Introducci?n a la Clasificaci?n Parte II

May 4, 2017

1 Introducción al Aprendizaje Supervisado - Clasificación (Parte II)

- Naïve Bayes.
- Logistic Regression.
- Support Vector Machines.
- Ensemble Methods.
- Random Forests.
- Enfoques para problemas comunes.
- Conclusiones.

1.1 5to año - Ingeniería en Sistemas de Información

1.1.1 Facultad Regional Villa María

1.1.2 Introducción

- Considerando las bases teóricas vistas la pasada clase, el objetivo de esta clase es ver y analizar diversos métodos de clasificación, con el propósito de comprender varios de los modelos que forman el estado del arte en el aprendizaje supervisado.

1.1.3 Naïve Bayes Classifier

Retomando el Teorema de Bayes, dada una observación (una fila de X) que asumimos **IID** $x_1, x_2, ..., x_n$,

$$P(y \mid x_1, x_2, ..., x_n) = \frac{P(y)P(x_1, x_2, ..., x_n \mid y)}{P(x_1, x_2, ..., x_n)}$$

... Naïve Bayes Classifier es un clasificador que intenta aproximar $P(y \mid x_1, x_2, ..., x_n)$ tomando la muy simplista (*naïve*) asunción de que los predictores son independientes entre sí. Analíticamente,

$$P(x_i \mid y, x_1, ..., x_{i-1}, x_{i+1}, ..., x_n) = P(x_i \mid y)$$

Recordemos que dados A, B, si A es independiente de B entonces $P(A \cap B) = P(A)P(B)$. Entonces la primera ecuación se puede reescribir como

$$P(y \mid x_1, x_2, ..., x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, x_2, ..., x_n)}$$

Considerando que $P(x_1, x_2, ..., x_n) = \sum_{c=1}^{C} P(y_c)P(x_1, x_2, ..., x_n \mid y_c)$ es constante y muy compleja de tratar (**¿por qué?**), se deduce que

$$P(y \mid x_1, x_2, ..., x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

Por lo tanto,

$$\hat{y} = \arg\max_{y} P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

- Pese a la simplista asunción tomada, Naïve Bayes es uno de los mejores y más rápidos clasificadores, especialmente en lo referido a procesamiento de texto.
- Por otra parte, no es considerado un buen estimador, por lo que las probabilidades estimadas (por medio de *predict_proba*) no deben tomarse muy seriamente. Tampoco resulta ideal para datasets con muchos features numéricos.

Tutorial recomendado: Working with Text Data.
Libro recomendado para procesamiento de datos: Dipanjan Sarkar - Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from your Data (2016)

1.1.4 Logistic Regression

Una pregunta común es ¿por qué no utilizar los métodos de regresión para predecir las clases?

- La dificultad es que la mayoría de los métodos de regresión trabajan asumiendo un orden sobre las salidas. Como las salidas no están ordenadas, no es posible utilizar la diferencia entre ellas para entrenar iterativamente el modelo.

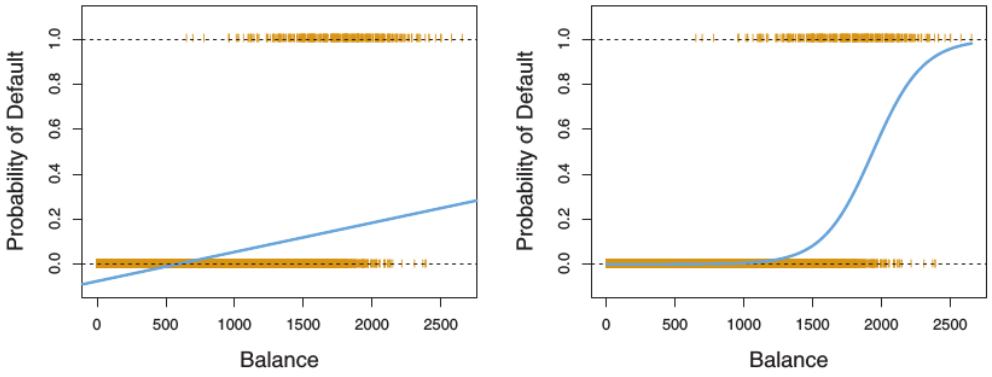
- Supongamos que queremos predecir la condición clínica de un paciente que llega a emergencias en base a sus síntomas. Su condición clínica puede ser {Aflicción cardíaca, Sobredosis, Reacción alérgica}. Podríamos intentar hacer una regresión de las salidas como variables cuantitativas de la siguiente forma

0 - Aflicción cardíaca
1 - Sobredosis
2 - Reacción alérgica

- Un problema es que estamos asumiendo un orden de las salidas, en donde asumimos sin fundamentos 1) que una SD está entre una AC y una RA y 2) que la diferencia entre una AC y una SD es la misma que la diferencia entre SD y RA.

- Este problema podría solventarse si limitamos las clases a dos. En ese sentido, supongamos que limitamos la cantidad de clases a AC y SD; una predicción de 0.23 se asociaría con una AC, mientras que una predicción de 0.9 se asociaría con una SD. Aquí el otro problema: supongamos que usamos un método de regresión lineal, ¡el valor de predicción iría hacia el infinito, pudiendo incluir probabilidades negativas o > 1!

- No obstante, es posible solventar esto aplicándole un "aplanamiento" a las salidas para que pasen de $[-\infty, \infty]$ a probabilidades [0,1].



Logistic Regression

- Uno de los métodos clásicos que realizan esto recibe el nombre de regresión logística (**logistic regression**). El mismo propone modelar $P(y \mid X)$ directamente (es un método discriminatorio) a través de la función logística. Para el caso de dos clases 0 y 1:

$$P(y = 1 \mid X = X_0) = \frac{\exp(\beta_0 + \beta^T X_0)}{1 + \exp(\beta_0 + \beta^T X_0)}$$

  donde β son los coeficientes de regresión, cumpliendo el mismo rol que los coeficientes vistos en la regresión lineal. (Notar que $\exp(x)$ equivale a e^x).

Fuente: Figura 4.2 de Hastie et. al. 2013

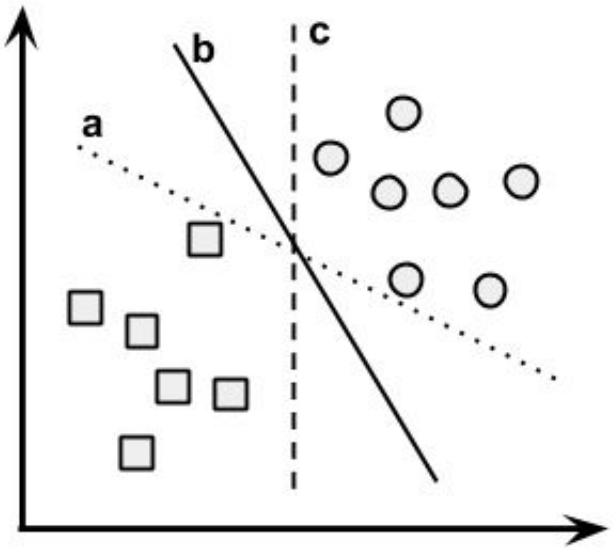
- Los coeficientes β son desconocidos. Para estimarlos se utiliza el método de *Maximum Likelihood*. Recordemos que en estadística, la likelihood (verosimilitud) de un modelo es la probabilidad de los parámetros del mismo dados los datos. Así, la función de likelihood está dada por $\mathbb{L}(\theta \mid X) = P(x_1, x_2, ..., x_n \mid \theta) = P(x_1 \mid \theta) \times P(x_2 \mid \theta) \times ... \times P(x_n \mid \theta) = \prod_{i=1}^{n} P(x_i \mid \theta)$.
- El objetivo del método Maximum Likelihood es encontrar los parámetros θ para los cuales se maximiza la verosimilitud de los datos (en otras palabras, bajo qué parámetros es más probable que los datos hayan sido generados). Este método también puede utilizarse para verificar si dadas unas muestras que por ejemplo se asumen bajo una distribución normal, cuáles son los parámetros μ y σ de la misma.
- Dados los datos IID $x_1, x_2, ..., x_n$, el máximo likelihood está dado por

$$\theta_{ML} = \arg\max_{\theta \in \Theta} \hat{\preccurlyeq}(\theta \mid x_1, x_2, ..., x_n)$$

- Logistic Regression es uno de los clásicos para problemas de clasificación binaria. Para clasificaciones multiclase, se utiliza el esquema One vs Rest.

1.1.5 Support Vector Machines (SVM)

- Una SVM puede ser imaginada como una superficie que define un límite entre varios puntos de datos, los cuales representan ejemplos de distintas clases.
- El objetivo de una SVM es crear un límite, llamado hiperplano, que separe las particiones de datos de la forma más homogénea y con mayor distancia posible.



SVM

- Es uno de los mejores métodos de clasificación, habiendo explotado en popularidad en los últimos años. Puede ser adaptado para casi cualquier problema de aprendizaje; su enorme flexibilidad hace que sea un excelente método empleado en campos como reconocimiento de patrones en una imágenes, procesamiento de texto y detección de eventos muy raros.

Fuente: https://pradeepadhokshaja.blogspot.com.ar/2016/06/optical-character-recognition-using.html

- En un espacio d-dimensional, un hiperplano se define como un sub-espacio de dimensión $(d-1)$. Ejemplo: en tres dimensiones, un hiperplano equivale al plano tal como fue estudiado en Geometría Analítica. Recordemos que un plano es el análogo en dos dimensiones a un punto ($d = 0$) y a una línea ($d = 1$).
- En dos dimensiones, un hiperplano se define por la ecuación

$$\beta_0 + \beta_1 x + \beta_2 y = 0$$

Dado un punto (x^*, y^*) que no satisface la anterior ecuación sino que

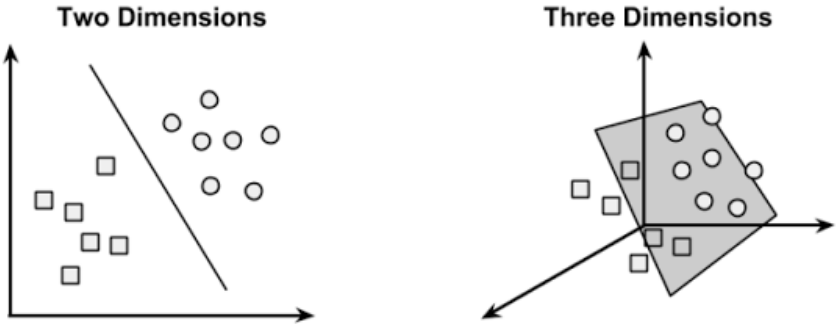
$$\beta_0 + \beta_1 x^* + \beta_2 y^* > 0$$

decimos que (x^*, y^*) se encuentra hacia uno de los lados del plano. Análogamente, si

$$\beta_0 + \beta_1 x^* + \beta_2 y^* < 0$$

diremos que (x^*, y^*) se encuentra hacia el otro lado del plano (si esto no queda claro a simple vista, imaginar el ejemplo de una recta. Se muestran ejemplos en la figura).

Fuente: https://pradeepadhokshaja.blogspot.com.ar/2016/06/optical-character-recognition-using.html



- Extendiendo esta noción a un espacio d-dimensional, y considerando que queremos clasificar cada observación en el conjunto binario de clases $C = \{-1, 1\}$, de forma que

  $$\beta_0 + \beta_1 x_1^* + ... + \beta_d x_d^* > 0$$

  si la observación $y_i = 1$, y

  $$\beta_0 + \beta_1 x_1^* + ... + \beta_d x_d^* < 0$$

  si la observación $y_i = -1$

- Suponiendo que existe un hiperplano capaz de separar perfectamente las observaciones de acuerdo a cada clase, el mismo consituye un clasificador que naturalmente separa las clases clasificadas como -1 de las clasificadas con 1, de forma que

  $$y_i(\beta_0 + \beta_1 X_{i1}^* + ... + \beta_d X_{id}^*) > 0$$

  dada una matriz X de datos de entrada.

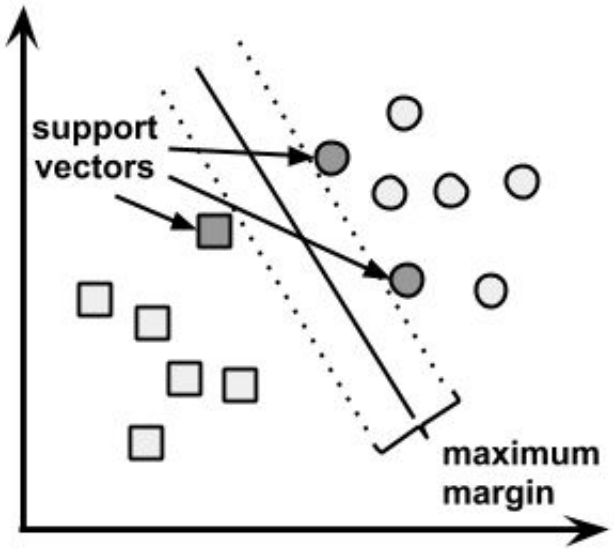
- Esencialmente, lo que hace una SVM es encontrar aquel hiperplano que maximiza el margen que separa las clases (**maximum margin hyperplane, MMH**), puesto la misma es la que se espera minimice el error de clasificación. Aquellos puntos ubicados en tales márgenes se denominan **support vectors**.

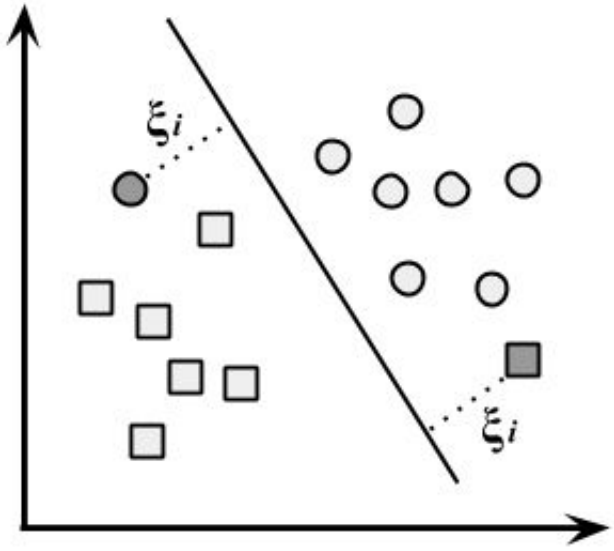
Fuente: https://pradeepadhokshaja.blogspot.com.ar/2016/06/optical-character-recognition-using.html

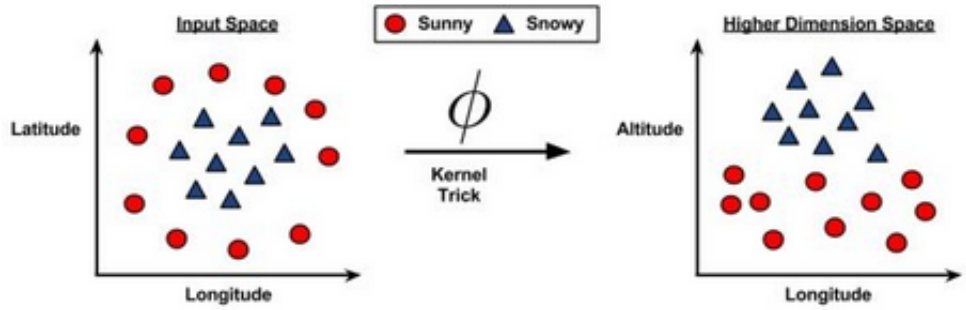
- Un problema se origina cuando los datos con los que contamos no son linealmente separables como en el caso de la figura anterior.
- En tal caso, dos enfoques suelen usarse. En el primero, el algoritmo utiliza emplea un margen flexible denotado por ξ, además de una función de costo C que es aplicada para los casos en los cuales un punto no pertenece correctamente a su hiperplano. El objetivo de SVM pasa ahora a ser la minimización de la función de costo C.

Fuente: https://pradeepadhokshaja.blogspot.com.ar/2016/06/optical-character-recognition-using.html

- El segundo enfoque, por otra parte, consiste en utilizar un operador (kernel) no lineal en lugar de utilizar el producto punto, de modo de poder generar un MMH en otro espacio.


support
vectors
maximum
margin


ξ_i
ξ_i


Input Space
Sunny
Snowy
Higher Dimension Space
Latitude
Longitude
ϕ
Kernel
Trick
Altitude
Longitude

```
action(move(A,B)) , goal(on(C,D))
on(C,D) ?
+--yes: [0]
+--no:  action(move(C,D)) ?
        +--yes: [1]
        +--no:  action(move(D,B)) ?
                +--yes: [0.9]
                +--no:  [0.81]
```

Regla de decisión

1.1.6 Decision Tree

- Un árbol de decisión es un modelo predictivo que intenta explicar los datos (X, y) como conjuntos de **reglas de decisión**. Recibe ese nombre porque puede representarse como una estructura de árbol.
- Se compone de dos tipos de nodos: los internos y las hojas. Los nodos internos definen reglas de decisión que consultan si una determinada condición es satisfecha. Cada nodo no hoja amplía en 1 la profundidad del árbol.
- Las hojas, por su parte, tienen un valor de predicción (para el caso de los árboles de regresión) o una clase (para los árboles de clasificación), dependiendo del problema que estén resolviendo.
- Técnica con altos rendimientos e interpretable, pero **muy propensa al overfitting**: la profundidad elegida del árbol es muy importante; es deseable "podar" el árbol al reducir su profundidad para evitar que el overfitting llegue a límites muy altos.

Fuente: New York Times

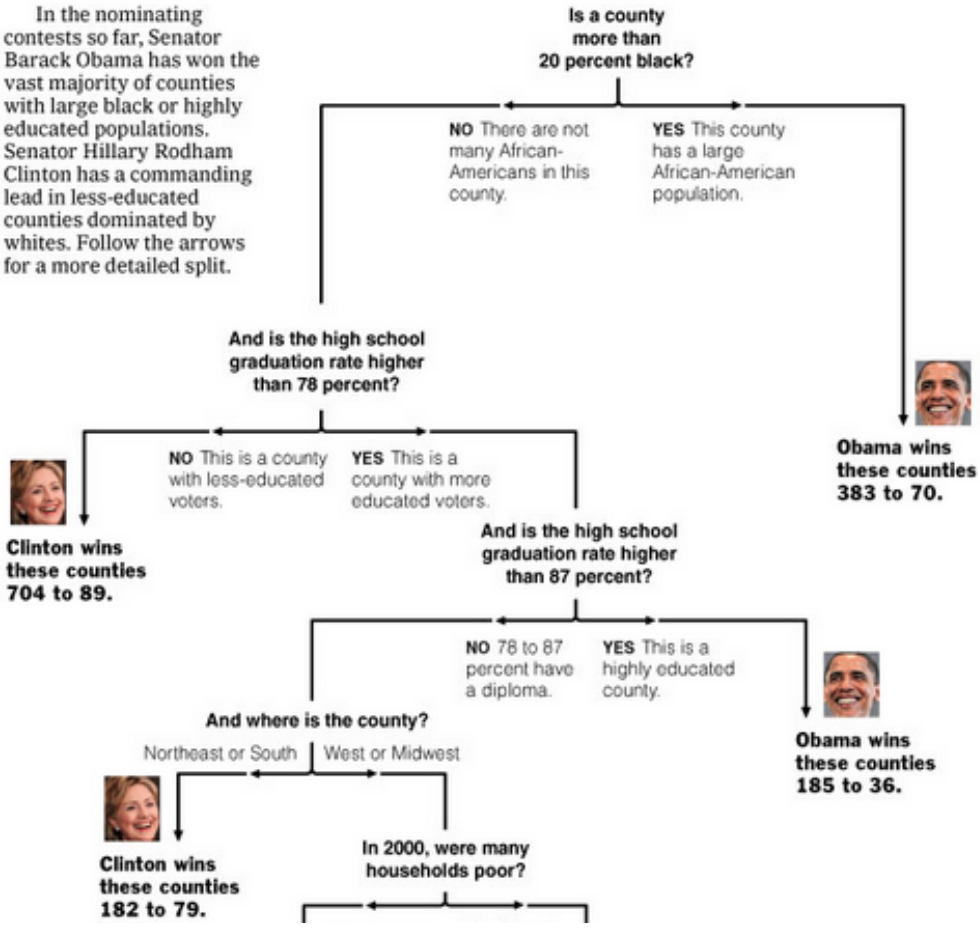
1.1.7 Ensemble Learning

- Los métodos de Ensemble Learning combinan varios modelos para resolver un problema de predicción. Cada uno de esos modelos aprenden y realizan predicciones independientemente, para luego combinar las mismas de forma tal de generar una única predicción igual o mejor que cualquier predicción realizada por un único modelo.
- Se conocen como modelos débiles, puesto que necesitan de modelos específicos para poder combinar sus predicciones.
- Un método muy conocido de ensemble es **Bootstrap Aggregating** (conocido como **bagging**).

Fuente: Udacity Course - Machine Learning for Trading (por Georgia Tech)

1.1.8 Random Forests

- Random Forest es un método de ensemble learning que agrega múltiples árboles de decisión.
- Es una mejora del bagging, que es un método que hace bootstrapping sobre varios modelos y promedia sus salidas para obtener la media (para problemas de regresión) o la clase seleccionada por votación (para problemas de clasificación).



Árbol de decisión

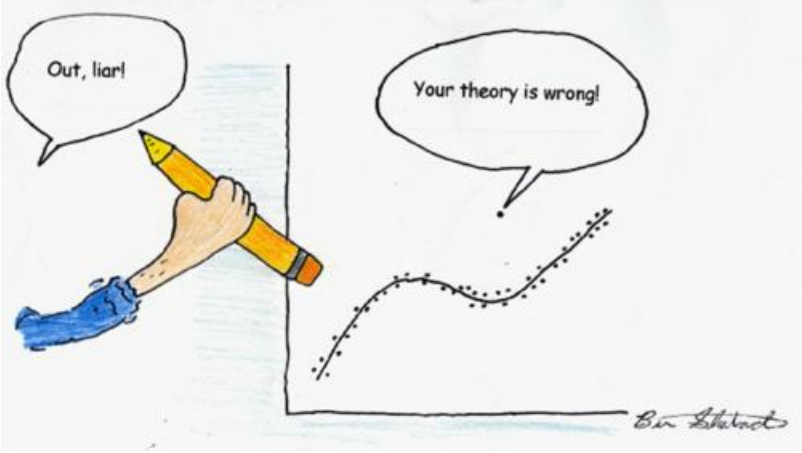
- Por medio de bootstrapping sobre cada una de las divisiones del árbol, los Random Forest crean automáticamente una alta cantidad de árboles de decisión sobre los mismos, con el objetivo de encontrar árboles de decisión que obteniendo la salida tomando la media de todas las predicciones de los árboles creados para problemas de regresión, o bien eligiendo la clase de salida mediante votación para problemas de clasificación.
- La gran mayoría de tales árboles generados automáticamente arrojan pésimas predicciones; no obstante las mismas se cancelan entre sí dando lugar a aquellos árboles que mejor se ajustan a los datos.

Ventajas * Tiene rendimientos a nivel del estado del arte, al igual que las SVM.
Desventajas * Es un método muy propenso al overfitting. * Al usar los árboles de decisión de esta manera, se pierde bastante de su interpretabilidad.

1.1.9 Enfoques para problemas comunes en Aprendizaje Supervisado

Outliers

- Un *outlier* es un punto que presenta una anomalía con respecto a nuestros demás datos. En la regresión, se considera también como outliers a aquellos puntos muy específicos para los cuales nuestra predicción $\hat{y}_i$ se encuentra muy lejos del valor real y_i.



Fuente: https://statsland.wordpress.com/2012/09/24/outliers-are-they-good-or-bad/

- Tales puntos suelen ser particularmente molestos, ya que no podemos explicar por qué la predicción está tan lejos, y al ser muy baja su cantidad no afectan demasiado el error global, por lo que cambiar el modelo sólo por ellos no tiene mucho sentido.
- El enfoque más común (pero muchas veces incorrecto) es asumir que fueron producto de un error en la toma de datos y eliminarlo.
- De alguna forma los outliers tienen que ser considerados (como mínimo tener el registro de que ocurrieron); pues a menudo suelen significar que existe un feature que no fue considerado en la toma o generación de datos.

Predicción multi-label

- Para algunos datasets, las observaciones están etiquetadas con más de una salida.

- El enfoque más común para estos casos es utilizar un predictor (regresor o clasificador) por cada uno de los labels. En el caso de la clasificación, este enfoque consiste en utilizar un clasificador binario OneVsRest para cada una de las clases.

Dataset con información faltante

- Existen casos donde los datasets no contienen información para todos sus features. Para estos casos suelen tomarse dos enfoques. Uno es eliminar las observaciones afectadas del dataset.
- Otro enfoque consiste en utilizar un predictor (ej Random Forest) para estimar, en base a las demás observaciones que contienen valor en el feature, cuál es el valor que podría tener ese feature.

Curse of Dimensionality

- *The Curse of Dimensionality* (Bellman, 1957) se refiere al problema donde, a medida que crece linealmente la cantidad de dimensiones de nuestros datos, la complejidad inherente de procesarlas crece a la vez en un orden exponencial.
- En ML, esto tiene dos consecuencias principales. La primera es que a medida que aumentan los features, se necesitan cada vez más datos para tener una muestra representativa de los mismos que abarque una parte significativa de las combinaciones de todos los features.
- La segunda consecuencia es que, al existir tantas combinaciones de los features, pasa a haber una enorme cantidad de regiones distintas en la función que intentamos aproximar, por lo que muchos métodos no pueden capturar la forma de una función tan compleja.
- Una forma de mitigarlo la vamos a ver en la clase siguiente un método llamado ***Principal Components Analysis*** (PCA), que nos ayuda a reducir la dimensionalidad de nuestro dataset.

1.1.10 Conclusiones de la Introducción al Aprendizaje Supervisado

- El aprendizaje supervisado permite predecir salidas de una función desconocida $f(X)$ al tomarla como una caja negra para entradas X no observadas, dado un entrenamiento previo con $(X, f(X))$ conocidos.
- Sus técnicas permiten obtener un gran tasa de aciertos con métodos de variada complejidad para un rango muy importante de problemas, en campos diversos como el reconocimiento de imágenes, robótica, procesamiento de texto, entre otros.
- En las clases hasta aquí se mostró una introducción al aprendizaje supervisado, mostrando cuáles son sus principales características, modelos y cómo evaluarlos.
- Debido a que el campo es muy amplio, muchos modelos han quedado fuera del alcance de estas clases; no obstante confiamos que al conocer las bases y al haber implementado varios, el aprendizaje de nuevas técnicas no será dificultoso puesto que la gran mayoría se como una extensión de lo visto en estas clases.

Problemas éticos El aprendizaje automático no exime al humano de ejercer mecanismos de control tanto a nivel micro, de quien diseña o implementa un modelo, como a nivel macro, de la sociedad. Si tal control no es efectivo, estas técnicas pueden amplificar problemas sociales preexistentes dando lugar a efectos colaterales muy serios, incluyendo

- Catástrofes.
- Daños contra las minorías y distintos miembros de la sociedad (discriminación, xenofobia, etc.).
- Incremento en la desigualdad de la distribución de la riqueza.

Debe existir un fuerte debate sobre el rol de los sistemas automáticos en nuestro día a día para poder minimizar los costos sociales que los mismos pueden generar.

De Weapons of Mass Destruction The Book Website:

> But as Cathy O'Neil reveals in this urgent and necessary book, the opposite is true. The models being used today are **opaque**, **unregulated**, and **uncontestable**, even when they're wrong. Most troubling, they reinforce discrimination: If a poor student can't get a loan because a lending model deems him too risky (by virtue of his zip code), he's then cut off from the kind of education that could pull him out of poverty, and a vicious spiral ensues. Models are propping up the lucky and punishing the downtrodden, creating a "toxic cocktail for democracy." Welcome to the dark side of Big Data. Tracing the arc of a person's life, O'Neil exposes the black box models that shape our future, both as individuals and as a society. These "weapons of math destruction" score teachers and students, sort résumés, grant (or deny) loans, evaluate workers, target voters, set parole, and monitor our health.

Algunas lecturas: * Artículo: Data science instrumenting social media for advertising is responsible for todays politics * Artículo: Artificial Intelligence's White Guy Problem * Artículo: The Dark Secret at the Hearth of AI * Web: Future of Life Institute

1.1.11 Ejercicios

1. Utilizar, al igual que en los anteriores prácticos, una semilla de random_state igual al número de orden en Entregas TPs.

2. Elegir dos clasificadores y realizar alguna predicción en un dataset a su elección tal como lo venimos haciendo hasta ahora, mostrando para cada clasificador la tasa de aciertos junto con sus respectivos precision y recall (ayuda: utilizar *classification_report* de *sklearn.metrics* para no tener que calcular ambos a mano). Se alienta a que busquen nuevos datasets y cómo implementar clasificadores, por lo que al menos uno de los clasificadores seleccionados debe haber sido uno de los vistos en esta clase o algún otro no visto anteriormente.

3. Explicar el paso a paso de la implementación, y de la comparación de sus errores. Explicar por qué creen que un clasificador se desempeñó mejor o similarmente que el otro.

Fecha de entrega: **17/05/2017**.

Nota: la resolución de los ejercicios es **individual**; en el caso de que dos ejercicios enviados contengan un código igual o muy similar (sin considerar los comentarios), se los considerará a ambos como desaprobados. La reutilización del código de los notebooks está permitida (por ejemplo para confeccionar gráficos).

Algunas fuentes de datasets:
https://www.kaggle.com/datasets
http://scikit-learn.org/stable/datasets/index.html
https://archive.ics.uci.edu/ml/datasets.html

In [12]:
```
# Fuente de datasets en sklearn
from sklearn import datasets

# Descomentar para obtener info sobre los datasets de sklearn.
# Incluye algunos datasets que pueden cargarse.
#help(datasets)
```