaluation* or *prediction*. If the data contain a time dimension, the problem at hand is *time-series analysis*, and best algorithms are linear or logistic regression, and feed forward back propagation networks, while decision trees, genetic learning, Bayes classifier and production rules show less than optimal results.

The output variable(s) of a predictive model can be categorical, as well. It is not easy to differentiate prediction from classification or estimation, but the purpose of a predictive model is to determine future outcome rather than the current behaviour.

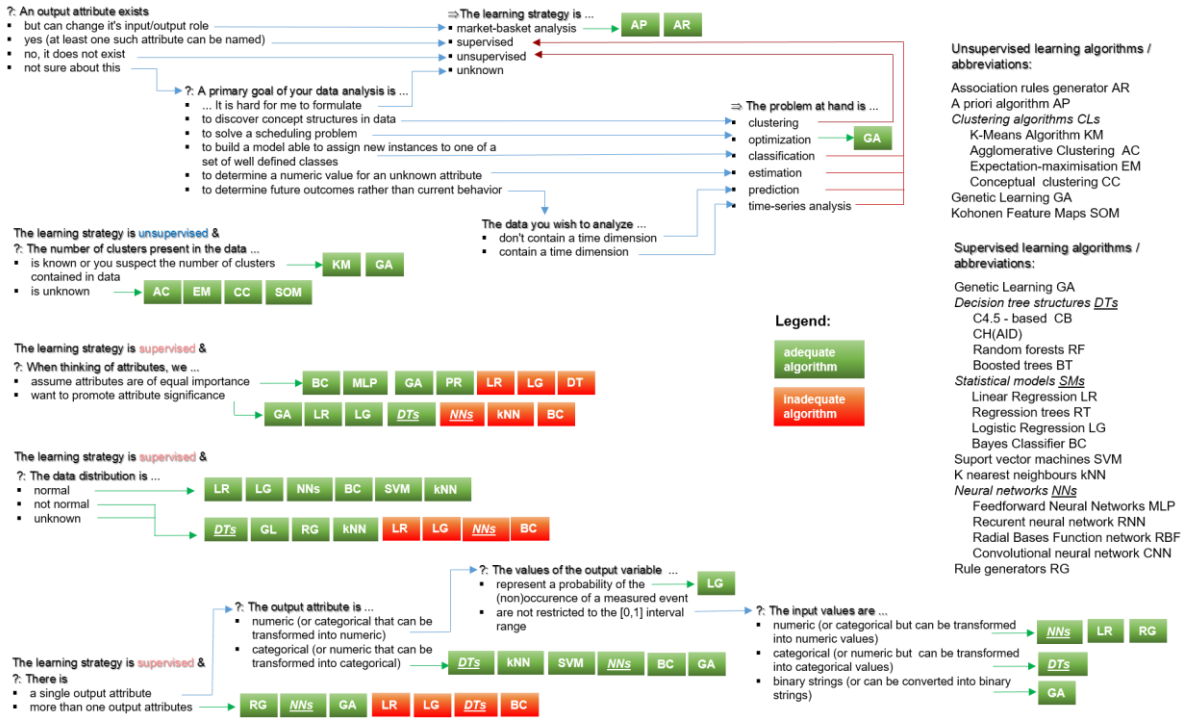
Goal	Task	Output
Segment the data collection /find natural groupings in data	clustering	-
Assign a new entity/observation to one of the known categories/classes	classification	category/class
For known numeric inputs determine the value of an unknown output variable	estimation	numeric value
Foresee the future output value from the input data	prediction	category or numeric value
Determine the sequential value in time-dependent data	Time-series analysis	numeric value
Find associated items in a data collection	market-basket analysis	-

The size, quality, and nature of the data

The saying "Garbage in – garbage out" reflects an unfortunate reality in DS. Missing data, erroneous data, strongly correlated data, outliers, all have to be cleaned out, as they affect the modelling power of all algorithms. However, some algorithms (like feed-forward neural networks trained with error back propagation, and Bayes classifier) can handle missing values or noisy data efficiently.

Some algorithms demand data transformation. Clustering algorithms work with numeric data, due to the calculation of a distance between cluster members, so non-numeric data should be substituted by numeric equivalents. The same situation is with regression algorithms, and neural networks, where in addition to type conversions, data have to be normalized (scaled usually to the interval [0,1]). On the other hand, decision trees and Bayes classifiers can be built using both numeric and categorical input attributes.

There are some heuristics that help distinguishing DS algorithms and identifying those that best match the given quality and nature of the data.



The available computational time

The increasing amount of data is no compensation for the lack of processing time and urge for decision making. If computational time is an issue, then faster and more accurate learning algorithms are preferred, such as decision trees, rule generators, Bayes classifier, linear and logistic regression, while the same performance can't be expected from genetic learning or neural networks.

The urgency of the task

Model building can take a long time (days, weeks, months...), depending on the size of the dataset. In case when we cannot afford that long time and we have no suitable hardware to speed it up, alternative approaches must be sought (e.g., dataset reduction, or renting a PaaS, etc.).

Statistical models

Common statistical DS techniques are linear regression and regression trees, logistic regression, Bayes classifier and clustering techniques.

Statistical regression is a supervised technique that generalizes a set of numeric data by creating a mathematical equation relating one or more input attributes to a single output attribute. *Linear regression* attempts to model the variation in a dependent variable as a linear combination of one or more independent variables. *Logistic regression* is a nonlinear regression technique that associates a conditional probability value with each entity. *Regression trees* take the form of *decision trees* where the leaf nodes of the tree are numeric (the average of output attribute values for all instances passing through the tree to the leaf node position) rather than categorical values.

Bayes classifier builds a classification model assuming all input attributes to be of equal importance and independent of one another.

Clustering algorithms are appropriate for dataset segmentation. They differ in steps undertaken to achieve their goal. Agglomerative clustering is applied to partition data instances into disjoint clusters. Conceptual clustering builds a concept hierarchy to partition data instances. Expectation maximization algorithm uses a statistical parameter adjustment technique to cluster data instances.

The best known clustering algorithm is k-means algorithm. It assigns k centers to represent the clustering of N instances (k<N) and each of the k clusters is the mean of its assigned instances. *Support Vector Machines* (*SVM*) are based on the Statistical Learning Theory concept of decision planes that define decision boundaries – separate objects having different class memberships. Often more complex structures are needed to make an optimal separation, so we use polynomial, sigmoid or RBF (Radial Basis Functions) kernel functions.

Nearest neighbour method is a very specific approach, as it stores instances in a classification table together with their known class, rather than building a generalized model of the data. K- nearest neighbour method classifies a new instance with the most common class of its k-nearest neighbours.

Both clustering techniques and k-nearest neighbour methods imply some metrics for measuring instances similarity (Euclidean distance, Jaccard similarity, cosine similarity, etc.)

When should they be used, and which ones?

Regression models are applicable to numeric data.

Linear regression is an appropriate DS tool when the relationship between the dependent and independent variables is nearly linear. The data model in a form of a linear equation is used to predict numerical output values, but also for classification – to determine the combination of input values that best indicates the affiliation of entities to a particular category.

Regression trees are more accurate than linear regression equations when the data to be modelled is nonlinear. They are combined with linear regression to form what are known as model trees. The leafs of a model tree are linear regression equations that fit better to a nonlinear nature of data.

Logistic regression can be used to build supervised learner models for datasets having a binary outcome. If as a result of modeling we expect the probability that something will (not) happen, logistic regression is a good choice – it will translate the results into a range of [0,1].

Bayes classifier builds classification and prediction models for data collections having categorical, numeric, or both data types.

A widespread application of *agglomerative clustering* is its use as a prelude to other clustering techniques.

SVM perform classification tasks by constructing hyperplanes in a multidimensional space that separates cases of different class labels. SVM also support regression and can handle multiple continuous and categorical variables.

Nearest neighbour algorithm offers an alternative approach to classification problems.

What are the advantages of these models?

Statistical methods in general have a solid theoretical background, they are reliable and data scientists are familiar with underlying concepts from high school math.

Linear regression models are easy to understand and deploy.

Bayes classifier offers a simple, yet powerful supervised classification technique. It is easy and fast to predict the class of entities in the test data set. It also performs well in a multi class environment. It can be applied to data collections containing a wealth of missing attribute values – they are simply ignored. When assumption of independence holds, a Naive Bayes classifier performs better compared to other models like logistic regression and you need less training data.

Conceptual clustering systems are particularly appealing because the trees they form have been shown to consistently determine psychologically prefered levels in human classification hierarchies. Also these systems well explain their behaviour.

Expectation-maximization implements a statistical model that is guaranteed to converge to a maximum likelihood score (though this maximum may not be global).

What are the deficiencies of these models?

Regression models require a dichotomous dependent variable (for e.g. presence/absence of occurrence). They are sensitive to the appearance of outliers in the data. Another requirement is that no high correlations exist between predictors.

Linear regression is a poor choice when the outcome is binary. The problem lies in the fact that the value restriction placed on the dependent variable is not observed by the regression equation, i.e. values of the dependent variable are unbounded in both the positive and negative direction. Using logistic regression instead is a solution.

Limitation of *Naive Bayes* algorithm is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent.

Bayes classifier performs not so well in case of numerical variable(s) compared to categorical input variables. For numerical variables, normal distribution is assumed (bell curve, which is a strong assumption).

If a categorical variable has a category (in the test data set), which was not observed in a training data set, then Bayes model will assign it a zero probability and will be unable to make a prediction. To solve this, we can use a simple smoothing technique called *Laplace estimation*.

Naive Bayes is also known as a bad estimator, so the probability outputs are not to be taken too seriously.

Problems associated with *k-nearest neighbour* approach are twofold: first, computation times will be a problem when the classification table contains thousands or million of records; second, the approach has no way of differentiating between relevant and irrelevant attributes.

What are the caveats of using these models?

Statistical data mining techniques correspond to specific learning strategies. Many of them have limiting assumptions about the nature of the input and output data. Predictive statistical models should be used with caution – the further we go into the future, the greater the prediction error.

Lack of explanation about the nature of the data models leaves us responsible for much of the interpretation about what has been found.

Decision tree structures

Decision tree is a simple structure where nonterminal nodes represent tests on one or more attributes and terminal nodes reflect decision outcomes. The main goal of tree building is to minimize the number of tree levels and tree nodes, thereby maximizing data generalization. Many variations of the underlying idea exist. Improved versions of the well-known Quinlan's C4.5 algorithm are implemented in many commercial DS tools (the Java update, J48, can be found in Weka).

CHAID algorithm (Chi-square Automatic Interaction Detector) uses multiway splits instead of binary splits, where more than two splits can occur from a single parent node.

The Random Forest(s) algorithm trains a number of trees on slightly different subsets of data, in which a case is added to each subset containing random selections from the range of each variable. This group of trees is similar to an ensemble. Each decision tree in an ensemble votes for the classification of each input case.

The boosted trees algorithm (adaptation of the Stochastic Gradient Boosting technique) builds a model through successive iterations. Firstly, decision trees for each class of the categorical dependent variable are built. Secondly, a very small tree, the so-called base learner is built using just a subsample of the training data. Predicted values of the training set model and the base learner model are used to calculate the residuals. A prediction update factor is calculated from the residuals and applied to generate the final prediction for the second tree. This process is continued until the sum of residuals declines below a threshold value.

When should they be used, and which ones?

Decision trees are a popular structure for supervised learning.

CHAID is favoured in applications involving market segmentation studies. The algorithm can predict numeric and categorical output variables, and therefore it can be applied to both classification and estimation problems.

The Random Forest(s) algorithm is also very efficient for classification.

Boosted Trees models are among the most popular and effective machine learning approaches for both regression and classification.

What are the advantages of these models?

Decision trees are easy for us to understand, can be transformed into rules, and generalizes the data well. They make no prior assumptions about the nature of the data and can build models with datasets containing numeric as well as categorical data.

The Random Forest(s) algorithm can handle very large datasets, missing data, as well as unbalanced data sets. They provide estimates of attribute importance. Above all, the algorithm is fast and very accurate.

Boosted Trees models can achieve impressive performance with minimal hyperparameter tuning.

What are the deficiencies of these models?

Classical decision trees are unstable, meaning that small variations in the training data can result in different attribute selection in each choice point within the tree. The effect can be significant as attribute choices affect all descendent subtrees.

Output attributes have to be categorical, and multiple output attributes are not allowed. Trees created from numeric data can be quite complex as attribute splits for numeric data are typically binary.

What are the caveats of using these models?

In real-world problems, decision tree structures can become very large and less comprehensible to the user.

Neural networks

Neural networks offer a mathematical model that attempts to mimic the human brain. In general, knowledge is represented as a layered set of interconnected processor nodes, similarly as neurons of the brain. Each node has a weighted connection to several other nodes in the adjacent layers. Individual nodes take the input received from connected nodes and use the weights together with a simple function (called activation function) to compute output values.

Learning is accomplished by modifying network connection weights, while a set of input instances is repeatedly passed through the network. Once trained, an unknown instance passing through the network is assigned a value (or a class, or a cluster) according to the value(s) seen at the output layer. Depending on the application, the output layer of the neural network may contain one or several nodes.

Learning continues until a specific termination condition is satisfied – convergence of the network to a minimal total error value, a specific time criterion, or a maximum number of iteration.

Neural Networks can be classified into different types (feedforward, recurrent, radial basis function, Kohonen self organizing, and modular neural network), depending on the specificity of their architectural design and learning strategy.

In a *feedforward neural network*, information flows in just one direction from input to output layer (via hidden nodes if any. Nodes are also called “perceptrons”, hence the name Multilayer Perceptron).

In recurrent neural network (RNN) connections between units form a directed cycle. The output of a layer becomes the input to the next layer, which is typically the only layer in the network, thus the output of the layer becomes an input to itself forming a feedback loop. This allows the network to have memory about the previous states and use that to influence the current output.

The hidden layer of a *Radial Basis Function neural network* includes a radial basis function (implemented as a gaussian function) and each node represents a cluster center. The network learns to designate the input to a center and the output layer combines the outputs of the radial basis function and weight parameters to perform its task.

Kohonen self-organizing neural network consists of fully connected input and output layers. The output layer is organized as a two-dimensional grid. The Euclidean distances between the input data and each output layer node with respect to the weights are calculated and the weights of the closest node to the input data and its neighbor nodes are updated to bring them closer to the input data. The weights represent the position of instances in the output layer node.

Modular neural network breaks down a large network into smaller independent neural network modules. The smaller networks perform specific tasks which are later combined as part of a single output of the entire network.

Deep learning is an advanced machine learning technique that makes use of extremely sophisticated neural networks. It is called “deep” because the models generated are significantly more complex or deep than traditional neural networks. Deep learning networks do not necessarily need structured/labeled data for a classification task - the input (the data of images) is sent through different layers of the network, hierarchically defining specific features of images.

Convolutional neural network (CNN) is a class of deep neural networks. CNNs are a variant (regularized versions) of multilayer perceptrons, meaning some form of magnitude measurement of weights is added to the loss function.

When should they be used, and which ones?

Neural networks can be used for all types of learning tasks.

For classification and prediction we usually use *feed forward networks* trained with error back propagation algorithm or genetic learning (an architecture also known as multilayer perceptron). They can be used for applications that require a time element to be included in the data.

Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. This makes them applicable to tasks that require processing sequence of time phased input data (speech or handwriting recognition, frame-by-frame video classification, etc.)

Radial basis function neural network is used in classification, function approximation, time series prediction problems, etc.

For unsupervised clustering problems we use *Kohonen networks (self-organizing maps*).

Deep learning should be applied when data is problematic for traditional analytic techniques. Specifically, wide data (data sets with a large number of data points or attributes in every record compared to the number of records - such as image data) and highly correlated data (data with similar or closely related values - such as genomes data) can present problems for traditional analytic methods.

CNNs have applications in image and video recognition, recommender systems, image classification or segmentation (especially medical image analysis), natural language processing, brain-computer interfaces, and time series analysis (in the financial domain).

What are the advantages of these models?

Neural networks can be used for both supervised and unsupervised tasks.

Neural networks perform well in many industries. For e.g. they predict failure in order to prolong operation time; forecast foreign exchange rates, interest payment rates and share prices; predict company bankruptcies, predict risk, etc.

Neural networks perform well with missing or noisy data. The activation function smooths out the variations of input values caused by outliers or random errors.

Neural networks can predict both numeric and categorical outputs. However, categorical output data conversion can be a challenging task.

Deep learning has significantly improved our ability to understand and analyze image, sound and video. This has been made possible by major advances in machine learning research as well as vast increases in both available data and massive computing power.

CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage of CNNs.

What are the deficiencies of these models?

Modelling with neural networks is known to be a time-consuming, laborious endeavour, without guarantees that interesting and potentially useful patterns will be revealed.

Many parameters influence the neural network model quality. We ought to decide which input attributes to use, which data representation scheme is best for the output, which termination criteria is best, how to configure and finetune the configuration, etc.

Neural networks are prone to overfitting. If we are continuously searching for useful knowledge hidden in data, some ”patterns“ will eventually emerge, but they may be just a chance occurrence in the data. Such patterns are not generalising – predicting well for instances that we have yet not observed. Overtrained neural networks achieve excellent results on a training dataset, but a poor one on a test set.

There is an evident lack of explanation power of a neural network model. Due to their multilayer nonlinear structure, neural networks are non-transparent: the obtained results can hardly be understood by the user (outputs are often not traceable and not recalculate-able by humans). As the connection between input and output cannot be represented in a simplified way, we refer to the inner structure of the network and learning processes as a black box. If explanations on what was learned are important, neural networks are a poor choice.

The explainability of a neural network is decreasing with an increasing results accuracy, and the results accuracy is growing with more complex models like deep neural networks. *In deep learning models* thousands (and even billions) of rules or parameters can define the model. As a result, the exact internal processing pathways are a black box even to the data scientist! One should not confuse black box processing with blind faith – the model must undergo cross-validation and auditing to avoid artificially engineering a deceptively accurate response (aka overfitting).

Deep learning requires much more data for model building than their predecessors or traditional machine learning algorithms.

What are the caveats of using these models?

Building data models with neural networks demand large numbers of training instances, and statistically representative data both in the training and test sets. If these conditions are not met, other algorithms should be considered for modelling.

As there is no unique parameters setting for neural network architectures, one should be prepared to experiment with a large number of data models in a trial-and-error manner.

Note, however, that in spite of a great popularity of deep learning, CNN and RNN models, many of them require expensive hardware in order to build them with sufficient accuracy if the datasets are large (and they often *are* large). This hardware is necessary in order to reduce model-learning time (which is often measured by days, weeks, or even months). Thus it is important to stress that studying thoroughly the nature of the data before opting for deep learning over more traditional techniques. Note that in practice *many* DS problems can be solved *without* using deep learning approaches.

Both machine learning and deep learning are not actually simultaneously applicable to most cases. Deep learning is best seen as a supplement, not a replacement, for traditional analytic methods.

Text mining and text analysis

Text is everywhere: all communication between people is “coded” in media as text, user-generated content and interaction on electronic media usually takes the form of text. In principle, text is just another form of data. If we want to apply the many DS algorithms that we have at our disposal to mine free text, we must engineer the text data representation to match the algorithms.

Specifics of text mining and text analysis

Textual data is specific in many ways. Text is unstructured – words can have varying lengths, text fields can have varying number of words and a meaningful word order. Text data is relatively “dirty” –

people write ungrammatically, misspell words, run words together, abbreviate unpredictably, and punctuate randomly. Text may contain synonyms and homographs, and the meaning of terms is domain- and context-dependent. For these reasons, text must undergo a good amount of preprocessing (normalization, stemming, stopwords removal) and representation in a feature-value matrix form before it can be used as input to a data mining algorithm.

When should specific approaches and algorithms be used, and which ones?

Text models created directly from words appearing in documents

Basic bag-of-words representation is usually the first choice of data scientists and works surprisingly well on a variety of tasks (such as clustering and classification of documents). When particular phrases in text are significant but their component words may not be, we use the so-called *n*-grams representation – each document is represented by its words, pairs of adjacent words, …, sequences of *n* adjacent words.

Sometimes we need more sophistication – we want to be able to recognize common named entities in documents (extract phrases annotated with terms like `person` or `organization`). Named entity extractors have to be trained on a large corpus, or manually coded with extensive knowledge of such names, sometimes in particular areas of expertise (industry, government, pop culture, etc.).

Topic models

Because of the complexity of language and documents, sometimes we want an additional layer between the text contained in a document and the model, the so called topic layer. Therefore, instead of words in a document being used directly by the final classifier, the words map to one or more devised topics. These topics are, in advance, separately modelled in a corpus. Topic layer approach is clustering of terms (preprocessed words). Terms associated with the topic, and also any term weights, are learned by the topic modelling process.

What are the advantages of these models?

Learning *direct models* is relatively simple and efficient.

Topic models find latent topics in text: words are mapped to (unobserved) topics, and topics map to documents. They can yield better performance than direct models. In addition, the latent information is often interesting and useful in its own right.

What are the deficiencies of these models?

Direct models are not always an optimal solution.

Topic models are more complex and more expensive to learn. In the topic modelling process, as with clusters, topics emerge from statistical regularities in the data and they are not guaranteed to correspond to topics familiar to people.

Models generated from textual data are highly context- and language-sensitive, so wide cross-industry application and deployment of developed models over different languages is not possible.

What are the caveats of using these models?

Dealing with text requires dedicated preprocessing steps and sometimes specific expertise on the part of the DS team.

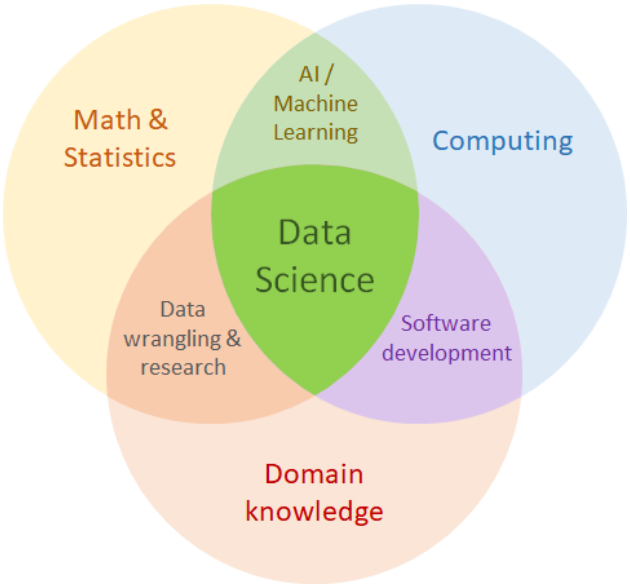
Resources for teachers

Different tools, frameworks and modeling environments are part of DS methodology. Teaching DS assumes introducing these resources to the students/trainees. Some of these resources are

essential, regardless of the level (beginner, intermediate, advanced); others are case- or domain-specific and can be included/excluded from a DS course topics, depending on the desired course outcomes.

Must-teach topics and related resources

It is widely accepted that DS lies in the intersection of math & statistics, computing and domain knowledge. When teaching (as well as practicing) DS, most essential and widely applicable and used resources come from math & statistics and computing.



Mathematical foundations and related resources

Mathematics is a foundational field of DS, having in mind that mathematical concepts and methods are important to identify patterns and assist in creating algorithms. Learning theoretical background of DS, as well as successful design and implementation of DS methods, requires the knowledge of several topics from:

- linear algebra
- calculus
- graph theory
- optimization

Linear algebra is required in data analysis and machine learning to understand how algorithms actually work. It is necessary to learn fundamentals of working with data in vector and matrix form, acquire skills for solving systems of linear algebraic equations and find the basic matrix decompositions and general understanding of their applicability.

Calculus in DS is mostly used to formulate the functions used to train algorithms to reach their objective. The topics to be covered include: functional mappings (single and multivariate functions), differentiability, limits (case of sequences, functions), integration, sequences, basics of ordinary and partial differential equations, thus sequentially building up a base for the basic optimization methods.

Graph is a mathematical structure that is the most relevant for DS. Graphs are used to model various data structures and interactions, and to study pairwise relationships between objects and entities. Basic knowledge of graph theory is required, such as types of graphs, properties, algorithms on

graphs, graph metrics, etc. Graph DS libraries, such as GDSL, NetworkX, Goblin, and others are available.

Optimization methods are powerful techniques mostly used in machine learning, as almost every ML algorithm involves an optimization problem: how to minimize some kind of estimation error subject to various constraints. The topics related to optimization should include: mathematical modelling (how to formulate a problem), linear and integer programming (simplex algorithm, Branch and Bound, Branch and Cut, Branch and Price), Nonlinear programming (gradient-based methods, Newton method and variants, projection methods,…), heuristic and metaheuristic methods (local-search based heuristics, genetic algorithms, particle swarm optimization,…). State-of-the art exact solvers such as CPLEX and LINGO are available to the academic community free of charge.

Statistics and related resources

When conducting basic and more advanced statistical analysis, the SPSS software (Statistical Package for Social Sciences), software environment R and the accompanying R studio are commonly used.

The basic suggested literature for mastering and getting to know how to work in SPSS is the SPSS survival manual. In SPSS, basic data manipulation can be easily done (option Data and Transform), alongside parametric and nonparametric tests (Compare means and Nonparametric tests), and linear regression (Regression - Linear). More advanced multivariate analysis, such as factor analysis (Dimension reduction), clustering (Classify), survival analysis (Survival) and logistics regression (Regression - Binary Logistics) can be performed as well.

The suggested literature for mastering data analysis in R is R for data science. The basic R packages used to conduct statistical analysis are:

- visualisation and nonparametric tests: coin, gmodels, ggplot2
- clustering: cluster, fpc, NbClust, biclust
- factor analysis: REdaS, car, Psych

These packages are sufficient for data preparation, modelling, hypothesis testing, as well as graphical presentation of results.

Programming languages, tools and libraries

Most DS programming nowadays is done using Python or R. In addition, since most of machine learning in DS is supervised learning with datasets originating from relational databases, a good command of SQL is a must.

Python. In practice, certain modules and packages from The Python Standard Library (like statistics, random, itertools, json, timeit, urllib, sys and the like) provide solid foundations for handling routine tasks in data analysis. There are numerous tutorials on the Web related to these modules and packages.

However, most DS problems require using specific packages and libraries beyond The Python Standard Library. Scrappy and BeautifulSoup are extremely useful for building Web crawlers and scraping tools to retrieve structured data from the Web. Numpy and Scipy are essential for working with arrays, using linear algebra, and applying advanced statistics and optimization. Pandas is an absolute must for working with structured datasets (dataframes). Scikit-Learn set of packages is an industry standard for various machine learning tasks. Last but not least, Matplotlib and Seaborn are powerful libraries for data visualization in Python.

In addition, there are several essential Python frameworks for working with neural networks (see the dedicated subsection below).

When it comes to data analysis integrated tools and environments, Jupyter Notebook is widely accepted as the gold standard. Google Colab (Colaboratory) is a similar cloud-based tool. For a more automated approach, Google's Cloud AutoML enables training high-quality custom machine learning models with minimal effort and machine learning expertise. It relies on Google's powerful infrastructure. However, it is not free.

R. Often used by statisticians and in research, R is still an important DS language. The R libraries typically used for statistical analysis are listed in the previous subsection. Its most important libraries for data scientists include ggplot2 for data visualization, data.table for working with large amounts of data, dplyr for different sorts of data manipulation, and mlr3, XGBoost and caret for different machine learning tasks.

As for the R programming and analysis environment, the most widely used one is RStudio.

Neural networks and deep learning

Neural network frameworks enable quick and easy development of neural network models, and hide implementation details and complexity.

The most popular neural network/deep learning framework is Tensorflow, developed by Google. The second one is PyTorch, which has been developed by Facebook. Both provide a similar set of basic features. PyTorch provides more flexibility for experiments and is more suitable for neural network research. Both frameworks have Python API, support advanced neural network architectures, GPU and distributed execution.

Keras is a high level neural network API which is available on top of Tensorflow, and makes it easy for beginners to start coding neural networks on the layer level, without the need to understand the details of tensor operations and optimizations.

Tensorflow has very good documentation and tutorials, available at https://www.tensorflow.org/tutorials. Each tutorial includes a Colaboratory (or just Colab) notebook, which enables running the example in the cloud, using a Python-based notebook. This is very suitable for teaching purposes, since it doesn't require students to install and configure the complex Tensorflow stack that includes a number of different components and dependencies, and can be tricky to set up.

Text mining and text analysis

These resources are abundant, and one can categorize them as Python- or R-based resources, or as general NLP models, or by level (introductory, intermediate, advanced) and the like.

Python-based resources

As with most machine learning and neural network related technologies today, Python became the effective *lingua franca*. Google's TensorFlow and Keras libraries offer a good starting point for building TG-related RNNs. A good alternative is PyTorch, with its intuitive RNN architectures.

Other popular Python-based resources include:

- Natural Language Toolkit, NLTK, platform for building Python programs to work with human language data; iIncludes NLTK Corpora, a built-in support for dozens of corpora and trained models and many other resources
- spaCy, a free, open-source library for advanced NLP in Python; includes a number of trained models and pipelines, tokenization, lemmatization, etc.; open-source, industry level, all-things-NLP Python framework

- AllenNLP, a general framework for deep learning for NLP, based on PyTorch
- Fast.ai, a deep learning library built to make deep learning accessible to people without technical backgrounds; used for NLP, text mining and other applications
- TorchText, data processing utilities and popular datasets for NLP
- Gensim, a Python library for topic modelling, document indexing and similarity retrieval with large corpora
- OpenNMT, an open source ecosystem for neural machine translation and neural sequence learning
- Unitex, an open source, cross-platform, multilingual, lexicon- and grammar-based corpus processing suite
- TXM, a modular platform that includes techniques for analysis of large body of texts
- UDPipe (in Python), language-agnostic tokenization, tagging, lemmatization and dependency parsing of raw text, as well as language models building
- Sentence transformers, a Python framework for state-of-the-art sentence and text embeddings; it is based on BERT / RoBERTa / DistilBERT / ALBERT / XLNet implemented using PyTorch; documentation
- HuggingFace Transformers, thousands of pretrained models to perform tasks on texts such as classification, information extraction, question answering, summarization, translation, text generation, etc. in 100+ languages; based on PyTorch and TensorFlow, allowing one to use either of these two frameworks
- Introducing Text Analytics and Natural Language Processing with Python, a free EdX course that introduces the overall text analytics process (from data collection and preprocessing to evaluation of the results); useful to both those who are new to Python and those who have Python programming experience; assumes no previous knowledge of text analysis and mining
- Natural Language Processing in Python, a 2 hour tutorial that walks one through all major steps of the text mining process, without assuming any pre-knowledge related to text analysis; assumes familiarity with programming in Python; materials
- Modern NLP in Python tutorial, offers an introduction to text analytics with spaCy, a widely used Python framework for a variety of text analysis tasks; assumes a working knowledge of Python, but does not require any pre-knowledge of text analysis; materials

R-based resources

- Text as Data course, an overview of popular techniques for collecting, processing, and analyzing text-based data – including screen-scraping, mining data from application programming interfaces or APIs, topic modeling, text networks, and advanced text classifiers; good for those students who are familiar with R, but are newcomers to text analysis
- Text Mining with R – A Tidy Approach, publicly available book, with an accompanying GitHub repositorium with R Notebooks for all the topics covered in the book; excellent for someone who is a newcomer to text analysis but has at least a basic familiarity with R
- Introduction to Text Analytics with R, a series of 12 YouTube videos that gradually, through a practical example, walk one through all the phases of the text mining process; all the code and the datasets used in the examples are also made available (in the description of each video)
- Text Mining for Learning Content Analysis, a workshop run at LASI 2019 conference, with the objective to introduce the topic of text mining and to illustrate how it has been used in Learning Analytics research; materials; slides
- UDPipe (in R), language-agnostic tokenization, tagging, lemmatization and dependency parsing of raw text, as well as language models building

General language models, lexical databases and reference sites

OpenAI offers excellent autoregressive language models, such as GPT-2 and its cutting edge successor, GPT-3. For example, GPT-3 was used to generate an entire article for The Guardian.

Max Woolf's text generator, textgenrnn, built in Python on top of TensorFlow and Keras, offers a good entry point for understanding and hacking these technologies, especially for pythonistas. Uroš Krčadinac's generator, autoprose is built as an extension of textgenrnn, adjusted for both English and Serbian language. It is focused on generating entire trees of possible textual suggestions instead of successive linear suggestions.

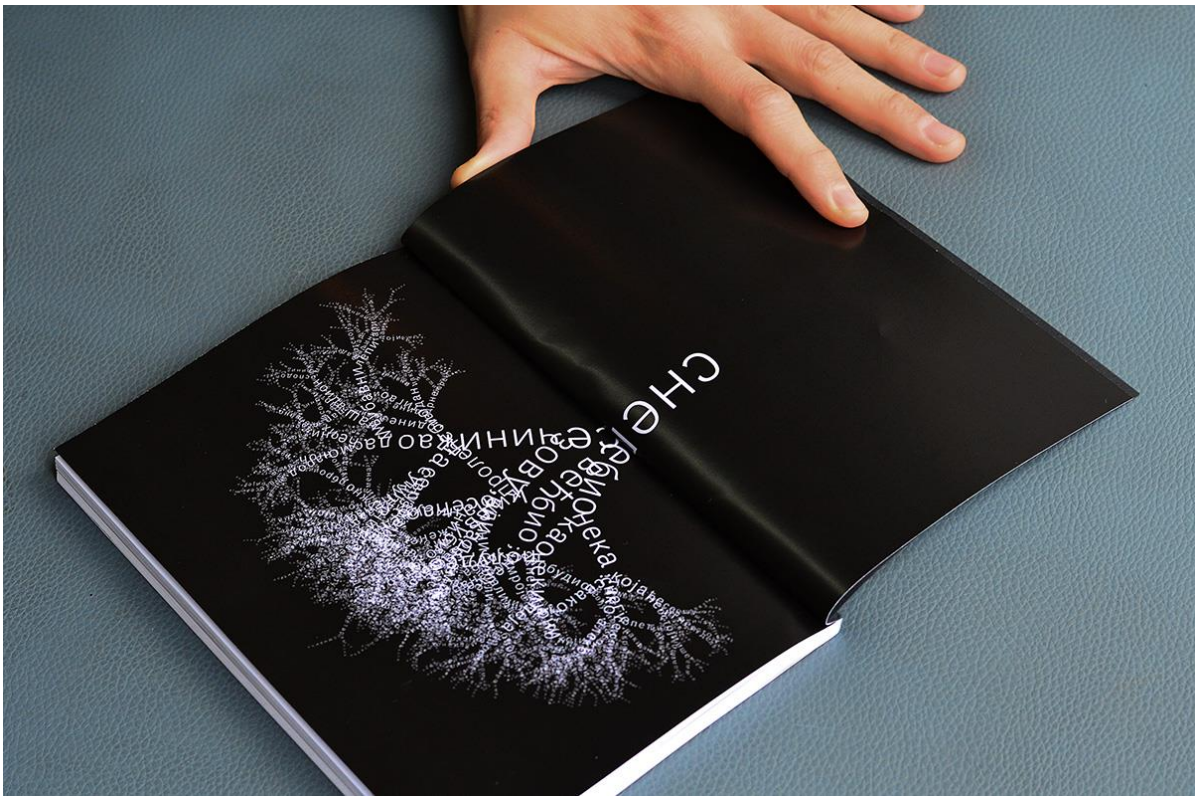


Figure 1. Uroš Krčadinac's generative visualization of many possible textual branches generated via a RNN. The project is online, interactive and live, in the form of a Web application. As a media art installation, it was shown at the Gallery of the Serbian Academy of Sciences and Arts, Belgrade Museum of Science and Technology, and Pančevo Gallery of Contemporary Art. It won the national award for the Best New Media Art Installation, *ARTificial Belgrade*, awarded by the Everseen company.

- BERT & OpenAI GPT-2, large-scale unsupervised language models that generate coherent paragraphs of text, achieve state-of-the-art performance on many language modeling benchmarks, and perform rudimentary reading comprehension, machine translation, question answering, and summarization – all without task-specific training
- GPT-3, an autoregressive language model that uses deep learning to produce human-like text; it is the third-generation language prediction model in the GPT-n series (and the successor to GPT-2) created by OpenAI
- WordNet, a large lexical database of English; nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept; synsets are interlinked by means of conceptual-semantic and lexical relations
- FrameNet, a lexical database of English that is both human- and machine-readable, based on annotating examples of how words are used in actual texts
- VerbNet, the largest on-line network of English verbs that links their syntactic and semantic patterns; it is a hierarchical, domain-independent, broad-coverage verb lexicon with mappings to other lexical resources, such as WordNet

- BabelNet, an innovative multilingual encyclopedic dictionary, with wide lexicographic and encyclopedic coverage of terms, and a semantic network/ontology which connects concepts and named entities in a very large network of semantic relations, made up of about 20 million entries
- The NLP Pandect, a reference place for everything about NLP (conferences, journals, podcasts, frameworks, ...)
- NLP Progress, repository to track the progress in NLP, including the datasets and the current state-of-the-art for the most common NLP tasks, for a variety of human languages

Domain-specific topics and related resources

Kaggle is an invaluable DS resource for teachers and students alike. Its extremely useful datasets are domain-specific, well-described, and are organized in a highly structured, easy-to-search collection. Domain-specific categories include business, economics, health, arts and entertainment, etc. There are thousands of datasets, most of them open and freely available. Whatever the domain the teachers want to use in their courses, it is highly likely that a suitable dataset is available on Kaggle to support it.

In addition, there are also many notebooks related to different domains that one can use to learn from, explore or customize to their needs. Kaggle also organizes competitions, a great way to assess relevant DS skills and see how good others are in running data analysis in a specific domain (the competition results and analyses are available as well).

There are other prominent collections of domain-related datasets, repositories and other DS resources. Although less comprehensive than Kaggle, UCI Machine Learning Repository is popular with DS teachers, as it organizes datasets not only by domain, but also by task (such as classification, regression,...), data type (sequential, time series, text,...), attribute type (categorical, numerical, mixed) etc. This makes it easy to select a suitable dataset to present and work with in class.

Towards AI site publishes resources for AI, machine learning and related discipline. It has recently published this collection of dataset repositories and finders. Likewise, the well-maintained and curated Carnegie Mellon collection of repositories of datasets and other general and domain-specific resources covers a broad spectrum of resources and useful links.

Examples

The DS case studies briefly presented here illustrate the general methodological principles and approaches outlined in the previous sections. The steps indicated in parentheses in the case studies refer to the steps indicated in the figure shown in the section *The big picture*.

Case study 1 – The impact of macroeconomic factors on the gender gap in the labor market

Gender gap is an important indicator of the achieved level of humanization of a society and also a precondition for its economic growth and development (step 1). The aim of this case study is to identify macroeconomic factors that have a statistically significant impact on gender gap variations in the labor market (step 2). The analysis includes EU member states (which gained membership in the period 2004-2014) and selected Balkan countries (Albania, Bosnia and Herzegovina, Macedonia, Montenegro and Turkey) (step 3).

In order to test the main hypothesis (H_1: There are specific macroeconomic factors, in the labor markets of the Balkan countries and EU member states that significantly affect the variations of the gender gap), a multiple regression model was built (step 4). The dependent variable of the model is the Gender

Gap Index economic participation, and the independent variables are macroeconomic indicators such as: population (working age population aged 15-64 expressed in thousands), GDP per employed person (according to current prices, international dollar), annual GDP growth rate (%), gross fixed capital formation (constant 2000, US $ 106), total investment measured as% of GDP, value added measured as% of GDP (agriculture), value added measured as% of GDP (industry) and value added measured as% of GDP (services) (steps 5 and 6).

The data of macroeconomic indicators are taken from the official database of the International Labor Organization, which consolidates data related to key labor market indicators. The data of the value of the gender gap sub-index are taken from the "Global Gender Gap Report 2013" taken from the official website of the World Economic Forum. All data refer to 2013 (step 7).

An attempt to build a model based on the stated "original" values of the variables was not successful, due to a disturbed assumption related to the presence of autocorrelation among the data. To remove this obstacle, for all listed variables (including the dependent one) the increment was calculated as the difference between the value of the variable in the current year and the value of the variable in the previous year. The obtained increments replaced the original values with the model of the included variables (step 10).

The variables of annual GDP growth rate (%) and value added measured as % of GDP (industry) were excluded from the analysis as they violated the assumption of multicollinearity (step 10). A regression model was constructed using SPSS software (step 15) - see Figure 1.

Independent variables	Coefficients	Standard errors	p-value
(const.)	0.001	0.003	0.789
Population	0.000	0.000	0.000
GDP per employed person (current prices)	0.001	0.000	0.010
Gross fixed capital formation	0.000	0.000	0.000
Total investment	-0.002	0,001	0.003
Value-added as% of GDP (agriculture)	0.004	0,002	0.022
Value-added as% of GDP (services)	0.000	0,000	0.000

Mean value of the dependent variable	0.0003	**R^2**	0.389
The sum of the squares of the residual	0.4310	**Adjusted R^2**	0.373
Standard deviation of the dependent variable	0.0555	**F (6;224)**	23.698
Standard regression error	0.0439	**p – value (F)**	0.000
Durbin-Watson		2.270	

Figure 1: Regression model (Dependent variable: Gender Gap Index of economic participation)

All assumptions related to the quality of the evaluated model were met, such as linearity, residual normality, autocorrelation, multicollinearity and heteroscedasticity (step 16). R Square is a good measure to determine how well the model fits the dependent variable. The R Square (R^2 or the coefficient of determination) is 0.389, which means that 38.9% of variance in the dependent variable is explained by the independent variables in a regression model. In other words, It means that 38.9% of the data fit the regression model (step 17). The model is statistically significant (p = 0.000).

The aim of constructing this model was to identify macroeconomic factors that make influence on the gender gap in the labor market of selected countries. In this sense, the macroeconomic indicators like populations, GDP per employed person, gross fixed capital formation, total investment as% of GDP,

value added as % of GDP in agriculture and value added as % of GDP in services had a statistically significant impact on the Gender Gap Index of economic participation. Because the increments are included in the model instead of the original values in order to eliminate the positive autocorrelation that appeared among the data, the value of the obtained coefficients with predictor variables is low and inadequate for interpretation in terms of explanatory qualities.

Case study 2 – Accommodation facilities clustering based on web data

Understanding the problem

Kopaonik is an all-seasons's touristic location with a plethora of accommodation facilities. On the site https://www.booking.com/city/rs/kopaonik.sr.html one can find official information about these facilities, but also reviews left by visitors (steps 1,3 and 7). In order to present the different accommodation options to tourists, it would be preferable to group similar accommodation facilities together, according to their characteristics, but also to the given reviews (step 2). As we certainly do not want to read all the reviews, we turn to automatic discovery of the cluster structure from available data (step 4).

Understanding the data

Information fetched from the documents on the website include both structured data (facility name, category expressed by number of stars, rating, total number of reviews, average price per night stay) and unstructured text (facility description, reviews) (steps 5,6 and 8). Therefore, adequate preprocessing techniques should be applied. Figure 1 shows part of the original content.

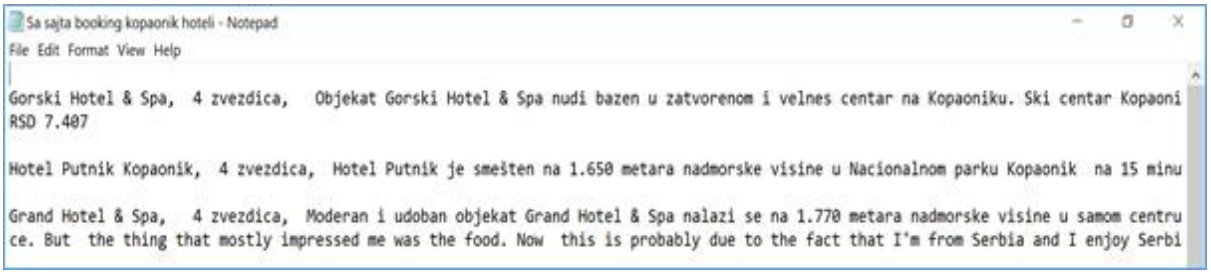


Figure 1: Content fetched by the crawler

Data preprocessing

Missing attribute values should be replaced - for e.g. if a facility has no star- categorisation. Irrelevant attributes should be excluded from modelling – for e.g. name of the facility is important only for referencing purposes (step 12). To facilitate the process of matching keywords and documents, we should apply:

1. tokenization – remove all punctuation marks (and substitute Serbian letters Ć,ć,Č,č with C,c, Ž,ž with Z,z, and Đ,đ with Dj,dj)
2. normalization – all characters are converted to lowercase (optional - filters in Weka can do this)
3. stopwords removal – common words that do not help distinguish reviews are removed (step 10).

Figure 2 shows part of the content in .CSV file format. Notice that column C contains text data that should be transformed to a bag of words (the ***Nominal ToString*** and ***StringToWordVector*** filter in Weka).

	A	B	C	D	E	F	G
1	Ime hotela	Broj zvezdi	Recenzija	Ocena	Opisna ocena	Ukupno recenzija	ProseCna cena po
2	Gorski Hot	4	Objekat Gorski Hotel i Spa nudi bazen u zatv	9.5	Izuzetan	528	7.407
3	Hotel Putn	4	Hotel Putnik je smesten na 1.650 metara nad	9.1	Izvanredan	412	7.172
4	Grand Hot	4	Moderan i udoban objekat Grand Hotel i Spa r	9	Izvanredan	497	9.406
5	Grey Hotel	4	Hotel Grey Kopaonik nalazi se na Kopaoniku n	9.5	Izuzetan	65	22.116

Figure 2: Data in a .CSV file format ready to be loaded into the Weka tool

We can choose between Boolean, term-frequency (TF) and TF-IDF inverse document frequency word-vector representations. In our analysis we used the TF approach and restricted ourselves to the 50 most frequent words (step 12).

Modelling

Prior to modeling, it is necessary to add another field at the end of the file, named ***cluster***. For this we apply the ***AddCluster*** filter. The algorithm will write the assigned cluster value here.

The deterministic simple k-means algorithm (***SimpleKMeans***) is used for clustering (step 14). The number of clusters is set as k=4 (determined in Weka experimenter as an optimum) (step 15).

Figure 3 shows a few rows from the file with the cluster assignments – on the left hand side is the beginning of the table (with the facility name in the first column) and on the right hand side is the end of the table (the columns with the last few terms and the cluster assignments). Figure 4 shows the central tendencies in each cluster, while Figure 5 visualizes the cluster memberships for 12 hotels on Kopaonik (step 16).

Relation: Ocene hotela na Kopaoniku-w

No.	1: Ime hotela	2: Broj zvezdica	3: (
	Nominal	Numeric	Nu
1	Hotel Kralj...	4.0	
2	Apart & Sp...	4.0	
3	Hotel Milm...	3.0	
4	Hotel Club ...	0.0	
5	Hotel Putni...	4.0	
6	Grey Hotel...	4.0	
7	Hotel Juni...	3.0	

rs.unsupervised.attribute.NominalToString-C3-weka.filters.unsupervised.attribute.S

37: rooms	38: ski	39: smesten	40: spa	41: staff	42: very	43: water	44: cluster
Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Nominal
0.0	0.0	1.0	0.0	0.0	0.0	0.0	cluster1
1.0	0.0	1.0	2.0	1.0	0.0	0.0	cluster1
0.0	0.0	0.0	1.0	0.0	0.0	0.0	cluster1
0.0	0.0	0.0	0.0	0.0	1.0	0.0	cluster1
0.0	1.0	1.0	0.0	0.0	0.0	0.0	cluster2
0.0	1.0	0.0	0.0	0.0	0.0	0.0	cluster2
0.0	0.0	0.0	1.0	0.0	0.0	0.0	cluster2

Figure 3: Tabular facility grouping

Analyzing the central tendencies (centroids) of each cluster and the representative keywords used in reviews of facilities grouped together, we can understand the visitors' perceptions of these facilities.

Deployment

The described clustering process can be useful in the tourism sector for segmentation of accommodation facilities not according to formal criteria such as stars-rating, size of rooms, etc. but according to the overall impression the facility has made on visitors (step 18).

		Cluster#			
Attribute	Full Data	0	1	2	3
	(12.0)	(4.0)	(3.0)	(3.0)	(2.0)
Broj zvezdica	2.8333	2.75	3.6667	1.3333	4
Ocena	8.5833	7.975	9.2	8.3333	9.25
Opisna ocena	Veoma dobar	Veoma dobar	Izuzetan	Izvanredan	Izuzetan
Ukupno recenzija	221	158.5	186.3333	144.6667	512.5
ProseCna cena po nocenju (RSD)	8.9211	7.831	11.9573	7.6813	8.4065
Grand	0.3333	0	0	0	2
Hotel	1	0.75	1	0.6667	2
Kopaonik	1.25	0.75	1.3333	1.6667	1.5
Kopaoniku	0.4167	0	0.3333	1	0.5
Objekat	0.3333	0	0.3333	0.6667	0.5
SPA	0.25	0	0	0	1.5
barom	0.25	0	0	1	0
bazen	0.25	0.5	0	0	0.5

Figure 4: Final cluster centroids

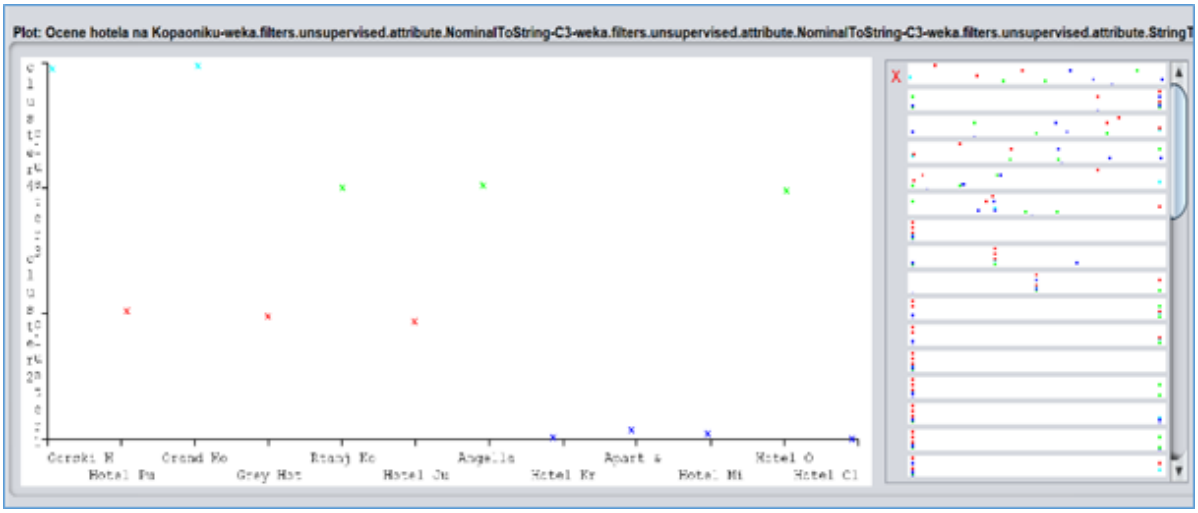


Figure 5: Visualization of clustering results

Case study 3 – Predicting the number of passengers in public transport

Understanding the problem

The behaviour of passengers in public transport in urban areas is characterized by the degree of their mobility, traffic demands, travelling distances, the available lines and the number of passengers, as an indicator of (dis)balance between provided services and demands on the market. If known, these parameters may contribute to efficient and correct decision making at all levels of management in transport organizations. (step 1)

The only public transportation option in Subotica is by bus. The Public Transport Company providing this service has had a problem of unexpected increase in the number of passengers on a daily basis, the situation that urged for immediate solution. As a plan of bus activities for the next day is always made on the prior day, an ability to foresee fluctuations in passenger number was highly desirable. (step 2)

The problem lies in unpredictable behaviour of people that don't use daily/monthly tickets, but make a decision about travelling by bus *ad hoc* and this way they contribute to the increased demand for transportation. Consequently, the task is to predict the number of “irregular” users, i.e. to analyse the number of singular bus tickets sold (step 4). If a manager has to resolve a situation of increase in the number of passengers, i.e. to decide to which bus line to assign an additional vehicle (and often this means to reduce the number of vehicles on other lines) it is practical to know the structure of the “unexpected” passengers, especially if the bus routes can be “associated” with some part of the population (the working population, the college population, elderly people, etc.)

We divided the number of sold bus tickets into two main categories: tickets sold by full price and tickets sold by reduced price to 50%. The discounted tickets are available only for elderly people (over 70 years), pensioners (again mainly the older population), and children from 6 to 10 years accompanied by an adult (but their ratio in the number of such tickets is negligible), while it is assumed that tickets at full price are bought by the rest of the population.

Understanding the data

The data were collected from the Public Transport Company in Subotica and the Hydro-meteorological Service of the Republic of Serbia and originated from a period of one year, resulting in 730 daily observations (step 7).

Data preprocessing

Since 3.5% of meteorological data were missing, the missing data were gathered from the closest meteorological stations in the area, and new, interpolated data on whether conditions in Subotica in the time period under investigation were calculated to replace the missing ones (step 10).

After the correlation analysis, some of the input variables were discarded from further analysis, while some others were grouped together in order to get a new, cumulative variable. For example, the variable *Price of premium gasoline* was highly correlated (r=0.999) with the variable *Price of Regular gasoline* and *Price of diesel* (r=0.989). Therefore these three variables were all discarded and a new averaging variable *Price of gasoline* was introduced. The numeric attribute *Day* with values 1-7 was discarded due to the high correlation (r=-0.758) with the numeric variable *Type of a day* with values 50, 10 and 12 denoting workdays, weekends, and holidays, respectively, and only the latter was used in the analysis. The temperatures measured at night and in the morning were also highly correlated (r=0.978), and the variable *The lowest night temperature* was selected for the analysis (step 12). Consequently, the final input variables were: *Type of a day,* ratio of *Price of gasoline* and *Price of tickets*, ratio of *Earnings* and *Expenditures*, *Air pressure*, *The lowest night temperature*, *Air moisture*, *Direction of the wind*, *Speed of the wind, Cloudiness, General weather condition, Rainfalls, Condition of the ground*, and *Height of snow.*

Furthermore, records collected on some special date, such as May 1st, November 1st, etc. containing outliers, were removed from the data set, as they always show extremely high values due to specific habits and behaviour of local inhabitants, which could “confuse” the neural network during the training phase (step 10).

Modelling

For the prediction of the number of sold bus tickets in public transport we decided on two supervised learning models, one non-linear (feed forward neural networks with back propagation learning) and one linear (the multiple linear regression model). This choice was made according to the existing heuristics for DS algorithm selection and the fact that the nature of relations we were hoping to reveal was unknown. On the other hand, we were limited in algorithm options by the choice of Weka software tool that we used for analysis (step 14).

Neural Network Architecture

For the initial architecture of the neural network we selected the following physical configuration:

1. The number of nodes in the output layer is two as the neural network should model the behaviour of two output variables: the number of sold tickets by full price and by the reduced price;
2. The number of nodes in the input layer is determined by the number of input variables and equals 13. The number of hidden layers was set to one;
3. The number of neurons in the hidden layer was decreased from the initial 25, with a step of 5, to 5 neurons in the hidden layer, resulting in five different physical architectures;
4. As a discriminator we selected the standard propagation function.

5. At the beginning of training we set the transfer function of the input and the output layer to linear function, while the activation function of neurons in the hidden layer was set to hyperbolic tangens and afterwards it was changed to parabolic and sigmoid functions;
6. The standard error back propagation algorithm and the modified Delta rule were used in the model. The amount of data in the training set compared to the amount of data in the test set was 80:20. We chose the random presentation of samples from the training and the test data sets to the neural network. The desired accuracy we were looking for was set to 90%. This value was calculated by RMS error and the maximal error during training, by formula:

$$\varepsilon = 0{,}9\varepsilon_k + 0{,}1\varepsilon_m \quad (1)$$

Where ε is the calculated error that shouldn't exceed 10%, ε_k stands for RMS error and ε_m for maximal error;

7. The values of learning parameters □ (momentum), □∈[0.0, 0.9] and □ (learning rate), □ ∈[0.1, 0.9], were varied during the training phase with a step of 0.1. (step 15)

Results of different neural network configurations

Five neural network architectures that achieved the best test results under the evaluation criteria are shown in table 1. As can be seen, the best predictive performance was achieved by the network that had 5 neurons in the hidden layer, and the transfer functions of neurons in the hidden and the output layer were set to sigmoid functions (step 16). Further improvement and the final neural network architecture were gained after removing individual connections that were under the relevance threshold, set to 0.02, in the I ranked architecture. As a result of this change, the maximal training error of the network was reduced to 0.4065, the training RMS error was 0.0836, while the maximal testing error of the network was 0.4070 and the test RMS error was 0.1097 (step 17).

Layers of neurons		Learning parameters		RMS	Epoch	Rank
Hidden	Output					
No. of neurons		Learning Rate	Momentum			
5	2	0.4	0.7	0.1132	780	I
10	2	0.8	0.4	0.1137	530	II
10	2	0.6	0.5	0.1148	620	III
10	2	0.1	0.0	0.1152	4660	IV
20	2	0.1	0.2	0.1155	2580	V

Table 1: Structure, learning parameters, and RMS error for five "best" neural network architectures

The Multiple Regression Model

As an alternative to the nonlinear neural network model we applied, on the same data set, the multiple linear regression methodology. Firstly we tried to predict the number of passengers who would pay the full price of a bus ticket, and secondly we repeated the process for forecasting the number of tickets sold by reduced price to 50% of its value (step 15). After the estimation of regression parameters on the training sample and the application of the obtained regression function to the test sample, we computed the RMS error for both predicted variables. The results are summarised in table 2 (step 17).

	Number of sold bus tickets	
	by full price (100%)	by reduced price (50%)
RMS error	0,222630421	0,131068527
Significance	$6,28 > F_{0,05}(13,129) = 1,74918$ $6,28 > F_{0,01}(13,129) = 2,18846$	$1,91 > F_{0,05}(13,129) = 1,74918$ $1,91 < F_{0,01}(13,129) = 2,18846$

Table 2: Prediction accuracy of multiple linear regression models

The 10% error, that we managed to achieve, is in most practical applications quite acceptable, having it in mind when obtained results are utilised. When compared to linear regression modelling, the final neural network model was still more successful, having smaller prediction error. Since the RMS error of the multiple linear regression is worse than the RMS obtained with nonlinear MLP (RMS=0.1132), for both the prediction of the number of sold bus tickets by full and by reduced price, we can conclude that the neural network showed to be more adequate for forecasting the number of passengers in public transport (step 13).

Deployment

Oscillations in the number of passengers known in advance can easily be incorporated into the resource allocation plan of a public transport company. Accurate prediction of passengers' number can decrease the risk and uncertainty in the decision making process (step 18).

Case study 4 – The impact of internal factors on business success

Understanding the problem

The main goal of every enterprise is long-term business. In order to achieve that, enterprises must run business activities successfully. Business success is usually measured by profitability. Profitability is a key prerequisite for the growth and development of a business and the achievement of its core business goal. Besides continuously measuring profitability, management of enterprises must identify which factors have significant influence on profitability (step 1). The aim of this case study is to identify and measure the impact of internal factors on the business success of meat processing enterprises expressed through profitability (step 2). Panel analysis is constructed for the sample which includes 24 enterprises in Serbia at the period from 2007 to 2016 (step 3-4).

Understanding and preparing the data

The case study is based on financial statements of meat processing enterprises for a period 2007-2016 (step 5). The original sample consisted of 34 meat processing enterprises, but in order to build a balanced panel model, the final sample covered 24 enterprises that were observed in the period 2007-2016 (step 6). The source of data was the Agency for Business Registers of the Republic of Serbia. The data was collected from the Scoring database for 2019. (step 7).The business success of selected enterprises in the survey is measured by its profitability. The accounting rate of Return on assets (ROA), as a measure of productivity, was in function of the dependent variable, while the size of the enterprise, age, debt ratio, quick ratio, inventory, sale growth and capital turnover ratio were found as independent variables. Before being included in the model, the data were recalculated according to the appropriate formulas, for all variables except inventory, age and size (step 10).

Modeling

In accordance with the aim of case study the following hypothesis was set up (H_1: “Internal factors, such as size of company, age, debt ratio, quick ratio, inventory, sale growth and capital turnover ratio, have a significant influence on profitability of meat processing enterprises in Serbia”).

The question “Which model to choose” is frequently raised when conducting empirical research. Panel model diagnostic shows that the pooled OLS (Ordinary Least Squares) model is appropriate (step 14). After satisfying all assumptions, the pooled OLS model is performed (step 15-16). The coefficient estimations are given in Figure 1 (step 17).

Variables	*Coefficient*	*Std. Error*	*t-ratio*	*p-value*
const	0.2798	0.0584	4.783	<0.00001***
Size	-0.0025	0.0084	-0.303	0.76144
Age	**-0.0019**	0.0007	-2.724	0.00692***
Debt ratio	**-0.2413**	0.0316	-7.629	<0.00001***
Quick ratio	**0.0161**	0.0094	1.713	0.08792*
Inventory	0.0264	0.0442	0.598	0.55015
Sale growth	**0.0301**	0.0144	2.086	0.03805**
Capital turnover ratio	**-0.0897**	0.0258	-3.472	0.00062***
R-squared	0.370765	Adjusted R-squared		0.351779
F(7, 232)	19.52881	P-value(F)		0,0000
Log-likelihood	289.5504	Akaike criterion		-563.1007
Schwarz criterion	-535.2556	Hannan-Quinn		-551.8812
rho	-0.052180	Durbin-Watson		1.959131

Figure 1: Pooled OLS model

Based on the results of the panel analysis (see Figure 1), it can be concluded that most variables included in the model are statistically significant. Variables age, debt ratio and capital turnover ratio are statistically significant at the level of significance of 1%, the variable sale growth is statistically significant at the level of significance of 5% and the variable quick ratio is statistically significant at the level of significance of 10%. Other variables have not a statistically significant impact on the dependent variable, which means that the null hypothesis was partially accepted. The research results indicate that quick ratio and sales growth have a significant positive impact on profitability of enterprises in the food processing industry. On the other side, age, debt ratio and capital turnover have a significant negative impact on return on assets. Furthermore, the results show that size and inventory have insignificant influence on profitability of food processing enterprises

Deployment

The results of this research can be useful for many internal and external users of financial statements of food processing companies in order to realize adequate business decisions. They can help them to better understand the impact of internal factors on profitability and make better decisions about investment in this sector (step 18).

Case study 5 – Modeling the gender pay gap

Understanding the problem

Better economic status of women in the labor market and reduction of the gender pay gap is an important determinant of economic and social progress of the country. Gender pay gap is one of the key indicators of women's access to economic opportunities and undoubtedly one of the most constant features of the labor market (Step 1). The aim of this analysis is to determine whether there is a difference between men and women regarding wages (Step 2). For the analysis we have used data collected by the survey EU-SILC in 2014 in Serbia (Step 3). Mincer earnings function according to which individuals' earnings are a function of the achieved level of education and work experience, served as the basis for the analysis of factors that determine the formation of wages and constructing the regression model (Step 4).

Understanding and preparing the data

Data collected by the Survey on Income and Living Conditions - EU-SILC for 2014 were used for the purposes of modelling (step 5). This survey was conducted by the Statistical Office of The Republic of Serbia (Step 7). The analysis included 1891 people. List of variables is presented in Figure 1. Individuals with extremely low or high earnings were excluded from the analysis. Variables such as sectors of activity in which respondents work and occupation were excluded from the analysis because of incomplete data (Step 6). The survey data used in this study were not adequate for determining the size of the gender gap in earnings because a large number of respondents did not provide information about the wage size. Therefore only the existence of the gender pay gap could have been analyzed. Problems related to the collection of data relevant for insight into the gender differences on the labor market and insight into the existing structure of the workforce is an enormous obstacle for research endeavors dealing with this issue in most of the Balkan countries. (Step 10).

Name of variable	Coding
Dependent variable	
The logarithmic value of the net monthly salary of the employee	
Independent variables	
Gender	Female = 1 Male = 0
Age	Ratio
Highest education: secondary education	Secondary education = 1 Else = 0
Highest education: tertiary education	Tertiary education = 1 Else = 0
Years of working experience	Ratio
Person is employed by public sector	Public sector = 1 Others = 0
Person is employed by private sector	Private sector = 1 Others = 0
Hours of work during the week	Ratio

Figure 1: List of variables used in the model

Modeling

Regression model was estimated by the method of least squares. Some variables are excluded from the analysis, e.g. tertiary education (none of the respondents who gave information on the monthly

earnings belonged to this category), as well as private property company (if the employee is with the company) and work experience measured in years, because they demonstrated multicollinearity (Step 15). Preliminary analysis was conducted in order to satisfy the assumptions of linearity and normality, and absence of autocorrelation, multicollinearity and heteroscedasticity (Step 16). The coefficient of determination (R^2) whose role is to point out how much of the variance of the dependent variable (the logarithmic value of net earnings) explains the model is 0.154 or 15.4% (see Figure 2). Adjusted coefficient of determination (Adjusted R^2) is 0.152, or 15.2%. It provides the better assessment of the actual value of the coefficient of determination in the population. Low coefficient of determination indicates the importance of other, non-economic factors. The model is statistically significant ($F(6,1885) = 68.826$; $p < 0.005$). The highest standardized coefficient beta is related to the variable employed by the company in public ownership (0.26) and variable gender (-0.21), which means that these variables are the largest contributors to explanation of the variation in the dependent variable (the logarithmic value of net earnings), when subtracting the variance explained by other variables in the model. What is important to indicate is that the standardized regression coefficient on the variable gender is negative (-0.068), which indicates that the females earn less than males in Serbia. All other statistically significant predictor variables included in the model have a positive sign, which brings us to the conclusion that they have positive influence on the dependent variable. (Step 17).

Independent variables	Coefficient	Standardized Coefficients (Beta)	p-value
Constant	4.315		0.000
Gender	-0.068	-0.21	0.000
Age	0.001	0.10	0.000
Secondary education	0.052	0.12	0.000
Employed by the company in public ownership	0.085	0.26	0.000
Hours worked per week	0.001	0.09	0.000
The mean value of the dependent variable	4.456	R^2	0.154
The standard deviation of the dependent variable	0.158	Adjusted R^2	0.152
Sum of squared residuals	40.142	$F_{(6,1885)}$	68.826
The standard error of regression	0.146	p-value (F)	0.000
Durbin-Watson		1.768	

Figure 2: Regression model

Deployment

The results of this study can help government officials, entrepreneurs and employers to understand that failure to comply with the principle of equality and equal opportunities for women and men is considered a violation of basic human rights. As a result, there are significant losses in the economy of countries such as loss of business and economic benefits, and insufficient use of available human resources. If there is no economic independence, all other measures taken to improve the position of women in society, in general, have much less success and influence. (Step 18).

Case study 6 – Modeling the antibiotic dose

Understanding the problem

Escherichia coli (E. coli) is a Gram-negative, rod-shaped bacterium that is a common cause of urinary infections. In recent years, in some strains, due to high resistance to frequently used antibiotics, this type of urinary infections are becoming very difficult to treat (step 1). A new medication, “Antibiotic X” has been recently developed, preliminary testing in a clinical setting has been performed, but it has

yet to undergo broad clinical trials. The aim of this study is to analyze the data from the preliminary study and to predict the effective dose of Antibiotic X for further testing (step 2).

Understanding and preparing the data

The analysis includes data obtained from 100 patients diagnosed with severe multiresistant E. coli urinary infections based on the number bacterial colony forming units per milliliter of urine (cfu/ml, all patients had initial values of 10^8 cfu/l) recruited from a general hospital in Belgrade Serbia (step 3). All the participants were given relatively small doses of antibiotic which produced a significant, but not sufficient reduction of bacterial urine concentrations (sufficient reduction is defined as a decrease to an acceptable concentration of 50000 cfu/ml). Bacterial urine concentrations were measured using a novel microbiological test capable of precisely assessing the exact cfu numbers rather than ranges. Significant negative correlation existed between antibiotic dose and urine E. coli concentrations and the relationship was assumed to be linear in nature. In order to test the main hypothesis (H1: there is a safe, non-toxic dose of Antibiotic X that can lower the urine E. coli concentrations to 50000 cfu/ml), a linear regression model was built (step 4). The dependent variable of the model is the drug dose (milligrams per day) and the independent variable is the bacterial urine concentration (steps 5 and 6). The data for the patient sample were taken from the general hospital database as results from the preliminary study, in the form of MS Excel sheet. All data refer to the study from 2020 (step 7). No additional modifications or transformations of the data were required (step 10).

Modeling

A regression model was constructed using SPSS software (step 15). Regression model summary, results from ANOVA test and coefficient values are presented in figure 1. All assumptions related to the quality of the evaluated model were met, such as linearity and heteroscedasticity (step 16). As seen from the figure, R square value (R^2 or the coefficient of determination) was 0.883 and adjusted R square value was very similar (0,881). This essentially means that 88.3% of the data for E. Coli concentration and drug dose fit the regression model (step 17). The model was found to be statistically highly significant (p = 0.000). Regression equation was determined as:

Antibiotic dose = -0.00005 * E.Coli concentration + 590 mg

For the desired urine level of 50000 cfu/ml it was estimated that the Antibiotic X daily dose of 587.5mg was required.

Deployment

The results will help determine the minimal effective dose of the new antibiotic. The potential toxicity and side effects of this dose of the antibiotic will be assessed in a future study. This future study will be performed on a larger sample, it will be multicentric, blinded and randomized. If the medication proves to be nontoxic, it could become an integral part of conventional treatment protocols for multiresistant E. Coli urinary infections (step 18).

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate

1	,939[a]	,883	,881	46,11726

a. Predictors: (Constant), E.Coli concentration

b. Dependent Variable: Antibiotic dose

ANOVA

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	1565716,152	1	1565716,152	736,183	,000[b]
	Residual	208426,598	98	2126,802		
	Total	1774142,750	99			

a. Dependent Variable: Antibiotic dose

b. Predictors: (Constant), E.Coli concentration

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients			95,0% Confidence Interval for B	
		B	Std. Error	Beta	t	Sig.	Lower Bound	Upper Bound
1	(Constant)	590,040	11,091		53,198	,000	568,030	612,050
	E.Coli concentration	-4,781E-5	,000	-,939	-27,133	,000	,000	,000

a. Dependent Variable: Antibiotic dose

Figure 1. Regression model summary, results from ANOVA test and coefficient values

Case study 7 – Party Membership and Trust in Political Institutions in Europe[3]

Understanding the problem

Traditionally, political parties have been the main linkage between citizens and the state, and therefore very important for the overall health of democracy. Political parties socialize citizens into politics enabling them to identify relevant policy issues, they discursively structure political debate, and 'organize democracy' (Dalton et al., 2011). However, in the 21st century, political parties are becoming increasingly professionalized, meaning that their integrative function and linkages with the ordinary citizens are getting weaker (Scarrow et al., 2002) (step 1). Given the importance of political parties for democracy, on the one hand, and their professionalization and loss of mass support, on the other, the goal here (step 2) is to explore the impact of party membership on levels of political trust in Europe. Taking into consideration findings of earlier research according to which close ties with political parties enable citizens to effectively express their opinions and relate to political process, the hypothesis is that party membership and party closeness contribute to the development of political trust. Therefore, it can be expected that the erosion of party membership has a negative impact on levels of political trust, and overall democracy in Europe.

Understanding and preparing the data

In order to test the hypothesis, cumulative data from rounds 1-5 of European Social Survey (2002-2010) and the hierarchical linear (multilevel) regression models are used (steps 3 and 4). The analysis included data on 16 European countries in a time period covering 8 years, numbering a total of 199 160 observations (respondents) (steps 5, 7). European Social Survey represents a cross-national comparative survey that gathers micro-data biannually, implementing high quality standards and producing reliable and comparable survey data.

In the analysis, the data on institutional trust, self-reported party membership and party closeness were used, alongside with the data on gender, age, the level of education of respondents, self-reported data on satisfaction with income, average weekly time of watching television and reading newspapers, religiosity, citizenship, birth country and self-declared interest in politics. In addition, the level II country data on the existence of authoritarian legacy was also introduced (step 6). The data are provided in a structured and standardized form (downloadable in different formats - SPSS, STATA, CSV) and collected through rigorous survey procedure of standardized face-to-face interviews on representative national samples for each country (steps 8, 9).

Prior to using the data, especially in comparative analytical procedures, such as this one, different weights must be employed (population size weight combined with design and post-stratification weights). Furthermore, almost all of the data require different types of transformations and re-codings, in order to be transformed into different types of scales suitable for the chosen statistical procedures (for example, cleaning "no-answers" or missing data), or in order to be able to construct indexes out of several individual variables (such as index of political trust, made of individual items measuring trust in country's parliament, legal system, police and politicians, which required factor analysis and scale reliability testing prior to final computing of the scale) (step 10). Likewise, level II (country) variable on authoritarian legacy, was constructed as a dummy variable for the countries that had authoritarian types of political regimes until the early 1990's.

[3] This example is based on the paper by Marc Hooghe and Anna Kern (2013)

Modeling

In order to test the hypothesis, a hierarchical linear (multilevel) regression analysis was chosen (step 13). The dependant variable in all models was the index of political trust, whereas the level I (individual) independant variables were self-declared party membership and self-declared party closeness, controlling for the effects of age, gender, the level of education, income satisfaction, average weekly time of reading newspapers and watching television, religiosity, interest in politics, citizenship and birth country. In order to control the effect of the round in which data were gathered, a country-year level variable - time - was also introduced. At the country-level, the variable measuring authoritarian legacy of the country was introduced as the II level measurement which effect is also being controlled. Finally, cross level interactions were controlled - party membership, time and party closeness - in order to explore whether party membership and party affiliations are stable over time (step 14).

	Political trust				
	Model 0	Model I	Model II	Model III	Model IV
Individual-level variables					
Gender (Male = 1)		0.008 (0.006)	0.008 (0.006)	−0.024*** (0.006)	−0.024*** (0.006)
Year of birth		0.002*** (0.000)	0.002*** (0.000)	0.002*** (0.000)	0.002*** (0.000)
Education level		0.050*** (0.003)	0.050*** (0.003)	0.024*** (0.003)	0.024*** (0.003)
Satisfaction with income		0.250*** (0.004)	0.250*** (0.004)	0.240*** (0.004)	0.240*** (0.004)
Watching television		0.006*** (0.002)	0.006*** (0.002)	0.005** (0.002)	0.005** (0.002)
Reading newspaper		0.028*** (0.003)	0.028*** (0.003)	0.010*** (0.003)	0.010*** (0.003)
Member of political party (Yes = 1)		0.148*** (0.015)	0.148*** (0.015)	0.096*** (0.015)	0.093** (0.035)
Close to political party		0.178*** (0.003)	0.178*** (0.003)	0.138*** (0.003)	0.108*** (0.008)
Citizen of country (Yes = 1)		−0.177*** (0.021)	−0.177*** (0.021)	−0.185*** (0.021)	−0.185*** (0.021)
Born in the country (Yes = 1)		−0.097*** (0.014)	−0.097*** (0.014)	−0.089*** (0.014)	−0.089*** (0.014)
Religiousness		0.063*** (0.001)	0.063*** (0.001)	0.062*** (0.001)	0.062*** (0.001)
Political interest				0.177*** (0.004)	0.177*** (0.004)
Country-year level variables					
Time		−0.049** (0.017)	−0.048** (0.017)	−0.048** (0.017)	−0.048** (0.017)
Country-level variables					
Authoritarian legacy			−1.030*** (0.213)	−1.012*** (0.212)	−1.012*** (0.212)
Cross-level interactions					
Member of political party * Time					0.001 (0.010)
Close to political party * Time					0.010*** (0.002)
Intercept	−0.036 (0.152)	0.366* (0.141)	0.671*** (0.128)	0.672*** (0.127)	0.672*** (0.127)
Individual-level variance	2.098	1.975	1.975	1.957	1.956
Country-year-level variance	0.078	0.059	0.059	0.058	0.058
Country-level variance	0.667	0.489	0.269	0.265	0.265
Deviance	713,351	701,311	701,293	699,434	699,415

Note: Entries are parameter estimates and standard errors (in parentheses) of a multilevel linear regression. All models include 199,160 individuals on the first level, 121 country-years on the second level and 30 countries on the third level.
Sign.: * < 0.05, ** < 0.01, *** < 0.001.

Source: Hooghe, Kern, 2013: 7

The base model (0) shows that 73.8% of total variance is found at individual level, additional 23.5% between the countries and 2.7% between survey waves within countries, indicating enough between country differences to engage multilevel models (step 15). The next four models (step 16) differentiate the I and II level variables, and each subsequent model includes additional variables at both levels. All four models show that individual (I) level variables (except gender in the first model) are significantly associated with the level of trust. Education level, religiosity, income satisfaction, reading newspapers, watching TV, party membership, and feeling closeness to a certain party are positively associated with political trust. The first model introduces the wave as an II level variable, and shows that political trust across Europe tends to decline over the years. In the second model another II level variable (coded as dummy variable) is added - authoritarian legacy. There is a clear negative effect of the authoritarian legacy of the country on the political trust of the respondents. The additional variables are introduced at I level (political interest) in the third model. Adding political interest improved the model and did not affect much either individual or contextual coefficients. In the fourth model the interactions between I and II level indicators were used: whether is respondent a member of political party and (*) wave, and how close to the party respondents is and (*) wave. Those who are closely connected to a party tend to have more political trust over time. There is no effect of the party membership over time on political trust.

Deployment

This model could be applied in exploration of interactions between political trust and party membership in Europe on the data gathered in the future ESS survey waves (steps 18-21).

Case study 8 – A spatial domain Image Processing in AstroPython

Understanding the problem

This case study examines one of two different forms of data which can occur in complex data analysis in astronomy, but is not limited to this science since images are relevant to medicine, satellite surveys as well as security. The Coma cluster (Abell 1656) is a large cluster of galaxies that contains over 1,000 identified galaxies. It is located in and named after the constellation Coma Berenices (step 1).The cluster's mean distance from Earth is 321 million light years. About 90% of the mass of the Coma cluster is believed to be in the form of dark matter. However, the distribution of dark matter throughout the cluster is poorly constrained.

The aim of the data analysis is to subtract the emission from the two bright giant elliptical galaxies NGC 4889 and NGC 4874, especially their diffuse stellar halo emission, in order to reveal hidden compact sources.. By mapping the visible matter, we can discover the "hidden" mass (i.e. dark matter) required to explain the observed dynamics of the galaxy cluster (step 2).

In astronomy, we have an actual image of the sky, and a catalog of sources (images of planets, stars, galaxies, etc) (step 3). Astronomical images (and catalogs for that matter) are most often stored in FITS format, which stands for Flexible Image Transport System. There are several programs for opening and examining FITS images. In this case study, the Astropy package needed to be installed.

Astronomical images (but not limited to it) are blurred images of distant compact sources due to diffuse emission related to different gas tracers across the electromagnetic spectrum. To extract meaningful positions of objects, a combination of MIN-MAX spatial filters can be used. MAX and MIN filters attribute to each pixel in an image a new value equal to the maximum or minimum value in a neighborhood around that pixel. The neighborhood stands for the shape of the filter. Particularly, we can apply lower envelope filter LOW = MAX(MIN), to distinguish smooth edges of compact sources (step 4).

Understanding and preparing the data

Here is the original optical image of visible matter in Coma cluster (its Pythonic view in magma colormap with pixels on axes) with its tabular description (steps 5-7):

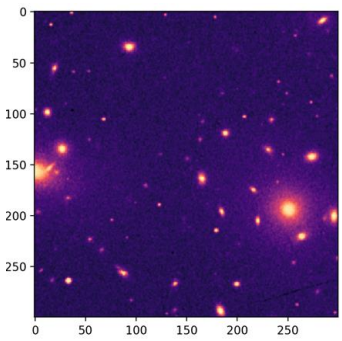


Image scaling:	Log, values range from 696.0 to 15435.0
Image size(degrees):	0.14166666 x 0.14166666
Image size(pixels):	300 x 300
Requested Center:	coma cluster
Requested Center:	194.9529, 27.980600000000003
Coordinate System:	J2000.0
Map projection:	Tan

We use the Astropy package to display the structure of the FITS file of the Coma cluster image (step 8):

```
>>> fits.info(image_file)
```

Filename: coma_DSSred.fits

No.	Name	Ver	Type	Cards	Dimensions	Format
0	PRIMARY	1	PrimaryHDU	138	(300, 300)	float32

Finally we show the summary of the POSS survey which imaged Coma cluster:

First Digitized Sky Survey: Red Plates

Short name[s] used to specify survey:DSS1R,DSS1 Red

Description
This survey is the POSS1 Red plates from the original POSS survey. It covers the sky north of -30 degrees declination.

Provenance	Data taken by CalTech Compression and distribution by Space Telescope Science Institute.
Copyright	CalTech, National Geographic Society. Full copyright notice
Regime	Optical
NSurvey	1
Frequency	461 THz
Bandpass	437-491 THz
Coverage	North of -30 degrees declination
PixelScale	1.7"
Units	Scaled densities
Resolution	Depends on plate. Typically 2"
Coordinates	Equatorial
Projection	Schmidt (distorted Tangent plane projection)
Equinox	2000
Epoch	1945-1958
Reference	Lasker, *et al*., 1990, *A.J*. 99 Characteristics of the DSS surveys are summarized in this document.

In this case, data preprocessing (steps 10-12) is related to Image Filtering Techniques. The principal objective of Image Filtering is to process an image so that the result is more suitable than the original image for a particular application. Filtering is a neighborhood operation, in which the value of any given pixel in the output image is determined by applying some algorithm to the values of the pixels in the neighborhood of the corresponding input pixel.

However, there are mainly two approaches towards image enhancement: Spatial Domain Approach and Frequency Domain Approach. Here we use Spatial Domain Approach, which refers to the image plane itself and involves direct manipulation of the pixels of an image.

Modeling

A common way to manipulate an image in order to highlight features that might not be obvious at first glance, is to modify the pixel values by applying a filter-function to the image. The way these filter-functions are applied is to replace the value of each pixel by another value that is related in some way to the values of surrounding pixels. For example, a $maxFilter()$ function might replace each pixel value by the maximum pixel value in a 3×3 or 5×5 box surrounding the pixel (the pixel itself is also included). Similarly, minFilter() would replace each pixel by the minimum value in the box. We are interested in filtering out some diffusion emission from the Coma cluster image in order to get point sources.

Now we display the Coma cluster image as a float numpy array:

minim = minFilter(image, filt)

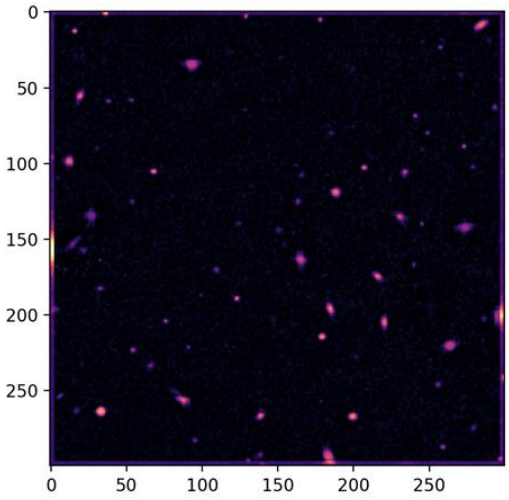
Now let us make the filter size 7×7 pixels, and ignore the edge of the image where the filter would run over the edge. We will first manipulate our image with minFilter and maxFilter as follows:

open = maxFilter(minim, filt)

To filter out diffuse emission and display image with point sources we use the following Python command:

plt.imshow(image - open)

Here is a Pythonic view of our Coma cluster image in magma colormap, but with removed diffusion emission, with isolated point-sources:



Deployment

Whether we are aware of it or not, computer vision is everywhere in our daily lives. For one, filtered photos are ubiquitous in our social media feeds, news articles, magazines, books, sciences – everywhere! Most applications in computer vision and computer graphics involve the concept of image filtering to reduce noise and/or extract useful image structures. MIN and MAX filters have been used in contrast enhancement and normalization, texture description, edge detection, and thresholding in noisy images across different disciplines. Spatial filters can be used also in medicine to filter out EEG signals which are masked due to diffusion emission caused by skull tissues; also, detection of signals in aquatic ambients due to high diffusion of ambient relies on upgraded filters of this kind.