ESPECIALIZACIÓN EN INGENIERÍA EN SISTEMAS DE INFORMACIÓN

MAESTRÍA EN INGENIERÍA EN SISTEMAS DE INFORMACIÓN

1. Nombre de la actividad curricular: Aplicación de Técnicas de Aprendizaje Automático en Ingeniería del Software y Desarrollo de Sistemas Inteligentes

2. Año Académico: 2020

3. Docentes: Dra. Mariel Alejandra Ale y Dra. Luciana Ballejos.

4. Fundamentación

Hoy en día es común encontrar noticias sobre aplicaciones de Aprendizaje Automático (AA), minería de datos, análisis de grandes datos y la forma en que las mismas cambian la sociedad. Sin embargo, es sorprendente el poco impacto que han tenido estas tecnologías revolucionarias en la Ingeniería del Software (IS) en sí misma. Probablemente, uno de los obstáculos para la incorporación de estas tecnologías en la IS es el desconocimiento general de cómo las mismas podrían aplicarse y los problemas y desafíos que pueden ayudar a resolver.

Si bien el AA es una disciplina madura que tiene sus orígenes en los años 50, la perspectiva de su influencia en la IS es menos común y es difícil por estos días encontrar en el área referencias o una introducción que sean a la vez accesibles y pedagógicas dirigidas a ingenieros de software. Aún más, algunos de los métodos de AA que se están actualmente utilizando en la IS no son analizados en profundidad en la comunidad.

La variedad de aplicaciones del AA a los problemas del mundo real es amplia e incluye desde sistemas recomendadores de servicios sociales y comerciales, a productos altamente regulados, como los prototipos de vehículos autónomos. El desarrollo de aplicaciones con componentes de AA presenta una serie de inconvenientes y falencias específicas. Una de las razones principales es el cambio de paradigma que ha introducido el aprendizaje automático en el desarrollo del software. Tradicionalmente, los sistemas de software se construían de una forma deductiva, escribiendo las reglas que gobiernan el comportamiento del sistema como código de programación. Sin embargo, con las técnicas de AA, estas reglas son inferidas a partir de los datos de entrenamiento. Este cambio de paradigma hace que el razonamiento sobre el comportamiento de un sistema de software con componentes de aprendizaje automático sea difícil, resultando en sistemas de software que son intrínsecamente complejos de probar y verificar. De hecho, el comportamiento aprendido de un sistema basado en aprendizaje automático puede ser incorrecto, aun cuando el algoritmo esté correctamente implementado, situación que no puede ser detectada por las técnicas de testeo tradicionales. El desafío en el área entonces es cómo desarrollar y testear de forma efectiva este tipo de sistemas, dado que no tienen especificaciones completas o incluso código fuente asociado a algún comportamiento crítico.

A lo anterior se le agrega que, aun cuando existe cierto consenso sobre los procesos de alto nivel que abarcan el desarrollo de un sistema inteligente, el uso de métodos informales en algunos de dichos procesos (manejo del conjunto de datos, selección del algoritmo a utilizar, entrenamiento, entre otros), hace que sea difícil reproducir o extender los resultados obtenidos en un sistema concreto.

Por otro lado, los procesos de la Ingeniería del Software pueden verse beneficiados por los métodos de aprendizaje automático y desarrollar, de esta forma, mejores artefactos de una forma efectiva y eficiente. Dentro de las áreas en las que el aprendizaje automático puede ser útil, se encuentran la predicción o estimación de medidas de atributos asociados a los procesos, productos o recursos (calidad, tamaño,

costo, esfuerzo, tasa de defectos, reusabilidad, entre muchos otros), descubrimiento de propiedades internas o externas de los procesos, productos o recursos, transformación de productos para obtener un atributo nuevo o mejorado, síntesis o generación de varios productos, reuso de productos o procesos, mejora de procesos y gestión de productos.

Por todo lo expuesto, y dados los desafíos descriptos, es imperativo que tanto la comunidad de AA como los ingenieros de software sean capaces de desarrollar métodos innovativos para resolver esta problemática.

Los alumnos desarrollarán el conocimiento y las habilidades requeridas para analizar, diseñar e implementar sistemas inteligentes que empleen técnicas de aprendizaje automático en diversas actividades del proceso, aplicables a cualquier dominio. Se enfatizará la comprensión conceptual de las herramientas fundamentales y su aplicación en la práctica.

5. Objetivos

El objetivo general de este curso es proporcionar a los alumnos los conceptos, enfoques metodológicos y principales campos de aplicación de los Sistemas Inteligentes, así como formarlos en el uso de estos conocimientos y habilidades en el abordaje del análisis, diseño e implementación de este tipo de sistemas.

Se espera que al final del curso los alumnos puedan identificar las ocasiones en las que sea factible y adecuado incorporar estrategias de aprendizaje automático en el diseño de software.

Como objetivos específicos, se pretende que los alumnos sean capaces de:

1. Identificar y resolver las problemáticas particulares asociadas al desarrollo de sistemas inteligentes.
2. Seleccionar las técnicas de aprendizaje automático adecuadas a la problemática a resolver.
3. Puedan aprovechar los beneficios de las técnicas de aprendizaje automático en las actividades propias de la Ingeniería del Software.

6. Contenidos

TEMA 1: Proceso de Desarrollo de Software

Definición. Procesos del Ciclo de Vida del Software. Estándar ISO 12207: 2017: procesos de acuerdo, procesos organizacionales del proyecto, procesos del proyecto, procesos técnicos

TEMA 2: Aprendizaje Automático

Introducción al aprendizaje automático. Aplicaciones comunes. Aprendizaje Supervisado. Aprendizaje No Supervisado. Aprendizaje por Refuerzo. Aprendizaje Profundo.

TEMA 3: Desarrollo de Sistemas Inteligentes

Sistemas Inteligentes. Definición ¿Cuándo usar un sistema inteligente? Requerimientos del Modelo. Recolección, Limpieza y Etiquetado de Datos. Ingeniería de Características. Entrenamiento, Evaluación, Despliegue y Monitoreo del Modelo.

TEMA 4: El Aprendizaje Automático en los procesos de Ingeniería del Software

Aplicación de AA y ejemplos en diversas actividades de la IS:

- predicción y estimación (calidad, costo, tamaño, esfuerzo de desarrollo, esfuerzo de mantenimiento, confiabilidad, defectos, reusabilidad, tiempo de ejecución
- descubrimiento de modelos y propiedades (identificación de objetos, límites de operación normal, anomalías).
- transformación (paralelización, modularización, mapeo de objetos).
- generación y síntesis (casos y datos de prueba, diseño de reglas de reparación).
- reuso (identificación, recuperación y organización de componentes reusables).

- Obtención de requerimientos.
- gestión del conocimiento de desarrollo (recolección y gestión del conocimiento generado durante el desarrollo del software).

TEMA 5: Futuro del Aprendizaje Automático

Claves para entender el futuro del AA en el ámbito de la IS. Posibles campos de aplicación futuros. AA en un futuro 5G. Consideraciones de seguridad y éticas.

7. Metodología de Enseñanza y Formación práctica

Las clases serán del tipo teórico-práctico, en donde se expondrán los conceptos teóricos y se realizará la resolución de guías de ejercicios con el propósito de afianzar los conocimientos.

Aproximadamente la mitad del tiempo del curso será destinada a la formación práctica, a través de la resolución de guías con problemas y casos de estudio, como así también a la resolución de los trabajos prácticos, de tal manera que los alumnos puedan consultar al profesor los problemas o dudas que surjan de la elaboración de los mismos.

Pedir a las docentes que completen con: lugar donde se desarrollan, modalidad de supervisión y modalidades de evaluación.

8. Carga horaria total

	Tiempo Estimado (hs.)		
Unidad Temática	**Teoría**	**Práctica**	**Total**
Tema 1	5		5
Tema 2	5	10	15
Tema 3	10	10	20
Tema 4	8	10	18
Tema 5	2		2
Total (hs.):	30	30	60

9. Modalidad de Evaluación

Se requieren conocimientos básicos de Ingeniería del Software. La evaluación de los conocimientos y capacidades adquiridas durante el cursado se realizará por medio de un examen escrito y la resolución de un trabajo práctico integral.

El examen escrito es integrador, final e individual. El trabajo práctico se enfoca en la resolución de casos de estudio, utilizando los lenguajes, métodos y técnicas desarrollados en el cursado.

También se realiza un seguimiento de los alumnos durante el cursado, los cuales son expuestos al empleo de lenguajes, técnicas, métodos y herramientas, mediante la resolución de casos de estudio y guías de ejercicios prácticas, con el propósito de afianzar los conceptos teóricos.

10. Requisitos de aprobación y promoción

Para la aprobación del curso se requiere que los alumnos obtengan, como mínimo, una calificación de 7 (siete) en cada instancia de evaluación (Ordenanza Nro. 1313). La nota final corresponderá en un promedio de todas las calificaciones obtenidas.

11. Infraestructura y equipamiento

La infraestructura y ámbitos a utilizar en el dictado son los siguientes:

1. Campus virtual: el material bibliográfico del curso, las presentaciones y los enunciados de las ejercitaciones y trabajos prácticos se encuentran disponible en el campus virtual de la Facultad Regional Santa Fe.
2. Aulas: las clases teóricas se desarrollan en un aula con capacidad para 50 estudiantes, equipo de proyección y acceso a internet mediante conexión wifi. Todo el equipamiento mencionado es empleado en el dictado de las clases teóricas.
3. Laboratorio: se dispone de 2 laboratorios, LABSIS 4 y LABSIS 5 con capacidad para 20/25 estudiantes, en ambos casos con acceso a internet y disponibilidad de proyector. La utilización de uno u otro laboratorio depende de la cantidad de estudiantes inscriptos en el curso.

12. Bibliografía

Al-Jamimi, H. A., & Ahmed, M. (2013, June). Machine learning-based software quality prediction models: state of the art. In *2013 International Conference on Information Science and Applications (ICISA)* (pp. 1-4). IEEE.

Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., ... & Zimmermann, T. (2019, May). Software engineering for machine learning: a case study. In *Proceedings of the 41st International Conference on Software Engineering: Software Engineering in Practice* (pp. 291-300). IEEE Press.

Arpteg, A., Brinne, B., Crnkovic-Friis, L., & Bosch, J. (2018, August). Software engineering challenges of deep learning. In *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* (pp. 50-59). IEEE.

Bennaceur, A., Giannakopoulou, D., Hähnle, R., &Meinke, K. (2016). Machine learning for dynamic software analysis: Potentials and limits (dagstuhl seminar 16172). *In Dagstuhl Reports (Vol. 6, No. 4)*. Schloss Dagstuhl-Leibniz-ZentrumfuerInformatik.

Bosch, J., Olsson, H. H., &Crnkovic, I. (2018). It takes three to tango: Requirement, outcome/data, and AI driven development. In *SiBW* (pp. 177-192).

Breck, E., Cai, S., Nielsen, E., Salib, M., & Sculley, D. (2016). What's your ML Test Score? A rubric for ML production systems. At https://ai.google/research/pubs/pub45742

Fabijan, A., Dmitriev, P., Olsson, H. H., & Bosch, J. (2017, May). Theevolutionofcontinuousexperimentation in software productdevelopment: from data to a data-drivenorganization at scale. In *Proceedingsofthe 39th International Conferenceon Software Engineering* (pp. 770-780). IEEE Press.

Hains, G., Jakobsson, A., &Khmelevsky, Y. (2018, April). Towards formal methods and software engineering for deep learning: security, safety and productivity for dl systems development. In *2018 Annual IEEE International Systems Conference (SysCon)* (pp. 1-5). IEEE.

Hill, C., Bellamy, R., Erickson, T., & Burnett, M. (2016, September). Trials and tribulations of developers of intelligent systems: A field study. In *2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 162-170). IEEE.

Kanewala, U., &Bieman, J. M. (2014). Testing scientific software: A systematic literature review. *Information and software technology, 56(10)*, 1219-1232.

Kasparov, G. (2017). *Deep thinking: where machine intelligenceends and human creativitybegins*. PublicAffairs.

Khomh, F., Adams, B., Cheng, J., Fokaefs, M., &Antoniol, G. (2018). Software Engineering for Machine-Learning Applications: The Road Ahead. *IEEE Software*, *35*(5), 81-84.

Kim, M., Zimmermann, T., DeLine, R., &Begel, A. (2016, May). The emerging role of data scientists on software development teams. In *Proceedings of the 38th International Conference on Software Engineering* (pp. 96-107). ACM.

Kim, M., Zimmermann, T., DeLine, R., &Begel, A. (2017). Data scientists in software teams: State of the art and challenges. *IEEE Transactionson Software Engineering*, *44*(11), 1024-1038.

Louridas, P., & Ebert, C. (2016). Machine Learning. *IEEE Software*, 33(5), 110–115.

Lwakatare L.E., Raj A., Bosch J., Olsson H.H., Crnkovic I. (2019) A Taxonomy of Software Engineering Challenges for Machine Learning Systems: An Empirical Investigation. In: Kruchten P., Fraser S., Coallier F. (eds) *Agile Processes in Software Engineering and Extreme Programming. XP 2019*. Lecture Notes in Business Information Processing, vol 355. Springer, Cham

Meinke, K., &Bennaceur, A. (2018). Machine Learning for Software Engineering: Models, Methods, and Applications. In *Proceedings of the 40th International Conference on Software Engineering: Companion Proceedings* (pp. 548-549).

Murphy, C., Kaiser, G. E., & Arias, M. (2007). An approach to software testing of machine learning applications. At https://www.cs.upc.edu/~marias/papers/seke07.pdf

Nascimento, N. M., & de Lucena, C. J. P. (2017, July). Engineeringcooperativesmartthingsbasedonembodiedcognition. In *2017 NASA/ESA ConferenceonAdaptive Hardware and Systems (AHS)* (pp. 109-116). IEEE.

Nascimento, N., Lucena, C., Alencar, P., &Cowan, D. (2018). Software engineers vs. machine learning algorithms: An empirical study assessing performance and reuse tasks. *arXiv preprint arXiv:1802.01096*.

Nushi, B., Kamar, E., Horvitz, E., &Kossmann, D. (2017, February). On human intellect and machine failures: Troubleshooting integrative machine learning systems. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Oizumi, W., Sousa, L., Garcia, A., Oliveira, R., Oliveira, A., Agbachi, O. I., & Lucena, C. (2017, September). Revealingdesignproblems in stinkycode: a mixed-methodstudy. In *Proceedingsofthe 11th BrazilianSymposiumon Software Components, Architectures, and Reuse* (p. 5). ACM.

Partridge, D. (2013). *Artificial intelligence and software engineering*. Routledge.

Patel, K., Fogarty, J., Landay, J. A., & Harrison, B. (2008, April). Investigating statistical machine learning as a tool for software development. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 667-676). ACM.

Rashid, E., Patnayak, S., &Bhattacherjee, V. (2012). A survey in the area of machine learning and its application for software quality prediction. *ACM SIGSOFT Software Engineering Notes*, *37*(5), 1-7.

Schelter, S., Böse, J. H., Kirschnick, J., Klein, T., & Seufert, S. (2017). Automatically tracking metadata and provenance of machine learning experiments. In *Machine Learning Systems workshop at NIPS*.

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D. & Young, M. (2014). Machine learning: The high interest credit card of technical debt.*In SE4ML: Software Engineering for Machine Learning (NIPS 2014 Workshop)*.

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D. &Dennison, D. (2015). Hidden technical debt in machine learning systems. In *Advances in neural information processing systems* (pp. 2503-2511).

Sridhar, V., Subramanian, S., Arteaga, D., Sundararaman, S., Roselli, D., &Talagala, N. (2018). Model governance: Reducing the anarchy of production {ML}. In *2018 {USENIX} Annual Technical Conference*

({USENIX} {ATC} 18) (pp. 351-358).

Wen, J., Li, S., Lin, Z., Hu, Y., & Huang, C. (2012). Systematic literature review of machine learning based software development effort estimation models. *Information and Software Technology*, *54*(1), 41-59.

Zhang, D., & Tsai, J. J. (2003). Machine learning and software engineering. *Software Quality Journal*, *11*(2), 87-119.

Zhang, D., & Tsai, J. J. (Eds.). (2005). *Machine learning applications in software engineering* (Vol. 16). WorldScientific.

Zhang, D. (Ed.). (2006). Advances in machine learning applications in software engineering. Igi Global.