AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ INFORMATYKI, ELEKTRONIKI I TELEKOMUNIKACJI

KATEDRA INFORMATYKI

PRACA DYPLOMOWA MAGISTERSKA

Falls prediction using deep learning time series analysis techniques

Przewidywanie zasłabnięć używając technik analizy szeregów czasowych i głębokich sieci neuronowych

Autor: ***inż. Maciej Błaszczyk***
Kierunek studiów: ***Informatyka***
Opiekun pracy: ***dr inż. Maciej Wielgosz***

Kraków, 2020

Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „ Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.", a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.) „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej „sądem koleżeńskim", oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

……………………………………..

Pragnę złożyć serdeczne podziękowania moim rodzicom za udzielone wsparcie. Wyrazy uznania również należą się Agacie za wiarę, motywację oraz pomoc w zakresie poprawności lingwistycznej.

Składam szczególne podziękowania mojemu Promotorowi, dr inż. Maciejowi Wielgoszowi za życzliwość, cenne uwagi merytoryczne, wszechstronną pomoc oraz poświęcony czas.

Contents

Introduction

Nowadays, research and development of artificial intelligence has become widespread in every field of science. The emerging trend of completely replacing man with a machine poses new challenges to create a human-like being in the context of decision-making skills. The human ability to perceive the world (the environment in which it operates) and the ability to learn is a major challenge when trying to create beings similar to humans. Joint decision-making might be better than in the case of a decision by a single individual. A system as a structure composed of agents who have the ability to solve tasks based on decision-making. In this work, various approaches to agent learning process will be considered to create final prediction. By appointing agents-experts in a given field by modifying their attributes of which they perceive the environment, there is a chance to get a better prediction model.

In the bibliography, there will be modern state-of-the-art papers described multi-agent systems, artificial intelligence used in these types of systems and some most important milestones of this field of knowledge as well as some psychological works about human behaviour.

1.1 Goal of the thesis

The goal of this work is to research and analyze the strategies of learning in agent systems. After analyzing actual solutions which are created over multi agent systems would be considered style of learning and perceving environemnt by each agent. In these work will be used advanced ensemble learning techniques. Different types of approaches will be used to construct the final prediction model. What differentiates this work from other projects, is mainly the approach to learning for each agent. An environmental overhead that limits the perception of agents, making them specific experts in a given field As part of the work will be the creation of simulation groups having human traits whose task will be to take the best decision. Strategies of using agents in service are discussed, in relation to the imposed conditions by the system environment. The creation of the model by the agent with limited ability to perceive the environment will be checked for many combinations of available machine learning algorithms. What's more, combining these models, including those that proved to be relatively inferior, will improve the final predictive model. Finally, the problem under consideration will be a set of data determining the prices of plots (including houses) in the USA based on parameters determining: geographical location, size of the property, condition and construction of the house.

1.2 Elo

siema elo co tam

Machine learning in multi agent system

In this chapter, it will be presented an overview of the state of the art in the areas related to the topic of this work. In first subsection is a description of several types of machine learning used in agent systems. Learning process of agent is also demonstrated. Various machine learning approaches (supervised, semi-supervised, unsupervised, reinforcement) were quoted together with the methods and results of learning by agents.

Next subsection in this chapter has been devoted to the explanation of ensemble learning. State of the art of this thesis is focused on: stacking, boosting, bagging techniques. The application of these methods and important conclusions that were formulated during the experiments were quoted.

Then, in following subsection there is a definition of regression and classification algorithms. Focused mainly on: Decision Tree, Support Vector Regressor, K nearest neighbours, Bayesian Ridge. The literature review includes the use of these methods in agent systems and the justification for their use in solving the current problem.

Another subsection is dedicated to the presentment of prepared multi agent system. The architecture of the following system is estabilished, along with a details of created environment. Roles and types of existing agents are exposed. Relations between agents are shown. The way of perceiving the environment (sensors) and the possible interactions are documented.

2.1 Types of machine learning

Nowadays, the subject of data science has evolved that four types of machine learning approaches has been distinguished. Research on artificial intelligence arose from analyzing the behavior of the human brain. Main difference based on the type of data they input and output and the type of task or problem that they are intended to solve. Thus, machine learning algorithms are organized into taxonomy, based on desired outcome of the algorithm. Defined approaches organized in the following groups [14]:

- Supervised learning — where the algorithm maps incoming data (input) to desired output,
- Unsupervised learning — this approach models a set of inputs (unavailability of labeled examples),
- Semi-supervised learning — labeled and unlabeled examples ad combined in order of creation an appropriate function or classifier,
- Reinforcement learning — where algorithm is trained using a system of reward and punishment. Every action has impact in the environment that generates a feedback that guides the learning method.

The variety of tasks that are solved by artificial intelligence forces a broader view of the issue under consideration. The essence of this is the proper selection of existing methods of artificial intelligence in order for the model created to be the best.

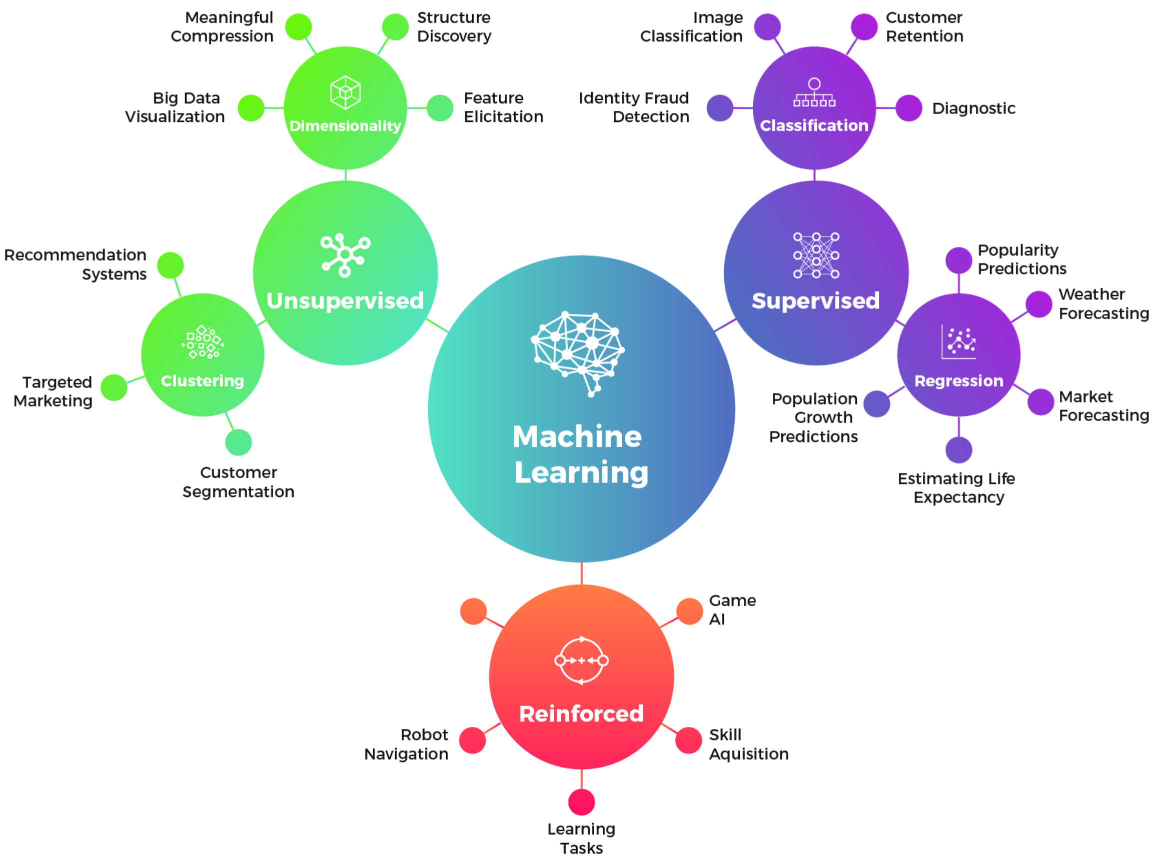
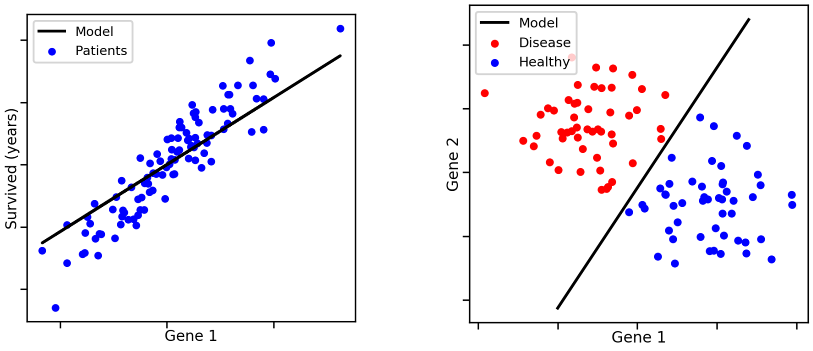


Figure 2.1: The application of artificial intelligence along with the division of machine learning into these branches [15].

On Figure 2.1 are shown examples of different approaches of using artificial intelligence. Machine learning algorithms have found wide application in scientific world. Applying artificial intelligence in systems has been popular recently. Nowadays, collecting data takes place everywhere. In this way, the information collected can be used to solve and support decision-making process. Along with the development of issues related to the subject of AI, many extensions of classic machine learning methods have been created. The technique chosen for the solution depends on the type of data and is often supported by another learning method. However, the use of machine learning does not always bring the intended results. This is done, e.g by the lack of adequate data (insufficient quality or quantity), so there is a need to use other methods known to date.

Supervised learning

Supervised learning is a learning model built to make prediction with a given an unforeseen input instance. Known input data set and its mapped output values are taken to learn (training process) the regression or classification model. Therefore, classification and regression algorithms are used to develop predictive model which describes a given issue. Thus, the model is predictive because it relies on statistical and probabilistic techniques to predict the correct mapping values based on historical observations [12, 34].



(a) Linear regression

(b) Object classification

Figure 2.2: Regression and classification as a supervised technique. Epidemic spread by distinguishing between smaller sub-tasks [5].

Mostly depends on problem category which needs to be solved supervised learning provides various techniques of creating predictive model [8, 11]:

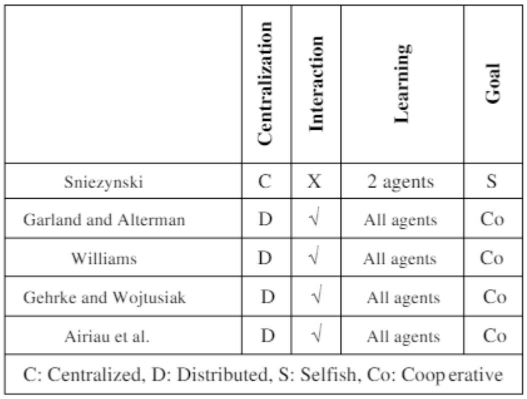
1. Regression techniques — predict or forecast continuous responses. The goal is to estimate the correct output, given a feature vector. There are several types of regression concepts: simple linear regression (presented on figure 2.2), polynomial regression, support vector regression, decision tree regression, random forest regression

2. Classification techniques (example is presented on figure 2.2)— the goal is to predict the categorical class labels (discrete, unordered values, group membership etc.) of new instances based on past observations. There are two main types of classification problem: binary classification and multi-class classification.

In [27], Khaled and other authors noticed that applying various supervised machine learning algorithms to the problem is not easy because there is a possibility to provide ”correct” behavior for a given decision situation. In this paper, there is a consideration above multi-agent learning aspects. Furthermore, based on Weiss [24] publication they described aspects for characterizing learning in multi agent system:

1. degree of decentralization — learning centralized or decentralized,
2. interaction-specific aspects (by Weiss [24] ”level of interaction”) — nature of interactions,
3. involvement specific aspects — global or local learning,
4. goal-specific aspects — selfish or collective agent goals

As a consequence, they created comparison of supervised learners an the learning aspects based on other related works. Fish bank game is solved by Sniezynski [35] using supervised rule learning. Enhance coordination in environment of heterogeneous agents are done using learning coordinated procedures by Garland and Alterman [22]. Learn individual’s ontoligies in the field of semantic webs using inductive learning algorithms was under consideration by Williams [40]. Gehrke and Wojtusiak contemplated at route planning using rule induction algorithm [23]. Decision tree learning was used to support plan applicability testing, through adding learning capabilities into BDI model by Airiau et. al [13]. Hence the conclusion about supervised learners and learning aspects are formulated in presented table.



	Centralization	Interaction	Learning	Goal
Sniezynski	C	X	2 agents	S
Garland and Alterman	D	√	All agents	Co
Williams	D	√	All agents	Co
Gehrke and Wojtusiak	D	√	All agents	Co
Airiau et al.	D	√	All agents	Co

C: Centralized, D: Distributed, S: Selfish, Co: Cooperative

Figure 2.3: Supervised learners vs learning aspects [27].

In [33], Sardinha, Choren, Milidiu and Pereira de Lucena proposed a methodology for introducing machine intelligence in multi-agent system. As a case study is presented Trading

Agent Competition. They implemented methodology which reduces the complexity of modeling, implementing and evaluating learning techniques in large scale multi agent-system. This approach provides an improvement of perfmormance about 98% due to the introdution of supervised machine learning algorithms.

Semi-supervised learning

Semi-supervised learning is used to perform certain learning tasks using labeled and unlabeled data [21]. In fact, it permits harnessing the large amounts of unlabeled data which exists in many use cases in combination with typically smaller sets of labeled data - hence this learning approach is useful when the cost associated with labeling is too high to allow a fully labeled training process. Therefore it can be used with methods such a classification, regression and prediction.

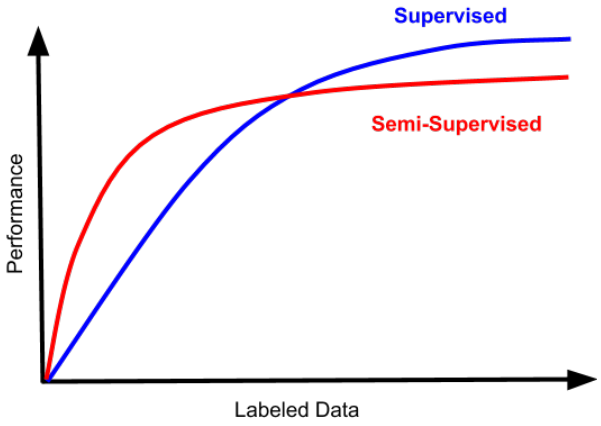
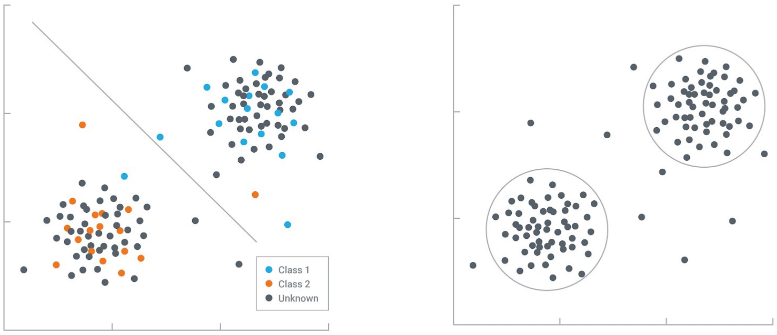


Figure 2.4: Training process performance depends on size of labeled data on the example of blockchain data set [7].

In paper [37], authors have done research on analyzation and predicition human flow. They constructed a multi-agent system which simulate and generate prepared data of people flows around a large facility. Different approach to learning (deep learning) were benchmarked. Simulation data and real data were used in the learning phase, so learning can be considered as semi-supervised learning.

Unsupervised learning

Unsupervised learning is completely different approach than it was described in previous subsections. Most or all of the data cannot be presorted or preclassified beforehand, so the complexity of machine learning algorithm is growing dramatically causing the process more and more time intensive. However, this makes unsupervised learning seductive in applications where data is cheap to obtain, but labels are either expensive or not available. The main assumption of the approach is to discover the structure through common elements in data. Popular techniques include clustering and dimensionality reduction [41, 36].



(a) Supervised learning [6]

(b) Unsupervised learning [12]

Figure 2.5: Differences in approach to learning on the example of classification objects.

In [20], authors proposed usage of unsupervised learning injected into multi-agent system for Checkers game whose architecture is based on adaptive and multi-layer perception neural network. Besides there are several types of agents so the agents are trained for a distinct stages of the game. Each agent corresponds to a multi-layer perception neural network whose weights are updated by Temporal Difference methods [9].

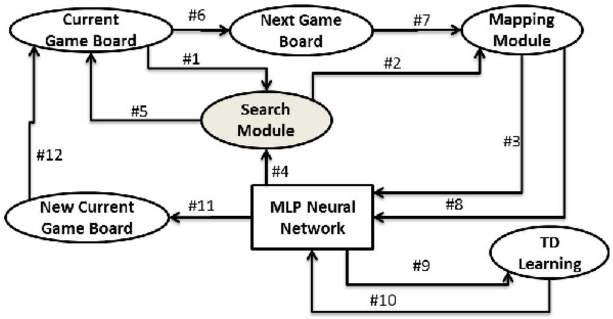


Figure 2.6: Learning process for each agent [20].

In fact, at the late game each agent is an expert in a particular profile (have been trained to become an expert) of endgame board-state. On the figure 2.6 is presented learning process for each agent. As a conclusion: creating environment where agents have a chance to become an expert could have positive influence on working entire system and using an adaptive neural network improve overall performance of the multi-agent system.

Reinforcement learning

This learning approach has three primary components: the agent (the learner or actor), the environment (everything the agent interacts with) and actions (how agent interact). Reinforcement learning is concerned about an agent that interacts with the environment and learns an optimal policy by trail and error [29]. Feedback loop based on which the agent is rewarded is shown on figure 2.7. The reward received is defined by the *reward function*, which maps state/action pairs to the real number [4]. The objective is for the agent to choose actions that maximize the expected reward over a given amount of time.

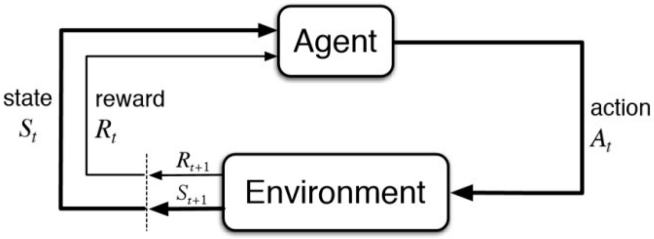


Figure 2.7: Basic idea and elements involved in a reinforcement learning model [29].

In [38], Deepak and Kulkarni proposed a novel approach to multi-agent cooperation for dynamic product availability in the market. Agents cooperate with each other to improve possible profits, thus it is said that retailers learn cooperatively depends on the actual situation. The problem turns out to be Markov decision process model because of the dealer's inventory strategy, refill period and entry procedure of the customers.

Cooperation methods of cooperative reinforcement learning of each agent have been proposed:

1. group method — rewards are distributed in a sequence of steps,
2. dynamic method — for each action rewards are distributed,
3. goal-oriented method — when the agent achieve the goal-state then follows reward distribution.

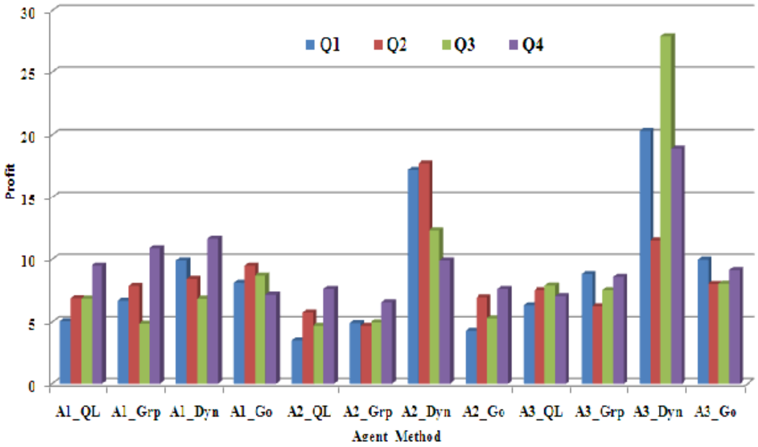


Figure 2.8: Quarterly profit earned by all sellers using four methods of learning [38].

As a conclusion from the article, is that cooperation methods gives a healthy performance in high-density, incompletely and composite situation, thus methods are able to guarantee best rewards through learning process and change with a group in incomplete action strategies.

2.2 Ensemble learning

Ensemble learning is a machine learning paradigm where multiple learners are trained to create base model [43]. Ensemble learning attempts to construct a set of hypotheses and combine them to use, but in contrast to the classical approach of machine learning, in which learners in one case, they learn one hypothesis based on the training data. Applying ensemble methods can boost weak learners to strong learners. Weak learners are better than random guess but cannot make accurate and satisfactory predictions. According to the hypothesis that weaker learner will strengthen potenially more promising learners in the decision-making process.

Considering how to create base learners, ensemble methods can be divided into two groups:

1. sequential ensemble methods - base learners are generated sequentially,
2. parallel ensemble methods - base learners are generated in parallel.

There are common types of ensembles:

- Bayes optimal classifier
- bootstrap aggregating (bagging)
- boosting
- Bayesian model averaging
- Bayesian model combination
- bucket of models
- stacking

In ensemble techniques, there are many taxonomies [25]. Under consideration is a group of methods based on data resampling. Consequently, one method from this group generates different training sets to obtain unique learner.

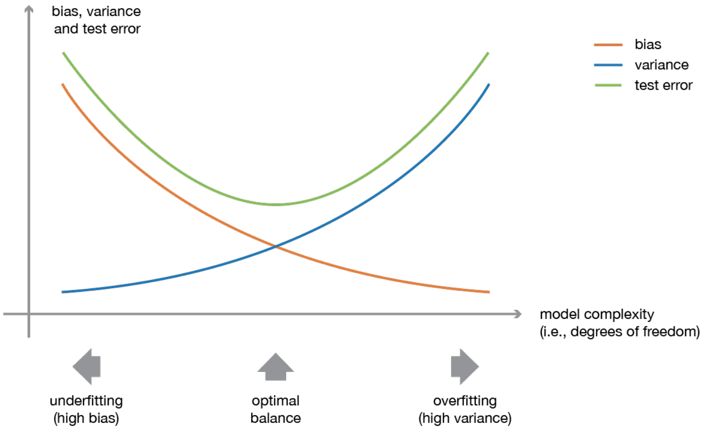


Figure 2.9: Bias-variance trade-off [3].

Figure 2.9, demonstrate a trade-off between bias, variance, test error values and model complexity. In fact, the requirement to solve the problem using created model is to have enough flexibility to resolve the underlying complexity of the data, but also having not too much degrees of freedom to avoid high level of variance and be more robust. Described techniques were mainly used as a meta-algorithms which combine several machine learning algorithms in order to decrease variance (bagging), decrease bias (boosting) and improve predictions (stacking).

Stacking

Stacking as a ensemble learning method is popular to achieve greater predictive accuracy using combination of lower level base learner (In fact it is one single high-level base learner created as a combination low-level learners) [39]. Collecting the output of each base learner into a new set of data is a first step in this approach. Care is taken to ensure that the base learners are formed from a batch of training data. A new data is treated as a data for another learning problem.

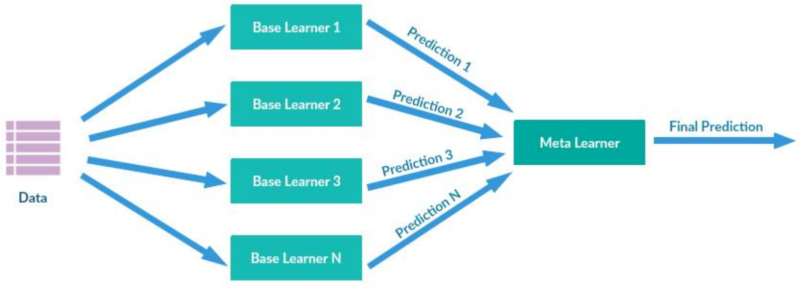


Figure 2.10: Meta learner as a combination of base learners [10].

In the fig. 2.10 is shown data flow using stacking as an ensemble method. Different weak learners are fitted independently. However, using an extension of stacking is getting more and more popular. Modified version is called multi-level stacking, consists doing stacking with multiple layers.

Bootstrap Aggregating (Bagging)

Bagging is an ensemble method for improving unstable estimation or classification schemes. Considered method is about splitting training data into smaller constant parts. For a given training set of size *s*, bagging draws *t* random instances with replacement (i.e. using a uniform distribution) [28, 17]. Building ensembles using bagging method based on different part of training data is done with learning by single method.

In [16], Breiman defined bagging as a variance reduction technique for a given base procedure, mostly a decision trees or functions which fitting a linear model. Breiman also considered about number of Bootstrap replicates. Several number of replicates where benchmarked for classification and regression, however number of enough replicates depends mainly on the data set(size and number of labels).

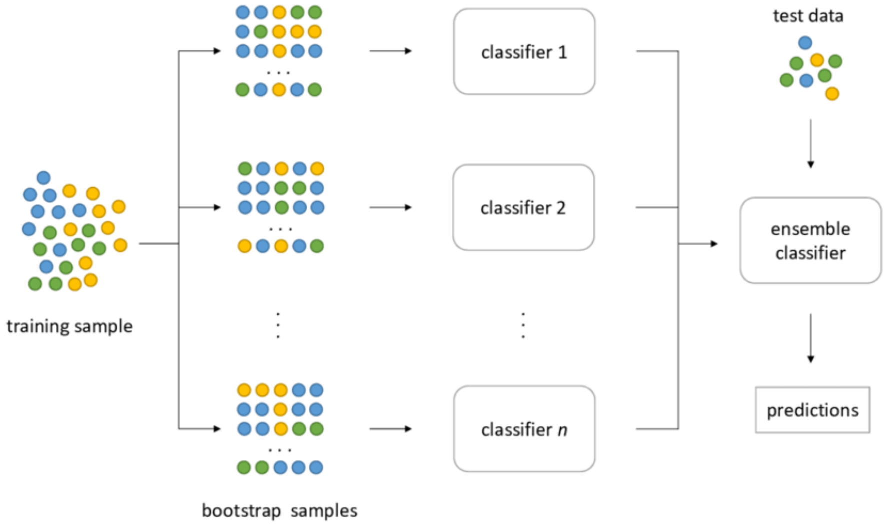


Figure 2.11: In bagging, training instances can be sampled several times for the same predictor [1]

Figure 2.11 gives a sample how Bootstrap Aggregating works on a imaginary set of data. Breiman (1996) showed the effectiveness of using these ensemble method. Bagging is effective on ”unstable” learning algorithms where making small changes have large influence on predictions. In fact, Breiman in 1996 claimed about unstable learning algorithms and nowadays neural networks and decision trees are simple examples of unstable learning algorithms.

Table 2.1: Bagged Misclassification rates. Results appeared for the 10, 25, 50, 100 numbers of bootstrap replicas used on waveform dataset [16].

No. Bootstrap Replicates	Misclassification Rate
10	21.8
25	19.4
50	19.3
100	19.3

In table 2.1 are shown the results of using bootstrap replicates and its misclassification rates. Breiman observed, that getting most of the improvement is using only 10 bootstrap replicates. The unbagged rate is 29.1. Consequently, it can be seen that the trade off between the efficiency and complexity of the learning algorithm and the profit in the form of a misclassification metric is when increasing the number of replicas we do not achieve much improvement. It is better to use less than 25 replicas on the used data set.

Boosting

Boosting keeps track of the performance of the learning algorithm and concentrates on instances that have not been correctly learned. This ensemble method also uses different subset of training data. Rather than random choosing the training instances using uniform distribution, this approach chooses the training subsets to favor the instances that have not been accurately learned [28]. This approach considers homogenous weak learners and learns them sequentially in a very adaptative way to combines them following a deterministic strategy. In fact, after several steps, the predicition is done by taking a weighted vote of predictions of each classifier, with the weights which are proportional to each classifier's accuracy metric on its training set.

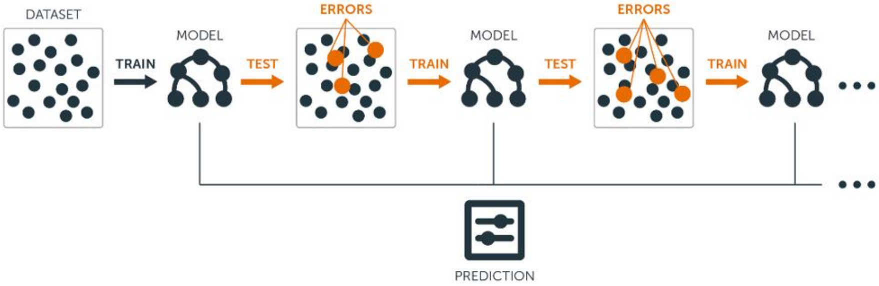


Figure 2.12: Data flow of gradient boosting machine [2].

Boosting as a ensemble method focus at reducing bias, so the base models are often considered for boosting models with low variance and high level of bias. For comparison to bagging, this approach can not be done in parallel. However, it requires careful tuning of different hyper-parameters. Boosting involved into powerful forms i.e.: Arcing[18], AdaBoosting[42], Gradient Boosting [30], XGBoost [26].

2.3 Multi agent system

Multi agent system also known as ”self organized system” is a computerized system constructed by many agents who interact with each other. Multi-agent systems can solve many complex problems that are a may be insoluble using monolithic systems.

The system inherits features from active autonomous agents that make up the agent system. A group of agents creates a system and has its own properties such as: autonomy, rationality, activity, goal orientation, adaptability and mobility.

The multi-agent system strives to achieve the set global goal and takes place in a *decentralized* mode. In the consequence of autonomously decisions made by agents, the set goal is achieved. In general, no agent is favored or privileged, and therefore cannot enforce actions on other entities. The ability to *self-organize* (or reorganize), which is done to a certain extent by agents, is the reason for pursuing the set goal. Mechanism of synchronization and coordination of activities in the system is not strictly defined (it is not imposed externally or centrally). Therefore, *asynchrony* is an important feature that describes this type of system. System under consideration is characterized by *openness* - it means that an agent can join or resign from participation in the pursuit of global goals at any time. There is a need to implement mechanisms to respond by this feature (area of communication and organization). It is worth noting that the *interopability* feature reinforces the property of openness of the agent system. As a result, agents who do not come from this project (they were built up in another production process) may join such a system. New quality is brought by the mobility of agents. Thanks to the system *dispersion*, all operations can be spaced apart in space, which allows easy scaling and load balancing. It is worth noting that the future agent's state is *difficult to predict*. This problem occurs as a consequence of the fact that agents are autonomous entities. In summary, even knowledge of the functioning of autonomous agents does not guarantee full knowledge and ability to predict the effects of interactions in the system.

Widely used approach to system design and evolution necessitated various modifications classic agent system. Many new extensions based on (MAAI) multi-agent systems of artificial intelligence have been proposed, for example: DAI - Distributed Artificial Intelligence System. DAI and other modifications have been already applied for simulating societies, Modelling Social Systems, in the field connected with climate, energy, epidemiology, conflict management and much more. These system designing is highly popular where increasing demands are placed on the amount of data processing (big data), processing time, scaling, high availability and ability to make autonomous decisions.

2.3.1 Learning by the agent

As Russel and Norvig [32] have designed structure of learning agent architecture to four elements:

1. learning element,
2. performance element,
3. critic element,
4. problem generator

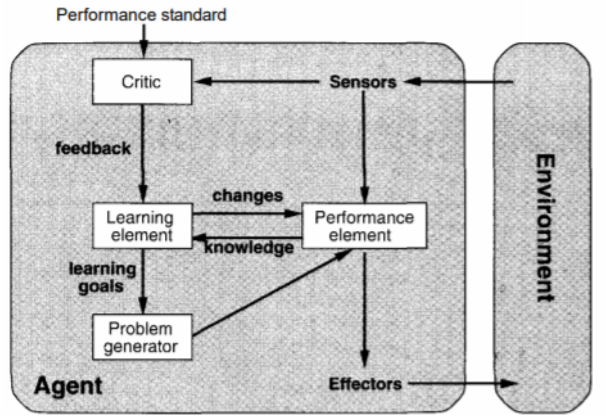


Figure 2.13: A general model of learning agents [32].

On the figure 2.13 are shown these elements and main interaction patters. Agent perceives on every moment and makes a decision depends on performance input [27]. On the one hand the agent uses feedback to determine what should be improved to increase the performance. On the other hand, the problem generator introduce agent to new informative experiences through suggesting actions.

2.3.2 Environment

The appearing 'agent system environment' statement thesis whether the agent system is modelled separately or with it. The answer to the thesis is the approach used to describe the phenomena occurring in the surrounding environment. An object representing the environment in a relatively isolated agent system would perform all these processes inside. Then there is the necessity to place responsibility contact-agents as a boundary between the system and the environment. However, the environment as non-restrictionized gives environmental field depending on the need for its modelling.

When enriching with the agent system environment model, general formal conditions have been imposed:

- the environment of this system is modelled by an isolated system and intentionally specified from the system environment,
- the boundary between the agent system and the environment in question is set up using contact agents and their possible external actions,
- the environment model created accepts changes in its state as a result of agent's actions and offers the opportunity to observe changes in states using other actions,
- if required, the dynamics of the environment is modelled independently. There is a chance to observe the changes taking place in this dynamics.

Environmental properties generate many decisions when designing the agent system architecture for a specific application. The general environment's characteristics of this system mainly apply to the functions of agents that they are to fulfil when designing architecture. There are many aspects to consider when defining the broadly understood term ”environment in an agent system”.

Observability

First of all, the observability of the environment by a particular agent affects his ability to take action. An important aspect is also what an agent can observe and which information extracted from the environment is actually needed to take the appropriate action. The following observability scale was adopted due to the level of the agent's perception of the environment:

- full observable - when the agent can get full and complete information needed for his action mechanism. The agent operates in the conditions created so-called ”full information”,
- partial observable (partial unobservable) - when the agent does not work in conditions where it has access to full information. Through this type of approach, an agent can

make suboptimal, but still practical decision that are crucial for the proper functioning of the agent system,

- unobservable - when agent does not have access to the information that he needs to perform the intended action. This agent can complete the necessary knowledge using another agent who has the ability to observe the environment,
- relative observability - occurs when the environment is unobservable for all agents and they must derive knowledge through contact agents. Relative property is very often overlooked in the literature.

In the presented system only by communication it is possible to solve problems related to the lack of adequate environment's observability. In fact, the terms accessible/inaccessible are used instead of relative observability. However, it has been noted that, among others, through relative observability, agents organize themselves into the system and make suboptimal decisions.

Stability

Secondly, general characteristics of the environment also describes the type of environment. Resolving between dynamic and static environments is possible by determining if agent actions are able to change the state of the environment.

The environment is *static* for each agent in this system if these agents do not change the state of the environment. It was stated that the static statement will clearly determine how the environment works (no relative static).

The agent system in which the *dynamic* environment operates places high demands on the designed system. Appropriate precision used during the prediction of the environment evolution is the key to the meet requirement when using a dynamically changing system environment. In the described situation, we are not able to clearly determine whether the agents made optimal decisions.

Determinism

If there is no uncertainty associated with the new state of the environment and that the defined action brings a clear and properly describe effect, then from the agent's point of view the environment is *deterministic*.

On the other hand, when the agent's interactions and actions are able to cause an unknown response of the reaction model, then the environment is defined as *non-deterministic*. This is due to the poor precision of predicting response to agent interactions. The consequence of this approach may be that the taken actions will be unfavorable or incorrect for the agent.

Modelling

The discussion about whether the environment is modelled as *continuous* or *discrete* corresponds to whether the defined agent perception actions sample the state of the environment. Typically, a discrete agent structure is used (hence discrete modelling of the environment), even if there are problems with the quality and reliability of the mapping.

Episodicity

This feature facilitates the design of agent systems. During the episode, the agent performs closely related activities. The steps in the algorithm of conduct are short and there are few (except when learning occurs), then there is no need to hold and analyze information about past episodes.

2.4 Proposed multi-agent system

2.4.1 Architecture

When designing the architecture of the agent system, it is important to correctly interpret the set functionality requirements. Formulating the purpose of scientific work did not explicitly set the requirements that the system must meet. An additional challenge is to be able to anticipate additional problems before they occur. Architecture must meet all assumptions in accordance with the definition of agent system with autonomous units. The following is the architecture of the agent system that meets the requirements.

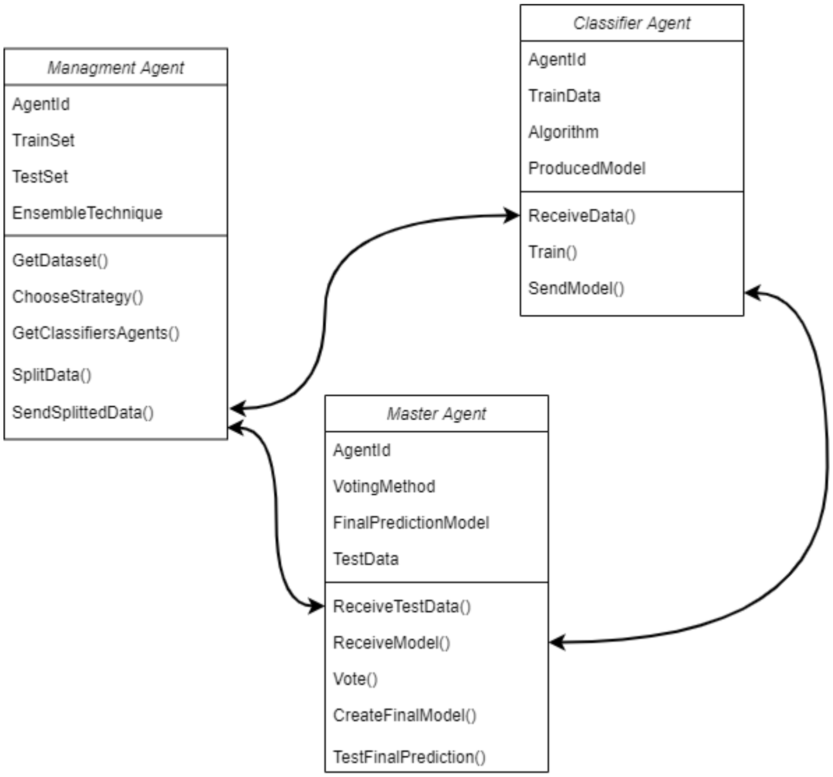


Figure 2.14: Architecture of proposed agent system.

As shown on the figure 2.15, system comprises three types of agent: DataManager agent, Classifier agent, Master agent. The proposed agent architecture present data flow and end to end process of creating final prediction model. In presented architecture, each agent has unique objectives and responsibilities. Only working in collaboration, these three types of agents would achieve an overall goal which is accuracy in predicting house price.

Communication

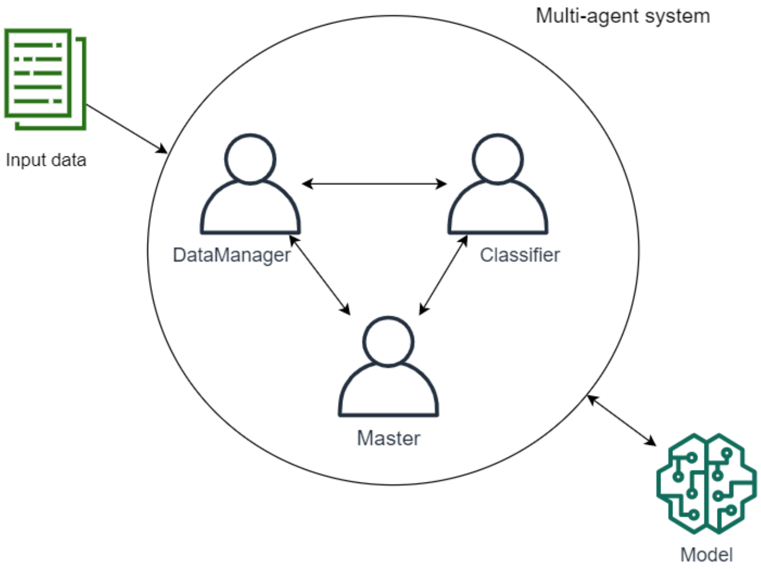


Figure 2.15: Process of achieving the goal due to communication with 3 classifiers agents.

Environment

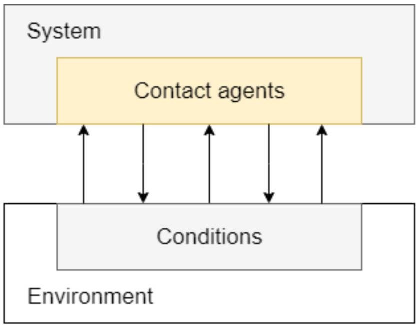


Figure 2.16: Architecture of proposed agent system.

2.4.2 Agents specification

Presented multi agent system is composed of three main types of agent: management agent, classifier agent, master agent (Fig. 1). Below is presented description of responsibility and interactions for each agent. For every agent of designed system is presented data flow and possible permissions.

Managment Agent

The agent receives information (data set) that creates the environment in system. Then he decides which part of the data set to devote to the model's training purposes and which part to extract for testing the obtained prediction. Main responsibility of this autonomous unit is to distribute the training set among classifier agents.

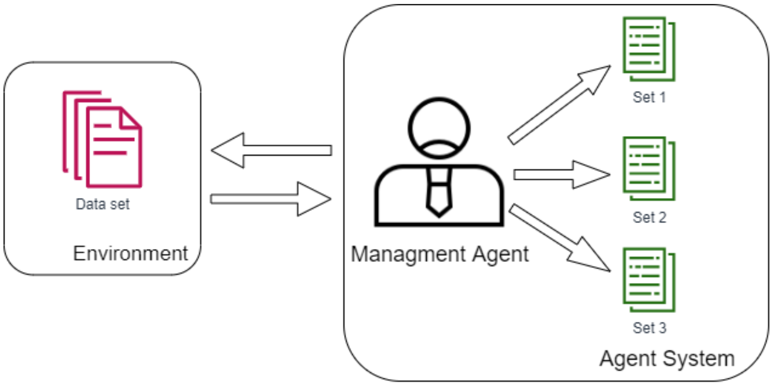


Figure 2.17: Process of achieving the goal due to communication with 3 classifiers agents.

As can be shown on Fig. 2.17, managment agent is a contact agent. It is an interface of other agents to perceive the environment. When choosing a strategy for splitting a data set, the agent decides whether:

- if each agent receive the same part,
- whether information is to be equally divided for each individual,
- if the agent get full information from a given part (full number of features describing a given observation).

Classifier Agent

This type of agent is the only one on the system that can occur (and even should) in many cases. In the proposed solution, classifier agent does not have direct access to the environment. Hence, it uses contact agent - management agent that gives him right permissions to perceiving environment. It is treated as a black box in which it receives data on input and a model on output.

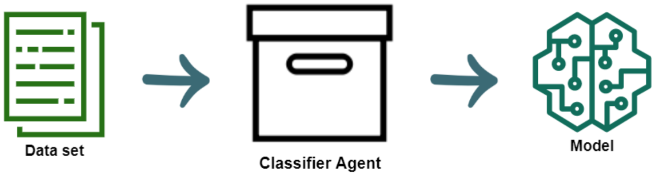


Figure 2.18: Main concept of the classifying agent is the encapsulation machine learning algorithm.

As shown on the fig. 2.18, classifier agent does not know if the collection he receives is a complete set. Completeness of the information received says that the agent has get the full number of features for the obtained observations. However, he can only get a certain part of the collection. He does not need to get information from other agents of the same type (who can implement a different type of machine learning algorithm) regarding their perception of the environment. It is planned to perform experiments using following machine learning algorithms:

- K nearest neighbours (KNN),
- Bayesian Ridge,
- Logistic Regression,
- Decision Tree.

Trained model is a result (artifact) of the agent interaction. Then, it communicates with the master agent in order to send him the created model.

Master Agent

There is only one instance of this stand-alone agent in the system. Responsibility of this agent is to communicate with classifiers agents in order to receive a trained model from them. Then, he sums up all the trained models to create one final prediction model.

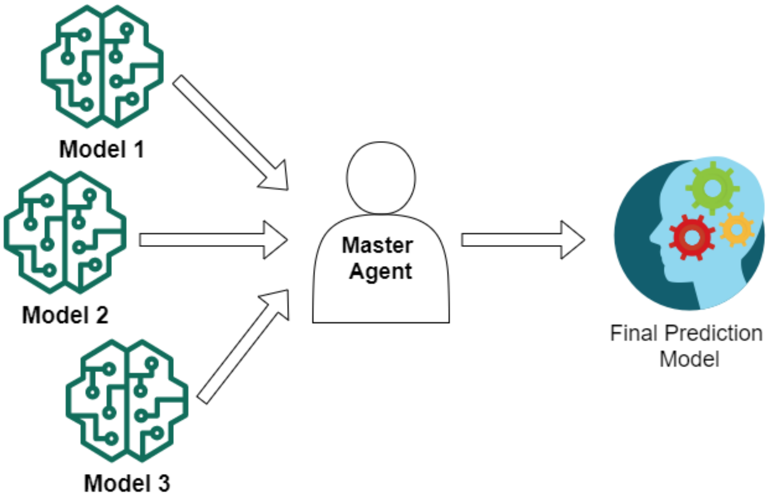


Figure 2.19: Process of achieving the goal due to communication with 3 classifiers agents.

As shown on the Fig. 2.19, master agent as an autonomous unit is able to sum the received models. In order for the quality of the received artifact to be as high as possible, master agent can sum the models according to the adopted strategy. So far, several methods have been proposed:

- majority voting - every model instance is important only if receives more than half of the votes. If none of the prediction get more than a half of the votes, it is said that ensemble method might not create a stable prediction for this instance,
- weighted voting - modified version of majority voting is done by increasing the importance of one or some models,
- simple averaging - every model has same weight and then average model is calculated,
- weighted averaging - slightly modified version of simple averaging. Prediction of each model is multiplied by the weight, then average is calculated.

Only the *average voting* and *weighted voting* will be examined in experiments. They were chosen because the stated goal of the thesis suggests not to award better quality models. It is not known if a poor-quality predictive model will have a positive effect on the sum with other models.

2.5 Summary

The first section was devoted to a review of the literature related to machine learning issues in agent systems. Quated results and observations were posted that helped to better plan and perform experiments.

Next section focuses on analysing and citing literature where methods of ensemble learning were used. The focus was on three types: stacking, boosting and bagging.

Section contains a theoretical introduction, which has been written based on [19].The first subchapter related to the general characteristics of agent systems was prepared by the paraphrase of general description in 2nd chapter and ”2.6 Characteristics of agent system. Summary” from the cited literature. Theoretical introduction to the definition of the environment in agent system has been completely prepared on paraphrasing subsection ”2.2 Agent System as a relatively isolated system”.

The last section was devoted to the description of the designed agent system. Aspects such as architecture, environment, communication in the agent system and agent specifications have been described.

Experiments

This chapter defined the experiments that are carried out as part of the laboratory. The first part was assigned to the presentation of the data set on which experiments have been made. All labels and the degree of correlation between them were explained. Second subchapter introduce to preprocessing process which was realized. Multi agent system and used learning strategy is demonstrated for each experiment in third subsection. The fourth chapter describes the implementation details: the technology used, the implementation of the multi-agent system and its mechanisms. The fifth section was devoted to the presentation of the results obtained. A number of charts and diagram descriptions have been included.

3.1 Dataset information

The analysis dataset contains house sale prices for King County (includes also Seatlle), Washington in United States of America. Collected data constitute 21613 observations houses sold beetween May 2014 and May 2015.

Labels in dataset

Dataset consists of information of price, date of sale, id of sale and 18 house features. The below labels juxtaposition presents description of 18 parameters specify each property for sale:

- **house** - parameters which describe only house (apartment):

  bedrooms - number of bedrooms,

  bathrooms - number of bathrooms, where 0.5 accounts for a room with a toilet but no shower, *sqft_living* - square footage of the apartments interior living space (m^2),

sqft_living15 - living room area in 2015 (if the renovation was done in 2014-2015),

floors - total floors in house,

view - an index from 0 to 4 of how good the view of property was,

condition - an index from 1 to 5 on the overall condition of the apartment,

grade - an index from 1 to 13, where 1-3 falls short of building construction and design, 7 has an average level of construction and design, and 11-13 have a high quality level of construction and design,

yr_built - the year of the house was initially built,

yr_renovated - the year of the house last renovation,

- **property** - labels which specify property on sale:

sqft_lot - square footage of the land space,

sqft_lot15 - square footage of the land space in 2015,

sqft_above - square footage of the interior housing space that is above ground level,

sqft_basement - square footage of the interior housing space that is below ground level

- **location** - labels correlated with geographical location

lat - latitude (geographical coordinate),

lon - longitude (geographical coordinates),

zip_code - zip-code area the house is located,

waterfront - if property has a waterfront view

Labels corellation

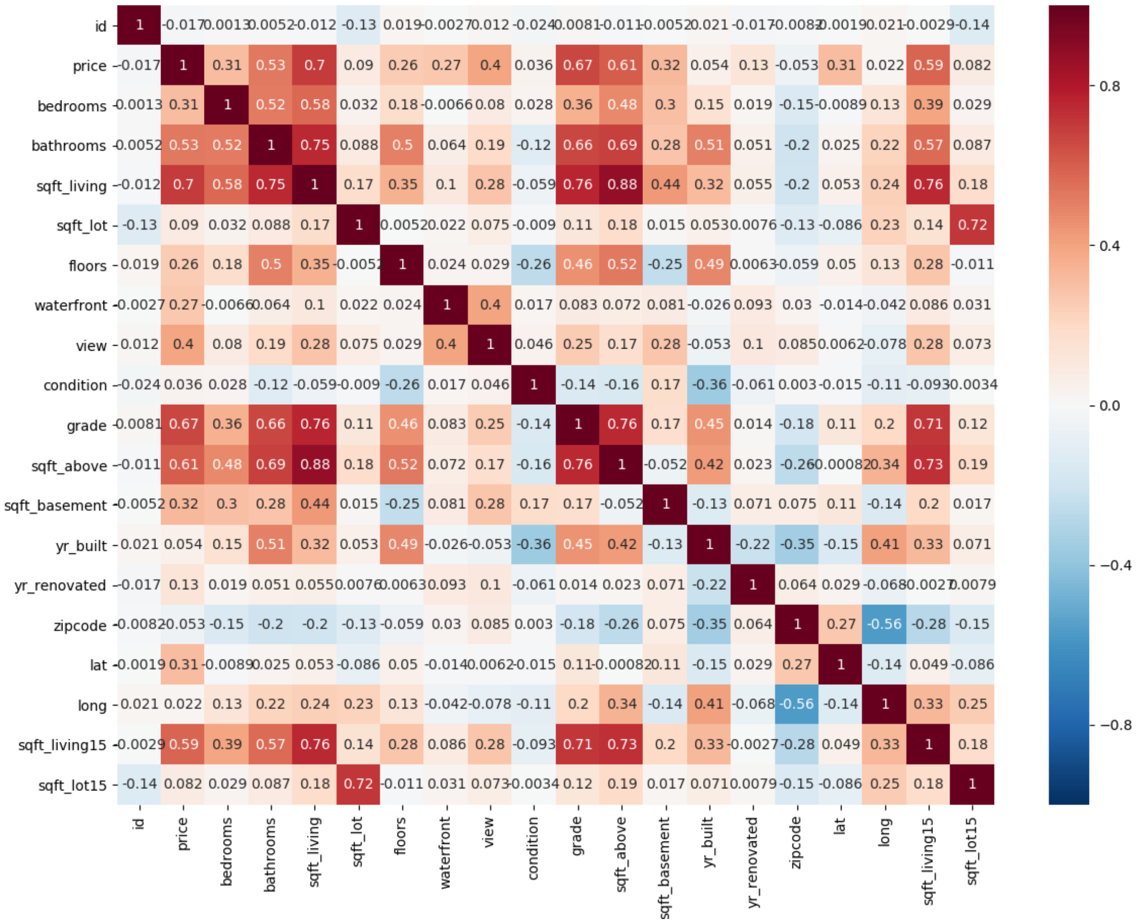


Figure 3.1: Heatmap presents corelation between labels.

3.2 Data preprocessing

3.3 Experiments definition

Agent configurations, environment and learning method has been described for each type of experiment. Experiment definition contains: *Overview* (motivation), *Description* and *Schema*. Test instances differ from each other, among others, approach to learning, the agent's ability to observe the environment and specializations. There are two types of agents in this multi-agent system: Classifier Agent and Master Agent. The method of collecting results and communication between agents is similar for all established experiments. Communication is one-sided. Classifier agents only communicate with the master agent. Each regressor agent builds a training model based on which price prediction is created. The agent sends the created prediction model to the master agent. The master agent builds a general predictive model based on the received models from Classifier Agents.

The main outline of learning and data flow in this system is described in the following diagram:

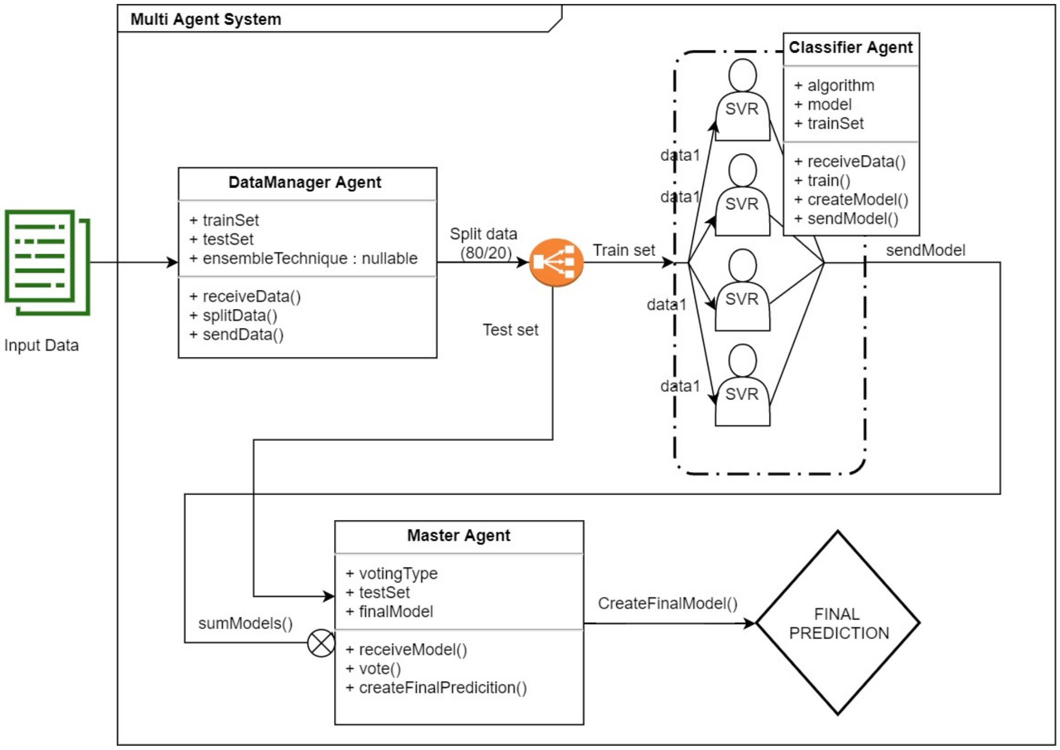


Figure 3.2: The multi-agent system architecture where is presented data flow during classification process.

1. **Experiment**

   Overview

   The purpose of this experiment is to check how the multi-agent system works when each instance of classifier agents is learning from the same part of the data set. The visibility of agents who create a predictive model with the same classification algorithm is not limited. The population of agents in the system is not large (5 species), so as to check whether the potentially weak model created by a single agent can have a positive effect on the master agent final prediction.

   Description

   (a) **Environment**

   In order to check the obtained predictive model in a multi-agent system, the environment is to be created by five instances of agents:

      i. four classifier agents

      ii. one master agent

   (b) **Agents**

   There are two types of agents in the system:

      i. Classifier Agent - each agent implements the same machine learning algorithm. Agents has full information about the environment. Machine learning methods used as a part of this experiment:

         A. Decision Tree Classifier

         B. Support Vector Regressor (SVR)

         C. K-Nearest Neighbours (K-NN)

         D. Bayesian Ridge

      ii. Master Agent as a meta-classifier

   (c) **Dataset**

   The data set is splitted into training and test in the proportion of 80/20. Each Classifier Agent receives the same training set. The master agent uses the test set to classify the final predictive model.

   (d) **Final Prediciton**

   The agent's master vote is done using the arithmetic average. The models are added together without rewarding potentially better predictive models.

Schema

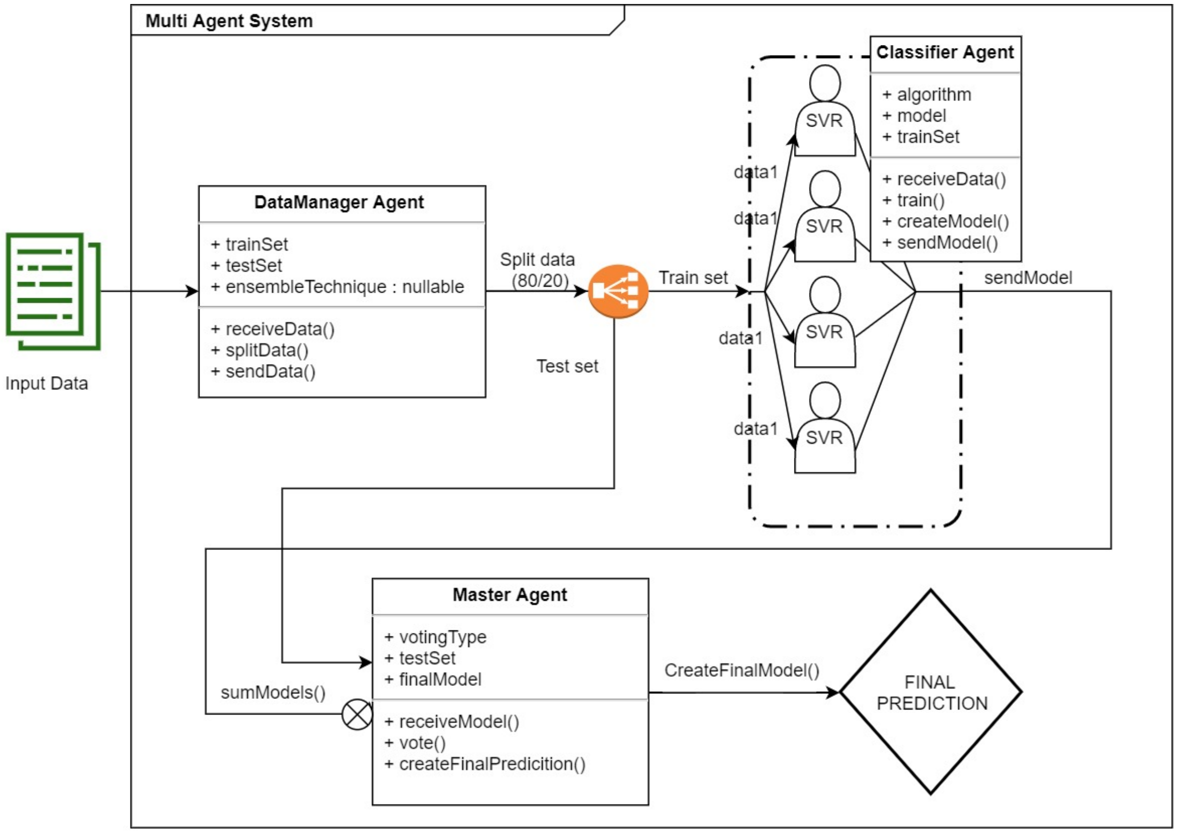


Figure 3.3: Experiment 1 schema. Each Classifier Agent uses *SVR* algorithm

2. **Experiment**

   Overview

   The motivation to accomplish this experiment is to check the impact of the size of the population in the agent environment on the final predictive model. The number of models that are combined in the process of creating the concluding model is related to the size of the population in the agent system environment. Data distribution to agents, machine learning algorithms, and process of creation the final model by master agent are similar as it was described in experiment 1.

   Description

   (a) **Environment**
   The multi-agent system environment is implemented by a large number(9 species) of agents. The learning strategy is implemented by creating a model constructed by the artifacts of average sum of eight machine learning algorithms (methods are duplicated):

      i. eight agents implementing machine learning algorithm
      ii. one master agent to create concluding decision

   (b) **Agents**
   There are two types of agents in the system:

      i. Classifier Agent - each agent implement the same machine learning algorithm. Agent observability of the environent is not limited. Machine learning methods used as a part of this experiment:

         A. Decision Tree Classifier
         B. Support Vector Regressor (SVR)
         C. K-Nearest Neighbours (K-NN)
         D. Bayesian Ridge

      ii. Master Agent as a meta-classifier

   (c) **Dataset**
   Each agent using the machine learning algorithm gets the same part of the data set (80% of the entire set, 20% is intended for the classification of the obtained results).

(d) **Final Prediciton**

The lead agent uses the weighted average method to create the final prediction model.

Schema

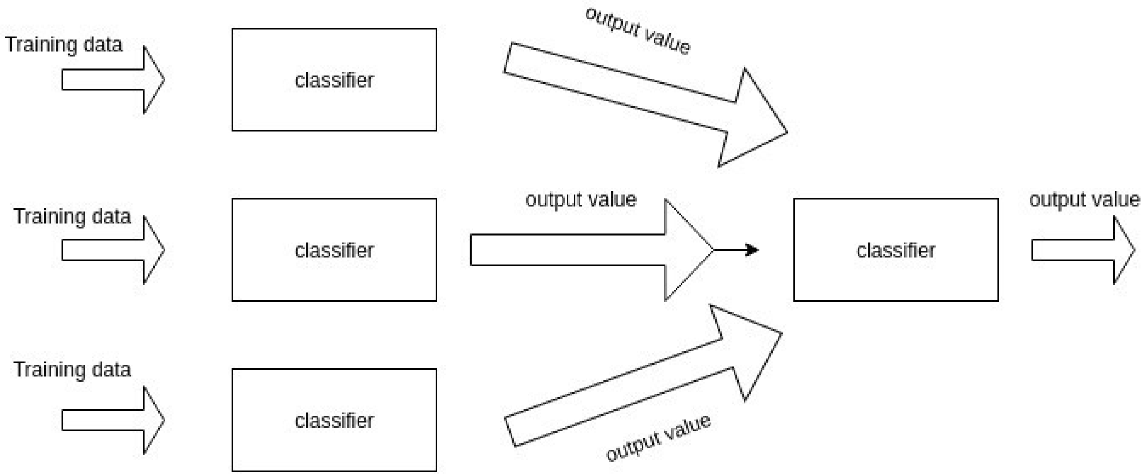


Figure 3.4: Experiment 2 schema

3. **Experiment**

   Overview

   The motivation to conduct the experiment is to analyze the machine learning strategies used in the multi-agent system. The strategy being considered is stacking as an ensemble learning technique.The models will be created by agents which iplements different types of learning algorithms. The impact of theoretically inferior models on the final solution will be contemplated.

   Description

   (a) **Environment**
   The multi-agent system environment is created by five species of agents. The learning strategy is implemented by creating a model constructed by the artifacts of four machine learning algorithms:

      i. four agents implementing machine learning algorithm
      ii. one master agent to create concluding decision

   (b) **Agents**
   There are two types of agents in the system:

      i. Classifier Agent - each agent implement the same machine learning algorithm. Agent observability of the environent is not limited. Machine learning methods used as a part of this experiment:

         A. Decision Tree Classifier
         B. Support Vector Regressor (SVR)
         C. K-Nearest Neighbours (K-NN)
         D. Bayesian Ridge

      ii. Master Agent as a meta-classifier

   (c) **Dataset**
   Each agent using the machine learning algorithm gets the same part of the data set (80% of the entire set, 20% is intended for the classification of the obtained results).

   (d) **Final Prediciton**
   The lead agent uses the weighted average method to create the final prediction model.

Schema

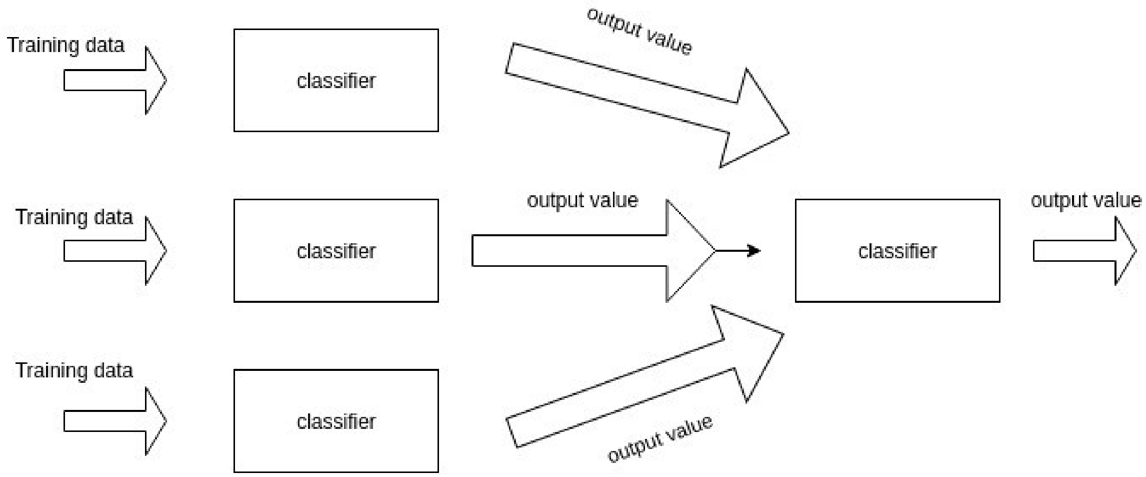


Figure 3.5: Experiment 3 schema

4. **Experiment**

   Overview

   The idea for which this experiment was performed is the use learning strategy which use ensemble learning techniques: bagging and stacking. The perception of individuals in the environment is limited due to the different part of the dataset obtained by the agents (bagging). However, agents have full knowledge of the data they received to make decisions when building the model. Only two types of classification algorithms are used to create ensemble model (stacking).

   Description

   (a) **Environment**
   Agents perception of the entire environment is limited. In multi-agent system enronment is created from:

      i. four agents implementing machine learning algorithm
      ii. one master agent to create ensemble model

   (b) **Agents**
   There are two types of agents in the system:

      i. Classifier Agent - Each agent receive a different part of the training set (perception of the environment is limited). Agent implements one of the following classification algorithms:
         A. Decision Tree Classifier
         B. K-Nearest Neighbours (K-NN)
      ii. Master Agent as a meta-classifier

   (c) **Dataset**
   Data set has been evenly divided into five parts in the following proportions: 20:20:20:20:20. Four of the five parts will be used as training data and the last part will be used to classify final prediction model.

   (d) **Final Prediciton**
   The single isntance of meta-classifier will use average voting when building the final model. Master agent assesses the built model on the last part of the divided data set.

Schema

schemat

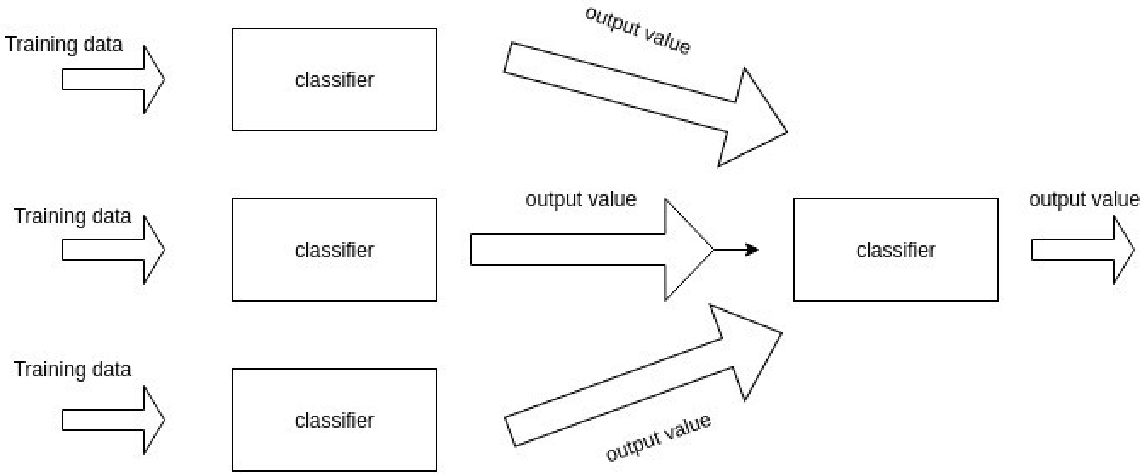


Figure 3.6: Experiment 4 schema

5. **Experiment**

   Overview

   The motivation in this experiment is to observe the decision making process when the agents cannot see the whole environment (partial observability). Agent-based environment is more difficult for each agent who create predictive model. Each agent will be an expert in a different area. Expert groups were formulated based on the correlation of labels (based on the heatmap). Every Classifier Agent implements a classification algorithm that builds a predictive model based on the labels defined in the group.

   Description

   (a) **Environment**

   The environment in multi-agent system is partially observable:

   i. four agents where each is an expert in a given field (based on correlation beetwen labels):

      A. specialization: bedrooms, bathrooms, sqft_above, sqft_living15

      B. specialization: sqft_living, sqft_above, sqft_living15, grade, sqft_living15

      C. specialization: yr_built, long, grade, floors, sqft_living15

      D. specialization: zip code, longitude, latitude, yr_renovated

   ii. one master agent has no limitation of environment observability

   (b) **Agents**

   There are two types of agents in the system:

   i. Classifier Agent - each agent implements one of the following classification algorithm:

      A. Decision Tree Classifier

      B. K-nearest neighbours (K-NN)

   ii. Master Agent as a meta-classifier

   (c) **Dataset**

   The data set is splitted 80/20 - 80% of the entire set to train model(each agent the same part), 20% is intended for the classification of the obtained results.

   (d) **Final Prediciton**

   The master agent uses average voting to build the model predictive model

Schema

schemat

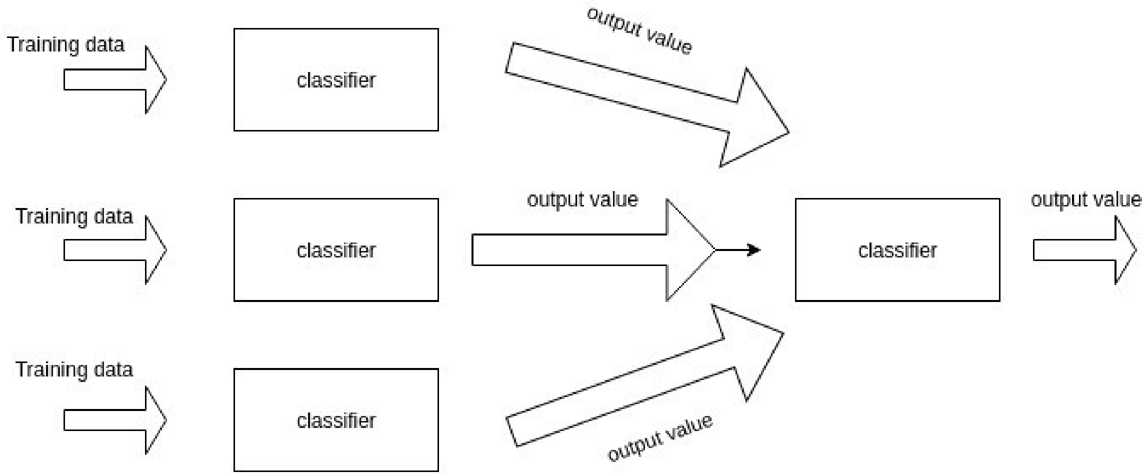


Figure 3.7: Experiment 5 schema

6. **Experiment** o czym ma to być: grupowanie po mojemu

   Overview

   The motivation to accomplish this the final model by master agent are similar as in Experiment 1.

   Description

   (a) **Environment**
   The multi-agent system constructed by the artifacts of average sum of sixteen machine learning algorithms (methods are duplicated):

      i. fifteen agents implementing machine learning algorithm

      ii. one master agent to create concluding decision

   (b) **Agents**
   There are two types of agents in the system:

      i. Classifier Agent - each agent of the environent is not limited. Machine learning methods used as a part of this experiment:

         A. Decision Tree Classifier

         B. Support Vector Regressor (SVR)

         C. K-nearest neighbours (K-NN)

         D. Bayesian Ridge

      ii. Master Agent

   (c) **Dataset**
   Each agent using the machine learning algorithm gets the same ssification of the obtained results).

   (d) **Final Prediciton**
   The lead agent uses the weighted average method to create the final prediction model.

   Schema
   schemat

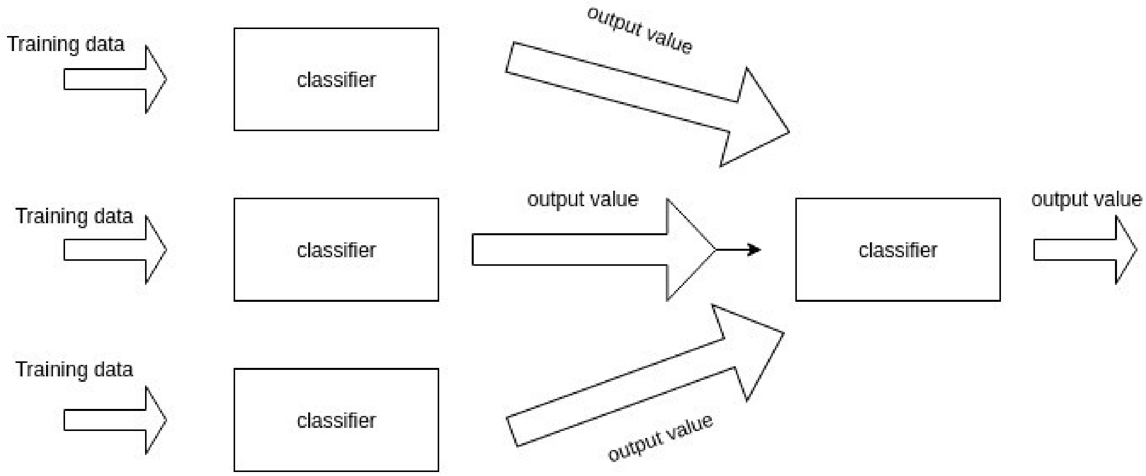


Figure 3.8: Experiment 6 schema

Experymenty chapter Konfiguracje agetnow + uczenia, podejście modelowe Srodowisko uczynić trudniejszym, agent ni widzi całosci srodowiska (obserwowalnosc czesciowa) Stwierdzic co sie dzieje jesli srodowisko nie jest obserwowalne (w pełni) Specjalizacja, sensory etc komunikacja agentów

3.4 Classification of the solution

3.5 Used technology

3.6 Results

Conclusions

List of Figures

List of Algorithms

Bibliography

[1] Bagging in Financial Machine Learning: Sequential Bootstrapping. URL https://hudsonthames.org/bagging-in-financial-machine-learning-sequential-bootstrapping-python/.

[2] Ensemble in Statistics. URL https://mateusmaiads.github.io/ensemble_qualify/#1.

[3] Ensemble methods: bagging, boosting and stacking. URL https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205.

[4] Introduction to Reinforcement Learning. http://www.noussintelligence.com/en/introduction-to-reinforcement-learning/.

[5] Microbiome Summer School 2017. https://aldro61.github.io/microbiome-summer-school-2017/sections/basics/.

[6] Polynomial Regression and Model Generalization in Machine Learning. https://developpaper.com/polynomial-regression-and-model-generalization-in-machine-learning/.

[7] Practical Machine Learning for Blockchain Datasets: Understanding Semi and Omni Supervised Learning. https://medium.com/intotheblock/practical-machine-learning-for-blockchain-datasets-understanding-semi-and-omni-supervised-learning-2a2611695b2.

[8] Regression in Machine Learning. https://medium.com/datadriveninvestor/regression-in-machine-learning-296caae933ec.

[9] Simple Reinforcement Learning: Temporal Difference Learning. https://medium.com/@violante.andre/simple-reinforcement-learning-temporal-difference-learning-e883ea0d65b0.

[10] Stacking in Machine Learning. URL `https://supunsetunga.blogspot.com/2016/06/stacking-in-machine-learning.html`.

[11] Supervised Learning: Basics of Classification and Main Algorithms. https://towardsdatascience.com/supervised-learning-basics-of-classification-and-main-algorithms-c16b06806cd3.

[12] Supervised Learning vs Unsupervised Learning. Which is better? https://lawtomated.com/supervised-vs-unsupervised-learning-which-is-better/.

[13] Airiau S., Padham L., S. S., S S.: Incorporating Learning in BDI Agents. In: *Adaptive Learning Agents and Multi-Agent Systems Workshop*, 2008.

[14] Ayodele T.O.: Types of Machine Learning Algorithms. In: *University of Portsmouth*, 2010.

[15] Barlow B., Hilton C.: The 4 Machine Learning Models Imperative for Business Transformation. https://www.rocketsource.co/blog/machine-learning-models/.

[16] BBEIMAN L.: Bagging Predictors. In: .

[17] Bühlmann, Peter: *Bagging, boosting and ensemble methods*.

[18] Breiman L.: Arcing Classifiers. In: .

[19] Dobrowolski G.: *Systemy agentowe w ujeciu pragmatycznym*. Wydawnictwo AGH, 2016.

[20] Duarte V.A.R., da Silva Julia R.M., Albertini M.K., Neto H.C.: MP-Draughts: Unsupervised Learning Multi-agent System Based on MLP and Adaptive Neural Networks. In: *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 920–927, 2015.

[21] van Engelen J.E., Hoos H.H.: A survey on semi-supervised learning. In: *Springer*, 2019.

[22] Garland A., Alterman R.: Autonomous Agents that Learn to Better Coordinate. In: *Autonomous Agents and Multi-Agent Systems*, vol. 8, pp. 267–301, 2004.

[23] Gehrke J.D., Wojtusiak J.: Traffic Prediction for Agent Route Planning. In: *Proceedings of the 8th International Conference on Computational Science, Part III*, ICCS '08, p. 692–701. Springer-Verlag, Berlin, Heidelberg, 2008. ISBN 9783540693888. URL `http://dx.doi.org/10.1007/978-3-540-69389-5_77`.

[24] Gerhard W.: *A Modern Approach to Distributed Modern Approach to Artificial Intelligence*. Massachusetts Institute of Technology, 1999. ISBN 0-262-23203-0.

[25] Graczyk M., Lasota T., Trawiński B., Trawiński K.: Comparison of Bagging, Boosting and Stacking Ensembles Applied to Real Estate Appraisal. In: .

[26] Hu T., Song T.: *Research on XGboost academic forecasting and analysis modelling*.

[27] Khalil K., Abdelaziz M., Nazmy T., M.Salem A.B.: Machine Learning Algorithms for Multi-Agent Systems. pp. 1–5. 2015. URL `http://dx.doi.org/10.1145/2816839.2816925`.

[28] Kotsiantis S.: *Emerging Artificial Intelligence Applications in Computer Engineering*.

[29] Martin W.N., Spears W.M.: *Foundations of Genetic Algorithms*. Elsevier, 2001. ISBN 978-1-55860-734-7.

[30] Natekin A., Knoll A.: Gradient Boosting Machines. In: .

[31] Opitz D., Maclin R.: Popular Ensemble Methods: An Empirical Study. In: .

[32] S. R., Norvig: *Artificial Intelligence A Modern Approach*. Alan Apt, 2006. ISBN 0-13-103805-2.

[33] Sardinha J.A.R.P., Choren R., Milidiu R.L., de Lucena C.J.P.: Engineering Machine Learning Techniques into Multi-Agent Systems. 2004.

[34] Shobha G., Rangaswamy S.: Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications. In: *Handbook of Statistics*, 2018.

[35] Sniezynski B.: Supervised Rule Learning and Reinforcement Learning in A Multi-Agent System for the Fish Banks Game. 2009.

[36] Talabis M.R.M., McPherson R., Miyamoto I., Martin J.L., Kaye D.: *Information Security Analytics*. Elsevier, 2014. ISBN 978-0-12-800207-0.

[37] Tang H., Matsubayashi T., Sato D., Toda H.: Time-Series Predictions for People-Flow with Simulation Data. In: T. Miller, N. Oren, Y. Sakurai, I. Noda, B.T.R. Savarimuthu, T. Cao Son, eds., *PRIMA 2018: Principles and Practice of Multi-Agent Systems*, pp. 655–662. Springer International Publishing, Cham, 2018. ISBN 978-3-030-03098-8.

[38] Vidhate D.A., Kulkarni P.: Innovative Approach Towards Coooperation Models for Multi-agent Reinforcement Learning (CMMARL). In: *Springer*, 2014.

[39] Wang G., Hao J., Ma J., Jiang H.: A comparative assessment of ensemble learning for credit scoring. In: *Expert Systems with Applications*, vol. 38(1), pp. 223 – 230, 2011. ISSN 0957-4174. URL `http://dx.doi.org/https://doi.org/10.1016/j.eswa.2010.06.048`.

[40] Williams A.B.: Learning to Share Meaning in a Multi-Agent System. In: *Autonomous Agents and Multi-Agent Systems*, vol. 8, pp. 165–193, 2004.

[41] Wittek P.: Quantum Machine Learning. In: *Academic Press*, 2014.

[42] Z. Qi F.M., Tian Y., Niu L., Shi Y., Zhang P.: Adaboost-LLP: A Boosting Method for Learning With Label Proportions. In: .

[43] Zhou Z.H.: *Ensemble Learning*, pp. 270–273. Springer US, Boston, MA, 2009. ISBN 978-0-387-73003-5. URL `http://dx.doi.org/10.1007/978-0-387-73003-5_293`.