Introducción al curso

Has comenzado una de las más completas revisiones de las herramientas de la ciencia de los datos que encontrarás en Internet actualmente. Esto no quiere decir que cubramos todas y cada una de las herramientas, pero más adelante en el curso introduciremos una lista comprensiva de las tares que un científico de datos necesita realizar y te daremos las dos o tres principales herramientas comerciales y de código abierto disponibles para completarlas. De igual forma explicamos cómo las herramientas se entrelazan en funcionalidad, cuáles son sus pros y contras y cómo estas herramientas pueden abordar todas las tuberías de datos de la ciencia de los datos. Vamos a comenzar con los datos. Los datos son obviamente importantes para los científicos de datos. En este curso, te mostraremos cómo gestionar, extraer, transformar, analizar y visualizar los datos. Ahora, podrías ser capaz de sobrevivir a la ciencia de los datos son ningún conocimiento en programación si usas el conjunto de herramientas adecuadas. Sin embargo, recomendamos que te familiarices con la programación y los marcos de programación relacionados con la ciencia de los datos. Para ayudarte, te presentaremos los lenguajes y marcos de programación más usados en la ciencia de los datos. Bien, dicho esto, hay demasiadas opciones de automatización disponibles en las ultimas herramientas que un científico de datos pueda usar. En este curso, explicaremos cómo hacer uso de dichas herramientas para ahorrar tiempo y descubrir la inspiración. La programación visual está disponible en muchas herramientas. En este curso, aprenderás como la programación visual puede ser usada para acelerar el tiempo de desarrollo y ayuda a los no programadores a entrar en el campo de la ciencia de los datos. El software de código abierto es el líder en el campo de la ciencia de los datos, pero el Costo Total de Propiedad, o “TCO”; a veces puede ser más altos para los software de código abierto. Debido a los costos de configuración, personalización y mantenimiento. Como resultado, el software comercial también tiene su lugar, especialmente desde la nueva generación de software comerciales de la ciencia de los datos que aprovecha el software de código abierto y los estándares abiertos. Lo cual facilita la migración entre las herramientas y puede reducir el TCO. En este curso, te presentaremos tanto los software de código abierto así como los comerciales y señalaremos sus fortalezas y debilidades para la ciencia de los datos. De igual forma te mostraremos las formas de aprovechar sus fortalezas. Finalmente, te mostraremos cómo la computación en la nube puede ser usada para acelerar y facilitar el trabajo de los científicos de datos. Te presentaremos las herramientas basadas en la nube más nuevas y comúnmente usadas para la ciencia de los datos. Además de las lecciones, este curso tiene numerosos laboratorios para que te familiarices con el material y también obtengas experiencia en las prácticas. También hay múltiples pruebas para evaluar tu aprendizaje. ¡No queda más nada por decir, así que empecemos! Estamos muy contentos de tenerte con nosotros para comenzar tu viaje en la ciencia de los datos. Si tienes algún problema con el material del curso, no dudes en contactarnos en el foro de discusión. ¡Así que vamos a comenzar!

Lenguajes de la Ciencia de los Datos

Los Lenguajes de la Ciencia de los Datos. Para cualquiera que este comenzando su viaje en la ciencia de los datos, la variedad de opciones técnicas pueden ser abrumadoras. Hay una gran cantidad impresionante de opciones cuando se trata de lenguajes de programación. Cada uno tiene sus propias fortalezas y debilidades y no hay una respuesta correcta a la pregunta de cual deberías aprender primero. La respuesta a esa pregunta depende en gran medida a tus propias necesidades, los problemas que intentas resolver o para quien los estas resolviendo. Python, R, y SQL son los lenguajes que te recomendamos que consideres como primeras opciones. Pero hay muchos otros que tienen sus propias fortalezas y características interesantes. Scala, Java, C++ y Julia son algunos de los más populares. Javascript, PHP, Go, Ruby y Visual Basic todos estos tienen también sus propios casos de uso. El lenguaje que escojas para aprender dependerá de las cosas que necesites lograr y de los problemas que necesites resolver. Esto también dependerá de la compañía en la que trabajes el rol que tengas y la edad de tu aplicación actual. Exploraremos las respuestas a esta pregunta a medida que profundicemos en los lenguajes populares en la industria de la ciencia de los datos. Hay muchos roles disponibles para las personas que están interesadas en involucrase en la ciencia de los datos. Analista de Negocios Ingeniero de Base de Datos, Analista de Datos, Ingeniero de Datos, Científico de Datos, Científico de Investigación, Ingeniero de Software, Gestor de Producto Estadísticos, Gestor de Proyecto y muchos más. Profundicemos un poco en lo que aprenderemos en la Lección 1. Nos enfocaremos en los tres principales lenguajes de la ciencia de los datos: Python, R y SQL donde cada uno de los cuales tienen sus propias lecciones. Después pasaremos a resaltar otros lenguajes notables y lo que los hace especiales. ¡Luego, terminaremos con una pequeña prueba para que demuestres tus conocimientos! Aquí está lo que discutiremos: Python R SQL, otros lenguajes dignos de tener en cuenta en la Ciencia de los Datos y pruebas prácticas

# Introducción a Python

En este video, revisaremos las características del alto nivel del lenguaje de programación de Python. Python es un lenguaje muy poderoso. Es el lenguaje de programación más popular para la ciencia de los datos. De acuerdo a la Encuesta Kaggle de la Ciencia de los Datos y Aprendizaje Automático, el 75% de los 10,000 encuestados alrededor del mundo reportaron que usan Python con regularidad. Glassdoor informó que en el año 2019 más del 75% de los puestos de la ciencia de los datos listados incluían Python en sus descripciones de cargo. Cuando se les preguntó cuál lenguaje debería aprender primero un aspirante a científico de datos, la mayoría de los científicos de datos contestaron Python. Probablemente estés pensando, ¿Por qué Python es tan popular? Bien, comencemos con las personas que usan Python. Si ya sabes cómo programar, entonces Python es excelente para ti porque usa una sintaxis, clara y legible. Puedes hacer muchas de las cosas que estás acostumbrado a hacer en otros lenguajes de programación pero con Python puedes hacerlos usando menos códigos. Si quieres aprender a programar, este es un gran lenguaje para comenzar debido a la enorme comunidad global y a la gran cantidad de documentación. De hecho, muchas encuestas realizadas en el año2019, revelaron que más del 80% de los profesionales de datos alrededor del mundo usan Python. Python es útil para muchas situaciones, incluyendo la ciencia de los datos, inteligencia artificial, aprendizaje automático, desarrollo web y dispositivos IoT como Raspberry Pi. Las grandes organizaciones que usan Python incluyen IBM, Wikipedia, Google, Yahoo!, CERN, NASA, Facebook, Amazon, Instagram, Spotify, y Reddit. Python es un poderoso lenguaje de programación de propósito general con el que se pueden hacer demasiadas cosas. Es ampliamente soportado por una comunidad global y dirigido por la Python Software Foundation.

Reproduce el video desde :2:13 y sigue la transcripción2:13

Python es un lenguaje de programación de propósito general de muy alto nivel que puede ser aplicado a diferentes clases de problemas.

Reproduce el video desde :2:20 y sigue la transcripción2:20

Este tiene una gran librería estándar la cual proporciona las herramientas adecuadas para tareas diferentes, incluyendo pero no limitado a las bases de datos, automatización, web scraping, procesamiento de texto, procesamiento de imágenes, aprendizaje automático y análisis de datos.

Reproduce el video desde :2:36 y sigue la transcripción2:36

Para la ciencia de los datos, puedes usar las librerías de computación científica de Python tales como Pandas, NumPy, SciPy, y Matplotlib.

Reproduce el video desde :2:45 y sigue la transcripción2:45

Para la inteligencia artificial, tiene TensorFlow, PyTorch, Keras, y Scikit-learn.

Reproduce el video desde :2:52 y sigue la transcripción2:52

Python también puede ser usado para el Procesamiento del Lenguaje Natural o por sus siglas en inglés (NLP) utilizando Natural Language Toolkit (NLTK). Otro punto de venta atractivo es la comunidad Python, la cual tiene una historia muy bien documentada de pavimentar el camino para la diversidad y los esfuerzos de inclusión en la industria de la tecnología como un conjunto. El lenguaje de Python tiene un código de conducta ejecutado por la Python Software Foundation la cual busca garantizar la seguridad y la inclusión de todos, tanto en las comunidades Python en línea como en persona. De igual forma hay comunidades como lo es PyLadies la cual busca la forma de crear espacios para que las personas interesadas en Python aprendan en entornos seguros e inclusivos. PyLadies es un grupo internacional de mentores que se enfoca en ayudar a más mujeres a convertirse en participantes activas y líderes en la comunidad de código abierto de Python.

# Introducción al Lenguaje R

En este video, discutiremos una breve visión general acerca del lenguaje de programación R. Después de nuestro último video acerca de Python, donde discutimos su amplia adopción, puede que te estés preguntando por qué deberías considerar el aprendizaje de cualquier otro lenguaje. Bien, según los resultados de la Encuesta Kaggle de la Ciencia de los Datos del año 2019, la cual tuvo más de 10k encuestados alrededor del mundo, ¡el aprender hasta 3 lenguajes puede aumentar tu salario! Y el lenguaje R tiene mucho que ofrecerte. Así como Python, R es gratuito, pero es un Proyecto GNU – en vez de ser de código abierto, es en realidad software libre. Por lo que si Python es de código abierto R es software libre, ¿cuál es la diferencia? Bueno, tanto el código abierto como el software libre generalmente se refieren al mismo conjunto de licencias. Muchos proyectos de código abierto usan la Licencia Pública General GNU, por ejemplo. Tanto el código abierto como el software libre soportan la colaboración. En la mayoría de los casos (pero no en todos), estos términos pueden ser usados intercambiablemente. La Open Source Initiative (OSI) apoya el código abierto mientras que la Free Software Foundation (FSF) defiende el software libre. El código abierto se enfoca más en las empresas, mientras que el software libre se enfoca más en un conjunto de valores. Regresemos a por qué deberías aprender Lenguaje R. Porque este es un proyecto de software libre, puedes usar el lenguaje de la misma forma que contribuyes al código abierto y permite la colaboración pública al igual que el uso privado y comercial. Además, el lenguaje R es otro lenguaje soportado por una amplia comunidad mundial de personas apasionadas por hacer posible el uso del lenguaje para poder resolver grandes problemas. ¿Para quién es el lenguaje R? Es muy usado por estadísticos, matemáticos y mineros de datos para poder desarrollar los software estadísticos, gráficos y análisis de datos. La sintaxis orientada a los arrays del lenguaje facilita la traducción de las matemáticas al código, especialmente para alguien sin o con tan solo un poco de conocimiento de programación. De acuerdo a la Encuesta Kaggle de la Ciencia de los Datos y Aprendizaje Automático, la mayoría de las personas aprende el lenguaje R aunque ya tengan unos cuantos años en su carrera de la ciencia de los datos, pero sigue siendo un lenguaje muy cómodo para aquellos que no tiene experiencia con los software de programación. El lenguaje R es muy popular en el mundo académico, pero entre las empresas que usan R están IBM, Google, Facebook, Microsoft, Bank of America, Ford, TechCrunch, Uber, y Trulia. ● R se ha convertido en el mayor repositorio de conocimiento estadístico al nivel mundial. ● A partir del año 2018, R tiene más de 15,000 paquetes públicamente lanzados, haciendo posible realizar complejos análisis exploratorios de datos. ● R se integra muy bien con otros lenguajes de computación, como lo son C++, Java, C, .Net y Python. ● Operaciones matemáticas convencionales como lo son la multiplicación de matrices la cuales funcionan desde la base. ● R posee más facilidades de programación orientadas a los objetos en comparación de la mayoría de los lenguajes de computación estadística. Existen muchas formas de conectarse con otros usuarios de R a nivel mundial. Las comunidades como user!, WhyR?, SatRdays y R-Ladies son fabulosas para poder conectarse. Y también puedes echar un vistazo a los sitios webs de R project para saber más acerca de las conferencias y eventos R.

# Introducción a SQL

En este video, vamos a echar un vistazo de alto nivel a SQL. SQL es un poco diferente de los otros lenguajes que hemos discutido hasta ahora. Primero que nada, se pronuncia formalmente “ess cue el”, aunque algunas personas lo dicen “sequel.” Aunque las siglas significan “Structured Query Language o Lenguaje de consulta estructurada”, muchas personas no consideran que SQL sea como otros lenguajes de desarrollo de software porque es un lenguaje no procedimental y su alcance está limitado a la consulta y gestión de datos. Aunque no es un lenguaje de “la ciencia de los datos” en sí, ¡los científicos de los datos normalmente lo utilizan ya que es es muy simple y poderoso! Otros hechos importantes acerca de SQL: es mucho más antiguo que Python y R, por al menos unos 20 años, ya que apareció por primera vez en el año 1974. Y, ¡SQL fue desarrollado en IBM! Este lenguaje es muy útil en el manejo de datos estructurados, es decir, los datos incorporan relaciones entre las entidades y las variables. SQL fue diseñado para la gestión de datos en bases de datos relacionales. Aquí podrás ver un diagrama el cual muestra la estructura general de una base de datos relacional. Una base de datos relacional está formada por colecciones de tablas de dos dimensiones, ejemplo, los conjuntos de datos y las hojas de cálculo de Microsoft Excel. Cada una de estas tablas entonces está formada por un número fijo de columnas y un número cualquiera de filas. ¡PERO! Aunque SQL fue originalmente desarrollado para su uso con bases de datos relacionales, ya que es tan generalizado y fácil de utilizar han sido desarrolladas interfaces SQL para muchos NoSQL y grandes repositorios de datos. El lenguaje SQL esta subdividido en varios elementos de lenguaje, incluyendo las clausulas, expresiones, pronósticos, consultas y declaraciones. Entonces ¿qué es lo que hace tan grandioso a SQL? Saber SQL te ayudará a realizar muchos trabajos diferentes en la ciencia de los datos, incluyendo empresas y análisis de datos, es algo necesario para la ingeniería de datos. Cuando las operaciones son realizadas con SQL, accedes a los datos directamente. No existe la necesidad de copiarlos de antemano. Esto puede acelerar considerablemente las ejecuciones del flujo de trabajo. SQL es el intérprete entre tú y la base de datos. SQL está bajo el Instituto Nacional Estadounidense de Estándares o los estándares “ANSI” por lo que si aprendes SQL y lo usas con una base de datos, serás capaz de aplicar fácilmente ese conocimiento de SQL a otras bases de datos. Existen muchas bases de datos SQL distintas, incluyendo MySQL, IBM Db2, PostgreSQL, Apache OpenOffice Base, SQLite, Oracle, MariaDB, Microsoft SQL Server y muchas más. La sintaxis de SQL que escribes podría cambiar un poco basado en el sistema de gestión de bases de datos relacionales que uses. Si quieres aprender SQL lo mejor que deberías hacer es concentrarse en una base de datos relacional específica y luego conéctate a la comunidad para esa plataforma en específico. ¡De igual forma existen muchos cursos de introducción a SQL disponibles!

# Otros Lenguajes

Hasta ahora, hemos discutido acerca de Python, R y SQL. En este video, discutiremos algunos otros lenguajes que tienen casos de uso convincentes para la ciencia de los datos. Bueno, entonces indiscutiblemente Python, R y SQL son los tres lenguajes más populares que utilizan los científicos de datos. Pero, hay muchos otros lenguajes que también valen la pena que te tomes un tiempo para considerar cual lenguaje usar para resolver un problema particular de la ciencia de los datos. Scala, Java, C++ y Julia son probablemente los lenguajes más tradicionales de la ciencia de los datos en esta diapositiva. ¡Pero JavaScript, PHP, Go, Ruby, Visual Basic y otros también han encontrado su lugar en la comunidad de la ciencia de los datos! No profundizaré tanto en cada uno de estos lenguajes, pero mencionaré algunos aspectos resaltantes. Java es un lenguaje de programación orientado a objetos de propósito general probado y verdadero. Este ha sido ampliamente adoptado en el espacio empresarial y está diseñado para ser rápido y escalable. Las aplicaciones Java son compiladas en código de bytes y se ejecutan en la Máquina Virtual Java o “JVM” Algunas herramientas notables de la ciencia de datos construidas con Java incluyen Weka para la minería de datos, Java-ML, el cual es una librería de aprendizaje automático, Apache MLlib el cual hace que el aprendizaje automático sea escalable y Deeplearning4j para el aprendizaje profundo. Apache Hadoop es otra aplicación basada en Java. Esta gestiona el procesamiento y almacenamiento de datos para las grandes aplicaciones de datos que son ejecutadas en sistemas clúster. Scala es un lenguaje de programación de uso general que proporciona el soporte a la programación funcional y a un sistema de tipo estático fuerte. Muchas de las decisiones de diseño en la construcción del lenguaje Scala fueron hechas para responder a las criticas de Java. Scala igualmente es interoperable con Java, ya que se ejecuta en JVM. El nombre “Scala” es una combinación de “scalable” y “language”. Este lenguaje está diseñado para crecer junto con las demandas de los usuarios. En la ciencia de los datos, el programa con mayor popularidad construido utilizando Scala es Apache Spark. Spark es un sistema de computación en clúster rápido y de propósito general. Este proporciona APIs que hacen más fácil la escritura de los trabajos paralelos y un motor optimizado que soporta los gráficos de computación general. Spark incluye Shark el cual es un motor de consulta, MLlib, para el aprendizaje automático, GraphX, para el procesamiento de gráficos y Spark Streaming. Apache Spark fue diseñado para ser más rápido que Hadoop. C++ es lenguaje de programación de propósito general. Es una extensión del lenguaje de programación C o “C con Clases”. C++ mejora la velocidad de procesamiento, permite la programación de Sistema y también permite un mejor control sobre la aplicación de software. Hay muchas organizaciones que usan Python u otros lenguajes de alto nivel para el análisis de datos y las tareas de exploración aún siguen confiando en C++ para desarrollar programas que alimentan esos datos a los clientes en tiempo real. Para la ciencia de los datos, una librería popular de aprendizaje profundo para el flujo de datos llamada TensorFlow fue construida con C++. Pero mientras que C++ es la base de TensorFlow, es ejecutada en una interfaz de Python, por lo cual no necesitas saber C++ para poder usarlo. MongoDB, una base de datos NoSQL para la gestión de grandes volúmenes de datos, fue construida con C++. Caffe es un repositorio de algoritmos de aprendizaje profundo construido con C++, Python y MATLAB bindings. Una tecnología central para la World Wide Web, JavaScript es un lenguaje de propósito general que se ha extendido más allá del navegador con la creación de Node.js y otros alcances del server side. Javascript NO está relacionado con el lenguaje Java. Para la ciencia de los datos, la implementación más popular es sin duda alguna TensorFlow.js. TensorFlow.js hace posible el aprendizaje automático y el aprendizaje profundo bien sea en Node.js o en el navegador. TensorFlow.js igualmente fue adoptado por otras librerías de código abierto incluyendo brain.js y machinelearn.js. El Proyecto R-js es otra gran implementación de JavaScript para la ciencia de los datos. R-js ha reescrito las especificaciones de algebra lineal del lenguaje R en Typescript. Esta reescritura permitirá una base para que otros proyectos implementen marcos de trabajo de base matemáticas más poderosos como lo son Numpy y SciPy de Python. Typescript es un super conjunto de JavaScript. Julia fue diseñado en el MIT para el análisis numérico de alto desempeño y la ciencia computacional. Permite un desarrollo rápido como Python o R, mientras produce programas que se ejecutan tan rápido como los programas de C o Fortran. Julia esta compilado lo que quiere decir que el código es ejecutado directamente en el procesador como código ejecutable y llama a las librerías de C, Go, Java, MATLAB, R, Fortran y Python e igualmente ha refinado el paralelismo. El lenguaje Julia es relativamente nuevo, ya que fue escrito en el año 2012, pero es muy prometedor para el futuro impacto en las industrias de la ciencia de los datos. JuliaDB es particularmente una aplicación útil de Julia para la ciencia de los datos. Es un paquete para trabajar con conjuntos de datos grandes y persistentes. Hasta ahí profundizaremos con respecto a los muchos lenguajes que son usados para resolver los problemas de la ciencia de los datos. Si tienes experiencia con algún lenguaje en particular, te recomiendo que hagas una búsqueda en la web para ver que podría ser posible en términos de usarlo en la ciencia de los datos. ¡Podrías estar sorprendido de las posibilidades!

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

# Categorias de las Herramientas de la Ciencia de los Datos

Las herramientas de código abierto están disponibles para diversas tareas de la ciencia de los datos. En este video, veremos las diferentes tareas de la ciencia de los datos. En los siguientes videos veremos las herramientas de código abierto más usadas para esas tareas. Las herramientas más importantes se cubren a lo largo de este curso. La Gestión de Datos es el proceso de persistencia y recuperación de datos. La Integración y Transformación de Datos, comúnmente conocida como Extracción, Transformación y Carga o por sus siglas en inglés “ETL”, es el proceso de recuperación de datos de sistemas de gestión de datos remotos. La transformación y carga de los datos en un Sistema local de gestión de datos también es parte de la Integración y Transformación de Datos. La Visualización de Datos es parte de un proceso inicial de exploración de datos, así como parte de un producto final. La Construcción de Modelo es el proceso de creación de un modelo de aprendizaje automático o aprendizaje profundo usando un algoritmo apropiado con muchos datos. El despliegue del modelo permite poner a disposición a las aplicaciones de terceros un modelo de aprendizaje automático o de aprendizaje profundo. El seguimiento y evaluación de los modelos, asegura un control continuo de la calidad del rendimiento de los modelos desplegados. Estas verificaciones son para la exactitud, equidad y la robustez del adversario. La gestión de activos de código usa el control de versiones y otras características de colaboración para facilitar el trabajo en equipo. La gestión de activos de datos implica los mismos componentes de versiones y de colaboración en los datos. La gestión de activos de datos a su vez apoya la replicación, copia de seguridad y la gestión de los derechos de acceso. Los entornos de desarrollo, comúnmente conocidos como Entornos de Desarrollo Integrado o “IDEs”, son herramientas que ayudan al científico de datos a implementar, ejecutar, probar y desplegar su trabajo. Los entornos de ejecución son herramientas en las que se realiza el pre procesamientos de datos, la capacitación de modelos y el despliegue. Finalmente, hay disponible una herramienta visual completamente integrada que cubre todos los componentes de la herramienta previa, bien sea parcial o por completo. Con esto concluye este video. En el siguiente video empezaremos a ver las herramientas de código abierto para tareas de la ciencia de los datos. Interfaz de usuario gráfica

Descripción generada automáticamente

# Herramientas de Código Abierto para la Ciencia de los Datos - Parte 1

En la parte uno de esta serie de dos partes, cubriremos la gestión de datos, integración de datos de código abierto, transformación y las herramientas de visualización. Las herramientas de gestión de datos de código abierto más usadas son las bases de datos relacionales como MySQL y PostgreSQL; NoSQL, MongoDB Apache CouchDB y Apache Cassandra y las herramientas basadas en archivos como Hadoop File System o los sistemas de archivos en la nube como Ceph. Por último, Elasticsearch es principalmente usado para almacenar datos de texto y crear un índice de búsqueda para una rápida recuperación de documentos. La tarea de integración de datos en el mundo clásico de almacenamiento de datos es llamado ETL que según sus siglas quiere decir “extraer, transformar y cargar”. Hoy en día los científicos de datos generalmente proponen el término “ELT” – Extraer, Cargar y Transformar “ELT” resalta el hecho de que los datos son volcados en algún lugar y el ingeniero de datos o el mismo científico de datos es el responsable de los datos. Hay otro termino para este proceso que ha surgido actualmente: “refinamiento y limpieza de datos”. Estas son las herramientas de integración y transformación de datos de código abierto más usadas: Apache AirFlow, originalmente creada por AirBNB; KubeFlow la cual permite ejecutar las tuberías de datos de la ciencia de los datos en Kubernetes; Apache Kafka la cual se originó de LinkedIn, Apache Nifi la cual ofrece un editor visual amigable, Apache SparkSQL (el cual te permite usar ANSI SQL y escalar para computar clústeres de 1000s de nodos) y NodeRED el cual también proporciona un editor visual. NodeRED consume tan pocos recursos que incluso funciona en pequeños dispositivos como Raspberry Pi. Bueno, ahora presentaremos las herramientas de visualización de datos de código abierto más usadas. Debemos distinguir entre las librerías de programación en las que debes usar el código y las herramientas que contienen una interfaz de usuario. Las librerías más populares son cubiertas en los próximos videos. Pixie Dust es una librería, pero viene con una interfaz de usuario la cual ayuda enormemente a cualquiera quien necesite graficar en Python. Hue tiene un enfoque similar el cual puede crear visualizaciones a partir de consultas en SQL. Kibana, es una aplicación web de exploración y visualización de datos está limitada a Elasticsearch (el proveedor de datos). Finalmente, Apache Superset es una aplicación web de exploración y visualización de datos. El despliegue del modelo es extremadamente importante. Una vez que has creado un modelo de aprendizaje automático capaz de predecir algunos aspectos claves del futuro, deberías hacer que ese modelo sea consumible por otros desarrolladores luego convertirlo a una API. Apache PredictionIO actualmente solo soporta modelos Apache Spark ML para el despliegue pero que sea soportada por cualquier tipo de librería está en camino. Seldon es un producto interesante ya que soporta todos los marcos de trabajo concluyendo TensorFlow, Apache SparkML, R, y scikit-learn. Seldon puede funcionar en Kubernetes y Redhat OpenShift. Otra manera de desplegar los modelos de SparkML es utilizando MLeap. Finalmente, TensorFlow puede servir con cualquiera de sus modelos utilizando el servicio de TensorFlow. Puedes desplegarlo en un dispositivo integrado como Raspberry Pi o un teléfono inteligente usando TensorFlow Lite e incluso desplegarlo en un navegador web usando TensorFlow dot JS. El monitoreo del modelo es otro paso crucial. Una vez que has desplegado un modelo de aprendizaje automático, es necesario hacer un seguimiento de su desempeño de predicción a medida que los nuevos datos llegan con el fin de mantener los modelos desactualizados. A continuación, encontrarás algunos ejemplos de herramientas de seguimiento de modelos: ModelDB es una base de metadatos de modelos de máquinas donde la información sobre los modelos puede ser almacenada y consultada. Nativamente soporta Apache Spark ML Pipelines y scikit-learn. Una herramienta genérica multi propósitos llamada Prometheus es también usada ampliamente para el seguimiento del modelo de aprendizaje automático, aunque no está hecha específicamente para este propósito. El desempeño del modelo no es medido exclusivamente con respecto a la precisión. Los modelos de tendencias contra grupos protegidos como el género o la raza también son importantes. El kit de herramientas de código abierto de IBM AI Fairness 360 hace precisamente esto. Detecta y mitiga las tendencias en los modelos de aprendizaje automático. Los modelos de aprendizaje automático, especialmente los modelos de aprendizaje profundo basados en las redes neuronales que pueden ser objeto de ataques adversos donde un atacante trata de engañar al modelo con datos manipulados o manipulando el propio modelo. El IBM Adversarial Robustness 360 Toolbox puede ser usado para detectar la vulnerabilidad de los ataques adversarios y ayuda a hacer mucho más robusto el modelo. Los modelos de aprendizaje automático frecuentemente son considerados como una caja negra que aplica algo de “magia” misteriosa. El kit de herramientas IBM AI Explainability 360 hace el proceso de aprendizaje automático mas entendible al encontrar ejemplos similares dentro del conjunto de datos que puede ser presentado a un usuario para su comparación de forma manual. El kit de herramientas IBM AI Explainability 360 también puede ilustrar la capacitación de un modelo de aprendizaje automático más simple al explicar cómo las diferentes variables de entrada afectan la decisión final del modelo. Las opciones de las herramientas de gestión de activos de código han sido ampliamente simplificadas: Para la gestión de activos de código – también conocida como gestión de versiones o control de versiones – Git es hoy en día el estándar. Han surgido múltiples servicios para soportar a Git, siendo claramente el más destacado GitHub, el cual proporciona alojamiento para la gestión de versiones de desarrollo de software. El segundo puesto es definitivamente para GitLab el cual ha tenido la ventaja de ser una plataforma completamente de código abierto que puedes alojar y administrar tú mismo. Otra opción interesante es Bitbucket. La gestión de activos de datos también conocida como gobernanza de datos o linaje de datos es otra parte crucial de la ciencia de los datos a nivel empresarial. Los datos tienen que ser versionados y anotados con metadatos. Apache Atlas es una herramienta que soporta esta tarea. Otro proyecto interesante es ODPi Egeria el cual es gestionado a través de Linux Foundation y es un entorno abierto. Este ofrece un conjunto de APIs abiertas, tipos y protocolos de intercambio que los repositorios de metadatos usan para compartir e intercambiar los datos. Y por último, Kylo que es una plataforma de software de código abierto para la gestión de Data Lake que provee un amplio soporte para la gran variedad de tareas de gestión de activos de datos. Con esto concluimos la primera parte de esta serie de dos partes. Ahora continuemos con la segunda parte.

# Herramientas de Código Abierto para la Ciencia de los Datos - Parte 2

En esta sección, vamos a cubrir el entorno de desarrollo, la integración de datos de código abierto, transformación y herramientas de visualización. Uno de los entornos de desarrollo actualmente más populares que los científicos de datos están utilizando es “Jupyter”. Jupyter surgió primeramente como una herramienta para la programación interactiva en Python, ahora soporta más de cien lenguajes de programación por medio de “kernels”. Kernels no deberían ser confundidos con los kernels del sistema operativo. Los Jupyter kernels están encapsulando los distintos interpretes interactivos para los diferentes lenguajes de programación. Una propiedad clave de Jupyter Notebooks es la habilidad de unificar la documentación, código, salida del código, los interpretes de comandos y las visualizaciones en un solo documento. JupyterLab es la siguiente generación de Jupyter Notebooks y a largo plazo reemplazará a Jupyter Notebooks. Los cambios a nivel de arquitectura que se están introduciendo en JupyterLab hacen que Jupyter sea más moderno y modular. Desde la perspectiva del usuario, la principal diferencia introducida por JupyterLab es la habilidad de abrir distintos tipos de archivos, incluyendo Jupyter Notebooks, datos y terminales. Entonces puedes arreglar estos archivos en el canvas. Aunque Apache Zeppelin ha sido completamente reimplantado, este está inspirado por Jupyter Notebooks y proporciona una experiencia similar. Un elemento clave es la capacidad de graficar integrado. En Jupyter Notebooks, es necesario el uso de librerías externas en Apache Zeppelin y el graficar no requiere de un código. También puedes extender estas capacidades mediante el uso de librerías adicionales. RStudio es uno de los entornos de desarrollo más antiguos para la estadística y la ciencia de los datos, siendo introducida en el año 2011. Es exclusivamente ejecutada en R y todas las librerías asociadas a R. Sin embargo, el desarrollo en Python es posible y R está estrechamente integrado en esta herramienta para proporcionar una experiencia de usuario óptima. RStudio unifica la programación, ejecución, depuración, acceso remoto a los datos, exploración de datos y visualización en una sola herramienta. Spyder intenta imitar el comportamiento de RStudio para llevar su funcionalidad al mundo de Python. Sin embargo, Spyder no tiene el mismo nivel de funcionalidad que RStudio, los científicos de datos lo consideran como una alternativa. Pero en el mundo de Python, Jupyter es usado con más frecuencia. Este diagrama muestra cómo Spyder integra el código, la documentación, las visualizaciones y otros componentes en un simple canvas. A veces tus datos no entran en la capacidad de almacenamiento o memoria principal de una sola computadora. Es allí donde entran los entornos de ejecución de clúster. El bien conocido marco de trabajo de computación en clúster Apache Spark es uno de los proyectos Apache más activos y es usado a lo largo de todas las industrias, incluyendo muchas compañías en la lista de Fortune 500. La propiedad clave de Apache Spark es la escalabilidad lineal. Lo que quiere decir que, si duplicas el número de servidores en un clúster, también duplicarás su desempeño. Después de que Apache Spark comenzó a ganar cuota de mercado y Apache Flink fue creado. La diferencia clave entre Apache Spark y Apache Flink es que Apache Spark es un motor de procesamiento de datos por lotes, capaz de procesar grandes cantidades de archivos de datos por archivo. Apache Flink, por otra parte, es una imagen de procesamiento de flujo de datos con su enfoque principal en el procesamiento de flujos de datos en tiempo real. Aunque el motor soporta ambos paradigmas de procesamientos de datos, Apache Spark es generalmente la opción en la mayoría de los casos de uso. Uno de los últimos desarrollado en los entornos de ejecución de la ciencia de los datos es llamado “Ray”, el cual se enfoca claramente en la formación de modelos de aprendizaje profundo a gran escala. Echemos un vistazo a las herramientas de código abierto para los científicos de datos que están completamente integradas y son visuales. Con estas herramientas, no necesitas conocimientos de programación. Las tareas más importantes son soportadas por estas herramientas, estas tareas incluyen la integración de datos, la transformación, la visualización de datos y la construcción de modelos. KNIME desarrollado en la Universidad de Constanza en el año 2004. Como puedes ver KNIME tiene una interfaz de usuario visual con la habilidad de arrastrar y soltar. También tiene capacidades de visualización ya incorporadas. Knime puede ser extendida para programar con R y Python y tiene conectores con Apache Spark. Otro ejemplo de este grupo de herramientas es Orange. Es menos flexible que KNIME pero mucho más fácil de usar. En esta sección, has aprendido acerca de las tareas más comunes en la ciencia de los datos y cuales herramientas de código abierto son relevantes para estas tareas. En el próximo video, discutiremos algunas herramientas comerciales establecidas que encontrarás en tu experiencia con la ciencia de datos. Vamos al siguiente video para obtener más detalles.

# Herramientas Comerciales para la Ciencia de los Datos

Previamente cubrimos las herramientas de código abierto para la ciencia de los datos. Ahora, veamos las opciones comerciales que encontrarás en muchos proyectos empresariales. Vamos a repasar nuestra visión general de las distintas categorías de herramientas. En la gestión de datos, la mayoría de los datos relevantes de una empresa se almacenan en una base de datos como lo son Oracle Database, Microsoft SQL Server, o IBM Db2. Aunque las bases de datos de código abierto están ganando popularidad, esos tres productos de gestión de datos se siguen considerando como los estándares de la industria. Lo que quiere decir que no desaparecerán en un futuro cercano. No se trata solamente de funcionalidad. Los datos son el corazón de cualquier organización y la disponibilidad de soportes comerciales juegan un papel muy importante. El soporte comercial es ofrecido directamente por los proveedores de software, los socios influyentes y las redes de soporte. Cuando nos enfocamos en las herramientas comerciales de integración de datos, hablamos de herramientas de “extracción, transformación y carga” o “ETL”. De acuerdo a Gartner Magic Quadrant (Cuadrante Mágico de Gartner), Informatica Powercenter e IBM InfoSphere DataStage son los líderes, seguidos por los productos de SAP, Oracle, SAS, Talend y Microsoft. Estas herramientas apoyan el diseño y despliegue de las tuberías de procesamiento de datos de ETL a través de una interfaz gráfica. También proporcionan conectores a la mayoría de los sistemas de información de destino comercial al igual que para los de código abierto. Finalmente, Watson Studio Desktop incluye un componente llamado Data Refinery, el cual permite La definición y ejecución de procesos de integración de datos en un estilo de hoja de cálculo. En el ámbito comercial, las visualizaciones de datos utilizan herramientas de inteligencia empresarial o “BI”. Su enfoque principal es crear informes visualmente atractivos y fáciles de comprender y tableros de control en vivo. Los ejemplos comerciales más importantes son: Tableau, Microsoft Power BI e IBM Cognos Analytics. Otro tipo de visualización se destina a los científicos de los datos en lugar de a los usuarios habituales. Un problema de muestra podría ser “¿Cómo pueden relacionarse las distintas columnas de una tabla?”. Este tipo de funcionalidad está incluida en Watson Studio Desktop. Si quieres construir un modelo de aprendizaje automático usando una herramienta comercial, deberías considerar el uso de un producto de minería de datos. Los más importantes de estos tipos de productos son: SPSS Modeler y SAS Enterprise Miner. Además, una versión de SPSS Modeler está igualmente disponible en Watson Studio Desktop, basada en la versión de la nube la de herramienta. Discutiremos más sobre las herramientas basadas en la nube en el próximo video. En el software comercial, el despliegue de modelos está estrechamente integrado en el proceso de construcción de modelos. Este diagrama muestra un ejemplo de SPSS Collaboration and Deployment Services en el cual son usados para desplegar cualquier tipo de activo creado por la suite de herramientas de Software SPSS. Otros proveedores usan el mismo tipo de proceso. Los software comerciales igualmente pueden exportar modelos en formato abierto. Ejemplo, SPSS Modeler permite la exportación de modelos como Predictive Model Markup Language, o PMML, el cual puede ser leído por numerosos paquetes de software comerciales y abiertos. Model monitoring es una nueva disciplina y actualmente no existen herramientas comerciales relevantes disponibles. Por lo tanto, el código abierto es la primera opción. Esto mismo pasa con la gestión de activos de código. El código abierto con Git y GitHub es el estándar efectivo. La gestión de activos de código, con frecuencia llamado gobernanza de datos o linaje de datos, es una parte fundamental de la ciencia de datos a nivel empresarial. Los datos deben ser versionados y anotados utilizando metadatos. Los proveedores, incluyendo Informatica Enterprise Data Governance e IBM, proporcionan herramientas para estas tareas específicas. IBM InfoSphere Information Governance Catalog cubre las funciones como el diccionario de datos, lo cual facilita el descubrimiento de los activos de datos. Cada activo de datos se asigna a un administrador de datos - - el propietario de los datos. El propietario de los datos es responsable de ese activo de datos y puede ser contactado. El linaje de datos también es cubierto; esto permite al usuario hacer un seguimiento de los pasos de transformación seguidos en la creación de los activos de datos. El linaje de datos también incluye una referencia a los daos de la fuente real. Se pueden agregar reglas y políticas para reflejar los complejos requisitos normativos y comerciales de privacidad y retención de datos. Watson Studio es un entorno de desarrollo completamente integrado para los científicos de datos. Es normalmente consumido a través de la nube, y cubriremos más sobre este en una siguiente lección. También hay una versión de escritorio disponible. Watson Studio Desktop combina Jupyter Notebooks con herramientas gráficas para maximizar el rendimiento de los científicos de datos. Watson Studio, conjuntamente con Watson Open Scale, es una herramienta completamente integrada que cubre el ciclo de vida completa de la ciencia de los datos y todas las tareas que hemos discutido previamente. Hablaremos más de ambos en la siguiente lección. Tan solo ten en cuenta que pueden ser desplegados en un centro de datos local que pueden ser Kubernetes o RedHat OpenShift. Otro ejemplo de una herramienta comercial totalmente integrada es H2O Driverless AI, la cual cubre el ciclo de vida completo de la ciencia de los datos. En esta lección, has aprendido cómo las tareas más comunes de la ciencia de los datos son soportadas por las herramientas comerciales. En el siguiente video, descubriremos herramientas de la ciencia de los datos que están disponibles exclusivamente en la nube.

# Herramientas basadas en la nube para la ciencia de los datos

Since we’ve previously covered open source tools for data science, let’s look at the commercial options you’ll find in many enterprise projects. Take another look at the overview of different tool categories. Since cloud products are a newer species, they follow the trend of having multiple tasks integrated in tools. This especially holds true for the tasks marked green in the diagram. Let’s start with the fully integrated visual tools category. Since these tools introduce a component where large scale execution of data science workflows happens in compute clusters, we’ve changed the title here and added the word “Platform.” These clusters are composed of multiple server machines, transparently for the user, in the background. Watson Studio, together with Watson OpenScale, covers the complete development life cycle for all data science, machine learning, and AI tasks. Another example is Microsoft Azure Machine Learning. This is also a fully cloud-hosted offering supporting the complete development life cycle of all data science, machine learning, and AI tasks. And finally, another example is H2O Driverless AI, which we’ve already introduced in the last video. Although it is a product that you download and install, one-click deployment is available for the common cloud service providers. Since operations and maintenance are not done by the cloud provider, as is the case with Watson Studio, Open Scale, and Azure Machine Learning, this delivery model should not be confused with Platform or Software as a Service -- PaaS or SaaS. In data management, with some exceptions, there are SaaS versions of existing open source and commercial tools. Remember, SaaS stands for “software as a service.” It means that the cloud provider operates the tool for you in the cloud. As an example, the cloud provider operates the product by backing up your data and configuration and installing updates. As mentioned, there is proprietary tooling, which is only available as a cloud product. Sometimes it’s only available from a single cloud provider. One example of such a service is Amazon Web Services DynamoDB, a NoSQL database that allows storage and retrieval of data in a key-value or a document store format. The most prominent document data structure is JSON (pronounced “jay-sun”). Another flavour of such a service is Cloudant, which is a database-as-a-service offering. But, under the hood it is based on the open source Apache CouchDB. It has an advantage: although complex operational tasks like updating, backup, restore, and scaling are done by the cloud provider, under the hood this offering is compatible with CouchDB. Therefore, the application can be migrated to another CouchDB server without changing the application. And IBM offers Db2 as a service as well. This is an example of a commercial database made available as a software-as-a-service offering in the cloud, taking operational tasks away from the user. When it comes to commercial data integration tools, we talk not only about “extract, transform, and load,” or “ETL” tools, but also about “extract, load, and transform,” or “ELT,” tools. This means the transformation steps are not done by a data integration team but are pushed towards the domain of the data scientist or data engineer. Two widely used commercial data integration tools are Informatica Cloud Data Integration and IBM’s Data Refinery. Data Refinery enables transformation of large amounts of raw data into consumable, quality information in a spreadsheet-like user interface. Data Refinery is part of IBM Watson Studio. The market for cloud data visualization tools is huge, and every major cloud vendor has one. An example of a smaller company’s cloud-based data visualization tool is DataMeer. IBM offers it’s famous Cognos Business intelligence suite as cloud solution as well. IBM Data Refinery also offers data exploration and visualization functionality in Watson Studio. Again, these are just some examples of a rapidly changing and growing commercial ecosystem among a huge number of established and emerging vendors. In Watson Studio, an abundance of different visualizations can be used to better understand data. For example, this 3D bar chart enables you to visualize a target value on the vertical dimension, which is dependent on two other values on the horizontal dimensions. Coloring enables you to visualize a third dimension. Hierarchical edge bundling enables you to visualize correlations and affiliations between entities. If sufficient, a classic bar chart can do the job as well, whereas a 2D scatter plot with a heat map shows two dependent data fields, one on the y axis and one as color intensity. A tree map shows distribution of subsets within a set, the famous pie chart does the same but in a non-hierarchical manner, and finally, a word cloud pops out significant terms in a document corpus. Model building can be done using a service such as Watson Machine Learning. Watson Machine Learning can train and build models using various open source libraries. Google has a similar service on their cloud called AI Platform Training. Nearly every cloud provider has a solution for this task. Model deployment in commercial software is usually tightly integrated to the model building process. Here is an example of the SPSS Collaboration and Deployment Services, which can be used to deploy any type of asset created by the SPSS software tools suite. The same holds for other vendors. In addition, commercial software can export models in an open format. As an example, SPSS Modeler supports exporting models as Predictive Model Markup Language, or “PMML,” which can be read by numerous other commercial and open software packages. Watson Machine Learning can also be used to deploy a model and make it available to consumers using a REST interface. Amazon SageMaker Model Monitor is an example of a cloud tool that continuously monitors deployed machine learning and deep learning models. Again, every major cloud provider has similar tooling. This is also the case for Watson OpenScale. OpenScale and Watson Studio… …unify the landscape. Everything marked in green can be done using Watson Studio and Watson OpenScale. We’ll cover Open Scale will be covered in a later video. You’ve learned how the most common tasks in data science are supported by commercial cloud tools. Integration provides us the ability to use the same tools for multiple tasks. In the next videos, we’ll look at packages, APIs, datasets, and models for data science.

# Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Correo electrónico Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Correo electrónico Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación Descripción generada automáticamenteLibrerías de la Ciencia de los Datos

Las librerías son una colección de funciones y métodos los cuales te permiten realizar una gran variedad de acciones sin tener que escribir el código tú mismo. Nos enfocaremos en la librería de Python: Librerías de Computación Científica en Python Librerías de Visualización en Python, Librerías de Aprendizaje Automático de Alto Nivel y Librerías de Aprendizaje Profundo – “Alto Nivel” sencillamente quiere decir que no tienes que preocuparte por los detalles, aunque esto dificulta el estudio o la mejora de las Librerías de Aprendizaje Profundo en Python Las librerías en otros lenguajes generalmente las librerías contienen módulos incorporados que proporcionan distintas funcionalidades que puedes usar directamente; algunas veces son llamados “frameworks o marcos de trabajo”. También hay muchas librerías, que ofrecen una amplia variedad de posibilidades. Pandas ofrece estructuras de datos y herramientas para la depuración, manipulación y análisis de datos efectivos. Esto proporciona herramientas para trabajar con los distintos tipos de datos. El instrumento principal de Pandas es una tabla de dos dimensiones que consiste en columnas y filas. Esta tabla es llamada “DataFrame” y está diseñada para proporcionar una fácil indexación para que puedas trabajar con tus datos. Las librerías NumPy están basadas en matrices, lo que te permite aplicar funciones matemáticas a estas matrices. Pandas está realmente construido basado en NumPy Los métodos de Visualización de Datos son una grandiosa forma de comunicarse con los demás y mostrar los resultados significativos del análisis. Estas librerías te permiten crear gráficos, tablas y mapas. El paquete Matplotlib es la librería más conocida para la visualización de datos y es excelente para hacer gráficos y diagramas. Igualmente, los gráficos son altamente personalizables. Otra librería de visualización de alto nivel, es Seaborn, el cual está basado en matplotlib. Seaborn hace fácil la generación de gráficos como mapas de calor, series temporales y gráficos de violín. Para el aprendizaje automático, la librería Scikit-learn contiene herramientas para el modelado estadístico, incluyendo regresión, clasificación, clustering o agrupación entre otros. Está construida basada en NumPy, SciPy y matplotlib y es prácticamente para comenzar. Para este enfoque de alto nivel, tu defines el modelo y especificas los tipos de parámetros que te gustaría usar. Para el aprendizaje profundo, Keras te permite construir el modelo estándar de aprendizaje profundo. Así como Scikit-learn, la interfaz de alto nivel te permite construir modelos de forma rápida y simple. Esto puede funcionar usando las unidades de procesamiento de gráficos (GPU), pero para muchos casos de aprendizaje profundo un entorno de nivel menor. TensorFlow es un marco de nivel bajo usado en la producción a gran escala de modelos de aprendizaje profundo. Está diseñado para la producción, pero puede ser difícil de manejar para la experimentación. Pytorch es usado para la experimentación, haciéndolo que sea simple para los investigadores probar sus ideas. Apache Spark es un marco de computación de clúster de propósito general el cual permite el procesamiento de datos usando clústeres de computación. Esto quiere decir que procesas los datos de forma paralela, utilizando múltiples computadoras simultáneamente. La librería Spark una funcionalidad muy similar a la de Pandas Numpy Scikit-learn Apache Spark data processing jobs puede usar Python R Scala o SQL Hay muchas librerías para Scala, la cual se usa predominantemente en la ingeniería de datos pero de igual forma se usa a veces en la ciencia de los datos. Vamos a discutir algunas de las librerías que son complementarias de Spark Vegas es una librería Scala para la visualización de datos estadísticos. Con Vegas, puedes trabajar con los archivos de datos, así como Spark DataFrames. Para el aprendizaje profundo, puedes usar BigDL. R tiene una funcionalidad incorporada para el aprendizaje automático y la visualización de datos, pero también existen diferentes librerías complementarias: ggplot2 es una librería popular para la visualización de datos en R. Igualmente puedes utilizar librerías que te permiten hacer la interfaz con Keras y TensorFlow. R ha sido el estándar de-facto para la ciencia de datos de código abierto, pero ahora está siendo sustituido por Python.

# Interfaces de Programación de Aplicaciones

En este vídeo discutiremos acerca de las Interfaces de Programación de Aplicaciones o también conocidas como APIs. Específicamente, discutiremos: Qué es una API Librerías API – API REST, incluyendo: Solicitud y Respuesta Una API permite que dos elementos de un software se hablen entre sí. Por ejemplo, tienes tu programa, tienes tus datos, tienes otros componentes de software. Usas la API para comunicarte con otros componentes de software. Usas la API para comunicarte con la API usando las entradas y salidas. No tienes que saber cómo funciona la API, solamente necesitas saber sus entradas y salidas. Recuerda, que la API sólo se refiere a la interfaz, o a la parte de la librería que tú ves. La “Librería” se refiere a todo el tema. Considera la librería pandas. Pandas es en realidad un conjunto de componentes de software, muchos de los cuales ni siquiera están escritos en Python. Tienes algunos datos. Tienes un conjunto de componentes de software. Nosotros usamos la Pandas API para procesar los datos comunicándonos con los otros componentes del software. Puede haber tan solo un componente de software en el back end, pero puede estar separado La API para diferentes lenguajes. Considera TensorFlow, escrito en C++. Hay APIs separadas en Python JavaScript C++ Java Go. La API es sencillamente la interfaz. También hay múltiples APIs desarrolladas por voluntarios para TensorFlow; como por ejemplo Julia MATLAB R Scala y muchas más. Las APIs REST son otro tipo de API muy popular. Estas te permiten comunicarte usando la Internet, aprovechando el almacenamiento y mayor acceso a los datos, algoritmos de inteligencia artificial y muchos otros recursos. RE significa “Representación” la S significa “Estado” y la T Significa “Transferencia”. En la API REST, tu programa es llamado el “cliente”. La API se comunica con un servicio web el cual llamas a través de Internet. Hay un conjunto de reglas que rigen (clic 4)Comunicación, (clic 5)Entrada o Solicitud, y (clic 6 ) Salida O Respuesta. Aquí hay algunos términos comunes relacionados con la API. Tú o tú código pueden ser vistos como un cliente. El servicio web se refiere a un recurso. El cliente encuentra el servicio a través de un punto final. El cliente envía la solicitud al recurso y la respuesta al cliente. Los métodos HTTP son una forma de transmitir datos a través de Internet le decimos a las APIs REST qué hacer enviando una solicitud. La solicitud es usualmente comunicada a través de un mensaje HTTP. Generalmente el mensaje HTTP contiene un archivo JSON, que contiene las instrucciones para la operación que nos gustaría que el servicio realizara. Esta operación se transmite al servicio web a través de Internet. El servicio realiza la operación. Similarmente, el servicio web devuelve una respuesta a través de un mensaje HTTP, donde la información se devuelve generalmente usando un archivo JSON. Esta información es transmitida de vuelta al cliente. La API Watson Text to Speech es un ejemplo de una API REST. Esta API convierte la voz a texto. En las API de llamadas, se envía una copia del archivo de audio a la API; este proceso es llamado solicitud de envío. La API envía la transcripción de texto de lo que el individuo está diciendo. La API está haciendo una solicitud de obtención. La API Watson Language-Translator provee otro ejemplo. Envías el texto que quieres traducir a la API, y ésta traduce el texto y te envía la traducción. En este caso traducimos de inglés a español. En este video, hemos discutido lo que es una API Librerías API – API REST, incluyendo Solicitud y Respuesta Gracias por ver este video.

# Conjuntos de datos - Mejorando la ciencia de los datos

En este video discutiremos los conjuntos de datos: qué son, por qué son tan importantes en la ciencia de los datos, y donde encontrarlos. Primero definamos brevemente lo que es un conjunto de datos. Un conjunto de datos es una colección estructurada de datos. Los datos incluyen información que podría ser representada como texto, números o archivos multimedia como imágenes, audio o video. Un conjunto de datos que es estructurado como datos tabulares comprende una colección de filas, que a su vez Comprenden columnas que almacenan la información. Un formato popular de datos tabulados es el de “valores separados por comas” o CSV. Un archivo de tipo CSV es un archivo de texto delimitado en el que cada línea representa una fila y los valores de los datos están separados por una coma. Por ejemplo, imagina un conjunto de datos de observaciones de una estación meteorológica. Cada fila representa una observación en un momento determinado, mientras que cada columna contiene la información sobre esa observación en particular, como la temperatura, humedad y cualquier otra condición meteorológica. Las estructuras de datos jerárquicas o de red son normalmente usadas para representar las relaciones entre los datos. Los datos jerárquicos son organizados en una estructura de árbol, mientras que los datos de red pueden ser almacenados de forma gráfica. Por ejemplo, las conexiones entre personas en un sitio web de redes sociales normalmente son representadas en forma de gráfico. Un conjunto de datos también puede incluir archivos de datos sin procesar, como lo son las imágenes o audios. El conjunto de datos MNIST es muy popular para la ciencia de los datos. Contiene imagines de dígitos escritos a mano y es comúnmente usado para entrenar los sistemas de procesamiento de imágenes. Tradicionalmente, la mayoría de los conjuntos de datos eran considerados privados porque contenían información patentada o confidencial, como por ejemplo datos de clientes, datos de precios o cualquier otra información comercialmente sensible. Estos conjuntos de datos no son normalmente compartidos públicamente. Con el tiempo, más y más entidades públicas y privadas como instituciones científicas, gobiernos, organizaciones e incluso empresas han comenzado a poner a disposición del público conjuntos de datos como “datos abiertos”, que permiten obtener una gran cantidad de información de manera gratuita. Por ejemplo, las Naciones Unidas al igual que gobiernos federales y municipales alrededor del mundo han publicado muchos conjuntos de datos en sus sitios web, que incluyen economía, sociedad, asistencia media, transporte, medio ambiente y mucho más. El acceso a estos y otros conjuntos de datos abiertos permite a los científicos de datos, investigadores, analistas y otras personas descubrir conocimientos anteriormente desconocidos, pero potencialmente útiles. Ellos pueden crear nuevas aplicaciones bien sea para fines comerciales como para el bien público. De igual forma pueden llevar a cabos nuevas investigaciones. Los datos abiertos han jugado un papel muy importante en el crecimiento de la ciencia de los datos, el aprendizaje automático y la inteligencia artificial y han proporcionado a los profesionales una forma de perfeccionar sus habilidades sobre una amplia variedad de conjuntos de datos. Existen muchas fuentes de datos abiertos en Internet. Puedes encontrar una lista extensa de portales de datos abiertos en todo el mundo en el sitio web de la Fundación de Conocimiento Abierto (FCA) datacatalogs.org. Las Naciones Unidas, la Unión Europea y muchas otras organizaciones gubernamentales e intergubernamentales mantienen repositorios de datos los cuales dan acceso a una amplia variedad de información. En Kaggle, la cual es una popular comunidad en línea de ciencia de los datos, podrás encontrar y contribuir con conjuntos de datos que podrían ser de interés general. Por último, pero no menos importante, Google proporciona un motor de búsqueda de conjuntos de datos que podrían ayudarte a encontrar los que tengan un valor particular para ti. Es importante reconocer que la distribución y el uso de datos abiertos podrían estar restringidos, según lo definan las condiciones de su licencia. En caso de la falta de una licencia para la distribución de datos abiertos, muchos conjuntos de datos eran compartidos en el pasado bajo licencias de software de código abierto. Estas licencias no fueron diseñadas para cubrir las consideraciones especificas relacionadas con la distribución y uso de los conjuntos de datos. Para tratar el tema, Linux Foundation creó el Community Data License Agreement, o CDLA. Dos licencias fueron inicialmente creadas para compartir datos: CDLA-Sharing y CDLA-Permissive. La licencia CDLA-Sharing te otorga permiso para utilizar y modificar los datos. La licencia estipula que si publicas tu versión modificada de los datos debes hacerlo bajo los mismos términos de la licencia de los datos originales. La licencia CDLA-Permissive igualmente te otorga permiso para utilizar y modificar los datos. Sin embargo, no estás obligado a compartir los cambios que le hagas a los datos. Debes tener en cuenta que ninguna de las dos licencias impone restricciones en los resultados que podría conseguir al usar los datos, lo cual es importante en la ciencia de los datos. Digamos, por ejemplo, que estas construyendo un modelo el cual realiza una predicción. Si estás capacitando a el modelo usando conjuntos de datos bajo la licencia de CDLA, no estás obligado a compartir el modelo, o a compartirlo bajo una licencia especifica si decides compartirlo. En este video has aprendido acerca de los conjuntos de datos abiertos, su rol en la ciencia de los datos y donde puedes encontrarlos. También hemos visto algo de Community Data License Agreement, el cual facilita el intercambio y como compartir los datos abiertos. Un aspecto importante que no hemos cubierto en este video es la calidad y precisión de los datos, los cuales podrían variar mucho dependiendo de quien haya recolectado y contribuido al conjunto de datos. Mientras que algunos conjuntos de datos abiertos pueden ser lo suficientemente buenos para uso personal, es posible que no cumplan los requisitos de la empresa debido al impacto que pueden tener en el negocio. En el próximo modulo, aprenderás acerca de Data Asset eXchange, el cual es un repositorio de datos abierto y curado.

# Data Asset eXchange

A pesar del crecimiento de los conjuntos de datos abiertos que están disponibles al público, aun así puede ser difícil descubrir conjuntos de datos que sean a la vez de alta calidad y que tengan términos de licencia y uso claramente definidos. Para ayudar a resolver este reto, IBM creó Data Asset eXchange, o “DAX”, el cual que mostraremos en este vídeo. DAX proporciona una fuente de confianza para encontrar conjuntos de datos abiertos que están listos para ser usados en aplicaciones empresariales. Estos conjuntos de datos los cuales cubren una amplia variedad de dominios, incluyendo imagines, video, texto y audio. Debido a que DAX proporciona un alto nivel de conservación de la calidad de los conjuntos de datos, así como las licencias y condiciones de uso, los conjuntos de datos DAX usualmente son fáciles de adoptar, bien sea en proyectos de investigación o comerciales. Siempre que sea posible, DAX apunta a poner a disposición los conjuntos de datos en el ámbito de una de las variantes de CDLA Community Data License Agreement, con el fin de fomentar el intercambio de datos y la colaboración. DAX también provee un lugar único para acceder a los conjuntos de datos únicos, en particular de los proyectos de IBM Research. Para facilitar a los desarrolladores el inicio en el uso de los conjuntos de datos, DAX también provee tutoriales notebooks que explican los fundamentos de la limpieza de datos, pre-procesamiento y el análisis exploratorio. Para algunos conjuntos de datos, también hay notebooks que ilustran cómo realizar análisis más complejos, como la creación de gráficos, análisis estadísticos, análisis de series cronológico, capacitación en los modelos de aprendizaje automático y la integración de aprendizaje profundo a través del uso de Model Asset eXchange, (un proyecto muy relacionado con DAX y también está disponible en el sitio web de IBM Developer). De esta forma, DAX ayuda a los desarrolladores a crear flujos de trabajo analíticos y de aprendizaje automático de principio a fin y a consumir datos y modelos abiertos con seguridad bajo los términos de licencia claramente definidos. Digamos que has encontrado un conjunto de datos que podrían ser de tu interés. En la página de conjunto de datos podrás descargar el archivo comprimido de los conjuntos de los datos de almacenamiento en la nube, explore el conjunto de los datos usando Jupyter Notebooks, revisar los metadatos del conjunto de datos, así como el formato, términos de licencia y el tamaño y una vista previa de algunas partes de los conjuntos de datos. La mayoría de los conjuntos de datos en DAX son complementados con uno o más Jupyter Notebooks que puedes usar para realizar la limpieza de datos, pre-procesamiento y el análisis exploratorio. Estos notebooks funcionan “tal cual” como en Watson Studio, la plataforma de ciencia Data Sciencedata de IBM. Jupyter Notebooks y Watson Studio serán discutidos más adelante durante este curso. En este video, has aprendido sobre el repositorio de datos abiertos de IBM y acerca de Data Asset eXchange. En el laboratorio practico tendrás la oportunidad de explorar el repositorio.

# Modelos de Aprendizaje Automático

En este video, te presentaremos los modelos de aprendizaje automático al igual que aprendizaje profundo. Los datos contienen una enorme cantidad de información que puede ser usada para resolver ciertos tipos de problemas. El alcance tradicional del análisis de datos, como una persona que inspecciona manualmente los datos o un programa computarizado especializado que automatice el análisis de los humanos, los cuales llegan rápidamente a sus limites debido a la cantidad de datos para ser analizados o a la complejidad del problema. El aprendizaje automático usa algoritmos – también conocidos como “modelos” – para identificar los patrones en los datos. El proceso por el cual el modelo aprende estos patrones a partir de los datos es llamado “capacitación del modelo”. Una vez que un modelo es capacitado, entonces puede ser usado para hacer predicciones. Una vez que se le presentan los nuevos datos al modelo, intenta hacer predicciones o decisiones basadas en los patrones que ha aprendido de los datos anteriores. Los modelos de aprendizaje automático pueden ser divididos en tres clases básicas: aprendizaje supervisado, aprendizaje no supervisado y aprendizaje reforzado. El aprendizaje supervisado es uno de los modelos de aprendizaje automático mas comúnmente usados. En el aprendizaje supervisado, un humano provee los datos de entrada y las salidas correctas. El modelo intenta identificar las relaciones y dependencias entre los datos de entrada y la salida correcta. En términos generales, el aprendizaje supervisado es usado para resolver los problemas de regresión y clasificación. Ahora vemos un ejemplo para cada tipo de problema: Los modelos de regresión son usados para predecir un valor numérico “real”. Por ejemplo, dada la información acerca de las ventas anteriores de casas, como lo son la ubicación geográfica, tamaño, número de cuartos y precio de venta, puedes capacitar un modelo para predecir el precio de venta estimado para otras casas con características muy similares. Los modelos de clasificación son usados para predecir si algo pertenece a una categoría o “clase” Por ejemplo, dado un conjunto de correos electrónicos con una designación de si son considerados o no como spam, un algoritmo puede ser capacitado para identificar los correos electrónicos no solicitados. En el aprendizaje no supervisado, los datos no son etiquetados por un humano. Los modelos deben analizar los datos e intentar identificar los patrones y la estructura dentro de los datos basándose solamente en las características de los propios datos. La agrupación y detección de anomalías son dos ejemplos claros de este tipo de aprendizaje. Los modelos de agrupación son usados para dividir cada registro de un conjunto de datos en uno de un pequeño numero de grupos similares. Un ejemplo de un modelo de agrupación podría ser proporcionar recomendaciones de compra para una tienda e-commerce basada en el comportamiento de las compras anteriores y el contenido de la cesta de compra. La detección de anomalías identifica los valores atípicos en un conjunto de datos, tales como, transacciones fraudulentas con tarjetas de crédito o intentos sospechosos de inicios de sesión en línea. El tercer tipo de aprendizaje, el cual es aprendizaje reforzado, esta vagamente basado en la forma en que los seres humanos y otros organismos aprenden. Piensa en un ratón en un laberinto. Si el ratón llega al final del laberinto, él obtiene un trozo de queso. Esta es la “recompensa” por haber terminado la tarea. El ratón aprende – por ensayo y error – cómo recorrer el laberinto para conseguir todo el queso que pueda. De forma similar, un modelo de aprendizaje reforzado aprende el mejor conjunto de acciones que debe tomar, dado su entorno actual, para obtener la mayor recompensa a lo largo del tiempo. Este tipo de aprendizaje ha sido muy exitoso recientemente al vencer a los mejores jugadores humanos en juegos como go, ajedrez y los videojuegos de estrategia más populares. El aprendizaje profundo es un tipo especializado de aprendizaje automático. Lo que se refiere a un conjunto general de modelos y técnicas que tratan vagamente de emular la manera en que el cerebro humano resuelve una amplia variedad de problemas. Es comúnmente usado para analizar el lenguaje natural, bien sea hablado y texto, así como las imágenes, audio y video para pronosticar los datos de series temporales y mucho más. El aprendizaje profundo ha tenido demasiado éxito recientemente en estas y otras áreas, por lo tanto se está convirtiendo en una herramienta cada vez más popular e importante para la ciencia de los datos. El aprendizaje profundo normalmente suele requerir conjuntos de datos muy grandes etiquetados para capacitar a un modelo, es intensivo en computación y usualmente requiere un hardware de propósito especial para lograr tiempos de capacitación aceptables. Puedes construir un modelo personalizado de aprendizaje profundo desde cero o también usar modelos pre-capacitados de los repositorios de modelos públicos. Los modelos de aprendizaje profundo son implementados usando marcos de trabajos populares como lo son TensorFlow, PyTorch, y Keras. Los marcos de trabajo de aprendizaje profundo generalmente proporcionan una API de Python API y la mayoría son compatibles con otros lenguajes de programación como C++ y JavaScript. Puedes descargar modelos de última generación pre-capacitados de los repositorios que son comúnmente referidos como modelos “zoos” Los modelos zoos más populares incluyen los proveídos por TensorFlow, PyTorch, Keras y ONNX. Los modelos igualmente son publicados por grupos de investigación académica y comercial. Mientras está fuera del alcance de este video explicar en detalle cómo podrías construir un modelo, vamos a resumir brevemente las tareas de alto nivel usando un ejemplo. Vamos a suponer que quieres habilitar una aplicación para identificar objetos en imágenes capacitando un modelo de aprendizaje profundo. Primero, recoges y preparas los datos que serán usados para capacitar el modelo. La preparación de los datos puede ser un proceso que consume mucho tiempo y trabajo. Para capacitar un modelo para detectar objetos en imágenes, necesitas etiquetar los datos brutos de capacitación, ejemplo, dibujando cajas limitadoras alrededor de los objetos y etiquetándolos. Después, construirás un modelo desde cero o seleccionarás un modelo existente que podría ser adecuado para la tarea bien sea desde un recurso público o privado. Entonces entrenas el modelo con los datos que has preparado. Durante la capacitación, tu modelo aprende a partir de los datos etiquetados, cómo identificar los objetos que son representados en una imagen. Una vez que has comenzado la capacitación, analizas los resultados y repites el proceso hasta que el desempeño del modelo entrenado cumpla con tus requisitos. Cuando el modelo entrenado se comporta como se desea, lo despliegas para ponerlo a disposición de tus aplicaciones. En este video, has aprendido acerca del aprendizaje automático al igual que el aprendizaje profundo, para que son usados y dónde encontrar los modelos de código abierto. En el siguiente video, te presentaremos el Model Asset eXchange, una colección curada de modelos de aprendizaje profundo listos para ser usados, así como también personalizables.

# El Model Asset Exchange

En este video, te presentaremos el Model Asset eXchange en IBM Developer, un recurso gratuito de código abierto para modelos de aprendizaje profundo. Durante este video nos referiremos a Model Asset eXchange como “MAX” En el video previo, describimos resumidamente las tareas de alto nivel que hay que completar para capacitar a un modelo desde cero. Debido al volumen de datos, trabajo, tiempo y recursos requeridos para completar las tareas, el tiempo de valoración puede ser bastante largo. Para reducir el tiempo de valoración, considera la posibilidad de tomar ventaja de los modelos pre-capacitados para ciertos tipos de problemas. Estos modelos pre-capacitados pueden estar listos para ser utilizados inmediatamente o pueden tomar menos tiempo para ser capacitados. El Model Asset eXchange es un repositorio gratuito de código abierto para microservicios de aprendizaje profundo listos para ser usados y personalizables. Estos microservicios están configurados para usar los modelos de aprendizaje profundo de última generación, pre-capacitados o personalizables, para resolver los problemas empresariales más comunes. Estos modelos han sido revisados, probados y pueden ser rápidamente desplegados en entornos locales y en la nube. Todos los modelos en MAX están disponibles bajo las licencias permisivas de código abierto, haciendo más fácil el uso para propósito personales, empresariales y reduce el riesgo de responsabilidades legales. En MAX, puedes encontrar los modelos para una variedad de dominios, incluyendo imagen, audio, video y análisis del lenguaje natural. Esta lista incluye una pequeña selección. En el laboratorio para este módulo, tendrás la oportunidad de explorar esos modelos. Echemos un vistazo a los componentes de un típico microservicio de modelos. Cada microservicio incluye los siguientes componentes: Un modelo de aprendizaje profundo pre-capacitado. Código que pre-procesa las entradas antes de ser analizada por el modelo y código que post-procesa la salida del modelo. Una API publica estandarizada que permite a las aplicaciones acceder a la funcionalidad de los servicios. Los microservicios de modelos de MAX son construidos y distribuidos como imágenes de Docker de código abierto. Docker es una plataforma de contenedores que hace mas fácil la construcción de aplicaciones y su despliegue en un entorno de desarrollo, prueba o producción. La fuente de imágenes Docker es publicada en GitHub y puede ser descargada, personalizada según sea necesario y utilizada en entornos personales o comerciales. Puedes desplegar y ejecutar estas imágenes en un entorno de prueba o producción usando Kubernetes, un sistema de código abierto para automatizar el despliegue, escalado y la gestión de aplicaciones en contenedores en nubes privadas, hibridas o públicas. Una plataforma popular Kubernetes a nivel empresarial es Red Hat OpenShift la cual está disponible en IBM Cloud, Google Cloud Platform, Amazon Web Services y Microsoft Azure. Los microservicios de modelos exponen una API REST que los desarrolladores pueden usar para incorporar el aprendizaje profundo en sus aplicaciones y servicios. Debido a que los API REST pueden ser consumidos usando cualquier lenguaje de programación, fácilmente puedes integrar estos servicios en un ecosistema existente. La API expone un punto final o “endpoint” de predicción y unos o más puntos finales de metadatos. Este ejemplo muestra los puntos finales del Microservicio de Detección de Objetos. El punto final del modelo/predicción tiene una imagen como entrada y regresa como respuesta una lista de objetos que fueron detectados en la imagen, junto con las coordenadas del cuadro delimitador que identifica donde el objeto detectado está ubicado. Algunos puntos finales de predicción igualmente pueden aceptar los parámetros de entrada adicionales que impactan en los resultados producidos, tales como los filtros. Este microservicio expone dos puntos finales de metadatos, /modelo/etiquetas y /modelo/metadatos. Estos puntos finales proporcionan información como los objetos que pueden ser detectados y el modelo de aprendizaje profundo usado para derivar las respuestas dada la entrada. En la parte del laboratorio para este módulo, tendrás la oportunidad de explorar y probar estos puntos finales utilizando un navegador web. Cada punto final acepta entradas de amigable aplicación, como una imagen en formato JPG, PNG o GIF en vez de una estructura de datos especifica del modelo. Cada punto final también genera salidas de amigable aplicación, como el JSON estandarizado, el cual es un formato poco pesado de intercambio de datos. Vamos a echar un vistazo más de cerca a lo que pasa cuando una aplicación invoca el punto final de la predicción. En este ejemplo, un usuario ha seleccionado una imagen en una aplicación web, el punto final de la predicción es invocado y la imagen es cargada. El microservicio prepara la imagen de entrada para el procesamiento, ejecuta el modelo de aprendizaje profundo que identifica los objetos de la imagen, genera una respuesta usando los resultados de predicción y regresa el resultado a la aplicación. La aplicación muestra los resultados dibujando cajas y etiquetas delimitadoras. En este video, hemos presentado el Model Asset eXchange, que es un repositorio gratuito y de código abierto para microservicios que pone a disposición de las aplicaciones y servicios en entornos locales, la nube y la funcionalidad de aprendizaje profundo. En el laboratorio, tendrás la oportunidad de probar un modelo de microservicio, explorar su API y aprender más acerca de cómo puedes aprovecharlo desde una aplicación web y una aplicación de Internet de las Cosas (IoT).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente