UNIVERSIDAD DE BUENOS AIRES

PROPUESTA DE TRABAJO PROFESIONAL

DeltaML
Machine Learning descentralizado utilizando Blockchain para fomentar participación en la red

---

Author:
Fabrizio GRAFFE 93158
Agustín ROJAS 91462

Supervisor:
Dr. Mariano BEIRÓ

5 de febrero de 2019

Índice general

1 Introducción

El siguiente documento presenta la propuesta de Trabajo Profesional de Ingeniería en Informática de los estudiantes Fabrizio Sebastian Graffe (padrón 93158) y Agutín Rojas (padrón 91462).
El objetivo es aplicar los conocimientos adquiridos durante la carrera, para lo cual el tema elegido es **"Machine Learning descentralizado utilizando Blockchain para fomentar participación en la red"**.
Machine learning (o "aprendizaje automatico") es la rama del area de la inteligencia artificial centrada en el estudio y construcción de sistemas capaces de aprender de los datos, identificar patrones y realizar decisiones con mínima intervención humana.
Blockchain [1] o DLT (Descentralized Ledger Technology), por su lado, es una tecnología que consta de un registro contable distribuido en una red de nodos. Este registro almacena las transacciones que se realizan entre los nodos de la red y cada nodo tiene una copia completa. A su vez, para evitar fraude y alteraciones al registro por cualquier nodo malicioso la tecnología cuenta con algoritmos de concenso y de prueba de trabajo realizado (Proof of Work). Las ventajas de la tecnología Blockchain por sobre bases de datos tradicionales son:

- **Mayor transparencia:** Todas las transacciones son publicas, por lo que cualquier participante de la red puede verlas.
- **Criptograficamente segura:** Debido a los algoritmos antes nombrados, para poder cometer fradue se necesitaria un poder de computo mayor al 51 % de la red (algo que no es posible hoy en dia, al menos contra las redes de Bitcoin o Ethereum, ni por las mayores empresas de software en el mundo).
- **Irreversibilidad:** Una transacción registrada en la blockchain no puede ser alterada por ninguno de los participantes de la red (otra vez, se necesitaria un poder de computo mayor al 51 % de la red).

2 Descripción del problema

Tanto el area de Machine Learning como la tecnología Blockchain son elementos que estan disrumpiendo la industria del software en la actualidad y cada vez tienen mas importancia en la vida diaria de las personas. En algunos casos su aplicación ha resultado en avances con un impacto positivo en la humanidad, pero tambien existen casos en los que se su impacto ha sido negativo.
Un ejemplo muy importante de mal uso de la tecnologia es facebook, que ha tenido varios casos de violación de la privacidad de los datos de sus usuarios, venta e intercambio de éstos con otras companias, uso de técnicas de aprendizaje automatico para generar adicción a las novedades en su plataforma y generación de cámaras de eco (echo chambers) por medio de filtrado de contenido (para mostrar a los usuarios solo opiniones similares a la suya, provocando así un refuerzo de su propia visión aprovechandose de sesgos propios de los humanos como el Sesgo de Confirmación o Confirmation Bias), por nombrar algunos.
Así como facebook incurrió en estas malas prácticas que debilitan y manipulan a los usuarios de su plataforma, muchas otras empresas e incluso estados también lo hacen día a día.
La creciente necesidad de mantener la privacidad de nuestros datos, que dia a dia son mas valiosos, está llevando a diferentes gobiernos a plantear regulaciones mas estrictas en lo relativo al uso de los mismos. Esto a su vez está impulsando una gran cantidad de iniciativas en el mundo para cambiar el paradigma actual donde las empresas son dueñas de los datos de las personas, a uno donde las personas sean dueñas de sus propios datos y puedan venderlos para un uso determinado y ningun otro. En el ámbito local se tiene el ejemplo de Wibson como una empresa que va en ese camino.
Debido a todo lo anterior mencionado, se eligió la temática de este trabajo teniendo en mente que en los años por venir se necesitarán maneras de entrenar modelos de Machine Learning sin violar la privacidad de las personas, es decir, sin tener una copia de sus datos en bases de datos de las companias. Por lo cual el presente trabajo trata de contribuir en este paso hacia un futuro donde las personas sean dueñas de su información.

3 Estado del arte

Actualmente existen varios proyectos en desarrollo y técnicas en investigación que intentan atacar la misma problematica de diferentes maneras.

- **Federated Learning:** El aprendizaje federado es un método de aprendizaje automático en el que el objetivo es entrenar a un modelo centralizado de alta calidad con datos de entrenamiento distribuidos entre un gran número de clientes, cada uno con conexiones de red poco confiables y relativamente lentas. Los algoritmos de aprendizaje para esta configuración funcionan de la siguiente manera: en cada iteración, cada cliente calcula de forma independiente una actualización del modelo actual en función de sus datos locales y comunica esta actualización a un servidor central, donde las actualizaciones del lado del cliente se agregan para calcular una nueva versión global del modelo. Los clientes típicos en este entorno son los teléfonos móviles, y la eficiencia de la comunicación es de suma importancia. Actualmente, Google está investigando [2] [3] [4] [5] [6] [7] esté método de aprendizaje automático y publicó varios papers al respecto, analizando mejoras y aplicaciones para dispositivos móbiles (tales como predicción de palabras para GBoard).

- **OpenMined: [8]** Marketplace descentralizado de modelos de machine learning. Opera sobre la red de Ethereum. Utiliza Federated Learning para el entrenamiento de los modelos predictivos sobre Tensorflow y Pytorch ya que permiten entrenamiento desde clientes web. Modelos encriptados con homomorphic encryption para una transmición de datos segura. Smart contracts regulan el marketplace y pagan a quienes gastaron poder de computo entrenando un proporcional de acuerdo a cuanto aportaron a la mejora del modelo global. Actualmente en desarrollo.

- **DML: [9]** Muy similar a OpenMined.

- **NumerAI: [10]** Fondo de inversion que genera predicciones utilizando modelos predictivos crowd-sourced. Transforma y regulariza datos financieros a Transforms and regularizes financial data into machine learning problems for global network of data scientists

  Los intercambios de Numerai están determinados por una IA, que es alimentada por una red de miles de científicos de datos anónimos. La innovación tecnológica que Numerai proporciona está en su uso del cifrado de preservación de la estructura que aplican en sus fuentes de datos. Su objetivo es evitar sesgos y sobreajuste, también hace posible que Numerai comparta sus fuentes de datos de forma gratuita con sus usuarios.

- **Augur: [11]** Es un protocolo de predicción de mercados destinado a ser propiedad de las personas que lo usan. Augur es un oraculo descentralizado y un protocolo peer to peer para la predicción de mercados. Augur es proyecto open-source. Es un conjunto de smart contracts escritos en Solidity que operan sobre Ethereum.

- **Golem Network: [12]** Primera supercomputadora descentralizada que crea un marktplace global de poder de computo. Golem conecta computadoras en una red peer-to-peer, permitiendo tanto a dueños de aplicaciones como a usuarios individuales (requestors") alquilar recursos de las maquinas de otros usuarios ("providers"). Los pagos entre requestors, providers y desarrolladores de apliaciones se realizan por medio de un sistema de transacciones basado en la red de Ethereum. El sistema está orientado a competir con los proveedores de infraestructura o servicios cloud para aplicaciones que requieran gran capacidad de computo reduciendo el precio de dicha capacidad. Como consecuencia, aplicaciones complejas como la representación CGI, el cálculo científico y el aprendizaje automático se volverían más accesibles.
- **OceanProtocol: [13]** Ecosistema destinado a comprartir activos y servicios. Los activos son datos y algoritmos. Los servicios son la integración, procesamiento, computación y almacenamiento. Ocean Protocol es un protocolo de intercambio de datos descentralizado, que permite que personas compartan y monetizen sus datos a la vez que garantiza control, auditabilidad, transparencia y is a decentralized data exchange protocol that lets people share and monetize data while guaranteeing control, auditability, transparency y conformidad a todos los actores involucrados to all actors involved.
- **Wibson: [14]** Mercado de datos descentralizado, basado en blockchain, que proporciona a las personas una forma segura y anónima de vender información privada validada en un entorno confiable. Opera sobre la red de Ethereum. Las personas pueden vender sus datos por medio de la aplicacion movil de Wibson.

4 Objetivos

El presente trabajo consta de los siguientes objetivos principales:

- Desarrollar un marketplace de modelos de ML que permita a usuarios dueños de estos modelos entrenarlos de manera descentralizada sin necesidad de violar la privacidad de los usuarios dueños de los datos que serán utilizados para dicho entrenamiento.
- Desarrollar una forma de recompensar a los usuarios que provean datos y poder de computor para el entrenamiento de los modelos.
- Desarrollar un framework de Federated Learning de código libre para generar un aporte a la comunidad.
- Desarrollar un caso de uso de un modelo que requiera entrenamiento para hacer prueba del sistema desarrollado durante este trabajo.

5 Carácteristicas del trabajo

5.1. Módulo de Machine Learning descentralizado

Este modulo tendrá como responsabilidad realizar entrenamientos de modelos de Machine Learning siguiendo el esquema de Federated Learning. Tendrá una interfaz lo suficientemente genérica como para poder usarse con un servidor central o en una red de nodos descentralizados (como se pretende hacer con la red de Ethereum).

5.2. Módulo de encriptacíon

Este modulo tendrá como responsabilidad la encriptación y desencriptación de los modelos que se transfieran en la red para su entrenamiento.
Para evitar el envío de datos por la red, y así preservar la privacidad, el esquema de Federated Learning establece que se envíen los modelos a los clientes que los entrenarán. Dichos modelos deben estar encriptados, ya que de otra forma cualquier participante malicioso de la red podría robarlo.
Para poder operar sobre un modelo encriptado existen técnicas tales como Hommomorphic Encryption.
El modulo deberá minimizar el riesgo de violación de la privacidad tanto del usuario que entrena el modelo con sus datos, como del usuario que desea obtener un modelo entrenado (sin que nadie se lo robe en el proceso), para esto se explorarán técnicas de Secure Multi party computation y Differential Privacy.

5.3. La red

La red constará de uno o mas smart contracts desplegados en la red de Ethereum que estarán integrados con IPFS como método de almacenamiento descentralizado.
La red proveerá medios para recompensar a los participantes de la misma, tales como pueden ser los usuarios que entrenan los modelos, según el nivel de su aporte a la calidad de las predicciones del modelo.

5.4. Interfaz web del marketplace

El sistema debe tener una interfaz web que permita que usuarios puedan construir un modelo de machine learning que use determinado tipo de datos y enviarlo para su entrenamiento a los diferentes nodos de la red que cumplan con las carácteristicas pedidas por el usuario dueño del modelo.

5.5. Cliente web/mobile

Este cliente deberá permitir que los usuarios decidan cuales de sus datos quieren que sean usados para entrenar modelos de machine learning.

6 Tecnologías

6.1. Blockchain

6.1.1. Ethereum network [15]

Ethereum es una plataforma descentralizada que ejecuta contratos inteligentes (smart contracts): aplicaciones que se ejecutan exactamente como fueron programadas sin ninguna posibilidad de downtime, censura, fraude o interferencia de una tercera parte.

6.1.2. Solidity [16]

Solidity es un lenguaje de alto nivel, orientado a objetos para implementar smart contracts. Los smart contracts son programas que gobiernan el comportamiento de cuentas dentro del ecosistema Ethereum.

6.1.3. Zeppelin OS [17]

ZeppelinOS es una plataforma de desarrollo diseñada para proyectos que involucren smart contracts. Permite realizar actualizaciones de los mismos de manera sencilla y provee incentivos económicos para crear un ecosistema saludable de aplicaciones seguras.

6.1.4. Open Zeppelin [18]

OpenZeppelin es un framework de smart contracts reutilizables para Ethereum y otras blockchains de EVM y eWASM. Reduce el riesgo de vulnerabilidades mediante el uso de un estándar probado y revisado por la comunidad.

6.1.5. Truffle [19]

Framework de desarrollo de smart contracts para blockchains que utilizan la Máquina Virtual Ethereum (EVM).

6.1.6. Ganache [20]

Ganache es una blockchain personal para el desarrollo en Ethereum (testnet local) que se puede utilizar para implementar contratos, desarrollar sus aplicaciones y ejecutar pruebas.

6.1.7. Metamask [21]

MetaMask es una extensión que permite ejecutar aplicaciones descentralizadas (dApps) de Ethereum en web browsers sin la necesidad de tener un full node de Ethereum.

6.2. Machine Learning

6.2.1. Python [22]

Lenguaje de alto nivel y tipado dinámico con soporte de paradigma de objetos que se utilizará para el desarrollo de los módulos de Federated Learning y Encriptación. Se eligió este lenguaje por su uso simple, por las herramientas desarrolladas para las areas de Machine Learning y análisis de datos, y su gran adopción en el mundo del desarrollo y ciencias de datos, lo que reduce, también, dificultades en la resolución de problemas que puedan surgir durante el proyecto.

6.2.2. Sci-kit Learn [23]

Es un conjunto de herramientas de codigo abierto, de facil uso, destinandas a la mineria de datos, al análisis de los mismos y al aprendizaje automatico, construido sobre NumPy, SciPy, y matplotlib.

6.2.3. PyTorch [24]

Es una biblioteca de código abierto de aprendizaje automatico para Python, basada en Torch, usada para aplicaciones tales como el procesamiento del lenguaje natural. Es desarrollada principalmente por el equipo de investigación de inteligencia artificial de Facebook. PyTorch provee 2 funcionalidades de alto nivel: Computo usando Tensores (como NumPy) con aceleración usando GPUs y Redes Neuronales profundas.

6.2.4. Tensorflow [25]

Es una biblioteca de software libre que se utiliza para realizar cálculos numéricos mediante diagramas de flujo de datos. Los nodos de los diagramas representan operaciones matemáticas y las aristas reflejan las matrices de datos multidimensionales (tensores) comunicadas entre ellas. Se utiliza en gran medida para tareas de aprendizaje automatico.

6.2.5. Keras [26]

Es una API de alto nivel de redes neuronales, escrita en Python y capaz de ser ejecutada por encima de Tensorflow, CNTK o Theano. Fue desarrollada para permitir experimentación rapida con modelos de aprendizaje automatico basados en redes neuronales.

6.2.6. Tensorflow.js [27]

Es una biblioteca de JavaScript que permite entrenar y desplagar modelos de Machine Learning en los web browsers y en aplicaciones Node.js. Posee comptibilidad con modelos de Keras y Tensorflow, por lo que permite importar y exportar modelos de, y a, estos frameworks.

6.3. Frontend

6.3.1. React [28]

React es una biblioteca Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre, han participado en el proyecto más de mil desarrolladores diferentes.

6.4. Storage

6.4.1. IPFS [29]

IPFS (InterPlanetary File System) es un sistema distribuido para almacenar y acceder a archivos, sitios web, aplicaciones y datos cuyo funcionamiento sintetiza e integra ideas exitosas previas de los sistemas peer-to-peer, incluyendo DHTs (Distributed Hash Tables), BitTorrent, Git y SFS (Self-Certified FileSystems).

7 Alcance

El alcance del presente Trabajo Profesional comprende:

- Desarrollo de un framework de machine learning descentralizado.
- Desarrollo de un modulo de encriptación.
- Desarrollo de marketplace de modelos de ML por medio de smart contracts con las reglas de negocio para monetizar la contribución de los diferentes participantes.
- Desarrollo de integración entre smart contracts e IPFS para almacenar datos de manera descentralizada.
- Desarrollo de interfaz web para el marketplace.
- Pruebas del sistema en testnet de ethereum local y posibles pruebas en testnets globales (Kovan, Rinkeby, Ropsten).

8 Plan de Trabajo

8.1. Equipo de trabajo

Equipo	
Integrante	**Rol**
Mariano Beiró	Tutor
Fabrizio Graffe	Desarrollador
Agustín Rojas	Desarrollador

8.2. Metodología

Se estima que el trabajo completo a realizar tomará 6 meses, con una dedicación de 70 hs mensuales al desarrollo del trabajo de parte de cada integrante (ver sección Sección 8.3).
El proceso de desarrollo será iterativo e incremental, con dos iteraciones iniciales de dos semanas cada una y cinco iteraciones de un mes, a lo largo de las cuales se efectuarán tanto tareas de análisis, desarrollo, testing y documentación como tareas de gestión, cada una con distinta carga horaria dependiendo de la etapa del proyecto. Para la gestión del proyecto, se adoptarán algunas prácticas de metodologías ágiles. Como en Scrum [30], se realizarán reuniones de Planificación, Revisión y Retrospectiva para cada iteración. Se llevarán a cabo informes de estado de las tareas 3 veces por semana con el tutor y diariamente entre los integrantes del equipo de desarollo, ya sea de manera presencial, vía mail o videollamada dependiendo de la disponibilidad de los involucrados. Se utilizará un tablero de Kanban para organizar el trabajo en cada iteración.
La gestión de riesgos se llevará a cabo utilizando una planilla en donde se registrarán los riesgos identificados, junto con su probabilidad de ocurrencia, impacto sobre el proyecto y su exposición. Se describirán también los planes de respuesta y planes de contingencia para cada riesgo.
Durante las reuniones de Revisión, se presentarán tanto al tutor como al cliente el avance alcanzado durante la iteración anterior. Se definirán prioridades, nuevos requerimientos y cambios en la planificación.

8.3. Estimación

Se muestra a continuación el listado de tareas necesarias para alcanzar los objetivos descritos anteriormente. Todos los esfuerzos se encuentran expresados en horas.

It.	Descripción	Esf.
	Propuesta de trabajo profesional	30
1	Aprendizaje desarrollo de aplicaciones descentralizadas	80
2	Aprendizaje de uso de IPFS como storage descentralizado	40
	Desarrollo de prueba de concepto (IPFS + Ethereum)	40
3	Investigación sobre métodos de encriptación	30
	Desarrollo de modulo de encriptación de modelos de ML	40
4	Investigación sobre Federated Learning y alternativas	30
	Desarrollo de modulo de Federated Learning	50
	Desarrollo de caso de uso para modulo de Federated Learning	30
5	Investigación sobre uso de React	20
	Desarrollo de aplicación web para marketplace descentralizado	70
6	Investigación de herramientas para desarrollo de smart contracts	30
	Desarrollo de infraestructura para ejecutar los smart contracts	30
7	Desarrollo de smart contracts para marketlace descentralizado	80
8	Integración de los modulos desarrollados con el marketplace descentralizado	50
	Pruebas del sistema en funcionamiento	30
	Reuniones	30
	Presentación	15

Además se debe tener en cuenta el tiempo dedicado a la administración del proyecto, estimando un 15 % del tiempo de desarrollo, lo cual resulta en:

Descripción	Esf.
Iteración 1	80
Iteración 2	80
Iteración 3	70
Iteración 4	110
Iteración 5	90
Iteración 6	60
Iteración 7	80
Iteración 8	80
Otros	75
Administración	105
Esfuerzo total	**830**

Descomponiendo las **830 hs** de esfuerzo total del proyecto en **meses de 70 hs** de esfuerzo por integrante (17 hs semanales), se llega a la estimación de una duración de **6 meses** para el presente trabajo.

8.4. Cronograma de entregables

A continuación se hace un cronograma tentativo de entregables al finalizar cada una de las iteraciones. Las mismas pueden ser modificadas en caso de verse necesario, por el tutor o por los desarrolladores del proyecto. La fecha de cada una de

las entregas serán determinadas en conjunto con el tutor.

It.	**Entregables**	**Hitos**
1	Directorio con documentación util para el proyecto	Curso de aplicaciones descentralizadas terminado [31] [32]
2	Prueba de concepto (IPFS + Ethereum)	Videos y lecturas sobre IPFS terminadas Prueba de concepto desarrollada
3	Modulo de encriptación de modelos	Investigación sobre hommomorphic encryption, multy-party encryption y alternativas terminada. Método de encriptación seleccionado Modulo de encriptación terminado
4	Modulo de Federated Learning	Investigación sobre Federated Learning y alternativas terminado. Pruebas de conepto usando Federated Learning terminadas. Desarrollo de framework de ML usando Federated Learning terminado.
5	Aplicación web para marketplace descentralizado	Desarrollo de frontend terminado.
6	Infraestructura para ejecutar los smart contracts Nodos IPFS en funcionamiento	Red para correr smarts contracts finalizada Investigación sobre las herramientas para despliegue de smart contracts finalizada Utilización de IPFS finalizada Pruebas en la infraestructura finalizadas
7	Smart contracts con lógica del negocio en funcionamiento	Desarrollo de smart contracts finalizados Pruebas de los smart contracts terminadas
8	Marketplace descentralizado usando smart contracts sobre Ethereum	Integración de frontend con smart contracts e IPFS terminado. Desarrollo de caso de prueba terminado Prueba exhaustiva de uno o dos casos de uso terminada

9 Referencias

[1] Satoshi Nakamoto. «Bitcoin: A Peer-to-Peer Electronic Cash System». En: (). URL: https://bitcoin.org/bitcoin.pdf.

[2] Keith Bonawitz y col. «Practical Secure Aggregation for Federated Learning on User-Held Data». En: (2016). URL: https://ai.google/research/pubs/pub45808.

[3] Jakub Konečný y col. «Federated Optimization: Distributed Machine Learning for On-Device Intelligence». En: (2016). URL: https://ai.google/research/pubs/pub45630.

[4] Jakub Konečný y col. «Federated Learning: Strategies for Improving Communication Efficiency». En: (2016). URL: https://ai.google/research/pubs/pub45648.

[5] Andrew Hard y col. «Federated Learning for mobile keyboard prediction». En: (2018). URL: https://ai.google/research/pubs/pub47586.

[6] Sebastian Caldas y col. «Expanding the Reach of Federated Learning by Reducing Client Resource Requirements». En: (2018). URL: https://ai.google/research/pubs/pub47774.

[7] Timothy Yang y col. «Applied Federated Learning: Improving Google Keyboard Query Suggestions». En: (2018). URL: https://ai.google/research/pubs/pub47655.

[8] *OpenMined*. Markeplace de modelos de deep learning. URL: https://www.openmined.org/.

[9] *DML*. Marketplace de modelos de machine learning. URL: https://decentralizedml.com/.

[10] *NumerAI*. Fondo de inversion que genera predicción con modelos de ML crowd-sourced. URL: https://numer.ai/.

[11] *Augur*. Protocolo de predicción de mercados en la Blockchain. URL: https://www.augur.net/.

[12] *Golem Network*. Supercomputadora descentralizada. URL: https://golem.network/.

[13] *OceanProtocol*. Ecosistema destinado a compartir datos como activos y ML como servicios de manera descentralizada. URL: https://oceanprotocol.com/.

[14] *Wibson*. Marketplace de datos descentralizado. URL: https://wibson.org/.

[15] Buterin Vitalik. *Ethereum: A secure descentralized generalized transaction ledger*. URL: https://www.ethereum.org.

[16] *Solidity*. Lenguaje de programación de smart contracts. URL: https://solidity.readthedocs.io/en/v0.5.3/.

[17] *ZeppelinOS*. Plataforma de desarrollo de smart contracts. URL: https://zeppelinos.org/.

[18] *OpenZeppelin*. Framework de desarrollo de smart contracts seguros. URL: https://openzeppelin.org/.

[19] *Truffle*. Framework de desarrollo de smart contracts. URL: https://truffleframework.com/truffle.

[20] *Ganache*. Testnet local para Ethereum Blockchain. URL: https://truffleframework.com/ganache.

[21] *Metamask*. Extensión para ejecutar aplicaciones descentralizadas en los web browsers. URL: https://metamask.io/.

[22] *Python*. Lenguaje de programación interpretado. URL: https://www.python.org/.

[23] *Scikit Learn*. Framework de machine learning para Python. URL: https://scikit-learn.org/stable/.

[24] *PyTorch*. Biblioteca de deep learning para Python. URL: https://pytorch.org/.

[25] *TensorFlow*. Biblioteca opensource para Machine Learning. URL: https://www.tensorflow.org/.

[26] *Keras*. Biblioteca de deep learning para Python que puede ejecutarse sobre Tensorflow. URL: https://keras.io/.

[27] *Tensorflow.js*. Biblioteca de machine learning que puede ser ejecutada en web browsers. URL: https://js.tensorflow.org/.

[28] *React*. Biblioteca de javascript para desarrollo de clientes web. URL: https://reactjs.org/.

[29] *IPFS*. Sistema descentralizado de almacenamiento de datos. URL: https://ipfs.io/.

[30] *SCRUM*. Metodología de desarrollo de Software, con ciclo de vida iterativo. URL: https://scrummethodology.com/.

[31] *The School of AI: Descentralized Applications course*. Curso de desarrollo de aplicaciones descentralizadas sobre la red de Ethereum. URL: https://www.theschool.ai/courses/decentralized-application/.

[32] *Coursera: Blockchain Specialization*. Especialización en Blockchain de coursera que contiene varios cursos. URL: https://www.coursera.org/specializations/blockchain.

10 Glosario

11 Curriculums

Fabrizio Sebastian Graffe

3162 Carlos Gardel, 1215 CABA
CABA, Argentina
☎ +54 11 42928787
+54 11 1551793613
✉ zebas.graffe@gmail.com

Education

2010 - Presente **Ingeniería en Informática**, *Facultad de Ingeniería, Universidad de Buenos Aires (UBA)*, Argentina.
Estudiante de ultimo año de la carrera de Ingeniería en Informática. Promedio: 7.07/10

Jul 2016 - Ago 2016 **Curso de Big Data**, *Mulesoft Academy*, Argentina.
Curso de Big Data usando Apache Spark como herramienta. Dictado por Luis Argerich (profesor en UBA).

Ago 2016 - Sept 2016 **Curso de Machine Learning**, *Mulesoft Academy*, Argentina.
Curso de Machine Learning usando Apache Spark como herramienta. Dictado por Luis Argerich (profesor en UBA).

Projects

Primavera 2013 **Clasificador de Textos** (C++)
• Diseño y desarrollo de un algoritmo con la capacidad de clasificar documentos de texto en grupos generados aplicando Clustering (técnica de aprendizaje no supervisado).
• Implementación de algoritmo de dos etapas. Primera etapa, elegir un grupo de semillas usando Clustering jerarquico aglomerativo. Segunda etapa, algoritmo K-Means usando las semillas elegidas en el paso anterior.
• Proyecto en equipo para la materia Organización de Datos (FIUBA).

Skills

Idiomas **Inglés**: Avanzado (escrito y hablado).
Actualmente asistiendo a clases in-company una vez a la semana.

Programación Scala, Python, Java, C, C#, C++, Node.js.

Tecnologías Linux, Apache Spark, Git, Pandas, Sklearn. (high, medium, low)

Data Engineer

Abr 2018 - Presente **Redbee studios**, Software factory:
Ofertador:
• Creación de prueba de concepto de motor de recomendaciones de productos financieros usando algoritmos de machine learning desarrollados en Python utilizando la biblioteca scikit-learn. Esto permitió la venta del proyecto a Cencosud. El motor fué portado mas adelante a Spark ML. • Las tareas involucraron exploración de los datos, investigación acerca de diferentes modelos de ML, desarrollo de prueba de concepto y luego de eso, un MVP el cual está actualmente desplegado y funcionando en un ambiente productivo. • Actualmente trabajando para mejorar la calidad de las recomendaciones, optimizar la performance del pipeline de datos y agregar nuevas fuentes de datos.

Feb 2017 - Abr 2018 **Redbee studios**, Software factory:
• Trabajo con equipo remoto situado en Seattle (USA) para empresa con producto de sistema de recomendaciones omnicanal con grandes clientes a nivel mundial. Comunicaciones en inglés.
Equipo Data, RichRelevance:
• Desarollo de ETLs que procesan comportamiendo de los usuarios en las páginas y producen reportes con insights sobre dichos datos en HDFS. Se usó Apache Spark (Scala) para el procesamiento de grandes volumenes de datos y ThoughtSpot para la creación de visualizaciones. • Se desarrolló una health check API (Play Framework) para reportar sitios con problemas, utilizando desviaciones standard de la media para calcular la severidad. • Git Flow como esquema de branchs.
Equipo Science Dev, RichRelevance:
• Tecnologías usadas: Scala + Apache Spark, Java + Apache Crunch. • Refactor a modelo de machine learning. • Desarrollo de prueba de concepto para implementación alternativa de uno de los productos de recomendación principales de la empresa. • Investigación para mejorar el uso de memoria para un algoritmo de predicción de clicks en ads.

Colaborador

2012 - 2015 **Algoritmos y Programación I**, Facultad de Ingeniería, Universidad de Buenos Aires:
▷ Preparación y dictado de clases ▷ Corrección de trabajos prácticos ▷ Asistencia a estudiantes durante clases prácticas
• El curso usa python como lenguaje de aprendizaje y cubre temas como estructuras condicionales, ciclos definidos e indefinidos, definición de funciones, estructuras de datos, algoritmos de ordenamiento, recursividad y nociones de programación orientada a objetos.

12 Plan de cursado

Graffe, Fabrizio Sebastian

Orientación: *Gestión Industrial de Sistemas*
1er Cuatrimestre 2019
Todas las materias de la carrera aprobadas.
Dedicará el cuatrimestre exclusivamente a la realización del trabajo profesional con el objetivo de finalizar la carrera de grado.

Rojas, Agustin

Orientación: *Gestión Industrial de Sistemas*
1er Cuatrimestre 2019
Todas las materias de la carrera aprobadas.
Dedicará el cuatrimestre exclusivamente a la realización del trabajo profesional con el objetivo de finalizar la carrera de grado.

13 Carta al departamento de computación

Sr. Director del Departamento de Computación de la Facultad de Ingeniería de la Universidad de Buenos Aires, Ing. Pablo Cosso
De mi mayor consideración:

Por la presente me dirijo a Usted a fin de elevar a su consideración el acta donde consta mi conformidad y la de los estudiantes Graffe, Fabrizio (Padrón No 93158) y Rojas, Agustin (Padrón No 91462), en la que se acuerda el tema de su Proyecto de Trabajo Profesional en Ingeniería Informática: "DeltaML: Machine Learning descentralizado utilizando Blockchain para fomentar participación en la red". Se acompañan copias del Proyecto de Trabajo Profesional así como los datos de los Srs. Graffe, Fabrizio y Rojas, Agustin.

Atentamente
Firma del Profesor responsable

14 Acta

En Buenos Aires al 20 día del mes de febrero del año dos mil diecinueve (2019) se reúnen en el Departamento de Computación el profesor Beiró, Mariano y los estudiantes de la carrera de Ingeniería en Informática Srs. Graffe, Fabrizio (Padrón No 93158) y Rojas, Agustin (Padrón No 91462) para acordar el Tema de Trabajo profesional para el Ciclo Superior de la Carrera de los Srs. Graffe, Fabrizio y Rojas, Agustin.

Luego de haber conversado sobre las áreas de interés de los estudiantes y habiendo propuesto el profesor una lista posible de temas, se acuerda establecer como Tema de Trabajo Profesional en Ingeniería Informática de los Srs. Graffe, Fabrizio y Rojas, Agustin: "DeltaML: Machine Learning descentralizado utilizando Blockchain para fomentar participación en la red".

El profesor Beiró, Mariano deja constancia que en su opinión, al haber elegido los Srs. Graffe, Fabrizio y Rojas, Agustin las asignaturas de la Orientación, de colocar la orientación de la Carrera de Ingeniería en Informática y sus correspondientes electivas, se puede dar por cumplida la elaboración del Plan de Estudio Personal. Deja constancia que los contenidos de las asignaturas de la orientación abarcan los conocimientos necesarios para desarrollar satisfactoriamente el trabajo profesional.

Sin más que tratar, se da por concluida la reunión firmándose tres ejemplares de la presente acta, uno para elevar a la Dirección del Departamento de Computación, otro para los estudiantes y el tercero para el profesor.

Srs. Graffe, Fabrizio y Rojas, Agustin
Firma de profesor responsable