Fase 0 selección de objetivo a solventar

- ¿Qué problema quieres resolver?
- ¿Cómo puedes ayudar a tu empresa a generar conocimiento?
- ¿Qué proceso manual se esta realizando que se podría realizar de manera automática a través de los datos?

Una vez definido el objetivo

- ¿Dónde saco los datos?
- ¿Es suficiente con una fuente de datos?
- Analizar si los datos iniciales se encuentran procesados de alguna manera

Fase 1 análisis de los datos

• ¿Cuántos registros hay?

– ¿Son demasiado pocos?

– ¿Son muchos y no tenemos Capacidad (CPU+RAM) suficiente para procesarlo?

– ¿Están todas las filas completas ó tenemos campos con valores nulos?

– En caso que haya demasiados nulos: ¿Queda el resto de información inútil?

• ¿Que datos son discretos y cuáles continuos?

• Muchas veces sirve obtener el tipo de datos: texto, int, double, float

• Si es un problema de tipo supervisado:

– ¿Cuál es la columna de “salida”? ¿binaria, multiclase?

– ¿Esta balanceado el conjunto salida?

• ¿Cuáles parecen ser features importantes? ¿Cuáles podemos descartar? ¿Por qué crees que tienes que descartar estas características?

- ¿Puedo transformar alguna variable para generar información adicional?
- ¿Conozco conocimiento experto para incorporarlo en el dataset? ¿Cómo lo puedo modelar?

• ¿Siguen alguna distribución?

• ¿Hay correlación entre features (características)?

• En problemas de NLP es frecuente que existan categorías repetidas ó mal tipeadas, ó con mayúsculas/minúsculas, singular y plural, por ejemplo “Abogado” y “Abogadas”, “avogado” pertenecerían todos a un mismo conjunto.

• ¿Estamos ante un problema dependiente del tiempo? Es decir, un TimeSeries[17].

• Si fuera un problema de Visión Artificial[18]: ¿Tenemos suficientes muestras de cada clase y variedad, para poder hacer generalizar un modelo de Machine Learning?

• ¿Cuáles son los Outliers? (unos pocos datos aislados que difieren drásticamente del resto y "contaminan" ó desvían las distribuciones) – Podemos eliminarlos? ¿es importante conservarlos? – son errores de carga o son reales?

• ¿Tenemos posible sesgo de datos? (por ejemplo, perjudicar a clases minoritarias por no incluirlas y que el modelo de ML discrimine)

Diferencia de objetivos

Predicción tiene que ver con la estimación, y la extrapolación en ámbitos continuos. Por ejemplo, en el tiempo o los habitantes en el censo, don

El **pronóstico** consiste en anticiparse a un hecho puntual en base a un conjunto pequeño de alternativas, por ejemplo, en una quiniela de fútbol pone "marque con una X su pronóstico" y las alternativas son 1-X-2, o un terremoto se debe pronosticar en base a las alternativas SI-NO. Los sistemas para abordar una u otra opción son claramente distintos. Los primeros se basan en la extrapolación y los segundos en una serie de características para anticiparse al hecho puntual (por ejemplo, en las quinielas saber si alguno de los equipos se juega algo importante como el descenso si hay un jugador importante lesionado o quién va a ser el árbitro del partido).

El **análisis prescriptivo**, por su parte, nos ayuda a utilizar toda la información anterior, proporcionada por las otras disciplinas de la analítica para dirigir y automatizar la toma de decisiones mediante la aplicación de modelos de inferencia y optimización matemática.

La segmentación con técnicas estadísticas es muy usada en diversos problemas. En marketing son útiles los modelos estadísticos para segmentar o dividir poblaciones en grupos distintos. Esto permite realizar campañas diferentes a cada uno de los grupos. Los modelos estadísticos de segmentación también son empleados en procesamiento de imágenes así como en algoritmos de compresión de imágenes. En general, los algoritmos de segmentación agrupan datos similares usando una serie de variables. Los métodos estadísticos más empleados para segmentar, el algoritmo k-means o k-medias.

El algoritmo k-means es un método de agrupamiento que divide un conjunto de n observaciones en k grupos distintos gracias a valores medios. Pertenece al ámbito de los algoritmos no supervisados, ya que las n observaciones no cuentan con una etiqueta que nos diga de qué grupo es cada dato, siendo los datos agrupados según sus propiedades o características.

Algoritmos:

- La regresión lineal es un algoritmo de aprendizaje supervisado que se utiliza en Machine Learning y en estadística. En su versión más sencilla, lo que haremos es “dibujar una recta” que nos indicará la tendencia de un conjunto de datos continuos (si fueran discretos, utilizaríamos Regresión Logística).

- Maquinas de Vector de Soporte: (Support Vector Machines, SVMs) son un conjunto de algoritmos de aprendizaje supervisado desarrollados por Vladimir Vapnik y su equipo en los laboratorios AT&T. Una SVM construye un hiperplano o conjunto de hiperplanos en un espacio de dimensionalidad muy alta (o incluso infinita) que puede ser utilizado en problemas de clasificación o regresión. Una buena separación entre las clases permitirá una clasificación correcta.

- Regresión Logística A partir de un conjunto de datos de entrada (características), nuestra salida será discreta (y no continua) por eso utilizamos Regresión Logística (y no Regresión Lineal). La Regresión Logística es un Algoritmo Supervisado y se utiliza para clasificación Vamos a clasificar problemas con dos posibles estados “SI/NO”: binario o un número finito de “etiquetas” o “clases”: múltiple.
- Los árboles de decisión son representaciones gráficas de posibles soluciones a una decisión basadas en ciertas condiciones, es uno de los algoritmos de aprendizaje supervisado más utilizados en machine learning y pueden realizar tareas de clasificación o regresión (acrónimo del inglés CART)
- KNN es un método de clasificación supervisada (Aprendizaje, estimación basada en un conjunto de entrenamiento y prototipos) que sirve para estimar la función de densidad de las predictoras por cada clase .
- ZeroR es el metodo de clasificación más simple que existe y depende solo en el target ignorando todos los predictores. El clasificador ZeroR simplemente predice sobre la clase o categoría principal (majority category). Aunque ZeroR tenga cero poder predictivo, es útil para determinar una base de performance sobre la cual medir los demás métodos de clasificación.
- Red de neuronas se pueden usar en aprendizaje supervisado:
    - En el aprendizaje supervisado, se nos da una serie de ejemplos de pares y el objetivo es encontrar una función en la clase permitido de funciones que corresponden con los ejemplos. La función de coste está relacionada con la falta de coincidencia entre nuestro mapeo y los datos, y contiene implícitamente el conocimiento previo sobre el dominio del problema.
    - Aprendizaje no supervisado: algunos datos se dan y la función de coste que se reduce al mínimo, que puede ser cualquier función de los datos y la salida de la red. La función de coste depende de la tarea (lo que estamos tratando de modelar) y nuestras a priori suposiciones implícitas (las propiedades de nuestro modelo, sus parámetros y las variables observadas).
    - Aprendizaje por refuerzo, los datos por lo general no se dan, pero generada por la interacción de un agente con el medio ambiente. En cada punto en el tiempo , El agente realiza una acción y el medio ambiente genera una observación y un costo instantáneo , De acuerdo con algunas dinámicas (por lo general desconocidos). El objetivo es descubrir una *política* para la selección de las acciones que minimiza una cierta medida de un costo a largo plazo, por ejemplo, el coste acumulativo esperado. La dinámica del medio ambiente y el coste a largo plazo para cada política general son desconocidos, pero pueden ser estimados.
- Naive Bayes: son una familia de simples "clasificadores probabilísticos" basados en la aplicación del teorema de Bayes entre las características. Se encuentran entre los modelos de red bayesianos más simples, pero junto con la estimación de la densidad del núcleo, pueden alcanzar niveles de precisión más altos.

Overfitting en Machine Learning

Es muy común que al comenzar a aprender machine learning caigamos en el problema del Overfitting. Lo que ocurrirá es que nuestra máquina sólo se ajustará a aprender los casos particulares que le enseñamos y será incapaz de reconocer nuevos datos de entrada. En nuestro conjunto de datos de entrada muchas veces introducimos muestras atípicas (ó anomalas) o con "ruido/distorción" en alguna de sus dimensiones, o muestras que pueden no ser del todo representativas. Cuando "sobreentrenamos" nuestro modelo y caemos en el overfitting, nuestro algoritmo estará considerando como válidos sólo los datos idénticos a los de nuestro conjunto de entrenamiento -incluidos sus defectos y siendo incapaz de distinguir entradas buenas como fiables si se salen un poco de los rangos ya prestablecidos

Prevenir el Sobreajuste de datos

Para intentar que estos problemas nos afecten lo menos posible, podemos llevar a cabo diversas acciones.

• Cantidad mínima de muestras tanto para entrenar el modelo como para validarlo.

• Clases variadas y equilibradas en cantidad: En caso de aprendizaje supervisado y suponiendo que tenemos que clasificar diversas clases o categorías, es importante que los datos de entrenamiento estén balanceados.

Supongamos que tenemos que diferenciar entre manzanas, peras y bananas, debemos tener muchas fotos de las 3 frutas y en cantidades similares. Si tenemos muy pocas fotos de peras, esto afectará en el aprendizaje de nuestro algoritmo para identificar esa fruta.

• Conjunto de Test de datos. Siempre subdividir nuestro conjunto de datos y mantener una porción del mismo "oculto" a nuestra máquina entrenada. Esto nos permitirá obtener una valoración de aciertos/fallos real del modelo y también nos permitirá detectar fácilmente efectos del overfitting /underfitting.

• Ajuste de Parámetros: deberemos experimentar sobre todo dando más/menos "tiempo/iteraciones" al entrenamiento y su aprendizaje hasta encontrar el equilibrio

• Cantidad excesiva de Dimensiones (features), con muchas variantes distintas, sin suficientes muestras. A veces conviene eliminar o reducir la cantidad de características que utilizaremos para entrenar el modelo. Una herramienta útil para hacerlo es PCA

• Quiero notar que si nuestro modelo es una red neuronal artificial -deep learning-, podemos caer en overfitting si usamos capas ocultas en exceso, ya que haríamos que el modelo memorice las posibles salidas, en vez de ser flexible y adecuar las activaciones a las entradas nuevas.

Problemas de clasificación con Clases desequilibradas

En los problemas de clasificación en donde tenemos que etiquetar por ejemplo entre "spam" o "not spam" ó entre múltiples categorías (coche, barco, avión) solemos encontrar que en nuestro conjunto de datos de entrenamiento contamos con que alguna de las clases de muestra es una clase "minoritaria" es decir,

de la cual tenemos muy poquitas muestras. Esto provoca un desbalanceo en los datos que utilizaremos para el entrenamiento de nuestra máquina.

¿Cómo nos afectan los datos desbalanceados? Por lo general afecta a los algoritmos en su proceso de generalización de la información y perjudicando a las clases minoritarias. Esto suena bastante razonable: si a una red neuronal le damos 990 de fotos de gatitos y sólo 10 de perros, no podemos pretender que logre diferenciar una clase de otra. Lo más probable que la red se limite a responder siempre “tu foto es un gato” puesto que así tuvo un acierto del 99% en su fase de entrenamiento.

Estrategias para el manejo de Datos Desbalanceados:

Tenemos diversas estrategias para tratar de mejorar la situación.

1. Ajuste de Parámetros del modelo: Consiste en ajustar parámetros ó métricas del propio algoritmo para intentar equilibrar a la clase minoritaria penalizando a la clase mayoritaria durante el entrenamiento. Ejemplos con ajuste de peso en árboles, también en logistic regression tenemos el parámetro class_weight= “balanced” que utilizaremos en este ejemplo. No todos los algoritmos tienen estas posibilidades. En redes neuronales por ejemplo podríamos ajustar la métrica de Loss para que penalice a las clases mayoritarias.

2. Modificar el Dataset: podemos eliminar muestras de la clase mayoritaria para reducirlo e intentar equilibrar la situación. Tiene como “peligroso” que podemos prescindir de muestras importantes, que brindan información y por lo tanto empeorar el modelo. Entonces para seleccionar qué muestras eliminar, deberíamos seguir algún criterio. También podríamos agregar nuevas filas con los mismos valores de las clases minoritarias, por ejemplo, cuadriplicar nuestras 492 filas. Pero esto no sirve demasiado y podemos llevar al modelo a caer en overfitting.

3. Muestras artificiales: podemos intentar crear muestras sintéticas (no idénticas) utilizando diversos algoritmos que intentan seguir la tendencia del grupo minoritario. Según el método, podemos mejorar los resultados. Lo peligroso de crear muestras sintéticas es que podemos alterar la distribución “natural” de esa clase y confundir al modelo en su clasificación

4. Balanced Ensemble Methods: Utiliza las ventajas de hacer ensamble de métodos, es decir, entrenar diversos modelos y entre todos obtener el resultado final (por ejemplo “votando”) pero se asegura de tomar muestras de entrenamiento equilibradas. Apliquemos estas técnicas de a una a nuestro código y veamos los resultados. PERO… antes de empezar, ejecutaremos el modelo de Regresión Logística “desequilibrado”, para tener un “baseline”, es decir unas métricas contra las cuales podremos comparar y ver si mejoramos