Application Form
Innovative Teaching Award 2020[1]

APPLICANT
Name (incl. academic degree/s): Ass. Prof. Thomas Lindner, PhD MIM BSc
Department: Global Trade
Academic unit: International Business / Jonas Puck
Telephone: DW 4368 **Email:** thomas.lindner@wu.ac.at
Members of the working group, if applicable[2]: Mag. Claus Aichinger, Thomas Lindner PhD
GENERAL INFORMATION
Course level master's **Course number:** 2196 **Semester**: winter 2019/20 **ECTS credits:** 5 **Course title:** Machine Learning and the MNC **Number of students in the course:** 30
Further information on the course: (e.g. prior knowledge of students, position in the curriculum/program) Machine Learning and the MNC is an elective course in the International Functional Management branch of the MSc/MIM degree International Management/CEMS
Links to the course's online environment: Here you can provide the jury with links to the contents of your course's online environment for review. The course uses the learn@WU environment (https://learn.wu.ac.at/dotlrn/classes/pool/2196.19w/one-community?page_num=0), a GitHub repository (https://github.com/caichinger/MLMNC2019), and Google Colaboratory (https://colab.research.google.com/github/caichinger/MLMNC2019) computing environment.

Information on application

Please use the template on the following pages to describe your course. Provide a brief description (up to 180 words) of your course design under item 1. If your course is selected for an award, this text will be published on the WU website along with the submitted application form.

The more detailed description of your course design (item 2 below) is structured into three parts:

- Section 2a provides the jury with an initial overview of your course.
- In section 2b, please provide detailed information on teaching methods and the media-based didactic elements used.
- The innovative character of your course as based on this year's topic is described in section 2c.

The questions mentioned in each section are intended as support for the description of your course design. Please complete the template directly in word and send it as a .doc or .pdf file to lehrenundlernen@wu.ac.at by **09 February 2020**.

1. SHORT DESCRIPTION OF THE COURSE DESIGN (max. 180 words)

If your course is selected for an award, this text will be published on the WU website along with the submitted application form.

In this class, students learn how to apply machine learning to prominent business processes in Multinational Corporations (MNCs) where computational tools are employed to help transform complex information into manageable outputs. The course is organized around a group project in which students apply machine learning techniques to business questions. Such questions include predicting customer churn, forecasting demand, detecting fraudulent firms, and predicting the success of telemarketing campaigns. Instruction and project work are highly participative. Students use the Python programming language to implement computational procedures and perform data analytics tasks. All learning materials, particularly code, are provided through the Google Colaboratory interactive cloud computing environment for immediate inspection, practice, and adaptation. Following the literate programming paradigm, business objectives, statistical modelling considerations and Python source code are weaved together. The chosen framework substantially facilitates critical reflection of key concepts, producing workable code without prior programming experience. The course combines technical skills with management competence regarding business and societal consequences of using automated processes in firms. Students train scientific peer-review by evaluating peers' projects and assessing how sound data analysis approaches are.

2. DETAILED DESCRIPTION OF THE COURSE DESIGN

2a.) Overview

- What are the learning outcomes to be achieved by the students?
- What are the content elements of the course and how is it structured?
- What are the elements on which the final grade is based?

Learning outcomes to be achieved by the students

The "Machine Learning and the MNC" class teaches students how they can use artificial intelligence to solve business problems.

1. Students get a basic understanding of how machines learn.
2. Students get hands-on experience by using the Python programming language to employ machine learning algorithms in an interactive literate programming environment.
3. Students connect these insights to simple business problems, such as allocating promotion budgets.
4. Students learn to solve complex business problems, such as identifying potentially fraudulent firms, using machine learning techniques.
5. Students learn how to critically discuss computation results and how companies can manage some of the ethical and governance challenges resulting from integrating machine learning into a decision-making process.
6. Students get insights into how machine learning relates to other emerging technologies, such as quantum computing and distributed-ledger-systems (blockchain).
7. Students develop teamwork and feedback skills by evaluating and suggesting ways to improve their peers' projects.
8. Finally, students improve their teamwork and presentation skills.

Content elements and structure

The course has eight sessions. Before session 1, students are expected to develop basic understanding of key machine learning concepts by familiarizing themselves with two introductory chapters in leading books in the field. Sessions 1-4 focus on developing a basic understanding of machine learning and the required Python programming skills to address the solution of business problems. Session 5 focuses on the ethical and policy challenges of using machine learning in a multinational company. Session 6 introduces students to other emerging technologies in a panel-discussion format with experts who work at research institutions developing and applying these technologies (quantum computing, artificial intelligence, and the blockchain). The remaining two sessions (7&8) are dedicated to the group project. Instructors are available for student questions between sessions. The sessions carry the below titles.

1. Introduction to the class and machine learning with Python.
2. Conceptual foundations of machine learning.
3. Business case and machine learning focus 1: Business analytics.
4. Business case and machine learning focus 2: Text mining.
5. Machine learning in the MNC: Recruit Japan and Vodafone (case studies).
6. Discussion session digital technologies: Artificial intelligence, machine learning, quantum computing, and the blockchain.
7. Coaching session for group projects.
8. Group project presentations and feedback.

Grading elements

The final grade is balanced between group-level (55%) and individual-level (45%) elements. The key element in the grade is a class paper (group-level) that is to be delivered three weeks after the final session. Groups consist of 4-5 students. The grading elements are the following:

1. 40% class paper due three weeks after session 8 (group-level)
2. 15% final presentation in session 8 (group-level)
3. 15% peer rating due three weeks after session 8 (individual-level)
4. 10% class participation in sessions 1-6 (individual-level)
5. 10% quizzes in sessions 2-4 (individual-level)
6. 10% peer feedback on another group's final presentation in session 8 (individual-level)

Each instructor evaluates student performance. Keeping the holistic impression in mind, Claus Aichinger focusses on the technical aspects, and Thomas Lindner focusses on the business applications.

2b.) Teaching methods

- Which teaching methods and media-based didactic methods do you use to help your students achieve the intended learning outcomes? What role does online support play in this regard?
- Why did you choose these specific methods? What do you hope to achieve with them? Which particular advantages do these methods have for you?
- How do you address these media-based didactic concepts during the course?
- How do students profit from the media-based didactic methods used in the course?

Overview of how the teaching methods and media-based didactic methods help students achieve the intended learning outcomes

This class creates a cooperative learning environment using interactive computing and literate programming to help students understand how machines learn, and how companies can employ machine learning techniques to solve business and societal problems. It is the first class to teach Python-based machine learning with concrete business applications in the International Management/CEMS program at WU, and one of the first such courses in general. The class uses four teaching methods and media-based didactic methods. First, the key novelty in this class is a combination of cooperative learning, interactive computing, and literate programming: Students can immediately view, explore, alter, and run the code used for class demonstrations in order to understand the discussed material during the sessions, as well as afterwards when they work together on exercises and projects. The instant feedback students receive from running the code and their peers allows them to quickly grasp the applications, underlying general concepts, and limitations of machine learning techniques such as random forests. Student groups can expand on the code discussed and shown in class in the online programming environment provided, as well as provide instant feedback to each other. As a result, students develop hands-on programming skills they can use to solve complex business problems using data. Second, readings in digitally available textbooks provide the basis for discussion in the initial foundational sessions. For each of the first four sessions, accompanying homework exercises encourage students to apply the learned procedures to solve small programming tasks in the online environment. By reviewing the exercise solutions in class, students reflect on their understanding of how machines learn, and how companies can employ simple machine-learning techniques. Third, interactive case study teaching provides students with insights into the managerial and societal challenges that emerge from using machine learning techniques in multinational corporations. Particularly, students are confronted with decision-making dilemmas and challenges for organizational adaptation in two of the largest MNCs: Vodafone and Recruit Japan. Fourth, guest lectures provide insights into the state of the art of emerging digital technologies beyond machine learning: quantum computing and the blockchain. These insights broaden the scope of how students think about digitalization in the corporate world in general and allow them to put their understanding of machine learning into a broader perspective. Overall, students receive inputs that help them become “translators” (Tietze et al., 2017) between the technical and business experts that develop firms’ digital strategies. In a business context, students can “augment” (Davenport & Kirby, 2016 pp. 16-17) their business skills with basic applications of artificial intelligence based on what they learn in this class. Such augmentation will be key in managing firms’ transition into an efficient use of machine learning methodologies. Consequently, this course gives students a strong competitive edge in the job market.

The role of online support for the learning outcomes

Students receive online support and media throughout the class through the GitHub, Google Colaboratory, and learn@WU environments. This is critical in support of the first teaching method described above. Particularly, students can use a shared online programming environment during instruction sessions and at home to understand, change, and develop the elements discussed in class. Because the code is in an online repository, student groups can assess each others’ code, and easily ask for input from the course instructors by simply inviting the instructor to the respective document who can then make necessary adjustments or provide a hint. Online support greatly benefits the learning process, in particular for interdisciplinary content such as in this course, where a lot of different aspects need to be considered at once and small mistakes weigh in heavily. Moreover, the online environment in Google Colaboratory does not require offline installation of software, and accesses computing power provided in the cloud. This eliminates differences among students in the degree to which they have access to expensive technology. In this class, GitHub, a leading online content management system, is used to collect lectures notes, in particular including Python code used in demonstrations in class, and source code for presentation slides. The GitHub repository feeds the Google Colaboratory computation environment where lecture notes are then made available for immediate use as described above.. Students have access to these online environments during the whole semester, and can draw on it in a substantive way when developing their group project, as well as to relate their feedback to an earlier application of similar code. The data for the group projects are also available in the online environment. Learn@WU is used as a complement to the GitHub environment. Basic materials, links, peer-evaluation, peer-feedback, and uploads for hand-ins are provided through learn@WU.

Choice of teaching methods and media-based didactic methods and their particular advantages

The core teaching methods in this class are cooperative learning, interactive computing, and literate programming. They are supplemented with reading assignments, homework exercises, interactive case-study teaching, and guest lectures. The key didactic challenge is bridging the gap between the business applications students know a lot about on the one hand, and machine learning theory and the required programming requirements that students have very little prior knowledge about on the other hand. The combination of cooperative learning, interactive computing, and literate programming allows us to bridge this gap.

- First, the class benefits from exchange between more and less "expert" (Sternberg, 1998) students. Because there are substantial differences among students' experience with programming, the cooperative learning (Foldnes, 2016) environment facilitates student-student instruction. Moreover, cooperative learning leads to better learning outcomes because of social interdependence: Students are more motivated to engage with the material if they are challenged to solve a task jointly. Finally, cooperative learning creates an environment where students are incentivized to provide feedback to each others' work. This is reflected in the grading approach which allocates points based on both, peer assessments (15%) and the quality of peer-feedback provided (10%).
- Second, the interactive computing (Wegner, 1998) approach gives students instant feedback on their understanding of the matter at hand. In interactive computing, program inputs are immediately evaluated and displayed. By providing immediate feedback, the oftentimes complex task of implementing a program can easily be broken into very small digestible steps. This is particularly useful when translating domain specific knowledge (e.g., the reasons why customers choose a competitors' products) and requirements into source code or implementing evolved computational procedures where the outcome is not obvious a-priori. Complementing literate programming, interactive computing allows the user to lay out a computation following their logic and flow of thought, taking one step at a time. In short, this approach translates "what if" questions to "let's try" activities.
- Third, literate programming (Knuth, 1984) brings the abstract structure of computer code close to human language and the underlying thought process. In applying literate programming, users weave together source code and natural language (as well as other media) to directly connect the business context (for example in one project, the reasons underlying customer churn) and the corresponding computational procedure (in case of customer churn, it could be a random forest prediction customer behavior or a model explaining variable importance). The more complex the task and the related computations, the more important this entanglement becomes. Not only is it easier to verify that the implemented computations satisfy the requirements and vice versa, a literate program in addition facilitates collaboration as it greatly simplifies the translations between different domain-specific means of expression.

Advantages of chosen teaching methods and media-based didactic methods

Taken together, the three core didactic methods make it possible for students that are relative novices in programming and relative experts in the business domain to develop their understanding of how decisions processes can be modelled and optimized by employing a prediction machine using an online Python engine. Moreover, the literate programming approach by construction develops a documented version of computer code that can be understood by peers or peer groups, who can in turn provide feedback and suggestions for improvement. Finally, because code development is interactive and based on the code discussed in class, students can break down the complex task of answering a business question using data into small steps for which they can receive help from "expert" students or the course instructors straightforwardly.

The supplementary didactic methods of reading assignments, interactive case studies, and guest lectures complement the core teaching method in three important ways. First, the reading assignments build a common foundation of knowledge that the introductory sessions start from. Second, the interactive case studies highlight broader societal and policy issues that emerge from using machine learning techniques. Third, the guest lecturers put the machine learning methodologies into a broader perspective including other emerging technologies, such as quantum computing and the blockchain.

Integration of media-based didactic concepts in the course

Student-student and instructor-student interaction are integrated into the course in several ways. First, students use the code provided by the instructors on GitHub for their exercise assignments, and hand-in the developed-upon code for assessment. In the process of developing more advanced code for the group project, students and instructors share code and provide comments and feedback. Moreover, peers within a group provide structured feedback (following a format provided by the instructors) to their colleagues, and students from different groups provide written feedback on methods. Both elements are graded to encourage critical and substantive reflection (with 15% and 10% of the overall grade, respectively). Second, selected students' solutions to homework exercises are shared with the class to highlight best practices and common difficulties with solving the problems given. Third, instructors provide feedback on the student groups' solutions to business problems. Because of the public nature of the data, these solutions can be compared to online sources in terms of metrics of fit.

Student benefits of media-based didactic concept

The combination of cooperative learning, interactive computing, and literate programming makes it possible to quickly develop programming skills without having to understand all fundamentals of a programming language and transparently link implementation to theory and business objectives. First, because literate programming relies on a code structure similar to the human thought process, students can understand in a step-by-step way how they can support their line of reasoning with machine learning-based methods of data analysis. Second, because the literate programming approach is combined with interactive computing in an online environment to which all students and instructors have access, feedback cycles are very short and students can deepen their understanding of certain blocks of code, while running other elements they are less interested in without deeper analysis. Third, the cooperative learning design in the online environment facilitates student-student learning: More experienced students can share their insights explicitly with less experienced students by producing and commenting code following other students' thought process. Fourth, because the quality of student-student interaction is graded, students have a strong incentive to share their insights with their peers, both within a student group and across groups. Fifth, because the online computing and content management software are free to access, and because no offline installations are required, all students have equally fast access to the same amount of computational power and storage, independent of their own financial or social background. Finally, the complementary use of well-documented case studies and media-supported inputs by guest lecturers puts the acquired machine learning skills into context with both broader societal questions and other emerging technologies.

2c.) Innovative character of the course

- To which of the categories listed under item 2 of the call for applications does your submission belong?
- Which of your concept's didactic elements do you consider to be particularly innovative with regard to this year's topic "Supporting learning processes online"?
- Transferability: To what extent is your course design transferable to other courses? Which of your concept's teaching methods and media-based didactic elements could be applied to other courses at WU?

Which of your concept's teaching methods and media-based didactic elements have room for improvement or need to be reconsidered the next time you offer this course?

Categorization of the submission

This submission belongs to the categories "Course designs which use media-based didactic approaches to encourage contact between teachers and students", and "Course designs that use media-based approaches to encourage peer interaction".

The course inherently uses media-based didactic approaches to maximize the interaction between instructors and students. This is because of the interactive computing and literate programming approaches to teaching: Students run the code presented in class simultaneously with the instructors, and students have access to the code in an "online phase" before class to experiment with it. Students can also change and experiment with the code after class in another "online phase" to understand "what if" questions by simply trying out what happens if they change a parameter, or a full block of code, in a "let's try!" sense. When students work on assignments available in GitHub outside class, they can request support or feedback by simply sharing the Google Colaboratory notebook or of course through learn@WU and e-mail. The homework assignments ask students to expand on the code presented in class, and the following session starts with a discussion and further expansion of students' submissions. Because of the literate programming paradigm employed in this class, students with little prior experience in programming can still follow the discussion and make their first steps in changing code. Consequently, entry barriers to the class are substantially reduced, and learning outcomes improved. The student-instructor interactions is further reinforced during the project phase, when students interact with instructors not only online but also in dedicated offline phases. These offline "coaching sessions" are supported by online submissions of project plans and coding drafts, which facilitate students' project management.

The combination of cooperative learning with interactive computing and literate programming through shared online code development and execution in GitHub and Google Colaboratory results in intense peer-interaction. While this interaction emerges in the foundational phase of the class, it becomes central to learning outcomes in the project phase. Because care is taken that student groups are similarly heterogeneous with regards to their prior programming experience, "expert" and "novice" students exist in all groups. Because students receive their group allocation in the first session, peer-interaction is important from the foundational sessions (1-4), through the interactive case study session (5), to the project phase (7&8). In the online environment, groups can co-develop code and, because of the interactive computing approach, can change, feedback, and develop each other's code. The discussions that emerge from this co-development were sometimes fierce, but student feedback shows they learn substantially more from a co-developed project than if they each had focused on individual segments of the project. Because of the literate programming paradigm, different levels of expertise are relatively easily reconciled, as less experienced students can still translate their thoughts into a pseudo-code structure that more advanced students can readily translate into code. The less experienced students, in turn, can experiment with this code in the iterative programming environment using GitHub and Google Colaboratory. Peer-activities are further encouraged because they are graded in two ways: Students receive points based on their peers' assessment (15%), and students' feedback is graded for quality (10%).

Particularly innovative didactic elements for "supporting learning processes online"

The central didactic novelty in this course in the context of "supporting learning processes online" is the online environment in which the students can, using interactive computing and literate programming, develop their "learning from data" skills using Python in a cooperative learning process. Because the full instruction and learning environment is online, available throughout the semester, and because students can experiment with existing code and develop new blocks, student-student and student-instructor interaction is directly linked to the learning objectives. Because literate programming makes cooperative learning between "expert" and "novice" students much easier than traditional programming paradigms, even students with hardly any programming experience can understand and develop quite complicated machine learning methodologies, such as random forests and neural nets. The interactive computing element, where blocks of code can be changed without the need to pre-install or prepare any software, experimentation and inquiry-based learning is facilitated. In support of experimentation, direct feedback from and interaction with peers and instructors is possible through shared notebooks in the Google Colaboratory environment, and through more traditional channels such as e-mails, learn@WU, and of course appointments with instructors. In addition, structured coaching sessions, peer-feedback, and peer-evaluation, all of which contribute to students' grades, encourages active participation in cooperative learning, both offline and online.

The online environment is based on a GitHub and Google Colaboratory link, which is state-of-the art in content development. This online environment is supplemented with digitally available readings, online case studies, and instructor materials. Cloud-based computing in this online environment has another key advantage in eliminating all prerequisites for software installation, computing power, and investment into state-of-the art hardware on the part of students. Consequently, the online environment prepared for this class also fosters diversity by eliminating privileged access. In addition, students translate the machine learning methodologies they learn in this class into decision support for common business problems, the underlying methodologies' abstract nature is linked to problems students have prior understanding for. Finally, and most importantly, the online environment facilitates cooperative learning under iterative computing and literate programming because physical barriers (for example because students work on different computers) and code-natural language differences (because of the literate programming paradigm) are reduced, and experimentation (because of the iterative computing approach) replaces theoretical "what if" discussions.

Transferability

The teaching concept can be transferred to a broad range of courses, particularly two types. First, other courses which focus on the role that digitalization will play in businesses in the future. Developing concrete applications of digital technologies in the management and strategy formulation processes is critical for students to understand the deep changes that the use of digital technologies will have in managing companies. In all such courses, students will have substantially different degrees of experience with the digital component, because this is only part of university curricula in business and economics to a limited degree. As a result, a cooperative learning environment that facilitates cooperative learning, reduces barriers to translating thoughts into code (literate programming), and encourages experimentation (interactive computing) is key to maximize learning outcomes. Second, the course design can be applied to all formats where learning outcomes include understanding of quantitative methods. Such methods are frequently best understood if students get the chance to apply the presented methods in the context of a more complex example. Particularly, applications to business problems highlight the limitations, possible extensions, and complications that emerge when using a quantitative methodology to support business decisions. Management programs at WU, particularly on the Master's level, for example, may benefit from using cooperative learning designs under online support to bridge gaps between students' understanding of quantitative methodologies.
It is worth noting that while this lecture used Python, GitHub and Google Colaboratory as tools, the outlined concepts can be implemented using different technologies (and avoid externally provided resources).

Potential for improvement

This course design was used for the first time in the winter semester 2019/20. There are a few changes to the setup that will make the course easier to follow for students in the next semesters. First, an e-learning module for Python provided by a gamified learning platform that was used in a parallel course (Data Science for International Business, LV 2194) will be used to give students additional room to practice their programming skills before the class starts. Second, the group projects will use business cases with more international focus in order to link the course better to the international management focus of the degree program. Third, students will receive more input on a smaller amount of topics. Particularly, the text mining component will be eliminated from the class. Fourth, the cooperative learning environment will be strengthened by giving students additional group tasks before they start the group projects. Finally, using an application to WU's team-teaching grant, offline student-instructor interaction will be further strengthened.

Attachments: Evaluation results, central references, links to online examples and software